University of Alberta

Developing and Evaluating Methods for Mitigating Sample Selection Bias in Machine Learning

by

Lourdes Pelayo Ramirez

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software Engineering and Intelligent Systems

Electrical and Computer Engineering

Fall 2011

Edmonton, Alberta

Abstract

The imbalanced learning problem occurs in a large number of economic and health domains of great importance; consequently, it has drawn a significant amount of interest from academia, industry, and government funding agencies. Several researchers have used stratification to alleviate this problem; however, it is not clear what stratification strategy is in general more effective: under-sampling, over-sampling or the combination of both. Our first topic evaluates the contribution of stratification strategies in the software defect prediction area. We study the statistical contribution of stratification in the new Mozilla dataset, a new large-scale software defect prediction dataset which includes both object-oriented metrics and a count of defects per module. Our second topic responds to the debate about the contribution of over-sampling, under-sampling and the combination of both with the employment of a full-factorial design experiment using the Analysis of Variance (ANOVA) over six software defect prediction datasets. We extend our research to develop a stratification method to mitigate sample selection bias in function approximation problems. The sample selection bias is present when the training and test instances are drawn from a different distribution, with the imbalance dataset problem considered a particular case of sample selection bias. We extend the well-known SMOTE over-sampling technique to continuous-valued response variables. Our new algorithm proves to be a valuable algorithm helping to increase the performance on function approximation problems and effectively reducing the impact of sample selection bias.

Acknowledgments

I would like to express my gratitude to my Supervisor Dr. Scott Dick, for his support and guidance through this research. Without his knowledge and assistance this study would not have been possible.

I am also very grateful to the rest of my supervisory committee for all their valuable suggestions to enhance this research.

I am deeply thankful to my family for their support and encouragement during the process of all my study.

Table of Contents

List of Tables

List of Figures

List of Symbols, Nomenclature or Abbreviations

ANOVA	Analysis of Variance
AUC	Area Under the Curve
CBO	Coupling Between Object classes
CPU	Central Processing Unit
df	Degrees of freedom
DIT	Depth of Inheritance Tree
KDD	Knowledge Discovery in Databases
k-NN	k-Nearest Neighbors
LCOM	Lack of Cohesion of Methods
MAR	Missing At Random
MCAR	Missing Completely At Random
NMAR.	Not Missing At Random
NOC	Number of Children
OO	Object Oriented
OLS	Ordinary Least Squares
OSS	One-Sided Selection
PACE	Projection Adjustment by Contribution Estimation
PCA	Principal Component Analysis
RBF	Radial Basis Function
RFC	Response for a Class
ROC	Receiver Operating Characteristic

SDP	Software Defect Prediction
SMOTE	Synthetic Minority Oversampling Technique
SSB	Sample Selection Bias
SVM	Support Vector Machine
SVR	Support Vector Regression
SOM	Self-Organizing Map
SMO	Sequential Minimal Optimization
WMC	Weighted Methods per Class

Chapter 1. Introduction

In recent years, the imbalanced learning problem (inductive learning when the phenomena of interest are a small minority of observations) has drawn a significant amount of interest from academia, industry, and government funding agencies. Imbalanced learning occurs in a large number of economic and health domains of great importance. In situations like credit card fraud detection, cancer detection, and software defect prediction the imbalanced dataset problem can have a great economic impact. As there is usually a higher cost in misclassifying the interesting class, doing so can easily cost billions of dollars every year across the world. For example, software defect prediction refers to forecasting which modules in a software system have latent defects, and how many. It is one strategy to reduce the incidence of software failures, which currently cost the U.S. economy alone more than $78 billion per year [1]. In the health domain, specifically in cancer studies, The National Institutes of Health in USA estimates there were close to 1.5 million new cases of cancer in 2009; the overall estimated costs of cancer in 2008 was $228.1 billion, including direct medical costs, indirect morbidity costs and indirect mortality costs [2]. A 2006 identity theft survey report sponsored by the Federal Trade Commission (FTC) estimated that approximately 8.3 million U.S. adults were victims of ID theft in 2005, of which 3.2 million reported that the misuse of their information was limited to the misuse of one or more of their existing credit card accounts in 2005. The estimate of total losses from ID theft in the 2006 survey was USD $15.6 billion. [3]. Defective software modules, cancer patients, and fraudulent credit card transactions are all minorities in their respective populations, usually by orders of

magnitude. Machine learning systems have been applied to these problems as well as many others; however, these highly skewed domains are a particular challenge for machine learners.

The imbalanced dataset problem is present where the number of samples in the training dataset for one class outnumbers the samples for the other class. The fundamental issue is the potential for imbalanced data to significantly compromise the performance of most standard learning algorithms. Traditional machine learners aim to maximize the overall accuracy, which will lead to a bias towards the majority class under imbalanced situations. However, with imbalanced datasets, the interesting class is often the minority class. In practical applications, the ratio of the small to the large classes can be drastic such as 1 to 100, 1 to 1,000, or 1 to 10,000 (and sometimes even more) (See, for example, [4-6]). Many machine learning algorithms assume training and test instances are drawn identically from a common distribution; however, in reality this assumption is no longer valid. This problem is referred to in the statistical literature as Sample Selection Bias (SSB), with the imbalanced dataset problem considered a particular case of SSB. Our research considers three problems in learning from imbalanced data or under sample selection bias.

Our first problem is software defect prediction (SDP), which plays an important role in the estimation of software reliability. Machine learning techniques have been applied as an attempt to estimate software reliability through the use of defect prediction datasets. However, any machine-learning approach is dependent on the quality of the data used to

train the algorithm, and this is a serious shortcoming of current software reliability modeling; publicly-available defect prediction datasets do not reflect current industry approaches to software design. Currently, the majority of publicly-available datasets are quite small (i.e. relatively few modules in the program), and collect only a small set of procedure-oriented metrics, meaning they do not reflect modern software development processes. At the present time we are aware of only one single publicly-available dataset, the Mozilla dataset [7], that simultaneously collects object-oriented metrics and defect measures over a large number of modules, and no exploration of this dataset has been reported in the literature. Existing studies of imbalanced data in software defect prediction are thus focused on examples that do not reflect modern industrial-quality software systems.

Our second problem responds to a debate in the machine learning community about the use of stratification. Different researchers have used stratification to alleviate the problem of imbalanced datasets; stratification reduces the number of samples of the majority class (under-sampling) and/or adds more samples to the minority class (over-sampling) in order to create a more balanced dataset. One question that arises at this point is what stratification technique would be more useful: over-sampling, under-sampling or a combination of both. More specifically, the question is what stratification alternatives work best across many datasets; it is expected that every technique works well for some datasets. Although there is a considerable amount of research on this topic, the existing work only uses a trial-and-error design; we argue, however, that the importance of under-sampling, over-sampling, or their *interaction* across many datasets (i.e. a generalizable

statement of the applicability of these different approaches) can only be determined through a factorial experimental design. Thus, at this time there is no *guidance* for the analyst on which stratification approaches would be best pursued.

Thirdly, research in learning from imbalanced datasets has focused on classification problems, i.e. those problems requiring the prediction of a categorical response variable. In the domain of software test management (for example), identifying which modules contain failures and which are failure free is an essential first step to allocating testing resources. However, software test management can only be *optimized* if we have a prediction of how many failures are present in each faulty module (i.e. we predict the value of a numeric response variable), which is a function approximation problem. Developing techniques to mitigate sample selection bias in function approximation problems is an important but under-explored problem for the machine learning community. Heckman [8] developed a procedure for correcting sample selection bias; unfortunately, that technique is limited to linear regression models. While a few authors have attempted to correct SSB in function approximation for more general machine learning techniques, these are all generalizations of cost-sensitive classification; there is at this time no stratification-based approach to correct for SSB.

The goal of our research is thus to characterize current stratification techniques on new datasets and in a generalizable framework; and to extend them to function approximation problems. Our research will be organized as three inter-related investigations. First, we address the need for modern SDP datasets by performing a case study of the new Mozilla

dataset collected in [9]. This is a new large-scale software defect prediction dataset which includes both object-oriented metrics and a count of defects per module. Due to the skewness of this dataset, function-approximation algorithms are ineffective in modeling this dataset; we thus convert it in a binary classification problem and employ stratification-based resampling. We obtain a classifier that exhibits greater sensitivity to the minority (fault-prone) class, even though this is only 5% of the original dataset. We address the question of over-sampling vs. under-sampling stratification techniques by employing full-factorial design experiments using the Analysis of Variance (ANOVA). We make use of ANOVA across a set of six software defect-prediction datasets employing the common tenfold cross-validation design in each cell with five levels for under-sampling and five levels for over-sampling. These experiments will determine the contribution of over-sampling, under-sampling and/or their interaction towards improving defect predictions in the datasets. Based on our results in Chapter 3, we choose the well-known SMOTE oversampling technique [10] for our experiments. Finally, we develop a stratification method to mitigate sample selection bias in function approximation problems. We propose an extension of the SMOTE over-sampling technique [10] to continuous-valued response variables. SMOTE often improves classification performance in a great proportion of imbalanced datasets, making it a good candidate for function approximators. Due to the fact that we no longer have an *a priori* division of the dataset into individual classes, this algorithm necessarily involves several steps. We first make use of the Expectation Maximization clustering algorithm to identify regions of the feature space with greater or lesser sampling densities. We select regions for over-sampling using a criterion that combines the local density and variance of the

dependent variable in that region. As with the original SMOTE algorithm, we then select a random point along the line joining two nearest neighbors in an under-represented region of the input space, and create a synthetic example at that point, using linear interpolation to determine values for both the predictor and response variables.

While the focus of our research in Chapters 3 and 4 is software defect prediction, we believe the results can be generalized beyond this domain. Chapter 3 is a case study of stratification in a classification problem, which has been well-studied. However, this is only a classification problem because the *original* function approximation problem (predicting the number of defects in each Mozilla module) could not be adequately solved. It is our belief that this is due to the combination of SSB and heteroskedasticity in the dataset; we thus reduced heteroskedasticity by converting the problem to a classification task, and reduced SSB by stratification. Thus, this case study illustrates one path to improve predictions in function-approximation problems under the combination of heteroskedasticity and SSB. Chapter 4 is a response to the general assertion by Weiss and Provost that over-sampling is generally unhelpful in imbalanced datasets [11]. This is a sweeping assertion with practical implications in many fields. While our study in Chapter 4 is confined to software defect prediction datasets, the methodology can be replicated in other fields. Finally, our new Numeric SMOTE technique is tested on a variety of benchmark datasets, and our results are thus not limited to a single class of function approximation problems.

The following chapters are arranged as follows: Chapter 2 presents a background section; in Chapter 3 we present the analysis of the Mozilla dataset; in Chapter 4 we present our study of the contribution of over-sampling and under-sampling; while in Chapter 5 we present our new stratification method for function approximation problems. Finally, Chapter 6 offers conclusions and discusses future work in this research.

Chapter 2. Background

2.1 Imbalanced Datasets

The imbalanced dataset problem undermines machine learning algorithms that do not explicitly consider differential error costs or class distributions. The basic design of most machine learning algorithms is to develop a representation of a problem (a decision tree, a neural network, a set of classification rules, etc.), and to measure the quality of that representation using a global performance measure. This internal representation will then be updated iteratively, in order to optimize the performance measure (a node in a decision tree is split on some attribute; node weights in a neural network are changed; rules are added or deleted from a ruleset) [12]. The difficulty is that the "quality" of the representation can only be measured using the data available to the system (the "training" data), and thus the performance measure is naturally biased towards the majority class. Data points belonging to the minority class will tend to have less influence on the iterative updates to the internal problem representation than those belonging to the majority class. Over the course of many training iterations, the problem representation thus loses some or all of its ability to accurately model the minority class, in favor of better modeling of the majority class [13].

Japkowicz and Stephen [14] explore the relationship between concept complexity, size of the training set and class imbalance level to identify class imbalance situations that are

most damaging for a standard classifier that expects balanced class distributions. They also compare over, under-sampling and cost-modifying. According to their results under-sampling is the least effective; the authors show that under-sampling is not useful in schemes where the majority class contains no irrelevant data and the classification is affected by removing significant data. Although the authors reported that cost-based methods perform better than random resampling strategies, they do not have the flexibility offered by resampling approaches such as generating novel examples or resampling different parts of the space differently [15].

Significant work has been done when dealing with the imbalanced dataset problem in classification problems. Monard and Batista analyze in [16] the problem of skewed class distributions and discuss the inadequacy of accuracy and error rate to measure the performance of learning systems when a class imbalance problem is present. They also survey a few methods of cost-sensitive classification and stratification to solve the problem of learning with imbalanced data sets. Visa and Ralescu [17] propose a fuzzy based classifier for imbalanced datasets that reduces sensitivity to the class imbalance by considering a relative frequency to the class size. In the same paper the sensitivity of the fuzzy classifier to the imbalance in the presence of overlap, when some data points may appear as (valid) examples in both classes, is investigated. The results show that in fact the overlap affects the classifier more than the imbalance. Later, the same authors [18] investigate some of the recent progress in the field of learning of imbalanced data including issues and methods in dealing with imbalance and their performance evaluation.

Jo and Japkowicz investigate in [15] whether the class imbalance in a dataset is responsible for the degradation of the performance of standard classifiers, or is the the problem of small disjuncts that affect their performance; small disjuncts are those which have a low coverage of the number of training examples they correctly classify. The authors show that the small disjunct problem is more responsible for this decrease in accuracy due to the presence of rare cases in domains with a small training set size and high complexity, compared to the class imbalance problem. They investigate solutions that address the class imbalance and small disjunct problem either individually or simultaneously. They propose a cluster-based over-sampling method that deal with class imbalance and small disjuncts simultaneously. The cluster-based over-sampling works by first clustering the minority and majority classes separately. Majority clusters, except for the largest one, are over-sampled to match the size of the largest majority cluster. The minority clusters are over-sampled to match the size of one majority cluster divided by the number of minority clusters. They compare their method against no resampling, cost modification, pruning the small disjuncts, over-sampling, under-sampling, and An's over-sampling method [19]. Their results show the resampling methods work generally better than cost modification and pruning of small disjuncts and their cluster based over-sampling works better than any other method.

Guo et al. [20] describe an approach that combines boosting, an ensemble-based learning algorithm, and generation of synthetic examples to improve the predictive power of classifiers against imbalanced data sets consisting of two classes. Joshi et al. make a comparison in [21] of three boosting algorithms CSB I , CSB2 [22] and AdaCost [23],

and propose two enhanced versions of AdaBoost [24] and SLIPPER [25] to perform better in situations involving rare classes. Weiss [26] discusses the role that rare classes and rare cases play in data mining, their problems such as improper evaluation metrics, lack of data, data fragmentation, inappropriate inductive bias and noise; as well as methods for addressing these problems including sampling, cost-sensitive learning and boosting.

Another approach to learning from imbalanced data sets, proposed by Provost and Fawcett [27], is to build a hybrid classifier that uses ROC analysis for comparison of classifier performance that is robust to imprecise class distributions and misclassification costs. An et al. [28] apply a rule induction technique to deal with imbalanced datasets. Li et al. [29] develop a new associative classification approach, called CRB (Classification based on Rare attributes and Bi-confidence), for imbalanced datasets. Akbani et al. [30] look at the issue of imbalanced data specifically in the context of support vector machines. Veropoulos et al. [31] also focus on support vector machines and considers the issue of data imbalance while discussing the balance between sensitivity and specificity. Another approach in SVMs is found on [32] where the authors proposal is to adjust the class boundary by modifying the kernel matrix, according to the imbalanced data distribution to help the SVM to improve the class prediction accuracy. In the context of linear program boosting, the paper by Leskovec and Shawe-Taylor [33] considers the implications of imbalanced data, and tests this on a text classification problem. Raskutti [34] also uses SVMs when dealing with a heavily unbalanced dataset. Li and others analyze in [35] the factors behind the significant performance drop of SVM when

learning from imbalanced data sets and propose a re-sampling method. They resample the imbalanced data by using variable self-organizing map (SOM) clustering and then prune the training set by means of a sample-pruning algorithm based on KNN. The pre-processing step improves the generalization ability of SVM on imbalanced datasets.

Other researchers have developed more specialized approaches for countering class imbalance. One approach is to redesign a specific algorithm to make it more responsive to the minority class; examples of this approach include [36, 37]. Another approach, often used in text mining, is feature selection. In a number of domains (usually characterized by a huge number of attributes and a limited number of examples, e.g. text mining, microarray data, etc.) the ordinary process of feature selection has been experimentally shown to also improve classifier performance in the presence of class imbalance [38].

A somewhat different approach termed "generative oversampling" is introduced in [39]. In this technique, synthetic examples are created by first learning the parameters of a given probability distribution, and then sampling from that distribution. The polynomial curve-fitting approach [40] proceeds by first finding a least-squares fit for a polynomial curve of degree n to the minority class examples in feature space. Then, synthetic examples are created at an even spacing along this curve. The degree n of the polynomial has to be determined empirically.

2.1.1 Stratification

Stratification is one of the two main methods for dealing with imbalanced datasets. It is used to modify the distribution of the classes in a dataset in order to homogenize them or make them more responsive towards the class of interest. Stratification differs from bagging or boosting. Bagging (bootstrap aggregation) is an ensemble meta-algorithm to improve classification and/or regression. A dataset is sampled with replacement to create new training datasets of mainly the same size, having a class distribution roughly the same as in the original dataset. A collection of training datasets is created, and then an ensemble of classifiers is trained, one to each new dataset. Classification of new examples is performed by a vote of the ensemble [41]. Boosting also creates an ensemble of classifiers, but does so sequentially. In each successive iteration, the probability of selecting an example is increased if that example is incorrectly classified [42].

The two basic forms of stratification are under-sampling and over-sampling. In under-sampling only a subset of majority class examples is randomly selected (without replacement) for inclusion in the modified dataset. Over-sampling refers to increasing the number of minority-class examples, by duplicating some or all of the minority-class examples and to adding these duplicates to the modified dataset. Several authors have expanded these two basic forms of stratification. Wilson's Editing [43] is an undersampling technique in which each example is classified using a 3-nearest-neighbor classifier based on all other examples in the dataset. Those that are misclassified are removed. One-sided selection (OSS) also uses the k-NN classification approach (this time

a 1-nearest-neighbor classifier), but then also uses Tomek links to remove boundary cases [44]. An approach introduced in [45] splits the majority class into clusters of equal size to the minority class, then unites each cluster with a copy of the minority class. An ensemble of classifiers is then built, with each classifier trained on exactly one of the united clusters. In [46] a genetic algorithm-based data sampling method named Evolutionary Sampling is introduced. This is an undersampling technique in which the majority-class samples to be included are determined by a genetic algorithm, using the area under the ROC curve (AUC) and the geometric mean accuracy per class as fitness functions. On two SDP datasets, this approach was shown to be consistently superior to other stratification techniques (with the exclusion of random undersampling). Zhang et al. [47] study the effects of under-sampling on the k-nearest neighbor kNN algorithm, using five different methods of choosing majority training examples: random selection, selection of near-miss examples and selection of most distant examples. For the near-miss examples the authors use 3 different ways: NearMiss-1 selects negative examples that are close to some of the positive examples; NearMiss-2 selects negative examples that are close to all positive examples; and NearMiss-3 selects a given number of the closest negative examples for each positive example. The authors find that among these 5 selection methods, the random and NearMiss-2 methods perform the best.

SMOTE [10] is a proposed over-sampling approach in which the minority class is over-sampled by creating "synthetic" examples rather than by over-sampling with replacement. To create a synthetic example, SMOTE searches for the nearest neighbors (having the same class label) of a minority-class example. A new synthetic example is

then created at a random point on the line connecting the two genuine samples (this assumes that each dimension of feature space forms a ratio scale), and the class label for the new example is the minority class; this is illustrated in Figure 2.1. Different synthetic examples are based on different neighbor pairs. The SMOTE algorithm is shown in Figure 2.2

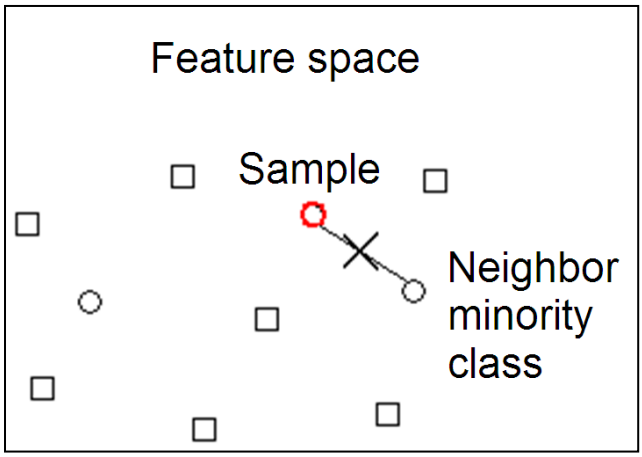

Figure 2.1 Generation of a Synthetic Example in SMOTE

```
Input: Oversampling rate overRate, Undersampling rate underRate
Output: Resampled dataset
1. MinSamples: number of minority class samples
2. MajSamples: number of majority class samples
3. numAttribs: number of attributes
4. MinArray[][]: minority class samples array
5. MajArray[][]: majority class samples array
6. newindex= 1
7. Synthetic[][]: array for synthetic samples
8. neighborArray[][]: array for nearest neighbors
9. if (overRate< 100 && 0<= overRate) (*If overRate is less than 100 only a random
smaller part of the training set will be used to generate synthetic examples*)
10.     T = (overRate/100) * MinSamples
11.     Oversample(overRate, T)      (*Call Oversample method*)
12. else  (*if overRate >= 100*)
13.     T=MinSamples
14.     Oversample(overRate, T)      (*Call Oversample method*)
15. endif
16. if(underRate > 0)
17.     Undersample(underRate)
18. endif
19. Oversample(overRate, T)
20. for i = 1 to T
21.     Compute T nearest neighbors and save them in neighborArray
22.     for j=1 to neigborArray.size
23.         for attr =1 to numAttribs
24.             dist=neighborArray[j][attr]-MinArray[i][attr]
25.             rand=random()
26.             Synthetic[newIndex][attr]=MinArray[i][attr]+rand*dist
27.         endfor
28.         newIndex++
29.     endfor
30. endfor   (*end of Oversample*)
31. Undersample(underRate)
32. num=MajSamples
33. N=100-underRate
34. samples=num*(N/100)
35. while(num>samples)
36.     rand=random();
37.     index=index*rand
38.     removeElement(MajArray[index])
39.     num--
40. endwhile (*end of Undersample*)
```

Figure 2.2 SMOTE Algorithm Derived from [10]

There are several extensions of the basic SMOTE algorithm. Batista, Prati and Monard [48] propose the SMOTE + Tomek and SMOTE + ENN algorithms. According to their results, oversampling methods perform better in contrast with undersampling methods, with the area under the ROC curve (AUC) as their performance measure. SMOTEBoost is a combination of the SMOTE algorithm and the boosting procedure developed in [49]. In [50] the authors create a new variant of SMOTE called Borderline-SMOTE, in which only the minority examples near the class boundary are over-sampled. Chawla et al. [51] also propose a hybrid method combining random undersampling and SMOTE; a fivefold cross-validation design is used to select the undersampling and oversampling rates.

One of the most recent variations of SMOTE was published by García et al. in 2008 [52] using surrounding neighbors to generate synthetic examples. The authors propose the use of an alternative neighborhood concept, namely surrounding neighborhood that takes both the proximity and the spatial distribution of the examples into account. Alternative concepts of neighborhood were proposed as a way to consider the geometrical location as well as the proximity of the points. Surrounding neighborhoods consider proximity and symmetry so as to define the general concept of neighborhood. Thus they try to search for neighbors close enough (in the basic distance sense), but also in terms of their spatial distribution with respect to a given object. For the Nearest Centroid Neighborhood (NCN) concept [53], let *p* be a point whose k neighbors should be found from a set of points $X = \{x_1, \ldots, x_n\}$. These *k* neighbors are such that they are as near *p* as possible and their centroid is also as close to *p* as possible. To satisfy the two conditions, Chaudhuri

proposes an iterative procedure where the first neighbor of p is its nearest neighbor, say r. The second neighbor q is such that the centroid of r and q is nearest to p. This definition gives rise to a kind of neighborhood in which both closeness and spatial distribution of neighbors are taken into account because of the centroid criterion. However, the iterative procedure clearly does not minimize the distance to the centroid because it gives precedence to the individual distances instead. The SMOTE – SN algorithm is given in Figure 2.3.

Input: Oversampling rate *overRate*, Undersampling rate *underRate*
Output: Resampled dataset
1. *MinSamples*: number of minority class samples
2. *MajSamples*: number of majority class samples
3. *numAttribs:* number of attributes
4. *MinArray[][]*: minority class samples array
5. *MajArray[][]*: majority class samples array
6. *newindex*= 0
7. *Synthetic[][]*: array for synthetic samples
8. *neighborArray[][]*: array for nearest neighbors
9. if (*overRate*< 100 && 0<= *overRate*) (*If *overRate* is less than 100 only a random smaller part of the training set will be used to generate synthetic examples*)
10. *T* = (*overRate*/100) * *MinSamples*
11. *NN=1*
11. Oversample(*NN, T*) (*Call Oversample method*)
12. else (*if overRate >= 100*)
13. T=*MinSamples*
14. *NN*=*overRate*/100
15. Oversample(*NN, T*) (*Call Oversample method*)
16. endif
17. if(*underRate* > 0)
18. Undersample(*underRate*)
19. endif
20. Oversample(*NN, T*)
21. for *p* = 1 to *T*
22. Find the nearest neighbor *r* for *p* and save it in *neighborArray*
23. for *i*=1 to *NN*-1
24. Find neighbor *q* for which the centroid of *q* is nearest to *p* in terms of Euclidean distance
25. Add *q* to *neighborArray*
26. endfor
27. for j=1 to *neighborArray.size*
28. for *attr* =1 to *numAttribs*
29. *dist=neighborArray[j][attr]-MinArray[i][attr]*
30. *rand*=random()
31. *Synthetic[newIndex][attr]=MinArray[i][attr]+rand*dist*
32. endfor
33. *newIndex++*
34. endfor
35. endfor (*end of Oversample*)

Figure 2.3. SMOTE - SN Algorithm derived from [52], part 1

```
36. Undersample(underRate)
37. num=MajSamples
38. N=100-underRate
39. samples=num*(N/100)
40. while(num>samples)
41.     rand=random();
42.     index=index*rand
43.     removeElement(MajArray[index])
44.     num--
45. endwhile (*end of Undersample*)
```

Figure 2.3. SMOTE - SN Algorithm derived from [52], part 2

2.1.2 Cost-sensitive classification

Numerous algorithms in machine learning assume that all errors in classification are equally important. However, when the minority class samples are the interesting ones the cost of misclassification errors is often higher when the errors are located in the minority class in contrast with the majority class [11]. Mistaking a minority class sample for a majority class sample is more costly to the end users of the classification. Cost-sensitivity helps to improve the classification of the class of interest (in this study the minority class). It can be implemented by modifying the weight on training instances according to the total cost assigned to each class or predicting the class with minimum expected misclassification cost.

Elkan [54] proves a theorem showing how to change the proportion of negative examples in a training set in order to make an optimal cost sensitive classification; however, it has little effect when using Bayesian and decision tree learning algorithms. Turney [55] introduces an algorithm for cost sensitive classification called ICET that is a hybrid of

genetic algorithms and decision trees. Ting and Zheng [56] explore boosting techniques for cost-sensitive classification focused on decision tree learning. The main idea is to re-weight the original training dataset using the history of misclassifications and the associated costs of misclassification for each class. In subsequent iterations, the classifier will focus on costly and hard to classify instances. Domingos proposed a general method for cost-sensitive classification called MetaCost in [57]. It is based on re-labeling training examples with their minimal-cost classes based on their class probabilities instead of their "optimal" class. The learning algorithm is then trained using this relabeled training set. As a meta-classifier, the MetaCost algorithm introduces cost-sensitive classification to arbitrary classifiers, without the need to explicitly formulate cost-sensitivity in the performance measures of those algorithms. For this reason, MetaCost is widely used when cost-sensitive classification is required. A cost-sensitive classification algorithm for graph-structured data (e.g. hyperlinks structures, sequences in natural language processing) is presented in [58], while Weiss and Tian extend the concept of "optimal cost" to incorporate data acquisition costs and the cost of modeling (i.e. CPU time) [59]. Ling and Sheng [60] present an overview of cost-sensitive learning methods. The study in [61] presents two empirical methods that deal with class imbalance using both resampling and cost-sensitive learning. The first method combines and compares several sampling techniques such as Tomek links, random under-sampling, random over-sampling, and SMOTE with cost sensitive learning using SVM as the base classifier. The second method proposes using cost-sensitive learning by optimizing the cost ratio (cost matrix) locally. Their experimental results show that the first method can reduce the misclassification costs, and the second method can improve the classifier performance.

2.2 Sample Selection Bias

Traditionally, it is assumed that training and test sets are sampled from the same distribution. However, in reality this assumption is often violated. Some of the examples sampled into the training data may actually be infrequent or not be represented in the testing data, and vice versa. When we face this problem, we are dealing with sample selection bias (SSB). The imbalanced dataset problem and covariate shift are particular cases of SSB. Although in several papers [62-64] covariate shift is investigated in the same context as SSB; this is further discussed in Section 2.2.1. SSB occurs when the training and test sets are not drawn from the same distribution. In the case of covariate shift, only the input distribution changes, whereas the conditional distribution of the outputs given the inputs remains unchanged [65]. In this research we deal with SSB and covariate shift in the same context in function approximation problems as done in [62, 66], while we refer to imbalanced datasets when working on classification on binary class problems.

The SSB problem is present in many real-world situations such as marketing solicitation, fraud detection, drug testing, loan approval, school enrollment, etc., where traditionally, many machine learning algorithms assume training and test instances are drawn identically from a common distribution. Although, in reality this assumption is no longer valid. Consider the drug testing case where a model is built to predict if a particular drug is effective for the entire population of individuals, but the available training data is typically a sample from previous hospital trials. The individuals participating in these

studies are representative of the patients at the hospital but not of the entire population. An inductive model constructed from a biased training set may not be as accurate on unbiased testing data.

James Heckman won the Nobel prize in Economics for his model for correcting sample selection bias [8]. The key insight in Heckman's work is that if we can estimate the probability that an observation is selected in the sample, we can use this probability estimate to correct the model. Before going into detail on Heckman's solution to SSB, we introduce the following example. Consider estimating a wage offer equation (a formula that attempts to determine what wage should be offered to a prospective hire) for people of working age. The aim of estimating wage equations often is to assign wage rates for those who are not currently working, so that they can be used in labor force models in economic studies. The wage offer equation can be represented by:

$$Y_{1i} = x_i\beta + \varepsilon_i \qquad (2.1)$$

where Y_{1i} is the wage, $\boldsymbol{x}_i$ are the observed variables relating to the i'th person's productivity, β is a parameter to be estimated and ε_i is an error term. By definition, this equation is supposed to represent all people of working age, whether or not a person is actually working at the time of the survey. However, we are not observing the equation for the population as a whole. Those in employment will tend to have higher wages than those not in the labor force would have, because their productivity attributes are higher. (This is a tremendous over-simplification, as is the linear form of Eq. 2.1; however, these simplifications are common in economics. For this example, the salient fact is that the productivity attributes of working people cannot be presumed to be representative of the

attributes of non-working people, but only the attributes of working people can be observed.) One thus expects that the results will tend to be biased. We could have for example two groups of people: industrious and lazy. Industrious people get higher wages and have jobs, lazy people do not. (We might also have lucky and unlucky groups, with perhaps statistically identical productivity attributes; however, this cannot be assumed.) The sample selection bias in problems like these needs to be corrected in order to obtain valid estimates.

Heckman's model is a parametric two-step estimator using maximum likelihood to correct SSB. It requires distributional assumptions regarding the disturbances and Heckman makes use of the typical assumptions for maximum likelihood. Specifically, ε_i and v_i, $i = 1 \ldots n$, are independently and identically distributed with mean zero and covariance matrix Σ where

$$\Sigma = \begin{pmatrix} \sigma_{\varepsilon}^2 & \sigma_{\varepsilon v} \\ \sigma_{\varepsilon v} & \sigma_{v}^2 \end{pmatrix} \tag{2.2}$$

The conventional sample selection model has the form:

$$Y_{1i} = x_i^{'}\beta + \varepsilon_i \quad (i = 1 \ldots n) \tag{2.3}$$

$$Y_{2i} = z_i^{'}\gamma + v_i \quad (i = 1 \ldots n) \tag{2.4}$$

where x and z are vectors of independent variables and β and γ are vectors of parameters to be estimated. ε and v are the error terms. Eq (2.3), known as the primary model, is the equation of primary interest and Eq (2.4), known as the participant model or selection model, is the reduced form for the latent variable capturing sample selection. The model

for Y_{1i} is the one we are interested in, but Y_{1i} is only observable if $Y_{2i} > 0$. Thus the observed dependent variable Y is

$$Y = Y_{1i} * d_i \tag{2.5}$$

$$d_i = 1 \; if \; Y_{2i} > 0, \; d_i = otherwise \tag{2.6}$$

The latent variable Y_{2i} itself is not observable, but only its sign: We only know that $Y_{2i} > 0$ if Y is observable, and $Y_{2i} < 0$ if not. d_i is an indicator function reflecting whether the primary dependent variable Y_{1i} is observed. Since the sign of Y_{2i} does not change if we multiply it by a positive constant, it can be assumed that the variance of v_i is equal to 1 without loss of generality.

We can use least squares to estimate the parameter β in our primary equation of interest (Eq (2.3)) over the n subsample corresponding to d_i=1

$$Y_{1i} = x_i\beta + \varepsilon_i; \; i = 1...n \tag{2.7}$$

however, least squares leads to biased estimates of β because the conditional expectation $E[\varepsilon_i \mid z_i, d_i = 1] \neq 0$. The general strategy proposed by Heckman is to overcome this through the inclusion of a correction term that accounts for $E[\varepsilon_i \mid z_i, d_i = 1]$. To employ this approach, take the conditional expectation of Eq (2.7) to get

$$E[Y_{1i} \mid z_i, d_i = 1] = x_i\beta + E[\varepsilon_i \mid z_i, d_i = 1]; i = 1...n \tag{2.8}$$

We use the assumption that ε_i and v_i, $i = 1 \ldots n$, are i.d.d.$(0,\sum)$ and the formula for the conditional expectation of a truncated random variable, note that

$$E[Y_{1i} \mid z_i, d_i = 1] = \frac{\sigma_{\varepsilon v}}{\sigma_v^2}\left\{\frac{\phi(z_i\gamma)}{\Phi(z_i\gamma)}\right\} \tag{2.9}$$

where ϕ (.) and Φ (.) denote the probability density and cumulative distribution functions of the standard normal distribution (N(0,1)). The term in curly brackets is known as the inverse Mills ratio [67]. Heckman's two-step estimator is based on

$$Y_{1i} = x_i\beta + \frac{\sigma_{\varepsilon v}}{\sigma_v^2}\left\{\frac{\phi(z_i\gamma)}{\Phi(z_i\gamma)}\right\} + \varepsilon_i \qquad (2.10)$$

The two-step estimator suggested by Heckman is to first estimate γ over all n observations by maximum likelihood Probit. A probit model is a variant of logistic regression used to analyze binomial response variables. After estimating γ, we use the estimation on the inverse Mills ratio and proceed to estimate β by ordinary least squares.

We apply this SSB correction to the estimation of the wage offering case. From Eq (2.3) Y_{1i} is the wage, $\boldsymbol{x}_i$ are the observed variables relating to the i-th person's productivity and ε_i is an error term. Y is observed only for workers, i.e. only people in work receive a wage. In Eq (2.4), Y_{2i} is the difference between the wage and the reservation wage. The reservation wage is the minimum wage at which the i-th individual is prepared to work. If the wage is below that they choose not to work. Our indicator variable for employment is d_i as in Eq (2.6). Following the procedure, we take the conditional expectation of Eq (2.7) upon the individual working and the values of x. Finally we use the inverse Mill's ratio and estimate the values of γ and β using maximum likelihood and ordinary least squares as previously indicated. Eq(2.10) gives us a more accurate wage estimation equation.

2.2.1 Machine learning approaches

Zadrozny presents in [68] a categorization of the different types of SSB. Using the same terminology, assume that standard classifier learning algorithms draw examples (x,y,s) from a distribution D with domain X x Y x S where X is the feature space, Y is the label space and S is a binary space. The variable s is a random variable which controls the selection of examples and takes values 1 or 0 ("sampled" or "not sampled"). Then, the predicate "s = 1" denotes that a labeled example ($\boldsymbol{x}$, y) (where $\boldsymbol{x}$ = (x_1, x_2, ... , x_k) is a feature vector of k feature values, and y is a class label) is sampled into the training set T from the universe of all labeled examples D, and "s = 0" denotes that ($\boldsymbol{x}$, y) is not selected. Additionally, the joint probability P($\boldsymbol{x}$, y) = P(y|$\boldsymbol{x}$) · P($\boldsymbol{x}$) denotes the unbiased or true probability to observe example ($\boldsymbol{x}$, y) in the universe of examples D. Many inductive learning algorithms model P($\boldsymbol{x}$, y) either via approximating conditional probability P(y|$\boldsymbol{x}$) or both P(y|$\boldsymbol{x}$) and P($\boldsymbol{x}$) in the same time. With these definitions, SSB is best described by a conditional dependency of s = 1 (or the predicate that "($\boldsymbol{x}$, y) is sampled" to be true) on both $\boldsymbol{x}$ and y, or formally, P(s = 1|$\boldsymbol{x}$, y).

There can be four types to consider regarding the dependence of s on the example (x,y).

1. No sample selection bias or P(s = 1|$\boldsymbol{x}$, y) = P(s = 1). In other words, the examples constitute a random sample from D, the selection process is independent from both feature vector $\boldsymbol{x}$ and class label y.

2. Feature bias or $P(s = 1|\boldsymbol{x}, y) = P(s = 1|\boldsymbol{x})$. The selection process is conditionally independent from y given $\boldsymbol{x}$, the selected sample is biased but the biasedness only depends on the feature vector x.
3. Class bias or $P(s = 1|\boldsymbol{x}, y) = P(s = 1|y)$. The selection process is conditionally independent from $\boldsymbol{x}$ given y, the selected sample is biased but the biasedness depends only on the label y.
4. Complete bias or $P(s = 1|\boldsymbol{x}, y)$ The selection process is dependent on both $\boldsymbol{x}$ and y. The selected sample is biased and we cannot hope to learn a mapping from features to labels using the selected sample, unless we have access to an additional feature vector $\boldsymbol{x}_s$ that controls the selection (that is, $P(s|\boldsymbol{x}_s, \boldsymbol{x}, y) = P(s|\boldsymbol{x}_s)$) for all the examples (even for the ones that have $s = 0$).

The first case where no sample selection bias is present or $P(s|\boldsymbol{x},y)=P(s)$ is very difficult to guarantee. For example, if the training set T does not include every possible feature vector $\boldsymbol{x}$, there is at least a dependency of s=1 on $\boldsymbol{x}$. It is rather difficult to exhaustively collect every feature vector for any real-world applications. No sample selection bias is likely an unrealistic assumption in practice. In the second case, the selection bias s is dependent on the feature vector $\boldsymbol{x}$ and it is conditionally independent of the true class label y given $\boldsymbol{x}$. This type of sample selection naturally exists. This type of bias occurs for example, in medical data where a treatment is not given at random, but the patients receive the treatment according to their symptoms which are contained in the example description (i.e. the $\boldsymbol{x}$ values). Therefore, the population that received the treatment in the past is usually not a random sample of the population that could potentially receive the

treatment in the future. In the third case, the selection bias is dependent only on the true class label *y*, and is independent from the feature vector $\boldsymbol{x}$. This occurs when there is a correlation between the label value and the likelihood of appearance in the database. Software defect prediction datasets, to cite an example, have a much higher number of instances with a small or null number of defects compared to the instances where the number of errors found on the modules are higher. In other words, the probability of finding instances with one or more errors is much lower than the probability of finding modules error-free. These probabilities are far from being equal as function approximators would expect. In the final case above, there is no assumption about any independence of *s* given $\boldsymbol{x}$ and *y*, and both the example description and its label influence whether the example will be chosen into the training set.

The third type of SSB (or class bias type) is the closest to the imbalanced dataset problem. If *s* is independent of $\boldsymbol{x}$ given *y* (that is $P(s|\boldsymbol{x}, y) = P(s|y)$), the selected sample is biased but the biasedness depends only on the label *y*. This corresponds to a change in the prior probabilities of the labels. This type of bias has been studied in the machine learning literature with approaches such as [69]. Elkan [54] and Japkowicz and Stephen [14] investigate the case of training data that is only biased with respect to the class ratio, this can be seen as sample selection bias where the selection only depends on *y*.

Fan and Davidson [70] analyze different types of SSB. Class bias (Case 3) describes the situation that the training data has different prior class probability distribution from the true probability distribution. To simulate different class biases, the authors partition the

training dataset into multiple class bins where each class bin keeps all the examples belonging to that particular class. To generate a biased training set, they first randomly choose a prior probability distribution for each class, fix the training set size to be the same as the original unbiased training set, then sample examples from each class bin with replacement. For a two class problem, if the randomly generated prior class probability is (0.2, 0.8) and the training set size is 100, then 20 examples will be sampled from the positive class and 80 examples will be sampled from the negative class. They proceed to analyze the effect of class bias on a model's accuracy, under different minority class percentages in the datasets. In the machine learning community this problem has been addressed as imbalanced datasets and has been approached mainly with cost-sensitive and stratification techniques.

2.3 Software Reliability

2.3.1 Software metrics and reliability

The software metrics of interest in this research are formally referred to as product metrics, as they concern the actual source code. Examples are McCabe's complexity [71] or Halstead's "software science" metrics [72]. Project metrics such as the number of tests written and passed, number of open and closed problem reports (among others), are beyond the scope of this research, as are size estimation metrics such as function or object points [73]. Product metrics express some aspect of a module (size, complexity, cohesion, coupling with other modules, etc.) as a numeric value. In the software metrics

literature, each new metric is argued on philosophical and measure-theoretic grounds, and then validated on actual project data. This validation usually takes the form of a correlation or regression analysis between metric values and a defect attribute (number of defects or occurrence of defects in a module); see for instance [74-76]. Literally hundreds of product metrics have been created in the last 30 years, and there has been significant debate on the merits of these measures. One key criticism is that the measure-theoretic underpinnings of these product metrics are often shaky; it is not always clear whether the numeric values of a metric properly belong to a ratio, interval, or even an ordinal measurement scale! Furthermore, it is remarkably difficult to even attempt to determine these basic facts; notions such as "complexity," for example, are based on subjective perceptions that are not amenable to formal mathematical analysis [77].

Software product metrics also tend to be strongly correlated with each other as well as with the number of defects, a phenomenon known as multicollinearity. This is a well-known problem in the software metrics literature [76, 78], which undermines the regression models commonly used in defect prediction; these models assume independence between the predictor attributes. Principal Components Analysis (PCA) is commonly employed to overcome this problem; in addition to reducing the dimensionality of a dataset, the resulting principal components are statistically independent from one another, eliminating multicollinearity [79, 80]. Heteroskedasticity [81] (non-uniform variance) is also a well-known problem in the software metrics literature. Furthermore, metric values on any project tend to be sharply skewed towards small values; the reason is that most modules in a software project are small and simple.

The large, complex modules are rarer; however, these are where most defects tend to be found.

Beginning in 1994, researchers have turned their attention to developing metrics specifically for object oriented (OO) programming, in order to reflect the great differences between procedure-oriented and object-oriented code. The first and best-known OO metrics are a suite of six developed in [82] and known as the CK metrics: Weighted Methods per Class (WMC), Depth of Inheritance Tree (DIT), Number of Children (NOC), Response For a Class (RFC), Coupling Between Object Classes (CBO), and Lack of Cohesion of Methods (LCOM). Also in 1994, Lorenz and Kidd [83] proposed additional OO metrics that concentrate on inheritance and message passing in a class. Numerous other metrics have also been proposed; extensive reviews and comparisons can be found in [74, 84-86]. Recent research [87] also concentrates on collecting run-time metrics, owing to the greater usage of dynamic libraries in OO systems.

2.3.2. Software defect prediction

We formulate software defect prediction (SDP) as a binary classification problem, in which the values of a set of software product metrics for a given code module are used to classify it as being either at low risk of containing a defect that leads to a failure (class 0), or at high risk of containing such a defect (class 1).

Although research in SDP consistently shows a reasonably strong linear correlation between software metrics and the number of defects per module, there is still no parametric model that has been developed to accurately forecast the number of faults in a software module. Ultimately, this stems from a lack of any theory that relates software metric values to reliability in a module. Each software project has a unique combination of intellectual difficulty, size, development team composition, and third-party components, among other factors. The result is that there is no "critical value" for any software metric that definitively indicates when a module will be of lower quality. In the end, the lack of a theory reduces parametric approaches for SDP to a trial-and-error search through an infinite space of model forms (e.g. the models in [88]). While it is possible to find a model that predicts faults reasonably well for one project, this model is usually not transferable to a different project. For such reasons nonparametric techniques, especially machine learning and computational intelligence, have been employed for software defect prediction.

2.4 Statistical methods

2.4.1 Principal Component Analysis

Principal Component Analysis (PCA) was invented by Karl Pearson in 1901 [89]. It is a statistical method to extract relevant information; it reduces data dimensionality by performing a covariance analysis between variables. PCA uses transformation to identify a set of uncorrelated variables called principal components. From this transformation the first principal component will have the highest variability in the data and each following component will have the highest variance while being uncorrelated to the previous

components. Principal components are linear combinations of the p random variables X_1, X_2, …, X_p. PCA depends mainly in on the covariate matrix Σ (or the correlation matrix ρ) of X_1, X_2, …, X_p. Let the random vector $\boldsymbol{X}' = [X_1, X_2, \ldots, X_p]$ have the covariance matrix Σ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$. Consider the linear combinations

$$\begin{aligned} Y_1 &= a_1'X = a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p \\ Y_2 &= a_2'X = a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p \\ &\vdots \\ Y_p &= a_p'X = a_{p1}X_1 + a_{p2}X_2 + \cdots + a_{pp}X_p \end{aligned} \tag{2.11}$$

Then,

$$Var(Y_i) = \boldsymbol{a_i'\Sigma a_i} \quad i = 1, \ldots, p \tag{2.12}$$

$$Cov(Y_i, Y_k) = \boldsymbol{a_i'\Sigma a_k}\ i, k = 1, .., p \tag{2.13}$$

The principal components are those uncorrelated linear combinations Y_1, $Y_2, \ldots, Y_p$ whose variances are as large as possible. The first principal component is the linear combination $a_1'X$ with maximum variance $Var(a_1'X)$ subject to $a_1'a_1 = 1$. The second principal component is the linear combination $a_2'X$ that maximizes $Var(a_2'X)$ subject to $a_2'a_2 = 1$ and $Cov(a_1'X, a_2'X) = 0$. Successively, the i-th principal component is the linear combination $a_i'X$ that maximizes $Var(a_i'X)$ subject to $a_i'a_i = 1$ and $Cov(a_1'X, a_k'X) = 0$ for $k < i$.

Having Σ as the covariance matrix associated with the random vector $X' = [X_1, X_2, \ldots, X_p]$ and Σ having the eigenvalue-eigenvector pairs (λ_1, e_1), (λ_2, e_2), …, (λ_p, e_p) where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p$. Then the i-th principal component is given by

$$Y_i = \boldsymbol{e_i'X} = e_{i1}X_1 + e_{i2}X_2 + \cdots + e_{ip}X_p, \quad i = 1,2, \ldots, p \tag{2.14}$$

With these choices,

$$Var(Y_i) = \boldsymbol{e_i'\Sigma e_i} = \boldsymbol{\lambda_i} \;\; i = 1, \ldots, p \tag{2.15}$$

$$Cov(Y_i, Y_k) = \boldsymbol{e_i'\Sigma e_k} = 0, \;\; i \neq k. \tag{2.16}$$

2.4.2 Two-sample t-test

A two-sample t-test is used to test the difference between two population means. A common application is to determine whether the means are equal. We define our hypotheses as:

$$H_0 : \mu_1 = \mu_2 \tag{2.17}$$

$$H_1 : \mu_1 \neq \mu_2 \tag{2.18}$$

where H_0 is the null hypothesis and H_1 is the alternative hypothesis, μ_1 is the mean of the control samples and μ_2 is the mean of the treatment samples. The two-sample t-test for comparing sample means is:

$$t_{=} = \frac{\bar{y}_1 - \bar{y}_2}{S_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \tag{2.19}$$

where $\bar{y}_1$ and $\bar{y}_2$ are the sample means, n_1 and n_2 are the sample sizes, S_p^2 is an estimate of the common variance $\sigma_1^2 = \sigma_2^2 = \sigma^2$ computed from

$$S_p^2 = \frac{(n_1 - 1)S_1^2 + (n_2 - 1)S_2^2}{n_1 + n_2 - 2} \tag{2.20}$$

and S_1^2 and S_2^2 are the two individual sample variances. To determine whether to reject the null hypothesis, we would compare t_0 to the t distribution with $n_1 + n_2 - 2$ degrees of freedom. If $t_0 > t_{\alpha/2, n_1+n_2-2}$ where $t_{\alpha/2, n_1+n_2-2}$ is the upper $\alpha/2$ percentage point of the t distribution with $n_1 + n_2 - 2$ degrees of freedom, we would reject H_0 and conclude that

the mean samples of the control and treatment experiments differ. We can find the values of $t_{\alpha/2,n_1+n_2-2}$ in statistical books such as [90].

2.4.3 Effect size

The effect size is a statistical concept that measures the magnitude or strength of the relationship between two experiments. The greater the effect size, the greater the difference between the experiments. One method to estimate effect size for independent samples t-tests is to calculate and interpret Cohen's d statistic [91]. We calculate Cohen's d statistic with:

$$Cohen's\ d = \frac{\mu_1 - \mu_2}{s^*} \tag{2.21}$$

where μ_1 and μ_2 are the means of the samples and s* is the pooled standard deviation calculated with:

$$s^* = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{(n_1 - 1) + (n_2 - 1)}} \tag{2.22}$$

After calculating Cohen's d, we can use the descriptors from Table 2.1 to interpret the effect size.

Table 2.1 Descriptors for Interpreting Effect Size

Value	Interpretation
≤ 0.2	Small effect size
0.2 < 0.5 < 0.8	Medium effect size
≥ 0.8	Large effect size

2.4.4 Tukey honestly significant difference test

Tukey's HSD test can be used when we are interested in comparing all pairs of *a* treatment means. For a pairwise comparison of all means, the null and alternative hypotheses are:

$$H_0 : \mu_i = \mu_j \tag{2.23}$$

$$H_1 : \mu_i \neq \mu_j \tag{2.24}$$

for all i ≠ j. The null hypothesis is rejected if the absolute value of the sample difference exceeds $T_\alpha = q_\alpha(a, f)\sqrt{\frac{MS_E}{n}}$ where $q_{\alpha(a,f)}$ is the critical value of the studentized range distribution [90], α is the significance level, *a* is the number of means, *f* represents the degrees of freedom, MS_E is the mean square error from ANOVA, and *n* is the number of observations.

2.4.5 ANOVA

ANOVA provides a statistical test of whether or not the means of several groups are all equal. Analysis of variance is an important technique for analyzing the effect of categorical factors on a response. An ANOVA decomposes the variability in the response variable amongst the different factors. It may be important to determine which factors have a significant effect on the response, and/or how much of the variability in the response variable is attributable to each factor.

The variability in a set of data quantifies the scatter of the data points around the mean. To calculate a variance, first the mean is calculated, then the deviation of each point from

the mean. If the deviations are squared before summation then this sum is a useful measure of variability, which will increase the greater the scatter of the data points around the mean. This quantity is referred to as a sum of squares (SS).

The name analysis of variance is derived from a partitioning of total variability into its component parts. The total corrected sum of squares is used as a measure of overall variability in the data. This total variability in the data can be partitioned into a sum of squares of the differences between the treatment averages and the grand average, plus a sum of squares of the differences of observations within treatments from the treatment average. The difference between the observed treatment averages and the grand average is a measure of the differences between treatment means, whereas the difference of observations within a treatment from the treatment average can be due only to random error. This can be expressed as:

$$SS_{Total} = SS_{Ttreatments} + SS_{Error} \quad (2.25)$$

Every SS was calculated using a number of independent pieces of information. When looking at the deviations of data around a central grand mean, there are n − 1 independent deviations or degrees of freedom (*df*). Combining the information on SS and df, we can arrive at a measure of variability per *df*. This is equivalent to a variance, and in the context of ANOVA is called a mean square. The mean squares for the terms on the right on Eq (2.25) can be expressed as:

$$MS_{Treatments} = \frac{SS_{Treatments}}{a-1} \quad (2.26)$$

$$MS_E = \frac{SS_E}{N-a} \quad (2.27)$$

where N is the total number of observations and a is the number of treatment means.

A formal test of the hypothesis of no differences in treatment means can be performed. The null hypothesis is expressed as:

$$H_0 = \mu_1 = \mu_2 = \cdots = \mu_a \tag{2.28}$$

If the null hypothesis of no difference in means is true, the ratio

$$F_0 = \frac{SS_{Treatments}/(a-1)}{SS_E/(N-a)} = \frac{MS_{Treatments}}{MS_E} \tag{2.29}$$

is distributed as F with a-1 and N-a degrees of freedom. Eq (2.29) is the test statistic for the hypothesis of no differences in treatment means. If the expected value of $MS_{Treatments}$ of the test statistic is greater than the expected value of the denominator, we should reject H_0 on the values of the test statistic that are too large. This implies an upper-tail, one-tail critical region. Therefore, we should reject H_0 and conclude that there are differences in the treatment means if

$$F_0 > F_{\alpha,a-1,N-a} \tag{2.30}$$

where F_0 is computed from Eq (2.29). The value $F_{\alpha,a-1,N-a}$ can be found on different statistical textbooks.

In an ANOVA analysis there are three assumptions we need to consider in order to have a valid test of the hypothesis of no difference in treatment means:

- The populations from which the samples were selected, are assumed to be normally distributed.
- The populations from which the samples were selected, are assumed to have the same variance.

- The samples selected from the different populations are assumed to be random and independent of each other.

A useful procedure to check the normality assumption is to construct a normal probability plot of the residuals. If the underlying error distribution is normal, this plot will resemble a straight line. Additionally, statistical tests may be performed to check the normality assumption if there is any difficulty interpreting the probability plot. There are three common statistical tests for this purpose: Anderson-Darling, Shapiro-Wilk and Kolmogorov-Smirnov.

The Anderson-Darling test is evaluates if a given sample of data did or did not come from a given probability distribution. For a given data set and distribution, the smaller the statistic test is, means the better the distribution fits the data, The formula for the test statistic A to evaluate if data $\{Y_1<\ldots<Y_n\}$ (data must be in order) comes from a distribution with cumulative distribution function is defined as:

$$A^2 = -N - S, \tag{2.31}$$

where $S = \sum_{i=1}^{N} \frac{(2_i-1)}{N} [\ln F(Y_i) + \ln (1 - F(Y_{N+1-i}))]$. F is the cumulative distribution function of the specified distribution.

The Shapiro–Wilk test tests the null hypothesis that a sample x_1, ..., x_n came from a normally distributed population. The test statistic is:

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{2.32}$$

where $x_{(i)}$ is the ith order statistic, $\bar{x} = (x_1 + \cdots + x_n)/n$ is the sample mean and the constants a_i are given by $(a_1, \cdots, a_n) = \frac{m^T V^{-1}}{(m^T V^{-1} V^{-1} m)^{1/2}}$ where $m=(m_1, \ldots, m_n)^T$.

The Kolmogorov-Smirnov two-sample test can be used to test for normality of distribution; in this case, samples are standardized and compared with a standard normal distribution. The test compares the cumulative normal distribution function and the sample cumulative distribution function. If the distribution being tested is a good fit for a normal distribution, then these two cumulative distribution functions should be close to each other. The K-S test is based on the maximum distance between these two functions. Given a sample of N observations, we determine

$$D = \max |F(X) - S_N(X)|, \tag{2.33}$$

where $S_N(X)$ is the sample cumulative distribution function and $F(X)$ is the cumulative normal distribution function. If the value of the D statistic is larger than the K-S critical value at a given level of significance, then we reject the hypothesis that the distribution of the data is normal. Tables of K-S critical values can be found on [92].

We define the terms kurtosis and skewness as they are mentioned in this section and calculated in further chapters. The kurtosis is a measure of how broadly or narrowly distributed a population is about the mean. If the kurtosis is positive, the distribution tends to have a very high sharp peak near the center and long thin tails at both ends. If it is negative, it tends to be relatively flat in the center and to drop off rapidly to zero near to the ends. The theoretical Normal distribution would have a kurtosis of zero. Skewness is a measure of how asymmetric a probability distribution is about the mean. If we plot

the distribution of the data and the skewness is positive, the distribution tends to have a long tail to the right. When it is negative, the long tail will be to the left. The skewness of a symmetric distribution is zero. Kurtosis differs from skewness primarily in that kurtosis measures the overall tendency toward long tails at both ends; skewness measures the degree which this tendency differs at the two ends.

ANOVA is generally robust to violations of the normality assumption. A summary of relevant mathematical work on previous work [93] was: "Nonnormality has little effect on inferences about means" in fixed effect analysis of variance. The most important departures from normality are likely to be when skewness and kurtosis have nonzero values. Although, in general, only the kurtosis of the distribution is likely to have any considerable effect on a F-test. If the kurtosis is high, the resulting F will tend to be too small, and the data will appear to be less significant than they really are. The opposite result occurs if the kurtosis of the samples population is smaller than zero. In that case, the F ratio would be too large and the null hypothesis may be rejected at a higher significance level than it should. However, when the number of samples is reasonably large, neither the skewness nor the kurtosis have much effect on mean squares, making ANOVA a very robust statistical test and relatively immune to violations in the normality assumption [94].

Several statistical tests are used to the check the equality of variance. One of them is the Barlett's test. Bartlett's test is used to test the null hypothesis, H_0 that all k population variances are equal against the alternative that at least two are different. The procedure

involves computing a statistic whose sampling distribution is closely approximated by the chi-square distribution with a-1 degrees of freedom when the *a* random samples are from independent normal populations. The test statistic is

$$x_0^2 = 2.3026\ \frac{q}{c} \qquad (2.34)$$

where

$q = (N-a)log_{10}S_p^2 - \sum_{i=1}^{a}(n_i-1)log_{10}S_i^2$, $\quad c = 1 + \frac{1}{3(a-1)}(\sum_{i=1}^{a}(n_i-1)^{-1} - (N-a)^{-1})$, $\quad S_p^2 = \frac{\sum_{i=1}^{a}(n_i-1)S_i^2}{N-a}$, and S_i^2 is the sample variance of the *i*th population. The quantity q is large when the sample variances S_i^2 differ greatly and is equal to zero when all S_i^2 are equal. Therefore, we should reject H_0 on values of X_0^2 that are too large; that is, we reject H_0 only when $X_0^2 > X_{\alpha,a-1}^2$ where $X_{\alpha,a-1}^2$ is the upper α percentage point of the chi-square distribution with a-1 degrees of freedom. Barlett's test is very sensitive to the normality assumption. Consequently, when the validity of this assumption is doubtful, Barlett's test should not be used. The modified Levene test is a procedure that is robust to departure from normality. To test the hypothesis of equal variances in all treatments, the modified Levene test uses the absolute deviation of the observations y_{ij} in each treatment from the treatment median, say $\tilde{y}_i$. Denote these deviations by

$$d_{ij} = |y_{ij} - \tilde{y}_i| \begin{cases} i = 1,2,\dots,a \\ j = 1,2,\dots,n_i \end{cases}$$

The modified Levene test then evaluates whether or not the mean of these deviations are equal for all treatments. If the mean deviations are equal, the variance of the observations in all treatments will be the same. The test statistic for Levene's test is simply the usual ANOVA F-statistic for testing equality of means applied to the absolute deviations.

When the null hypotheses for the normality and/or homogenous variances assumption are rejected, transformations of the original dataset may correct these violations. To transform data, you perform a mathematical operation on each observation, then use these transformed numbers in the statistical test. The most common data transformations are square root and log transformation. The square-root transformation consists of taking the square root of each observation; when there are negative numbers, it is needed to add a constant to each number to make them all positive. The log transformation consists of taking the base-10 log or base-e log of each observation. For the back transformation raise 10 or *e* to the power of the number. If there are zeros or negative numbers, a constant should be added to each number to make them positive and non-zero.

Other methods for dealing with heteroskedasticity are available. Besides applying a transformation, one can attempt to correct for non-uniform variance using a method such as Box's correction [95]. In [96] Piepho uses a 3-step procedure to overcome this problem by adjusting the degrees of freedom (*df*) according to Box's method in a two-way ANOVA (the design was a 2-way full-factorial experiment with blocking, comparable to our design in Chapter 4).

Consider the two-way ANOVA on Table 2.2:

Table 2.2 Two-way ANOVA

Source	Sum of Squares	Degrees of freedom	Mean Square	F_0
A treatments	SS_A	a-1	$MS_A = \frac{SS_A}{a-1}$	$F_0 = \frac{MS_A}{MS_R}$
B treatments	SS_B	b-1	$MS_B = \frac{SS_B}{b-1}$	$F_0 = \frac{MS_B}{MS_R}$
Interaction	SS_{AB}	(a-1)(b-1)	$MS_{AB} = \frac{SS_{AB}}{(a-1)(b-1)}$	$F_0 = \frac{MS_{AB}}{MS_R}$
Residual	SS_R	ab(n-1)	$MS_R = \frac{SS_E}{ab(n-1)}$	
Total	SS_T	abn-1		

The procedure suggested by Greenhouse and Geisser [97] to correct heteroskedasticity involves three steps:

Step 1. Test with usual degrees of freedom

Step 2. Box's conservative test

Step 3. Test using Box's ϵ

The procedure is shown in Table 2.3

Table 2.3 Three step procedure to correct heteroskedasticity

	F-test A treatments		F-test B treatments		F-Test Interaction	
	Numerator	Denominator	Numerator	Denominator	Numerator	Denominator
Step 1.	a-1	ab(n-1)	b-1	ab(n-1)	(a-1)(b-1)	ab(n-1)
Step 2.	1	$\frac{ab(n-1)}{(a-1)}$	1	$\frac{ab(n-1)}{(b-1)}$	1	$\frac{ab(n-1)}{(a-1)(b-1)}$
Step 3.	ϵ(a-1)	ϵab(n-1)	ϵ(b-1)	ϵab(n-1)	ϵ'(a-1)(b-1)	$\frac{\epsilon' ab(n-1)}{(a-1)(b-1)}$

If no significance is attained based on the usual degrees of freedom in Step 1, the analysis ends, since the adjusted df test will also be non-significant. If Box's conservative test (Step 2) is significant, the analysis may be terminated, since the adjusted df test will also

be significant. Otherwise, conduct the F-test using Box's correction (Step 3). The correction factors ϵ and ϵ' have to be estimated from the data, the details can be found on [97].

2.4.6 Bonferroni's correction

Bonferroni's correction is a method used to cancel out the problem of multiple comparisons. The correction is based on the idea that if an experimenter is testing *n* dependent or independent hypotheses on a set of data, then one way of maintaining the family-wise error rate is to test each individual hypothesis at a statistical significance level of 1/*n* times what it would be if only one hypothesis were tested. So, if it is desired that the significance level for the whole family of tests should be (at most) α, then the Bonferroni correction would be to test each of the individual tests at a significance level of α/*n*. For example, to maintain α=0.05 in making six comparisons, use an α-level of 0.05/6 = 0.0083.

2.5 Classification Algorithms

2.5.1 C4.5 decision tree

C4.5 is an algorithm for inducing classification rules in the form of decision trees from a set of given examples developed by Ross Quinlan [98]. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. C4.5 builds decision trees from a set of training data using the concept of information entropy. Nodes in a decision tree involve testing a particular attribute. For each attribute, we find the information gain and split the tree on the

attribute which information gain is the highest. This process starts at the root and is repeated recursively until a leaf node is hit, at which point the value in the leaf constitutes the output.

The general algorithm for building a decision tree is:

1. For each attribute *attr,* find the normalized information gain from splitting on *atrr*
2. Selecte *attr_best* as the attribute with the highest normalized information gain
3. Create a decision *node* that splits the training dataset on *attr_best*
4. Recurse on the sublists obtained by splitting on *attr_best*, and add those nodes as children of *node*.

2.5.2 Radial Basis Function Networks

A radial basis function (RBF) network is an artificial neural network that uses radial basis functions as activation functions. RBF networks typically have three layers: an input layer, a hidden layer with a non-linear RBF activation function and a linear output layer. In general the form taken by an RBF function is given as

$$g_i(x) = r_i\left(\frac{\|\boldsymbol{x}-\boldsymbol{v}_i\|}{\sigma_i}\right) \tag{2.35}$$

where $\boldsymbol{x}$ is the input vector, $\boldsymbol{v}_i$ is the vector denoting the center of the neuron unit g_i with σ_i as its unit width parameter,and r_i is the activation function. The most widely used from of RBF uses the Gaussian activation function given by

$$g_i(x) = exp\left(\frac{-\|\boldsymbol{x}-\boldsymbol{v}_i\|^2}{2\sigma_i^2}\right) \tag{2.36}$$

A typical output of an RBF network having *n* units in the hidden layer and *r* output units is given by:

$$o_i(x) = \sum_{i=1}^{n} w_{ij} g_i(\boldsymbol{x}) \quad j = 1, \ldots, r \tag{2.37}$$

where w_{ij} is the connection weight between the *i*-th neuron unit and the *j*-th output, and g_i is the *i*-th neuron unit.

The standard technique used to train an RBF network is the hybrid approach, which is a two-stage learning strategy. The first step is to train the RBF network layer to get the adaptation of centers and scaling parameters using the unsupervised training. The centers ($\mathbf{v}_i$) or RBFs for each hidden node define input vectors causing maximal activation of these units. The widths ($\boldsymbol{\sigma}_i$) of the RBFs of each hidden node determine the radii of the areas of the input space around the centers where activations of these nodes are significant. The second step is to adapt the weights of the output layer using the supervised training algorithm. For this step supervised learning-techniques such as the least-squares method or the gradient method can be used to update the weights between the hidden layer and the output layer.

RBFs networks can be used for both classification and function approximation. In classification the output of the learning algorithm is one of a set of possible classes rather than the value of a continuous function. Outputs of the estimated function are interpreted as being proportional to the probability that the input belongs to the corresponding class. Outputs with probability of 0 are interpreted as belonging to any other class.

2.5.3 RIPPER

Repeated Incremental Pruning to Produce Error Reduction (RIPPER) was proposed by William W. Cohen [99] as an optimized version of IREP [100]. IREP tightly integrates reduced error pruning with a separate-and-conquer rule learning algorithm. It builds up a

rule set in a greedy fashion, one rule at a time. After a rule is found, all examples covered by the rule (both positive and negative in binary classes) are deleted. This process is repeated until there are no positive examples, or until the rule found by IREP has an unacceptable large error rate.

The improvements made by Cohen include an alternative metric for assessing the value of rules in the pruning phase of IREP; a new heuristic for determining when to stop adding rules to a rule set; and an addition function to optimize a rule set in an attempt to more closely approximate conventional reduced error pruning.

RIPPER consists of two main stages. It first builds a ruleset by repeatedly adding rules to an empty ruleset until all positive examples are covered. Rules are formed by greedily adding conditions to the antecedent of a rule (starting with empty antecedent) until no negative examples are covered. The second stage further optimizes the ruleset initially obtained so as to reduce its size and improve its fit to the training data. These stages are repeated for *k* times. The rule induction algorithm is called inside RIPPER for *k* times, and at each iteration, the current dataset is randomly partitioned in two subsets: a growing set, that usually consists of 2/3 of the examples and a pruning set, consisting in the remaining 1/3. These subsets are used for two different purposes: the growing set is used for the initial rule construction (the rule growth phase) and the pruning set is used for the pruning (the rule pruning phase).

2.5.4 KStar

KStar is an instance-based learner which uses an entropy distance measure. Instance-based learners classify an instance by comparing it to a database of pre-classified examples. Instead of trying to create rules, the learner works directly from the examples themselves. An important problem is how to define similar instance and similar classification. The main components of an instance-based learner are the distance function which determines how similar two instances are, and the classification function which specifies how instance similarities yield a final classification for the new instance.
In instance-based learning, each new instance is compared with existing ones using a distance metric, and the closes existing instance is used to assign the class to the new one; this is the case of the nearest-neighbor classification method. Generally, instance-based learners use the Euclidean distance as their distance metric; KStar uses an entropy-based distance instead. The key idea of this entropy-based distance metric is to define the distance between two instances as the complexity of transforming one instance into the other.

To calculate the complexity first a finite set of transformations which map instances to instances is defined. A program to transform one instance *a* into another *b* is a finite sequence of transformations starting at *a* and terminating at *b*. The complexity of this program is simply the number of operations that are executed. However, if one only computes the complexity of the shortest program, the resulting distance measure is very sensitive to small changes in an instance. A more desirable distance measure can be created by summing over *all* programs that transform *a* into *b*. More specifically, the

complexity is determined as the probability that a randomly-selected program of length *c* will transform *a* into *b*; this probability is then summed over all *c*. The sum (*P**) is difficult to compute, but expression for it are provided for integer-, real-, and categorical-valued data, as well as a rule for combining individual attributes into a feature vector. The distance *K** is then defined as $K^*(b \mid a) = -\log_2 P^*(b \mid a)$ [101].

2.5.5 Random Forest

A random forest is an ensemble classifier that consists of a collection of decision trees. If the number of cases in the training set is *N*, each tree is grown by sampling *N* cases at random (with replacement) from the original data. This sample will be the training set for that tree. Given *M* input variables, a number $m<<M$ is specified such that at each node, *m* variables are selected at random out of the *M* and the best split on these *m* is used to split the node. The value of *m* is held constant for all trees, and trees are not pruned.

2.5.6 Support Vector Machines

Suppport Vector Machines (SVMs) is a machine learning algorithm introduced by Cortes and Vapnik [102]. The basic idea is to find an *N*-dimensional hyper plane that optimally separates the data into their two categories, in the case of binary classification. To construct the hyperplane, SVMs use a kernel induced feature space which casts non-linearly separable data into a higher dimensional space where the data are separable. A good separation is attained by the hyperplane with the largest distance to the nearest training data points of any class; the goodness of a hyperplane is defined by the width of

its margin, which is the width that the boundary can be increased before hitting a data point.

For a binary classification problem, let's consider l training examples $\{x_i,y_i\}$, i=1…l, where each example has d inputs ($x_i \in \mathbf{R}^d$), and a class label with one of two values ($y_i \in$ {-1,1}). All hyperplanes in R^d are parameterized by a vector ($\boldsymbol{w}$), and a constant (b), expressed in the equation

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{2.38}$$

where $\cdot$ denotes the dot product and $\boldsymbol{w}$ the vector orthogonal to the hyperplane. Given such a hyperplane ($\boldsymbol{w}$,b) that separates the data, this gives the function

$$f(x) = \text{sign}(w \cdot x + b) \tag{2.39}$$

which correctly classifies the training data. We consider those pairs $\{\lambda\mathbf{w},\lambda b\}$ for $\lambda \in \mathbf{R}^+$ that satisfy [103]:

$$x_i \cdot w + b \geq 1 \text{ when } y_i = +1 \tag{2.40}$$

$$\boldsymbol{x}_i \cdot \boldsymbol{w} + b \leq 1 \; when \; y_i = -1 \tag{2.41}$$

or more compactly,

$$y_i(\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 \quad \forall_i \tag{2.42}$$

For a given hyperplane ($\boldsymbol{w}$,b), all pairs $\{\lambda\boldsymbol{w},\lambda b\}$ define the exact same hyperplane, but each has a different functional distance to a given data point. To obtain the geometric distance from the hyperplane to a data point, we must normalize by the magnitude of $\boldsymbol{w}$. This distance is simply [103]:

$$d\big((\boldsymbol{w}, b), \boldsymbol{x}_i\big) = \frac{y_i(\boldsymbol{x}_i \cdot \boldsymbol{w} + b)}{\|\boldsymbol{w}\|} \geq \frac{1}{\|\boldsymbol{w}\|} \tag{2.43}$$

we obviously want the hyperplane that maximizes the geometric distance to the closest data points. We can see an example in Figure 2.4.

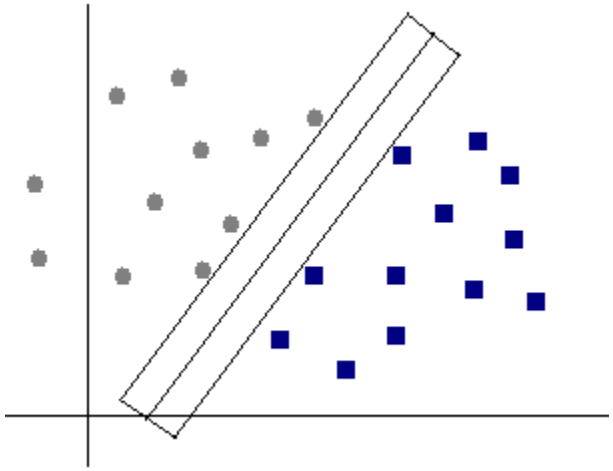

Figure 2.4 Hyperplane with the widest margin

To maximize the distance we want to minimize $\|\boldsymbol{w}\|$ subject to $y_i(\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 \quad \forall_i$.

The main method of doing this is with Lagrange multipliers. The problem is transformed into minimizing $W(\alpha) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$ subject to $\sum_{i=1}^{l} y_i \alpha_i = 0,\ 0 \leq \alpha_i \leq C\ (\forall_i)$, where α is the vector of l non-negative Lagrange multipliers to be determined and C is a constant. We define the matrix $(H)_{ij} = y_i y_j (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$ and minimize $W(\alpha) = -\alpha^T 1 + \frac{1}{2}\alpha^T H \alpha$ subject to $\alpha^T = 0, 0 \leq \alpha \leq C1$.

The optimal hyperplane can be written as:

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x}_i \qquad (2.44)$$

The vector $\boldsymbol{w}$ is a linear combination of the training vectors.

When the functional distance of an example is greater than 1 (when $y_i(\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1$) then α_i = 0. In this way, only the closest data points contribute to $\boldsymbol{w}$. The training examples for which α_i>0 are named support vectors, and these are the only ones needed in defining and finding the optimal hyperplane.

Assuming we have the optimal α (from which we construct $\boldsymbol{w}$), we still need to determine b to specify the hyperplane. To do this, take any positive and negative support vector, x^+ and x^-, for this we already know

$$(w \cdot x^+ + b) = +1$$

$$(w \cdot x^- + b) = -1$$

Solving these equations gives us

$$b = -\frac{1}{2}(\boldsymbol{w} \cdot \boldsymbol{x}^+ + \boldsymbol{w} \cdot \boldsymbol{x}^-) \qquad (2.45)$$

For data that are not linearly separable, we can pre-process the data so the problem is transformed into one of finding a simple hyperplane. The hyperplanes in the higher dimensional space are defined as the set of points whose inner product with a vector in that space is constant. The mapping used by SVMs schemes are designed to ensure that the dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function selected to suit the problem. We define a mapping $z=\varphi(x)$ that transforms the d dimensional input vector $\boldsymbol{x}$ into a higher d' dimensional vector $\boldsymbol{z}$. We hope to choose a $\varphi()$ so that the new training data $\{\varphi(\boldsymbol{x}_i), y_i\}$ is separable by a hyperplane.

Given a mapping $z=\varphi(x)$, we set up the new optimization problem by replacing all occurrences of $\boldsymbol{x}$ with $\varphi(\boldsymbol{x})$. Eq 2.44 would be

$$\boldsymbol{w} = \sum_{i=1}^{l} \alpha_i y_i \varphi(\boldsymbol{x}_i) \quad (2.46)$$

and Eq (2.39) would be

$$f(x) = sign(\boldsymbol{w} \cdot \varphi(\boldsymbol{x}) + b) \quad (2.47)$$

$$= sign([\sum_i \alpha_i y_i \varphi(\boldsymbol{x}_i)] \cdot \varphi(\boldsymbol{x}) + b)$$

$$= sign(\sum_i \alpha_i y_i (\varphi(\boldsymbol{x}_i) \cdot \varphi(\boldsymbol{x})) + b)$$

For these equations, anytime a $\varphi(\boldsymbol{x}_a)$ appears, it is always a dot product with some other $\varphi(\boldsymbol{x}_b)$. If we know the kernel for the dot product in the higher dimensional feature space,

$$K(\boldsymbol{x}_a, \boldsymbol{x}_b) = \varphi(\boldsymbol{x}_a) \cdot \varphi(\boldsymbol{x}_b) \quad (2.48)$$

we will not need to deal with the mapping $z=\varphi(\boldsymbol{x})$ directly; our classifier would be

$$f(x) = sign\left(\sum_i \alpha_i y_i \left(K(\boldsymbol{x}_i, \boldsymbol{x})\right) + b\right) \quad (2.49)$$

Once the problem is set up in this way, finding the optimal hyperplane proceeds as usual, with the difference that the hyperplane will be in some unknown feature space.

2.6 Function approximation algorithms

2.6.1 Support Vector Regression

SVMs can also be applied to function approximation problems by the introduction of an alternative loss function. A similarity of the maximal margin is constructed in the space of the target values *y* by using Vapnik's ϵ-insensitive loss function [104].

To estimate a linear regression

$$f(x) = \boldsymbol{w} \cdot \boldsymbol{x} + b \quad (2.50)$$

we minimize

$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}|y_i - f(\boldsymbol{x}_i)|_\varepsilon$ (2.51)

We can transform this into a optimization problem with constraints by introducing "slack" variables. We need two types of slack variable for the two cases $f(\boldsymbol{x}_i)$-$\boldsymbol{y}_i > \epsilon$ and y_i-$f(\boldsymbol{x}_i) > \epsilon$, denoted ξ^- and ξ^+ respectively.

The optimization problem is given by

minimize $\tau(\boldsymbol{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}(\xi_i^- + \xi_i^+)$ (2.52)

subject to $f(\boldsymbol{x}_i) - y_i \leq \varepsilon + \xi_i^-$ (2.53)

$y_i - f(\boldsymbol{x}_i) \leq \varepsilon + \xi_i^+$ (2.54)

$\xi_i^-, \xi_i^+ \geq 0$ (2.55)

Introducing Lagrange multipliers, we arrive at the following optimization problem (choosing previously C, $\epsilon \geq 0$):

maximize $W(\alpha, \alpha^*) = -\varepsilon\sum_{i=1}^{l}(\alpha_i^* + \alpha_i) + \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)y_i$

$$-\frac{1}{2}\sum_{i=1}^{l}(\alpha_i^* - \alpha_i)\left(\alpha_j^* - \alpha_j\right)k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) \quad (2.56)$$

subject to $0 \leq \alpha_i, \alpha_i^* \leq C$ for all $i = 1, \ldots, l$ and $\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0$. (2.57)

The regression takes the form

$f(x) = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)k(x_i, x) + b,$ (2.58)

where b is computed using the fact that Eq (2.53) becomes an equality with $\xi_i^-=0$ if $0 < \alpha_i < C$, and Eq (2.54) becomes an equality with $\xi_i^+=0$ if $0 < \alpha_i^* < C$.

2.6.2 Linear regression

Linear regression is an approach to model the relationship between a dependent variable Y and independent variables X. In simple linear regression if we want to model n data

points denoted by (X_1,Y_1), $(X_2,Y_2),\ldots(X_n,Y_n)$ there is one independent variable, X, and two parameters, β_0 and β_1 [105]:

$$Y = \beta_0 + \beta_1 X + E \tag{2.59}$$

where E is an error term and β_0 and β_1 are regression coefficients to be estimated. Using least squares we can estimate β_0 and β_1 as:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sum_{i=1}^{n}(X_i-\bar{X})^2} \tag{2.60}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1\bar{X} \tag{2.61}$$

where $\bar{X}$ and $\bar{Y}$ are the mean of the values on X and the observations on Y, respectively. The least-squares line may generally be represented by

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X \tag{2.62}$$

After estimating $\hat{\beta}_0$ and $\hat{\beta}_1$, a point estimate of E at the value X is calculated as

$$\hat{E} = Y - \hat{Y} = Y - \left(\hat{\beta}_0 + \hat{\beta}_1 X\right) \tag{2.63}$$

The estimated error $\hat{E}$ is typically called a residual. If there are n (X,Y) pairs (X_1,Y_1), $(X_2,Y_2),\ldots,(X_n,Y_n)$, then there are n residuals $\hat{E}_i = Y_i - \hat{Y}_i = Y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 X_i\right), i = 1, \ldots, n$

A multiple regression model is given by any second- or higher-order polynomial. Adding higher-order terms to a model can be considered equivalent to adding new independent variables. The following second-order model

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + E \tag{2.64}$$

can be rewritten as

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + E \tag{2.65}$$

The general form of a regression model for k independent variables is given by

$$Y = \beta_0 + \beta_1 X_i + \beta_2 X_2 + \cdots + \beta_k X_k + E \tag{2.66}$$

where $\beta_0, \beta_1, \ldots, \beta_k$ are the regression coefficients that need to be estimated.

In general, the least-squares method chooses as the best-fitting model the one that minimizes the sum of squares of the distances between the observed responses and those predicted by the fitted model. Thus, if

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_i + \hat{\beta}_2 X_2 + \cdots + \hat{\beta}_k X_k \quad (2.67)$$

represents the fitted regression model, then the sum of squares of deviations of observed *Y*-values from corresponding values predicted by using the fitted regression model is given by

$$\sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2 = \sum_{i=1}^{n} \left(Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_{1i} - \cdots - \hat{\beta}_k X_{1i}\right)^2 \quad (2.68)$$

The least-squares solution then consists of the values $\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_k$ for which the sum in Eq (2.68) is a minimum.

2.6.3 PACE regression

Projection Adjustment by Contribution Estimate (PACE) regression is a recent approach to fitting linear models developed by Wang in 2000 [106]. The basic idea of regression analysis is to fit a linear model to a set of data. Although the classical ordinal least squares (OLS) estimator is simple and has well-established theoretical justification, the models it produces are often not satisfactory; in some models produced by OLS have worse predictive performance on unseen data than simpler models that consider fewer variables. PACE regression improves the classical OLS regression by evaluating the effect of each variable and using a clustering analysis to improve the statistical basis for estimating their contribution to the overall regressions.

PACE regression consists of six procedures which build models by adjusting the orthogonal projections based on estimations of the expected dimensional contributions. Of the six procedures, the first two perform model selection, as they select a subset from a sequence. The next two procedures address the dimension-based situation, where each orthogonal dimension is tested and selected (or not) individually. The last two procedures update the absolute distances of the estimated model to values chosen appropriately from the non-negative half real line.

2.6.4 Multilayer perceptron

The multilayer perceptron is an artificial neural network structure that can be used for classification and regression. The multilayer perceptron consists of an input, an output layer and one or more hidden layers of nonlinearly-activating nodes. Each layer may have a different number of neurons, and even a different transfer function. Each neuron in one layer connects with a certain weight w_{ij} to every neuron in the following layer. The backpropagation algorithm, which was originally developed by Werbos in 1974 [107], is implemented in the multilayer perceptron topology. The algorithm is based on the gradient descent technique for solving an optimization problem, which involves the minimization of the network cumulative error E_c. E_c represents the sum of n squared errors E(k)'s where $E(k) = \frac{1}{2}\sum_{i=1}^{q}[t_i(k) - o_i(k)]^2$ is the square of the Euclidean norm of the difference between the target output vector $\boldsymbol{t}(k)$ and the output vector $\boldsymbol{o}(k)$ produced by the perceptron. n is the number of training patterns presented to the network for learning purposes. Using the sigmoid function as the activation function for all the neurons of the network, E_c is defined as [108]:

$$E_c = \sum_{k=1}^{n} E(k) = \frac{1}{2}\sum_{k=1}^{n}\sum_{i=1}^{q}[t_i(k) - o_i(k)]^2 \quad (2.69)$$

We update the weights of the neurons which minimize the error in the entire output given by E_c. The updating rule is expressed as:

$$\Delta w^l = -\eta \frac{\partial E(k)}{\partial w^l} \quad (2.70)$$

where $\frac{\partial E(k)}{\partial w^l}$ is the gradient of the error $E(k)$ with respect to the vector $\boldsymbol{w}^{(l)}$ corresponding to all interconnection weights between layer (l) and the preceding layer (l-1). η is the learning rate, usually between 0 and 1, selected to ensure that the weights converge to a response fast enough. $\Delta\mathbf{w}^{(l)}$ is the difference between the vectors $\boldsymbol{w}^{(l)}(k+1)$ and $\boldsymbol{w}^{(l)}(k)$. $\Delta w_{ij}^{(l)}$ is the weight update for neuron j of layer (l-1) connecting to neuron i located at layer l, and $o_j^{(l-1)}$ is the output of the j-th neuron at layer (l-1). The signal $tot_i^{(l)}$ is the sum of all signals reaching neuron (i) at a hidden layer (l) coming from the previous layer (l-1). Using chain rule differentiation we obtain:

$$\Delta \boldsymbol{w}^{(l)} = \Delta w_{ij}^{(l)} = -\eta \left[\frac{\partial E(k)}{\partial o_i^{(l)}}\right]\left[\frac{\partial o_i^{(l)}}{\partial tot_i^{(l)}}\right]\left[\frac{\partial tot_i^{(l)}}{\partial w_{ij}^{(l)}}\right] \quad (2.71)$$

For the case where layer (l) is the output layer (L), Eq (2.71) becomes:

$$\Delta w_{ij}^{(L)} = \eta\left[t_i - o_i^{(L)}\right]\left[f'\left(tot_i^{(L)}\right)\right] o_j^{(L-1)} \quad (2.72)$$

where $f'\left(tot_i^{(l)}\right) = \frac{\partial f(tot_i^{(l)})}{\partial tot_i^{(l)}}$

If we represent $\delta_i^{(L)} = [t_i - o_i^L]\left[f'(tot_i^{(L)}\right]$ as being the error signal of the i-th node of the output layer, we obtain the expression of the weight update at layer (L):

$$\Delta w_{ij}^{(L)} = \eta \delta_i^{(L)} o_j^{(L-1)} \quad (2.73)$$

In the case where f is the sigmoid function, the error signal is:

$$\delta_i^{(L)} = [(t_i - o_i^L) o_i^L (1 - o_i^L)] \quad (2.74)$$

If we propagate the error backward, for the case where (l) represents a hidden layer ($l < L$), the expression of $\Delta w_{ij}^{(l)}$ is given by:

$$\Delta w_{ij}^{(l)} = \eta \delta_i^{(l)} o_j^{(l-1)} \quad (2.75)$$

where the error signal $\delta_i^{(l)}$ is now expressed as a function of output of previous layers as:

$$\delta_i^{(l)} = f'(tot_i^{(l)}) \sum_{p=1}^{n_l} \delta_p^{(l+1)} w_{p_i}^{(l+1)} \quad (2.76)$$

When f is taken as the sigmoid function, $\delta_i^{(l)}$ becomes:

$$\delta_i^{(l)} = o_i^{(l)} (1 - o_i^{(l)}) \sum_{p=1}^{n_l} \delta_p^{(l+1)} w_{p_i}^{(l+1)} \quad (2.77)$$

The index n_l in Eqs (2.76) and (2.77) represents the total number of neurons on layer (l+1) and the index p represents the neurons of layer (l+1).

The procedure to train a multilayer perceptron network using the backpropagation algorithm is described as follows:

1. Initialize the weights to small random values
2. Choose an input-output pattern to the network (***x****(k)*, ***t****(k)*)
3. Compute the output values for all i neurons at every layer (l)
4. Calculate the total error value $E=E(k) + E$ and the error signal
5. Update the weights for all nodes in the network
6. Repeat the process from step 2 using another exemplar. After all exemplars have been used, we have trained one epoch.
7. Check the cumulative error E in the output layer until the error is lower than the predetermined value.

2.6.5 Sequential Minimal Optimization

Sequential Minimal Optimization (SMO) was proposed by John C. Platt [109]. It is an algorithm for training SVMs. SMO breaks the quadratic programming (QP) problem of SVMs into a series of small sub-problems. Using small QP problems helps in obtaining quick and simple analytical solution. The SMO chooses to solve the smallest possible optimization problem at every step, which involves two Lagrange multipliers. The Lagrange multipliers must obey a linear equality constraint. At every step, SMO optimizes two Lagrange multipliers, finds their optimal values, and updates the SVM to reflect the new optimal values. The key point is that solving for two Lagrange multipliers can be done analytically without invoking a quadratic optimizer.

There are two components to SMO: an analytic method for solving for the two Lagrange multipliers, SMO first computes the constraints on these multipliers and then solves for the constrained minimum, and a heuristic for choosing which multipliers to optimize. There are two separate choice heuristics, one for each Lagrange multiplier. For the first multiplier, each example is checked against the Karush–Kuhn–Tucker (KKT) [110] conditions, those examples violating these are eligible for optimization. The second Lagrange multiplier is chosen to maximize the size of the step taken during joint optimization. The algorithm is shown next:

1. Find a Lagrange multiplier α_1 that violates the KKT conditions for the optimization problem.
2. Select a second multiplier α_2 and optimize the pair (α_1,α_2).
3. Repeat steps 1 and 2 until convergence.

2.7 Performance measures

Confusion matrices (Kohavi and Provost, 1998) are commonly used when evaluating the performance of classification algorithms on machine learning as they contain information about actual and predicted classifications. Table 2.4 shows the confusion matrix used in binary classification problems. True negative (TN) is the number of correct predictions that an example is negative; false positive (FP) is the number of incorrect predictions that an example is positive; false negative (FN) is the number of incorrect of predictions that an example is negative; and true positive is the number of correct predictions that an example is positive.

Table 2.4 Confusion matrix for two classes

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Several performance measures can be obtained from a confusion matrix. Classification accuracy is the performance measure generally associated with machine learning algorithms and is defined as Accuracy = (TP + TN)/(TP + FP + TN + FN) .

2.7.1 Kappa statistic

As overall accuracy is a poor measure of classifier performance on imbalanced data, we use Cohen's Kappa statistic [111] computed from the confusion matrices of each experiment. The kappa statistic is intended to represent the chance-corrected agreement between two raters. This is calculated by [112]:

$$k = \frac{F_O - F_C}{1 - F_C} \tag{2.78}$$

where F_O is the observed agreement between raters, and F_C is the probability that the raters agree by chance. F_O is computed as the proportion of items the raters agree on to the total number of items. In a confusion matrix, this is the sum of true positives and true negatives divided by the total number of items. F_C , on the other hand, is calculated as [112]:

$$F_C = \sum_{j=1}^{n} X_j \cdot Y_j \tag{2.79}$$

where X and Y are the raters, and X_j and Y_j are given by

$$X_j = \left(\sum_{i=1}^{n} CM_{ij}\right)/N \tag{2.80}$$

$$Y_j = \left(\sum_{i=1}^{n} CM_{ji}\right)/N \tag{2.81}$$

where CM_{ij} is an element of the n x n confusion matrix CM, n is the total number of categories, and N is the total number of examples. X_j is the fraction of items rater X assigned to category j, and Y_j is the fraction of items rater Y assigned to category j. Eq. (2.79) takes the probabilities of the two judges agreeing on each category, and sums across all categories to arrive at the total agreement by chance. In our experiments we compute the kappa statistic between the outputs of a classifier and the correct classification for each input, yielding the chance-corrected accuracy of the classifier. This statistic essentially tells us the degree to which the classification is improved with respect to merely guessing the majority class. The kappa statistic expressed in terms of the confusion matrix elements is:

Kappa statistic =

((TN+TP)/(TN+FN+FP+TP)) - ((TN+FN)/(TN+FN+FP+TP) *

(TN+FP)/(TN+FN+FP+TP)) + ((FP+TP)/(TN+FN+FP+TP) *

(FN+TP)/(TN+FN+FP+TP)) /

(1- ((TN+FN)/(TN+FN+FP+TP) * (TN+FP)/(TN+FN+FP+TP)) +

((FP+TP)/(TN+FN+FP+TP) * (FN+TP)/(TN+FN+FP+TP))). (2.82)

The values of kappa are defined in the [-1,1] interval, where kappa=1 represents perfect agreement, kappa=0 represents agreement equivalent to chance and kappa=-1 represents perfect disagreement between raters. In highly skewed datasets, where classifiers may simply predict the majority class for all instances, we believe that this statistic will be more revealing than classification accuracy. However, the kappa statistic is not a true measure, as its possible range of values is in practice constrained by the dataset's characteristics; despite this, it is accepted as a tool for making comparisons within one dataset. Other alternatives would include the geometric mean of the individual accuracies for each class (as in [13, 113]), the area under the Receiver Operating Characteristic (ROC) curve (often referred to as AUC), the area under the Precision-Recall curve, the Kolmogorov-Smirnov (K-S) statistic, or the F-Measure [114]. However, the AUC for a classifier is considerably more difficult to compute; one must generate several instances of a base classifier on the same dataset, varying the learning parameters to obtain different confusion matrices. These yield points on the ROC curve; the full curve itself is then determined via interpolation between these points, and the AUC is then computed from these linear segments. As there is no know means to control the *x*-coordinate

positions of the points identified on the ROC curve (they will be dataset- and classifier-dependent), the AUC is a somewhat approximate statistic [115]. ROC curves and AUC have of course been used to compare learners in previous stratification studies, but we believe the Kappa statistic is a more appropriate point estimate within a single dataset in a parameter search such as in this dissertation. A similar critique applies to the area under the Precision-Recall curve. The geometric mean accuracy is again dataset-specific, in that the optimal Bayes classifier (and hence the optimal classification accuracy) varies from one dataset to another [12]. As with the kappa statistic, the geometric mean accuracy is a valid tool for making comparisons within a single dataset. The F-measure is the harmonic mean of precision and recall, and the K-S statistic examines the maximum difference in the empirical distributions of the classes. Both are again dataset-specific, and valid tools for comparisons within a dataset. However, unlike the kappa statistic, none of these last three explicitly considers class imbalance. We have thus determined that the kappa statistic is the most congruent to our objectives for the experiments in Chapters 3 and 4.

2.7.2 Root Mean Square Error

The Root Mean Square Error (RMSE) is a frequently used measure of the difference between the values predicted by a model and the observed value of the response variable. This is a measure of how accurately the model predicts the response variable. Lower values of RMSE indicate a better fit. RMSE is defined as:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{n}} \tag{2.83}$$

where y_i represents the observed value for the response variable and $\hat{y}_i$ is the predicted value.

2.7.3 Normalized Mean Square Error

We use normalized mean square error (NMSE) as our performance measure in Chapter 5. This the most common performance measure from the empirical evaluations of the existing methods for correcting selection bias in the machine learning literature [66, 68, 116-118]. NMSE is given by:

$$NMSE = \frac{1}{\delta^2 n} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2.84}$$

where δ^2 is the pooled standard deviation and it is defined as

$$\delta^2 = \frac{1}{n-1} \sum_{i=1}^{N} (y_i - \bar{y})^2 \tag{2.85}$$

where y_i represents the observed value for the response variable, $\hat{y}_i$ is the predicted value, $\bar{y}$ is the mean of the y_i observed values of the response variable.

2.7.4 Pearson Product Moment Correlation

The Pearson product moment correlation, called Pearson's *r*, reflects the degree of linear relationship between two variables. It ranges from +1 to -1. A correlation of +1 means that there is a perfect positive linear relationship between variables; a correlation of 0 means there is no linear relationship between the two variables while a correlation of -1 means that there is a perfect negative linear relationship. Pearson's *r* correlation coefficient is defined as the covariance of two variables divided by the product of their standard deviations [119]:

$$r = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2.86}$$

Substituting estimates of the covariances and variances based on a sample gives the sample correlation coefficient:

$$r = \frac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\bar{Y})^2}} \quad (2.87)$$

2.7.6 Coefficient of determination r^2

The coefficient of determination r^2 gives the proportion of the variance of one variable that is predictable from the other variable. It is a measure that allows us to determine how certain one can be in making predictions from a certain model. r^2 is expressed as a value between zero and one. A value of one indicates a perfect fit, and therefore, a very reliable model for future forecasts. A value of zero, on the other hand, would indicate that the model fails to accurately model the dataset. r^2 can be calculated as the square of Pearson's *r*.

Chapter 3. SMOTE-Based Stratification in Software Defect Prediction Datasets: A Case Study

3.1 Introduction

Software systems pervade our lives. They run our cars, fly our planes, manage our power grid, and the list goes on. Our modern technological society is absolutely dependent on software systems, and software failures can easily cause economic damage, personal injury or even death. Thus, it is unsettling to realize that the quality of software systems in general lags far behind the quality of any other engineered product that humans create. Software failures currently cost the U.S. economy alone more than $78 billion per year [120]. A great many researchers have proposed re-using existing software components (which presumably are of superior quality than new components) as a way to improve programmer productivity and the overall quality of a software system [121]. However, this improvement in quality depends on the reliability of the components being reused, which is usually unknown *a priori*; hence some means of estimating the reliability of a component is needed. This is a very active area of research, as accurately estimating software reliability would greatly benefit the software industry.

There are numerous approaches for acquiring a reusable software component. One could purchase commercial off-the-shelf (COTS) software [122]. Alternatively, a component could be developed in-house; many organizations have also made huge investments in

building up reusable component libraries, in the hopes of improving their software products and processes [123]. The emergence of the open-source software movements has also provided a new source of reusable software components. Open source involves two important properties: visible source code and a right to develop derivative works, meaning that the code can (often) be freely reused [124]. These alternatives all hold the promise of making software development quicker and cheaper, or delivering more functionality within a fixed budget and schedule. However, the quality of these components is unknown; by definition, a reused component was *not* primarily designed to be part of the new system [122, 124].

In all cases, incorporating reusable components is no mere matter of plug-and-play. A reuse process must be established, which will consider questions such as the appropriateness of the component for the new problem domain, the quality (reliability, performance, maintainability, etc.) of the component, availability of supporting software artifacts (requirements & design documents, test plans or test suites, etc.), and the degree of reengineering required. Processes for producing software in an organization must also change when the organization commits to building up a reusable component library [123]. In this chapter, we focus on one particular element of the reuse process, namely determining the reliability of a module in a software component.

Software reliability is defined as the probability that a given software system or component, operating in a given environment, will not fail for a given period of time [125]. It is notoriously difficult to estimate this probability; the history of attempts

stretches from the Jelinsky-Moranda model proposed in 1971 [126] through to the present day. A proxy for the actual (unknown) reliability is the predicted number of defects in a module, which is estimated using software metrics. Modules with a higher number of predicted failures require more quality improvement resources. Again, estimating this number has proven stubbornly difficult; there is no accepted form for the relationship between software metrics and reliability. An alternative approach (usually termed fault-proneness) simply attempts to model whether a module will experience a failure or not (i.e. defect prediction is converted into a binary classification problem) [78, 79]. On a practical level, most organizations rely on a very simple Pareto analysis: they select a few software product metrics, and select the 20% of modules with the highest metric values for additional quality improvement effort. This approach does help, but numerous defects still slip through into field-deployed systems.

Evidence has shown that human experts are very poor at identifying fault-prone software modules [127]. The current state of the art in software defect prediction (defined as a regression or function-approximation problem) is to attempt to fit a statistical model (e.g. [128]) to relate a set of software metrics to the number of defects that will arise in that module. However, software defect prediction is an inexact science. Each software project has a unique combination of intellectual difficulty, size, development team composition, and third-party components, among other factors. The result is that there is no "critical value" for any software metric that definitively indicates when a module will be of lower quality. Indeed, there is not even an accepted parametric model form (e.g. linear, quadratic, Poisson regression, etc.) for predicting the number of faults in a module. Part

of the problem is that there is no theory of software metrics to guide the selection of such a model form, meaning that parametric regression is reduced to a trial-and-error search through an infinite space of possible model forms. It is thus not surprising that nonparametric techniques, especially machine learning and computational intelligence, have been employed for software defect prediction. However, any machine- learning approach is dependent on the quality of the data used to train the algorithm, and this is a serious shortcoming of current software reliability modeling. Publicly-available datasets do not reflect current industry approaches to software design, i.e. data from large object-oriented (OO) programs is generally not available.

In this chapter we base our experiments on the Mozilla dataset, which is a new, large-scale software defect-prediction dataset created by Sadia [9] in 2005, derived from a major open-source software system [129]. Using the source code and defect-tracking database of the Mozilla project, the dataset was created from the 8,349 C++ classes that provide the back-end functionality for the Mozilla web browser. This dataset covers over 900,000 lines of C++ code. 18 OO were computed metrics for each class, and joined them to a defect measure extracted from the defect-tracking database (Bugzilla). The result is a dataset of 18 numeric predictor attributes, one numeric target attribute, with 8,349 records and no missing values, drawn from a modern, large-scale, industrial-quality software system. We attempt to predict both the number of defects and fault-proneness, the latter by converting the numeric target attribute into a binary class variable (this has the effect of reducing variance in the class variable). To deal with the remaining skewness in the data, we employ stratification-based resampling [10]. We thus obtain a

classifier that exhibits greater sensitivity to the minority (fault-prone) class, even though this is only 5% of the original dataset.

3.2 Background: The Mozilla Dataset

The Mozilla dataset was created in [9] in response to a crucial need for defect-prediction datasets that combine OO metrics and defect measures for large-scale modern software systems. The majority of publicly-available datasets are quite small (i.e. relatively few modules in the program), and collect only a small set of procedure-oriented metrics, meaning they do not reflect modern software development processes. In addition, some datasets do not relate the metrics for a module to any defect measure for that module; nothing beyond shotgun correlations may be found in such a dataset. Some defect-prediction datasets are based on reasonably large projects (including those archived at the PROMISE Software Engineering Repository [130], the NASA Metrics Data Program [131], and another NASA site [132]). Object-oriented (OO) metrics are collected for several datasets in [130] and [131], and several large ones in [132], while defect measures are collected for the datasets in [130] and [131]. However, to the best of our knowledge, there is no other publicly-available dataset that simultaneously collects object-oriented metrics and defect measures over a large number of modules (the OO metrics datasets in [130] are quite small; the KC1 dataset covers 170 classes, while the GSM class-level datasets are of a similar size). This imposes a serious limitation on defect-prediction models, as the modern practice of object-oriented development has substantially changed programming styles. Classes (the principal unit of OO functionality) tend to be made up

of very few methods, and these methods in turn tend to be quite small (one study found that more than half of the methods in OO programs might contain less than five lines of executable code!) Furthermore, there tend to be a large number of interacting classes in a software system [75, 133]. The design considerations of inheritance, polymorphism and operator overloading also cause OO programs to be fundamentally different in structure than procedure-oriented programs.

Mozilla is a good candidate for a large-scale OO dataset as it is a modern, large-scale industrial-strength software system. It is in fact one of the largest open-source systems in existence, with a total codebase of over 10 million lines (written in multiple languages). As a point of comparison some of the Mozilla modules are individually larger than the entire Apache web server system [134]. In the remainder of this section, we will first review the Mozilla system, and then discuss the development of this dataset and a statistical characterization of it.

3.2.1 History and development of Mozilla

The Mozilla project [135] is an open-source community dedicated to the development of a modern web browser and supporting utilities (email client, calendar, address book, internet chat and a web page composer). At the time of this writing, Mozilla produces the Firefox browser, which holds a significant share of the browser market, at about 18.5% world-wide in December 2010 [136]. However, the story of the Mozilla project reaches back seven years before the launch of Firefox, to the release of an initial version by

Netscape Inc. in 1998. The early development of the community was actually quite uneven, despite the sponsorship of Netscape and its assignment of Netscape employees as full-time Mozilla developers. Mozilla was (and is) a very large software system. The original architecture was deemed cumbersome, and there was a licensing requirement for proprietary libraries. Finally, it took roughly two and a half years from project inception until the first production-quality browser was launched. However, by the end of 2000, this situation was improving, with expanded documentation on the Mozilla architecture and technology (how to build and test the product), tutorials, refined processes and development tools. In fact, some of the Mozilla development tools (notably the Bugzilla defect- tracking system) have been spun off into successful open-source projects of their own [134, 135]. As of this writing, the ongoing development of the Mozilla project is managed by the Mozilla.org staff and supported by the Mozilla Foundation.

In the Mozilla architecture, back-end functionality is provided by C++ libraries, while user interfaces are written in a mix of C++, JavaScript and XUL. The base library is XPCOM, which is written in C++, and provides the windowing engine, network communications and web page parsing. Cross-platform portability for XPCOM and other C++ libraries is provided by the Netscape Portable Runtime Environment (a virtual machine). The XPConnect libraries are built on top of XPCOM, and provides object transparency (via wrappers) between XPCOM and the Javascript user-interface code. XUL is an XML schema used to specify user-interface components. It is integrated with XPCOM and XPConnect [135].

The Bugzilla defect-tracking system is Mozilla's principal quality assurance tool. Accessible via a Web interface, it allows users anywhere in the world to report bugs in the Mozilla system. Bugzilla uses a MySQL database as its back-end. The database has individual fields recording basic data concerning each bug (severity, operating system, bug status, etc.), while test scripts and possible patches (bug fixes) can be uploaded as "attachments." Approval of any bug fix normally requires two levels of review: by the module owner, and by a "super-reviewer" who is an expert on the affected subsystem. The change can then be committed to the software configuration management system (since 2009, the site's repository was transitioned from the CVS version control system to Subversion), and the bug marked as fixed. However, the Bugzilla system and the version control system are not integrated, and so there is no enforcement mechanism requiring that a patch be placed in Bugzilla when a change is made to the source-code repository. It is up to the individual developing the patch to actually upload it to Bugzilla. Further inquiry with the Mozilla.org staff also revealed that in cases where a solution was deemed "trivial" or involved a module "not under reviewer control," patches might well be ignored, or a developer might have erroneously selected the wrong bug resolution in Bugzilla (e.g. "fixed" instead of "duplicate").

3.2.2 Dataset creation

The Mozilla dataset was constructed to relate software metrics to defects found in the Mozilla project source code. The creators [9, 129] focused on extracting object-oriented software metrics from the C++ classes contained in Mozilla, and relating them to defects

found in those classes and reported via the Bugzilla defect-tracking system. Creation of this dataset involved querying the Bugzilla database, extracting the defect counts, extracting software metrics from the Mozilla source code, and merging the defect counts and metric values. Their approach is similar to [137, 138], where custom scripts are used to export data from software repositories for further analysis. The dataset is based on a copy of Mozilla's Bugzilla instance, donated in mid-2003. Using the Mozilla version control system they checked out the corresponding version of Mozilla for the analysis. They focused on those bugs whose resolution is "fixed," and which have an associated patch for a C++ file. Of 190,722 reported bugs for Mozilla, 52,252 bugs are classified as "fixed," but only 4,665 of these had associated patches; fewer than 12,000 total bug reports contained patches, and not all of these are considered "fixed," as also observed in [139]. The results only cover the "fixed" bugs with associated patches in C++ files.

The defects in Mozilla were quantified via the mechanism of deltas, which are atomic changes to a single file [134]. A single bug fix (known as a patch, and stored in a text file) could contain multiple deltas. The key pieces of information extracted from these files were the names of the source files in the CVS repository (these begin with "RCS file:") that are modified by the patch. These are extracted using Perl scripts. Each time the name of a file appears in a patch, it is counted as one delta. Once all filenames have been extracted from all patches, then the number of occurrences of the filename are counted, and this is the delta value for that file. Then they map these file-level deltas to deltas per class. Standard coding practices usually mandate that each class definition and each method definition for that class be placed in separate files; however, they found that

this was not always the case in Mozilla. In a number of cases, multiple classes were mixed into a single file, which complicates the assignment of file-level deltas to a class. Their approach was to divide the total deltas per file by the fraction of executable lines of code belonging to the different classes in the file (extracted by the metrics tool). They then summed these fractional deltas across all files that make up a single class, producing their class-level delta attribute. While this is an approximation, it is justified by prior software metrics research, which consistently finds a strong linear correlation between the number of executable lines of code and the number of defects in a code module [78].

The Krakatau Professional software metrics tool [140] was used to extract function-, file-, and class-level metrics for the code under consideration. This consisted of over 935,000 lines of C++ code, in 8,349 separate classes. The final dataset consists of the CK metrics (excluding CBO, which was uniformly 0) as well as the following metrics for each class:

• CSA: # of attributes of a class.

• CSAO: # of attributes and operations for a class.

• CSI: (NOOC * DIT) / Total # of methods.

• CSO: # of operations for a class.

• NAAC: # of attributes added relative to the superclass. For a root class NAAC= CSA.

• NAIC: # of attributes inherited from the superclass. For a root class NAIC=0.

• NOAC: # of operations added relative to the superclass. For a root class NAOC=CSO.

• NOIC: # of operations inherited from the superclass. For a root class , NOIC=0.

• NOOC: # of overridden operations in a class. For a root class, NOOC=CSO.

• NpavgC: Total # of parameters / # of methods.

• OSavg: WMC / # number of methods

• PA: # of references to private members within all methods of a class

• PPPC: Percentage of members declared public or private.

The final dataset thus contains 8,349 records, consisting of 18 OO metrics and a numeric dependent variable (Delta). We are not aware of any other software quality dataset in which a large number of OO metrics has been collected over so many classes, and associated with a defect rate of any sort. The dataset can be accessed at [7].

3.2.3 Statistical Moments

A statistical characterization of the Mozilla-Delta dataset was performed in [9]. We first discuss statistical moments of the individual attributes, including skewness and kurtosis.

Table 3.1: Statistical Moments of the Dataset

Variable	Min	Max	Mean	Median	Std. Dev.	Kurtosis	Skewness
CSA	0	108	2.580	0	6.500	6.050	55.067
CSAO	0	328	9.652	3	17.796	5.889	57.287
CSI	0	44	0.182	0	0.796	26.145	1411.309
CSO	0	274	7.068	2	13.151	6.106	59.884
DIT	0	13	0.504	0	1.117	3.314	1191.834
LCOM	0	4014	6.302	0	86.527	30.827	1191.834
NAAC	0	108	2.589	0	6.555	6.090	55.492
NAIC	0	104	1.509	0	6.774	6.726	52.201
NOAC	0	226	5.917	2	10.982	6.844	76.558
NOCC	0	186	0.327	0	2.685	45.149	2865.077
NOIC	0	430	11.994	0	48.905	5.447	31.173
NOOC	0	89	1.161	0	5.129	8.443	91.378
NPavgC	0	10	0.762	0.5	0.961	1.859	6.426
OSavg	0	33	1.530	1	1.522	5.296	44.644
PA	0	279	2.390	0	10.997	9.515	131.698
PPPC	0	150	87.740	100	26.644	-2.193	3.748
RFC	0	675	10.247	2	26.842	9.735	153.224
WMC	0	1472	15.746	3	48.945	11.432	214.463
Delta	0	104	0.140	0	1.49	44.139	2856.022

The minimum, maximum, mean, median, standard deviation, skewness and kurtosis of each individual attribute are shown Table 3.1. As can be seen, the attributes are all seriously skewed, many severely so. Most attributes (with the exception of PPPC and NPavgC) have extremely high skewness values. In the Mozilla dataset (again with the exception of PPPC) the kurtosis values are positive, many strongly so. This indicates that the distributions are leptokurtic, or very narrow and sharply peaked. The negative kurtosis value for PPPC indicates that this attribute is platykurtic, or very broad with a flattened peak. It is particularly significant to note the high skewness and kurtosis of the Delta attribute, which is our continuous target attribute. A histogram plot of the Delta

attribute (see Figure 3.1) also indicates a severe skewness in the data; we present the histogram on a semilogarithmic plot for convenience. These characteristics will tend to have a negative impact on data mining algorithms, and function approximation algorithms in particular [141].

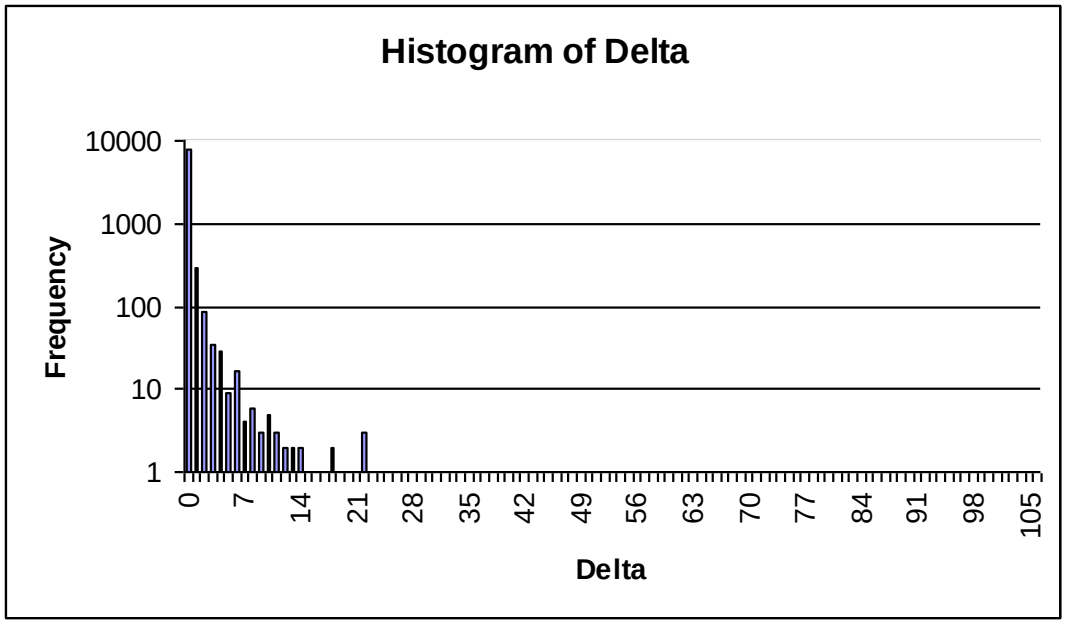

Figure 3.1: Semi-Logarithmic Histogram of the Delta Attribute

3.2.4. Correlation analysis

A correlation analysis of the predictor attributes vs. the class attribute is presented in Table 3.2 using Pearson's product-moment correlation coefficient.

As the reader will note, the correlations are surprisingly weak; the only exception is the CSI attribute, which is moderately correlated to Delta. In software defect prediction, one normally expects to find fairly high correlations between each attribute and the defect

attribute; that is, after all, what the original empirical validations of each metric were designed to demonstrate.

Table 3.2: Correlation of Software Metrics to Delta

CSA	0.0719
CSAO	0.113
CSI	0.5044
CSO	0.0977
DIT	0.1079
LCOM	0.0479
NAAC	0.1756
NAIC	0.0486
NOAC	0.073
NOCC	0.0498
NOIC	0.0575
NOOC	0.0985
NPavgC	0.0442
OSavg	0.0363
PA	0.0685
PPPC	-0.0442
RFC	0.118
WMC	0.2001

In Table 3.3, pairwise correlations between all of the predictor attributes (again using Pearson's coefficient) are presented, in order to determine if multicollinearity is present in the dataset. As can be seen, the picture is mixed; some attributes correlate strongly with each other, while others correlate very weakly. We only report *r*-values when the corresponding *p*-values are less than 0.01 (i.e. r-values are significant at α=0.01); when this is not the case, we merely report this fact. From Table 3.3, we conclude that multicollinearity is indeed present in this dataset, but it is not universal.

Table 3.3: Pairwise correlations between software metrics

	CSA	CSAO	CSI	CSO	DIT	LCOM	NAAC	NAIC	NOAC	NOCC	NOIC	NOOC	NPavgC	OSavg	PA	PPPC	RFC	WMC
CSA	1	0.8035	0.0185	0.5929	0.0593	0.3952	0.9897	*p>0.01*	0.579	*p>0.01*	0.0536	0.2798	0.1989	0.2687	0.5127	-0.135	0.4938	0.454
CSAO		1	0.1151	0.9555	0.1564	0.4717	0.7964	0.1019	0.8957	0.0327	0.1755	0.5314	0.3035	0.2817	0.5414	-0.0591	0.8246	0.7098
CSI			1	0.13	0.6792	*p > 0.01*	0.1079	0.3198	*p>0.01*	0.0489	0.5633	0.354	0.211	0.112	0.0737	*p>0.01*	0.1566	0.1871
CSO				1	0.1792	0.4429	0.5849	0.1234	0.9259	0.0442	0.2111	0.581	0.3127	0.2483	0.479	*p>0.01*	0.8713	0.7325
DIT					1	*p > 0.01*	0.0754	*0.452*	0.0524	0.0291	0.7791	0.3528	0.2743	0.151	0.092	*p>0.01*	0.2085	0.1684
LCOM						1	0.3916	*p>0.01*	0.388	*p>0.01*	0.0488	0.3048	0.0406	0.0778	0.3542	*p>0.01*	0.4811	0.3961
NAAC							1	0.0309	0.5721	*p>0.01*	0.0505	0.2764	0.1959	0.2659	0.5083	-0.1412	0.4926	0.4681
NAIC								1	0.0367	*p>0.01*	0.5367	0.2374	0.1269	0.0418	0.0493	0.0392	0.1101	0.0783
NOAC									1	0.0323	0.0719	0.2339	0.2743	0.2327	0.3767	*p>0.01*	0.7939	0.6778
NOCC										1	0.0323	0.0447	0.0436	*p>0.01*	*p>0.01*	*p>0.01*	0.046	0.0401
NOIC											1	0.3868	0.1974	0.0991	0.1061	0.0466	0.2377	0.1605
NOOC												1	0.2184	0.138	0.4213	*p>0.01*	0.5374	0.4272
NPavgC													1	0.3605	0.1434	-0.0401	0.2801	0.2752
OSavg														1	0.2868	-0.0381	0.3159	0.5134
PA															1	0.0771	0.5315	0.541
PPPC																1	*p>0.01*	-0.053
RFC																	1	0.861
WMC																		1

3.2.5 Principal Components Analysis

A principal components analysis of the Mozilla-Delta dataset, using the eigenvalues of the covariance matrix, is presented next. The eigenvalues are plotted in Figure 3.2. The *n* eigenvectors corresponding to the *n* largest eigenvalues are selected as the principal components of the dataset using the Cattell scree test [68]: one looks for a sharp steepening of the curve in Figure 3.2, and treat this as a cutoff point. The Karhunen-Loeve transformation is then applied for feature reduction. Examination of Figure 3.2 indicates that eigenvectors 14 through 18 should be selected as the orthonormal basis for reducing the dataset; these five eigenvectors are presented in Table 3.4.

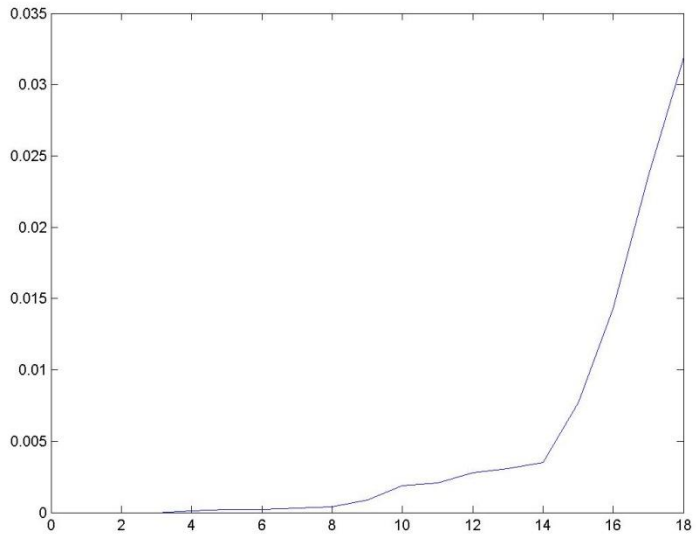

Figure 3.2 Eigenvalues of the Covariance Matrix

The ordering of eigenvectors in Table 3.4 is the same as the eigenvalues in Figure 3.2; the eigenvector covering the largest variance is PC5, with the amount of

variance covered dropping monotonically from right to left. From Table 3.4, one observes some interesting behaviors in the dataset when examining the factor loadings [69] (the advice in [70] to interpret loadings greater than 0.4 as being "strong" is followed here). Firstly, PC5 is heavily loaded on the PPPC attribute; recall that this attribute also exhibited unique behaviors in its statistical moments. PC4 is most strongly loaded on NOIC, with cross-loading on DIT; alignment with the other attributes is weak. PC3 extensively cross-loaded; the highest loadings are on NOIC, CSA, NAAC, and CSAO, but PC3 is not strongly loaded on any single metric. PC2 is strongly loaded on NPavgC, and PC1 is strongly loaded on OSavg. What this tells us is that, for the Mozilla-Delta dataset, PPPC, NPavgC and OSavg are very nearly orthogonal to one another, while NOIC and DIT are also somewhat independent of these three.

Table 3.4. Eigenvectors of the Covariance Matrix

Attribute	**PC1**	**PC2**	**PC3**	**PC4**	**PC5**
CSA	0.055489	-0.21135	0.349292	0.156546	0.070595
CSAO	-0.11728	-0.17425	0.329445	0.204015	0.040736
CSI	0.012005	0.001788	-0.0395	0.064328	0.00165
CSO	-0.16305	-0.1248	0.256807	0.18216	0.020671
DIT	0.11442	0.011404	-0.29214	0.445323	0.00596
LCOM	-0.02693	-0.06299	0.06726	0.032972	0.005841
NAAC	0.060699	-0.21355	0.348342	0.158285	0.073205
NAIC	-0.11	-0.06412	-0.17194	0.226948	-0.01703
NOAC	-0.12657	-0.1038	0.277501	0.13479	0.023233
NOCC	-0.00984	0.002878	0.000877	0.00434	0.000332
NOIC	0.026868	-0.16452	-0.44008	0.622703	-0.03453
NOOC	-0.18393	-0.11899	0.085896	0.219573	0.005799
NPavgC	-0.29091	0.856086	0.228383	0.31748	0.045387
OSavg	0.878041	0.228056	0.185782	0.147176	0.029306
PA	0.081914	-0.09466	0.15918	0.094145	-0.00702
PPPC	0.011142	0.007091	0.114503	0.039807	-0.99085
RFC	-0.04383	-0.09029	0.185706	0.148259	0.017187
WMC	0.108768	-0.03887	0.153319	0.107474	0.019749

3.3 Defect Prediction Experiments

In this section, we present the results of defect prediction experiments performed on both the original dataset and the dataset after reduction by the Karhunen-Loeve transform. The machine-learning task in defect prediction is to estimate the number of faults likely remaining in a code module. When this technique is employed in software test management, modules with a large number of predicted defects are selected for extra quality assurance effort (i.e. more testing and debugging resources will be allocated). The raw data from the experiment results of this chapter can be found on Appendix 1.

We employ statistical regression (linear and PACE regression), as well as machine learning algorithms (multilayer perceptrons, radial basis function networks, and SMO regression models) in modeling the original and reduced datasets. In general, we find that this dataset is difficult for prediction algorithms; our models all showed a very high root-mean-square (RMS) testing error in our experiments. All experiments are run in the WEKA data-mining environment, using a tenfold cross-validation design. All predictor attributes were normalized to [0,1], and there are no missing attribute values; the Delta attribute was not normalized.

The mean and standard deviation of the RMS errors for the best parameter selection we found for each model are presented in Table 3.5 for the original

dataset and in Table 3.6 for the reduced dataset. As can be seen, the average RMS errors in the tenfold cross-validation experiments are an order of magnitude greater than the average value of the Delta attribute itself (see Table 3.1). No method in Table 3.5 or 3.6 is statistically superior to any other, so we only analyze one of these results in detail. Ad-hoc analysis shows that SMO regression (which is based on support vector machines) produced a small practical improvement in the prediction results in the original dataset only; notice that the resulting RMS error was also less than that of any algorithm (including SMO regression) for the reduced dataset, while the standard deviation for SMO regression is less than half that for any other method in the original dataset.

Table 3.5. Regression on Full Dataset

Model	Mean RMS	Std. Dev. RMS
Linear Regression	1.3487	0.8495
PACE Regression	1.4608	1.6755
Radial Basis Function Network	1.1677	0.9412
Multilayer Perceptron	1.2242	0.9201
SMO Regression	1.0133	0.4079

Table 3.6 Regression on Reduced Dataset

Model	Mean RMS	Std. Dev. RMS
Linear Regression	1.1954	0.9213
PACE Regression	1.1954	0.9212
Radial Basis Function Network	1.1966	0.9067
Multilayer Perceptron	1.2000	0.9203
SMO Regression	1.1541	0.9966

We now investigate the SMO prediction errors (residuals) in detail. Table 3.7 (a) and (b) provides the r^2 (correlation coefficient) values for each partition of the

tenfold cross-validation experiments on the original and reduced datasets, respectively. r^2 is computed as the square of the Pearson's r between the actual and predicted values of the Delta attribute. Plainly, the prediction quality is generally poor, with one exception in the fifth partition of the full dataset.

Table 3.7. Actual vs. Predicted Deltas R^2

Partition	R^2
1	0.0007
2	0.0026
3	0.0144
4	0.0009
5	0.8904
6	0.00003
7	0.0547
8	0.0418
9	0.0118
10	0.0043

(a)

Partition	R^2
1	0.0008
2	0.0221
3	0.0002
4	0.0300
5	0.0504
6	0.0535
7	0.0005
8	0.0033
9	0.0036
10	0.0334

(b)

Recall from Section 3.2.3 that the maximum value of the Delta attribute was 104.0. In the SMO experiment on the full dataset, this record was assigned to the test set in partition 5, and was in fact predicted relatively well; the prediction for that record was 58.272 (residual = -45.728), meaning that this record was recognized as having a very high value. However, this was the exception rather than the rule. The predictions for virtually every record were close to 0, meaning that any record with a Delta value much different than 0 was seriously under-predicted (i.e. the residue was almost exactly the inverse of the Delta value, leading to a negative linear slope). The residue for the record with Delta=104 is the only one to break this pattern. As Pearson's coefficient is a product-moment

correlation, we suspect that this one reasonable prediction biased the entire correlation for this partition. All remaining partitions, in both the original and reduced datasets, show the same trend in the residues. This means that the SMO algorithm (the most accurate we found) was unable to recognize records with a Delta value differing from 0; as these are precisely the software modules a developer is interested in detecting, we must conclude that our defect-prediction experiments have been ineffective. Figure 3.3 shows the residual plot of delta versus residues for partition 5 of the full dataset.

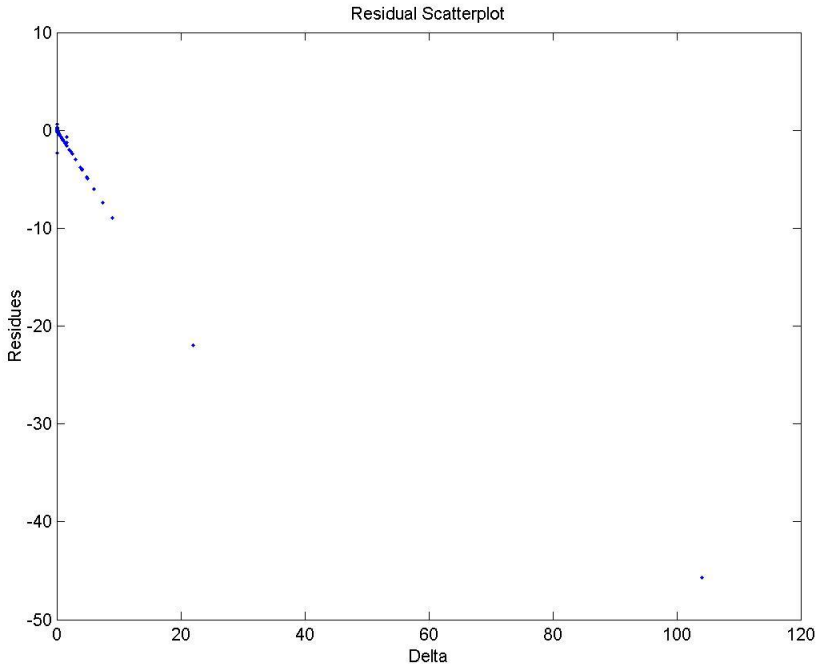

Figure 3.3. Delta vs. Residues, Full Dataset, Partition 5.

From the experiments, we could not use a threshold value of any metrics to identify modules with high Delta values. There was one exception to this finding, for the PPPC metric. Recall from Section 3.2.3 that the statistical moments of this

attribute are different from the other attributes, and that the eigenvector covering the greatest variance in this dataset loads very heavily on this attribute. We present a typical residue plot for this attribute in Figure 3.4, where we see a vertical linear structure that consumes most of the under-prediction errors. Importantly, this linear structure is found at high values of the PPPC attribute. It would thus seem reasonable to suggest that we could use a ranking on the PPPC value to identify high-risk software modules in this dataset with a reasonable degree of accuracy; modules in the top k%, ranked by the PPPC attribute, should be considered high-risk and receive additional quality-assurance effort.

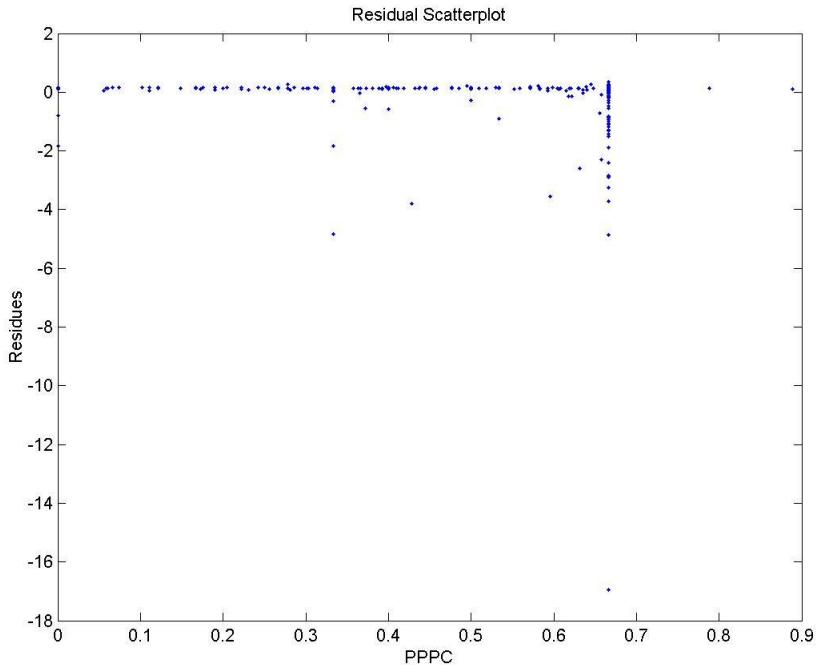

Figure 3.4 PPPC vs. Residuals, Full Dataset, Partition 10, SMO Regression

3.4 Fault-Proneness Experiments

In Section 3.3 we saw that treating defect prediction as a function approximation problem was generally ineffective, as every algorithm tended to simply predict a value of "0." We believe that this is due to a combination of heteroskedasticity and skewness in the data, and so in this section we attempt to reduce the effect of these problems in the dataset. We reduce the variance of the Delta attribute by converting it to a binary class (i.e. we convert the defect-prediction problem to a fault-proneness problem). We use stratification-based resampling on our fault-proneness experiments following [113, 142-144]; in particular we use uniform under-sampling and the SMOTE over-sampling algorithm [10]. SMOTE produces synthetic examples of the minority class, in order to avoid the tightly-focused decision regions that occur when minority-class samples are merely replicated. This combination of techniques should lead to better fault predictions.

The experiments were performed using several classification algorithms implemented in WEKA: Support Vector Machines, KStar instance-based classifier, RIPPER, Random Forests, Radial Basis Functions and decision trees. All experiments followed a tenfold cross-validation design, and we report the average and standard deviation of the statistics for the best parameterization of each algorithm. We first present the results of fault-proneness modeling without stratification in Table 3.8. We include both the accuracy and Kappa statistic to demonstrate how misleading classification accuracy is in this case. Every method

achieves 92-94% accuracy, but this is roughly the fraction of the dataset belonging to the majority class! The minority class is not being predicted well at all; this fact shows up clearly in the Kappa statistic, which is not greater than 0.1233 for any method. The skewness in the dataset is severely biasing the classifiers, and so we will next proceed with our stratification experiments.

Table 3.8. Results for original distribution

Algorithm	Accuracy	Std Dev (accuracy)	Kappa	Std Dev (Kappa)
J48 Decision Tree	94.01	0.0548	0.0066	0.0145
J48 Decision Tree (unpruned)	93.23	0.4623	0.0791	0.0468
RIPPER	93.89	0.2319	0.0041	0.0130
KStar	92.13	0.5429	0.1233	0.0559
Radial Basis Function Nets	94.02	0.0356	0.0000	0.0000
Random Forests	93.36	0.4122	0.1022	0.0656
Support Vector Machines	94.02	0.0356	0.0000	0.0000

We use different rates of over and under-sampling to modify the distribution of the classes in the training dataset of each fold in our cross-validation. We apply 30 resampling strategies, each one being a class-by-class specification of the degree of under-sampling or over-sampling to be applied. The rates of over-sampling are 0, 100, 200, 300, 400 and 500% representing the number of synthetic minority examples (as a percentage of the original minority class size) added to the training set. For under-sampling we remove 0, 20, 40, 60 and 80% of the examples from the majority class in the training set. In all cases, the test sets are unchanged (the original dataset is partitioned into training and testing sets prior to stratification; it is re-partitioned and re-stratified after each experiment). We report the average and standard deviation of the Kappa statistic on the

(unchanged) test data set for each strategy on the best parameterization of that algorithm in Tables 3.9 - 3.15. The highest Kappa statistic value obtained is highlighted in bold and underlined.

Table 3.9. J48 Decision trees

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.0066 ±0.015	0.0066 ±0.015	0.0723 ±0.022	0.0896 ±0.063	0.0897 ±0.052	0.1090 ±0.102
	20	0.0277 ±0.031	0.0149 ±0.019	0.1019 ±0.047	0.1254 ±0.059	0.1119 ±0.059	**0.1307** ±0.061
	40	0.0285 ±0.031	0.0270 ±0.028	0.0965 ±0.055	0.1018 ±0.053	0.1139 ±0.048	0.1216 ±0.067
	60	0.0683 ±0.051	0.0375 ±0.038	0.1212 ±0.049	0.1028 ±0.053	0.1106 ±0.046	0.1204 ±0.038
	80	0.1233 ±0.038	0.0996 ±0.052	0.1023 ±0.026	0.0948 ±0.027	0.0992 ±0.029	0.0966 ±0.022

Table 3.10. J48 Decision trees (unpruned)

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.0791 ±0.047	0.0791 ±0.047	0.0805 ±0.033	0.1073 ±0.062	0.1060 ±0.053	0.1148 ±0.082
	20	0.0986 ±0.053	0.0714 ±0.046	0.1016 ±0.056	0.1145 ±0.063	0.1168 ±0.057	0.1323 ±0.062
	40	0.0727 ±0.033	0.0909 ±0.044	0.1020 ±0.066	0.1095 ±0.049	0.1173 ±0.055	0.1311 ±0.068
	60	0.1097 ±0.044	0.0889 ±0.057	**0.1324** ±0.044	0.1032 ±0.059	0.1119 ±0.046	0.1222 ±0.032
	80	0.1102 ±0.049	0.1008 ±0.048	0.1023 ±0.025	0.0944 ±0.026	0.0966 ±0.031	0.1000 ±0.022

Table 3.11. RIPPER

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.0041 ±0.013	0.0131 ±0.024	0.0431 ±0.047	0.0270 ±0.028	0.0351 ±0.030	0.0232 ±0.019
	20	0.0152 ±0.036	0.0039 ±0.016	0.0333 ±0.038	0.0469 ±0.043	0.0462 ±0.071	0.0353 ±0.027
	40	0.0153 ±0.030	0.0037 ±0.015	0.0601 ±0.035	0.0820 ±0.055	0.0417 ±0.036	0.0825 ±0.063
	60	0.0279 ±0.040	0.0375 ±0.047	0.1027 ±0.044	**0.1129** ±0.060	0.0930 ±0.031	0.0883 ±0.037
	80	0.0743 ±0.052	0.1039 ±0.058	0.0999 ±0.028	0.0815 ±0.018	0.0819 ±0.015	0.0831 ±0.017

Table 3.12. KStar

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.1258 ±0.049	0.1258 ±0.049	0.1382 ±0.059	0.1352 ±0.053	0.1373 ±0.063	0.1381 ±0.053
	20	0.1267 ±0.049	0.1334 ±0.052	0.1488 ±0.063	0.1417 ±0.057	0.1439 ±0.054	0.1468 ±0.057
	40	0.1333 ±0.044	0.1332 ±0.056	0.1322 ±0.038	0.1327 ±0.065	0.1517 ±0.047	0.1433 ±0.052
	60	0.1512 ±0.053	**0.1648** ±0.057	0.1389 ±0.054	0.1341 ±0.047	0.1217 ±0.039	0.1149 ±0.044
	80	0.1348 ±0.062	0.1247 ±0.037	0.1067 ±0.033	0.1093 ±0.026	0.1013 ±0.029	0.1017 ±0.023

Table 3.13. Radial Basis Function Networks

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.0000 ±0.000	0.0000 ±0.000	0.0000 ±0.000	0.0427 ±0.062	0.0280 ±0.027	0.0515 ±0.055
	20	0.0000 ±0.000	0.0000 ±0.000	0.0033 ±0.010	0.0274 ±0.055	0.0327 ±0.048	**0.0858** ±0.074
	40	0.0000 ±0.000	0.0000 ±0.000	0.0513 ±0.085	0.0489 ±0.076	0.0660 ±0.060	0.0677 ±0.068
	60	0.0000 ±0.000	0.0000 ±0.000	0.0746 ±0.075	0.0518 ±0.052	0.0642 ±0.038	0.0649 ±0.025
	80	0.0273 ±0.052	0.0371 ±0.054	0.0658 ±0.033	0.0583 ±0.054	0.0715 ±0.035	0.0701 ±0.023

Table 3.14. Random Forests

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.0939 ±0.062	0.0916 ±0.066	0.1132 ±0.089	0.1148 ±0.045	0.0946 ±0.047	0.1296 ±0.076
	20	0.1151 ±0.059	0.1029 ±0.068	0.1205 ±0.059	0.1283 ±0.080	0.1057 ±0.071	0.140 ±0.085
	40	0.0907 ±0.049	0.1281 ±0.081	0.1286 ±0.086	0.1207 ±0.058	0.1148 ±0.076	0.1390 ±0.087
	60	0.1287 ±0.100	0.1358 ±0.060	0.1318 ±0.072	**0.1468** ±0.036	0.1285 ±0.048	0.1372 ±0.052
	80	0.1335 ±0.057	0.1445 ±0.034	0.1128 ±0.033	0.1356 ±0.029	0.1208 ±0.037	0.1197 ±0.034

Table 3.15. Support Vector Machines

		Over-sampling levels					
		0	100	200	300	400	500
Under-sampling levels	0	0.1088 ±0.038	0.1088 ±0.038	0.1243 ±0.051	0.1217 ±0.065	0.1421 ±0.043	0.1269 ±0.036
	20	0.1150 ±0.052	0.1233 ±0.043	**0.1514** ±0.047	0.1367 ±0.037	0.1200 ±0.035	0.1331 ±0.048
	40	0.0999 ±0.046	0.1188 ±0.064	0.1192 ±0.041	0.1137 ±0.050	0.1293 ±0.033	0.1454 ±0.047
	60	0.1222 ±0.036	0.1237 ±0.048	0.1356 ±0.046	0.1298 ±0.027	0.1136 ±0.028	0.1222 ±0.028
	80	0.1409 ±0.063	0.1303 ±0.038	0.1104 ±0.021	0.1067 ±0.016	0.1004 ±0.014	0.1117 ±0.017

In Tables 3.9-3.15, we are always able to improve the classification results with some combination of under- and over-sampling. However, the "best resampling" strategy changes with each algorithm, as does the relative improvement. KStar still produces the most balanced classifications, but is still far from the theoretical limit of 1.0 (for a perfect classification). We include sample confusion matrices from the KStar algorithm. Table 3.16 corresponds to one of the folds of the original dataset and Table 3.17 shows the confusion matrix in the same fold for

the resampling strategy Over-sampling = 100, Under-sampling = 60. Positive instances correspond to Delta=1 while negative instances have Delta=0. When dealing with imbalanced datasets there is always a tradeoff between accuracy in the minority and majority class. We want to increase the accuracy in the minority class without sacrificing too much accuracy in the majority class. In Table 3.17 we increase the number of correctly classified minority samples while the majority-class accuracy (true positive rate) remains above 92%. This is a hard task to achieve when the minority class only covers 5% of the dataset.

Table 3.16. KStar original dataset. Kappa=0.1398

	Predicted	
Actual	TN=755	FP=30
	FN=41	TP=8

Table 3.17. KStar resampled dataset. Kappa = 0.1792

	Predicted	
Actual	TN=723	FP=62
	FN=34	TP=15

3.5 Related work

Nonparametric techniques, especially machine learning and computational intelligence, have been employed for software defect prediction. Some examples include neural networks [145-151], genetic-fuzzy systems [152], neuro-fuzzy systems [153, 154], fuzzy classifiers [155, 156], genetic programming [144, 157, 158], fuzzy clustering [13, 129, 141, 159], k-means clustering [160], classification trees [161-163], support vector machines and their hybrids [164-166], ant colony

optimization [167] and classifier ensembles [168] using various defect-prediction datasets.

A recent benchmarking study of three machine-learning classifiers found that 71% of defective modules were detected, while the false alarm rate was 25% [169]. Predictive models are usually tested using the cross-validation experimental design, in which the dataset is partitioned into *k* equal subsets. The learning algorithm is trained using *k*-1 of these subsets, with the remaining one used as the holdout sample for estimating the accuracy of the resulting classifier. This process repeats *k* times, with each of the subsets being used exactly once as the holdout sample. For a detailed review of research in this area, see [170].

3.5.1 Cost-sensitive Classification in SDP

As previously discussed, the cost of misclassifying a positive example is often higher than classifying a negative example as positive. For example, classifying a fault-prone software module (minority class) as a fault-free module (majority class) could imply assigning fewer test resources to it – and then having the module fail (far more expensively) in field usage. On the other hand, classifying a fault-free module as faulty may imply expending more resources during the testing phase, but the module would be less likely to experience a failure in the field. Cost-sensitive classifiers explicitly consider these differential costs, and will minimize the total expected cost of errors, rather than just the number of errors as in most classifiers.

Cost-sensitive classification has been applied to SDP since 1999 [162]. Recently, Khoshgoftaar et al. [171] discuss the use of cost-sensitive boosting [56] for SDP. Ling et al. [172] proposed a system to predict the escalation risk of current defect reports for maximum return on investment (ROI), based on mining historic defect report data from an online repository. ROI was computed by estimating the cost of not correcting an escalated defect (false negative) to be seven times the cost of correcting a non-escalated defect (false positive). However, in a recent comparison on 15 benchmark datasets (including 7 SDP datasets), cost-sensitive classification was found to be inferior to random undersampling (discussed in the next subsection) when the cost ratios were high [173].

3.5.2 Stratification in SDP

Researchers have only recently investigated the use of stratification in SDP. Kaminsky and Boetticher [144] utilize genetic programming for software defect prediction. To compensate for data skewness they apply "equalized learning"; they replicate training instances in the minority class so that the training data will contain an equal distribution of instances (a simple oversampling technique, criticized in [10]), and they finally shuffle the dataset. Replication was also recently examined in [11], and found to be ineffective. SMOTE was first applied to SDP in [13], where the authors utilize three software metrics datasets [174-176], treating SDP as a multiclass problem based on clustering. They used random undersampling and SMOTE oversampling to calibrate a decision tree to

perform best on the classes of interest. Pelayo and Dick [113] also used uniform undersampling and SMOTE oversampling for SDP. Measuring performance by the geometric mean of the individual class accuracies (which is more sensitive to poor performance in a minority class than classification accuracy), this resampling approach led to an improvement of at least 23% in each of four datasets. Seiffert et al. [177] compare five stratification techniques and a boosting algorithm (AdaBoost) on 15 SDP datasets. They found that random undersampling was almost always (for 12 of 15 datasets) the best stratification technique (no other technique performed better on more than two single datasets), although boosting was superior. In [178] the authors present a critical review of the nature of the problem and the different strategies applied in the imbalanced learning scenario.

3.6 Conclusions

We have reported a case study of software defect prediction in the Mozilla dataset which is, to the best of the authors' knowledge, unique among publicly-available datasets in its combination of size, use of object-oriented metrics, and incorporation of a defect measure. Heteroskedasticity and skewness appear to be significant problems within this dataset, as with other defect-prediction datasets. We have pursued fault-proneness studies in this dataset with the aid of stratification-based resampling. This has led to improved predictions, as measured by the Kappa statistic. As usual in resampling studies, this means that an

improvement in the minority-class prediction was achieved at the cost of increased errors for the majority class. However, in the software domain, the price of allowing faults to slip through into field usage (i.e. the cost of misclassifying a minority-class example) is so high that we believe this tradeoff is advantageous.

This case study supports the use of stratification in SDP, especially SMOTE over-sampling and uniform under-sampling. And it does so on the best dataset currently available, one which better reflects modern software development, and which has furthermore not been previously analyzed. Thus, the ultimate outcome of this chapter is to validate our focus on SMOTE in Chapter 4, which also focuses on SDP.

Chapter 4. Evaluating Stratification Alternatives to Improve Software Defect Prediction

4.1 Introduction

In this chapter we address the question of what stratification technique would work best when dealing with imbalanced datasets: over-sampling, under-sampling or a combination of both. More specifically, the question is what stratification alternatives work best across *many* datasets; it is expected that *every* technique works well for *some* datasets. The goal of this chapter is to provide empirical evidence showing which alternative (under-sampling, over-sampling, or both) is *generally* (across multiple datasets) most effective in the software defect prediction (SDP) problem.

We employ the Analysis of Variance (ANOVA) to determine the relative contribution of under-sampling, over-sampling, and the interaction between the two on a number of SDP datasets. We focus on uniform under-sampling without replacement, the well-known SMOTE over-sampling technique [10], and a more recent variant of SMOTE [52]. Uniform under-sampling has been repeatedly identified as a very effective method for stratification in many domains [11, 179], while the SMOTE technique and its variants have been widely used and studied. Furthermore, SMOTE proved to be quite effective in our case study of SDP in Chapter 3. Using a collection of modern SDP datasets, we test the performance of

a single, standardized classification algorithm (c4.5 decision trees) when we consider five levels of under-sampling and over-sampling. To determine the prediction quality for each cell of our ANOVA design, we employ the tenfold cross-validation approach (yielding ten replicates per cell). The ANOVA results show that under-sampling, and the interaction between under- and over-sampling are significant on these datasets; however, the over-sampling techniques themselves are not. Tukey's Honestly Significant Difference test supports these results. Based on these findings, we suggest that both under- and over-sampling have value in software defect prediction; the software quality analyst using machine learning appears to be best served by employing both approaches.

4.2 Experimental Design and Methodology

4.2.1 Methods

Our research question in this chapter asks whether under-sampling, over-sampling, or a combination of both would be most effective for software defect prediction. We can cast this question in terms of statistical experimental design as follows: a treatment will consist of modifying a defect-prediction training dataset using a certain level of under-sampling, and a certain level of over-sampling, followed by the training and testing of a known (constant) classifier on that dataset (the holdout sample is not resampled). Under-sampling and over-sampling thus become the factors in this design. Our research question then translates to inquiring if the mean responses of a classifier to the factors of under-sampling or over-sampling, or their interaction, are significantly different. This question is

precisely what the Analysis of Variance (ANOVA) is intended to address [90]. We will employ ANOVA across a set of seven software defect-prediction datasets, employing the common tenfold cross-validation design in each cell; this will provide us with ten replicates per cell for determining the experimental error. We use the SMOTE implementation from [180], which employs K-D trees [181] instead of a brute-force neighbor search as in [10]. Other possibilities for a neighbor search include a box-assisted algorithm (also implemented in [180]), but the performance of a box-assisted search will be strongly influenced by the subspace selected for the search. K-D trees avoid this additional search of a parameter space and were still an order of magnitude faster in timing experiments than a brute-force search.

There is one significant complication in our experimental design: as with all pattern-recognition approaches, the outcomes of our experiments are expected to be dataset-dependent. Specifically, the relative importance of under-sampling and over-sampling in improving classifier performance is expected to vary from one dataset to another, and is not predictable *a priori*. This is again a natural consequence of the well-known lack of a theory of pattern recognition [182]. Hence, we must accommodate this heterogeneity by employing blocking. Our experimental design is thus a blocked full-factorial experiment, where the blocks correspond to the datasets. We have seven different datasets, and will employ five levels for under-sampling and five levels for over-sampling. We select the over-sampling levels 0,100, 200, 300 and 400 (the percentage of synthetic minority

samples added to the training dataset). We also select the under-sampling levels 0, 20, 40, 60 and 80%, which are the fraction of majority class samples removed in the resampled dataset. These levels were determined from [113], where they were quite effective in improving SDP performance. The linear model for this design is

$$y_{ijkl} = \mu + \tau_{ik} + \beta_{jk} + (\tau\beta)_{ijk} + \varepsilon_{ijkl} \tag{4.1}$$

$$\begin{cases} i = 0,20,40,60,80 \\ j = 0,100,200,300,400 \\ k = CM1, KC1, KC1class-level, KC2, JM1, PC1, Mozilla \\ l = 1,2..10 \end{cases}$$

where τ_{ik} is the effect of the *i*-th under-sampling level within the *k*-th dataset, β_{jk} is the effect of the *j*-th over-sampling level within the *k*-th dataset, $(\tau\beta)_{ijk}$ is the interaction of the *i*-th under-sampling level with the *j*-th over-sampling level within the *k*-th dataset, and ε_{ijkl} is the error term. The last index l represents the replications, arising from the ten fold cross-validation procedure. We will use the lack of fit test from [183, 184] to determine if this model is appropriate for our experiments. This test compares the residual error to the pure error from replicated design points, again using the F-test. The F-statistic is calculated by dividing the mean square error due to lack-of-fit by the mean square error due to replicate variation. If there is significant lack of fit, the model should not be used as a predictor of the response. The actual *p*-values for the lack of fit of our experiments are shown together with the analysis of variance results later in Section 4.3.

We note that the reliability of the cross-validation design in general was questioned in [185], on the basis that two different models induced from the same dataset could not be homogeneously ordered across every fold. The authors generate 1000 samples from a theoretical (logarithmic) distribution fitted to a dataset relating function points to man-hours on software projects (the so-called "Finnish" dataset). A linear regression model and an Estimation-by-Analogy (EBA) model are then fitted using 1000-fold cross-validation (also known as the leave-one-out test). The authors determined that, across the 1000 test samples, neither the linear regression nor the EBA models were consistently superior, irrespective of the exact performance measure used. However, this rank-based analysis is questionable at best, as prediction errors on each fold were converted to ranks without first determining if the difference between the prediction errors for that fold was in fact statistically significant.

For this experiment our dependent variable is Cohen's kappa statistic [186], computed from the out-of-sample confusion matrix (contingency table). In our experiment we compute the kappa statistic between the outputs of a classifier and the correct classification for each input, yielding the chance-corrected accuracy of the classifier.

Other studies [46, 177, 187] have examined the relative importance of different under-sampling and over-sampling techniques. However, it is important to note that, while several of these studies tested for statistical significance using

ANOVA and Tukey's test, none of them used a factorial experiment design between under- and over-sampling, making it impossible to quantify the significance of any interaction between them.

In [188] the authors present a study comparing boosting and bagging techniques with three different classifier algorithms on four datasets with eight bagging and boosting versions and seven different performance measures using ten-fold cross validation. They use two bagging techniques and combine boosting with two resampling techniques (SMOTE over-sampling and random under-sampling). Multiple classifiers are used, and the learning algorithm is considered a factor in their experimental design. The boosting/bagging technique is a second factor, as are class distribution, noise distribution and noise level. All five main factors, as well as all two-and three-way interactions, were significant in an ANOVA analysis. However, in their study there is no indication whether the model is linear, quadratic or a different order; it is assumed no investigation was made regarding the possibility of any curvature in the model and they adopt the linear model. There is thus no assurance whether their model is adequate to deconfound the effects of resampling and the effect of the learning algorithm. In addition, there is no indication of any blocking strategy used to address the heterogeneity between the experimental datasets.

4.2.2. Datasets

Firstly, we employ the Mozilla dataset described in Chapter 3, specifically the fault-proneness version (binary classification problem). Several SDP datasets created by the NASA Metrics Data Program are archived at the PROMISE repository [130]. We use the datasets denoted KC1, KC2, CM1, JM1 and PC1 in our experiments. These datasets can be described as quite large and skewed, but still having fifty or more examples of the minority class (meaning at least five minority samples in each test set), and low apparent noise (several PROMISE datasets have an inordinate number of "zero-size" modules). The project KC1 was extracted from includes 43,000 lines of C++ code distributed into 2109 modules. KC2 covers over 43,000 lines of C++ code in 379 functions. CM1 covers 20,000 lines of C code in 498 functions. JM1 is a real-time system written in C, containing approximately 315,000 lines of code in 10878 modules. PC1 covers 40,000 lines of C code in 941 functions. These datasets contain twenty-one software product metrics covering the product's size, complexity [71] and vocabulary [72] as follows:

- McCabe's line count of code
- McCabe cyclomatic complexity
- McCabe essential complexity
- McCabe design complexity
- Halstead total operators + operands
- Halstead volume
- Halstead program length

- Halstead difficulty
- Halstead intelligence
- Halstead effort
- Halstead
- Halstead's time estimator
- Halstead's line count
- Halstead's count of lines of comments
- Halstead's count of blank lines
- count of lines of code and comments
- unique operators
- unique operands
- total operators
- total operands
- count of the flow graph

Definitions for all of these metrics are provided in [189]. The KC1 dataset is based on an object oriented program, and so a version of the KC1 dataset is available that includes ten object-oriented metrics (including the CK metrics) as well as the older procedural-programming metrics above. Method-level metrics were aggregated within a class using the arithmetic sum. The metrics in the KC1 Class-level dataset are listed below; see [189] for definitions.

- PERCENT_PUB_DATA
- ACCESS_TO_PUB_DATA

- COUPLING_BETWEEN_OBJECTS
- DEPTH
- LACK_OF_COHESION_OF_METHODS
- NUM_OF_CHILDREN
- DEP_ON_CHILD
- FAN_IN
- RESPONSE_FOR_CLASS
- WEIGHTED_METHODS_PER_CLASS
- sumLOC_BLANK
- sumBRANCH_COUNT
- sumLOC_CODE_AND_COMMENT
- sumLOC_COMMENTS
- sumCYCLOMATIC_COMPLEXITY
- sumDESIGN_COMPLEXITY
- sumESSENTIAL_COMPLEXITY
- sumLOC_EXECUTABLE
- sumHALSTEAD_CONTENT
- sumHALSTEAD_DIFFICULTY
- sumHALSTEAD_EFFORT
- sumHALSTEAD_ERROR_EST
- sumHALSTEAD_LENGTH
- sumHALSTEAD_LEVEL
- sumHALSTEAD_PROG_TIME

- sumHALSTEAD_VOLUME
- sumNUM_OPERANDS
- sumNUM_OPERATORS
- sumNUM_UNIQUE_OPERANDS
- sumNUM_UNIQUE_OPERATORS
- sumLOC_TOTAL

4.3 Analysis of Variance Results

Our stratification experiments combine uniform under-sampling with the original SMOTE algorithm and one of its most recent variations (SMOTE-SN). The raw data for the experiments of this chapter can be found on Appendx 2. Our experiments are divided into four groups as explained further. In order to check for the normality assumption, we use the Kolmogorov-Smirnov (K-S) normality test. The results of the p-value for the K-S test are presented in table 4.1

Table 4.1. Kolmogorov-Smirnov normality test results

Datasets	SMOTE	SMOTE-SN
CM1	0.05	>0.15
KC1	>0.15	>0.15
KC2	0.07	0.09
KC1 class-level	<0.01	0.04
JM1	>0.15	0.04
PC1	0.08	0.15
Mozilla	>0.15	>0.15
CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla	<0.01	< 0.01
CM1, KC1, KC2, JM1, PC1, KC1-Class	<0.01	< 0.01
CM1, KC1, KC2, JM1, PC1	<0.01	< 0.01
KC1-Class, Mozilla	<0.01	< 0.01

As discussed in Chapter 2, while normality is an assumption in ANOVA, violations of normality often do not lead to a material difference in the analysis outcome. Specifically, in a large sample, the type-1 error rate will be essentially unchanged even if normality is violated. In our experiments, we have 5 levels for each factor, yielding 25 cells; with 10 replicates per cell, we thus have a total of 250 samples for each individual dataset. Our largest grouping, consisting of all 7 datasets, thus has 1750 samples in total.

We check the homogeneity of variance assumptions for ANOVA and the Tukey test using Levene's test as this test is not sensitive to the normality assumption. The results of Levene's test applied to each dataset and the different groups of datasets used in our experiments are displayed in Table 4.2.

Table 4.2. Levene's test P-values

Datasets	SMOTE	SMOTE-SN
CM1	0.221	0.507
KC1	0.743	0.527
KC2	0.905	0.732
KC1 class-level	0.834	0.896
JM1	0.096	0.143
PC1	0.447	0.531
Mozilla	0.103	0.156
CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla	< 0.001	< 0.001
CM1, KC1, KC2, JM1, PC1, KC1-Class	< 0.001	< 0.001
CM1, KC1, KC2, JM1, PC1	< 0.001	< 0.001
KC1-Class, Mozilla	< 0.001	< 0.001

As we observe in Table 4.2, all the individual datasets are homoskedastic. Since the Tukey tests shown in section 4.3.1 are conducted on individual datasets, they are valid. However, when combining the datasets, the homogeneity of variance is

no longer valid. To correct for heteroskedasticity, we follow Piepho's example [96] to adjust the degrees of freedom (*df*) by Box's method. According to the procedure, we first perform the ANOVA and then we use Box's method to correct for heteroskedasticity. We will first present our ANOVA tables followed by the results after adjusting the *df*. Our Tukey test results are shown after correcting for heteroskedasticity.

4.3.1. Main results

We first present in Table 4.3 the set of results using the original SMOTE followed by the results using SMOTE-SN in Table 4.4. We have carried out our factorial experiments with blocking across all seven of our datasets (KC1, KC2, CM1, JM1, PC1, KC1-Class, Mozilla). The results are analyzed using the two-way ANOVA procedure based on the model of Eq. (4.1). For both SMOTE and SMOTE-SN, under-sampling the datasets significantly affects the classification results. Surprisingly, the main effect of over-sampling by itself is not significant; however the interaction between over-sampling and under-sampling is significant at α=0.05. In each ANOVA table the lack of fit and pure error values are displayed immediately after showing the results for our factors. In all experiments, the lack-of-fit statistic for the linear model is insignificant at α=0.05.

To correct for heteroskedasticy, we follow Piepho's example to adjust the ANOVA *df* by Box's method. According to the procedure, only factors that are significant with the original ANOVA (using the standard *df*) will be considered

for the correction, as adjusting the *df* will not make the factor to be significant. We present after each ANOVA table with SMOTE and SMOTE-SN those factors that were significant on our ANOVA experiments and the original and adjusted *df*, as well as the *p*-value of the F-test using the adjusted *df*. Note that since we use the same number of levels and replications for SMOTE and SMOTE-SN, the number of *df* on such tables are the same.

Table 4.3: ANOVA results for CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla with SMOTE

Source	SS	df	Mean Square	F Value	P-value
Dataset	18.985	6	3.164	189.888	< 0.001
Under	0.428	4	0.428	25.688	< 0.001
Over	0.005	4	0.005	0.291	0.884
Under*Over	0.208	16	0.208	12.480	< 0.001
Lack of Fit	3.005	144	0.018	1.133	0.144
Pure Error	25.641	1575	0.016		

Table 4.4: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla with SMOTE

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	6	1719	1	286	< 0.001
Under	4	1719	1	430	< 0.001
Under*Over	16	1719	1	107	< 0.001

Table 4.5: ANOVA results for CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla with SMOTE-SN

Source	SS	df	Mean Square	F Value	P-value
Dataset	21.008	6	3.501	228.667	< 0.001
Under	0.447	4	0.447	29.170	< 0.001
Over	0.007	4	0.007	0.467	0.494
Under*Over	0.183	16	0.183	11.952	< 0.001
Lack of fit	2.943	144	0.018	1.155	0.113
Pure Error	23.699	1575	0.015		

Table 4.6: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, KC1-Class, Mozilla with SMOTE-SN

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	6	1719	1	286	< 0.001
Under	4	1719	1	430	< 0.001
Under*Over	16	1719	1	107	< 0.001

The *p*-values after Box's correction indicate that the heteroskedasticity between our datasets does not impact our findings. We thus proceed with the Tukey honestly significant difference (HSD) test to quantify the difference in means shown by our ANOVA results.

We use Tukey's HSD test [190] to determine whether under-sampling, over-sampling and/or their interaction are statistically significant in improving the classification of individual datasets. For each dataset, we compare the response for the original dataset against the responses for pure under-sampling; against the responses for pure over-sampling; and the responses for all resampling strategies.

We again present our results for the original SMOTE followed by the comparison using SMOTE-SN. The *p*-values of the Tukey tests are shown in Tables 4.7 and 4.8. We can see the individual datasets benefit differently from different resampling techniques, as expected.

Table 4.7: *p*-values for Tukey tests on individual datasets with SMOTE

Dataset	Under-sampling	Over-sampling	Interaction
CM1	0.516	0.086	0.042
KC1	< 0.001	0.031	0.014
KC2	< 0.001	0.048	0.001
KC1-Class	0.336	0.228	0.152
JM1	< 0.001	0.504	< 0.001
PC1	0.054	0.760	0.934
Mozilla	0.234	< 0.001	< 0.001

Table 4.8: *p*-values for Tukey tests on individual datasets with SMOTE-SN

Dataset	Under-sampling	Over-sampling	Interaction
CM1	0.619	0.007	0.525
KC1	< 0.001	0.671	0.025
KC2	0.001	0.028	0.001
KC1-Class	0.104	0.464	0.472
JM1	0.153	0.315	< 0.001
PC1	0.291	0.956	0.357
Mozilla	0.163	< 0.001	< 0.001

Considering the differences observed in the individual datasets, we next undertake a sensitivity analysis by removing selected datasets. The first to be removed is the Mozilla dataset; this dataset is unique in that it is significantly affected by both over-sampling and the under-over interaction – and yet was *not* significantly

affected by under-sampling. Furthermore, this dataset was not sourced from the NASA MDP program, while the remaining datasets have achieved "benchmark" status within the SDP research community. Our ANOVA analysis with the Mozilla dataset removed is presented in Tables 4.9 and 4.11, and again shows that under-sampling is significant at α=0.05, the main effect of over-sampling is not significant, and the interaction between over-sampling and under-sampling is significant.

Table 4.9. ANOVA results for CM1, KC1, KC2, JM1, PC1, KC1-Class with SMOTE

Source	SS	df	Mean Square	F Value	P-value
Dataset	11.534	5	2.307	121.964	< 0.001
Under	0.560	4	0.560	29.623	< 0.001
Over	0.029	4	0.029	1.523	0.193
Under*Over	0.192	16	0.192	10.168	< 0.001
Lack of fit	2.988	120	0.021	1.135	0.160
Pure Error	25.212	1350	0.019		

Table 4.10: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, KC1-Class with SMOTE

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	5	1470	1	294	< 0.001
Under	4	1470	1	368	< 0.001
Under*Over	16	1470	1	92	< 0.001

Table 4.11. ANOVA results for CM1, KC1, KC2, JM1, PC1, KC1-Class with SMOTE-SN

Source	SS	df	Mean Square	F Value	P-value
Dataset	13.015	5	2.603	150.692	< 0.001
Under	0.586	4	0.586	33.918	< 0.001
Over	0.002	4	0.002	0.105	0.981
Under*Over	0.149	16	0.149	8.631	< 0.001
Lack of fit	2.617	120	0.019	1.083	0.263
Pure Error	23.138	1350	0.017		

Table 4.12: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, KC1-Class with SMOTE-SN

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	5	1470	1	294	< 0.001
Under	4	1470	1	368	< 0.001
Under*Over	16	1470	1	92	< 0.001

We next remove the KC1-Class dataset. While the KC1 dataset covers over 2,000 methods, these are grouped in approximately 170 classes, making KC1-Class actually quite small. This is now also the only dataset containing object-oriented metrics. With the removal of this dataset, we are left purely with large SDP datasets containing function-level metrics. The results for this group are presented in Tables 4.13 and 4.15. Again, the results are the same; under-sampling is significant, the main effect of over-sampling is not, but the interaction between under- and over-sampling is. The results of Tables 4.9, 4.11, 4.13 and 4.15 show that the results of Tables 4.3 and 4.5 were not caused by unique results in the most "unusual" datasets.

Table 4.13. ANOVA results for CM1, KC1, KC2, JM1, PC1 with SMOTE

Source	SS	df	Mean Square	F Value	P-value
Dataset	8.406	4	2.101	154.905	< 0.001
Under	0.451	4	0.451	33.234	< 0.001
Over	0.005	4	0.005	0.387	0.818
Under*Over	0.087	16	0.087	6.418	< 0.001
Lack of fit	1.817	96	0.016	1.162	0.144
Pure Error	15.032	1125	0.013		

Table 4.14: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, with SMOTE

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	4	1221	1	305	< 0.001
Under	4	1221	1	305	< 0.001
Under*Over	16	1221	1	76	< 0.001

Table 4.15. ANOVA results for CM1, KC1, KC2, JM1, PC1 with SMOTE-SN

Source	SS	df	Mean Square	F Value	p-value
Dataset	9.877	4	2.469	185.596	< 0.001
Under	0.524	4	0.524	39.409	< 0.001
Over	0.003	4	0.003	0.225	0.924
Under*Over	0.163	16	0.163	12.243	< 0.001
Lack of fit	1.841	96	0.016	1.206	0.093
Pure Error	14.683	1125	0.013		

Table 4.16: *p*-values and adjusted degrees of freedom for CM1, KC1, KC2, JM1, PC1, with SMOTE-SN

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	4	1221	1	305	< 0.001
Under	4	1221	1	305	< 0.001
Under*Over	16	1221	1	76	< 0.001

4.3.2. Exploratory Analysis of Resampling Object-Oriented Datasets

As we have two datasets available that contain object-oriented metrics, we have undertaken an exploratory analysis of resampling in SDP datasets using object-oriented metrics. However, as KC1-Class contains a mixture of class-level and method-level metrics, and is in any case quite small, while the Mozilla dataset uses a different set of metrics, caution is urged in interpreting these results. An ANOVA analysis finds very different results from the full analysis in part A. In Table 4.17 we find that none of the effects we are testing were significant using SMOTE. Under-sampling, over-sampling and their interaction were all insignificant at α=0.05. However, with the SMOTE-SN algorithm, the interaction of under-sampling and over-sampling is significant as shown in Table 4.18.

Table 4.17: ANOVA results for KC1-Class and Mozilla with SMOTE

Source	SS	df	Mean Square	F Value	p-value
Dataset	10.907	1	10.907	539.635	< 0.001
Under	0.011	4	0.011	0.549	0.700
Over	0.025	4	0.025	1.214	0.304
Under*Over	0.026	16	0.026	1.301	0.192
Lack of fit	0.989	24	0.022	1.097	0.343
Pure Error	9.016	450	0.020		

Table 4.18. ANOVA results for KC1-Class and Mozilla with SMOTE-SN

Source	SS	df	Mean Square	F Value	p-value
Dataset	10.410	1	10.410	427.040	< 0.001
Under	0.026	4	0.026	1.081	0.365
Over	0.013	4	0.013	0.650	0.627
Under*Over	0.149	16	0.149	6.131	< 0.001
Lack of fit	1.458	24	0.032	1.375	0.113
Pure Error	10.608	450	0.024		

We also apply Box's method to correct heteroskedasticity in our object-oriented datasets. Table 4.19 and 4.20 show the *p*-values for our experiments using SMOTE and SMOTE-SN respectively. The resultant significant factors are still significant after Box's correction procedure; therefore, our ANOVA results remain valid.

Table 4.19: P-values and adjusted degrees of freedom for SMOTE

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	1	474	1	474	< 0.001

Table 4.20: P-values and adjusted degrees of freedom for SMOTE-SN

Factor	original df		adjusted df		p-value
	numerator	denominator	numerator	denominator	
Dataset	1	474	1	474	< 0.001
Under*Over	16	474	1	30	< 0.001

4.3.3. Discussion

Our results indicate that uniform under-sampling and the interaction between under-sampling and over-sampling are valuable in improving the accuracy of SDP models on our seven datasets. While over-sampling by itself is not significant at the 0.05 percent level, its utility in combination with random under-sampling cannot be overlooked. This finding holds for both the original SMOTE algorithm and the more recent SMOTE-SN variation.

4.4. Threats to Validity

In this section we examine the threats to internal and external validity in our experimental design. The threats to internal validity include adequacy of the linear model for ANOVA; the instrumentation threat of having metrics collected by different tools and personnel; and having a different set of metrics in two of the datasets. As to the first, we used the lack-of-fit test to protect against inadequacy of the model, as described in Section 4.2.1. Instrumentation threats arise from the fact that the datasets were collected at different locations and times by different personnel; they have been minimized as much as possible by using well-known benchmark datasets collected specifically for this purpose by NASA. Metrics in

the Mozilla dataset, which is the sole exception to this rule, was collected using a commercial tool. Collection of defect data for this dataset was performed on a full copy of the Bugzilla defect-tracking database; see [191] for details. Finally, the different metrics employed in KC1-Class and Mozilla are a consequence of using object-oriented development; this instrumentation threat is unavoidable as object-oriented programs are not reasonably represented by method-level metrics alone. However, our sensitivity analysis in Section 4.3.1 indicates that the results were not unduly biased by these two datasets.

Threats to external validity include the choice of levels adopted in our study, and the preponderance of datasets from NASA's MDP program, which could introduce a source bias into the study. The levels we chose for our ANOVA experiments were based on our previous work [113]; in those experiments we observed an improvement of at least 23% in the geometric mean accuracy of the c4.5 classifier on a subset of the datasets in the current study, using those levels. While this is only an empirical finding, and does not guarantee that these levels are the most effective in resolving class imbalance, we believe they are reasonable selections. The combination of under- and over-sampling can alter the class distributions by a factor of 25, which is greater than the class imbalance in any of our datasets. Source bias is an important concern, as six out of seven datasets were obtained from the NASA Metrics Data Program. These, however, are now widely-used software defect prediction benchmarks, and we have further alleviated the source bias by incorporating the Mozilla dataset.

4.5 Related work

There has been an on going debate in the machine learning community regarding the employment of over-sampling vs. under-sampling; opinions differ on whether one or the other (or a combination of both) is most effective for learning from imbalanced datasets in general. According to Weiss and Provost [11], undersampling datasets is enough on its own to have a better classification in the minority class. The authors assert that oversampling does not provide useful improvements over undersampling, while also increasing the time and memory requirements for learning algorithms. The researchers reach this conclusion after experiments on 26 datasets, which were first under-sampled to create controlled class distributions. Datasets used in their experiments include twenty datasets from the UCI repository [192], five datasets from previously published work by researchers at AT&T [25], and one dataset generated by the authors. Their results also show that there is no one "best" class distribution; the class distribution that led to the best classification accuracy (using the c4.5 decision-tree classifier, trained using default parameters) is dataset-dependent. Thus, merely balancing the class distributions does not necessarily yield the most accurate predictions. Estrabooks discusses in [193] the possibility of under-sampling the majority class, over-sampling the minority class, or both. After several experiments with different datasets they reached the conclusion that under-sampling is not better than over-sampling or vice versa, but a combination of them is the best option. They propose a resampling method involving a three-level combination approach:

classifier level which resulted from learners trained on data sets at different rates; expert level which combines the results of 10 classifiers located at the classifier level; and output level combining the results of the over-sampling and under-sampling located at the expert level. Drummond and Holte show in [194] that, using their own performance analysis technique (cost curves [195]) that undersampling generates more sensitivity to changes in the misclassification costs and class distribution. On the other hand, over-sampling is surprisingly ineffective when using C4.5's default settings; it often produces little or no change in performance when these factors are changed. Maloof in [196] compares learning from imbalanced or skewed datasets and learning when error costs are unequal, but unknown. He argues that while these problems are not exactly the same, they can be handled in the same manner. He over-sampled and under-sampled one of the datasets, showing that the ROC curves produced by this procedure are similar to those produced by varying the decision threshold or the cost matrix. His results suggest that re-sampling produces classifiers similar to those produced by directly varying the decision threshold or cost matrix. Adjusting the cost matrix, in turn, has the same effect as moving the decision threshold. Barandela et al. [197] compare over- and under-sampling over five datasets using the Nearest Neighbor (NN) rule for classification and the geometric mean as the performance measure. Their results indicate that, when the imbalance is not very severe, techniques for appropriately under-sampling the majority class are a better option; they suggest over-sampling the minority class only when the ratio of the majority and minority class is very high. Seiffert et al. [187] measure the contribution of data sampling

on imbalanced datasets. They apply 5 different resampling techniques: Random Undersampling, Random Oversampling, Wilson's Editing, SMOTE and borderline-SMOTE utilizing 10 different datasets and applying C4.5 as their classifier; their experiments are evaluated using the area under the ROC curve or AUC. Their results show performance improvement obtained by applying both over and under-sampling techniques compared to utilizing only a single sampling technique.

The previous studies each used a single classification algorithm to evaluate their stratification techniques. The intent is plainly to avoid confounding the results of the stratification technique (a preprocessing step) with the different capabilities of multiple classifiers. C4.5 decision trees, in particular, come close to being a de facto standard in this area of research. However, other authors *do* employ multiple classifiers in their studies, and then use statistical procedures such as ANOVA to deconfound the results. Van Hulse et al. [179] use 11 learning algorithms with 35 datasets applying 7 different performance measures. According to their results, random under-sampling and random over-sampling perform best compared to any of the "intelligent" sampling techniques such as SMOTE, Borderline-SMOTE, Wilson's editing, one-sided selection and cluster-based oversampling. The authors in [188] compare boosting and bagging techniques and perform statistical analysis using different learning algorithms. They use two bagging techniques and combine two resampling techniques with boosting. They compare 3 different classifier algorithms on four datasets with eight bagging and boosting versions

and seven different performance measures using ten-fold cross validation. They present *F*-statistics and *p*-values performed on almost 4 million experiments. Their results show that all five main factors in their experiment (learners, technique, class distribution, noise distribution and noise level) and all two and three-way interactions are significant at the 5% level. A hybridization of random undersampling and boosting (similar in philosophy to the SMOTEBoost technique) is presented in [198, 199].

4.5 Conclusions

In this chapter we use stratification to deal with the imbalanced dataset problem in software defect prediction datasets. Using the ANOVA procedure and a blocked factorial design we respond to the debate about the usefulness of over-sampling vs. under-sampling within the machine learning community. From our analysis we conclude that under-sampling and the interaction with over-sampling significantly improves the quality of software defect predictions for our datasets. However, the main effect of over-sampling was not significant. Based on these results, we do not advocate employing either over-sampling technique in isolation. Our results indicate that software quality analysts would likely benefit by applying both under- and over-sampling in creating SDP models; clearly, a structured method for exploring the resulting design space is needed. One possibility is to use the fivefold cross-validation strategy in [51]; another is to employ a classical response-surface methodology to find the optimal resampling

strategy. An investigation of these possibilities is urged, but is beyond the scope of this chapter.

Although the study presented in this chapter was performed on SDP datasets, we are interested in investigating the contribution of resampling strategies in more general datasets. Future work involves extending other experiments to binary classification problems in other domains. Since the datasets from different domains are expected to have very different characteristics, we will also carry out an analysis to determine what characteristics of the datasets would cause resampling to be more effective. In particular, we are interested in investigating whether there is a relationship between the concept complexity of imbalanced datasets and the effectiveness of under- vs over-sampling. Additionally, the small disjuncts problem in imbalanced datasets is another interesting characteristic.

Chapter 5. Synthetic Minority Oversampling for Function Approximation Problems

5.1 Introduction

To a large extent, research in learning from imbalanced datasets has focused on classification problems, i.e. those problems requiring the prediction of a categorical response variable. In the domain of software test management (to mention an example), identifying which modules contain failures and which are failure free is an essential first step to allocating testing resources. However, software test management can only be *optimized* if we have a prediction of how *many* failures are present in each faulty module (i.e. we predict the value of a *numeric* response variable). This is a function-approximation problem, which still suffers from the same sample selection bias as its classification-based counterpart (i.e. faults tend to cluster in just a few modules) [78]. There are numerous other function-approximation problems of importance, e.g. time series forecasting, stock market prediction, signal processing, etc. It is not unreasonable to expect that real-world problems in all of these domains may experience sample selection bias, due to the dynamics of the observed phenomena. Thus, developing techniques to mitigate sample selection bias in function approximation problems is an important problem for the machine learning community.

In his Nobel Prize-winning work[1], Heckman [8] developed a procedure for correcting sample selection bias; unfortunately, that technique is limited to linear regression models. In the years since, there has been additional work on correcting sample selection bias for regression problems in the econometrics community (see [200]). However, compared to the work on sample selection bias in classification algorithms, far less attention has been paid to this problem by the machine learning community. Sample selection bias for function approximation can interact with heteroskedasticity [81][2] in the continuous response variable, making predictions more error-prone than in classifiers. This has been directly observed in software defect prediction problems, where conversion of a numeric number of defects to a categorical occurrence of defects serves to reduce the heteroskedasticity of the dataset [141]. Existing work on sample selection bias in function approximation problems focuses on reweighting existing training samples [66, 116]. By giving greater weight to samples from an under-sampled region of the feature space, the selection bias is mitigated and a less-biased approximator can be learned. This is essentially a generalization of cost-sensitive classification, which is one of the two general techniques for correcting selection bias in classification problems. However, we are not aware of any previous investigation of stratification-based approaches. Such approaches would (one expects) have different properties than cost-sensitivity. As we can never tell in advance what pattern-recognition techniques will work best for a given problem, having such a different technique available can be expected to aid a data mining

[1] Noble Prize in Economic Sciences, 2000
[2] Nobel Prize in Economic Sciences, 2003

analyst. We would expect that stratification will work better than cost sensitivity in some cases, and vice versa. The goal of this chapter is thus to develop a stratification method to mitigate sample selection bias in function approximation problems.

We propose an extension of the SMOTE over-sampling technique [10] to continuous-valued response variables. SMOTE often improves classification performance in a great proportion of imbalanced datasets (either directly or through an interaction with under-sampling), making it a good candidate for function approximators. Furthermore, our case study in Chapter 3, as well as our investigations in Chapter 4, further support this focus on SMOTE. Our investigation will focus on the original SMOTE algorithm; extensions of more recent variations are beyond the scope of this chapter. Due to the fact that we no longer have an *a priori* division of the dataset into individual classes, this algorithm necessarily involves several steps. We first make use of the Expectation Maximization clustering algorithm to identify regions of the feature space with greater or lesser sampling densities. We select regions for over-sampling using a criterion that combines the local density and variance of the dependent variable in that region. As with the original SMOTE algorithm, we then select a random point along the line joining two nearest neighbors in an under-represented region of the input space, and create a synthetic example at that point, using linear interpolation to determine values for both the predictor and response variables. In order to evaluate our results we use sequential minimal optimization regression

(SMOreg, employing the default parameters in the WEKA implementation) as our function approximation algorithm and Normalized Mean Squared Error (NMSE) as our performance measure. Our algorithm reduces the NMSE on 20 out of 21 function approximation datasets analyzed in this paper; in 11 of them, the improvement is statistically significant at the 5% level.

5.2 System for Correcting Sample Selection Bias

As a starting point, we present two graphs of the function $f(x,y) = x * exp^{(-x^2-y^2)}$ which has similar behavior to our problem. The function does not generate points with uniform variance; instead we have values with higher variance concentrated in one area. The graph with the original distribution is shown in Figure 5.1. The values with the highest variance are located in the central region. However, our initial problem is similar to Figure 5.2, where the region of higher variance is much less densely sampled; this makes function approximation much more challenging. Our aim is to modify the distribution of training data points by adding synthetic examples or deleting over-represented samples, giving a more uniform distribution (resembling Figure 5.1 more than Figure 5.2).

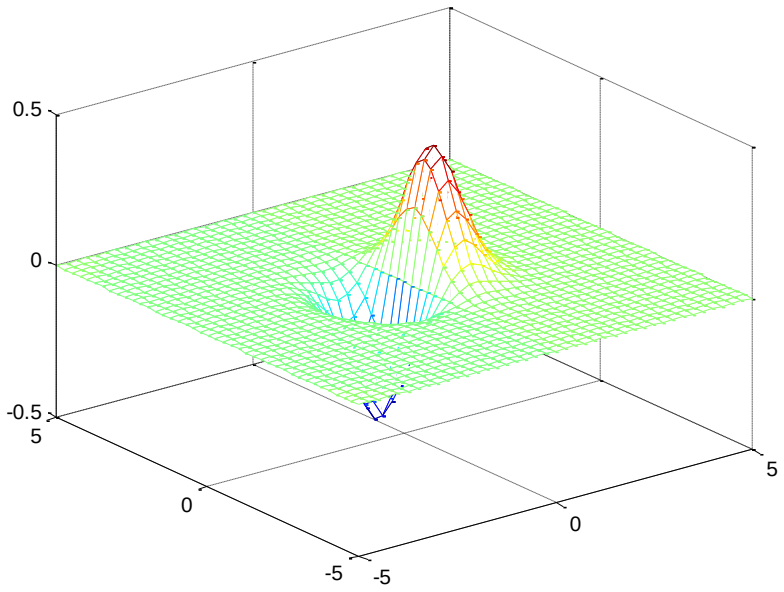

Figure 5.1. Original function $f(x,y)$

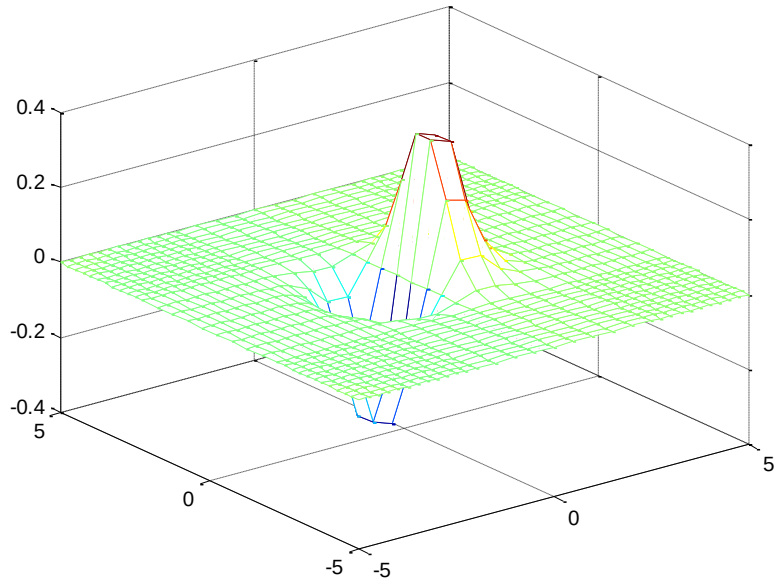

Figure 5.2. Function $f(x,y)$ Sampled on a Coarser Grid

In order to modify the distribution, we first need to identify those areas with higher or lower sampling density, and higher or lower variance. Our decision rule for over- or under-sampling must then integrate both factors (which we expect are independent). Our system follows the design in Figure 5.3.

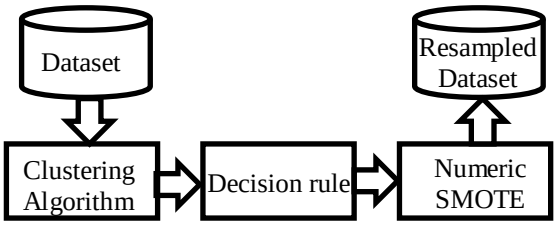

Figure 5.3. Numeric SMOTE Design

We first make use of a clustering algorithm to identify the regions of higher or lower density in the feature space. We select Expectation Maximization (EM) [201] as our clustering algorithm because it is a well-known and effective technique. The goal of this clustering algorithm is to maximize the overall probability or likelihood of the data, according to the clusters. We use maximum log-likelihood as our cluster validity criterion. We then proceed to calculate the *data volume* or the density of our clusters (see Sections 5.2.1 and 5.2.2, respectively); these are two options for expressing the density of a cluster, and we will empirically explore both alternatives. We combine the data volume or density with the variance of the dependent variable to create our resampling decision rule. The dataset behavior we are most concerned with is having high density areas with a low variance and low density areas with a high variance. Thus, we integrate these measures as:

$$Decision\ \ metric = \frac{density}{variance} \tag{5.1}$$

$$Decision\ \ metric = \frac{data\ volume}{variance} \tag{5.2}$$

Once we have the resampling decision metric, in our final decision rule we average the decision metrics from all the clusters. We then over-sample the clusters whose decision metrics are below the average and under-sample the clusters whose decision metrics are above the average. The Numeric SMOTE algorithm is explained in section 5.2.3.

5.2.1 Data Volume Measure

The data volume measure finds the number of points within an ε-neighborhood of each point in a cluster, and then finds the δ-neighborhood that would be required to enclose a fixed fraction of the total dataset at the average observed concentration. In other words, given an observed average density in a cluster, what is the volume needed to enclosed a fixed fraction of the total dataset? We use Algorithm 1 to compute the data volume measure.

We first calculate the diameter (maximum Euclidean distance between points) in the cluster. We then iteratively count the number of points in the cluster within an ε-neighborhood of each other, with ε increasing from 1% to 100% of the *dataset* diameter in 100 equal steps. (We demonstrate this is an adequate spatial sampling in the example in Section 5.2.1.1.) We use the dataset diameter because some

clusters can have points very close together or very sparse resulting in having very different cluster diameters; using the cluster diameters would thus distort the comparison of one cluster against another for our mean-based decision rule. In lines 7-19 of Algorithm 1 we calculate for each epsilon neighborhood the number of data points that are within a given neighborhood and then we average that value for all points (Eps in the algorithm). We aim to find at what δ-neighborhood every point has on average *n*% of all data points as its neighbors.

$$\sum_{i=1}^{numInst}\sum_{j=1}^{numInst}\begin{cases}0\ if\ \|instance_A - instance_B\| \geq epsNeighborhood\\ 1\ if\ \|instance_A - instance_B\| < epsNeighborhood\end{cases}$$

We experiment with *n* = 60, 70, 80 and 90% of the number of points; we set this percentage as a parameter (*n* in the algorithm) and calculate the percentage of number of points in line 20.

$$Avgnn = \frac{n}{100} * numberInstances \tag{5.3}$$

From lines 21 to 26 we determine the neighborhood size δ (*fdp* in Algorithm 1).

$$fdp = epsilonPoint(\min\{i : Eps(i) > Avgnn\}) \tag{5.4}$$

Finally, we subtract from 1 the result of dividing the neighborhood size δ (*fdp*) by the maximum distance between any two points in the dataset. The data volume measure will have higher (lower) values when the δ-neighborhood (*fdp*) required to enclose the Avgnn fixed fraction of the total dataset is lower (higher).

$$dvm = 1 - \left(\frac{fdp}{datasetDiameter}\right), \tag{5.5}$$

```
1. Input: Training dataset, n
2. Output: dvm measure
3. Calculate datasetDiameter
4. epsilonStep=100 (number of neighborhoods)
5. increment = datasetDiameter/epsilonStep
6. iter=1
7. for epsNeighborhood=0 + increment to datasetDiameter
8.    epsilonPoint[iter]=epsNeighborhood
9.    counter = 0
10.   for instanceA=1 to numberInstances
11.      for instanceB=1to numberInstances
12.         if eucDistance(instanceA,instanceB) < epsNeighborhood
13.                counter++
14.         end if
15.      end for
16.   end for
17. Eps[iter]=counter/numberInstances
18. iter++
19. end for
20. Avgnn= n/100 * numberInstances
21. for iter=1 to epsilonStep
22.    if Eps[iter] > Avgnn
23.         fdp = epsilonPoint[iter]
24.         break
25.    end if
26. end for
27. dvm = 1 – (fdp / datasetDiameter)
```

Figure 5.4. Algorithm 1: Data volume measure

5.2.1.1 Example

We show an example using the data volume measure calculated with one fold (90% training, 10% test data) of one of the datasets analyzed in Section 5.4 in this chapter. We use KC1 which is a software prediction dataset consisting of 144 instances and 94 continuous attributes where the target variable is the number of defects per module. (As we are using one fold in a cross-validation dataset, we are clustering only 121 points, and computing the data volume only for them.) Using EM to determine the clusters, we obtain five clusters for the training data. We

experiment with 50, 100 and 200 neighborhoods (2%, 1% and 0.5% of the dataset diameter for each neighborhood) with one fold for each dataset. We present 3 graphs for each cluster using the KC1 dataset for demonstration purposes. The first graph of each cluster shows the data volume with 50 neighborhoods; the second graph with 100 and the third graph with 200 neighborhoods.

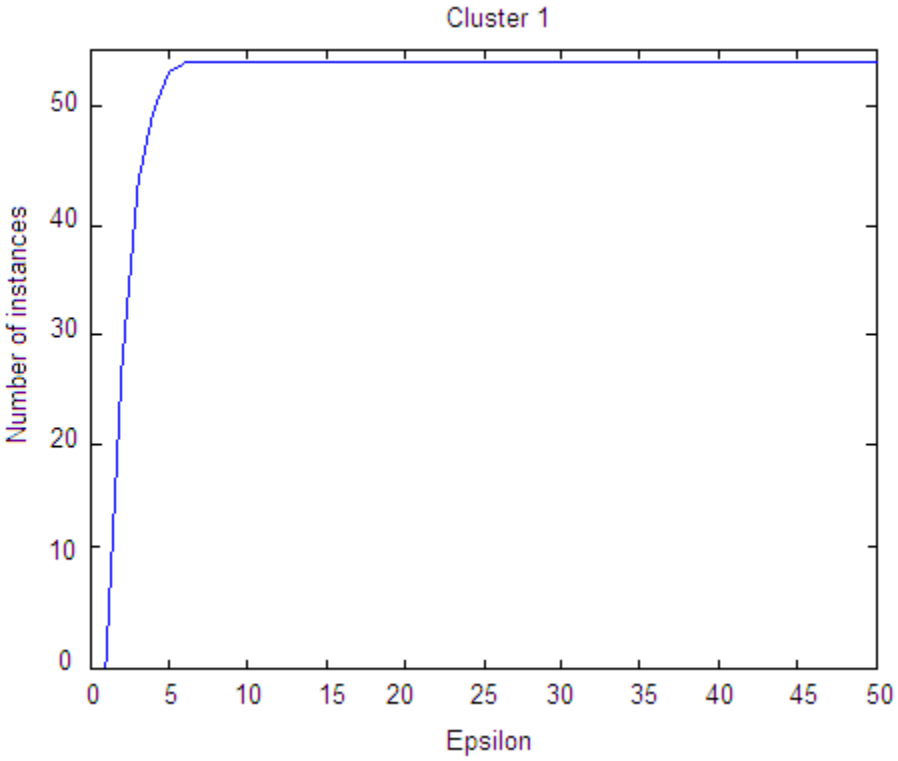

Figure 5.5 a) KC1 Cluster 1 with 50 neighborhoods

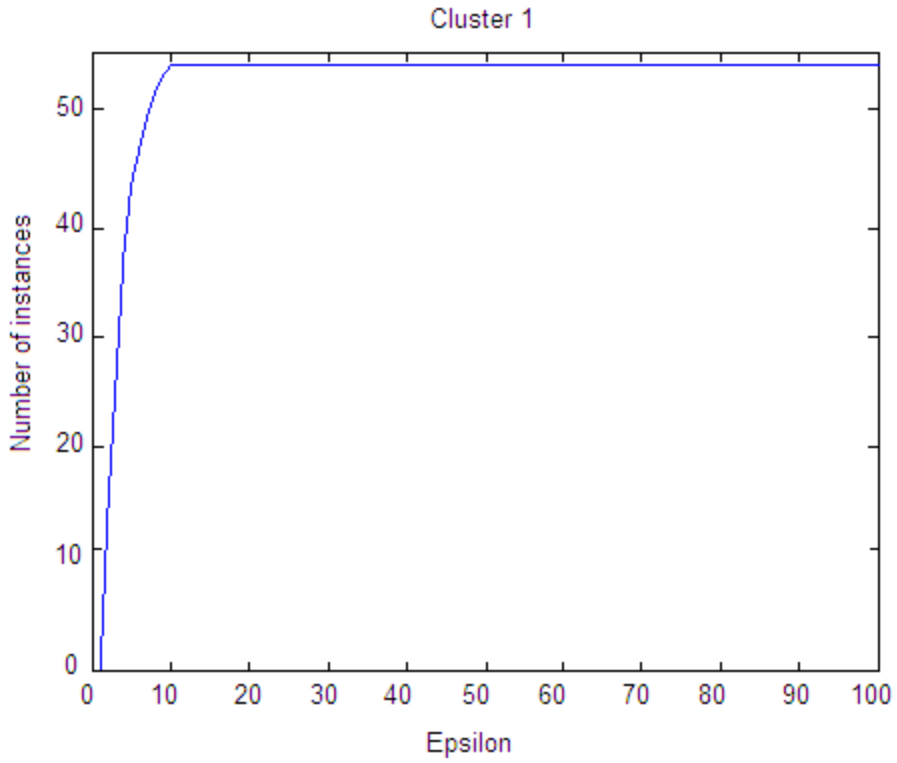

Figure 5.5 b) KC1 Cluster 1 with 100 neighborhoods

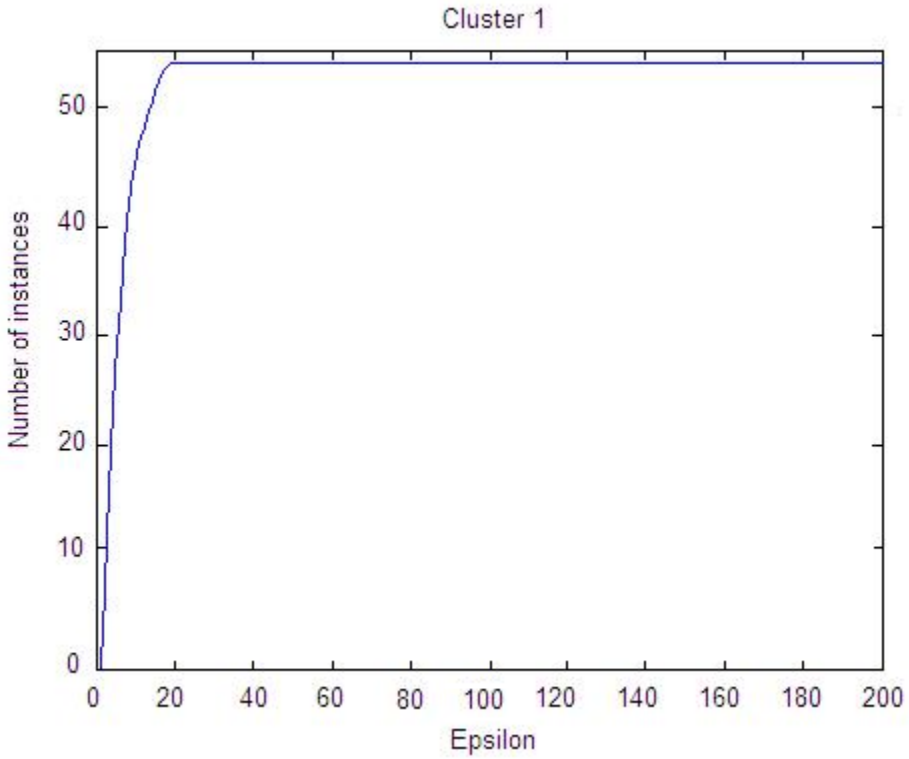

Figure 5.5 c) KC1 Cluster 1 with 200 neighborhoods

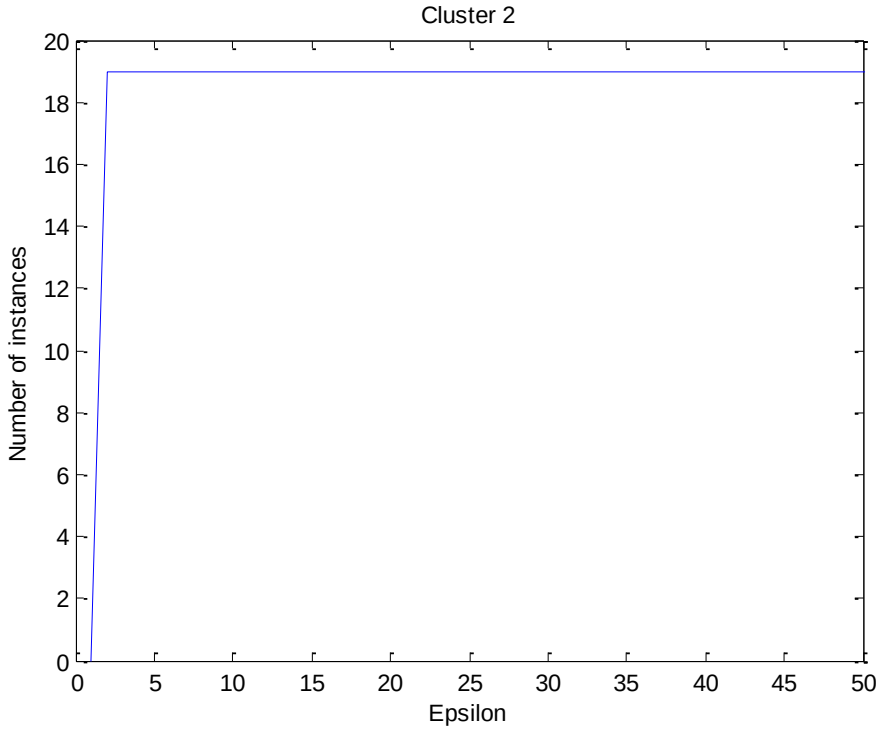

Figure 5.6 a) KC1 Cluster 2 with 50 neighborhoods

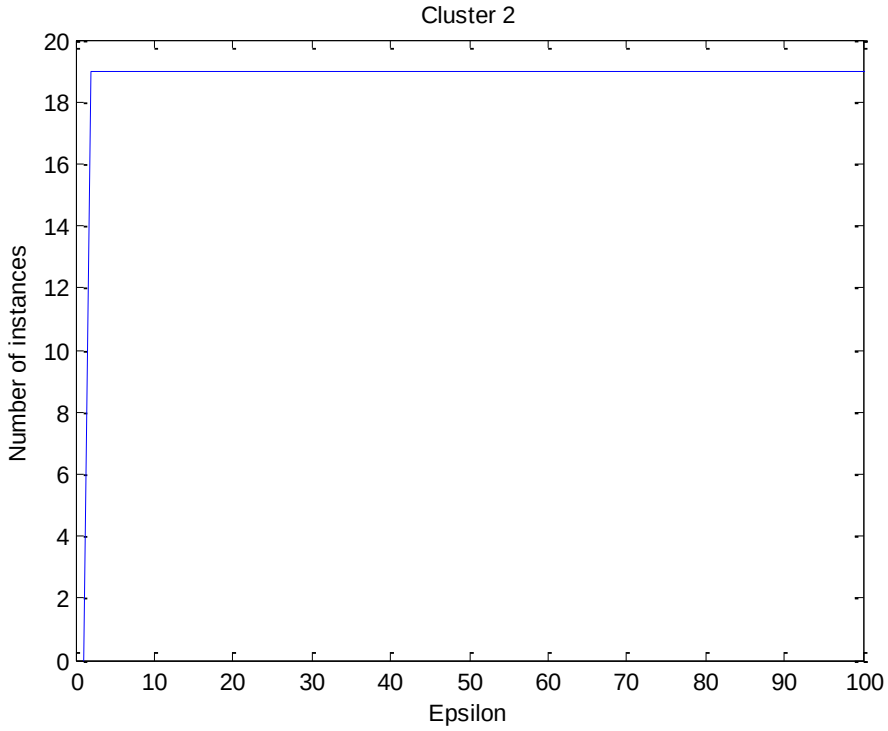

Figure 5.6 b) KC1 Cluster 2 with 100 neighborhoods

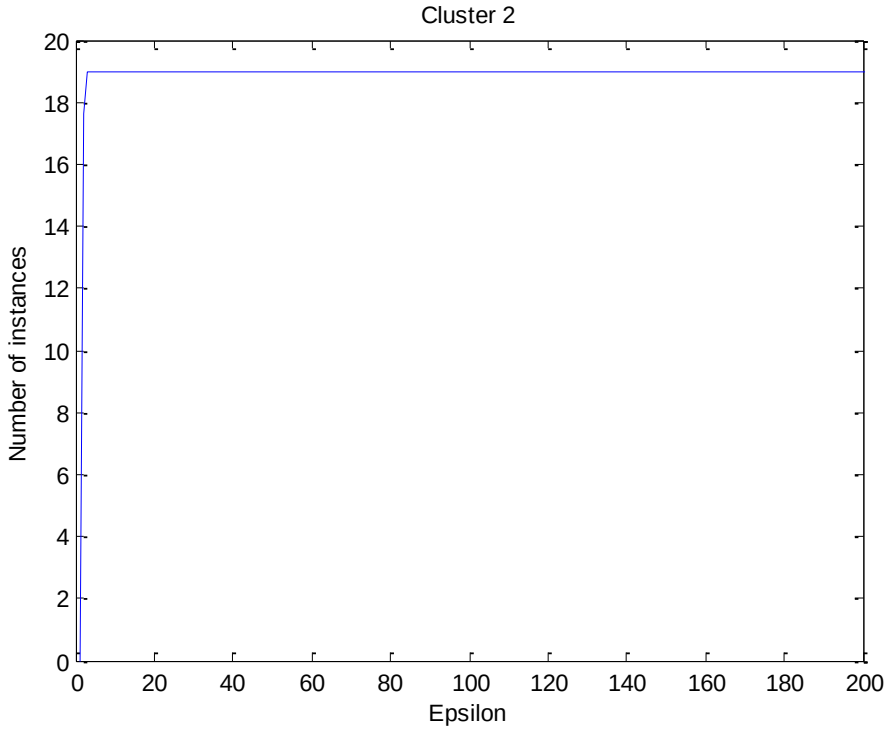

Figure 5.6 c) KC1 Cluster 2 with 200 neighborhoods

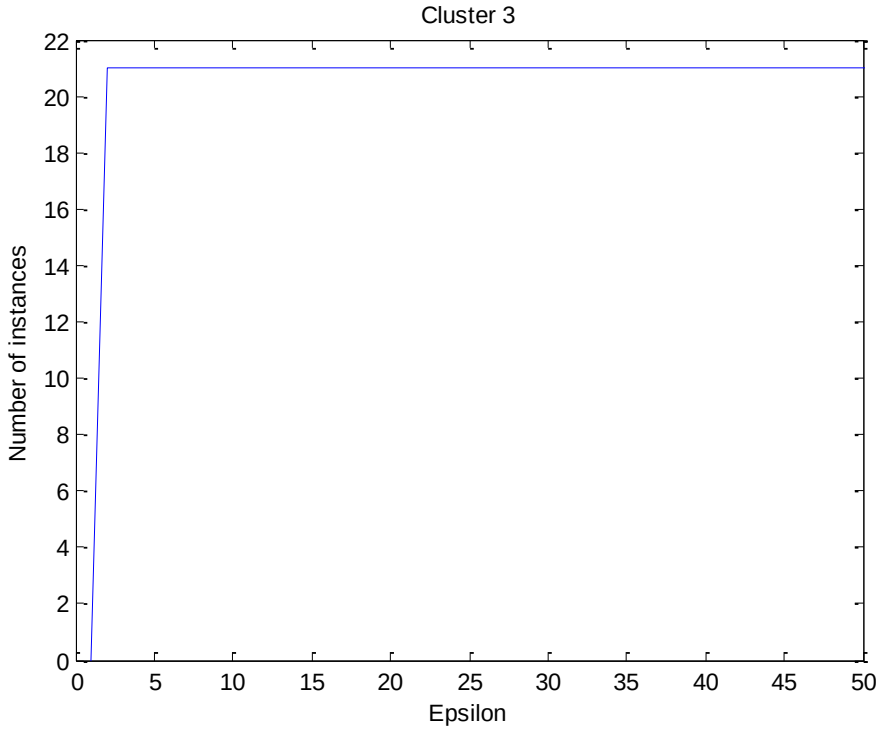

Figure 5.7 a) KC1 Cluster 3 with 50 neighborhoods

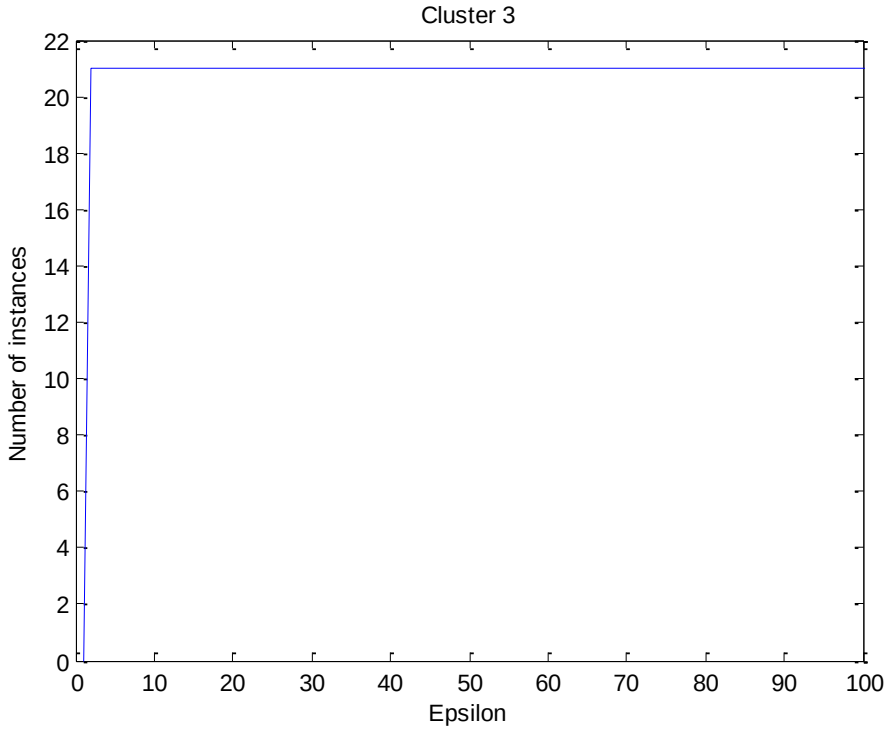

Figure 5.7 b) KC1 Cluster 3 with 100 neighborhoods

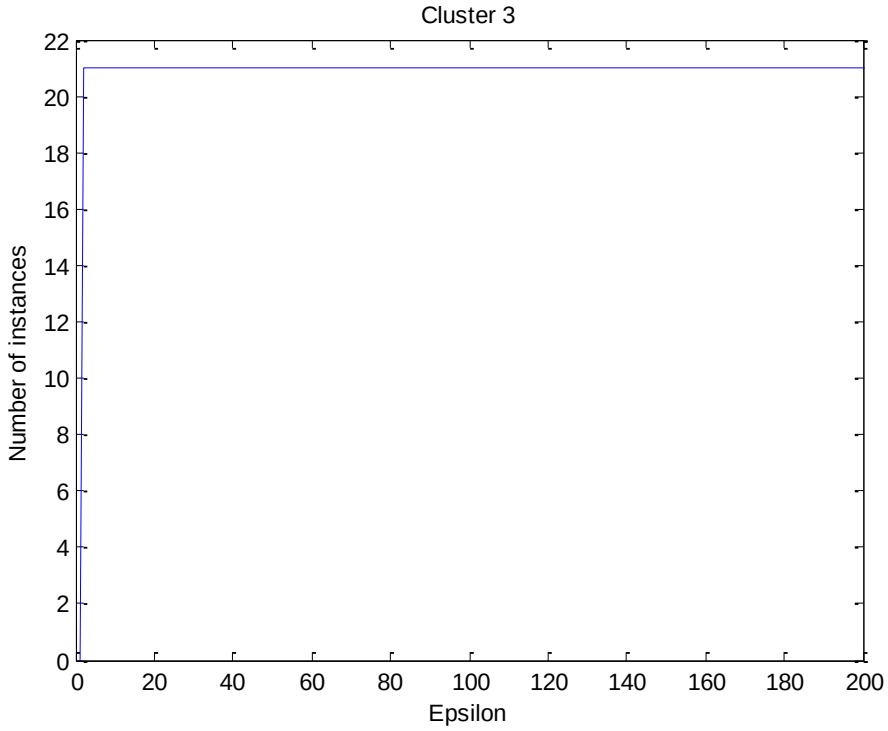

Figure 5.7 c) KC1 Cluster 3 with 200 neighborhoods

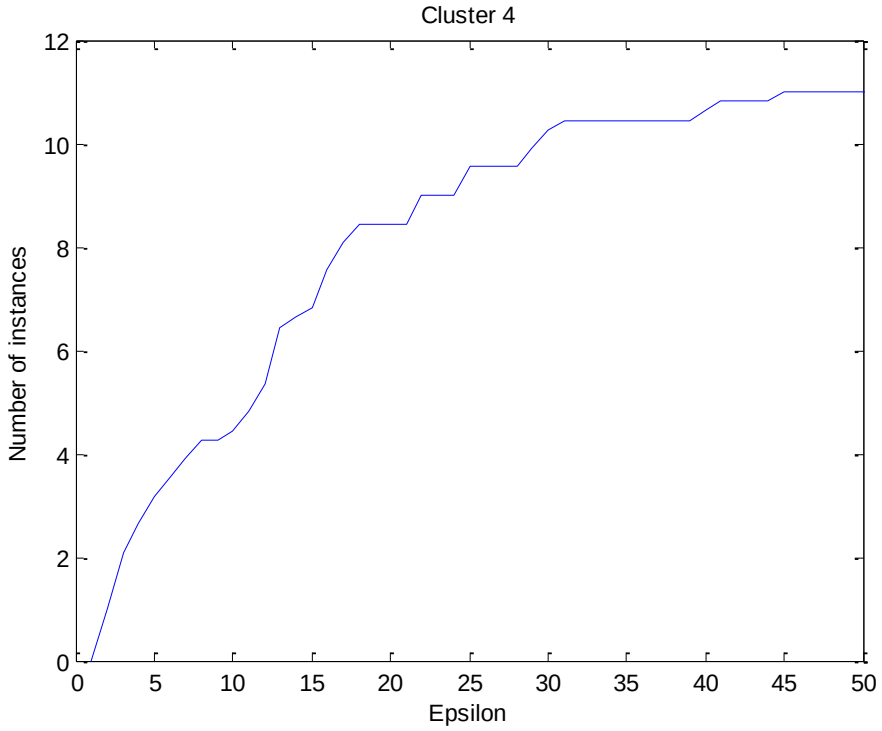

Figure 5.8 a) KC1 Cluster 4 with 50 neighborhoods

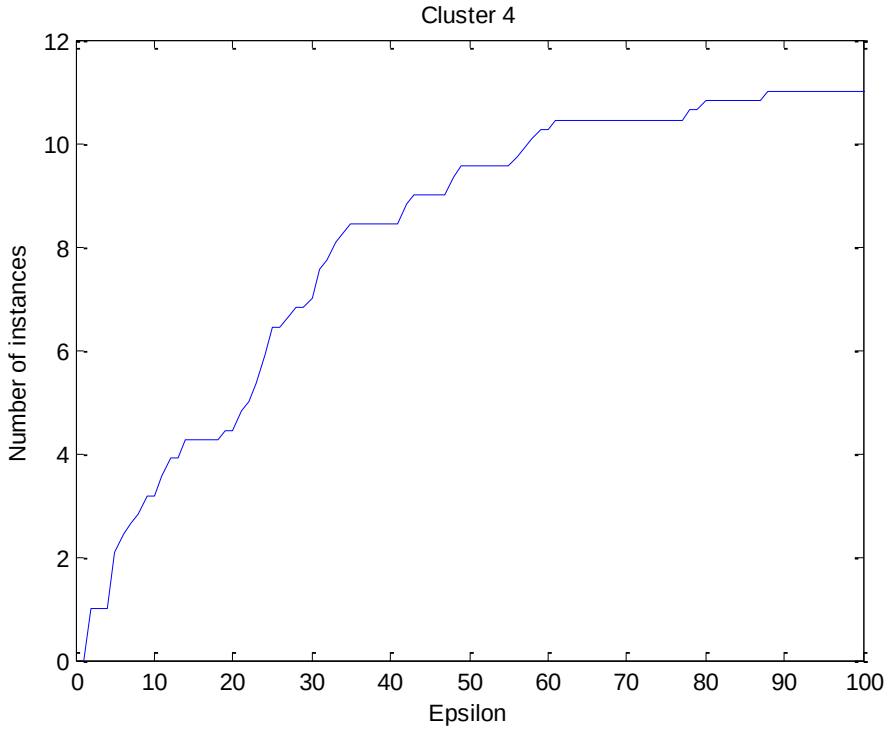

Figure 5.8 b) KC1 Cluster 4 with 100 neighborhoods

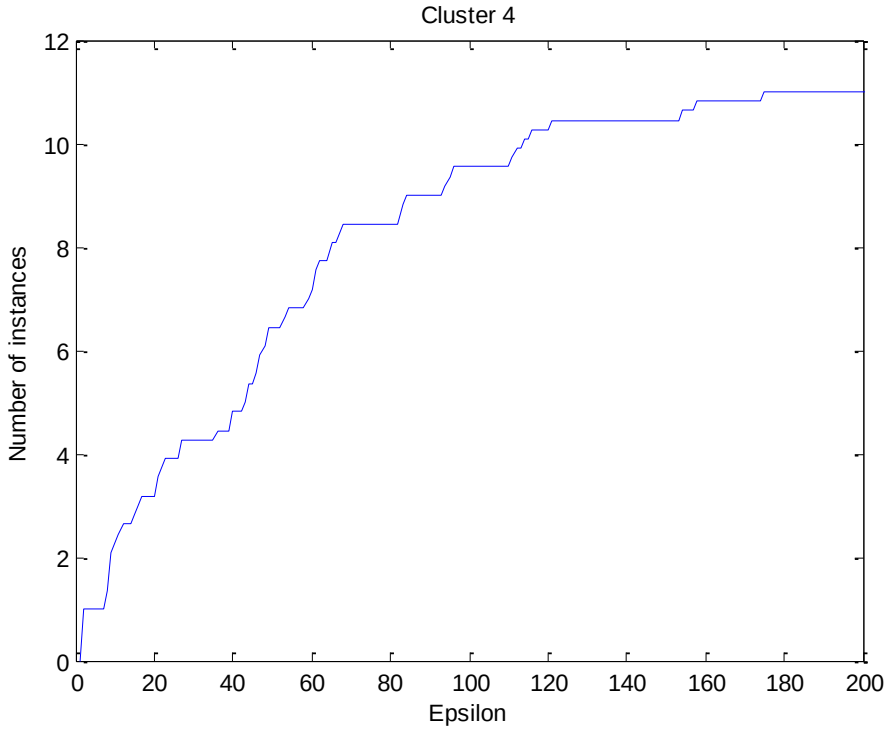

Figure 5.8 c) KC1 Cluster 4 with 200 neighborhoods

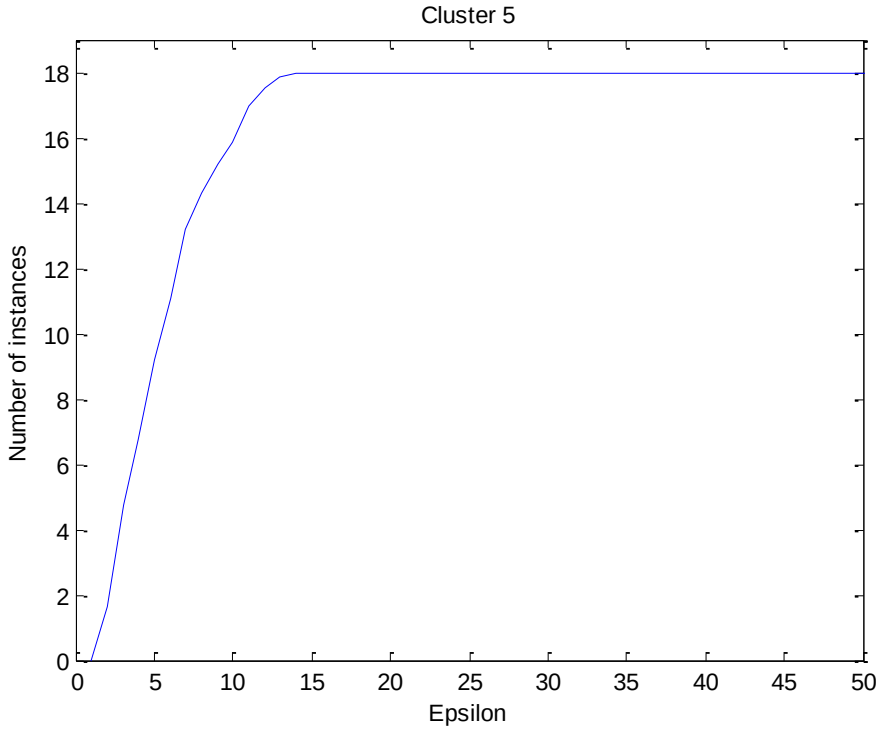

Figure 5.9 a) KC1 Cluster 5 with 50 neighborhoods

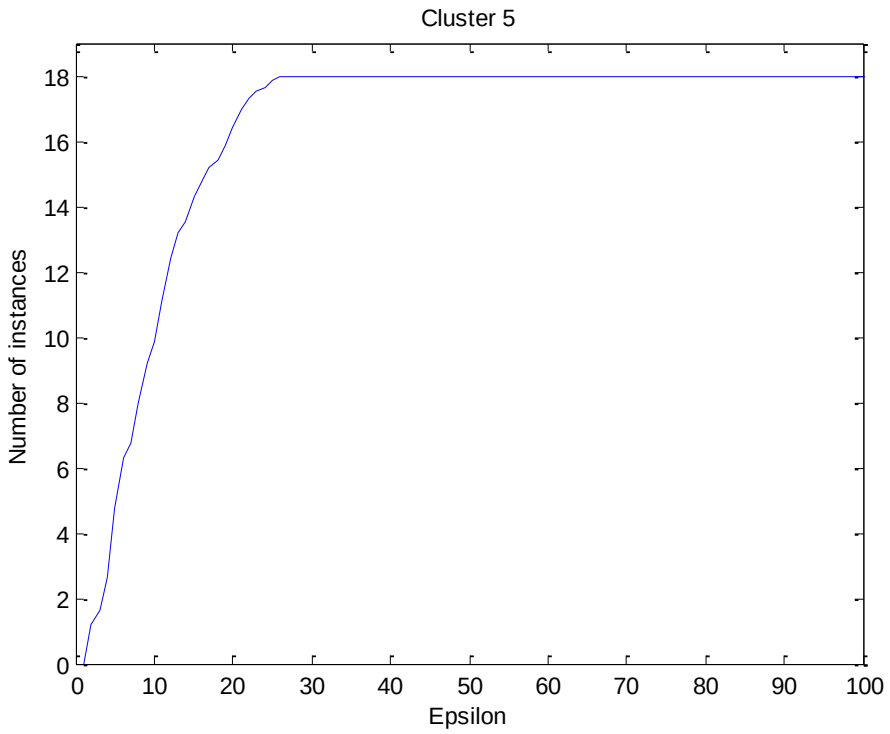

Figure 5.9 b) KC1 Cluster 5 with 100 neighborhoods

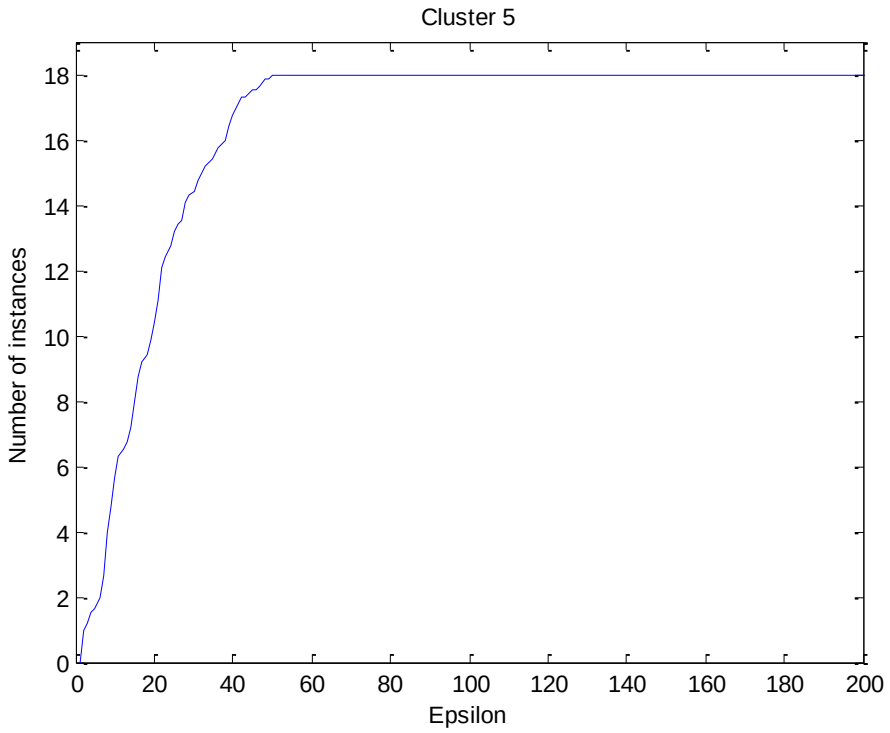

Figure 5.9 c) KC1 Cluster 5 with 200 neighborhoods

In Figures 5.5 – 5.9 the x-axis represents the spatial sampling of the clusters. The radius of a neighborhood is 18,982, 9,491 and 4,745.5 for 50, 100 and 200 neighborhoods respectively, based on a total dataset diameter of 949,100. The y-axis represents the number of instances in the cluster. We can notice that the different number of neighborhoods do not have significant effect on the results. The proportion of data points per neighborhood is consistent to the percentage of distance covered by the neighborhood. In other words, in this example sampling the number of points in the neighborhood at a spatial resolution of 2% of the dataset diameter reconstructs the distribution of data points as well as a resolution of 0.5%. We chose to set the number of iterations conservatively, and pick 100 iterations (representing 1% of the dataset diameter) in Algorithm 1. Thus, we have 100 iterations, with ε incremented by 9,491 where the dataset diameter=949,100 in the case of this example.

In Table 5.1 we present selected cluster characteristics for our example dataset. In this example, we have used n=80 as the fraction of the dataset to be enclosed in the final δ-neighborhood, as this was the value leading to the lowest NMSE in a parameter exploration. In Eq (5.3) for cluster 1, *Avgnn* is (80/100)*52 = 41.6 (cluster 1 has 52 instances). We determine *fdp* with Eq (5.4), where we obtain fdp = 37,976. Finally we calculate the data volume measure with Eq (5.5); dvm=1-(37,976 / 949,100) = 0.96.

The cluster diameter gives us an idea of how sparse or close together the data points are compared to the other clusters. We can see that in cluster four the maximum distance within the cluster is higher compared to the rest of the clusters and the data volume measure is the lowest. On the other hand, cluster three has the lowest cluster diameter and the highest data volume. When looking at the variance of the target variable (number of defects per module), we can notice that cluster four has the highest variance while cluster three has the lowest. For machine learning problems, it is likely easier to predict from areas as in cluster three where the variance is lower and there is a higher concentration of points. On the other hand, areas with similar behavior to cluster four, where the variance is quite high and the concentration of data points is much lower, can be expected to be more difficult.

Table 5.1. Values calculated for one fold of the KC1 training dataset

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
cluster diameter	84251	5381	1728	824730	229190
number of points	52	19	21	11	18
Avgnn	41.6	15.2	16.8	8.8	14.4
fdp	37976	9494	9492	389260	142410
dvm	0.96	0.98	0.99	0.59	0.85
variance	21.21	0.87	0.00	156.49	94.74

Looking at Figures 5.5 – 5.9 from each cluster, we can see how these results are represented in the graphs. We focus on the graphs with 100 neighborhoods. In cluster 3, 80% of the 21 data points in the cluster (n=80% where Avgnn =21*80% = 16.8) are within a distance = 9492 (fdp). For cluster 4, fdp = 389,260, which means at least 9 points (Avgnn = 8.8) are within a 389,260 distance. Cluster 5 has

at least 14 data points (80% of the number of points in the cluster) within a 142,410 distance. In Figure 5.7 for cluster 3, the δ-neighborhood (*fdp*) enclosing the fixed fraction of the dataset (*Avgnn*) is much lower compared to clusters 4 and 5 (Figures 5.7 and 5.8). The length of the neighborhoods in the x-axis in the graphs, as fractions of the dataset diameter, is fixed for all the clusters on this example, showing how 80% of the data points in cluster 3 are within a lower distance compared to the rest of the clusters. In Figure 5.7 for cluster 4 we can see it requires a much bigger distance to enclose its 80% data points.

After calculating the data volume measure and variance, we apply Eq (5.2) to calculate the decision metric. We average the decision metrics from all the clusters and will over-sample the clusters whose decision metric is below the average and under-sample the clusters with decision metric above the average. Once we have the resampling decision for the clusters, we are ready to apply Numeric SMOTE.

5.2.2 Standard density

Our alternative to the data volume measure is derived from the traditional interpretation of "density." In physics, density is calculated as mass divided by volume; in our case, we take the number of points divided by volume for each cluster. Because our dataset clusters arise from the EM algorithm, clusters should generally be hyper-ellipsoidal. We approximate the hypervolume of the cluster

(which is difficult to reliably determine) by the hypervolume of the ellipsoid covering that cluster, as determined by Principal Components Analysis (PCA). We first use the EM clustering algorithm in the same way we did with the data volume measure, calculating the maximum log-likelihood in order to set the number of clusters. Once we have the clusters, we calculate the hypervolume of the ellipsoid defined by PCA within each individual cluster. In order to calculate the hypervolume of the ellipsoids we use the lengths of the axes of the ellipsoid. The lengths of the axes of the ellipsoid are calculated as in [202]:

$$Axes\ length = \frac{1}{\sqrt{eigenvalue}} \tag{5.6}$$

For each cluster, the sum of the eigenvalues corresponds to the sum of the variance. In order to avoid numeric problems with computing hypervolumes in subspaces containing little to no variance, we only use eigenvalues greater than one percent of the variance in the cluster. Once we have the eigenvalues, we calculate the hypervolume of the ellipsoids with [203]:

$$V_{n+2} = \frac{2\pi}{n+2} Vn,\ with\ V_0 = 1, V_1 = 2, \tag{5.7}$$

where n is the number of dimensions. Finally, after calculating the hypervolume, we calculate the density as:

$$Density = \frac{number\ of\ points}{hypervolume} \tag{5.8}$$

We present in Figure 5.10 the algorithm to calculate the Standard density measure.

```
1. Input: Training dataset
2. Output: density
3. numberClusters=ExpectactionMaximization(trainingDataset)
4. for clusterNum=1 to numberClusters
5.      eigenValues[clusterNum]=PrincipalComponents(cluster[clusterNum])
6.      axesLength[clusterNum]=1/sqrt(eigenValues[clusterNum])
7.      numDim= size(axesLength)
8.      Eig=1
9.      for n =1 to numDim
10.         Eig=Eig*axesLength[n]
11.     end for
12.     if mod(numDim,2)==0
13.         VOld=1;
14.         init=0;
15.     else
16.         VOld=2;
17.         init=1;
18.     endif
19.     for a=init, init +2 to numAttribs-2
20.         VNew=2*pi/(a+2)*VOld
21.         VOld=VNew;
22.     endfor
23.     hyperVolume[clusterNum]=VNew*Eig
24.     density[clusterNum]=size(cluster[clusterNum])/hyperVolume[clusterNum]
25. end for
```

Figure 5.10. Standard density measure algorithm

We use the decision metric in Eq. (5.1) to determine the clusters to resample. Once we have our decision metric, we proceed in the same way we did with data volume measure, over-sampling those with lower metric values than the average, and under-sampling those with metrics above the average. We then proceed with the Numeric SMOTE procedure.

5.2.3 Numeric SMOTE

We extend the SMOTE over-sampling technique to continuous-valued response variables. SMOTE creates a new minority-class sample at a random point on the line connecting a minority class example with its nearest neighbor (of the same class) in feature space. In the case of Numeric SMOTE, we select a random point along the line joining two nearest neighbors in an under-represented region of the input space, and create a synthetic example at that point, using linear interpolation to determine values for both the predictor *and response* variables. Figure 5.11 shows the algorithm for Numeric SMOTE.

```
1. Input: cluster, resamplingRate, decisionRule
2. Output: resampled cluster
3. numberSamples=size(cluster)
4. if (decisionRule==oversample)
5.      N=resamplingRate/100 (it is assumed to resamplingRate be only multiples
of 100)
6.      numAttrs=number of Attributes
7.      Sample[][]=array with samples from cluster
8.      for i=1 to numberSamples
9.              Neighbor[N][numAttrs]=Generate N nearest neighbors for
Sample[i]
10.             for j=1 to N
11.                     randomVal=rand()
12.                     for attr=1 to numAttrs
13.                             diff=Sample[i,attr]-Neighbor[j,attr]
14.                             newSample[attr]=Sample[i,attr]+diff*randomVal
15.
        newSample(responseVar)=Sample[i](responseVar)*randomVal
16.                     end for
17.                     add newSample to cluster
18.             end for
19.     end for
20. end if
21. else if(decisionRule==undersample)
22.     N=resamplingRate/100*numberSamples (it is assumed
0<resamplingRate<100)
23.     for i=1 to numberSamples
24.             randomVal=rand()
25.             index=index*randomVal
26.             remove Sample[cluster,index]
27.     end for
28. end if
29. return cluster
```

Figure 5.11 Numeric SMOTE algorithm

5.3 Experimental Methodology

We evaluate the performance of the Numeric SMOTE algorithm using 21 well-known function approximation datasets. We use publicly available datasets from [192, 204, 205]. They range from 144 to 22,784 instances and from 6 to 94

continuous attributes. The datasets and their characteristics are listed in Table 5.2, where we provide the mean, median, standard deviation, kurtosis and skewness of the dependent variable.

Table 5.2. List of datasets

	Dataset	Instances	Attrib	Mean	Median	Std Dev	Kurtosis	Skewness
1	Abalone	4177	7	9.934	9.000	3.224	2.331	1.114
2	Ailerons	13750	40	-0.001	-0.001	0.000	2.592	-1.355
3	AutoMpg	398	4	23.515	23.000	7.816	-0.511	0.457
4	Auto_price	159	14	11445.730	9233.000	5877.856	2.600	1.592
5	Bank32nh	8192	32	0.084	0.027	0.122	3.757	1.966
6	Bank8FM	8192	8	0.162	0.117	0.152	0.491	1.057
7	CA Housing	20460	8	206855.817	179700.000	115395.616	0.328	0.978
8	CPU act	8192	21	83.969	89.000	18.402	12.725	-3.417
9	CPU small	8192	12	83.969	89.000	18.402	12.725	-3.417
10	Delta Ailerons	7129	5	0.000	0.000	0.000	2.922	0.294
11	Delta Elevators	9517	6	0.000	0.001	0.002	0.698	-0.176
12	Elevators	16559	18	0.022	0.020	0.007	7.618	2.425
13	House16H	22784	16	50074.440	33200.000	52843.476	20.327	3.755
14	House8H	22784	8	50074.440	33200.000	52843.476	20.327	3.755
15	Housing	506	12	22.533	21.200	9.197	1.495	1.108
16	KC1 - class	144	94	3.944	0.000	7.301	6.847	2.487
17	Kin8nm	8192	8	0.714	0.709	0.264	-0.534	0.090
18	Machine cpu	209	6	105.622	50.000	160.831	19.252	3.893
19	Pole	15000	48	28.945	0.000	41.726	-0.992	0.918
20	Puma32h	8192	32	0.000	0.000	0.030	0.040	0.016
21	Puma8nh	8192	8	1.162	1.115	5.622	-1.083	-0.150

We use a 10-fold cross validation design in all of our experiments. We resample the training sets only; the test sets are left unchanged. We cluster the training datasets, and apply our decision rule as described in Section 5.2; we repeat our experiments for both possible decision metrics (Eq. (5.1) or (5.2)).

Every dataset responds differently according to the resampling strategy (specification of the degree of under-sampling or over-sampling to be applied) as

observed in [11] and our earlier chapters. While over-sampling gives better results for some datasets, under-sampling is a better option for other datasets. The combination of both approaches has also been to be (statistically) significantly superior in terms of a performance measure (see Chapter 4, as well as other research, e.g. [187]). We explore using different combinations of resampling levels. The data mining literature makes extensive use of the trial-and-error parameter search paradigm, and we follow this design in our work. Normally, one would only analyze the best-performing parameter combination, as this is the only one that would be employed. In many practical data mining projects, a training set is often partitioned off from a test set, and then would be further subdivided into partitions for a cross-validation design (i.e. parameterization). The parameterized classifier is then applied to the out-of-sample test set. Alternatively (if it is assumed that there is a constant underlying model), entirely separate datasets may be used for parameterization and training/testing of the final classifier. Examples may be found in [11, 206-209]. We set five levels for under-sampling and five levels for over-sampling, having a total of 25 combinations. The over-sampling levels are 0, 100, 200, 300 and 400% (the percentage of synthetic examples added to the training dataset). We also use the under-sampling levels 0, 20, 40, 60 and 80%, which are the fraction of examples removed from the training dataset. These levels were determined from [113], where they were quite effective in improving SDP performance; they were also adopted in our research in Chapters 3 and 4. These ranges allow for a factor-of-25 change in the relative frequencies

of over- and under-sampled points, which should be enough to demonstrate improvement in all but the most heavily skewed datasets.

After resampling the training set, we evaluate it on the unaltered test set with sequential minimal optimization regression (SMOreg) [210] implemented in WEKA [211] using a Radial Basis Function Kernel and its default parameters. To evaluate the results from SMOreg we select the Normalized Mean Square Error (NMSE). Additionally, we perform a statistical analysis with two-sample t-tests and effect size on our datasets.

5.4 Experimental Results

We now present the results of our experiments. We first show the results comparing the original distribution of the dataset against the resampled set after using Numeric SMOTE with the data volume and standard density measures. We found that using the data volume measure produced better results on our datasets. Following, we thus compare the results of our algorithm with the data volume measure against those published by other authors using the same datasets and performance measure. However, because the authors use different function approximation algorithms we also include the results of our significance and effect-size tests on our data and the reported results in the literature. The raw data for the experiment results performed in this chapter can be found at [212]

Table 5.3 shows the results of Numeric SMOTE using the data volume and standard density. The second column shows the average and standard deviation of NMSE for each unaltered dataset. In the third column we find the results for Numeric SMOTE using the data volume. In the case of data volume measure, we explore different fractions of the dataset (variable n in Algorithm 1) to set our δ-neighborhoods. We only show the results where the lowest NMSE is obtained; the value of n is given as a percentage in the fourth column. We present in the fifth and sixth columns the best resampling combination; the fifth column corresponds to the over-sampling rate and the sixth column to the under-sampling rate. We finally show in the last three columns the results of Numeric SMOTE with the standard density and the corresponding resampling level where the lowest error is found, using the same format. Results improving on the original dataset are highlighted in bold.

Table 5.3. Results of Numeric SMOTE with data volume and standard density against the original distribution

Dataset	Original dataset	Data volume	n%	Over	Under	Std density	Over	Under
Abalone	0.468 ± 0.003	**0.429 ± 0.004**	80	100	0	**0.442 ± 0.003**	200	80
Ailerons	0.778 ± 0.009	**0.769 ± 0.006**	70	0	80	**0.765 ± 0.005**	0	80
AutoMpg	0.211 ± 0.011	**0.195 ± 0.008**	80	300	60	0.211 ± 0.011	0	0
Auto price	0.294 ± 0.014	**0.177 ± 0.018**	70	400	80	**0.215 ± 0.021**	100	20
Bank32nh	0.496 ± 0.012	**0.447 ± 0.023**	60	100	40	**0.449 ± 0.032**	100	60
Bank8FM	0.078 ± 0.010	**0.062 ± 0.004**	80	0	80	**0.076 ± 0.010**	0	60
CA Housing	0.394 ± 0.002	**0.385 ± 0.007**	70	0	80	**0.389 ± 0.006**	0	80
CPU act	0.507 ± 0.009	**0.458 ± 0.012**	80	400	20	**0.502 ± 0.019**	300	40
CPU small	0.494 ± 0.007	**0.343 ± 0.005**	70	400	80	**0.381 ± 0.009**	400	20
Delta Ailerons	0.935 ± 0.003	**0.908 ± 0.015**	70	400	40	**0.904 ± 0.003**	300	20
Delta Elevators	0.391 ± 0.009	**0.388 ± 0.012**	60	100	80	0.391 ± 0.009	0	0
Elevators	0.221 ± 0.007	**0.215 ± 0.005**	80	0	80	**0.218 ± 0.011**	200	60
House16H	0.833 ± 0.023	**0.812 ± 0.018**	60	0	80	**0.825 ± 0.029**	300	40
House8H	0.778 ± 0.024	**0.753 ± 0.011**	70	300	80	**0.748 ± 0.018**	400	60
Housing	0.323 ± 0.039	**0.311 ± 0.031**	60	0	40	**0.293 ± 0.011**	300	20
KC1 - class	0.902 ± 0.011	**0.861 ± 0.008**	80	0	20	**0.829 ± 0.048**	200	40
Kin8nm	0.573 ± 0.009	**0.565 ± 0.005**	80	0	40	**0.571 ± 0.006**	0	40
Machine cpu	0.113 ± 0.006	**0.089 ± 0.007**	90	100	0	**0.098 ± 0.011**	200	20
Pole	0.558 ± 0.089	0.558 ± 0.089	80	0	0	0.558 ± 0.089	0	0
Puma32h	0.833 ± 0.007	**0.819 ± 0.005**	70	0	40	**0.827 ± 0.003**	0	60
Puma8nh	0.642 ± 0.018	**0.628 ± 0.006**	80	0	60	**0.621 ± 0.009**	400	20

In Table 5.3 we can see that in the case of Numeric SMOTE with data volume we reduce the NMSE in 20 out of 21 datasets. As we can notice, the reduction on the NMSE is higher for some datasets, compared to others where the impact is not so high. For some datasets, the effect of over-sampling on high-variance areas works better; in other datasets, removing examples from low-variance areas gives better results; while for other datasets, using both strategies give the best results. The percentage parameter for the epsilon neighborhoods has some impact as well on the reduction of our performance measure. For Numeric SMOTE with standard density, we obtain a reduction in the NMSE in 18 out of 21 datasets. The resampling rate where we find the lowest NMSE is again, dataset dependent. We can observe that using Numeric SMOTE with the data volume measure we are able to reduce the NMSE for more datasets, and usually at a higher rate compared to the standard density measure.

The results of the t-test and effect size test for Numeric SMOTE with data volume and standard density are shown in Table 5.4. Numeric SMOTE with data volume is significant at the 5% level with a large effect size in these datasets on 18 out of 21 datasets. In the case of Numeric SMOTE with standard density, 13 datasets are significant at the 5% level with a large effect size.

Table 5.4. T-test and effect size for Numeric SMOTE with data volume and standard density

Dataset	Original dataset	Data volume	p-value	Effect size	Std density	p-value	Effect size
Abalone	0.468 ± 0.003	0.429 ± 0.004	**0.000**	11.03	0.442 ± 0.003	**0.000**	8.67
Ailerons	0.778 ± 0.009	0.769 ± 0.006	**0.019**	1.17	0.765 ± 0.005	**0.001**	1.79
AutoMpg	0.211 ± 0.011	0.195 ± 0.008	**0.001**	4.78	0.211 ± 0.011	1.000	0.00
Auto price	0.294 ± 0.014	0.177 ± 0.018	**0.000**	7.26	0.215 ± 0.021	**0.000**	4.43
Bank32nh	0.496 ± 0.012	0.447 ± 0.023	**0.000**	2.67	0.449 ± 0.032	**0.001**	1.94
Bank8FM	0.078 ± 0.010	0.062 ± 0.004	**0.000**	2.10	0.076 ± 0.010	0.545	0.28
CA Housing	0.394 ± 0.002	0.385 ± 0.007	**0.003**	1.75	0.389 ± 0.006	**0.031**	1.12
CPU act	0.507 ± 0.009	0.458 ± 0.012	**0.000**	4.61	0.502 ± 0.019	0.466	0.33
CPU small	0.494 ± 0.007	0.343 ± 0.005	**0.000**	24.82	0.381 ± 0.009	**0.000**	14.02
Delta Ailerons	0.935 ± 0.003	0.908 ± 0.015	**0.000**	2.50	0.904 ± 0.003	**0.000**	4.62
Delta Elevators	0.391 ± 0.009	0.388 ± 0.012	0.536	0.28	0.391 ± 0.009	1.000	0.00
Elevators	0.221 ± 0.007	0.215 ± 0.005	**0.042**	0.98	0.218 ± 0.011	0.478	0.03
House16H	0.833 ± 0.023	0.812 ± 0.018	**0.036**	1.01	0.825 ± 0.029	0.504	0.31
House8H	0.778 ± 0.024	0.753 ± 0.011	**0.011**	1.34	0.756 ± 0.018	**0.033**	1.04
Housing	0.323 ± 0.039	0.311 ± 0.031	0.457	0.34	0.293 ± 0.011	**0.041**	1.05
KC1 - class	0.902 ± 0.011	0.861 ± 0.008	**0.000**	4.26	0.829 ± 0.048	**0.001**	2.10
Kin8nm	0.573 ± 0.009	0.565 ± 0.005	**0.028**	1.10	0.571 ± 0.006	0.567	0.26
Machine cpu	0.113 ± 0.006	0.089 ± 0.007	**0.000**	4.00	0.098 ± 0.011	**0.002**	1.69
Pole	0.558 ± 0.089	0.558 ± 0.089	1.000	0.00	0.558 ± 0.089	1.000	0.00
Puma32h	0.833 ± 0.007	0.819 ± 0.005	**0.000**	2.30	0.827 ± 0.003	**0.028**	1.10
Puma8nh	0.642 ± 0.018	0.628 ± 0.006	**0.041**	1.04	0.621 ± 0.009	**0.005**	1.59

While it is common practice in the data mining literature (as in [10]) to only analyze the "best" parameter combination (as determined by a trial-and-error parameter exploration such as ours), this is perhaps statistically questionable. Accordingly, we apply Bonferroni's correction to of our results. We use in total 25 resampling strategies per dataset; to maintain α=0.05 in making 25 comparisons an α-level of 0.05/25=0.002 should be used. After applying Bonferroni's method Numeric SMOTE with data volume is significant in eleven datasets. When using the standard density measure, only seven datasets are significant at the adjusted α-level.

We also compare in Table 5.5 our Numeric SMOTE with data volume results against two publications exploring a subset of our datasets: [62, 66]. They use NMSE as their performance measure and ten-fold cross validation. However, the authors use different function approximation algorithms. The author in [66] use penalized least mean squares while the results from [62] use a version of Support Vector Regression (SVR) [213] available from LibSVM [214] with a Gaussian kernel with parameter $\sigma = \sqrt{d/2}$, where d is the number of attributes in the dataset. In our comparative experiments below, we use sequential minimal optimization regression (SMOreg) implemented in WEKA with a Gaussian kernel; for comparison purposes we also use $\sigma = \sqrt{d/2}$ (hence the results in Table 5.5 differ slightly from Table 5.4). Table 5.5 presents the original/un-weighted dataset results as well as the results after applying the corresponding method followed by their standard deviation. The first column for each algorithm shows the original

(unweighted) distribution of the dataset. In the second column we present the results after applying the method using a ten-fold cross validation experimental design. The Kernel Mean Matching (KMM) method from [66] is re-executed in [62] using the same SVR as the function approximation algorithm; it is shown in the last column of the table.

Table 5.5. Comparison of Numeric SMOTE data volume against other approaches

Dataset	Numeric SMOTE		Correcting Sample Selection Bias by Unlabeled Data [66]		Sample Selection Bias Correction Theory [62]		
	Original	resampled	un-weighted	KMM	un-weighted	clustered	KMM
Abalone	0.419 ± 0.012	0.381 ± 0.008	1.000 ± 0.080	0.600 ± 0.100	0.654 ± 0.019	0.623 ± 0.034	0.709 ± 0.122
Ailerons	0.745 ± 0.008	0.732 ± 0.004	1.500 ± 0.060	1.200 ± 0.200	----------	---------	---------
Bank32nh	0.472 ± 0.018	0.429 ± 0.004	23.000 ± 4.000	19.000 ± 2.000	0.903 ± 0.022	0.635 ± 0.046	0.691 ± 0.055
Bank8FM	0.061 ± 0.003	0.052 ± 0.005	0.500 ± 0.100	0.470 ± 0.050	0.085 ± 0.003	0.068 ± 0.002	0.079 ± 0.013
CA Housing	0.386 ± 0.011	0.372 ± 0.007	2.290 ± 0.010	1.240 ± 0.090	0.395 ± 0.010	0.375 ± 0.010	0.595 ± 0.054
Cpu act	0.297 ± 0.007	0.216 ± 0.017	10.000 ± 1.000	1.900 ± 0.200	0.673 ± 0.014	0.568 ± 0.018	0.518 ± 0.237
Cpu small	0.412 ± 0.012	0.308 ± 0.008	9.000 ± 2.000	2.000 ± 0.500	0.682 ± 0.053	0.408 ± 0.071	0.531 ± 0.280
Delta Ailerons	0.908 ± 0.015	0.876 ± 0.011	0.510 ± 0.010	0.401 ± 0.007	----------	---------	---------
Housing	0.405 ± 0.031	0.399 ± 0.021	0.800 ± 0.200	0.760 ± 0.070	0.509 ± 0.049	0.482 ± 0.042	0.469 ± 0.148
Kin8nm	0.548 ± 0.011	0.532 ± 0.007	0.850 ± 0.200	0.810 ± 0.200	0.594 ± 0.008	0.574 ± 0.018	0.704 ± 0.068
Puma8nh	0.637 ± 0.012	0.618 ± 0.005	1.100 ± 0.100	0.830 ± 0.030	0.685 ± 0.013	0.641 ± 0.012	0.903 ± 0.059

We apply two-sample t-tests and effect size test to compare the different approaches. In Table 5.6 we can observe that Numeric SMOTE algorithm with

data volume is significant at the 0.05 level in 10 out of 11 datasets; we also have a large effect size in these datasets. In the case of housing dataset we have a small improvement although our experiments are not significant at the 5% level. In [66] the authors obtain a significant error reduction in 7 out of 11 datasets with a large effect size as well. The result of the experiments in [62] are significant at the 5% level in 8 out of 9 datasets. When comparing the NMSE values on the original distribution from Numeric SMOTE to the unweighted NMSE values from [62], we notice that in all the cases the *original* NMSE with Numeric SMOTE is always lower compared to unweighted values in [62],[66]. This appears to be due to the different SVRs used in the experiments. This at the same time decreases the impact of Numeric SMOTE on the error reduction since there is less margin to improve. The last two columns correspond to the method from [66] re-executed in [62]. From these results, only the bank32nh dataset is significant at the 5% level. In four datasets: abalone, CA housing, kin8nm and puma8nh, the NMSE increases after applying the KMM method producing a negative effect size.

Table 5.6 P-values and effect size of Numeric SMOTE against other approaches

Dataset	Numeric SMOTE		Correcting Sample Selection Bias by Unlabeled Data		Sample Selection Bias Correction Theory. Clustered		Sample Selection Bias Correction Theory. KMM	
	p-value	Effect size	p-value	Effect size	p-value	Effect size	p-value	Effect size
Abalone	0.000	3.73	0.000	4.42	0.025	1.13	0.193	-0.63
Ailerons	0.000	2.05	0.000	16.40	--------	--------	--------	-------
Bank32nh	0.000	3.30	0.000	12.63	0.000	7.43	0.000	5.06
Bank8FM	0.000	2.18	0.673	0.19	0.000	6.67	0.189	0.64
CA Housing	0.004	1.51	0.001	2.03	0.000	2.00	0.000	-5.15
Cpu act	0.000	6.23	0.411	0.38	0.000	6.51	0.069	0.92
Cpu small	0.000	10.2	0.014	1.26	0.000	4.37	0.128	0.75
Delta Ailerons	0.000	2.43	0.000	11.23	---------	--------	--------	-------
Housing	0.620	0.23	0.000	4.80	0.203	0.59	0.436	0.36
Kin8nm	0.001	1.74	0.563	0.27	0.007	1.44	0.001	-2.27
Puma8nh	0.001	2.07	0.660	3.66	0.000	3.52	0.000	-5.10

We also apply Bonferroni's correction on our t-tests results in Table 5.6. Similarly to our experiments in Tables 5.3 and 5.4, we apply 25 resampling strategies per dataset. Thus, using Bonferroni's correction we again set a significance threshold of 0.05/25=0.002. Looking at our results in table 5.6, we have 9 out of 11 datasets significant at the corrected α significance level. Note, however, that there is not enough information available from the competing methods [62],[66] to apply Bonferroni's correction; the results reported are simply those from the "best" parameter combination the authors identified.

5.5 Related Work

Since Heckman's work on sample selection bias, others authors have proposed different approaches employing his methodology. In the following subsections, we look at the work that has been done to correct SSB in the areas of econometrics and machine learning.

5.5.1 Econometrics

Heckman's initial solution to sample selection bias was a maximum likelihood estimator. When the model is estimated by maximum likelihood the parameter estimates are fully efficient. Although maximum likelihood is the best when the model is correctly specified, it is possible to trade off some efficiency for estimators that are more robust to distributional assumptions. One way to relax normality, while remaining in the maximum likelihood framework, was suggested by Lee [215, 216]. Lee proposes transforming the stochastic components of the model into random variables that can be characterized by the bivariate normal distribution. For example, suppose the errors ε and v are drawn respectively from the non-normal but known distributions F(ε) and G(v). It is possible to transform ε and v into normal disturbances via the functions J1 and J2, which involve the inverse standard normal distribution function, such that:

$$\varepsilon^* = J_i(\varepsilon) \equiv \Phi^{-1}[F(\varepsilon)] \tag{5.9}$$

$$v^* = J_2(v) \equiv \Phi^{-1}[G(v)] \tag{5.10}$$

where the transformed errors ε^* and v^* now have standard normal distributions. The joint distribution of the transformed errors is now fully characterized through the bivariate normal distribution. Cuddeback et al. present in [217] a study on social services where they correct SSB. They use multiple regression, specifically binary probit regression, to estimate the combined and independent effects of the variables used to determine selection bias. They focus on the use of sample selection methods to detect and statistically correct for selection bias resulting from non-participation when selected individuals refuse to participate in a study or some others are lost due to transience, death, or other reasons.

Little and Rubin [218] analyze the problem of missing data; however, they are generally concerned with cases in which some of the features of an example are missing, and not with cases in which whole examples are missing. They developed a taxonomy for describing the assumptions concerning the dependence of the missing data on observed data. Generally there are three types of missing data mechanisms: *Missing At Random (MAR)* means that the missing value for X can be explained by other variables in the data set. After accounting for these other variables, however, the missing values are random. *Missing Completely At Random (MCAR)* means that the probability of having a missing value for X is unrelated to the value of X itself or to any other variables in the data set. *Not Missing At Random (NMAR)* is the case where the missing value is not completely random and cannot be completely explained by other variables in the data set. Even after the relationships between missing and observed data are accounted for,

missing values remain conditioned on other unobserved factors, including the missing values themselves. These three cases of missing data are related to the cases 1, 2 and 4 of SSB that are mentioned on [68] and discussed further in the next section; case 1 of SSB is related to MCAR, case 2 to MAR and case 4 to NMAR.

5.5.2 Machine learning

After Heckman's work for correcting sample selection bias in linear regression models, other authors have investigated how to mitigate the sample selection problem in broader areas (especially classification problems) using machine learning techniques. The authors use different approaches such as cost-sensitive techniques, resampling, and estimation of the density ratio between the training and test distribution. We cite some of related work on classification followed by work on function approximation problems.

Zadrozny [68] investigates the effect of sample selection bias on classification problems. The author compares Bayesian classifiers, decision tree learners and support vector machines and evaluates the effect of sample selection bias when applying them. A resampling method is then applied to correct the bias. More recently, the authors in [118] argue that most classifier learning algorithms may or may not be affected by sample selection bias as this will depend on the dataset as well as the heuristics or inductive bias implied by the classifier learning algorithm and their appropriateness to the particular dataset. Hein presents a study in [219]

about the problem of binary classification under sample selection bias from a decision-theoretic perspective. Starting from a derivation of the necessary and sufficient conditions for equivalence of the Bayes classifiers of training and test distributions, Hein provides conditions under which sample selection bias does not affect the performance of a classifier. Dudík et al. [220] study maximum entropy density estimation under sample selection bias and propose three bias correction approaches: taking advantage of unbiased sufficient statistics which can be obtained from biased samples; estimating the biased distribution and then factoring the bias out; and approximating the latter by only using samples from the sampling distribution. [70] explores how to use unlabeled data in a semi-supervised learning framework to improve the accuracy of descriptive models constructed from biased training samples. Singhi and Liu [221] study selection bias in the context of classification and also conclude that selection bias has less negative effect in classification than in function approximation.

In several papers SSB and covariate shift are treated as the same problem [64, 117, 222]. However, covariate shift is generally considered a case of SSB [66]. SSB occurs when the training and test sets are not drawn from the same distribution. In the case of covariate shift, only the input distribution changes, whereas the conditional distribution of the outputs given the inputs remains unchanged [65]. Covariate shift is related to the feature bias case (second case in Zadrozny's categorization). It describes the sample selection bias scenario where

p(s = 1|x, y) = p(s = 1|x) [223]. Feature bias is investigated in different works such as [224] and [222].

A few authors have also investigated the impact of sample selection bias and covariate shift on function approximation problems. In order to reduce the influence of covariate shift problem, some researchers estimate the ratio of the test and training probability density functions (pdf) (called density ratio or importance) by estimating training and test input pdfs separately and then taking the ratio of the estimated densities [225].

$$Importance: w(\boldsymbol{x}) = p_{te}(\boldsymbol{x}) / \, p_{tr}\,(\boldsymbol{x}) \tag{5.11}$$

where pte(x) is the test input pdf and ptr(x) is the training input pdf.

Let p(y|x,θ) be the model of the conditional density which is parameterized by $\theta=(\theta^1,\ldots,\theta^m)' \in \Theta \subset R^m$. Having independently and identically distributed samples of size n, denoted by $(x^{(n)},y^{(n)})=(x_t,y_t)$:, t=1,…,n, we obtain a predictive density $p(y|x,\hat{\theta})$ by giving an estimate $\hat{\theta} = \hat{\theta}(x^{(n)}, y^{(n)})$.

They re-weight the log likelihood of each training instance (x,y) using maximum likelihood estimation.

$$L_w(\theta|x^n, y^n) := -\sum_{t=1}^{n} l_w(x_t, y_t|\theta) \tag{5.12}$$

where $l_w(x, y|\theta) = -w(x) \log p(y|x, \theta)$. Then the maximum weighted log-likelihood estimate, denoted by $\hat{\theta}_w$, is obtained by maximizing Eq (5.12) over Θ. The weight function $w(x)=p_{te}(x)/p_{tr}(x)$ is the optimal choice for sufficiently large *n* in terms of the expected loss with respect to $p_{tr}(x)$.

The authors in [116] use instead a direct importance estimation method which directly estimates the importance without pdf estimation of the training and testing set. Their idea is to find an importance estimate $\widehat{w}(\boldsymbol{x})$ such that the Kullback-Leibler divergence between the test distribution and the weighted training distribution is minimized. The importance w(**x**) can be modeled by the following linear model:

$$\widehat{w}(\boldsymbol{x}) = \sum_{l=1}^{b} \alpha_l \varphi_l(\boldsymbol{x}) \quad (5.12)$$

where $\{\alpha_l\}_{l=1}^{b}$ are parameters to be learned from data samples and $\{\varphi_l(\boldsymbol{x})\}_{l=1}^{b}$ are basis functions such that $\varphi_l(\mathbf{x}) \geq 0$ for all $\mathbf{x} \in D$ and for l=1,2,…,b. Using the model $\widehat{w}(\boldsymbol{x})$, the test input pdf $p_{te}(\mathbf{x})$ can be estimated by

$$\hat{p}_{te} = \widehat{w}(\boldsymbol{x}) p_{tr}(\boldsymbol{x}) \quad (5.13)$$

The parameters $\{\alpha_l\}_{l=1}^{b}$ are learned in Eq. (5.12) so that the Kullback-Leibler divergence from $p_{te}(\boldsymbol{x})$ *to* $\hat{p}_{te}(\boldsymbol{x})$ is minimized.

Huang et al. [66] consider both the feature and class bias cases of SSB. They use information criteria to reweight samples under certain restrictions from a Borel probability distribution on training samples and another distribution from test samples. Storkey and Sugiyama [117] provide a generative framework for analysis of covariate shift. They evaluate Bayesian generative approaches to cope with such situations where the joint training distribution and the joint test distribution are different and when unlabeled test input points are available. Cortes and others [62] theoretically analyze the error introduced by estimating sample selection bias from data. Their analysis covers the kernel mean matching

procedure and a cluster based estimation technique. In [65] the contributing authors consider general dataset shift scenarios including covariate shift. Sugiyama and coworkers also discuss the problems of model selection and active learning in the covariate shift scenario [63]. They evaluate the conditional expectation of the generalization error given training inputs. The authors argue that the conditional expectation framework is more data-dependent and thus more accurate than the methods based on the full expectation. Smith and Elkan [64] build generative classifiers that address the selection bias problem in transfer learning.

5.6 Summary and Future Work

We present Numeric SMOTE, a new stratification algorithm to alleviate the sample selection bias focusing on function approximation problems. This algorithm is an extension of the SMOTE over-sampling technique using the combination of density and variance as the resampling criterion. We use two different measures for this purpose: standard density and data volume. When comparing the results, Numeric SMOTE with data volume reduces the NMSE in 20 out of 21 datasets; by comparison the standard density reduces the NMSE in only 18 datasets. We make use of two-sample t-test and effect size to perform a statistical analysis. The results of Numeric SMOTE with data volume are significant at the 5% level in 11 datasets while Numeric SMOTE with standard density is significant in 7 out of 21 datasets. We also compare our algorithm with

two other published papers dealing with sample selection bias. The results from [66] are significant at the 5% level in 7 out of 11 datasets while the results from [62] are significant in 8 out of 9 datasets. Our approach is significant in 9 out of 11 of the compared datasets.

Our experiments showed that Numeric SMOTE can be a valuable algorithm helping to increase the performance on function approximation problems and effectively reducing the impact of sample selection bias. In this research, Numeric SMOTE is applied to a broad variety of function approximation datasets indicating that it is potentially applicable in numerous domains. Researchers in need of alternative techniques to correct for sample selection bias (in order to avoid relying solely on reweighting samples) may benefit from Numeric SMOTE in their studies.

In future work we plan to investigate the effect of small disjuncts in the presence of SSB on function approximation problems. Furthermore, the results from previous studies indicate that using only under-sampling techniques can be a better option when the effect of SSB is not too drastic, applying over-sampling only when there is a stronger presence of SSB on the data. We will be performing an empirical analysis based on this idea.

Chapter 6. Conclusions

In this research we have investigated strategies to deal with the problem of imbalanced datasets, which impacts a large number of economic and health domains of great importance. To alleviate the problem of imbalanced datasets, several researchers have used stratification to modify the distribution of the classes in the dataset. We study the effectiveness of stratification strategies through a case study of the new Mozilla dataset, a new large-scale software defect prediction dataset which includes both object-oriented metrics and a count of defects per module. Due to the ineffectiveness of function approximation algorithms in modeling this dataset, we converted it in a binary classification problem and employed stratification-based resampling. We improved the classification on the minority (fault-prone) class, even though this is only 5% of the original dataset, improving as well the Kappa statistic performance measure. We then examine the more general question of what resampling strategies are best employed in software defect prediction generally. We employ full-factorial design experiments using the Analysis of Variance (ANOVA) over six software defect prediction datasets. Our analysis shows that under-sampling and the interaction with over-sampling significantly improves the quality of software defect predictions. However, the main effect of over-sampling was not significant.

Finally, we developed a stratification method to mitigate sample selection bias in function approximation problems. The sample selection bias problem is worse for

function approximators than for classifiers, due to the greater variance of the continuous response variable, and therefore we need strategies to overcome this complexity. We extended the well-known SMOTE over-sampling technique to continuous-valued response variables. Our Numeric SMOTE algorithm involves several steps; we first make use of a clustering algorithm to identify the regions of higher or lower density in the feature space, we then calculate the decision rule and we finally apply the resampling strategy. We propose two different density measures to determine the decision rule: data volume and the standard density. Data volume expresses the density of a cluster by the volume needed to enclosed a fixed fraction of the total dataset, while the standard density is computed by the number of points divided by volume for each cluster. We combine the data volume or density with the variance of the dependent variable to create our resampling decision rule which determines what clusters in the training set will be over-sampled or under-sampled. Our algorithm is tested on 21 benchmark datasets ranging from 144 to 22,784 instances and from 6 to 94 continuous attributes. Numeric SMOTE proved to be significant at the 5% level in 11 out of 21 benchmark datasets.

Numeric SMOTE can be applied to multiple areas of research. Our algorithm provides an alternative tool for dealing with SSB in function approximation problems, that appears to be competitive with existing approaches. In this research we use a total of 21 publicly available datasets with numeric target

variables but this can be extended to other studies involving datasets with continuous target variables.

As part of our future work, we plan to extend the full-factorial ANOVA design performed on Chapter 4 to other domains with binary-class datasets. Since the datasets from different domains will have characteristics wildly different, we will also carry out an analysis to determine what characteristics of the datasets would cause resampling to be more effective. This analysis will be also applied to continuous response variables with the use of Numeric SMOTE. We are also interested in investigating whether there is a relationship between the concept complexity of imbalanced datasets and the effectiveness of under- vs over-sampling on both binary and continuous response variables. Previous studies have suggested to apply only under-sampling when the SSB is not severe, and combine it with over-sampling only when there is a stronger presence of SSB in the data. We will dedicate part of our future work to the exploration of this suggestion. Finally, we intend to further investigate the small disjuncts problem in classification as well as function approximation problems.

The list of contributions generated by this research is as follows:

1. L. Pelayo and S. Dick, "Applying Novel Resampling Strategies To Software Defect Prediction," in Proc. North American Fuzzy Information Processing Soc., San Diego, CA, 2007, pp. 69-72.

2. L. Pelayo, A. Sadia, and S. Dick, "Predicting Object-Oriented Software Quality: A Case Study," International Journal of Intelligent Control and Systems (IJICS), v.14 (3), 2009, pp 180-192.
3. L. Pelayo and S. Dick, "Evaluating stratification alternatives to improve software defect prediction," IEEE Transactions on Reliability, submitted, 2010.
4. L. Pelayo and S. Dick, "Synthetic Minority Oversampling for Function Approximation Problems," IEEE Trans. Knowledge and Data Engineering, submitted, 2011.

Bibliography

[1] J. Johnson, "Let's Stop Wasting $78 Billion per Year," *CIO Magazine*, October 15 2001.

[2] A. C. Society, "Cancer Facts & Figures 2009," American Cancer Society, Available: http://oralcancerfoundation.org/facts/pdf/Us_Cancer_Facts.pdf, 2009.

[3] Synovate, "Federal Trade Commissioner-Identity Theft Survey Report," Available: http://www.ftc.gov/os/2007/11/SynovateFinalReportIDTheft2006.pdf, 2006.

[4] R. Pearson, G. Goney, and J. Shwaber., "Imbalanced clustering for microarray time-series," in *ICML'03 Workshop on Learning from Imbalanced Data Sets*, 2003.

[5] G. Wu and E. Y. Chang, "Class-boundary alignment for imbalanced dataset learning," in *ICML'03 Workshop on Learning from Imbalanced Data Sets*, 2003.

[6] H. He and X. Shen, "A Ranked Subspace Learning Method for Gene Expression Data Classification," in *International Conference on Artificial Intelligence*, Las Vegas, Nevada, USA, 2007.

[7] S. Dick, "Mozilla-Delta Dataset," University of Alberta: http://www.ece.ualberta.ca/~dick/datasets/Mozilla.csv, 2006.

[8] J. J. Heckman, "Sample selection bias as a specification error," *Econometrica*, vol. 47 (1), pp. 153-161, 1979.

[9] A. Sadia, "Mining software quality data from a large-scale open-source software system," M.Sc. thesis, Electrical and Computer Engineering, University of Alberta, Edmonton, 2005

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Syntethic Minority Over-Sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16 pp. 321-357, June 2002.

[11] G. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *Journal of Artificial Intelligence Research*, vol. 19 pp. 315 - 354, 2003.

[12] T. M. Mitchell, *Machine Learning*. New York, NY: McGraw-Hill, 1997.

[13] S. Dick and A. Kandel, "Data Mining with Resampling in Software Metrics Databases," in *Artificial Intelligence Methods in Software Testing*, M. Last, A. Kandel, and H. Bunke, Eds.: World Scientific, 2004, pp. 175–208.

[14] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6 (5), pp. 429-449, 2002.

[15] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *SIGKDD Explorations*, vol. 6 (1), pp. 40-49, 2004.

[16] M. C. Monard and G. Batista, "Learning with skewed class distributions," *Advances in Logic, Artificial Intelligence and Robotics*, pp. 173–180, 2002.

[17] S. Visa and A. Ralescu, "Learning imbalanced and overlapping classes using fuzzy sets," in *Proc. of the ICML- 2003 Workshop: Learning with Imbalanced Data Sets II*, Washington, DC., 2003, pp. 97-104.

[18] S. Visa and A. Ralescu, "Issues in mining imbalanced data sets-a review paper," in *Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference*, 2005, pp. 67–73.

[19] G. An, "The Effects of Adding Noise During Backpropagation Training on a Generalization Performance," *Neural Computation,* vol. 8 (3), pp. 643-674, 1996.

[20] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: the databoost-im approach," *ACM SIGKDD Explorations Newsletter,* vol. 6 (1), pp. 30-39, 2004.

[21] M. V. Joshi, V. Kumar, and R. C. Agarwal, "Evaluating boosting algorithms to classify rare classes: comparison and improvements," in *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, 2001, pp. 257-264.

[22] K. M. Ting, "A comparative study of cost-sensitive boosting algorithms," in *Proc. of 17th International Conf on Machine Learning*, Stanford University, CA, 2000, pp. 983-990.

[23] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "AdaCost: Misclassification cost-sensitive boosting," in *Proc. of Sixth International Conference on Machine Learning*, Bled, Slovenia, 1999, pp. 97-105.

[24] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine learning,* vol. 37 (3), pp. 297-336, 1999.

[25] W. Cohen and Y. Singer, "A simple, fast, and effective rule learner," in *Proc. 16th Nat. C. Artificial Intelligence*, Orlando, FL, USA, 1999, pp. 335-342.

[26] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM SIGKDD Explorations Newsletter,* vol. 6 (1), pp. 7-19, 2004.

[27] F. Provost and T. Fawcett, "Robust classification for imprecise environments," *Machine Learning*, vol. 42 (3), pp. 203-231, 2001.

[28] A. An, N. Cercone, and X. Huang, "A case study for learning from imbalanced data sets," *Advances in Artificial Intelligence*, pp. 1-15, 2001.

[29] Y. Li and Y. Yin, "A Novel Approach to Classify Imbalanced Dataset Based on Rare Attributes and Double Confidences," in *Knowledge Discovery and Data Mining, 2009. WKDD 2009. Second International Workshop on*, 2009, pp. 535-538.

[30] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," *Lecture Notes in Computer Science*, vol. 3201 pp. 39-50, 2004.

[31] K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," in *Proceedings of the International Joint Conference on Artificial Intelligence*, Stockholm, 1999.

[32] G. Wu and E. Y. Chang, "KBA: Kernel boundary alignment considering imbalanced data distribution," *IEEE T. Know. Data Eng*, vol. 17 (6), pp. 786-795, 2005.

[33] J. Leskovec and J. Shawe-Taylor, "Linear programming boosting for uneven datasets," in *ICML*, 2003, p. 456.

[34] B. Raskutti, "Extreme Re-balancing for SVM's: a case study," in *ICML-KDD'2003 Workshop: Learning from Imbalanced Data Sets*, 2003.

[35] P. Li, P. L. Qiao, and Y. C. Liu, "A hybrid re-sampling method for SVM learning from imbalanced data sets," in *Proceedings of the 2008 Fifth IC on Fuzzy Systems and Knowledge Discovery*, Washington, DC. USA, 2008, pp. 65-69.

[36] E. DeRouin, J. Brown, H. Beck, L. Fausett, and M. Schneider, "Neural network training on unequally represented classes," in *Intelligent Engineering Systems Through Artificial Neural Networks*, C. H. Dagli, S. R. T. Kumara, and Y. C. Shin, Eds. New York, NY: ASME Press, 1991, pp. 135-141.

[37] S. Wang, K. Tang, and X. Yao, "Diversity Exploration and Negative Correlation Learning on Imbalanced Data Sets," in *Proc. Int. Joint C. Neural Networks*, Atlanta, GA, USA, 2009, pp. 3259-3266.

[38] M. Wasikowski and X.-W. Chen, "Combating the Small Sample Class Imbalance Problem Using Feature Selection," *IEEE Trans. Knowledge and Data Engineering*, vol. 22 (10), pp. 1388-1400, 2010.

[39] A. Liu, C. Matinn, B. La Cour, and J. Ghosh, "Effects of Oversampling Versus Cost-Sensitive Learning for Bayesian and SVM Classifiers," in *Data Mining*, R. Stahlbock, S. F. Crone, and S. Lessmann, Eds. New York, NY: Springer, 2010, pp. 159-192.

[40] S. Gazzah and N. E. B. Amara, "New Oversampling Approaches Based on Polynomial Fitting for Imbalanced Data Sets," in *Proc. IAPR Wkshp. Document Analysis Systems*, Nara, Japan, 2008, pp. 677-684.

[41] L. Breiman, "Bagging Predictors," University of California, Berkeley 1994.

[42] H. Schwenk and Y. Bengio, "AdaBoosting neural networks: application to on-line character recognition," in *Int. C. Artificial Neural Networks*, Lausanne, Switzerland, 1997, pp. 967-972.

[43] D. Wilson, "Asymptotic properties of nearest neighbor rules using edited data sets," *IEEE Trans. Systems, Man and Cybernetics*, vol. 2 (3), pp. 408-421, 1972.

[44] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Set: One-Sided Selection," in *Proc 14th Int'l Conf. Machine Learning*, 1997.

[45] H. Ahumada, G. L. Grinblat, L. C. Uzal, P. M. Granitto, and A. Ceccatto, "REPMAC: A new hybrid approach to highly imbalanced classification problems," in *Proc. Int. C. Hybrid Intelligent Systems*, Barcelona, Spain, 2008, pp. 386-391.

[46] D. J. Drown, T. M. Khoshgoftaar, and N. Seliya, "Evolutionary Sampling and Software Quality Modeling of High-Assurance Systems," *IEEE*

Trans. Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 39 (5), pp. 1097-1107, 2009.

[47] J. Zhang and I. Mani, "knn approach to unbalanced data distributions: A case study involving information extraction," in *Proc. of the ICML-2003 Workshop: Learning with Imbalanced Data Sets II*, Washington, DC., 2003, pp. 42-48.

[48] G. Batista, R. Prati, and M. C. Monard, "A study of the Behavior of Several Methods for Balancing Machine Learning Training Data," *Sigkdd Explorations*, vol. 6 (1), pp. 20-29, 2004.

[49] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *Proc. 7th Eur. C. Principles and Practice of Knowledge Discovery in Databases*, Dubrovnik, Croatia, 2003, pp. 107-119.

[50] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning," *Lecture Notes in Computer Science*, vol. 3644 pp. 878-887, 2005.

[51] N. V. Chawla, D. A. Cieslak, L. O. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," *Data Mining and Knowledge Discovery*, vol. 17 pp. 225-252, 2008.

[52] V. García, J. S. Sánchez, and R. A. Mollineda, "On the use of surrounding neighbors for synthetic over-sampling of the minority class," in *Proc. 8th C. Simulation, Modelling and Optimization* Santander, Cantabria, Spain 2008, pp. 389-394.

[53] B. B. Chaudhuri, "A new definition of neighborhood of a point in multi-dimensional space," *Pattern Recognition Letters*, vol. 17 (1), pp. 11-17, 1996.

[54] C. Elkan, "The Foundations of Cost-Sensitive Learning " in *Proc. Int. Joint C. Artificial Intelligence*, Seattle, WA, USA, 2001, pp. 973-978.

[55] P. D. Turney, "Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm," *Journal of Artificial Intelligence Research*, vol. 2 pp. 369-409, 1995.

[56] K. M. Ting and Z. Zheng, "Boosting trees for cost-sensitive classifications," in *Proc. Eur. C. Machine Learning*, Chemnitz, Germany, 1998, pp. 191-195.

[57] P. Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive," in *Proc Int. C. Knowledge Discovery and Data Mining*, San Diego, CA, USA, 1999, pp. 155-164.

[58] P. Sen and L. Getoor, "Cost-sensitive learning with conditional Markov networks," *Data Mining and Knowledge Discovery,* vol. 17 pp. 136-163, 2008.

[59] G. M. Weiss and Y. Tian, "Maximizing classifier utility when there are data acquisition and modeling costs," *Data Mining and Knowledge Discovery,* vol. 17 pp. 253-282, 2008.

[60] C. X. Ling and V. S. Sheng, "Cost-sensitive learning and the class imbalance problem," *Encyclopedia of Machine Learning,* 2008.

[61] N. Thai-Nghe, Z. Gantner, and L. Schmidt-Thieme, "Cost-Sensitive Learning Methods for Imbalanced Data," in *IEEE International Joint Conference on Neural Networks*, Barcelona, Spain, 2010.

[62] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, "Sample selection bias correction theory," in *Proceedings of the International Conference on Algorithmic Learning Theory*, 2008, pp. 38-53.

[63] M. Sugiyama, N. Rubens, and K. R. Müller, "A conditional expectation approach to model selection and active learning under covariate shift," *Dataset shift in machine learning,* pp. 107-130, 2009.

[64] A. T. Smith and C. Elkan, "Making generative classifiers robust to selection bias," in *Proceedings of the ACM SIGKDD International Conference*, 2007, pp. 657-666.

[65] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, Eds., *Dataset Shift in Machine Learning*. Cambridge, MA: MIT Press, 2009, p. 248.

[66] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007.

[67] C. Gourieroux, A. Monfort, E. Renault, and A. Trognon, "Generalised residuals," *Journal of Econometrics*, vol. 34 (1-2), pp. 5-32, 1987.

[68] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc. Int. C. Machine Learning*, Banff, AB, Canada, 2004, p. 114.

[69] C. Bishop, *Neural networks for pattern recognition*. Oxford, UK: Clarendon Press, 1995.

[70] W. Fan and I. Davidson, "On sample selection bias and its efficient correction via model averaging and unlabeled examples," in *SIAM Data Mining Conference*, Minneapolis 2007.

[71] T. J. McCabe, "A Complexity Measure," *IEEE Trans. Software Eng*, vol. 2 (4), pp. 308 - 320, 1976.

[72] M. H. Halstead, *Elements of Software Science*. New York: Elsevier, 1977.

[73] B. Boehm, B. Clark, E. Horowitz, and C. Westland, "Cost models for future software life cycle processes: COCOMO 2.0," *Annals of Software Engineering*, vol. 1 (1), pp. 57-94, 1995.

[74] L. C. Briand, J. Daly, V. Porter, and J. Wust, "A comprehensive empirical validation of design measures for object-oriented systems," in *Proc. Int. Soft. Metrics. Symp.*, Bethesda, MD, USA, 1998, pp. 246-257.

[75] D. Glasberg, K. El-Emam, W. Melo, and N. Madhavji, "Validating object-oriented design metrics on a commercial Java application," National Research Council of Canada, Technical Report NRC 44146 2000.

[76] R. Harrison, S. Counsell, and R. Nithi, "Coupling metrics for object-oriented design," in *Int. Soft. Metrics Symp.*, Bethesda, MD, USA, 1998, pp. 150-157.

[77] L. C. Briand, K. El Emam, and S. Morasca, "On the application of measurement theory in software engineering," International Software Engineering Research Network, Technical Report ISERN-95-04 1995.

[78] K. E. Emam, S. Benlarbi, N. Goel, and S. N. Rai, "The confounding effect of class size on the validity of object-oriented metrics," *IEEE Trans. Soft. Eng.*, vol. 27 pp. 630-650, 2001.

[79] K. E. Emam, "A methodology for validating software product metrics," *National Research Council of Canada, Technical Report NRC/ERB-1076*, 2000.

[80] J. C. Munson and T. M. Khoshgoftaar, "Software metrics for reliability assessment," in *Handbook of Software Reliability Engineering*, M. R. Lyu, Ed. New York, NY: McGraw-Hill, 1996, pp. 167-254.

[81] R. Engle, "Time-series Econometrics: Cointegration and Autoregressive Conditional Heteroskedasticity," Royal Swedish Academy of Sciences, Stockholm, Sweden 2003.

[82] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," *IEEE Trans. Software Engineering*, vol. 20 (6), pp. 476-493, 1994.

[83] M. Lorenz and J. Kidd, *Object-Oriented Software Metrics*. Englewood Cliffs, NJ. USA: Prentice Hall, 1994.

[84] L. C. Briand, J. Daly, and J. Wust, "A unified framework for cohesion measurement in object-oriented systems," *Empirical Software Engineering*, vol. 3 (1), pp. 65-117, 1998.

[85] S. Counsell, S. Swift, and J. Crampton, "The interpretation and utility of three cohesion metrics for object-oriented design," *ACM Trans. Soft. Eng. Method*, vol. 15 pp. 123-149, 2006.

[86] T. M. Meyers and D. Binkley, "An empirical study of slice-based cohesion and coupling metrics," *ACM Trans. Software Engineering and Methodology*, vol. 17 pp. 2:1-2:27, 2008.

[87] E. Arisholm, L. C. Briand, and A. Foyen, "Dynamic coupling measurement for object-oriented software," *IEEE Trans. Soft. Eng,* vol. 30 pp. 491-506, 2004.

[88] K. Gao and T. M. Khoshgoftaar, "A Comprehensive Empirical Study of Count Models for Software Fault Prediction," *IEEE Trans. Reliability,* vol. 56 (2), pp. 223-236, 2007.

[89] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine Series 6,* vol. 2 (11), pp. 559-572, 1901.

[90] D. Montgomery, *Design and analysis of experiments*, 5th ed. New York: John Wiley, 2001.

[91] J. Cohen, *Statistical power analysis for the behavioral sciences,* 2nd ed. Hillsdale, NJ: Lawrence Erlbaum, 1988.

[92] E. H. Wolf and J. I. Naus, "Tables of Critical Values for a k-Sample Kolmogorovb-Smirnov Test Statistic," *Journal of the American Statistical Association,* pp. 994-997, 1973.

[93] H. Scheffe, *The Analysis of Variance*. New York: Wiley, 1959.

[94] H. R. Lindman, *Analysis of variance in complex experimental designs*. San Francisco: W. H. Freeman & Co., 1974.

[95] G. E. P. Box, "Some theorems on quadratic forms in the study in analysis of variance problems II. Effect of inequality of variance and correlation between errors in the two-way classification," *Annals of Mathematical Statistics,* vol. 25 pp. 484-498, 1954.

[96] H. P. Piepho, "Detecting and handling heteroscedasticity in yield trial data," *Communications in Statistics-Simulation and Computation,* vol. 24 (1), pp. 243-274, 1995.

[97] S. W. Greenhouse and S. Geisser, "On methods in the analysis of profile data," *Psychometrika,* vol. 24 (2), pp. 95-112, 1959.

[98] J. R. Quinlan, *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[99] W. W. Cohen, "Fast effective rule induction," in *Proceedings of International Conference on Machine Learning*, Lake Tahoe, CA, 1995, pp. 115-123.

[100] J. Fürnkranz and G. Widmer, "Incremental reduced error pruning," in *Machine Learning: Proceedings of the Eleventh Annual Conference*, New Brunswick, New Jersey, 1994.

[101] J. G. Cleary and L. E. Trigg, "K^*: An Instance-based Learner Using an Entropic Distance Measure," 1995, pp. 108-114.

[102] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20 (3), pp. 273-297, 1995.

[103] B. Schölkopf and A. J. Smola, *Learning with kernels*: MIT Press, 2002.

[104] V. N. Vapnik, *The nature of statistical learning theory*. NY: Springer, 1995.

[105] D. G. Kleinbaum, L. L. Kupper, and K. E. Muller, *Applied regression analysis and other multivariable methods*: Duxbury Pr, 2007.

[106] Y. Wang, "A new approach to fitting linear models in high dimensional spaces," PhD thesis, Department of Computer Science, University of Waikato, New Zealand, 2000

[107] P. J. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences," PhD thesis, Harvard University, 1974

[108] F. O. Karray and C. W. D. Silva, *Soft Computing and Intelligent Systems Design: Theory, Tools and Applications*: Addison Wesley, 2004.

[109] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," *Advances in Kernel Methods-Support Vector Learning*, vol. 208 pp. 98–112, 1999.

[110] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of 2nd Berkeley Symposium*, Berkeley, 1951, pp. 481-492.

[111] J. Cohen, "A Coefficient of Agreement for Nominal Scales," *Educational and Psychological Measurement,* vol. 20 (1), pp. 37-46, 1960.

[112] B. D. Eugenio and M. Glass, "The kappa statistic: a second look," *Computational Linguistics,* vol. 30 (1), pp. 95-101, 2004.

[113] L. Pelayo and S. Dick, "Applying Novel Resampling Strategies To Software Defect Prediction," in *Proc. North American Fuzzy Information Processing Soc.*, San Diego, CA, 2007, pp. 69-72.

[114] A. Folleco, T. M. Khoshgoftaar, and A. Napolitano, "Comparison of Four Performance Metrics for Evaluating Sampling Techniques for Low Quality Class-Imbalanced Data," in *Proc. Int. C. Machine Learning and Applications*, San Diego, CA, USA, 2008, pp. 153-158.

[115] A. P. Bradley, "The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algortihms," *Pattern Recognition,* vol. 30 (7), pp. 1145-1159, 1997.

[116] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bunau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Proc. Ann. C. Neural Information Processing Systems* Vancouver, BC, Canada, 2008, pp. 1433–1440.

[117] A. J. Storkey and M. Sugiyama, "Mixture regression for covariate shift," *Advances in Neural Information Processing Systems,* vol. 19 pp. 1337-1344, 2007.

[118] W. Fan, I. Davidson, B. Zadrozny, and P. S. Yu, "An improved categorization of classifier's sensitivity on sample selection bias," in *Proc. IEEE Int. C. Data Mining* Houston, TX, USA, 2005, pp. 605-608.

[119] K. Pearson, "Notes on the History of Correlation," *Biometrika,* vol. 13 (1), pp. 25-45, 1920.

[120] M. Levinson, "Let's Stop Wasting $78 Billion per Year," in *CIO Magazine* October 15, 2001.

[121] N. Y. Lee and C. R. Litecky, "An empirical study of software reuse with special attention to Ada," *IEEE Trans. Soft. Eng,* vol. 23 (9), pp. 537-549, 1997.

[122] S. Sedigh-Ali, A. Ghafoor, and R. Paul, "Software Engineering Metrics for COTS Based Systems," *IEEE Computer,* vol. 34 (5), pp. 44-50, 2001.

[123] M. Morisio, M. Ezran, and C. Tully, "Success and Failure Factors in Software Reuse," *IEEE Trans. Soft. Eng,* vol. 28 (4), pp. 340-357, 2002.

[124] D. Spinellis and C. Szyperski, "How is open source affecting software development?," *IEEE Software,* vol. 21 (1), pp. 28-33, 2004.

[125] M. A. Friedman and J. M. Voas, *Software Assessment: Reliability, Safety, Testability*. New York, NY: John Wiley & Sons, Inc., 1995.

[126] Z. Jelinski and P. B. Moranda, "Software reliability research," in *Proc. Statistical Computer Performance Evaluation*, Providence, RI, USA, 1971, pp. 465-484.

[127] P. Tomaszewski, J. Håkansson, H. Grahn, and L. Lundberg, "Statistical models vs. expert estimation for fault prediction in modified code - an industrial case study," *Journal of Systems and Software,* vol. 80 (8), pp. 1227-1238, 2007.

[128] T. M. Khoshgoftaar and K. Gao, "Count models for software quality estimation," *IEEE Trans. Reliability,* vol. 56 (2), pp. 212-222, 2007.

[129] S. Dick and A. Sadia, "Fuzzy Clustering of Open-Source Software Quality Data: A Case Study of Mozilla," in *Proc. Int. Joint. C. Neural Networks* Vancouver, BC, 2006, pp. 4089-4096.

[130] G. Boetticher, T. Menzies, and T. Ostrand, "PROMISE Repository of empirical software engineering data," Department of Computer Science, West Virginia University, http://promisedata.org/repository, 2008.

[131] C. McNemar, "NASA/WVU IV&V Facility Metrics Data Program," NASA Independent Verification & Validation Facility: http://mdp.ivv.nasa.gov, 2004.

[132] Staff, "OO Java metrics," NASA Software Technology Assurance technology Center: http://satc.gsfc.nasa.gov/metrics/codemetrics/oo/java/index.html, 1999.

[133] N. Wilde, P. Matthews, and R. Huitt, "Maintaining object-oriented software," *IEEE Software*, vol. 10 (1), pp. 75-80, 1993.

[134] A. Mockus, T. Roy, and J. D. Herbsleb, "Two case studies of open source software development: Apache and mozilla," *ACM Trans. Soft. Eng. Meth.*, vol. 11 (3), pp. 309-346, July 2002.

[135] Mozilla.org, "Mozilla.org – Home of the Mozilla Project," The Mozilla Foundation: http://www.mozilla.org, 2006.

[136] Staff, "Browser market share white paper," Janco Associates Inc.: http://www.e-janco.com/browser.htm, 2009.

[137] O. Alonso, P. T. Devanbu, and M. Gertz, "Database techniques for the analysis and exploration of software repositories," in *Proc. Int. Wkshp Mining Software Repositories*, Edinburgh, Scotland, 2004, pp. 37-41.

[138] S. McLellan, A. Roesler, Z. Fei, S. Chandran, and C. Spinuzzi, "Experience using web-based shotgun measures for large-scale system characterization and improvement," *IEEE Trans. Soft. Eng*, vol. 24 (4), pp. 268-277, 1998.

[139] M. Fischer, M. Pinzger, and H. Gall, "Analyzing and relating bug report data for feature tracking," in *Proc. 10th Working Conference on Reverse Engineering*, Victoria, BC. Canada, 2003, pp. 90-99.

[140] Staff, "Power Software - Products - Krakatau Professional," Power Software, http://www.powersoftware.com/kp/, 2008.

[141] S. Dick, A. Meeks, M. Last, H. Bunke, and A. Kandel, "Data Mining in Software Metrics Databases," *Fuzzy Sets and Systems,* vol. 145 pp. 81-110, 2004.

[142] C.-P. Chang and C.-P. Chu, "Defect prevention in software processes: An action-based approach," *Journal of Systems and Software,* vol. 80 (4), pp. 559-570, 2007.

[143] S. Dick and A. Kandel, "The Use of Resampling in Software Metrics Datasets," in *Artificial Intelligence Methods in Software Testing*, A. K. M. Last, and H. Bunke, Ed. Singapore: World Sci. Pub. Co., 2004, pp. 175-208.

[144] K. Kaminsky and G. D. Boetticher, "Better Software Defect Prediction using Equalized Learning with Machine Learners," in *Proc. Knowledge Sharing and Collaborative Engineering* St. Thomas, US Virgin Islands, 2004.

[145] I. Gondra, "Applying machine learning to software fault-proneness prediction," *Journal of Systems and Software,* vol. 81 (2), pp. 186-195, 2008.

[146] Q. P. Hu, Y. S. Dai, M. Xie, and S. H. Ng, "Early software reliability prediction with extended ANN model," in *Proc. Int. Computer Software and Applications Conf.* , Chicago, IL, USA, 2006, pp. 234-239.

[147] M. M. T. Thwin and T.-S. Quah, "Application of neural networks for software quality prediction using object-oriented metrics," *Journal of Systems and Software,* vol. 76 (2), pp. 147-156, 2005.

[148] N. Karunanithi and Y. K. Malaiya, "Neural networks for software reliability engineering," in *Handbook of Software Reliability Engineering*, M. R. Lyu, Ed. New York, NY, USA: McGraw-Hill, 1996, pp. 699-728.

[149] T. M. Khoshgoftaar and R. M. Szabo, "Using neural networks to predict software faults during testing," *IEEE Trans. Reliability,* vol. 45 (3), pp. 456-462, 1996.

[150] M. Shin and A. L. Goel, "Knowledge discovery and validation in software metrics databases," *Proc. SPIE,* vol. 3695 pp. 226-233, 1999.

[151] A. R. Gray, "A simulation-based comparison of empirical modeling techniques for software metric models of development effort," in *Proc. Int. C. Neural Information Processing*, Perth, Australia, 1999, pp. 526-531.

[152] E. Baisch and T. Liedtke, "Comparison of conventional approaches and soft-computing approaches for software quality prediction," in *Proc. Int. C. Syst., Man, Cybern*, Orlando, FL, USA, 1997, pp. 1045-1049.

[153] E. Baisch, T. Bleile, and R. Belschner, "A neural fuzzy system to evaluate software development productivity," in *Proc. Int. C. Syst., Man, Cybern*, Vancouver, BC, Canada, 1995, pp. 4603-4608.

[154] J. S. Mertoguno, R. Paul, N. G. Bourbakis, and C. V. Ramamoorthy, "A neuro-expert system for the prediction of software metrics," *Eng. Apps. Art. Int.*, vol. 9 pp. 153-161, 1996.

[155] C. Ebert, "Fuzzy classification for software criticality analysis," *Expert Sys. With Apps*, vol. 11 pp. 323-342, 1996.

[156] C. Ebert and E. Baisch, "Knowledge-based techniques for software quality management," in *Computational Intelligence in Software Engineering*, W. Pedrycz and J. F. Peters, Eds. River Edge, NJ, USA: World Scientific, 1998, pp. 295-320.

[157] T. M. Khoshgoftaar, M. P. Evett, E. B. Allen, and P. D. Chien, "An application of genetic programming to software quality prediction," in *Computational Intelligence in Software Engineering*, W. Pedrycz and J. F. Peters, Eds. River Edge, NJ, USA: World Scientific, 1998, pp. 176-195.

[158] T. M. Khoshgoftaar and Y. Liu, "A Multi-Objective Software Quality Classification Model Using Genetic Programming," *IEEE Trans. Reliability*, vol. 56 (2), pp. 237-245, 2007.

[159] X. Yuan, T. M. Khoshgoftaar, E. B. Allen, and K. Ganesan, "An application of fuzzy clustering to software quality prediction," in *Proc IEEE Symp. App. Spc. Soft. Eng. Tech.*, Richardson, TX, USA, 2000, pp. 85-90.

[160] N. Seliya and T. M. Khoshgoftaar, "Software Quality Analysis of Unlabeled Program Modules With Semisupervised Clustering," *IEEE Trans. Systems, Man and Cybernetics, Part A: Systems and Humans,* vol. 37 (2), pp. 201-211, 2007.

[161] M. A. D. Almeida, H. Lounis, and W. L. Melo, "An investigation on the use of machine learned models for estimating software correctability," *Int. J. Soft. Eng. Knowl. Eng.,* vol. 9 pp. 185-215, 1997.

[162] T. M. Khoshgoftaar, E. B. Allen, W. D. Jones, and J. P. Hudepohl, "Data mining for predictors of software quality," *Int. J. Soft. Eng. Knowl. Eng.,* vol. 9 pp. 547-563, 1999.

[163] T. M. Khoshgoftaar, E. B. Allen, W. D. Jones, and J. P. A. Hudepohl, "Classification-tree models of software-quality over multiple releases," *IEEE Trans. Reliability,* vol. 49 (1), pp. 4-11, 2000.

[164] Y. Bo and L. Xiang, "A study on software reliability prediction based on support vector machines," in *IEEE Int. C. Industrial Engineering and Engineering Management*, Singapore, 2007, pp. 1176-1180.

[165] K. O. Elish and M. O. Elish, "Predicting defect-prone software modules using support vector machines," *Journal of Systems and Software,* vol. 81 (5), pp. 649-660, 2008.

[166] P.-F. Pai and W.-C. Hong, "Software reliability forecasting by support vector machines with simulated annealing algorithms," *Journal of Systems and Software,* vol. 79 (6), pp. 747-755, 2006.

[167] O. Vandecruys, D. Martens, B. Baesens, C. Mues, M. De Backer, and R. Haesen, "Mining software repositories for comprehensible software fault prediction models," *Journal of Systems and Software,* vol. 81 (5), pp. 823-839, 2008.

[168] N. Raj Kiran and V. Ravi, "Software reliability prediction by soft computing techniques," *Journal of Systems and Software,* vol. 81 (4), pp. 576-583, 2008.

[169] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," *IEEE Trans. Software Engineering*, vol. 33 (1), pp. 2-13, 2007.

[170] C. Catala and B. Dirib, "A systematic review of software fault prediction studies " *Expert Systems with Applications*, vol. 36 (4), pp. 7346-7354, 2008.

[171] T. M. Khoshgoftaar, E. Geleyn, L. Nguyen, and L. Bullard, "Cost-Sensitive Boosting in Software Quality Modeling," in *Proc. IEEE Int. Symp. High Assurance Systems Engineering*, Tokyo, Japan, 2002, pp. 51-60.

[172] C. X. Ling, S. Sheng, T. Bruckhaus, and N. H. Madhavji, "Predicting Software Escalations with Maximum ROI," in *Proc. IEEE Int. C. Data Mining*, Houston, TX, USA, 2005, pp. 717-720.

[173] C. Seiffert, T. M. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "A Comparative Study of Data Sampling and Cost Sensitive Learning," in *IEEE International Conference on Data Mining Workshops*, Pisa, Italy, 2008, pp. 46-52.

[174] W. DeVilbiss, "A Comparison of Software Complexity of Programs Developed Using Structured Techniques and Object-Oriented Techniques," Master's Thesis thesis, University of Wisconsin-Milwaukee, 1993

[175] R. K. Lind, "An Experimental Study of Software Metrics and Their Relationship to Software Errors," Master's thesis, University of Wisconsin-Milwaukee, 1986

[176] M. R. Lyu, "Data and Tool Disk," in *Handbook of Software Reliability Engineering*, M. R. Lyu, Ed. New York, NY: McGraw-Hill, 1996.

[177] C. Seiffert, T. M. Khoshgoftaar, and J. Van Hulse, "Improving Software-Quality Predictions With Data Sampling and Boosting," *IEEE Trans. Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 39 (6), pp. 1283-1294, 2009.

[178] H. Haibo and E. A. Garcia, "Learning from Imbalanced Data," *IEEE Trans. Knowledge and Data Engineering,* vol. 21 (9), pp. 1263-1284, 2009.

[179] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental Perspectives on Learning from Imbalanced Data," in *Proc. Int. C. Machine Learning*, Corvalis, OR, USA, 2007, pp. 935-942.

[180] J. Kam and S. Dick, "Comparing nearest-neighbour search strategies in the SMOTE algorithm," *Canadian J. Electrical and Computer Engineering,* vol. 31 (4), pp. 203-210, 2006.

[181] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM,* vol. 18 (9), pp. 509-517, 1975.

[182] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, NY: John Wiley & Sons, Inc., 2001.

[183] P. G. Mathews, *Design of Experiments with MINITAB*. Milwaukee, WI, USA: Quality Press, 2005.

[184] M. G. Marasinghe and W. J. Kennedy, *SAS for Data Analysis: Intermediate Statistical Methods*. New York, NY: Springer Science+Business Media LLC, 2008.

[185] I. Myrtveit, E. Stensrud, and M. Shepperd, "Reliability and validity in comparative studies of software prediction models," *IEEE Transactions on Software Engineering,* vol. 31 (no. 5), pp. 380-391, 2005.

[186] L. Rourke, T. Anderson, D. R. Garrison, and W. Archer, "Methodological Issues in the Content Analysis of Computer Conference Transcripts," *Int. J. Artificial Intelligence in Education,* vol. 12 pp. 8-22, 2001.

[187] C. Seiffert, T. M. Khoshgoftaar, and J. Van Hulse, "Hybrid sampling for imbalanced data," in *Proc. IEEE Int. C. Information Reuse and Integration*, Las Vegas, NV, USA, 2008, pp. 202-207.

[188] T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Comparing boosting and bagging techniques with noisy and imbalanced data,"

Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, vol. 41 (3), pp. 552-568, 2011.

[189] J. Long, "Metrics Data Program," NASA IV&V Facility, http://mdp.ivv.nasa.gov/index.html., 2008.

[190] M. L. Berenson, D. M. Levine, and M. Goldstein, *Intermediate statistical methods and applications: A computer package approach*: Prentice-Hall, 1983.

[191] L. Pelayo, A. Sadia, and S. Dick, "Predicting Object-Oriented Software Quality: A Case Study," *International Journal of Intelligent Control and Systems (IJICS),* 2009.

[192] C. L. Blake and C. J. Merz, "UCI Repository of Machine Learning Databases," Irvine, CA: Department of Computer Science, University of California, 1998.

[193] A. Estrabooks, T. Jo, and N. Japkowicz, "A Multiple Resampling Method for Learning from Imbalanced Datasets," *Computational Intelligence,* vol. 20 (1), pp. 18-36, 2004.

[194] C. Drummond and R. C. Holte, "C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," in *ICML Workshop on Learning from Imbalanced Datasets II* Washington, DC, USA, 2003.

[195] C. Drummond and R. C. Holte, "Explicitly representing expected cost: An alternative to ROC representation," in *Proc. Int. C. Knowledge Discover and Data Mining*, Boston, MA, USA, 2000, pp. 198-207.

[196] M. Maloof, "Learning when data sets are imbalanced and when costs are unequal and unknown," in *Int. C. Machine Learning Wkshp. on Learning From Imbalanced Datasets* Washington, DC, USA, 2003.

[197] R. Barandela, R. M. Valdovinos, J. S. Sánchez, and F. J. Ferri, "The Imbalanced Training Sample Problem: Under or over Sampling?," *Lecture Notes in Computer Science,* vol. 3138 pp. 806-814, 2004.

[198] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: Improving Classification Performance when Training Data is Skewed," in *Proc. Int. C. Pattern Recognition*, Tampa, FL, USA, 2008, pp. 1-4.

[199] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A Hybrid Approach to Alleviating Class Imbalance," *IEEE Trans. Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40 (1), pp. 185-197, 2010.

[200] F. Vella, "Estimating Models with Sample Selection Bias: A Survey," *The Journal of Human Resources*, vol. 33 (1), pp. 127-169, 1998.

[201] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques, 2nd Ed.* San Francisco, CA, USA: Morgan Kaufmann, 2005.

[202] P. J. Olver and C. Shakiban, *Applied Linear Algebra*. Upper Saddle River, N.J.: Prentice-Hall, 2006.

[203] S. Sýkora, "Volume Integrals over n-Dimensional Ellipsoids." vol. 1: Stan's Library, Editor S.Sykora, 2005.

[204] R. Camacho, "Investigations of Rui Camacho," http://www.liaad.up.pt/~tau/data.html, 1997.

[205] Delve, "Delve datasets," http://www.cs.toronto.edu/~delve/data/datasets.html.

[206] K. D. Kedarasetti, L. Kurgan, and S. Dick, "Classifier ensembles for protein structural class prediction with varying homology," *Biochemical and Biophysical Research Communications*, vol. 348 (3), pp. 981-988, 2006.

[207] L. Kurgan, A. A. Razib, S. Aghakhani, S. Dick, M. J. Mizianty, and S. Jahandideh, "CRYSTALP2: Sequence-based Protein Crystallization Propensity Prediction," *BMC Structural Biology*, vol. 9 (1), pp. 1-14, 2009.

[208] K. P. Wu and S. D. Wang, "Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space," *Pattern Recognition*, vol. 42 (5), pp. 710-717, 2009.

[209] C. L. Huang and C. J. Wang, "A GA-based feature selection and parameters optimizationfor support vector machines," *Expert Systems with applications*, vol. 31 (2), pp. 231-240, 2006.

[210] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to the SMO Algorithm for SVM Regression.," *IEEE Transactions on Neural Networks*, 1999.

[211] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, vol. 11 (1), 2009.

[212] L. Pelayo and S. Dick, "Numeric SMOTE raw data," University of Alberta: http://www.ece.ualberta.ca/~dick/datasets/NumericSmoteRawData.csv, 2011.

[213] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support Vector Regression Machines," *Advances in Neural Information Processing Systems 9, NIPS*, pp. 155-161, 1997.

[214] C.-C. Chang and C.-J. Lin, "LIBSVM : a library for support vector machines," Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[215] L. F. Lee, "Some approaches to the correction of selectivity bias," *The Review of Economic Studies*, vol. 49 (3), pp. 355-372, 1982.

[216] L. F. Lee, "Generalized econometric models with selectivity," *Econometrica: Journal of the Econometric Society*, vol. 51 (2), pp. 507-512, 1983.

[217] G. Cuddeback, E. Wilson, J. G. Orme, and T. Combs-Orme, "Detecting and statistically correcting sample selection bias," *Journal of Social Service Research*, vol. 30 (3), pp. 19-33, 2004.

[218] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*, 2nd ed.: Wiley, 2002.

[219] M. Hein, "Binary Classification under Sample Selection Bias," in *Dataset Shift in Machine Learning*
J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, Eds. Cambridge, MA: MIT Press, 2009, p. 248.

[220] M. Dudík, R. E. Schapire, and S. J. Phillips, "Correcting sample selection bias in maximum entropy density estimation," *Advances in neural information processing systems*, vol. 17 2005.

[221] S. Singhi and H. Liu, "Feature subset selection bias for classification learning," in *Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, Pennsylvania, 2006.

[222] J. Ren, X. Shi, W. Fan, and P. S. Yu, "Type-independent correction of sample selection bias via structural discovery and re-balancing," in *Proceedings of the Eighth SIAM International Conference on Data Mining*, Atlanta, Georgia, USA, 2008, pp. 565–576.

[223] A. Margolis, "A Literature Review of Domain Adaptation with Unlabeled Data," University of Washington, http://ssli.ee.washington.edu/~amargoli/review_Mar23.pdf, 2011.

[224] W. Fan and I. Davidson, "Reverse testing: an efficient framework to select amongst classifiers under sample selection bias," in *12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, 2006, pp. 147-156.

[225] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Statistical Planning & Inference*, vol. 90 (2), pp. 227-244, 2000.

Appendix 1

Chapter 3 raw data

Appendix 1 contains the raw data used on the experiments with the Mozilla dataset in Chapter 3. The raw data for the function approximation experiments with the original dataset is displayed first, followed by the reduced dataset after applying principal component analysis. The data for each classification algorithm from the classification experiment results are presented after the function approximation experiment data.

The structure of Appendix 1 is:

- Function approximation results with SMO Regression
    - Original dataset
    - Reduced dataset
- Classification results
    - J48 Decision trees
    - J48 Unprunned Decision trees
    - RIPPER
    - KStar
    - Radial Basis Functions
    - Random Forests
    - Support Vector Machines

Function approximation results with SMO Regression

The data are divided into 10 folds used for our 10-fold cross validation procedure. In each column, the first number is the real value, followed by the second number which is the predicted value.

Original dataset

Fold 1	0.000, 0.129	0.000, 0.124	0.000, 0.133	0.000, 0.120
0.000, 0.080	0.000, 0.121	0.000, 0.131	0.000, 0.090	0.000, 0.128
0.000, 0.134	0.000, 0.121	0.000, 0.132	0.000, 0.064	0.000, 0.080
0.000, 0.115	0.000, 0.133	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.118	0.000, 0.121
0.000, 0.105	5.400, 0.124	0.000, 0.115	0.000, 0.088	0.000, 0.126
0.000, 0.124	0.000, 0.153	0.000, 0.133	0.000, 0.121	0.000, 0.131
0.000, 0.121	0.000, 0.132	0.000, 0.121	0.000, 0.146	0.000, 0.110
0.000, -0.140	0.000, 0.121	0.000, 0.106	0.000, 0.121	0.000, 0.123
2.000, 0.114	0.000, 0.130	0.000, 0.092	0.000, 0.000	2.844, 0.153
0.000, 0.122	0.000, 0.120	0.000, 0.153	0.000, 0.121	9.170, 0.088
0.000, 0.126	0.000, 0.050	0.000, 0.121	0.000, 0.144	0.000, 0.121
0.000, 0.121	0.000, 0.121	0.000, 0.068	0.000, 0.121	0.000, 0.115
0.000, 0.088	0.000, 0.141	0.000, 0.068	0.000, 0.121	0.000, 0.119
0.000, 0.121	0.000, 0.116	0.000, 0.121	0.000, 0.112	0.000, 0.115
0.000, 0.121	0.000, 0.127	0.000, 0.120	0.000, 0.113	0.000, 0.114
0.000, 0.113	0.000, 0.121	0.000, 0.150	0.000, 0.121	0.000, 0.134
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.145	0.000, 0.121
0.000, 0.141	0.000, 0.121	0.000, 0.127	0.000, 0.121	0.000, 0.121
0.000, 0.109	0.000, 0.133	0.000, 0.117	0.180, 0.103	0.328, 0.123
0.000, 0.121	0.000, 0.060	0.000, 0.097	0.000, 0.121	0.000, 1.990
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.153	0.000, 0.121
0.000, 0.133	0.000, 0.109	0.000, 0.134	0.000, 0.115	0.465, 0.115
0.000, 0.121	0.000, 0.113	0.000, 0.121	0.000, 0.153	0.000, 0.111
0.000, 0.121	0.000, 0.121	0.000, 0.127	0.000, 0.120	0.000, 0.121
0.000, 0.114	0.000, 0.121	0.000, 0.121	0.000, 0.121	8.000, 0.118
0.000, 0.035	0.000, 0.126	0.000, 0.121	0.000, 0.125	0.000, 0.119
0.000, 0.104	0.000, 0.122	0.000, 0.127	0.000, 0.121	0.000, 0.121
0.000, 0.121	0.000, 0.122	0.000, 0.139	0.000, 0.121	0.000, 0.088
0.000, 0.115	0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.125
0.000, 0.121	0.000, 0.121	0.000, 0.108	0.000, 0.134	0.000, 0.122
0.456, 0.141	0.000, 0.118	0.000, 0.127	0.000, 0.120	0.000, 0.063

0.000, 0.145	0.000, 0.104	0.000, 0.122	0.000, 0.121	0.000, 0.123
0.000, 0.104	0.000, 0.070	0.000, 0.111	0.000, 0.097	0.000, 0.140
0.000, 0.121	0.000, 0.121	0.000, 0.126	0.000, 0.121	0.000, 0.121
0.000, 0.133	0.000, 0.121	0.000, 0.064	0.000, 0.133	0.000, 0.121
18.240, 0.138	0.000, 0.121	0.000, 0.121	0.000, 0.129	0.000, 0.120
0.000, 0.114	0.000, 0.143	0.000, 0.129	7.325, 0.126	0.000, 0.080
0.000, 0.091	0.000, 0.114	0.000, 0.123	0.000, 0.142	0.000, 0.116
0.000, 0.121	0.000, 0.109	0.000, 0.121	0.000, 0.153	0.000, 0.137
0.000, 0.194	0.000, 0.140	0.000, 0.121	0.000, 0.121	0.000, 0.117
0.000, 0.121	0.000, 0.121	0.000, 0.125	0.000, 0.126	0.000, 0.139
0.000, 0.146	0.000, 0.121	0.000, 0.127	0.000, 0.117	0.000, 0.130
0.000, 0.002	0.000, 0.121	0.000, 0.114	0.000, 0.105	0.000, 0.128
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.129	0.000, 0.102
0.000, 0.121	0.000, 0.123	0.000, 0.119	0.000, 0.148	0.000, 0.130
0.000, 0.114	0.000, 0.138	0.000, 0.111	0.000, 0.126	0.000, 0.107
0.000, 0.085	0.000, 0.117	0.000, 0.121	0.000, 0.121	0.000, 0.137
0.000, 0.121	0.000, 0.121	0.000, 0.118	0.000, 0.161	0.000, 0.121
0.000, 0.142	0.000, 0.121	0.000, 0.141	0.000, 0.153	0.000, 0.146
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.103	0.000, 0.121
0.000, 0.132	0.000, 0.120	0.000, 0.118	0.000, 0.131	0.000, 0.120
0.000, 0.138	0.000, 0.121	0.000, 0.109	0.000, 0.100	0.000, 0.121
0.090, 0.133	0.000, 0.030	0.000, 0.153	0.000, 0.121	0.000, 0.119
0.000, 0.122	0.000, 0.143	0.000, 0.121	0.000, 0.132	0.000, 0.121
0.000, 0.121	0.000, 0.121	0.000, 0.077	0.000, 0.121	0.000, 0.121
1.000, 0.119	0.000, 0.121	0.000, 0.122	0.000, 0.121	1.000, 0.078
0.000, 0.121	0.000, 0.126	0.000, 0.121	0.000, 0.149	0.000, 0.121
0.000, 0.127	0.000, 0.121	0.000, 0.122	0.000, 0.121	0.000, 0.121
0.000, 0.121	0.000, 0.148	0.000, 0.102	0.000, 0.121	0.000, 0.121
2.000, 0.116	0.000, 0.121	0.000, 0.130	0.000, 0.121	0.000, 0.121
0.000, 0.121	0.000, -0.024	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.124	0.000, 0.121	0.000, 0.134	2.000, 0.117	0.000, 0.121
0.000, 0.136	0.000, 0.121	0.000, 0.117	0.000, 0.130	0.000, 0.117
0.000, 0.121	0.000, 0.121	0.000, 0.128	0.000, 0.139	0.000, 0.151
0.000, 0.121	0.000, 0.130	0.000, 0.127	0.000, 0.134	0.000, 0.126
0.000, 0.121	0.000, 0.137	0.000, 0.129	0.000, 0.126	0.000, 0.133
0.000, 0.153	0.000, 0.153	0.000, 0.144	0.000, 0.122	0.000, 0.112
0.000, 0.121	0.474, 0.130	0.000, 0.121	0.000, 0.126	0.000, 0.119
0.000, 0.148	0.000, 0.107	0.000, 0.124	0.000, 0.121	0.000, 0.114
0.000, 0.121	0.000, 0.121	0.000, 0.132	0.000, 0.121	0.000, 0.127
2.000, -0.011	0.000, 0.121	0.000, 0.152	0.000, 0.089	0.000, 0.121
0.000, 0.100	0.000, 0.121	0.000, 0.079	0.000, 0.121	0.000, 0.121
0.000, 0.177	0.000, 0.121	0.000, 0.121	0.000, 0.111	0.000, 0.121
0.000, 0.155	0.000, 0.098	0.000, 0.129	0.000, 0.119	0.000, 0.114
0.000, 0.128	0.000, 0.114	0.000, 0.121	0.000, 0.121	0.000, 0.133
0.000, 0.121	0.000, 0.097	0.000, 0.119	0.000, 0.153	0.000, 0.116
0.000, 0.140	0.000, 0.137	0.000, 0.121	0.000, 0.150	0.000, 0.139

2.000, 0.118	0.000, 0.131	0.000, 0.130	0.000, 0.121	0.000, 0.121
0.000, 0.128	0.000, 0.090	0.000, 0.083	0.000, 0.030	0.000, 0.129
0.000, 0.134	0.000, 0.084	0.000, 0.123	0.000, 0.134	1.000, 0.134
0.219, 0.153	0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.097	0.000, 0.121	0.000, 0.126	0.000, 0.121	0.000, 0.183
0.000, 0.121	0.000, 0.117	0.000, 0.126	0.000, 0.141	0.000, 0.019
0.000, 0.121	0.000, 0.109	0.000, 0.121	0.111, 0.120	0.000, 0.132
0.000, 0.128	0.000, 0.121	0.000, 0.130	0.000, 0.118	0.000, 0.134
0.000, 0.119	0.000, 0.133	0.000, 0.121	0.281, 0.133	0.000, 0.121
0.100, 0.144	0.000, 0.121	0.000, 0.124	0.214, 0.153	0.000, 0.112
0.000, 0.122	0.000, 0.135	0.000, 0.121	0.000, 0.118	0.000, 0.116
0.000, 0.121	0.000, 0.119	0.000, 0.153	0.000, 0.118	0.000, 0.123
0.000, 0.123	0.000, 0.118	0.000, 0.086	2.000, 0.116	0.000, 0.121
0.000, 0.123	0.000, 0.121	0.000, 0.099	0.000, 0.121	0.000, 0.134
0.000, 0.121	0.000, 0.119	0.000, 0.112	0.000, 0.089	0.000, 0.115
0.000, 0.133	0.000, 0.130	0.000, 0.136	0.000, 0.121	0.000, 0.121
0.000, 0.120	0.000, 0.118	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.116	12.727, 0.056	0.000, 0.121	0.000, 0.035	0.000, 0.121
0.000, 0.100	0.000, 0.121	0.000, 0.137	0.000, 0.121	0.000, 0.081
0.000, 0.144	0.000, 0.125	0.000, 0.128	0.000, 0.139	0.000, 0.133
0.000, 0.121	0.000, 0.129	0.000, 0.117	0.000, 0.096	0.000, 0.121
0.000, 0.094	0.000, 0.109	0.000, -0.001	0.000, 0.121	0.000, 0.121
0.000, 0.129	0.000, 0.121	0.000, 0.148	0.000, 0.138	0.000, 0.138
0.000, 0.060	0.000, 0.124	0.000, 0.121	0.000, 0.153	0.000, 0.121
0.000, 0.121	0.000, 0.136	0.000, 0.121	0.000, 0.140	5.136, 0.142
1.000, 0.153	0.000, 0.121	0.000, -0.025	0.000, 0.121	0.110, 0.142
0.000, 0.123	1.000, 0.105	0.000, 0.132	0.000, 0.082	1.480, 0.151
0.000, 0.121	0.000, 0.115	0.825, -0.077	0.000, -0.056	0.000, 0.110
0.000, 0.121	0.000, 0.126	0.000, 0.088	0.000, 0.121	0.000, 0.086
0.000, 0.132	0.000, 0.108	0.000, 0.122	0.000, 0.122	0.000, 0.187
0.000, 0.116	0.000, 0.121	0.110, 0.115	0.000, 0.182	0.000, 0.112
0.000, 0.129	0.000, 0.153	2.000, 0.119	0.000, 0.125	0.000, 0.113
0.000, 0.121	0.000, 0.127	0.000, 0.121	0.514, 0.118	0.000, 0.121
0.000, 0.121	0.000, 0.121	0.000, 0.131	0.000, 0.090	0.000, 0.116
0.000, 0.123	0.000, 0.126	0.000, 0.121	0.000, 0.136	0.000, 0.121
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.120	0.000, 0.121
0.000, 0.122	0.000, 0.121	0.000, 0.086	0.000, 0.121	0.000, 0.126
0.233, 0.121	0.000, 0.117	0.000, 0.119	0.000, 0.118	0.000, 0.121
0.000, 0.121	0.000, 0.133	0.922, 0.112	0.000, 0.121	0.000, 0.132
0.000, 0.121	0.000, 0.121	0.000, 0.129	0.000, 0.115	0.000, 0.121
0.000, 0.111	0.000, 0.127	0.000, 0.121	0.000, 0.121	0.000, 0.112
1.000, 0.130	0.000, 0.121	0.000, 0.121	0.000, 0.140	0.000, 0.135
0.000, 0.122	0.000, 0.133	0.000, 0.122	0.000, 0.192	0.000, 0.123
0.000, 0.121	0.000, 0.121	0.000, 0.115	0.000, 0.111	0.000, 0.121
0.000, 0.097	0.000, 0.120	0.000, 0.157	0.000, 0.109	0.000, 0.096
0.000, 0.111	0.000, 0.138	0.000, 0.146	0.000, 0.135	0.000, 0.121

3.220, 0.103	0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.150
0.000, 0.105	0.000, 0.052	0.000, 0.121	0.000, -0.026	0.000, 0.136
0.000, 0.130	0.000, 0.125	0.000, 0.123	0.000, 0.121	3.270, 0.126
0.000, 0.114	0.000, 0.138	0.000, 0.121	0.000, 0.106	0.000, 0.121
0.000, 0.121	0.000, 0.094	0.000, 0.121	0.000, 0.121	0.000, 0.119
0.000, 0.121	0.000, 0.085	0.000, 0.121	0.000, 0.133	0.000, 0.121
0.000, 0.149	0.000, 0.117	0.000, 0.141	0.000, 0.121	0.000, 0.126
0.000, 0.112	0.000, 0.123	0.000, 0.139	0.000, 0.090	0.000, 0.108
0.000, 0.096	0.000, 0.154	0.000, 0.105	0.000, 0.141	0.000, 0.132
0.000, 0.121	0.000, 0.091	0.000, 0.130	0.000, 0.104	0.000, 0.116
0.000, 0.121	0.000, 0.146	0.000, 0.115	0.000, 0.122	0.248, 0.141
0.000, 0.110	0.000, 0.101	0.000, 0.121	0.000, 0.101	0.000, 0.131
0.000, 0.115	0.250, 0.116	0.000, 0.150	0.000, 0.119	0.000, 0.121
0.000, 0.130	0.000, 0.109	0.000, 0.124	0.000, 0.109	0.000, 0.119
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.118	0.000, 0.121	0.000, 0.095	0.000, 0.121	0.000, 0.121
0.000, 0.125	0.000, 0.121	0.000, 0.131	0.000, 0.085	0.000, 0.121
0.050, 0.153	0.000, 0.144	0.000, 0.116	0.000, 0.150	0.000, 0.121
0.000, 0.102	0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.121
0.000, 0.124	0.000, 0.121	0.000, 0.146	0.000, 0.027	0.000, 0.186
0.000, 0.121	0.000, 0.153	0.000, 0.084	0.000, 0.127	0.000, 0.121
0.000, 0.148	0.000, 0.118	0.000, 0.121	0.964, 0.115	0.000, 0.118
0.000, 0.153	0.000, 0.121	0.000, 0.153	0.000, 0.113	0.654, 0.139
0.000, 0.121	0.000, 0.069	0.171, 0.124	0.000, 0.221	0.000, 0.152
1.000, 0.205	0.000, 0.125	0.000, 0.130	0.000, 0.121	0.000, 0.039
0.000, 0.046	0.000, 0.118	0.000, 0.111	0.000, 0.131	0.000, 0.153
0.000, 0.121	0.000, 0.115	0.000, 0.121	0.000, 0.121	0.000, 0.126
0.000, 0.121	0.000, -0.026	0.000, 0.042	0.000, 0.114	0.000, 0.118
0.000, 0.121	0.000, 0.183	0.000, 0.121	0.000, 0.121	0.000, 0.115
0.000, 0.033	0.000, 0.113	0.000, 0.105	0.000, 0.129	0.000, 0.122
9.977, 0.052	2.160, 0.130	0.000, 0.121	0.000, 0.183	0.000, 0.130
0.000, 0.117	0.000, 0.121	0.000, 0.121	0.000, 0.183	0.000, 0.133
0.000, 0.126	0.086, 0.125	0.000, 0.121	0.000, 0.083	
0.000, 0.153	0.000, 0.141	0.000, 0.121	0.000, 0.059	
0.000, 0.121	0.000, 0.133	0.000, 0.121	0.000, 0.121	
0.000, 0.121	0.000, 0.121	0.000, 0.095	0.000, 0.129	
0.000, 0.121	0.000, 0.129	0.000, 0.121	0.000, 0.138	
0.000, 0.121	0.000, 0.121	0.000, 0.121	0.000, 0.102	
0.000, 0.130	0.000, 0.091	0.000, 0.146	0.000, 0.122	
0.000, 0.102	0.000, 0.121	0.000, 0.131	0.000, 0.121	
1.856, 0.135	0.000, 0.121	0.000, 0.101	0.000, 0.128	
0.000, 0.130	0.000, 0.107	0.000, 0.161	0.000, 0.121	
0.000, 0.121	0.000, 0.117	0.000, 0.100	0.000, 0.121	
6.000, 0.116	0.000, 0.108	0.000, 0.154	0.000, 0.121	
0.000, 0.121	0.000, 0.111	0.000, 0.111	0.000, 0.121	
0.000, 0.136	0.000, 0.121	0.000, 0.121	0.000, 0.138	

Fold 2	10.945, 0.100	0.000, 0.099	0.000, 0.052	0.000, 0.063
0.000, 0.380	0.000, 0.118	0.000, 0.109	0.000, 0.154	0.000, 0.118
0.000, 0.106	0.000, 0.118	0.000, 0.109	0.000, 0.118	0.000, 0.118
0.000, 0.104	0.000, 0.118	0.000, 0.115	0.000, 0.121	0.000, 0.139
0.110, 0.139	0.000, 0.130	0.000, -0.021	0.000, 0.155	0.000, 0.113
0.000, 0.118	0.000, 0.118	0.000, 0.102	0.000, 0.165	0.000, 0.151
0.000, 0.157	0.000, 0.124	0.000, 0.117	0.000, 0.125	0.000, 0.127
0.000, 0.095	0.000, 0.126	0.000, 0.118	0.000, 0.118	1.294, 0.143
4.000, 0.115	0.000, 0.118	0.000, 0.116	0.000, 0.110	0.000, 0.118
0.000, 0.083	0.000, 0.118	0.000, 0.115	0.000, 0.118	0.000, 0.065
0.000, 0.128	0.000, 0.118	0.000, 0.118	0.000, 0.034	0.000, 0.080
0.000, 0.118	0.000, 0.117	0.000, 0.125	0.000, 0.117	0.000, 0.118
0.000, 0.108	0.000, 0.122	0.000, 0.118	0.000, 0.125	0.000, 0.118
0.000, 0.118	0.000, 0.118	0.000, 0.107	0.000, 0.118	0.000, 0.118
0.000, 0.136	0.000, 0.135	0.000, 0.054	0.000, 0.129	1.000, 0.093
0.000, 0.118	0.000, 0.118	0.175, 0.053	0.000, 0.147	0.000, 0.118
0.000, 0.045	0.000, 0.118	0.000, 0.118	0.000, 0.015	0.000, 0.085
0.000, 0.118	0.000, 0.124	0.000, 0.118	0.000, 0.138	0.000, 0.118
0.000, 0.101	0.000, 0.142	0.000, 0.118	0.000, 0.164	0.000, 0.128
0.000, 0.077	0.000, 0.118	0.000, 0.131	0.000, 0.093	0.000, 0.134
0.000, 0.114	0.000, 0.144	0.000, 0.118	0.000, 0.118	0.000, 0.149
0.000, 0.118	0.000, 0.118	0.000, 0.127	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.130	0.000, 0.144	0.000, 0.126	0.000, 0.075
0.000, 0.133	0.000, 0.131	0.000, 0.133	0.000, 0.118	0.000, 0.118
0.000, 0.128	0.000, 0.118	0.000, 0.145	0.000, 0.113	0.000, 0.118
0.000, 0.132	0.000, 0.118	0.000, 0.118	0.000, 0.106	0.000, 0.157
0.000, 0.154	0.000, 0.118	0.000, -0.212	13.410, 0.096	0.000, 0.136
0.000, 0.034	1.300, 0.075	0.000, 0.132	0.000, 0.114	0.000, 0.138
0.000, 0.118	0.000, 0.132	0.000, 0.118	0.000, 0.106	0.000, 0.117
0.000, 0.119	0.000, 0.115	0.000, 0.118	0.000, 0.106	0.000, 0.154
0.000, 0.118	0.000, 0.118	0.000, 0.126	0.000, 0.131	0.000, 0.125
0.000, 0.118	0.000, 0.118	0.000, 0.139	0.000, 0.118	0.000, 0.144
0.000, 0.113	0.000, 0.093	0.000, 0.138	0.000, 0.140	0.000, 0.105
0.000, 0.070	0.000, 0.126	0.000, 0.118	0.000, 0.118	0.000, 0.105
0.000, 0.127	0.000, 0.135	0.000, 0.118	0.000, 0.134	0.000, 0.118
0.000, 0.100	0.000, 0.118	0.000, 0.121	0.000, 0.144	0.000, 0.100
0.000, 0.122	0.000, 0.130	0.000, 0.118	0.000, 0.128	0.000, 0.118
0.000, 0.118	0.000, 0.118	0.000, -0.015	0.000, 0.040	0.000, -0.036
0.000, 0.137	0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.120
0.000, 0.141	0.000, 0.125	0.000, 0.090	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.124	0.000, 0.140	0.000, 0.118	0.000, 0.082
0.000, 0.115	0.000, 0.081	0.000, 0.116	0.000, 0.118	0.000, 0.118
0.000, 0.100	0.000, -0.056	0.000, 0.094	0.000, 0.107	0.000, 0.132
0.000, 0.123	0.000, 0.132	0.000, 0.102	0.000, 0.118	0.000, 0.118
0.000, 0.143	0.000, 0.118	0.000, 0.122	0.000, 0.118	0.000, 0.101
0.000, 0.134	0.000, 0.108	3.000, 0.049	0.000, 0.144	0.000, 0.118

0.000, 0.118	0.000, 0.109	0.000, 0.087	0.000, 0.052	0.000, 0.059
3.388, -0.018	0.000, 0.117	0.000, 0.155	0.000, 0.114	0.000, 0.114
0.000, 0.062	0.000, 0.115	0.000, 0.126	0.000, 0.109	0.000, 0.118
0.000, 0.120	0.000, 0.132	0.000, 0.118	0.000, 0.109	0.000, 0.069
0.000, 0.114	2.000, 0.139	0.000, 0.122	0.000, 0.107	0.477, 0.154
0.000, 0.132	0.000, 0.048	0.000, 0.133	0.000, 0.115	0.000, 0.106
0.000, 0.096	0.000, -0.082	0.000, 0.118	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.154	0.000, 0.118	0.000, 0.056	0.000, 0.166
0.000, 0.109	0.000, 0.118	0.000, 0.122	0.000, 0.118	0.000, 0.120
0.000, 0.118	0.000, 0.129	0.000, 0.126	0.000, 0.118	0.000, 0.118
0.000, 0.116	0.000, 0.118	0.000, 0.148	0.000, 0.130	0.000, 0.118
0.000, 0.035	0.000, 0.916	0.000, 0.118	0.000, 0.119	0.000, 0.102
0.000, 0.154	0.000, 0.091	0.000, 0.118	0.000, 0.155	0.000, 0.136
0.000, 0.118	0.610, 0.057	0.000, 0.129	0.000, 0.118	0.000, 0.131
0.000, 0.118	0.000, 0.116	0.000, 0.118	0.000, 0.118	0.000, 0.105
0.000, 0.035	0.000, 0.118	0.000, 0.130	0.000, 0.145	0.000, 0.121
0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.131	0.000, 0.118
0.000, 0.130	0.000, 0.128	0.000, 0.038	0.000, 0.054	0.000, 0.118
0.000, 0.095	0.000, 0.129	0.000, 0.125	0.000, 0.080	0.000, 0.078
0.000, 0.114	0.000, 0.118	0.000, 0.111	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.118	0.000, 0.113	0.000, 0.132	0.000, 0.088
0.000, 0.117	0.000, 0.131	0.000, 0.132	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.154	0.000, 0.127	0.000, 0.118	0.000, 0.118
0.000, 0.161	0.000, 0.118	0.000, 0.050	0.000, 0.118	0.000, 0.118
0.000, 0.121	0.000, 0.136	0.000, 0.118	0.000, 0.119	0.000, 0.143
1.984, 0.109	0.000, 0.118	0.000, 0.126	0.000, 0.114	0.000, 0.122
0.000, 0.118	0.000, 0.114	0.000, 0.118	0.000, 0.154	0.000, 0.122
0.000, 0.144	0.000, 0.127	0.000, 0.128	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.123	0.000, 0.143	0.000, 0.118	0.000, 0.118
0.000, 0.810	0.000, 0.118	0.000, 0.101	0.000, 0.123	0.000, 0.113
0.000, 0.118	0.000, 0.118	0.000, 0.129	0.000, 0.108	0.000, 0.118
0.000, 0.147	0.000, 0.118	0.000, 0.142	0.000, 0.118	0.000, 0.139
4.000, 0.120	0.000, 0.114	0.000, 0.118	0.000, 0.143	0.000, 0.109
0.638, 0.122	0.000, 0.155	0.000, 0.105	0.000, 0.126	0.000, 0.058
0.000, 0.113	0.000, 0.120	0.000, 0.143	0.000, 0.077	0.000, 0.119
0.000, 0.144	0.000, 0.114	0.000, 0.083	8.200, 0.124	0.000, 0.118
0.000, 0.087	0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.154
0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.115	0.000, 0.118
0.000, 0.125	0.000, 0.121	0.000, 0.118	0.000, 0.123	0.000, 0.118
0.000, 0.079	0.000, 0.118	0.000, 0.118	0.828, 0.115	0.000, 0.138
0.000, 0.143	0.000, 0.118	0.000, 0.118	0.000, 0.154	0.000, 0.089
0.000, 0.118	0.000, 0.118	0.000, 0.151	0.000, 0.145	0.000, 0.109
0.000, 0.130	1.000, 0.117	0.000, 0.102	0.000, 0.118	0.000, 0.121
0.000, 0.143	0.000, 0.118	0.000, 0.118	2.000, 0.129	0.000, 0.124
0.000, 0.119	0.000, 0.083	0.000, 0.114	0.000, 0.118	0.000, 0.118
0.000, 0.131	0.000, 0.118	0.000, 0.119	0.000, 0.118	0.293, 0.093

0.000, 0.116	0.000, 0.126	0.000, 0.116	0.000, 0.127	0.000, 0.118
0.000, 0.098	0.000, 0.101	0.000, 0.106	0.000, 0.118	0.000, 0.102
0.000, 0.118	0.000, 0.125	0.000, 0.145	0.000, 0.117	0.000, 0.118
0.000, 0.118	0.000, 0.118	0.000, 0.099	0.000, 0.003	0.000, 0.141
0.000, 0.118	0.000, 0.132	0.000, 0.123	0.000, 0.099	0.000, 0.073
0.000, 0.104	0.000, 0.076	0.000, 0.118	0.676, 0.179	0.000, 0.118
0.000, 0.119	0.000, 0.108	0.000, 0.098	0.000, 0.129	0.000, 0.118
0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.136	1.143, 0.120
0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.118	0.000, 0.118
0.000, 0.126	0.000, 0.115	0.000, 0.136	0.000, 0.118	0.000, 0.122
0.000, 0.154	0.000, 0.118	1.000, 0.154	0.000, 0.147	0.000, 0.109
0.000, 0.228	0.000, 0.118	0.000, 0.112	0.000, 0.118	0.000, 0.114
0.976, 0.165	0.000, 0.118	0.000, 0.120	0.000, 0.130	0.000, 0.154
0.000, 0.131	0.000, 0.118	0.000, 0.118	0.000, 0.097	0.000, 0.118
0.000, 0.118	0.000, 0.150	0.000, 0.106	0.000, 0.114	0.000, 0.118
0.000, 0.153	0.000, 0.118	0.273, 0.084	0.000, 0.120	0.000, 0.118
0.000, 0.118	0.000, 0.052	0.000, 0.105	0.000, 0.154	0.000, 0.117
0.000, 0.097	0.000, 0.095	0.000, 0.119	2.000, 0.114	0.000, 0.118
0.000, 0.147	0.000, 0.133	0.000, 0.093	0.000, 0.082	0.000, 0.093
0.000, 0.118	0.000, 0.083	0.000, 0.118	0.000, 0.118	0.000, 0.130
0.000, 0.118	0.000, 0.114	0.000, 0.118	0.000, 0.126	0.000, 0.126
0.000, 0.221	0.000, 0.118	0.000, 0.107	0.000, 0.125	9.475, 0.122
0.000, 0.093	0.000, 0.131	0.000, 0.121	0.000, 0.154	0.000, 0.107
0.000, 0.060	0.000, 0.123	0.000, 0.118	0.000, 0.131	0.000, 0.115
0.000, 0.102	0.000, 0.136	0.000, 0.026	0.000, 0.118	0.000, 0.117
0.000, 0.137	0.000, 0.108	0.000, 0.095	0.000, 0.119	0.000, 0.118
0.000, 0.106	0.000, 0.126	0.000, 0.118	0.000, 0.125	0.000, 0.118
0.000, 0.104	0.000, 0.095	0.000, 0.118	0.000, 0.109	0.000, 0.143
0.000, 0.145	2.000, 0.111	0.000, 0.131	0.000, 0.128	0.000, 0.118
0.000, 0.118	0.000, 0.113	0.000, 0.132	0.000, 0.126	0.000, 0.118
0.000, 0.118	0.000, 0.103	0.000, 0.133	0.000, 0.114	0.000, 0.114
0.000, 0.154	0.000, 0.118	2.000, 0.154	0.000, 0.118	0.000, 0.130
0.000, 0.118	0.000, 0.128	0.000, 0.103	0.000, 0.108	0.000, 0.118
0.000, 0.118	0.000, 0.118	1.000, 0.143	0.000, 0.118	0.000, 0.118
0.000, 0.118	0.000, 0.106	0.000, 0.118	0.000, 0.118	0.000, 0.087
0.000, 0.122	0.000, 0.112	0.000, 0.094	0.000, 0.118	0.000, 0.118
0.000, 0.112	0.000, 0.116	0.000, 0.154	0.000, 0.104	0.000, 0.118
0.000, 0.118	0.000, 0.114	0.000, 0.128	0.000, 0.132	0.000, 0.115
0.000, 0.118	0.000, 0.118	0.000, 0.224	0.000, 0.118	0.000, 0.089
0.000, 0.132	0.000, 0.110	0.000, 0.113	0.000, -0.015	0.000, 0.124
0.000, 0.118	0.000, 0.118	0.000, 0.035	0.000, 0.118	0.000, 0.128
0.000, 0.115	0.000, 0.133	0.000, 0.082	0.000, 0.120	0.000, 0.118
0.000, 0.173	0.000, 0.118	0.000, 0.123	0.000, 0.143	0.000, 0.078
0.000, 0.121	0.000, 0.142	0.000, 0.133	0.000, 0.124	0.000, 0.151
0.000, 0.118	0.000, 0.118	2.000, 0.125	0.000, 0.146	0.000, 0.118
0.000, 0.114	0.000, 0.128	0.000, 0.118	0.000, 0.100	2.000, 2.704

0.000, 0.117	0.056, 0.127	0.000, 0.118	3.000, 0.038
0.000, 0.091	0.000, 0.123	0.000, 0.154	0.000, 0.117
0.000, 0.118	0.380, 0.012	0.000, 0.118	2.000, 0.106
0.000, 0.118	0.000, 0.099	0.320, 0.078	0.000, 1.050
0.000, 0.131	0.000, 0.118	0.000, 0.120	0.000, 0.118
0.000, 0.118	0.000, 0.154	0.000, 0.118	0.000, 0.125
0.000, 0.118	0.000, 0.118	0.000, 0.112	0.000, 0.115
0.000, 0.118	0.000, 0.118	0.000, 0.097	0.000, 0.135
0.000, 0.118	0.000, 0.118	0.000, 0.094	
0.000, 0.116	0.000, 0.118	0.000, 0.127	
0.000, 0.118	0.000, 0.118	0.000, 0.118	
0.000, 0.118	0.000, 0.118	0.000, 0.111	
0.000, 0.118	0.000, 0.128	0.000, 0.029	
0.000, 0.118	0.000, 0.138	0.000, 0.118	
0.000, 0.127	0.000, 0.116	0.000, 0.128	
0.000, 0.122	0.000, 0.115	0.000, 0.113	
0.000, 0.133	0.000, 0.138	0.000, 0.118	
0.000, -0.081	0.000, 0.119	0.000, 0.115	
0.000, 0.104	0.000, 0.137	0.000, 0.129	
0.000, 0.118	0.000, 0.131	0.000, 0.106	
0.000, 0.118	0.000, 0.118	0.000, 0.118	
0.000, 0.118	0.000, 0.091	0.000, 0.004	
0.000, 0.118	0.000, 0.118	0.000, 0.127	
0.000, 0.127	0.000, 0.118	0.000, 0.118	
0.000, 0.126	0.000, 0.118	0.000, 0.154	
0.000, 0.107	0.000, 0.121	0.000, 0.118	
0.000, 0.012	0.000, 0.102	0.000, 0.118	
0.000, 0.123	0.000, 0.118	0.000, 0.118	
0.000, 0.128	0.000, 0.112	0.000, 0.144	
0.000, 0.154	0.714, 0.162	0.000, 0.118	
0.000, 0.133	0.000, 0.110	0.000, 0.141	
0.000, 0.136	0.000, 0.159	0.000, -0.051	
0.000, 0.077	0.000, 0.118	0.000, 0.142	
0.000, 0.155	0.183, 0.125	0.000, 0.121	
0.000, 0.137	0.000, 0.118	0.000, 0.028	
0.000, 0.151	0.000, 0.118	0.000, 0.118	
0.000, 0.154	0.000, 0.118	0.000, 0.128	
0.000, 0.120	0.000, 0.118	0.000, 0.118	
0.000, 0.118	0.000, 0.118	0.000, 0.121	
0.000, 0.118	0.000, 0.063	0.000, 0.118	
0.000, 0.118	0.000, 0.100	0.000, 0.118	
0.864, 0.126	0.000, 0.171	0.000, 0.123	
0.048, 0.125	0.000, 0.066	0.000, 0.118	
0.000, 0.100	0.000, 0.127	0.000, 0.136	
0.000, 0.118	0.000, 0.118	0.000, 0.138	
0.000, 0.110	0.000, 0.125	0.000, 0.118	

Fold 3	0.000, 0.122	0.000, 0.107	0.000, 0.119	0.000, 0.124
0.000, 0.122	0.000, 0.112	0.000, 0.105	0.000, 0.122	0.000, 0.122
0.000, 0.122	5.460, 0.065	0.000, 0.122	0.000, 0.107	0.000, 0.122
0.000, 0.107	0.000, 0.122	0.000, 0.099	0.000, 0.109	0.000, 0.151
0.000, 0.110	0.000, 0.122	0.000, 0.122	0.000, 0.115	0.000, 0.122
0.000, 0.122	0.000, 0.084	0.000, 0.122	0.000, 0.129	0.000, 0.122
0.000, 0.119	0.000, 0.122	0.000, 0.122	0.000, 0.053	0.000, 0.201
0.000, 0.122	0.000, 0.107	0.000, 0.122	0.000, 0.122	0.000, 0.090
0.000, 0.103	0.000, 0.102	0.000, 0.129	0.000, 0.110	0.000, 0.103
0.000, 0.078	0.000, 0.108	0.000, 0.107	0.000, 0.089	0.000, 0.122
0.000, 0.122	0.000, 0.131	0.000, 0.122	0.000, 0.122	0.000, 0.136
0.000, 0.130	0.000, 0.122	0.000, 0.071	0.000, 0.096	0.000, 0.106
0.000, 0.122	0.000, 0.097	0.000, 0.122	0.000, 0.131	0.000, 0.128
0.000, 0.103	0.000, 0.107	0.000, 0.122	0.000, 0.100	0.000, 0.108
0.000, 0.359	0.000, 0.122	0.000, 0.109	0.000, 0.106	0.000, 0.122
0.000, 0.122	0.000, 0.081	0.000, 0.130	0.000, 0.101	0.000, 0.114
0.000, 0.122	0.000, 0.097	0.000, 0.122	0.000, 0.123	0.000, 0.097
0.350, 0.097	0.000, 0.070	0.375, 0.107	0.000, 0.122	0.000, 0.107
0.000, 0.117	0.000, 0.107	0.000, 0.152	0.000, 0.106	0.000, 0.122
0.000, 0.126	0.000, 0.109	0.000, 0.107	0.000, 0.122	0.000, 0.095
0.000, 0.119	0.000, 0.106	0.000, 0.122	0.750, 0.092	0.000, 0.122
0.000, 0.097	0.000, 0.097	0.000, 0.122	0.000, 0.107	0.000, 0.010
0.000, 0.122	0.000, 0.080	0.000, 0.122	0.000, -0.017	3.000, 0.791
0.000, 0.081	0.000, 0.040	0.000, 0.049	0.000, 0.113	0.000, 0.122
0.000, 0.093	0.000, 0.116	0.000, 0.105	0.000, 0.122	0.000, 0.131
0.000, 0.122	0.000, 0.107	0.000, 0.123	0.000, 0.122	0.000, 0.068
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.130, 0.113
0.862, 1.121	2.000, 0.113	0.368, 0.254	0.000, 0.099	0.000, 0.116
0.000, 0.115	0.000, 0.123	0.000, 0.122	0.000, 0.119	0.000, 0.122
0.000, 0.093	0.000, 0.122	0.000, 0.103	1.000, 0.098	0.000, 0.108
0.000, 0.107	0.000, 0.088	0.000, 0.084	0.000, 0.109	0.000, 0.117
0.000, -0.040	0.000, 0.122	0.000, 0.117	0.000, 0.159	0.000, 0.018
0.000, 0.122	0.000, 0.070	0.000, 0.070	0.000, 0.099	0.000, 0.122
0.000, 0.106	0.000, 0.102	0.000, 0.122	0.000, 0.064	0.000, 0.122
0.000, 0.133	0.000, 0.100	0.000, 0.091	0.000, 0.110	0.000, 0.111
0.000, 0.036	0.000, 0.128	0.000, 0.121	0.000, 0.102	0.000, 0.122
0.000, 0.123	0.000, 0.041	0.000, 0.122	0.000, 0.120	0.000, 0.077
0.000, 0.122	0.000, 0.079	0.000, 0.122	0.504, 0.113	0.000, 0.129
0.000, 0.095	0.000, 0.105	0.000, 0.112	0.000, 0.122	0.000, 0.131
0.000, 0.113	0.000, 0.104	0.000, 0.122	2.247, 0.111	0.110, 0.096
0.000, 0.122	0.000, 0.131	0.000, 0.113	0.000, 0.121	0.000, 0.126
0.000, 0.151	0.000, 0.122	0.000, 0.139	0.000, -0.151	0.000, 0.107
0.000, 0.209	0.000, 0.115	0.000, 0.062	0.110, 0.130	0.108, 0.131
0.000, 0.124	0.000, 0.108	0.000, 0.107	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.098	0.000, 0.046	0.000, 0.122	0.000, 0.122
0.000, 0.116	0.000, 0.122	0.000, 0.122	0.000, 0.112	0.000, 0.133

0.000, 0.126	0.000, 0.111	0.000, 0.076	0.000, 0.122	0.000, 0.067
0.000, 0.107	0.000, 0.110	0.000, 0.122	0.000, 0.040	0.000, 0.720
4.000, 0.102	0.000, -0.188	0.000, 0.122	0.000, 0.122	0.000, 0.080
0.000, 0.089	0.000, 0.122	0.000, 0.119	0.000, 0.122	0.000, 0.122
0.000, 0.125	0.000, 0.107	0.000, 0.122	0.000, 0.094	0.000, 0.122
0.000, 0.106	0.000, 0.122	0.000, 0.122	0.000, 0.092	0.000, 0.122
0.000, 0.120	0.000, 0.111	0.000, 0.138	0.000, 0.122	0.000, 0.122
0.000, 0.124	1.000, 0.071	0.000, 0.122	0.000, 0.110	0.000, 0.097
0.000, 0.084	0.000, 0.085	0.000, 0.109	0.000, 0.131	0.130, 0.102
0.000, 0.108	0.000, 0.130	0.000, 0.104	0.000, 0.122	0.000, 0.122
0.000, -0.037	0.000, 0.122	0.000, 0.122	0.000, 0.107	0.000, 0.121
0.000, 0.068	0.786, 0.127	0.000, 0.040	0.000, 0.122	0.000, 0.122
0.000, 0.279	0.000, 0.131	0.000, 0.125	0.000, 0.108	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.131	0.000, 0.122
0.000, 0.112	0.000, 0.122	0.000, 0.104	0.000, 0.077	0.000, 0.122
0.000, 0.122	0.000, 0.124	0.000, 0.033	0.000, 0.104	0.000, 0.106
0.000, 0.122	0.000, 0.106	0.000, 0.125	0.000, 0.115	0.000, 0.081
0.000, 0.122	0.000, 0.122	0.000, 0.175	0.000, 0.030	0.000, 0.111
0.000, 0.149	0.000, 0.081	0.000, 0.122	0.000, 0.113	0.000, 0.109
0.000, 0.122	0.000, 0.103	0.000, 0.108	0.000, 0.032	0.000, 0.122
0.000, 0.122	1.000, -0.005	0.000, 0.122	0.000, 0.114	0.000, 0.122
0.000, 0.103	0.000, 0.120	0.000, 0.122	0.000, 0.092	0.000, 0.122
0.000, 0.100	4.000, 0.126	0.000, 0.119	0.000, 0.108	0.000, 0.112
0.000, 0.141	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	4.000, 0.070
0.000, 0.107	0.000, 0.121	0.000, 0.122	0.000, 0.030	0.000, 0.122
0.000, 0.060	0.000, 0.043	0.000, 0.129	0.000, 0.106	0.000, 0.139
0.000, 0.119	0.000, 0.088	0.000, -0.115	0.000, 0.148	0.000, 0.103
0.000, -0.063	0.000, 0.122	0.000, 0.076	0.000, 0.006	0.000, 0.115
0.000, 0.113	0.000, 0.187	0.000, 0.122	0.000, 0.122	0.540, 0.119
0.000, 0.121	0.000, 0.099	0.000, 0.122	0.000, 0.097	0.000, 0.122
0.000, 0.122	0.000, 0.141	0.000, 0.122	0.000, 0.084	0.000, 0.122
0.000, 0.415	0.000, 0.093	0.000, 0.122	0.000, 0.122	0.000, 0.093
0.000, 0.083	0.472, 0.091	0.000, 0.126	0.000, 0.107	0.000, 0.122
0.000, -0.247	0.000, 0.114	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.021, 0.180	0.000, 0.096	0.000, 0.126
0.000, 0.077	0.000, 0.134	0.000, 0.082	0.819, 0.092	0.000, 0.110
0.000, 0.122	0.000, 0.121	0.000, -0.123	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.130, 0.110	0.000, 0.106	0.000, 0.113
0.000, 0.107	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.120
0.000, 0.101	0.000, 0.122	1.000, 0.123	0.000, 0.130	0.000, 0.100
0.000, 0.122	0.000, 0.106	0.000, 0.104	0.000, 0.122	0.000, 0.106
0.000, 0.121	0.000, 1.454	0.000, 0.122	1.206, 0.072	0.000, 0.106
0.000, 0.095	0.000, 0.122	0.130, 0.101	0.000, 0.103	0.000, 0.122
0.000, 0.066	0.000, 0.122	0.000, 0.122	0.000, 0.109	1.000, 0.114
0.000, 0.122	0.000, 0.048	0.000, -0.055	0.000, 0.126	0.000, 0.121

0.000, 0.094	0.000, 0.093	0.000, 0.126	0.000, 0.122	0.000, 0.120
0.000, 0.126	0.000, 0.122	0.000, 0.122	0.000, 0.124	0.000, 0.122
0.000, 0.119	0.000, 0.087	0.000, 0.123	0.000, 0.122	0.000, 0.116
0.000, -0.070	0.027, 0.116	0.000, 0.104	0.000, 0.122	0.000, 0.055
0.000, 0.093	0.000, 0.122	0.000, 0.107	0.000, 0.098	0.000, 0.096
0.000, 0.122	0.000, 0.111	0.000, 0.098	0.000, 0.122	0.000, 0.109
0.000, 0.109	0.000, 0.132	0.000, 0.122	0.000, 0.096	2.000, 0.103
0.000, 0.109	0.000, 0.107	0.000, 0.122	0.000, 0.122	0.000, 0.107
0.000, 0.122	0.000, 0.122	0.000, 0.104	0.000, 0.087	0.000, 0.092
0.000, 0.122	0.000, 0.129	0.465, 0.115	0.000, 0.104	0.000, 0.086
0.000, 0.108	0.000, 0.122	0.000, 0.107	0.000, 0.109	0.000, 0.103
0.000, 0.057	0.000, 0.107	0.000, 0.122	2.000, 0.009	0.000, 0.117
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.093	0.000, 0.099
0.000, 0.069	0.000, 0.122	0.000, 0.123	0.000, 0.121	0.000, 0.122
0.000, 0.096	0.000, 0.122	0.000, 0.084	0.000, 0.102	0.000, 0.105
2.100, 0.109	0.000, 0.101	0.000, 0.107	0.000, 0.122	0.000, 0.122
0.000, 0.090	0.000, 0.113	0.000, 0.133	0.000, 0.120	0.000, 0.110
0.000, 0.122	0.000, 0.126	0.000, 0.122	0.000, 0.107	0.000, 0.131
0.000, 0.122	0.000, 0.117	0.000, 0.112	0.000, 0.122	0.000, 0.107
0.000, 0.111	0.000, 0.122	0.000, 0.252	0.000, 0.122	0.000, 0.108
0.000, 0.122	0.000, 0.122	0.303, 0.121	0.000, 0.138	0.000, 0.098
0.000, 0.060	0.000, 0.118	0.000, 0.005	0.000, 0.121	0.000, 0.156
0.000, 0.074	0.044, 0.073	0.000, 0.066	0.000, 0.097	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 1.123	0.000, 0.099
0.000, 0.134	0.000, 0.118	0.000, 0.096	0.000, 0.088	0.000, 0.122
0.000, 0.112	0.000, 0.122	0.000, 0.130	0.000, 0.116	0.000, -0.130
0.000, 0.104	0.000, -0.001	0.000, 0.129	0.000, 0.105	0.000, 0.122
0.000, 0.065	0.000, 0.122	0.000, 0.122	0.000, -0.100	0.000, 0.089
0.000, 0.118	0.000, 0.006	0.000, 0.122	0.000, 0.118	0.000, 0.130
0.000, 0.088	0.000, 0.122	1.000, 0.097	0.000, 0.108	0.000, 0.061
0.000, 0.161	0.000, 0.104	0.000, 0.115	1.000, 0.121	0.000, 0.000
0.000, 0.042	0.000, 0.122	0.000, 0.110	0.000, 0.122	0.000, 0.106
0.000, 0.188	0.000, 0.122	0.000, 0.139	0.000, 0.093	2.080, 0.104
0.000, 0.132	0.000, 0.092	0.000, 0.123	0.000, 0.089	0.000, 0.121
0.000, 0.123	0.000, 0.093	0.000, 0.122	0.000, 0.131	0.000, 0.134
0.000, 0.080	0.000, 0.135	0.000, 0.120	0.000, 0.122	0.140, 0.105
0.000, 0.122	0.322, 0.099	0.000, 0.122	0.000, 0.114	0.000, 0.122
0.000, 0.122	0.000, 0.131	0.000, 0.122	0.000, 0.093	0.000, 0.061
0.000, 0.120	0.000, 0.107	0.000, 0.122	0.000, 0.085	0.000, 0.122
0.000, 0.122	0.000, 0.116	0.000, 0.116	0.000, 0.095	0.000, 0.122
0.000, 0.071	0.000, 0.122	0.000, 0.119	0.000, 0.103	0.000, 0.090
0.000, 0.130	0.000, 0.091	0.000, 0.122	0.000, 0.093	0.000, 0.095
0.000, 0.126	0.000, 0.111	0.000, 0.122	0.000, 0.122	0.000, 0.092
0.000, 0.107	1.404, 0.055	0.000, 0.010	0.000, 0.099	0.000, 0.110
0.000, -0.047	0.000, 0.122	0.572, 0.130	0.000, 0.132	0.070, 0.107
0.000, 0.111	0.000, 0.078	0.000, 0.179	0.000, 0.122	0.150, 0.091

0.000, 0.122	0.000, 0.112	0.000, 0.122	20.000, 0.476
0.000, 0.125	0.000, 0.138	0.000, 0.131	0.000, 0.122
2.758, 0.051	0.000, 0.122	2.200, 0.125	0.000, 0.106
0.000, 0.133	0.000, 0.122	0.000, 0.122	0.000, 0.137
0.000, 0.118	0.000, 0.093	1.000, -0.124	0.000, 0.078
0.000, 0.119	0.000, 0.104	0.000, -0.260	0.000, 0.108
0.000, 0.091	0.000, 0.122	0.000, 0.044	0.000, 0.129
0.000, 0.050	0.000, 0.122	0.000, 0.093	0.000, 0.122
0.000, 1.736	0.000, 0.122	0.000, 0.131	
0.000, 0.122	0.000, -0.074	0.000, 0.122	
0.000, 0.840	0.000, 0.129	0.000, 0.102	
0.000, 0.122	0.000, 0.054	0.000, 0.120	
0.000, 0.105	0.000, 0.122	0.000, -0.019	
0.000, 0.122	0.000, 0.099	0.000, 0.122	
0.000, 0.078	0.000, 0.055	0.000, 0.122	
0.000, 0.122	0.000, 0.116	0.000, 0.123	
0.000, 0.122	0.000, 0.122	0.000, 0.122	
0.000, 0.129	0.000, 0.090	0.000, 0.122	
0.000, 0.120	0.000, 0.122	0.000, 0.122	
1.076, 0.110	0.000, 0.115	0.000, 0.122	
0.000, 0.077	0.000, 0.115	0.000, 0.079	
0.000, 0.127	0.000, 0.089	0.000, 0.150	
0.000, 0.227	0.000, 0.122	0.000, 0.122	
0.000, 0.122	0.000, 0.090	0.000, 0.103	
0.000, 0.112	0.000, 0.086	0.000, 0.122	
0.000, 0.098	0.000, 0.090	0.000, 0.139	
0.000, 0.116	0.000, 0.106	0.000, 0.090	
1.076, 0.110	0.000, 0.122	0.000, 0.090	
0.000, 0.103	0.000, 0.122	0.000, 0.122	
0.000, 0.122	0.000, 0.122	0.000, 0.076	
0.000, 0.120	0.000, 0.122	0.000, 0.122	
0.000, -0.014	0.000, 0.064	0.000, 0.122	
0.000, 0.122	0.000, 0.112	0.000, 0.113	
0.000, 0.122	0.000, 0.122	0.000, 0.015	
0.000, 0.122	0.000, 0.106	0.000, 0.122	
0.000, 0.122	0.000, 0.110	0.000, 0.121	
0.000, 0.122	0.000, 0.107	0.000, 0.122	
0.000, 0.055	0.000, 0.122	0.000, 0.122	
0.000, 0.122	0.000, 0.122	0.000, 0.121	
0.000, 0.119	0.000, 0.079	0.000, 0.122	
0.000, 0.121	0.000, 0.105	0.334, 0.099	
0.000, 0.126	0.000, 0.107	0.000, 0.057	
0.000, 0.122	0.000, 0.110	0.000, 0.003	
0.000, 0.122	0.000, 0.104	0.000, 0.116	
0.000, 0.020	0.193, 0.098	0.000, 0.117	
0.000, 0.121	0.000, 0.122	0.000, 0.104	

Fold 4	0.000, 0.176	5.004, 0.154	0.000, 0.113	0.000, 0.113
	0.000, 0.136	0.000, 0.115	0.000, 0.167	0.000, 0.112
0.000, 0.113	0.000, 0.113	0.000, 0.137	0.000, 0.133	0.000, 0.107
0.000, 0.113	0.000, 0.370	0.000, 0.113	0.000, 0.079	0.000, 0.070
0.000, 0.113	0.000, 0.108	0.000, 0.048	0.000, 0.089	0.000, 0.122
0.000, 0.113	0.000, -0.092	0.000, 0.122	0.000, 0.113	0.000, 0.113
0.000, 0.119	0.000, 0.181	0.000, 0.109	0.000, 0.133	0.000, 0.113
0.000, 0.113	0.000, 0.165	0.000, 0.073	0.000, 0.154	0.000, 0.118
0.000, 0.108	0.000, 0.120	0.000, 0.092	0.000, 0.094	0.000, 0.113
0.000, 0.113	0.000, 0.167	0.000, 0.113	0.000, 0.138	0.000, 0.113
0.000, 0.113	0.000, 0.102	0.000, 0.113	0.000, 0.139	0.000, 0.109
0.000, 0.113	0.000, 0.143	0.000, 0.113	2.568, 0.319	0.000, 0.113
0.000, 0.127	0.000, 0.113	0.000, 0.159	0.000, 0.113	0.000, 0.113
0.000, 0.145	0.000, 0.145	0.000, 0.113	0.000, 0.113	0.000, 0.129
0.000, 0.113	0.337, 0.125	0.000, 0.113	0.000, 0.120	0.000, 0.150
0.000, 0.113	0.000, 0.119	0.000, 0.113	0.000, 0.112	0.000, 0.103
0.000, 0.128	0.000, 0.112	0.000, 0.113	0.000, 0.188	0.000, 0.115
0.000, 0.113	0.000, 0.113	0.000, 0.109	0.000, 0.091	0.000, 0.143
0.000, 0.114	0.000, 0.188	5.000, 0.110	0.000, 0.113	0.000, 0.147
0.000, 0.113	0.000, 0.107	0.000, 0.173	0.000, 0.113	0.000, 0.113
0.000, 0.083	0.000, 0.126	0.000, 0.177	0.000, 0.113	0.000, 0.113
0.000, 0.113	0.000, 0.104	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.115	0.000, 0.112	0.000, 0.133	0.000, 0.183	0.000, 0.161
0.000, 0.074	0.000, 0.113	0.000, 0.188	0.456, 0.147	0.000, 0.064
0.000, 0.113	0.000, 0.124	0.000, 0.116	0.000, 0.113	0.000, 0.166
0.000, 0.113	0.000, 0.113	0.000, 0.138	0.000, 0.113	0.000, 0.104
0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.128	0.000, 0.113
0.000, 0.113	0.000, 0.105	0.000, 0.107	0.000, 0.096	0.000, 0.113
0.000, 0.114	0.000, 0.117	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.101	0.000, 0.152	0.000, 0.146	0.000, 0.168	0.000, 0.113
12.000, 0.116	0.000, 0.124	0.000, 0.119	0.333, 0.159	0.000, 0.113
0.551, 0.119	0.000, 0.113	0.810, 0.172	0.000, 0.161	0.000, 0.120
0.000, 0.101	0.000, 0.164	0.000, 0.151	0.000, 0.132	0.000, 0.178
0.000, 0.113	0.000, 0.131	0.000, 0.103	0.500, 0.108	0.000, -0.087
0.000, 0.065	0.000, 0.156	0.000, 0.114	0.000, 0.113	0.000, 0.109
0.000, 0.113	0.470, 0.118	0.000, 0.188	0.000, 0.095	0.000, 0.121
0.000, 0.105	0.000, 0.113	0.000, 0.142	1.000, 0.171	0.000, 0.113
0.000, 0.113	0.000, 0.164	0.000, 0.112	0.000, 0.113	0.000, 0.122
0.000, 0.113	0.000, 0.113	0.000, 0.178	0.000, 0.120	0.000, 0.113
0.000, 0.147	0.000, 0.136	0.000, 0.113	0.000, 0.109	0.000, 0.086
0.000, 0.163	0.000, 0.113	1.000, 0.188	0.000, 0.113	0.000, 0.098
0.000, 0.113	0.000, 0.127	0.000, 0.117	0.000, 0.101	0.000, 0.198
0.000, 0.140	0.000, 0.139	0.000, 0.119	0.000, 0.123	0.000, 0.187
0.000, 0.114	0.000, 0.111	0.000, 0.104	0.000, 0.113	0.000, 0.113
0.000, -0.052	0.000, 0.113	0.000, 0.113	0.000, 0.100	0.000, 0.113
4.000, 0.093	0.000, 0.119	0.000, 0.112	1.036, 0.122	0.000, 0.095

0.000, 0.119	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.118
0.000, 0.128	0.000, 0.113	0.000, 0.186	0.000, 0.114	0.000, 0.116
0.000, 0.113	0.000, 0.096	0.000, 0.113	0.000, 0.188	0.000, 0.117
0.000, 0.122	0.000, 0.112	0.000, 0.108	0.000, 0.113	0.000, 0.113
0.000, 0.126	0.000, 0.104	0.000, -0.018	0.000, 0.125	0.000, 0.116
0.000, 0.120	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.122
0.000, 0.099	0.000, 0.180	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.108	0.000, 0.113	0.000, 0.105	0.000, 0.178	0.000, 0.069
0.000, 0.147	0.000, 0.073	0.000, 0.106	0.000, 0.113	0.000, 0.127
0.000, 0.136	0.000, 0.113	0.000, 0.118	0.000, 0.113	0.000, 0.140
0.000, 0.159	0.000, 0.113	0.000, 0.113	0.000, -0.180	0.000, 0.160
1.078, 0.107	0.000, 0.113	0.000, 0.148	0.000, 0.120	0.000, 0.113
0.000, 0.118	0.000, 0.112	0.000, 0.113	0.000, 0.188	0.000, 0.113
0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.159	0.000, 0.154	0.000, 0.111	0.000, 0.139	21.552, 0.035
0.000, 0.084	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.132
0.000, 0.143	0.000, 0.107	0.000, 0.113	0.000, 0.113	0.000, 0.121
0.000, 0.129	0.000, 0.113	0.000, 0.152	0.000, 0.113	0.000, 0.118
0.000, 0.134	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.121
0.000, 0.098	0.000, 0.102	0.000, 0.114	0.000, 0.113	0.000, 0.161
0.000, 0.127	0.000, 0.119	0.000, 0.095	0.000, 0.187	0.000, -0.013
0.000, 0.299	0.000, 0.138	0.000, 0.113	0.000, 0.113	0.045, 0.123
0.000, 0.116	0.000, 0.128	0.000, 0.125	0.000, 0.139	0.000, 0.113
0.000, 0.113	0.000, 0.111	0.000, 0.109	1.522, 0.153	0.000, 0.094
0.000, 0.171	0.000, 0.131	0.000, 0.113	0.000, 0.113	0.000, 0.151
0.000, 0.131	0.000, 0.102	0.000, 0.024	0.000, 0.113	0.000, 0.124
2.000, 0.110	0.000, 0.070	0.000, 0.086	0.000, 0.071	0.000, 0.113
0.000, 0.109	0.000, 0.105	0.000, 0.188	0.000, 0.119	0.000, 0.175
0.000, 0.113	0.000, 0.117	0.000, 0.113	6.000, 0.159	0.000, 0.113
0.000, 0.117	0.000, 0.155	0.000, 0.163	0.000, 0.124	0.000, 0.113
0.308, 0.113	0.000, 0.113	2.000, 0.139	0.000, 0.113	0.000, 0.157
0.000, 0.113	0.000, 0.054	0.000, 0.110	0.000, 0.113	0.000, 0.113
0.000, 0.128	0.000, 0.123	0.000, 0.174	0.000, 0.113	0.000, 0.112
0.000, 0.105	0.000, 0.147	0.726, 0.138	0.000, 0.113	0.000, 0.113
0.000, 0.176	0.000, 0.116	0.000, 0.113	0.000, 0.095	0.000, 0.120
0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.127	0.000, 0.084
0.000, 0.134	0.000, 0.113	0.000, 0.097	0.000, 0.113	0.000, 0.110
0.000, 0.113	0.000, 0.123	0.000, 0.110	0.000, 0.113	0.000, 0.119
0.000, 0.187	0.000, 0.045	0.000, 0.113	0.000, 0.116	0.000, 0.114
0.000, 0.113	0.000, 0.122	0.000, 0.086	0.180, 0.106	0.000, 0.158
0.000, 0.113	0.000, 0.115	0.000, 0.113	0.000, 0.129	0.000, 0.113
0.000, 0.114	0.000, 0.109	0.000, 0.109	0.000, 0.111	0.000, 0.120
0.000, 0.113	0.000, 0.120	0.000, 0.121	0.000, 0.113	0.000, 0.118
0.000, 0.113	0.000, 0.113	0.000, 0.120	0.000, 0.135	0.000, 0.113
0.000, 0.113	0.000, 0.113	0.000, 0.109	0.000, 0.157	0.000, 0.113
0.000, 0.118	0.000, 0.113	0.000, 0.092	0.000, 0.113	0.000, 0.102

0.000, 0.113	0.000, 0.152	0.000, 0.119	0.000, 0.004	0.000, 0.144
0.000, 0.113	0.000, 0.113	0.000, 0.107	0.000, 0.113	0.000, 0.178
0.000, 0.126	0.000, 0.133	0.000, 0.118	0.000, 0.113	0.000, 0.119
0.000, 0.154	0.000, 0.113	0.000, 0.110	1.190, 0.172	0.000, 0.113
0.000, 0.157	0.000, 0.113	0.000, 0.168	0.000, 0.111	0.000, 0.113
0.000, 0.089	0.000, 0.109	0.000, 0.116	0.000, 0.113	0.000, 0.087
0.000, 0.113	0.000, 0.094	0.000, 0.138	0.000, -0.089	0.000, 0.088
0.000, 0.112	0.000, 0.106	0.000, 0.173	0.000, 0.118	0.000, 0.120
0.000, 0.109	0.000, 0.113	0.000, 0.113	0.000, 0.109	0.000, 0.113
0.000, 0.113	0.000, 0.133	0.000, 0.113	0.000, 0.188	0.000, 0.155
0.000, 0.113	0.000, 0.134	0.000, 0.113	0.000, 0.113	0.000, 0.115
0.000, 0.113	0.000, 0.110	0.000, 0.113	0.000, 0.072	0.000, 0.132
0.000, 0.128	0.000, 0.150	0.000, 0.119	0.000, 0.110	0.000, 0.112
0.000, 0.113	0.000, 0.137	0.000, 0.122	0.000, 0.058	0.000, 0.115
0.000, 0.113	0.000, 0.141	0.000, 0.113	0.000, 0.100	0.000, 0.113
0.000, 0.188	0.000, 0.113	0.000, 0.098	0.000, 0.119	0.618, 0.109
0.000, 0.137	0.000, 0.113	0.000, 0.157	0.000, 0.188	0.000, 0.128
0.000, 0.111	0.000, 0.113	0.000, 0.100	0.000, 0.151	0.000, 0.113
0.000, 0.188	0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.113	2.850, 0.124	0.000, 0.113	0.000, 0.127	0.000, 0.113
0.000, 0.102	1.590, 0.111	0.000, 0.129	0.000, 0.113	0.000, 0.151
0.000, 0.108	0.250, 0.128	0.455, 0.163	0.000, 0.116	0.000, 0.099
0.000, 0.113	0.000, 0.155	0.000, 0.143	0.000, 0.105	0.000, 0.119
0.000, 0.104	0.000, 0.163	0.000, 0.113	0.000, 0.091	0.000, 0.145
0.000, 0.112	0.000, 0.109	0.000, 0.158	0.000, 0.113	0.000, 0.113
0.000, 0.142	0.000, 0.113	0.000, 0.174	1.000, 0.114	0.000, 0.089
0.000, 0.120	0.000, 0.122	0.000, 0.113	0.000, 0.136	0.000, 0.157
0.000, 0.147	0.000, 0.148	0.000, 0.129	0.000, 0.096	0.000, 0.188
0.000, 0.113	0.000, 0.130	0.000, 0.188	0.000, 0.121	0.000, 0.113
0.000, 0.032	0.000, 0.188	0.000, 0.123	0.000, 0.090	0.000, 0.113
0.000, 0.165	0.000, 0.128	0.000, 0.096	0.000, 0.113	0.000, 0.091
0.000, 0.113	0.000, 0.177	0.000, 0.026	0.000, 0.118	1.000, 0.121
0.000, 0.113	0.000, 0.099	0.000, 0.076	0.000, 0.146	0.000, 0.132
0.000, 0.118	0.000, 0.113	0.000, 0.152	0.000, 0.145	0.000, 0.119
0.000, 0.113	0.000, 0.113	0.000, 0.130	0.000, 0.134	0.000, 0.143
0.000, 0.121	0.000, 0.113	0.400, 0.153	0.183, 0.113	0.000, 0.442
0.000, 0.076	0.000, 0.071	0.000, 0.113	0.000, 0.116	0.000, 0.119
0.000, 0.113	0.476, 0.113	0.000, 0.119	0.000, 0.116	0.000, 0.113
0.000, 0.113	0.000, 0.168	0.000, 0.044	0.000, 0.112	7.000, 0.071
0.000, 0.086	0.000, 0.113	0.000, 0.163	0.000, 0.172	0.000, 0.113
0.000, 0.153	0.000, 0.115	0.000, 0.140	0.000, 0.113	0.000, 0.113
0.000, 0.113	0.000, 0.112	0.000, 0.133	0.000, 0.080	0.000, 0.188
0.000, 0.113	0.000, 0.182	0.000, 0.127	0.000, 0.133	0.000, 0.187
0.000, 0.157	0.000, 0.126	0.000, 0.113	0.000, 0.498	0.000, 0.174
0.000, 0.113	0.000, 0.143	0.000, 0.113	0.000, 0.113	0.000, 0.113
0.000, 0.113	0.000, 0.113	0.000, 0.113	0.000, 0.118	0.000, 0.121

0.000, 0.113	0.000, 0.113	0.000, 0.122	0.000, 0.113
0.000, 0.113	0.000, 0.124	0.000, 0.149	0.000, 0.113
0.000, 0.113	0.000, 0.136	0.000, 0.120	0.000, 0.082
0.000, 0.111	0.000, 0.113	0.000, 0.128	0.000, 0.110
0.000, 0.157	2.000, 0.181	1.000, 0.211	2.442, 0.127
0.000, 0.113	0.000, 0.113	0.000, 0.149	0.000, 0.113
0.000, 0.140	0.000, 0.047	0.000, 0.105	0.000, 0.114
0.000, 0.066	1.086, 0.095	0.000, 0.118	0.000, 0.113
0.000, 0.078	0.000, 0.166	0.000, 0.113	0.000, 0.113
0.000, 0.117	0.000, 0.113	3.360, 0.165	
0.000, 0.010	0.000, 0.159	0.000, 0.104	
0.000, 0.113	0.000, 0.188	0.791, 0.113	
0.000, 0.113	0.000, 0.113	0.000, 0.113	
0.000, 0.126	0.000, 0.171	0.000, 0.113	
0.000, 0.113	0.000, 0.113	0.000, 0.120	
0.000, 0.113	0.000, 0.066	0.000, 0.136	
0.000, 0.110	0.000, 0.113	0.000, 0.101	
0.000, 0.129	0.000, 0.102	0.000, 0.113	
0.000, 0.113	0.000, 0.188	0.000, 0.161	
1.000, 0.115	0.000, 0.113	0.000, 0.113	
0.000, 0.113	1.000, 0.102	0.000, 0.114	
0.000, 0.144	0.000, 0.096	0.000, 0.098	
0.000, 0.128	0.000, 0.128	0.000, 0.113	
0.000, 0.163	0.000, 0.113	0.000, 0.111	
0.203, 0.125	0.000, 0.115	0.000, 0.145	
0.000, 0.098	0.000, 0.113	1.132, 0.144	
0.000, 0.113	0.000, 0.113	0.000, 0.110	
0.000, 0.160	0.110, 0.156	0.000, 0.116	
0.000, 0.165	0.000, 0.113	0.000, 0.113	
0.000, 0.110	0.000, 0.121	0.000, 0.113	
0.000, 0.113	0.000, 0.193	0.000, 0.113	
0.000, 0.160	0.000, 0.104	0.000, 0.171	
0.000, 0.148	0.210, -0.031	0.000, 0.114	
0.000, 0.096	0.000, 0.188	0.000, 0.112	
0.000, 0.113	0.000, 0.039	0.000, 0.104	
0.000, 0.184	0.000, 0.113	0.000, 0.167	
0.000, 0.110	0.000, 0.125	0.000, 0.113	
0.000, 0.150	0.000, 0.137	0.000, 0.180	
0.000, 0.146	0.000, 0.110	0.000, 0.154	
0.000, 0.127	0.000, 0.113	0.000, 0.113	
0.000, 0.148	1.000, 0.173	0.000, 0.113	
0.000, 0.113	0.000, 0.129	0.000, 0.188	
0.000, 0.117	0.000, 0.154	0.000, 0.159	
0.000, 0.112	0.000, 0.113	0.000, 0.130	
0.000, 0.123	0.000, 0.152	0.000, 0.138	
0.000, 0.113	0.000, 0.155	0.000, 0.111	

Fold 5	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027
	0.000, 0.027	0.000, 0.027	0.000, 0.008	0.000, 0.027
0.680, 0.024	0.000, 0.027	0.000, 0.029	0.000, 0.029	4.000, -0.001
0.000, 0.027	0.000, 0.027	0.000, 0.022	0.000, 0.023	0.000, 0.017
0.000, 0.020	0.000, 0.037	0.000, 0.027	0.000, 0.020	0.000, 0.018
0.000, 0.023	0.000, 0.027	0.000, 0.026	0.000, 0.015	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.025	0.000, 0.027	0.000, 0.023	2.000, 0.015	0.000, 0.027
0.000, 0.027	0.000, 0.016	0.000, 0.027	0.000, 0.012	0.516, 0.022
0.000, 0.020	0.000, 0.024	0.000, 0.015	0.000, 0.027	0.000, 0.024
0.000, 0.025	0.000, 0.027	0.000, 0.023	1.176, 0.019	0.000, 0.010
0.000, 0.027	0.000, 0.031	0.000, 0.028	0.000, 0.026	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.020	0.000, 0.043	0.000, 0.027
0.000, 0.027	0.000, 0.023	0.000, 0.029	0.000, 0.021	0.000, 0.026
0.000, 0.027	0.000, 0.040	0.000, 0.022	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, -0.010	0.000, 0.027	0.000, 0.025
0.000, 0.026	0.000, 0.020	0.000, 0.027	0.000, 0.020	0.000, 0.025
0.000, 0.022	0.000, 0.020	0.000, 0.027	0.000, 0.016	0.000, 0.011
0.000, 0.039	0.000, 0.026	0.000, 0.027	0.000, 0.082	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.021	0.000, 0.025
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, -0.012	0.000, 0.028
0.000, 0.019	0.000, 0.024	0.000, 0.027	0.000, 0.021	0.000, 0.027
0.000, 0.027	0.000, 0.021	0.000, 0.026	0.000, -0.077	0.000, 0.027
0.000, 0.027	0.000, 0.028	0.000, 0.021	0.000, -0.010	0.000, 0.018
0.000, 0.027	1.625, 0.034	0.000, 0.026	0.000, 0.023	0.000, 0.027
0.000, 0.019	0.000, 0.022	0.000, 0.025	0.000, 0.029	0.000, 0.016
0.000, 0.027	0.000, 0.027	0.000, 0.012	0.000, 0.027	0.000, 0.027
0.000, 0.017	0.000, 0.026	0.000, 0.009	0.000, 0.027	0.000, 0.026
1.000, 0.029	0.000, 0.017	0.000, 0.027	4.000, 0.024	2.457, 0.069
0.000, 0.021	0.000, 0.027	0.000, 0.022	0.000, 0.024	0.000, 0.042
0.000, 0.027	0.000, 0.026	3.000, -0.009	0.000, 0.013	0.000, 0.020
0.000, 0.021	0.000, -0.013	0.000, 0.013	0.000, 0.027	0.000, 0.025
0.000, 0.027	0.000, 0.012	0.000, 0.027	0.000, 0.021	0.000, 0.027
0.000, 0.024	0.000, 0.027	0.000, 0.027	0.000, 0.023	0.000, 0.027
0.000, 0.029	22.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.022	0.000, 0.046	0.000, 0.024	0.000, 0.027	0.000, 0.023
0.000, 0.027	0.000, 0.022	0.000, 0.023	0.000, 0.027	0.000, 0.027
0.400, -0.017	0.000, 0.028	0.000, 0.026	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.029	0.000, 0.027	0.000, 0.016
0.000, 0.026	0.000, 0.027	0.000, 0.019	0.000, 0.027	0.500, 0.026
0.000, 0.029	0.000, 0.027	0.000, 0.037	0.000, 0.027	0.000, 0.029
0.000, 0.027	0.000, 0.029	0.000, 0.016	0.000, 0.027	0.000, 0.020
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027	1.581, 0.909
0.000, 0.019	0.000, 0.024	0.000, 0.044	0.000, 0.025	0.036, 0.033
0.000, 0.004	0.000, 0.027	0.000, 0.009	0.000, 0.018	0.000, 0.028
0.000, 0.027	0.000, 0.028	0.000, 0.027	0.000, 0.027	0.000, 0.024

0.000, 0.029	0.000, 0.027	0.000, 0.027	0.000, 0.025	0.000, 0.012
0.000, 0.024	2.000, 0.027	0.000, 0.027	0.000, 0.025	0.000, 0.027
0.000, 0.027	0.000, 0.004	0.000, 0.020	0.000, 0.027	0.000, 0.024
1.000, -0.013	0.000, 0.027	0.000, 0.027	0.000, 0.024	0.000, 0.024
0.000, 0.027	0.000, 0.023	0.000, 0.027	0.000, 0.027	1.672, 0.425
0.000, 0.033	0.000, 0.018	0.000, 0.023	0.000, -0.002	0.000, 0.015
0.000, 0.027	0.000, 0.027	0.000, 0.021	0.000, 0.027	9.000, 0.026
0.000, 0.021	0.000, 0.054	0.000, 0.027	0.000, 0.024	0.000, 0.047
0.000, 0.018	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.025	0.000, 0.025	0.000, 0.024	0.000, 0.027	0.000, 0.023
0.000, 0.027	0.000, -0.018	0.000, 0.027	0.000, 0.028	0.000, 0.027
0.000, 0.027	0.000, 0.004	0.000, 0.029	0.465, 0.024	0.000, 0.027
0.000, 0.019	0.000, 0.023	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.668	0.056, 0.027	0.000, 0.028	0.000, 0.029	0.000, 0.024
0.000, 0.021	0.000, 0.027	0.000, 0.027	0.000, 0.017	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.029	0.934, 0.025	0.000, 0.024
0.000, 0.027	0.000, 0.029	0.000, 0.027	0.000, 0.026	0.000, 0.022
0.000, 0.051	0.000, 0.031	0.000, 0.027	1.347, 0.014	0.000, 0.024
0.000, 0.027	0.000, 0.025	0.000, 0.031	0.000, 0.027	0.000, -0.070
0.000, 0.006	0.000, 0.031	0.000, 0.024	0.000, 0.026	0.000, 0.020
0.000, 0.027	0.000, -0.040	0.000, -0.018	0.000, 0.027	0.000, 0.030
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.028	0.000, 0.027
0.000, 0.026	0.000, 0.010	0.637, 0.010	0.000, 0.027	0.000, 0.013
0.000, 0.027	0.000, 0.016	0.000, 0.027	0.000, 0.027	0.000, 0.031
0.000, 0.027	0.000, 0.025	0.000, 0.027	0.000, 0.031	0.000, 0.046
0.000, 0.027	0.000, 0.027	0.000, 0.023	0.000, 0.022	0.000, 0.029
0.000, 0.020	0.000, 0.033	0.000, 0.019	0.000, 0.027	2.000, -0.007
0.000, 0.026	0.000, 0.027	0.000, 0.027	0.000, -0.004	0.000, 0.023
0.000, 0.028	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.017
0.000, 0.027	0.000, 0.029	0.000, 0.029	0.000, 0.008	0.000, 0.018
0.000, 0.015	0.000, 0.027	0.000, 0.029	0.000, 0.027	0.000, 0.026
3.000, 0.020	0.000, 0.027	0.000, 0.016	0.000, 0.030	0.000, 0.023
0.000, -2.313	0.027, 0.024	0.000, 0.027	0.000, 0.025	0.000, 0.018
3.850, 0.034	0.000, 0.027	0.000, -0.016	0.000, 0.026	0.000, 0.010
0.000, 0.019	0.000, 0.025	0.000, 0.028	0.000, 0.029	0.000, 0.026
1.000, 0.011	0.000, 0.026	0.000, 0.027	0.000, 0.025	0.000, 0.027
0.000, 0.024	0.000, 0.027	0.000, -0.003	0.000, 0.027	0.000, 0.027
0.000, 0.015	0.000, 0.106	0.000, 0.000	0.000, 0.027	0.000, 0.027
0.000, 0.041	0.000, 0.000	0.000, 0.027	0.000, 0.027	0.000, 0.015
0.000, 0.027	0.000, 0.023	0.000, 0.012	0.000, 0.029	0.500, 0.027
0.000, 0.020	0.000, 0.013	0.000, 0.027	0.000, 0.028	0.000, 0.027
0.000, 0.027	0.000, 0.026	0.000, 0.023	0.000, 0.020	0.000, 0.029
0.000, 0.021	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.015	0.000, 0.027	0.000, 0.021	0.000, 0.026
0.972, 0.021	0.000, 0.018	0.000, 0.027	0.000, 0.027	0.000, 0.014
0.000, 0.006	4.830, 0.021	0.000, 0.021	0.000, 0.271	0.000, 0.027

0.000, 0.027	0.000, 0.024	0.000, 0.020	0.000, 0.024	0.000, 0.027
0.000, 0.027	0.000, -0.012	0.000, 0.027	0.000, -0.171	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.026
0.000, 0.029	0.000, 0.034	0.000, 0.027	0.000, 0.027	0.000, 0.030
0.000, 0.027	0.000, 0.031	0.000, 0.027	0.000, 0.017	0.000, 0.029
0.000, 0.027	0.000, 0.027	0.000, 0.029	0.000, 0.024	0.000, 0.027
0.000, 0.028	1.000, 0.027	0.000, 0.023	0.000, 0.019	0.000, 0.027
0.000, 0.028	0.000, 0.022	0.000, 0.023	0.000, 0.027	0.000, 0.022
0.000, 0.019	0.000, 0.034	0.000, 0.027	0.000, 0.027	7.392, 0.016
0.000, 0.022	0.000, 0.013	0.000, 0.027	0.000, 0.027	0.000, 0.020
0.000, 0.032	0.000, 0.023	0.000, 0.011	0.000, 0.024	0.000, 0.005
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.032
0.000, 0.021	0.000, 0.027	0.000, 0.014	0.000, 0.024	0.000, 0.016
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.028	0.000, 0.027
0.000, 0.023	0.000, 0.027	0.000, 0.008	0.000, 0.027	0.000, 0.027
0.000, 0.025	0.000, 0.030	0.000, 0.026	0.000, 0.024	0.000, 0.027
1.000, 0.021	0.000, 0.028	2.215, 0.022	0.000, 0.027	0.000, 0.027
0.000, 0.026	0.000, 0.027	0.000, 0.024	0.000, 0.002	0.000, 0.026
0.000, 0.026	0.000, 0.010	0.000, 0.025	0.000, 0.019	0.000, 0.025
0.000, 0.027	0.000, 0.027	0.000, 0.024	0.000, 0.027	0.000, 0.031
0.000, 0.011	0.000, 0.023	0.000, 0.027	0.000, 0.022	0.000, 0.027
0.000, 0.027	0.000, 0.013	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.018	0.000, 0.027	0.000, 0.024	0.000, 0.027
0.000, 0.026	0.000, 0.025	0.000, 0.027	0.000, 0.013	0.000, 0.027
0.000, 0.028	0.000, 0.028	0.000, 0.027	0.000, 0.027	0.000, 0.026
0.000, 0.027	0.000, 0.028	0.000, 0.027	0.000, 0.027	0.000, 0.019
0.000, 0.027	0.000, 0.013	0.000, 0.027	0.000, 0.026	0.000, 0.020
0.000, 0.015	0.000, 0.027	0.000, 0.027	0.000, 0.067	0.000, 0.027
2.000, -0.007	0.000, 0.004	0.000, 0.027	0.000, 0.021	0.000, 0.001
0.000, 0.010	0.000, 0.027	0.000, 0.027	104.000,	0.000, 0.025
0.000, 0.032	0.000, 0.027	0.000, 0.029	58.272	0.000, 0.027
3.000, 0.028	0.000, 0.026	0.000, 0.027	0.000, 0.027	5.000, 0.022
0.000, 0.027	0.000, 0.030	0.000, 0.025	0.000, 0.025	0.000, 0.027
0.000, 0.031	0.000, 0.022	0.000, 0.032	0.000, 0.027	0.000, 0.022
0.000, 0.006	0.000, 0.028	0.000, 0.020	0.000, 0.026	0.000, 0.027
0.000, 0.028	0.000, 0.027	0.000, 0.027	0.000, 0.024	0.000, 0.027
0.000, 0.016	0.000, 0.010	0.000, 0.040	0.000, 0.022	0.000, 0.027
0.000, 0.021	0.000, 0.027	0.000, 0.027	0.000, 0.018	0.000, 0.009
0.000, 0.020	1.000, 0.024	0.000, 0.027	0.000, 0.006	0.000, 0.027
0.000, 0.027	0.000, 0.010	0.000, 0.028	0.000, 0.021	0.000, 0.027
0.000, 0.024	0.000, 0.024	0.000, 0.027	0.000, 0.027	0.000, 0.027
0.000, 0.030	0.000, 0.027	0.000, 0.029	0.000, 0.008	0.000, 0.027
0.000, 0.027	6.000, 0.022	0.000, 0.231	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.015	0.304, 0.025	0.000, 0.027	0.000, 0.027
0.130, 0.026	0.000, 0.027	0.000, 0.027	0.000, 0.028	0.000, 0.024
0.000, 0.023	0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, 0.027

0.000, 0.026	0.000, 0.027	0.000, 0.027	0.000, -0.003
0.000, 0.021	0.000, 0.026	0.000, 0.026	0.000, 0.027
0.509, 0.011	0.000, 0.025	0.000, 0.021	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.026	0.000, 0.012
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.858, 0.026
1.000, 0.077	0.000, 0.009	0.000, 0.027	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.062	0.000, 0.027
0.000, 0.031	0.000, 0.040	0.000, -0.010	0.000, 0.027
0.000, 0.027	0.000, 0.027	0.000, 0.027	0.000, -0.013
0.000, 0.015	4.000, 0.028	0.000, 0.022	0.000, 0.023
1.313, 0.027	0.000, 0.026	0.000, 0.019	
0.000, 0.028	0.000, 0.023	0.000, 0.027	
0.000, 0.027	0.000, 0.027	0.000, 0.022	
0.000, 0.027	0.000, 0.038	0.000, 0.029	
0.000, 0.022	0.000, 0.027	0.000, 0.027	
0.000, 0.029	0.000, 0.027	0.000, 0.012	
0.000, 0.027	0.000, 0.027	0.000, 0.027	
0.000, 0.024	0.000, 0.027	0.000, 0.027	
0.000, 0.027	0.000, 0.027	0.000, 0.027	
0.000, 0.020	0.000, 0.024	0.000, 0.029	
0.000, 0.026	22.000, 0.014	0.000, 0.016	
0.000, 0.018	0.000, 0.026	0.000, 0.027	
0.000, 0.021	0.000, -0.021	0.000, 0.027	
0.000, 0.027	0.000, 0.021	0.000, 0.020	
0.000, 0.026	0.000, 0.024	0.000, 0.024	
0.000, 0.028	0.000, 0.026	0.000, 0.027	
0.000, 0.024	0.000, 0.027	0.000, 0.020	
0.000, 0.022	0.000, 0.021	0.000, 0.015	
0.000, 0.017	0.000, 0.027	0.000, 0.027	
0.000, 0.026	0.000, 0.027	0.000, 0.016	
0.000, 0.013	0.000, 0.021	0.000, 0.027	
0.000, 0.029	0.000, 0.027	0.000, 0.027	
0.000, 0.027	0.000, 0.027	0.000, 0.026	
0.000, 0.027	0.000, 0.022	0.000, 0.021	
0.000, 0.027	0.000, 0.027	0.000, 0.027	
0.000, 0.019	0.000, 0.027	0.186, 0.026	
0.000, 0.025	0.000, 0.027	0.000, 0.027	
0.000, 0.022	0.000, 0.029	0.000, 0.025	
0.000, 0.027	0.000, 0.027	0.000, 0.019	
0.000, 0.033	0.365, 0.027	0.000, 0.018	
0.000, 0.055	0.000, 0.010	0.000, 0.026	
0.000, 0.027	0.000, 0.022	0.000, 0.019	
0.000, 0.028	0.000, 0.027	0.000, 0.027	
0.000, 0.027	0.000, 0.027	0.000, 0.027	
0.000, 0.027	0.000, 0.027	0.000, 0.027	
0.000, 0.026	0.000, 0.027	0.000, 0.027	

Fold 6	0.000, 0.126	0.000, 0.138	0.000, 0.146	0.000, 0.125
	0.000, 0.125	0.000, 0.127	0.000, 0.125	0.000, 0.090
0.000, 0.127	0.000, 0.125	0.000, 0.125	0.000, 0.124	0.000, 0.098
0.000, 0.126	3.000, 0.055	0.000, 0.096	0.000, 0.097	5.000, -0.148
1.000, 0.128	5.760, 0.125	0.000, 0.108	0.000, 0.125	0.000, 0.123
0.000, 0.075	0.000, 0.122	0.000, 0.146	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.125	0.000, 0.106	0.000, 0.115	0.000, 0.120
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.114	0.000, 0.124
0.000, 0.125	0.000, 0.122	0.000, -0.012	0.000, 0.077	0.000, 0.130
0.000, 0.125	0.000, 0.123	0.000, 0.160	0.000, 0.122	0.000, 0.125
0.000, 0.125	0.000, 0.110	0.000, 0.121	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.063	0.000, 0.113	0.000, 0.081	0.000, 0.125
0.000, 0.112	0.000, 0.125	0.000, 0.123	0.000, 0.065	0.000, 0.132
0.000, 0.109	0.000, 0.120	0.000, 0.122	0.000, 0.125	0.000, 0.126
0.000, 0.085	0.000, 0.125	0.000, 0.016	0.000, 0.129	0.000, 0.125
0.000, 0.116	0.000, 0.125	0.000, 0.128	0.000, 0.125	0.000, 0.127
0.000, 0.126	0.000, 0.122	0.000, 0.118	0.000, 0.131	0.000, 0.125
0.000, 0.126	0.000, 0.123	0.000, 0.037	0.000, 0.119	0.000, -0.021
0.476, 0.118	0.000, 0.135	2.000, 0.168	0.000, 0.079	0.000, 0.112
0.000, 0.103	0.000, 0.121	0.000, 0.123	0.000, 0.128	0.000, 0.125
0.000, 0.222	0.000, 0.125	0.000, 0.111	0.000, 0.125	0.000, 0.196
0.000, 0.115	0.000, 0.018	0.000, 0.125	0.000, 0.125	2.000, 0.084
0.000, 0.044	0.000, 0.112	0.000, 0.133	0.000, 0.125	0.000, 0.125
0.000, 0.117	0.000, 0.117	0.000, 0.144	0.000, 0.171	0.000, 0.125
0.000, 0.125	0.000, 0.118	0.000, 0.122	0.000, 0.083	0.000, 0.125
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, -0.564	0.000, 0.113
0.000, 0.127	0.000, 0.082	0.000, 0.125	0.489, 0.123	0.000, 0.116
0.000, 0.115	0.000, 0.127	0.000, 0.125	0.000, 0.098	0.000, 0.125
0.000, 0.125	0.000, 0.126	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.120	0.000, 0.128	0.000, 0.125	0.000, 0.125
0.000, 0.122	1.000, 0.109	0.000, 0.131	0.000, 0.118	0.000, 0.109
0.000, 0.125	0.000, 0.137	0.000, 0.123	0.000, 0.125	0.000, 0.047
0.000, 0.098	0.000, 0.088	0.074, 0.067	0.000, 0.127	0.000, 0.119
0.000, 0.123	0.000, 0.135	0.000, 0.100	0.000, 0.131	0.000, 0.123
0.000, 0.125	0.000, 0.118	0.000, 0.095	0.000, 0.125	0.000, 0.167
0.000, 0.125	0.000, 0.038	0.000, 0.125	0.000, 0.055	0.000, 0.113
0.000, 0.125	0.000, 0.112	0.000, 0.125	0.000, 0.123	0.000, 0.125
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125
2.000, 0.109	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.000, 0.127	0.000, 0.147	0.000, 0.125	0.000, 0.100	0.000, 0.125
0.000, 0.121	0.000, 0.125	0.000, 0.126	1.000, 0.119	0.000, 0.125
0.375, 0.123	0.000, 0.106	0.000, 0.112	0.000, 0.081	0.000, 0.114
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, -0.382	1.624, 0.116
0.000, 0.103	0.000, 0.121	0.000, 0.118	0.000, 0.142	0.000, 0.119
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.136
0.000, 0.125	0.000, 0.130	0.000, 0.125	0.000, 0.118	0.000, 0.125

0.000, 0.092	0.000, 0.117	0.000, 0.112	0.000, 0.125	0.000, 0.093
0.000, 0.102	2.010, 0.112	0.000, 0.115	0.000, 0.125	0.000, 0.084
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.104	0.000, 0.125
0.000, 0.128	0.000, 0.133	0.000, 0.112	0.000, 0.125	0.000, 0.122
0.000, 0.125	0.000, 0.109	0.000, 0.125	0.000, 0.161	0.000, 0.125
0.000, 0.125	0.000, 0.094	0.000, 0.221	0.000, -0.638	0.000, 0.125
0.000, 0.123	0.000, 0.130	0.000, 0.148	0.000, 0.114	0.000, 0.125
0.000, 0.097	0.000, 0.079	0.000, 0.116	0.000, 0.126	0.000, 0.125
0.000, 0.125	0.000, 0.089	0.000, 0.130	0.000, 0.122	0.000, 0.108
0.000, 0.109	0.000, 0.128	0.000, 0.124	0.000, 0.102	5.000, 0.319
0.000, 0.104	0.433, 0.050	0.000, 0.125	0.820, 0.077	0.000, 0.060
0.000, 0.129	0.000, 0.123	0.000, 0.117	0.000, 0.192	0.000, 0.125
0.000, 0.129	0.000, 0.123	0.000, 0.106	0.000, 0.123	0.000, 0.123
0.000, 0.119	0.000, 0.121	0.000, -0.049	0.000, 0.078	0.000, 0.123
0.000, 0.125	0.000, 0.125	0.000, 0.068	0.000, 0.112	0.000, 0.107
0.000, 0.145	0.000, 0.110	0.000, 0.115	0.000, 0.119	0.000, 0.125
0.000, 0.118	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.123
0.000, 0.125	0.000, 0.125	0.000, 0.124	0.000, 0.125	0.000, 0.135
0.000, 0.125	0.000, 0.124	0.000, 0.120	0.456, 0.111	0.000, 0.117
0.000, 0.119	0.000, 0.103	0.000, 0.125	1.118, 0.110	0.000, 0.119
0.000, 0.125	0.000, 0.180	0.000, 0.117	0.000, 0.125	0.000, 0.125
0.000, 0.123	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.125	0.000, 0.123	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.102	0.000, 0.046	0.000, 0.159	0.000, 0.134
0.000, 0.122	0.000, 0.134	0.000, 0.065	0.000, 0.125	0.000, -0.049
0.000, 0.104	0.000, 0.108	0.150, 0.113	0.000, 0.125	0.000, 0.125
0.000, 0.113	0.000, 0.125	0.000, 0.136	0.000, 0.125	0.000, 0.125
0.000, 0.132	0.000, 0.125	0.000, 0.118	0.000, 0.125	0.070, 0.122
0.000, 0.125	0.000, 0.122	0.000, 0.125	0.459, 0.042	0.000, 0.125
0.000, 0.125	0.000, 0.107	0.000, 0.117	0.000, 0.109	0.000, 0.121
0.000, 0.133	0.000, 0.100	0.000, 0.125	0.000, 0.093	0.000, 0.125
0.000, 0.111	0.000, 0.128	0.000, 0.123	0.000, 0.122	0.014, 0.122
0.000, 0.084	0.000, 0.127	0.458, 0.126	0.000, 0.110	0.000, 0.125
0.000, 0.129	0.000, 0.125	0.000, 0.042	0.000, 0.125	0.000, 0.118
0.000, 0.114	0.000, 0.119	0.000, 0.073	0.000, 0.107	0.000, 0.120
0.000, 0.070	0.000, 0.125	0.000, 0.083	0.000, 0.125	0.000, 0.125
0.000, 0.107	0.000, 0.125	0.000, 0.142	0.000, 0.125	0.000, 0.125
0.000, 0.130	0.000, 0.125	0.000, 0.123	2.000, 0.126	0.000, 0.110
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.134	0.000, 0.173
0.000, 0.125	0.000, 0.146	0.000, 0.128	0.000, 0.125	0.000, 0.109
0.000, 0.099	0.000, 0.125	0.040, 0.123	0.000, 0.125	0.000, 0.121
0.000, 0.112	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.115	0.000, 0.112	0.000, 0.086	0.000, 0.125
0.000, 0.125	0.000, 0.117	0.000, 0.113	0.000, 0.129	6.990, 0.115
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, -1.047	0.000, 0.125
0.162, 0.102	0.000, 0.133	0.000, 0.123	0.000, 0.119	0.000, 0.125

0.000, 0.104	0.000, 0.124	0.000, 0.125	0.000, 0.122	0.000, 0.105
0.000, 0.125	0.000, 0.123	0.000, 0.125	0.000, 0.125	0.000, 0.079
0.000, 0.125	0.000, 0.121	0.000, 0.114	0.000, 0.311	0.000, 0.029
0.000, 0.125	0.000, 0.383	0.000, 0.096	0.000, 0.125	0.000, 0.125
0.000, 0.178	0.000, 0.000	0.000, 0.096	0.000, 0.109	0.000, 0.112
0.000, 0.128	0.000, 0.114	0.000, 0.125	0.168, 0.112	0.000, 0.125
0.000, 0.118	0.000, 0.118	0.000, 0.119	0.000, 0.125	0.000, 0.128
0.000, 0.123	0.000, 0.128	0.000, 0.133	0.456, 0.119	0.000, 0.089
0.000, 0.107	0.000, 0.067	0.000, 0.011	0.000, 0.108	0.000, 0.169
0.000, 0.173	0.000, 0.096	0.000, -0.020	0.000, 0.153	0.000, 0.121
0.000, 0.125	0.000, 0.100	0.000, 0.151	0.000, 0.067	0.000, 0.127
0.000, 0.130	0.000, 0.120	0.000, 0.121	0.000, 0.123	0.000, 0.114
0.000, 0.122	0.000, 0.108	0.000, 0.125	0.000, 0.126	0.130, 0.097
0.000, 0.180	0.000, 0.130	0.000, 0.125	0.000, 0.125	0.000, 0.123
0.000, 0.083	0.000, 0.125	0.000, 0.127	0.000, 0.098	0.000, 0.085
0.000, 0.122	0.000, 0.122	0.000, 0.131	0.000, 0.034	0.000, 0.125
0.000, 0.125	11.660, 0.112	0.000, 0.115	0.000, 0.117	0.000, 0.125
0.000, -0.012	0.000, 0.123	0.000, 0.121	0.000, 0.125	0.000, 0.125
0.000, 0.120	0.000, 0.119	0.000, 0.108	0.000, 0.124	0.000, 0.127
0.000, -0.008	0.000, 0.173	0.000, 0.093	0.000, 0.080	0.000, 0.109
0.507, 0.118	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.121
0.000, 0.125	0.000, 0.125	0.000, 0.111	0.000, 0.125	0.000, 0.128
0.000, 0.164	0.000, 0.128	0.000, 0.099	0.000, 0.125	0.000, 0.125
0.000, 0.110	0.000, 0.123	0.000, 0.125	0.000, 0.122	0.000, 0.096
0.000, 0.062	0.000, 0.111	0.000, 0.127	0.000, 0.118	0.000, 0.131
0.000, 0.125	0.000, 0.120	0.000, 0.125	0.000, 0.117	0.000, 0.077
0.000, 0.094	0.000, 0.129	0.000, 0.125	0.000, 0.126	0.000, 0.196
0.000, 0.006	0.000, 0.125	0.000, 0.090	0.000, 0.103	0.000, 0.044
0.000, 0.125	0.000, 0.128	0.000, 0.126	0.000, 0.116	0.000, 0.079
0.000, 0.125	0.000, 0.101	0.000, 0.121	0.000, 0.125	0.000, 0.121
0.000, 0.125	0.000, 0.128	0.000, 0.125	0.000, 0.113	0.000, 0.125
0.000, 0.123	0.000, 0.113	0.000, 0.123	0.000, 0.123	0.000, 0.105
0.000, -0.035	0.000, 0.133	0.000, 0.125	0.382, 0.110	0.000, 0.108
1.916, -0.294	0.000, 0.117	0.000, 0.108	0.000, 0.140	0.000, 0.068
0.000, 0.108	0.000, 0.125	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.032, 0.117	0.000, 0.125	0.150, 0.119	0.000, 0.125	0.000, 0.125
0.000, 0.125	0.000, 0.101	0.000, 0.107	0.000, 0.117	0.000, 0.113
0.000, 0.125	0.000, 0.120	0.964, 0.125	0.000, 0.125	0.668, -0.039
0.000, 0.127	0.000, 0.138	0.000, 0.125	0.000, 0.135	0.000, 0.073
0.000, 0.115	0.000, 0.099	0.000, 0.118	0.000, 0.130	0.000, 0.125
0.000, 0.130	0.000, 0.127	0.000, 0.125	0.000, 0.125	0.000, 0.125
0.000, 0.109	0.545, 0.115	0.000, 0.130	0.000, 0.125	0.000, 0.113
0.000, 0.125	0.000, 0.116	0.000, 0.125	0.000, 0.114	0.000, 0.092
0.000, 0.123	0.000, 0.104	0.180, 0.115	0.000, 0.125	0.000, 0.120
0.000, 0.047	0.000, 0.125	0.000, 0.081	0.000, 0.130	0.000, 0.099
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.142, 0.127	0.000, 0.125

0.000, 0.113	0.000, 0.104	0.000, 0.124	0.000, 0.123
0.000, 0.125	0.000, 0.125	0.000, 0.125	0.603, 0.112
0.000, 0.097	0.000, 0.125	0.000, 0.123	0.000, 0.125
0.000, 0.122	0.000, -0.174	0.000, 0.119	0.000, 0.125
0.000, 0.124	0.000, 0.123	0.000, 0.125	0.000, 0.121
0.000, 0.100	0.000, 0.125	0.000, 0.107	0.000, 0.124
0.000, 0.113	1.500, 0.112	0.000, 0.105	0.000, 0.125
0.000, 0.114	0.000, 0.125	0.000, 0.113	0.000, 0.125
1.000, 1.093	0.000, 0.126	0.072, 0.120	0.000, 0.125
0.000, 0.140	0.000, 0.122	0.000, 0.120	
0.000, 0.119	0.000, 0.138	0.000, 0.125	
0.000, 0.125	0.000, 0.125	0.000, 0.120	
0.000, 0.122	0.000, 0.114	0.000, 0.129	
0.000, 0.120	8.000, 0.067	0.000, 0.117	
0.136, 0.099	0.000, 0.125	0.000, 0.119	
0.000, 0.125	0.000, 0.125	0.000, 0.128	
0.000, 0.123	0.000, 0.125	0.000, 0.125	
0.000, 0.143	0.000, 0.123	0.000, 0.128	
0.000, 0.140	0.000, 0.125	0.000, 0.125	
0.000, 0.104	0.000, 0.100	0.000, 0.123	
0.000, 0.104	0.000, 0.112	0.000, 0.100	
0.000, 0.077	0.000, 0.112	0.000, 0.116	
0.000, 0.129	0.000, 0.186	0.000, 0.138	
0.000, 0.105	0.000, 0.125	0.000, 0.125	
0.000, 0.116	0.000, 0.124	0.000, 0.125	
0.048, 0.124	0.000, 0.113	0.000, 0.139	
0.000, 0.112	0.000, 0.087	0.000, 0.097	
0.000, -2.819	0.000, 0.125	0.000, 0.123	
0.000, 0.121	0.000, 0.158	0.000, 0.109	
0.000, 0.125	0.000, 0.109	0.000, 0.086	
0.000, 0.125	0.000, 0.115	0.000, 0.130	
0.000, 0.106	0.000, 0.138	0.000, 0.107	
0.000, 0.124	0.000, 0.236	0.000, 0.125	
0.000, 0.146	0.000, 0.111	0.000, 0.125	
0.000, 0.125	0.000, 0.126	0.000, 0.126	
0.000, 0.115	0.000, 0.081	0.000, 0.107	
0.000, 0.110	0.000, 0.120	0.000, 0.125	
0.000, 0.053	0.000, 0.123	0.000, 0.125	
0.000, 0.141	0.000, 0.129	0.000, 0.140	
0.000, 0.125	0.000, 0.125	0.000, 0.125	
0.000, 0.118	0.000, 0.102	0.000, 0.125	
0.000, 0.126	0.000, 0.123	0.000, 0.139	
0.000, 0.128	0.000, 0.125	0.000, 0.124	
0.000, 0.130	0.000, 0.089	0.000, 0.086	
0.000, 0.123	0.000, 0.123	0.596, 0.123	
0.000, -0.079	0.000, 0.116	0.000, 0.125	

Fold 7	0.000, 0.135	0.000, 0.151	0.000, 0.139	0.000, 0.147
	0.000, 0.151	5.000, 0.153	0.000, 0.145	0.000, 0.136
0.000, 0.151	0.000, 0.151	0.000, 0.141	0.612, 0.134	0.333, 0.164
0.000, 0.147	0.090, 0.129	0.000, 0.148	0.000, 0.115	0.000, -0.185
0.000, 0.134	0.000, 0.135	0.000, -0.024	0.000, 0.151	0.000, 0.150
0.000, 0.135	0.000, 0.151	0.000, -0.157	0.000, 0.151	0.500, 0.134
0.000, 0.151	0.000, 0.076	0.000, 0.129	0.000, 0.121	0.000, 0.117
0.000, 0.151	0.000, 0.151	0.000, 0.095	0.000, 0.151	0.000, 0.151
0.000, 0.157	0.000, 0.151	0.000, 0.150	0.000, 0.135	0.000, 0.146
0.000, 0.137	0.000, 0.130	0.000, 0.120	0.000, 0.121	0.000, 0.131
0.000, 0.151	0.000, 0.151	0.000, 0.163	0.000, 0.151	0.000, 0.136
0.000, 0.151	0.000, 0.037	0.000, 0.151	0.000, 0.122	0.000, 0.151
0.000, 0.151	0.000, 0.106	0.000, 0.087	0.000, 0.151	0.210, 0.129
0.000, 0.151	0.000, 0.142	0.000, 0.151	0.000, 0.119	0.000, 0.134
0.000, 0.151	0.000, 0.136	0.000, 0.132	0.000, 0.151	0.000, 0.151
0.000, 0.130	0.000, 0.114	0.000, 0.151	0.000, 0.151	0.000, 0.129
0.000, 0.153	0.000, 0.131	0.000, 0.141	0.000, 0.136	0.000, 0.151
0.000, 0.045	0.000, -0.025	0.000, 0.151	0.000, 0.153	0.000, 0.212
0.000, 0.151	0.000, 0.146	0.000, 0.151	0.000, 0.105	0.000, 0.151
0.000, 0.151	0.000, 0.156	0.000, 0.130	0.000, 0.152	0.000, 0.151
0.000, 0.151	0.000, 0.118	0.000, 0.141	0.000, 0.150	0.000, 0.082
0.000, 0.151	0.000, 0.151	0.000, -0.054	0.000, 0.146	0.000, 0.115
0.000, 0.151	0.000, 0.149	0.000, 0.142	0.000, 0.151	0.000, 0.151
0.846, 0.134	0.000, 0.139	0.000, 0.009	6.000, 0.018	0.000, 0.150
0.000, 0.151	0.000, 0.153	0.000, 0.139	0.000, 0.122	0.000, 0.126
0.000, 0.103	0.000, 0.131	0.000, 0.134	0.000, 0.146	3.122, 0.137
0.000, 0.151	0.000, 0.110	0.000, 0.154	0.000, 0.151	0.000, 0.151
0.000, 0.152	0.000, 0.101	0.000, 0.051	0.000, 0.151	0.000, 0.137
0.000, -0.021	0.000, 0.102	0.202, 0.151	0.000, 0.151	0.000, 0.146
0.000, 0.148	0.000, 0.151	0.000, 0.151	0.000, 0.051	0.000, 0.151
0.000, 1.174	0.000, 0.117	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.142	0.000, 0.151	0.000, 0.151	0.000, 0.119	0.000, 0.151
0.000, 0.175	0.000, 0.140	0.000, 0.151	0.000, 0.138	0.000, 0.120
3.100, 0.131	0.000, 0.180	0.000, 0.042	0.000, 0.151	0.000, 0.130
0.000, 0.135	0.000, 0.093	0.000, 0.123	0.000, 0.120	0.000, 0.151
0.000, 0.105	0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.145	0.000, 0.134	0.000, 0.151	0.000, 0.151	0.000, 0.147
0.000, 0.153	0.000, 0.102	0.000, 0.151	0.000, 0.081	0.000, 0.151
0.000, 0.118	0.000, 0.148	0.000, 0.151	0.000, 0.151	0.000, 0.138
0.000, 0.135	0.000, 0.124	0.000, 0.151	0.000, 0.151	0.000, -0.133
0.000, 0.141	0.000, 0.138	0.000, 0.119	0.000, 0.151	0.000, 0.143
0.000, 0.152	0.000, 0.113	0.000, 0.151	0.000, 0.151	0.000, 0.062
0.000, 0.123	0.000, 0.154	0.000, 0.109	0.000, 0.125	0.000, 0.151
0.000, 0.139	0.000, 0.108	0.000, 0.141	0.000, 0.151	0.000, 0.134
0.000, 0.106	2.442, 0.148	0.000, 0.066	0.000, 0.149	0.000, 0.151
0.000, 0.137	0.000, 0.151	0.000, 0.380	0.000, 0.126	0.000, 0.151

0.000, 0.151	0.000, 0.151	0.000, 0.131	0.000, 0.143	0.310, 0.069
0.000, 0.155	0.000, 0.151	0.000, 0.141	0.000, 0.139	0.000, 0.151
0.000, 0.151	0.000, 0.141	0.000, 0.151	0.000, 0.159	0.000, 0.135
0.000, 0.151	0.000, 0.099	0.000, 0.151	0.000, 0.151	5.496, 0.095
0.000, 0.151	0.000, 0.083	0.000, 0.151	0.000, 0.151	0.000, 0.135
0.000, 0.151	0.000, 0.151	1.022, 0.139	0.000, 0.121	0.000, 0.005
0.000, 0.148	0.000, 0.136	0.000, 0.123	0.000, 0.134	0.000, 0.151
0.000, 0.134	10.000, 4.207	0.000, 0.151	0.000, 0.135	0.000, 0.156
0.000, 0.118	0.000, 0.156	0.000, 0.151	0.000, 0.157	0.000, 0.151
0.000, 0.151	0.000, 0.151	0.000, 0.099	0.000, 0.154	0.000, 0.141
0.000, 0.139	0.000, 0.122	0.000, 0.151	0.000, 0.126	0.000, 0.151
0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.139	0.000, 0.151
0.000, 0.078	0.000, 0.110	4.000, 0.029	0.000, 0.067	0.000, 0.154
0.000, 0.151	0.000, 0.115	0.000, 0.130	0.000, 0.148	0.640, 0.105
0.000, 0.113	0.350, 0.121	0.000, 0.064	0.000, 0.151	0.000, 0.151
0.000, 0.143	0.000, 0.142	0.000, 0.156	0.000, 0.105	0.000, 0.133
0.000, 0.153	0.000, 0.134	0.000, 0.151	0.000, 0.151	0.000, 0.272
0.000, 0.155	0.000, 0.151	0.000, 0.106	0.000, 0.126	0.000, 0.127
0.000, 0.151	0.000, 0.142	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.080	0.000, 0.165	0.000, 0.151	0.000, 0.181	0.000, 0.142
0.000, 0.141	0.000, 0.151	0.000, 0.151	0.000, 0.126	0.000, 0.135
0.000, 0.094	0.000, 0.138	0.000, 0.156	0.000, 0.151	0.000, 0.056
0.000, 0.151	0.000, 0.108	0.000, 0.150	0.000, 0.127	0.000, 0.055
0.000, 0.100	0.000, 0.151	0.000, 0.151	0.000, 0.121	0.000, 0.147
0.000, 0.149	0.000, 0.065	0.000, 0.121	0.000, 0.125	0.000, 0.151
0.023, 0.110	0.000, 0.149	0.000, 0.151	0.000, -0.027	0.110, 0.132
6.000, 0.059	0.000, 0.151	0.000, 0.134	0.000, 0.052	0.000, 0.016
0.000, 0.151	0.000, 0.148	0.000, 0.134	0.000, 0.151	0.000, 0.149
0.000, 0.092	0.000, 0.134	0.000, 0.153	0.000, 0.018	0.000, 0.135
0.000, 0.151	0.000, 0.151	0.000, 0.135	0.000, 0.152	0.000, 0.151
0.000, 0.127	0.000, 0.087	0.000, 0.147	0.000, 0.151	0.000, 0.134
0.000, 0.602	0.000, 0.145	0.000, 0.153	0.000, 0.135	0.000, 0.133
0.000, 0.156	0.000, 0.151	0.000, 0.112	0.000, 0.156	0.000, 0.128
0.000, 0.123	0.000, 0.151	0.000, 0.132	0.000, 0.110	0.000, 0.151
0.000, 0.148	0.000, 0.151	0.000, 0.133	0.000, 0.151	0.000, 0.140
0.000, 0.150	0.000, 0.152	0.000, 0.135	0.000, 0.139	0.000, 0.142
0.000, 0.151	0.000, 0.136	0.000, 0.124	0.000, 0.128	0.000, 0.139
0.160, 0.143	0.000, 0.128	0.000, 0.165	0.000, 0.141	0.000, 0.151
0.000, 0.004	0.000, 0.102	0.000, 0.151	0.000, -0.020	0.000, 0.119
0.000, 0.151	0.000, 0.151	1.000, 0.134	0.000, 0.151	0.000, 0.125
0.896, 0.151	0.000, 0.134	0.000, 0.170	0.000, 0.152	2.000, 0.150
0.000, 0.147	0.000, 0.149	0.000, 0.151	0.000, 0.116	0.000, 0.151
0.000, 0.152	0.000, 0.145	0.440, 0.148	0.000, 0.090	0.000, 0.151
0.000, 0.109	0.000, 0.132	0.000, 0.151	0.000, -0.069	0.000, 0.095
3.858, -0.587	0.000, 0.111	0.000, 0.153	0.000, 0.151	0.000, 0.139
0.000, 0.151	0.000, 0.186	0.000, 0.151	0.000, 0.141	0.000, 0.151

0.000, 0.153	0.000, 0.130	0.000, 0.151	0.000, -0.047	0.000, 0.137
0.000, 0.151	0.000, 0.104	1.000, 0.134	0.000, 0.152	0.000, 0.123
0.000, 0.134	0.000, 0.152	0.000, 0.136	1.210, 0.148	0.000, 0.153
0.000, 0.151	0.000, 0.121	0.000, 0.148	0.000, 0.135	0.000, 0.102
0.000, 0.090	0.000, 0.155	0.000, 0.152	0.000, 0.151	0.000, 0.128
0.000, 0.136	0.000, 0.115	0.000, 0.134	0.000, 0.151	0.000, 0.151
0.000, 0.153	0.000, 0.148	0.000, 0.134	0.000, 0.153	0.000, 0.151
0.000, 0.151	0.000, 0.138	0.000, 0.152	0.000, 0.151	0.000, 0.115
0.187, 0.136	0.000, 0.165	0.000, 0.161	0.000, 0.136	0.000, 0.169
0.000, 0.143	0.000, 0.091	0.000, 0.135	0.000, 0.171	0.000, 0.156
0.000, 0.138	0.000, 0.137	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.155	0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.135
0.000, 0.151	0.000, 0.151	0.000, 0.121	0.000, 0.141	0.000, 0.151
3.858, -0.587	0.000, 0.151	0.000, 0.149	0.000, 0.151	0.000, 0.119
0.000, 0.131	2.000, 0.129	0.000, 0.080	0.000, 0.155	0.000, 0.154
0.000, 0.137	0.000, 0.113	0.000, 0.000	0.000, 0.151	0.000, 0.150
0.000, 0.151	2.000, 0.120	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.151	0.000, 0.144	0.000, 0.151	0.000, 0.151	0.000, 0.152
0.000, 0.126	0.000, 0.152	0.000, 0.143	0.000, 0.082	0.000, 0.150
0.000, 0.151	0.000, 0.145	0.000, 0.130	0.000, 0.096	0.000, 0.151
0.000, 0.151	0.000, 0.160	0.000, 0.151	0.000, 0.154	0.000, 0.129
0.000, 0.115	0.000, 0.156	0.000, 0.109	0.000, 0.151	0.000, 0.147
0.000, 0.142	0.000, 0.138	0.000, 0.141	0.000, 0.122	0.000, 0.151
0.000, 0.066	0.000, 0.151	0.000, 0.145	0.000, 0.151	0.000, 0.100
0.000, 0.151	0.000, 0.038	0.000, 0.135	0.000, 0.151	0.000, -0.020
0.000, 0.151	0.000, 0.148	0.000, 0.096	0.000, 0.120	0.000, 0.151
0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.138	0.000, 0.130
0.000, 0.161	0.000, 0.148	0.000, 0.151	0.000, 0.151	0.000, 0.121
0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.151	0.000, 0.151
0.000, 0.151	0.000, 0.149	0.000, 0.111	0.000, -0.725	0.000, 0.151
0.000, 0.121	0.000, 0.134	1.000, 0.120	0.000, 0.152	0.000, 0.151
0.000, 0.151	0.000, 0.151	0.000, 0.125	0.000, 0.151	0.000, 0.134
0.000, 0.151	0.000, 0.151	0.000, 0.115	0.000, 0.151	0.000, 0.151
0.000, 0.151	0.000, 0.151	0.000, 0.105	0.000, 0.130	0.000, 0.082
0.000, 0.130	0.000, 0.151	0.000, 0.130	0.000, 0.135	0.000, 0.150
0.000, 0.145	0.000, 0.151	0.000, 0.136	0.000, 0.120	0.000, 0.151
0.000, 0.151	0.000, 0.129	0.000, 0.151	0.000, 0.157	0.000, 0.162
0.000, 0.151	0.000, 0.151	0.630, 0.035	0.000, 0.127	0.000, 0.130
0.000, 0.151	0.000, 0.140	0.000, 0.121	0.000, 0.134	0.000, 0.095
0.000, 0.151	0.000, 0.159	0.000, 0.151	0.000, 0.141	0.187, 0.136
0.000, 0.143	0.000, 0.130	0.000, 0.140	0.000, 0.151	0.000, 0.156
0.000, 0.134	0.000, 0.150	0.000, 2.829	0.000, 0.001	0.000, 0.134
0.000, 0.151	0.000, 0.151	0.000, 0.155	0.000, 0.108	0.000, 0.134
0.000, 0.151	0.000, 0.151	0.000, 0.152	0.000, 0.147	0.000, 0.149
0.000, 0.145	0.000, 0.160	0.000, 0.148	0.000, 0.176	0.000, 0.134
0.000, 0.146	0.000, 0.151	0.000, 0.147	0.000, 0.151	0.000, 0.062

0.000, 0.140	0.000, 0.152	0.000, 0.137	0.000, 0.135
0.000, 0.137	0.000, 0.122	0.000, 0.151	0.000, 0.145
0.000, 0.151	0.000, 0.152	0.000, 0.151	0.000, 0.144
0.000, 0.151	0.000, 0.136	0.000, 0.141	0.000, 0.155
0.000, 0.151	0.000, 0.151	0.000, 0.144	0.000, 0.151
0.000, 0.141	0.000, 0.151	0.000, 0.090	0.000, 0.151
0.000, 0.076	0.000, 0.139	0.000, 0.151	0.000, 0.142
0.000, 0.149	0.000, 0.127	0.280, 0.125	0.000, 0.145
0.000, 0.151	0.000, 0.151	0.000, 0.127	1.500, 0.117
0.000, 0.144	0.000, 0.151	0.000, 0.118	
0.000, 0.151	0.000, 0.152	0.000, 0.127	
0.316, 0.152	0.000, 0.150	0.000, 0.152	
0.000, 0.163	0.524, 0.135	0.000, 0.140	
0.000, 0.134	0.000, 0.151	0.000, 0.149	
0.000, 0.151	0.000, 0.142	0.000, 0.129	
0.000, 0.151	0.000, 0.151	0.000, 0.034	
0.000, 0.182	0.000, 0.149	0.000, 0.080	
0.000, 0.152	0.000, 0.140	0.000, 0.079	
0.000, 0.126	0.270, 0.132	0.000, 0.032	
0.000, 0.151	0.000, -0.350	0.000, 0.151	
0.000, -0.047	0.000, 0.133	0.000, 0.131	
0.000, 0.151	0.000, 0.115	0.973, 0.145	
0.000, 0.146	0.000, 0.122	0.000, 0.137	
0.000, 0.130	0.000, -0.062	0.000, 0.151	
0.000, 0.043	0.000, 0.151	0.000, 0.148	
1.000, 0.598	0.000, 0.151	0.000, 0.140	
0.000, 0.153	0.000, 0.149	0.000, 0.151	
0.000, 0.134	0.000, 0.151	0.000, 0.151	
0.000, 0.108	0.000, 0.151	0.000, 0.137	
0.000, 0.099	0.000, 0.151	0.000, 0.132	
0.000, 0.138	0.000, 0.070	0.000, 0.137	
0.000, 0.151	0.000, 0.151	0.000, 0.151	
0.000, 0.151	0.000, 0.076	0.000, 0.181	
0.706, 0.156	2.000, 0.144	0.000, 0.151	
0.000, 0.149	0.000, 0.040	0.000, 0.128	
0.000, 0.151	0.000, 0.143	0.000, 0.182	
0.000, 0.127	0.000, 0.145	0.000, 0.143	
0.000, 0.151	0.000, 0.151	0.000, 0.151	
0.000, 0.151	0.000, 0.134	0.000, 0.159	
6.000, -1.368	0.000, 0.157	0.000, 0.122	
0.000, 0.040	0.000, 0.151	0.000, 0.114	
0.000, 0.151	0.000, 0.151	0.000, 0.150	
0.000, 0.151	0.000, 0.143	0.000, 0.145	
0.160, 0.131	0.000, 0.135	0.000, 0.136	
0.000, 0.151	0.000, 0.151	0.084, 0.152	
0.000, 0.163	0.000, 0.126	0.000, 0.091	

Fold 8	0.000, 0.150	0.000, 0.125	0.000, 0.107	0.000, 0.167
	0.000, 0.110	0.000, 0.107	0.000, 0.164	0.000, 0.107
0.000, 0.107	0.000, 0.143	0.000, 0.097	0.000, 0.107	0.000, 0.167
0.000, -0.008	0.000, 0.140	0.000, 0.184	0.000, 0.162	0.000, 0.111
0.000, 0.132	0.000, 0.126	0.000, 0.107	0.000, 0.143	0.000, 0.117
0.000, 0.164	0.000, 0.107	0.000, 0.123	0.000, 0.150	0.000, 0.109
0.000, 2.209	0.000, 0.107	0.000, 0.117	0.000, 0.106	0.000, 0.123
0.000, 0.150	0.000, 0.162	0.000, 0.115	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.097	0.000, 0.064	4.000, 0.135	0.000, 0.117
0.000, 0.114	0.000, 0.107	0.000, 0.057	6.027, 0.122	0.000, 0.144
0.000, 0.111	0.000, 0.107	5.500, 0.165	0.000, 0.120	0.000, 0.107
0.000, 0.156	0.000, 0.109	0.000, 0.125	0.000, 0.160	0.000, 0.096
0.000, 0.022	1.000, 0.111	0.000, 0.106	0.000, 0.062	0.000, 0.137
0.000, 0.107	0.000, 0.130	0.000, 0.180	0.000, 0.017	0.000, 0.126
0.157, 0.152	0.000, 0.107	0.000, 0.162	0.000, 0.107	0.000, 0.155
0.000, 0.184	0.000, 0.151	0.000, 0.138	0.000, 0.143	0.000, 0.142
0.000, 0.112	0.000, 0.184	0.000, 0.107	0.000, 0.151	1.000, 0.111
0.000, 0.101	0.000, 0.122	0.000, 0.150	0.000, -0.068	0.000, 0.151
1.000, 0.117	0.000, 0.104	0.000, 0.107	0.000, 0.110	0.000, 0.115
0.000, 0.107	0.000, 0.107	0.000, 0.110	0.000, 0.112	0.000, 0.137
0.000, 0.107	0.000, 0.107	0.000, 0.184	0.000, 0.103	0.000, 0.159
0.000, 0.115	0.000, 0.135	0.000, 0.112	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.150	0.000, 0.128	0.000, 0.107
0.000, 0.136	0.000, 0.080	0.000, 0.122	0.000, 0.077	0.000, 0.106
0.000, 0.148	0.000, 0.111	0.000, 0.103	0.000, 0.107	0.000, 0.143
0.000, 0.140	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.113
0.000, 0.114	0.000, 0.134	0.000, 0.099	0.000, 0.107	0.000, 0.107
0.000, 0.143	0.000, 0.123	0.000, 0.112	0.000, 0.106	0.000, 0.139
0.000, -0.018	0.000, 0.122	0.000, 0.111	0.000, 0.136	0.000, 0.107
1.000, 0.184	0.000, 0.107	0.000, 0.106	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.184	0.000, 0.107	0.000, 0.107	0.000, 0.113
0.000, -0.050	0.000, 0.129	0.000, 0.137	0.000, 0.184	14.000, 0.155
0.000, 0.154	0.000, 0.118	0.000, 0.107	0.000, 0.128	0.000, 0.107
0.000, 0.107	0.000, 0.184	0.000, 0.184	0.000, 0.092	0.000, 0.107
0.000, 0.107	0.000, 0.145	23.000, 0.124	0.000, 0.115	0.000, 0.109
0.000, 0.423	0.000, 0.107	0.000, 0.137	0.000, 0.106	0.000, 0.107
0.000, 0.174	0.000, 0.087	0.000, 0.127	0.000, 0.108	0.000, 0.118
0.000, 0.184	0.000, 0.107	0.000, 0.107	0.000, 0.149	0.000, 0.137
0.000, 0.140	0.000, 0.145	0.000, 0.113	0.000, 0.161	0.000, 0.134
0.000, 0.107	0.000, 0.073	0.000, 0.143	0.000, 0.057	0.000, 0.177
0.000, 0.107	0.000, 0.177	0.000, 0.107	0.000, 0.135	0.000, 0.159
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.145	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.151	0.000, 0.140	0.076, 0.105	0.000, 0.137
0.000, 0.134	0.000, 0.107	0.000, 0.107	0.000, 0.124	0.000, 0.174
0.330, 0.120	0.000, 0.130	0.000, 0.138	0.000, 0.107	0.000, 0.107

0.000, 0.114	0.000, 0.034	0.364, 0.142	0.000, 0.107	0.000, 0.107
0.000, 0.115	0.000, 0.381	0.000, 0.134	0.000, 0.107	0.000, 0.107
0.000, -0.126	0.000, 0.106	0.000, 0.146	0.000, 0.184	0.000, 0.107
0.000, 0.107	0.000, 0.098	0.000, 0.184	0.000, 0.106	0.000, 0.107
0.000, 0.111	0.000, 0.184	0.000, 0.108	0.000, 0.107	0.000, 0.089
0.000, 0.135	2.000, 0.123	1.000, 0.140	0.000, 0.102	0.000, 0.115
0.000, 0.107	0.000, 0.150	0.000, 0.107	0.000, 0.121	0.000, 0.081
0.000, 0.116	0.000, -1.165	0.000, 0.141	0.000, 0.139	0.000, 0.135
0.000, 0.184	0.000, 0.107	0.000, 0.036	0.000, 0.107	0.270, 0.095
0.000, 0.109	0.000, 0.107	0.000, 0.159	0.000, 0.107	0.000, 0.107
0.000, 0.105	0.000, 0.107	0.000, 0.183	0.000, 0.128	0.000, 0.123
0.000, 0.075	0.116, 0.116	0.000, 0.118	0.000, 0.140	0.000, 0.109
0.000, 0.126	0.000, 0.169	0.000, 0.111	0.000, 0.107	0.000, 0.138
2.000, 0.107	0.000, 0.107	0.000, 0.107	1.000, 0.124	0.000, 0.071
0.000, 0.159	1.009, 0.088	0.000, 0.148	0.000, 0.107	0.000, 0.184
0.000, 0.099	0.000, 0.107	0.000, 0.155	0.000, 0.150	3.000, 0.122
0.000, 0.107	0.000, 0.169	0.000, 0.107	0.000, 0.140	0.000, 0.115
0.000, 0.107	0.000, 0.107	0.000, 0.135	0.000, 0.165	0.000, 0.107
0.000, 0.120	0.000, 0.137	0.000, 0.169	0.000, 0.107	0.000, 0.107
0.000, 0.110	0.000, 0.131	0.000, 0.127	1.000, 0.106	0.000, 0.093
0.000, 0.107	0.000, 0.107	0.000, 0.118	0.000, 0.119	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.103	0.000, 0.135
0.000, 0.130	0.000, 0.143	0.000, 0.107	0.000, 0.107	0.000, 0.177
0.000, 0.115	0.000, 0.107	0.000, 0.095	0.000, 0.107	0.000, 0.114
0.000, 0.107	0.000, 0.178	0.000, 0.105	0.000, 0.108	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.116	0.000, 0.137
0.000, 0.107	0.000, 0.048	0.000, 0.143	0.000, 0.107	0.000, 0.112
0.000, 0.114	0.000, 0.093	0.000, 0.107	0.000, 0.171	0.000, 0.112
0.000, 0.146	0.000, 0.166	0.000, 0.105	0.000, 0.107	0.000, 0.115
0.000, 0.120	4.000, 0.016	0.000, 0.124	0.000, 0.128	0.000, 0.119
0.000, -0.024	0.000, 0.131	0.000, 0.114	1.000, 0.142	0.000, 0.099
0.000, 0.184	0.000, 0.107	0.000, 0.108	0.000, 0.134	0.000, 0.107
0.000, 0.450	0.000, 0.107	0.000, 0.164	0.000, 0.107	0.000, 0.135
0.000, 0.107	0.000, 0.106	0.000, 0.113	0.000, 0.120	0.000, -0.387
0.000, 0.107	0.000, 0.124	0.000, 0.134	0.000, 0.114	0.000, 0.107
0.000, 0.184	0.000, 0.153	0.000, 0.107	0.000, 0.083	0.000, 0.127
0.000, 0.157	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.165
0.000, 0.101	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107
1.000, 0.113	0.000, 0.120	0.000, 0.112	0.000, 0.107	0.000, 0.069
0.000, -0.018	0.000, 0.121	0.000, 0.107	0.000, 0.118	0.000, 0.116
0.000, 0.130	0.000, 0.087	0.000, 0.107	0.000, 0.121	0.000, 0.156
0.000, 0.111	0.000, 0.107	2.000, 0.129	0.000, -0.203	0.000, 0.107
0.000, 0.107	0.000, 0.127	0.000, 0.114	0.000, 0.130	1.000, 0.138
0.000, 0.140	0.000, 0.026	0.000, 0.137	0.000, 0.142	0.000, 0.148
16.080, 0.093	0.000, 0.107	0.000, 0.107	0.000, 0.116	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.181	0.000, 0.106	0.000, 0.184

0.000, 0.107	0.000, 0.091	0.000, 0.093	4.000, 0.004	0.000, 0.144
0.000, 0.163	0.000, 0.107	0.000, 0.108	0.000, 0.131	0.000, 0.147
0.000, 0.071	0.000, 0.121	0.000, 0.107	0.991, 0.081	0.000, 0.103
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.034
0.000, 0.061	0.000, 0.107	0.000, 0.107	0.000, 0.105	0.996, 0.141
0.000, 0.107	0.000, 0.115	0.000, 0.184	0.000, 0.112	0.000, 0.107
0.000, 0.137	0.000, 0.104	0.000, 0.137	0.000, 0.096	0.000, 0.107
0.000, 0.084	0.000, 0.134	0.000, 0.120	0.000, 0.112	0.000, 0.110
0.000, 0.107	1.324, 0.139	0.000, 0.137	0.000, 0.107	0.000, 0.115
0.000, 0.107	0.000, 0.125	0.000, 0.107	0.000, 0.108	0.000, 0.098
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.118	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.896, 0.121
0.000, 0.127	8.047, 10.179	0.000, 0.125	0.000, 0.114	0.000, 0.107
0.000, 0.115	0.000, 0.112	0.000, 0.107	0.000, 0.073	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.102	0.000, 0.106	0.000, 0.107
0.000, 0.136	0.000, 0.116	0.000, 0.107	0.000, 0.107	0.000, 0.184
0.000, 0.097	0.094, 0.113	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.114	0.000, 0.107	0.000, 0.107	0.000, 0.144	0.000, -0.134
0.000, 0.114	0.000, 0.137	0.000, 0.164	0.000, 0.111	0.000, 0.107
0.000, 0.111	0.000, 0.144	0.000, 0.107	0.000, 0.169	0.000, 0.138
0.000, 0.139	0.000, 0.118	0.000, 0.107	0.000, 0.107	0.000, 0.117
0.000, 0.107	0.000, 0.058	0.000, 0.107	0.000, 0.108	0.000, 0.088
0.000, 0.087	0.000, 0.114	0.000, 0.107	0.000, 0.140	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.153	0.000, 0.169	0.000, 0.107
0.000, 0.133	0.000, 0.107	0.000, 0.140	0.000, 0.145	0.000, 0.107
0.000, 0.107	0.000, 0.096	0.000, 0.107	0.000, 0.108	0.000, 0.110
0.000, 0.107	0.000, 0.115	0.000, 0.120	0.000, 0.125	0.000, 0.107
0.000, 0.139	0.000, 0.123	0.000, 0.184	0.000, 0.107	0.000, 0.130
0.000, 0.102	0.000, 0.107	0.000, 0.106	0.000, 0.082	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.106
0.000, 0.107	0.000, 0.111	0.000, 0.162	0.000, 0.075	0.000, 0.107
0.000, 0.107	0.000, 0.107	0.000, 0.128	0.000, 0.107	0.000, 0.107
0.000, 0.109	0.000, 0.107	0.000, 0.139	0.000, 0.097	0.000, 0.136
0.000, 0.107	0.000, 0.110	0.000, 0.107	6.000, 0.137	0.000, 0.111
0.000, 0.158	0.000, 0.107	0.000, 0.115	0.000, 0.157	0.000, 0.109
0.000, 0.117	0.000, 0.137	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.107	0.000, 0.119	0.000, 0.020	0.000, 0.142	0.500, 0.184
0.000, 0.107	0.000, 0.123	0.000, 0.108	0.000, 0.127	0.000, 0.107
0.000, 0.149	0.000, 0.148	5.005, 0.145	0.000, 0.095	0.000, 0.106
0.000, 0.134	0.000, 0.108	0.000, 0.158	0.000, 0.184	0.000, 0.135
0.000, 0.107	0.000, 0.107	0.000, 0.086	0.000, 0.108	0.000, 0.140
0.000, 0.107	0.000, 0.107	0.000, 0.105	0.000, 0.184	0.000, 0.107
0.000, 0.107	0.000, 0.097	0.000, 0.064	0.000, 0.158	0.000, 0.141
0.000, 0.107	0.000, 0.184	0.000, 0.148	0.000, 0.107	0.000, 0.091
0.000, 0.049	0.000, 0.107	0.000, 0.143	0.000, 0.107	0.000, 0.057
1.000, 0.112	0.000, 0.143	0.000, 0.209	0.000, 0.107	0.000, 0.113

0.000, 0.107	0.000, 0.149	0.000, 0.176	0.000, -0.151
0.000, 0.107	0.000, 0.107	0.000, 0.142	0.000, 0.107
0.000, 0.111	0.000, 0.107	0.000, 0.326	0.000, 0.109
0.000, 0.184	0.000, 0.107	0.000, 0.128	0.000, 0.117
0.000, 0.107	0.000, 0.107	0.000, 0.110	0.000, 0.104
0.000, 0.107	0.000, 0.107	0.000, 0.107	0.000, 0.107
0.000, 0.164	0.000, 0.168	0.000, 0.184	0.000, 0.135
0.000, 0.148	0.000, 0.107	0.000, 0.140	0.000, 0.100
0.000, 0.132	0.000, 0.107	0.000, 0.109	0.000, 0.135
0.000, 0.108	0.000, 0.121	0.000, 0.107	
0.000, 0.107	0.000, 0.118	0.000, 0.107	
0.000, 0.123	0.000, 0.107	0.000, 0.103	
0.000, 0.123	0.000, 0.175	0.000, 0.184	
0.000, 0.184	0.000, 0.110	0.000, 0.151	
0.000, 0.184	0.000, 0.109	0.000, 0.107	
0.000, 0.107	0.000, 0.136	0.000, 0.107	
0.000, 0.107	0.000, 0.107	0.000, 0.107	
0.000, 0.107	0.000, 0.143	0.000, 0.112	
0.000, 0.145	0.000, 0.128	0.000, 0.143	
0.000, 0.107	0.000, 0.128	0.000, 0.134	
0.000, 0.050	0.000, 0.107	0.000, 0.104	
0.243, 0.106	0.000, 0.044	0.000, 0.107	
0.000, 0.107	0.000, 0.091	0.583, 0.134	
0.000, 0.107	0.000, 0.089	0.000, 0.110	
0.000, 0.110	0.000, 0.107	0.000, 0.107	
0.000, 0.115	0.000, 0.107	0.000, 0.133	
0.000, 0.155	0.000, 0.122	0.000, 0.063	
0.000, 0.107	0.000, 0.113	0.000, 0.105	
0.000, 0.147	0.000, 0.135	0.000, 0.066	
0.000, 0.128	0.000, 0.114	0.000, 0.141	
10.000, 0.107	0.000, 0.107	0.000, 0.145	
0.000, 0.148	0.000, 0.184	0.000, 0.095	
0.000, 0.107	0.000, 0.107	0.000, 0.127	
0.000, 0.131	0.000, 0.094	0.000, 0.159	
0.000, 0.150	0.000, 0.107	6.017, 0.152	
0.000, 0.119	0.000, 0.100	0.000, 0.107	
0.000, 0.135	0.000, 0.135	0.000, 0.152	
0.000, 0.107	0.000, 0.133	0.000, 0.116	
0.000, 0.111	0.000, 0.155	0.000, 0.107	
0.000, 0.127	0.000, 0.107	0.000, 0.101	
0.000, 0.142	0.000, 0.107	0.000, 0.163	
0.000, 0.109	0.000, 0.107	0.000, 0.155	
0.000, 0.106	0.000, 0.099	0.000, 0.126	
0.000, 0.107	0.000, 0.115	0.000, 0.160	
0.000, 0.161	0.760, 0.140	0.000, 0.086	
0.000, 0.112	0.000, 0.099	0.000, 0.107	

Fold 9	0.000, 0.123	0.000, 0.125	0.000, 0.122	0.000, 0.040
	0.000, -0.017	0.000, 0.134	0.000, 0.134	0.000, 0.122
0.000, 0.119	0.000, 0.143	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.107	0.000, 0.122	0.000, 0.058	0.000, 0.128	0.000, 0.073
0.000, 0.091	0.000, 0.127	0.000, 0.128	0.000, 0.122	0.000, 0.122
0.000, 0.113	0.000, 0.122	0.000, 0.122	0.000, 0.128	0.000, 0.122
0.000, 0.135	0.000, 0.122	18.000, 0.119	0.000, 0.174	0.000, 0.131
0.000, 0.126	0.000, 0.143	0.000, 0.122	0.295, 0.124	0.000, 0.113
0.000, 0.098	0.000, 0.105	0.000, 0.122	0.000, 0.129	0.000, 0.127
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.063	0.000, 0.127
0.000, 0.122	0.000, 0.121	0.000, 0.122	0.000, 0.122	0.000, 0.107
0.000, 0.122	0.000, 0.134	0.000, 0.122	0.000, 0.055	0.000, 0.124
0.000, 0.134	0.000, 0.128	0.000, 1.045	10.590, 0.119	0.000, 0.108
0.000, 0.117	0.000, 0.133	0.000, 0.132	0.000, 0.127	0.000, 0.122
0.000, 0.122	0.000, 0.095	0.000, 0.115	0.000, 0.106	0.000, 0.122
0.000, 0.126	0.000, 0.122	0.000, 0.122	0.000, 0.124	0.000, 0.127
0.000, 0.135	1.000, 0.221	0.000, 0.122	0.000, 0.062	0.000, 0.123
0.000, 0.122	0.000, 0.109	0.000, 0.127	0.000, 0.127	0.000, 0.095
0.000, 0.122	0.000, 0.138	0.000, 0.130	0.000, 0.099	0.000, 0.122
0.000, 0.128	0.000, 0.134	0.040, 0.143	0.000, 0.124	0.000, 0.129
0.000, 0.106	0.000, 0.136	0.000, 0.122	0.000, 0.122	0.000, 0.121
0.000, 0.123	0.000, 0.081	0.000, 0.122	0.000, 0.117	0.000, 0.122
0.000, 0.122	0.000, -0.350	0.000, 0.122	0.000, 0.122	0.000, 0.141
0.000, 0.122	0.000, 0.122	0.000, 0.085	0.000, 0.123	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.113	0.000, 0.108	0.000, 0.122
0.000, 0.151	0.000, 0.128	1.586, 0.211	0.000, 0.105	0.000, 0.122
0.000, 0.117	0.000, 0.122	0.000, 0.122	0.000, 0.128	0.000, 0.122
0.000, 0.136	0.000, 0.135	0.000, 0.086	0.000, 0.114	0.000, 0.122
0.000, 0.115	0.000, 0.126	0.000, 0.133	0.000, 0.082	0.000, 0.058
0.000, 0.102	0.000, 0.072	0.000, 0.111	0.000, 0.124	0.000, 0.135
0.000, 0.124	0.000, 0.127	0.000, 0.015	0.000, 0.128	0.000, 0.136
0.000, 0.101	0.000, 0.121	0.000, 0.138	0.000, 0.143	0.000, 0.114
0.000, 0.139	0.000, 0.122	0.000, 0.078	0.000, 0.122	0.000, 0.137
0.000, 0.143	0.000, 0.120	0.000, 0.132	0.000, 0.139	0.000, 0.121
1.000, 0.161	0.000, 0.130	0.000, 0.081	0.000, 0.124	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.091
0.000, 0.122	0.000, 0.005	0.000, 0.116	0.000, 0.144	0.000, 2.282
0.000, 0.139	0.000, 0.122	0.000, 0.096	0.000, 0.042	0.000, 0.126
0.000, 0.131	4.000, 0.049	0.000, 0.138	0.000, 0.122	0.000, 0.116
0.000, 0.117	0.000, 0.109	0.000, -0.021	0.000, 0.096	0.000, 0.120
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.111	0.000, 0.103
0.000, 0.122	0.000, 0.081	0.000, 0.110	0.000, 0.118	0.000, 0.143
0.000, 0.080	0.000, 0.382	0.000, 0.127	0.000, 0.122	0.000, 0.123
0.000, 0.143	0.000, 0.122	0.000, 0.105	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.147

0.100, 0.136	0.000, 0.145	0.000, 0.114	0.000, 0.133	0.000, 0.143
0.000, 0.122	0.000, 0.122	0.000, 0.109	0.000, 0.100	0.000, 0.354
0.000, 0.076	0.000, 0.091	0.000, 0.122	0.000, 0.132	0.000, 0.135
0.000, 0.097	0.000, 0.122	0.000, 0.122	0.000, 0.131	0.000, 0.062
0.000, 0.136	0.360, 0.117	0.000, 0.122	0.000, 0.121	0.000, 0.122
0.000, 0.122	0.000, 0.137	0.000, 0.071	0.000, 0.143	0.000, 0.133
0.000, 0.122	0.000, 0.109	0.000, 0.116	0.000, 0.146	0.000, 0.122
0.000, 0.122	0.000, 0.153	0.000, 0.122	0.000, 0.122	0.000, 0.094
0.000, 0.122	0.000, 0.064	0.000, 0.122	0.000, 0.082	0.000, 0.132
0.000, 0.139	0.000, 0.129	0.000, 0.122	0.000, 0.128	0.567, 0.103
0.000, 0.062	0.000, 0.122	0.000, 0.145	0.000, 0.131	0.000, 0.106
0.000, 0.122	0.000, 0.107	0.000, 0.117	0.000, 0.143	0.000, 0.122
0.000, 0.119	0.000, 0.122	0.000, 0.137	0.000, 0.130	0.014, 0.143
3.000, 0.125	0.000, 0.095	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.119	0.000, 0.122	0.000, 0.122	0.000, 0.136	0.000, 0.125
0.000, 0.120	0.000, 0.124	0.000, 0.142	0.000, 0.119	0.000, 0.119
0.523, 0.121	0.000, 0.141	0.000, 0.122	0.000, 0.120	2.048, 0.120
0.000, 0.127	0.000, 0.149	0.000, 0.146	0.000, 0.044	0.000, 0.119
0.000, 0.122	0.000, 0.143	0.000, 0.118	0.000, 0.122	0.000, 0.132
0.000, 0.122	0.000, 0.122	0.000, 0.143	0.000, 0.122	0.000, 0.122
0.000, 0.130	0.000, 0.122	0.000, 0.114	0.000, 0.121	0.000, 0.122
0.000, 0.112	0.000, 0.124	0.000, 0.122	0.000, 0.012	0.000, 0.122
0.000, 0.122	0.000, 0.179	0.000, 0.126	0.000, 0.085	0.000, 0.125
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.103
0.000, 0.126	0.000, 0.111	0.000, 0.019	0.000, 0.105	0.000, 0.122
1.000, 0.147	0.000, 0.122	0.000, 0.139	0.000, 0.138	0.000, 0.102
0.000, 0.107	0.000, 0.122	0.000, 0.128	0.000, 0.121	0.000, 0.094
0.000, 0.122	0.000, 0.099	0.000, 0.111	0.000, 0.146	0.000, 0.122
0.303, 0.123	0.000, 0.122	0.000, 0.133	0.000, 0.039	4.000, 0.130
1.000, 0.134	0.000, 0.001	0.000, 0.117	0.000, 0.120	0.000, 0.122
0.000, 0.009	0.000, 0.122	0.000, 0.132	0.000, 0.131	0.000, 0.139
0.000, 0.129	0.000, -0.151	0.000, 0.134	0.000, 0.122	0.000, 0.122
0.000, 0.111	2.455, -0.089	0.000, 0.134	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.132	0.000, 0.122	0.000, 0.122	0.000, 0.131
7.925, 0.142	4.000, 0.122	2.844, 0.122	0.000, 0.139	0.000, 0.128
0.000, 0.115	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.112
0.000, 0.143	2.000, 0.104	0.000, 0.122	0.000, 0.121	0.000, 0.122
0.000, 0.041	0.000, 0.134	0.000, 0.122	0.000, 0.130	0.000, 0.135
0.000, 0.120	0.000, 0.094	0.000, 0.122	0.000, 0.137	0.090, 0.135
0.000, 0.125	0.000, 0.122	0.000, 0.122	0.000, 0.136	0.000, 0.122
0.000, 0.122	0.000, 0.031	0.000, 0.090	0.000, 0.122	5.150, 0.137
0.000, 0.122	0.000, 0.137	0.000, 0.122	0.000, 0.122	0.000, 0.114
0.000, 0.103	0.000, 0.122	0.000, 0.105	0.000, 0.087	0.000, 0.127
0.000, 0.127	0.000, 0.122	0.000, 0.108	1.000, 0.117	0.000, 0.126
0.000, 0.091	0.000, 0.093	0.000, 0.122	0.000, 0.122	0.000, 0.144
0.000, 0.111	0.000, 0.123	0.000, 0.144	0.000, 0.122	0.000, 0.122

0.000, 0.069	0.000, 0.122	0.000, 0.143	0.000, 0.114	0.000, 0.143
0.000, 0.122	0.000, 0.107	0.000, 0.132	0.000, 0.135	0.000, 0.115
0.000, 0.130	0.000, 0.114	7.610, 0.134	0.000, 0.107	0.000, 0.137
0.000, 0.112	0.000, 0.077	0.000, 0.024	0.000, 0.140	0.000, 0.122
0.000, 0.123	0.000, 0.121	0.000, 0.125	0.000, 0.082	0.000, 0.123
0.000, 0.122	0.000, 0.129	0.000, 0.115	0.706, 0.134	0.000, 0.124
0.000, 0.128	0.000, 0.122	0.000, 0.134	0.000, 0.122	0.000, 0.119
0.000, 0.127	0.000, 0.120	0.000, 0.069	0.000, 0.395	0.000, 0.068
0.000, 0.149	0.000, 0.122	2.000, 0.129	0.000, 0.127	0.000, 0.122
0.000, 0.122	0.000, 0.114	0.000, 0.123	0.000, 0.122	0.000, 0.090
0.000, 0.127	6.000, -0.009	0.000, 0.122	0.000, 0.122	0.000, 0.107
0.000, 0.123	0.000, 0.122	0.000, 0.122	1.116, 0.133	0.000, 0.112
0.000, 0.122	0.000, 0.144	0.000, 0.136	0.000, 0.098	0.000, 0.112
0.000, 0.123	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.155	0.000, 0.131	0.000, 0.122	0.000, 0.133
0.000, 0.122	0.000, 0.112	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.137	0.000, 0.143	0.000, 0.094	0.000, 0.122	0.000, 0.142
0.000, 0.092	0.000, 0.087	0.000, 0.120	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.097	0.000, 0.126
0.000, 0.092	0.000, 0.122	0.000, 0.128	0.000, 0.082	0.000, 0.107
0.210, 0.142	0.000, 0.122	0.000, 0.128	0.000, 0.110	0.000, 0.122
0.000, 0.106	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.127	0.000, 0.095	0.000, 0.113	0.000, 0.027
0.000, 0.130	0.000, 0.123	0.000, 0.128	0.000, 0.112	0.000, 0.144
0.000, 0.122	0.000, 0.143	0.000, 0.105	0.000, 0.122	0.000, 0.041
0.000, 0.103	1.000, 0.132	0.000, 0.136	0.000, 0.055	0.000, 0.112
0.000, 0.363	0.000, 0.125	0.000, 0.119	0.000, 0.126	0.000, 0.122
0.000, 0.124	0.000, 0.059	0.000, 0.138	0.400, 0.058	0.000, 0.117
0.000, 0.141	0.465, 0.115	0.000, 0.125	0.000, 0.020	0.000, 0.091
0.000, 0.132	0.000, 0.078	0.000, 0.112	0.000, 0.100	0.000, 0.122
0.000, 0.136	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.099
0.000, 0.124	0.000, 0.133	0.000, 0.122	0.000, 0.127	0.000, 0.122
0.000, 0.134	0.000, 0.113	0.000, 0.123	0.000, 0.174	0.000, 0.122
0.000, 0.122	0.000, 0.078	0.000, 0.127	0.000, 0.137	0.000, 0.097
0.000, 0.104	0.000, 0.123	0.000, 0.122	0.000, 0.122	0.000, 0.118
0.000, 0.096	0.000, 0.124	0.000, -0.171	0.000, 0.122	0.000, 0.122
0.000, 0.136	0.000, 0.122	0.000, 0.133	0.000, 0.122	0.000, 0.078
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.097
0.000, 0.116	0.313, 0.143	0.640, 0.128	0.000, 0.122	0.000, 0.130
0.000, 0.122	0.000, 0.123	0.000, 0.089	0.000, 0.137	0.000, 0.128
0.000, 0.119	0.000, 0.113	0.000, 0.094	0.000, 0.122	0.000, 0.129
0.000, 0.118	0.000, 0.090	0.000, 0.122	0.000, 0.122	0.000, 0.127
0.000, 0.130	0.000, 0.105	0.000, 0.120	0.000, 0.095	1.000, 0.121
0.644, 0.122	0.000, 0.125	0.000, 0.122	0.000, 0.122	0.000, 0.091
0.000, 0.122	0.000, 0.096	0.000, 0.123	0.000, 0.122	0.000, 0.107
0.000, 0.114	0.000, 0.125	0.000, 0.122	0.000, 0.124	0.000, 0.122

0.000, 0.122	0.000, 0.122	0.000, 0.130	0.000, 0.114
0.000, 0.120	0.530, 0.130	0.000, 0.121	0.000, 0.113
0.000, 0.109	0.000, 0.107	0.000, 0.132	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.116	0.000, 0.122
0.000, 0.126	0.000, 0.122	0.000, 0.122	0.000, 0.079
0.000, 0.136	0.000, 0.122	0.000, 0.122	0.000, 0.122
3.000, 0.134	0.000, 0.029	0.000, 0.430	12.208, -
0.000, 0.127	10.164, -	0.000, 0.133	0.349
0.000, 0.122	0.134	0.000, 0.111	0.000, 0.150
0.000, 0.122	0.000, 0.122	0.000, 0.114	0.000, 0.122
0.197, 0.130	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.134	0.000, 0.122	
0.000, 0.122	0.848, 0.123	0.000, 0.122	
0.000, 0.052	0.000, 0.122	0.000, 0.110	
0.000, 0.122	0.000, 0.122	0.000, 0.119	
0.000, 0.122	0.000, 0.108	0.000, 0.125	
0.000, 0.116	0.000, 0.101	0.000, 0.113	
0.000, 0.122	0.000, 0.140	2.656, 0.172	
0.000, 0.084	0.000, 0.122	0.000, 0.112	
0.000, 0.122	0.000, 0.133	0.000, 0.127	
0.000, 0.114	0.000, 0.122	0.000, 0.072	
0.000, 0.123	0.000, 0.122	0.000, -0.012	
0.000, 0.122	0.000, 0.109	0.000, 0.122	
0.000, 0.122	0.000, 0.122	0.000, 0.119	
0.000, 0.122	0.000, 0.122	0.000, 0.127	
0.000, 0.122	0.000, 0.146	0.000, 0.137	
0.000, 0.131	0.000, 0.136	0.000, 0.122	
0.000, 0.122	0.000, 0.117	0.000, 0.109	
0.000, 0.122	0.000, 0.180	0.000, 0.143	
0.000, 0.143	0.000, 0.099	0.000, 0.122	
0.000, 0.130	0.000, 0.120	0.000, 0.122	
0.000, 0.122	0.000, 0.133	0.000, 0.122	
0.000, 0.132	2.448, 0.131	0.000, 0.122	
0.000, 0.122	0.000, 0.124	0.000, 0.122	
0.000, 0.122	0.000, 0.122	0.000, 0.092	
0.000, 0.099	0.000, 0.122	0.000, 0.056	
0.000, 0.131	0.000, -0.034	0.000, 0.124	
0.000, 0.122	0.080, 0.073	0.000, 0.122	
0.000, 0.102	0.000, 0.109	0.000, 0.118	
0.000, 0.127	1.970, 0.133	0.000, 0.103	
0.000, 0.141	0.000, 0.141	0.000, 0.118	
0.000, 0.126	0.000, 0.122	0.884, 0.109	
0.000, 0.122	0.000, 0.122	0.064, 0.118	
0.000, 0.132	0.000, 0.122	0.000, 0.125	
0.000, 0.123	4.000, 0.094	0.000, 0.118	
0.000, 0.143	0.000, 0.127	0.000, 0.121	

Fold 10	0.000, 0.134	0.000, 0.140	0.000, 0.154	0.000, 0.275
	0.000, 0.134	0.000, 0.129	0.000, 0.134	0.000, 0.128
0.000, 0.141	0.000, 0.131	0.000, 0.134	0.000, 0.134	0.000, 0.134
0.000, -0.109	0.000, 0.134	0.000, 0.128	0.000, 0.134	0.000, 0.137
0.000, 0.138	0.000, 0.134	0.000, 0.134	1.630, 0.134	0.000, 0.154
0.000, 0.134	0.000, 0.134	0.000, 0.127	0.000, 0.126	0.000, 0.134
0.000, 0.100	0.000, 0.090	0.000, 0.134	0.000, 0.139	0.000, 0.135
0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.133
0.000, 0.126	0.000, 0.134	0.000, 0.076	0.000, 0.128	0.288, 0.134
0.000, 0.126	0.000, 0.134	0.000, 0.134	0.000, 0.121	0.000, 0.134
0.000, 0.166	0.000, 0.134	0.000, 0.121	0.000, 0.138	0.000, 0.138
0.000, 0.134	0.000, 0.134	0.000, 0.143	0.000, 0.141	0.333, 0.166
0.000, 0.134	0.041, 0.122	0.000, 0.134	0.000, 0.145	0.000, 0.128
0.000, 0.147	0.000, 0.087	0.000, 0.127	0.695, 0.131	0.000, 0.134
0.000, 0.134	0.000, 0.131	0.000, -0.184	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.126	0.000, 0.134	0.000, 0.154	0.000, 0.126
0.000, 0.132	0.000, 0.108	0.000, 0.139	0.000, 0.119	0.000, 0.166
0.000, 0.166	0.000, 0.133	0.000, 0.134	0.000, 0.145	0.000, 0.112
0.000, 0.132	0.000, 0.123	0.000, 0.128	0.000, 0.088	0.000, 0.134
0.000, 0.156	0.000, 0.095	0.000, 0.037	0.000, 0.134	0.000, 0.134
0.000, 0.110	0.000, 0.130	0.000, 0.153	0.000, 0.076	0.000, 0.160
0.000, 0.134	0.000, 0.144	0.000, 0.132	0.456, 0.147	0.000, 0.134
0.000, 0.134	0.000, 0.154	0.000, 0.144	0.000, 0.139	0.000, 0.134
0.000, -0.038	0.000, 0.090	0.000, 0.166	0.000, 0.130	0.000, 0.133
0.000, 0.157	3.000, 0.701	1.000, 0.144	0.000, 0.095	0.000, 0.083
0.000, 0.166	0.000, 0.139	0.000, 0.128	0.000, 0.166	0.000, 0.143
0.000, 0.137	2.000, 0.118	0.000, 0.144	0.000, 0.134	0.000, 0.132
0.000, 0.137	0.000, 0.134	0.000, 0.152	0.000, 0.138	0.000, 0.134
0.000, 0.160	0.000, 0.059	0.000, 0.140	0.000, 0.134	0.000, 0.206
0.000, 0.134	0.000, 0.136	0.000, 0.142	0.000, 0.134	3.000, 0.104
0.000, 0.125	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.134
0.000, 0.117	0.000, 0.166	0.000, 0.105	0.000, 0.134	0.000, 0.158
0.000, 0.090	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.134
0.000, 0.166	0.000, 0.099	0.000, 0.134	0.000, 0.134	0.000, 0.139
0.000, 0.134	0.000, -0.086	0.000, 0.134	0.000, 0.113	0.000, 0.134
0.000, 0.147	0.000, 0.134	0.000, 0.138	0.000, 0.134	0.000, 0.134
0.000, 0.130	0.000, 0.144	0.000, 0.134	0.000, 0.137	0.000, 0.117
0.000, 0.134	0.000, 0.132	0.000, 0.075	0.000, 0.224	0.000, 0.120
1.314, 0.136	0.000, 0.139	0.000, 0.128	0.000, 0.138	0.000, 0.134
0.000, 0.130	0.000, 0.134	0.000, 0.129	0.000, 0.099	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.122	0.000, -0.708	0.000, 0.106
0.000, 0.131	0.000, 0.118	0.000, 0.148	0.000, 0.126	0.000, 0.134
0.000, 0.073	0.000, 0.133	0.000, 0.134	0.000, 0.134	0.000, 0.134
0.000, 0.154	0.000, 0.134	0.000, 0.134	0.000, 0.138	0.000, 0.166
0.000, 0.135	0.000, 0.098	0.000, 0.142	0.000, 0.149	0.000, 0.134
0.000, 0.159	0.000, 0.134	0.000, 0.134	0.000, 0.130	0.000, 0.142

0.000, 0.136	0.000, 0.140	0.000, 0.133	0.000, 0.134	0.000, 0.119
0.000, 0.118	0.000, 0.070	0.000, 0.150	0.000, 0.129	0.000, 0.153
0.303, 0.134	0.000, 0.153	0.000, 0.112	0.000, 0.145	0.000, 0.166
0.000, 0.125	0.000, 0.141	0.000, 0.134	0.000, 0.154	0.000, 0.137
0.000, 0.162	0.000, 0.149	0.000, 0.117	0.000, 0.134	0.000, 0.134
0.000, 0.135	0.000, 0.134	0.000, 0.145	0.000, 0.137	0.000, 0.146
0.000, 0.155	0.000, 0.125	1.000, 0.144	0.000, 0.129	0.000, 0.146
0.000, 0.118	0.000, 0.134	0.000, 0.134	0.417, 0.132	0.000, 0.147
0.000, 0.140	0.000, 0.166	0.000, 0.139	0.000, 0.134	0.000, 0.134
0.000, 0.141	0.000, 0.268	0.000, 0.134	0.000, 0.134	0.000, 0.138
0.000, 0.148	0.000, 0.134	0.000, 0.123	0.000, 0.142	0.000, 0.029
0.000, 0.137	0.000, 0.154	0.000, 0.145	0.000, 0.151	3.000, 0.146
0.000, 0.134	0.000, 0.166	0.000, 0.134	0.000, 0.137	0.000, 0.134
0.000, 0.134	0.000, 0.142	0.000, 0.109	0.000, 0.124	0.000, 0.036
0.000, 0.134	0.000, 0.160	0.000, 0.134	0.000, 0.055	0.000, 0.159
0.000, 0.134	0.000, 0.154	0.000, 0.135	0.000, 0.166	0.000, 0.141
0.000, 0.130	0.000, 0.134	0.000, 0.158	0.000, 0.139	0.000, 0.128
0.000, 0.134	0.000, 0.149	0.000, 0.132	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.166	0.000, 0.134	0.000, 0.145	0.000, 0.125
0.000, 0.134	0.000, 0.134	0.000, 0.156	0.000, 0.166	0.000, 0.148
0.000, 0.128	0.000, 0.129	0.000, 0.132	0.000, 0.138	0.000, 0.134
0.000, 0.149	1.000, 0.137	0.000, 0.128	0.000, 0.134	0.000, 0.135
0.000, 0.134	0.000, 0.097	0.000, 0.109	0.000, 0.101	0.000, 0.121
0.000, -0.007	0.000, 0.134	0.000, 0.118	0.000, 0.134	0.000, 0.166
0.000, 0.166	0.000, 0.140	0.000, 0.134	0.000, 0.134	0.000, 0.124
0.000, 0.134	3.000, 0.402	0.000, 0.128	0.000, 0.136	0.000, 0.124
0.000, 0.134	0.000, 0.124	0.000, 0.134	0.000, 0.134	0.000, 0.131
0.000, 0.134	0.000, 0.134	0.000, 0.144	0.000, 0.276	0.000, 0.135
0.000, 0.148	0.709, 0.127	0.000, 0.134	0.000, 0.133	0.000, 0.080
0.000, 0.039	0.000, 0.144	0.180, 0.132	0.000, 0.134	3.666, 0.104
0.000, 0.121	1.400, 0.115	0.000, 0.161	0.000, 0.111	0.000, 0.134
0.000, 0.105	0.000, 0.134	0.000, 0.152	0.000, 0.166	2.000, 0.178
0.000, 0.119	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.148
0.000, 0.134	0.000, 0.156	0.000, 0.129	0.000, 0.137	0.000, 0.139
0.000, 0.101	0.000, 0.140	0.000, 0.132	0.000, 0.126	0.000, 0.134
0.000, 0.134	0.132, 0.182	0.650, 0.107	0.000, 0.134	1.225, 0.130
0.000, 0.111	0.000, 0.136	0.000, 0.134	0.000, 0.162	0.602, 0.136
0.000, -0.027	0.000, 0.134	0.000, 0.139	0.000, 0.094	0.000, 0.128
0.000, 0.101	0.000, 0.134	0.000, 0.076	0.000, 0.116	0.000, 0.083
0.000, 0.139	0.000, 0.134	0.000, 0.135	0.000, 0.166	0.000, 0.135
0.000, 0.166	0.000, 0.138	0.000, 0.134	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.149	0.000, 0.144	0.000, 0.132	0.000, 0.139
0.000, 0.124	0.000, 0.142	0.000, 0.157	0.000, 0.129	0.000, 0.057
0.000, 0.170	0.000, 0.134	0.000, 0.134	0.000, 0.131	0.000, 0.134
0.000, 0.134	0.000, 0.127	0.000, 0.132	0.000, 0.133	0.000, 0.132
0.000, 0.129	0.000, 0.166	0.000, 0.142	0.000, 0.129	0.000, 0.134

0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.141	0.000, 0.161
0.000, 0.134	0.000, 0.129	5.000, 0.139	0.000, -0.827	0.000, 0.134
0.000, 0.156	0.000, 0.134	0.000, 0.080	0.000, 0.107	0.000, 0.138
0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, -0.026
0.000, 0.186	0.000, 0.134	0.000, 0.134	0.000, 0.130	0.000, 0.140
0.000, 0.134	0.000, 0.134	0.000, -0.130	0.070, 0.166	0.000, 0.148
0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.124	0.000, 0.080
0.000, 0.134	0.000, 0.122	0.000, 0.134	0.000, 0.134	0.000, 0.154
0.000, 0.087	0.000, 0.131	0.000, 0.134	0.000, 0.099	0.000, 0.128
0.000, 0.134	0.000, 0.154	0.000, -0.036	0.000, 0.176	0.000, -0.022
0.000, 0.081	0.000, 0.166	0.000, 0.141	0.000, 0.134	0.000, 0.131
0.000, 0.125	0.000, 0.134	0.000, 0.133	0.000, 0.071	0.000, 0.134
0.000, 0.134	0.000, 0.133	0.000, 0.134	0.000, 0.134	4.983, 0.140
0.000, 0.134	0.000, 0.134	0.000, 0.138	0.000, 0.134	0.000, 0.134
0.000, 0.145	0.000, 0.148	0.000, 0.124	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.149	0.000, 0.106	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.123	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.122	0.000, 0.134	0.000, 0.166
0.000, 0.134	0.000, 0.128	0.000, 0.134	0.000, 0.136	0.000, 0.134
0.000, 0.112	0.000, 0.036	0.000, 0.134	0.000, 0.121	0.000, 0.133
0.000, 0.101	0.000, 0.148	0.000, 0.134	0.000, 0.148	0.000, 0.134
0.000, 0.138	0.000, 0.157	0.000, 0.154	0.000, 0.189	0.000, 0.153
0.000, 0.134	0.000, 0.134	0.430, 0.275	0.000, 0.134	0.000, 0.108
0.000, 0.134	0.000, 0.142	0.000, 0.122	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.124	0.000, 0.134	0.000, 0.134
0.000, 0.148	0.000, 0.138	0.000, 0.071	0.000, 0.131	0.000, 0.142
0.000, 0.134	0.000, 0.134	0.000, 0.148	0.000, 0.110	0.000, 0.134
1.000, 0.009	0.000, 0.134	0.000, -0.364	0.000, 0.134	0.000, 0.134
0.000, 0.118	0.000, 0.134	0.000, 0.126	0.000, 0.095	0.000, 0.134
0.000, 0.152	0.000, 0.134	0.000, 0.134	0.976, 0.170	1.448, 0.136
0.000, 0.134	0.000, 0.140	0.000, 0.248	0.000, 0.134	0.000, 0.120
0.000, 0.156	0.000, 0.132	1.970, 0.145	0.000, 0.126	0.000, 0.134
0.000, 0.148	0.000, 0.153	0.000, 0.134	0.000, 0.134	0.000, 0.126
0.000, 0.145	0.000, 0.131	0.000, 0.180	0.000, 0.138	0.000, 0.134
0.000, 0.134	0.000, 0.134	0.000, 0.144	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.166	0.000, 0.128	0.000, 0.123	0.000, 0.148
0.000, 0.119	0.000, 0.139	0.000, 0.134	0.000, 0.134	0.000, 0.147
0.000, 0.166	0.000, 0.133	0.000, 0.130	0.000, 0.134	0.000, 0.134
0.000, 0.134	0.000, 0.125	0.000, 0.123	0.000, 0.117	0.000, 0.134
0.000, 0.115	0.000, 0.134	0.000, 0.134	0.000, 0.134	0.000, 0.130
0.000, 0.130	0.000, 0.166	0.000, 0.134	0.000, 0.147	0.000, 0.134
0.000, 0.151	1.023, 0.130	0.000, 0.134	0.000, 0.134	0.000, 0.136
0.000, 0.107	0.000, 0.145	0.000, 0.095	0.000, 0.133	0.000, 0.128
0.000, 0.132	0.000, 0.207	0.000, 0.134	0.070, 0.149	0.000, 0.150
0.000, 0.134	3.775, 0.058	0.000, 0.159	0.000, 0.116	0.000, 0.134
0.000, 0.112	0.000, 0.134	0.000, 0.276	0.000, 0.134	0.000, 0.134

0.000, 0.114	0.000, 0.073	0.000, 0.134	0.000, 0.131
0.000, 0.129	0.000, 0.106	0.000, 0.119	0.000, 0.134
0.000, 0.133	0.000, 0.166	0.000, 0.166	0.000, 0.134
0.000, 0.117	0.000, 0.134	0.000, 0.132	0.000, 0.135
0.000, 0.121	0.000, 0.134	0.000, 0.110	0.000, 0.126
0.000, 0.124	0.000, 0.148	0.000, 0.136	0.000, 0.134
0.000, 0.134	0.000, 0.120	0.000, 0.122	0.000, 0.166
0.000, 0.149	0.000, 0.133	0.000, 0.121	0.000, 0.134
0.000, 0.134	0.000, 0.130	0.000, 0.134	
0.130, 0.154	0.000, 0.134	0.165, 0.114	
0.000, 0.140	0.000, 0.134	0.000, 0.144	
0.000, 0.127	0.000, 0.134	1.203, 0.128	
0.000, 0.141	0.000, 0.156	0.000, 0.132	
0.000, 0.343	0.000, 0.134	0.000, 0.134	
0.000, 0.148	0.000, 0.134	0.000, 0.132	
0.000, 0.057	0.000, 0.123	0.000, 0.139	
0.000, 0.104	0.000, 0.135	0.000, 0.134	
0.000, 0.101	0.000, 0.129	0.000, 0.127	
0.000, 0.138	0.000, 0.015	0.000, 0.134	
0.000, 0.159	0.000, 0.134	0.000, 0.134	
0.000, 0.122	0.000, 0.088	0.000, 0.136	
0.000, 0.142	0.000, 0.129	0.000, 0.134	
0.000, 0.134	0.000, 0.140	0.000, 0.134	
0.000, 0.134	0.000, 0.137	0.000, 0.134	
1.527, 0.101	0.000, 0.134	0.000, 0.134	
0.000, 0.136	0.000, 0.134	0.000, 0.149	
0.000, 0.134	0.000, 0.098	0.000, 0.151	
0.000, 0.134	17.075, 0.118	0.000, 0.134	
0.000, 0.107	0.000, 0.133	0.000, 0.134	
0.000, 0.146	0.000, 0.148	0.000, 0.130	
0.000, 0.131	0.000, 0.134	0.000, 0.134	
0.000, 0.131	0.000, 0.166	0.400, 0.133	
0.000, 0.134	0.000, 0.134	0.000, 0.134	
0.000, 0.130	4.030, 0.240	0.000, 0.166	
0.000, 0.134	1.000, 0.122	0.000, 0.134	
2.523, 0.122	0.000, 0.086	0.000, 0.044	
0.000, 0.134	0.000, 0.143	0.000, 0.134	
0.000, 0.134	0.000, 0.134	0.000, 0.131	
3.370, 0.130	0.000, 0.134	0.665, 0.107	
0.000, 0.134	0.000, -0.023	0.000, 0.134	
0.000, 0.161	0.000, 0.136	0.000, 0.138	
0.000, 0.141	0.000, 0.134	0.000, 0.134	
0.000, 0.153	1.066, 0.130	0.000, 0.134	
0.000, 0.146	0.000, 0.124	0.000, 0.134	
0.000, 0.134	0.000, 0.153	0.000, 0.137	
3.000, 0.129	0.000, 0.134	0.000, 0.070	

Reduced dataset

Fold 1	1.000, 0.211	0.000, 0.141	0.000, 0.064	0.000, 0.161
	0.000, 0.121	0.000, 0.110	0.000, 0.107	0.000, 0.110
0.000, 0.080	0.000, 0.150	0.000, 0.110	0.000, 0.110	0.000, 0.110
0.000, 0.140	0.000, 0.110	0.000, 0.087	0.000, -0.094	0.000, 0.079
0.000, -0.006	0.000, 0.095	0.000, 0.110	0.000, 0.181	0.000, 0.158
0.000, 0.110	1.000, 0.158	0.000, 0.179	0.000, 0.146	0.000, 0.185
0.000, 0.110	0.000, 0.146	0.000, 0.110	0.000, -0.066	0.000, 0.144
0.375, 0.146	0.000, 0.072	0.000, 0.110	0.000, 0.141	0.000, 0.114
0.000, 0.136	0.000, 0.125	0.000, 0.110	0.000, 0.132	0.000, 0.110
0.000, -0.006	0.000, 0.146	0.000, 0.157	0.000, 0.143	0.000, 0.198
0.000, 0.110	0.000, 0.140	0.000, 0.166	0.000, 0.110	0.000, 0.167
0.000, 0.110	0.000, 0.106	0.000, 0.125	0.000, 0.140	0.000, 0.110
0.000, 0.136	0.000, 0.110	0.000, 0.110	0.000, 0.110	0.000, 0.110
0.000, 0.110	0.000, 0.175	5.760, 0.110	0.000, 0.110	0.000, 0.110
0.000, 0.159	0.000, 0.146	0.000, 0.091	0.000, -0.092	0.000, 0.137
0.000, 0.154	3.000, 0.125	0.000, 0.142	0.000, 0.179	0.000, 0.116
0.187, 0.172	0.000, 0.114	2.000, 0.097	0.000, 0.198	0.000, 0.110
0.000, 0.110	0.000, 0.110	0.000, 0.029	0.000, 0.134	0.000, 0.146
0.000, 0.146	0.000, 0.110	0.000, 0.143	0.000, 0.133	0.000, 0.126
0.000, 0.110	0.000, 0.110	0.000, 0.114	0.000, 0.146	0.000, 0.108
0.000, 0.110	0.000, 0.134	0.000, 0.142	0.000, 0.133	0.000, 0.107
0.000, 0.125	0.000, 0.110	0.000, 0.143	0.000, 0.181	0.000, -0.094
0.000, 0.171	0.000, 0.110	0.000, 0.117	0.000, 0.110	0.000, 0.170
0.000, 0.110	0.000, 0.148	0.000, 0.110	0.000, 0.110	0.000, -0.116
0.000, 0.110	0.000, 0.151	0.000, 0.110	0.000, 0.068	0.000, 0.141
0.000, 0.110	0.000, 0.143	0.000, 0.125	0.000, 0.110	0.000, 0.110
0.000, 0.151	1.000, 0.134	1.314, 0.144	0.000, 0.119	0.000, 0.110
0.000, 0.110	0.000, 0.125	0.000, 0.018	0.000, 0.110	0.000, 0.139
0.000, 0.141	0.000, 0.128	0.000, 0.127	0.000, 0.110	0.000, 0.110
0.000, 0.137	0.000, -0.020	0.000, 0.110	0.000, 0.074	0.000, 0.170
0.000, 0.083	0.000, 0.110	0.000, 0.110	0.000, 0.191	0.000, -0.017
0.000, 0.110	0.000, 0.144	0.000, 0.110	0.000, 0.077	0.000, 0.141
0.000, 0.165	0.000, 0.140	0.000, 0.146	0.000, 0.096	3.000, 0.159
0.000, 0.163	0.000, 0.139	0.000, 0.110	0.000, 0.049	0.000, 0.110
0.000, 0.149	0.000, 0.110	0.000, 0.200	0.000, 0.110	0.000, 0.135
0.000, 0.110	0.000, 0.118	0.000, 0.110	0.000, -0.321	0.000, 0.110
0.000, 0.160	0.000, 0.016	0.000, 0.094	0.000, 0.125	0.000, 0.110
0.000, 0.069	0.000, 0.128	0.000, 0.110	0.000, 0.153	0.000, 0.138
0.884, 0.159	0.000, 0.107	0.000, 0.112	0.000, 0.136	0.000, 0.048
0.000, 0.153	0.000, 0.110	0.000, 0.015	0.000, 0.074	0.000, 0.110
0.000, 0.011	0.000, 0.110	0.000, 0.110	0.000, -0.027	0.000, 0.146
0.000, 0.145	0.000, 0.136	0.000, 0.146	0.000, 0.110	0.000, 0.138
0.000, 0.130	0.000, 0.110	0.000, 0.135	0.000, 0.110	6.000, 0.135
0.000, 0.151	0.000, 0.110	2.000, 0.140	0.000, 0.141	0.000, 0.037

0.000, 0.116	0.000, 0.079	0.000, 0.110	0.000, 0.163	0.000, 0.110
0.000, 0.170	0.000, 0.077	0.000, 0.160	2.000, 0.171	0.000, 0.114
0.000, 0.183	0.000, 0.110	0.000, 0.117	0.000, 0.110	0.000, 0.110
0.000, 0.110	0.000, 0.079	0.000, 0.139	0.000, 0.110	0.000, 0.079
0.000, 0.110	0.000, 0.077	0.000, 0.110	3.000, 0.103	0.000, 0.142
0.000, 0.172	0.000, 0.110	0.000, 0.110	0.000, 0.092	0.000, 0.110
0.000, 0.126	1.000, 0.113	0.000, 0.076	0.000, 0.086	0.000, 0.124
0.000, 0.110	0.000, 0.110	0.000, 0.110	1.000, 0.141	0.000, 0.110
0.000, 0.110	2.000, 0.047	0.000, 0.120	0.000, -0.040	0.000, 0.146
0.000, 0.110	1.143, 0.137	0.000, 0.135	0.000, 0.110	0.000, 0.143
0.000, 0.142	0.000, 0.135	0.000, 0.168	0.000, 0.149	0.000, 0.156
0.000, 0.048	0.000, 0.141	0.000, 0.040	0.000, 0.110	0.000, 0.110
0.000, 0.128	0.000, 0.110	0.000, 0.114	0.000, 0.157	0.000, 0.110
0.000, 0.089	0.000, 0.109	0.000, 0.110	0.000, 0.100	0.000, 0.110
0.000, 0.105	0.308, 0.159	0.000, 0.123	0.000, 0.012	0.000, -0.079
0.000, -0.052	0.000, 0.183	0.000, 0.101	0.000, 0.110	0.000, 0.110
0.000, 0.110	0.000, 0.101	0.000, 0.170	0.000, 0.110	0.000, 0.110
0.000, -0.004	0.000, 0.110	0.000, 0.160	0.000, 0.164	0.000, 0.468
0.000, 0.110	0.000, 0.128	0.000, 0.149	0.000, -0.058	0.000, 0.110
0.000, 0.110	0.000, -0.021	0.000, 0.110	0.000, 0.156	0.000, 0.090
0.000, 0.130	1.000, 0.134	0.000, 0.110	0.000, -0.165	0.000, 0.143
0.000, 0.110	0.000, 0.126	0.000, 0.110	0.516, 0.148	0.000, 0.146
0.000, 0.168	0.000, 0.160	0.000, 0.110	0.000, 0.110	0.000, 0.158
0.000, 0.033	0.000, 0.135	6.000, 0.122	0.000, 0.175	0.000, 0.150
0.000, 0.249	0.000, 0.110	0.000, 0.074	0.000, 0.110	0.000, 0.046
0.000, 0.110	0.000, 0.102	0.000, 0.148	0.000, 0.066	0.000, 0.098
0.000, 0.110	0.000, 0.036	0.400, 0.149	0.000, 0.110	0.000, -0.114
0.000, 0.147	0.000, 0.144	0.000, 0.110	0.000, 0.110	0.000, 0.146
0.000, 0.142	0.000, 0.150	0.000, 0.130	0.000, 0.157	0.000, 0.110
0.638, 0.117	0.000, 0.133	0.000, 0.141	0.000, 0.146	0.000, 0.172
0.000, 0.159	0.000, -0.081	0.000, 0.110	0.090, 0.176	0.000, 0.159
0.000, 0.068	0.000, -0.049	0.000, 0.064	0.000, 0.110	0.524, 0.159
0.000, 0.118	0.000, 0.123	0.000, 0.110	0.654, 0.147	0.000, 0.110
0.000, 0.142	0.000, 0.517	0.000, 0.110	0.000, 0.128	0.000, 0.128
2.844, 0.110	0.000, 0.110	0.000, 0.110	0.000, 0.110	0.000, 0.146
0.000, 0.061	1.347, 0.165	0.072, 0.080	0.000, 0.090	0.000, 0.160
0.000, 0.145	0.000, 0.110	0.000, 0.062	0.000, 0.146	0.000, 0.132
0.000, 0.110	0.000, 0.486	0.000, 0.110	0.000, 0.110	0.000, 0.159
1.000, 0.128	0.000, 0.110	0.000, -0.047	0.000, 0.132	0.000, 0.110
0.000, 0.110	0.000, 0.110	0.000, 0.174	0.000, 0.150	0.000, 0.110
0.000, 0.110	0.000, 0.136	0.000, 0.143	0.000, 0.110	0.000, 0.110
0.140, 0.156	0.000, 0.110	0.000, 0.110	0.000, -0.036	0.000, 0.087
0.000, 0.148	0.000, 0.110	0.000, 0.110	0.000, 0.129	0.000, 0.146
0.000, 0.105	0.000, 0.135	0.000, 0.110	0.000, 0.110	0.000, 0.110
0.320, 0.181	0.000, 0.110	0.000, 0.177	0.000, 0.104	0.000, 0.110
0.000, 0.015	0.000, 0.111	0.000, 0.110	0.000, 0.002	0.000, -0.216

0.000, 0.127	0.000, 0.135	0.000, 0.110	0.000, 0.096	0.000, 0.072
0.000, 0.110	0.000, 0.254	0.000, 0.110	0.000, 0.146	0.000, 0.168
0.000, 0.003	0.000, -3.922	0.000, 0.065	0.000, 0.173	0.000, 0.146
0.000, 0.110	0.000, 0.190	0.000, 0.110	0.000, 0.135	0.000, 0.110
0.000, 0.140	0.000, 0.073	0.000, 0.122	0.000, 0.110	0.000, 0.115
0.000, 0.134	0.000, 0.163	0.000, 0.178	0.000, -0.475	0.000, 0.110
0.000, 0.146	0.000, 0.140	0.000, 0.154	0.000, 0.110	0.000, 0.145
0.000, 0.110	0.000, 0.110	0.000, 0.149	0.000, 0.185	0.000, 0.134
0.000, 0.163	0.000, 0.115	0.000, 0.097	0.000, 0.129	0.000, 0.159
0.000, 0.111	0.000, 0.110	0.000, 0.167	0.000, 0.110	0.000, 0.170
0.000, 0.110	1.586, 0.120	0.000, 0.110	0.000, 0.110	0.000, -0.001
0.000, 0.110	0.000, 0.146	0.000, 0.110	0.000, 0.164	0.000, 0.186
0.000, 0.091	0.000, 0.110	0.000, 0.143	0.000, 0.133	0.000, 0.167
0.000, 0.130	0.000, 0.173	0.000, 0.009	0.000, -0.147	0.000, 0.114
0.000, 0.110	0.000, 0.110	0.000, 0.110	0.000, 0.102	0.000, 0.182
0.000, -0.023	0.000, 0.071	0.000, 0.110	0.000, 0.134	0.000, 0.143
0.000, 0.091	0.074, 0.111	0.000, 0.110	0.000, 0.127	0.000, 0.110
0.000, 0.032	0.000, 0.110	5.000, 0.142	0.000, 0.110	0.000, 0.114
0.000, 0.181	0.000, 0.183	0.000, 0.102	0.000, 0.125	2.442, 0.128
0.000, 0.110	0.000, 0.150	0.000, 0.110	0.000, -0.003	0.000, -0.061
0.000, 0.110	0.000, 0.110	0.000, 0.149	0.000, 0.110	1.000, 0.146
0.000, 0.141	1.300, 0.130	0.000, 0.110	0.000, 0.149	0.000, 0.110
0.000, 0.136	0.000, 0.110	0.000, 0.137	0.000, 0.028	0.000, 0.110
3.850, 0.171	0.000, 0.146	4.000, 0.143	0.000, 0.174	0.000, 0.065
0.000, 0.148	0.000, 0.182	0.000, 0.141	0.000, 0.110	0.000, 0.185
0.000, 0.134	0.000, 0.146	0.000, 0.110	0.000, -0.047	0.000, 0.142
0.000, 0.177	0.000, 0.085	0.000, 0.110	0.000, 0.025	0.000, 0.142
0.000, 0.159	0.000, 0.172	0.000, 0.110	0.000, 0.157	0.000, 0.135
0.000, 0.110	0.000, 0.114	0.000, 0.091	0.000, 0.110	0.000, 0.110
0.000, 0.110	0.000, 0.154	0.000, 0.140	0.000, 0.009	0.000, 0.028
0.000, 0.110	0.000, 0.178	0.000, 0.110	0.000, 0.110	0.000, 0.110
0.000, 0.113	0.000, 0.148	0.000, 0.110	0.000, 0.167	0.000, 0.127
0.000, 0.130	0.000, 0.110	0.000, 0.166	0.000, 0.110	0.000, 0.158
0.000, 0.146	0.000, 0.110	0.848, 0.110	0.000, 0.074	0.000, 0.114
0.000, 0.110	0.476, 0.178	0.000, 0.077	0.000, 0.128	0.000, 0.012
0.000, 0.027	0.000, -0.075	0.000, 0.084	0.000, 0.110	0.000, 0.170
0.000, 0.135	0.000, 0.065	0.000, 0.106	0.000, 0.124	0.000, 0.110
0.000, 0.142	0.000, 0.078	0.000, 0.000	0.000, 0.054	0.000, 0.110
1.000, -0.048	0.000, 0.129	0.000, 0.110	0.000, 0.099	0.000, 0.171
0.000, 0.110	0.000, 0.146	0.000, 0.123	0.000, 0.125	0.000, 0.110
0.000, 0.115	0.000, 0.110	0.000, 0.170	0.000, -0.101	0.000, 0.110
0.000, 0.076	0.000, 0.241	0.000, 0.014	0.000, 0.110	0.000, 0.138
0.000, -0.229	0.000, 0.110	0.000, 0.106	0.000, 0.131	0.000, 0.118
0.000, -0.141	0.000, 0.129	0.000, 0.162	0.000, 0.110	0.000, 0.077
0.000, 0.129	0.000, 0.142	0.000, 0.156	0.000, 0.147	0.000, 0.110
0.000, 0.146	0.000, 0.136	0.000, 0.110	0.000, 0.110	0.000, 0.020

0.000, 0.153	0.000, 0.143	0.000, 0.183	0.000, 0.110
0.000, 0.110	0.000, 0.111	0.000, 0.120	0.000, -0.146
0.000, 0.110	0.000, 0.107	0.000, 0.110	0.000, 0.110
0.000, 0.144	0.000, -0.007	0.000, 0.110	0.000, 0.119
0.000, 0.110	0.000, 0.170	0.000, 0.110	0.000, 0.137
0.000, 0.148	0.000, 0.159	0.150, 0.128	0.000, 0.146
0.000, 0.172	0.000, 0.178	0.000, 0.110	0.000, 0.134
0.000, 0.150	0.000, 0.110	0.000, 0.147	0.000, 0.090
0.000, 0.049	0.000, 0.088	0.000, 0.110	0.000, 0.110
0.000, 0.145	0.000, 0.183	0.000, 0.130	0.000, 0.098
0.000, 0.142	0.000, 0.110	0.000, 0.109	4.000, 0.128
0.165, 0.147	0.000, 0.110	0.000, 0.167	0.000, 0.021
0.000, 0.110	0.000, 0.111	0.000, 0.110	0.000, 0.172
0.000, 0.110	0.000, 0.110	0.000, 0.110	0.000, 0.110
0.000, 0.110	0.000, 0.198	0.000, 0.110	0.000, 0.136
0.000, 0.162	0.000, 0.110	0.000, 0.110	0.000, 0.169
0.000, 0.110	0.000, 0.110	0.000, 0.140	0.000, 0.114
0.000, 0.110	0.000, 0.153	0.000, 0.110	0.000, 0.110
0.000, 0.142	0.000, 0.110	0.000, -0.035	0.000, 0.110
0.000, 0.142	0.000, 0.110	0.714, 0.099	
0.000, 0.110	0.000, 0.110	0.000, 0.110	
0.000, 0.142	0.000, 0.110	0.000, 0.110	
0.000, 0.165	0.000, 0.158	0.000, 0.091	
0.000, 0.110	0.000, 0.110	0.000, 0.146	
0.000, 0.159	0.000, 0.145	0.000, 0.146	
0.000, 0.112	0.000, 0.110	0.000, 0.068	
0.000, 0.126	0.000, 0.129	0.000, 0.141	
0.000, 0.110	0.000, 0.110	0.000, 0.110	
0.000, 0.172	0.000, 0.115	0.000, 0.167	
0.000, 0.421	0.000, 0.118	0.000, 0.110	
0.000, 0.110	0.000, 0.082	0.000, 0.192	
0.000, 0.068	0.000, 0.140	0.000, 0.110	
0.000, 0.110	0.964, 0.150	0.000, 0.050	
0.000, 0.110	0.000, 0.110	0.000, 0.162	
0.000, 0.114	0.000, -0.001	0.000, 0.118	
0.000, 0.137	0.076, 0.126	0.000, 0.115	
0.000, 0.110	0.000, 0.110	0.000, 0.110	
0.000, 0.150	0.000, 0.110	0.000, 0.110	
0.000, 0.090	0.000, 0.150	0.000, 0.110	
0.000, 0.121	0.000, 0.110	0.000, 0.110	
0.000, 0.110	0.000, 0.075	0.000, 0.110	
0.000, 0.110	0.000, 0.007	2.844, 0.168	
0.000, 0.143	0.000, 0.145	0.000, 0.110	
0.000, 0.110	0.000, 0.109	0.000, 0.110	
0.000, 0.139	0.000, 0.110	0.000, 0.118	
0.000, 0.045	0.000, 0.035	0.000, 0.093	

Fold 2	0.000, 0.000	0.000, 0.000	0.000, -0.027	21.552, 0.122
	0.000, 0.022	0.000, 0.000	0.000, -0.008	0.000, 0.021
0.000, 0.019	0.000, 0.015	0.000, 0.024	0.000, 0.018	0.000, 0.028
0.000, 0.000	0.000, 0.032	0.000, 0.018	0.000, -0.004	0.000, 0.032
0.000, 0.000	0.000, 0.008	0.000, 0.000	0.000, 0.021	0.110, 0.009
0.000, 0.014	0.000, 0.000	0.000, 0.011	0.000, 0.000	0.000, 0.027
17.075, 0.012	0.000, -0.006	0.000, 0.029	0.000, 0.000	0.000, 0.000
0.000, 0.023	0.000, 0.000	0.000, 0.024	0.000, 0.000	0.000, 0.000
0.000, 0.007	0.000, 0.000	0.000, 0.025	0.000, 0.000	0.000, 0.023
0.000, 0.061	0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.027	0.000, 0.002	0.000, 0.000	0.000, -0.010
0.000, 0.000	0.000, 0.018	0.000, 0.003	0.000, 0.022	0.000, 0.000
0.000, 0.012	0.350, 0.019	0.000, 0.000	0.000, 0.019	0.000, 0.024
0.000, -0.049	0.000, 0.014	0.000, 0.000	1.000, 0.015	0.000, 0.000
0.000, 0.000	0.000, 0.017	0.000, 0.023	0.000, 0.016	0.000, 0.019
0.000, 0.035	0.000, 0.021	0.000, 0.000	0.000, 0.018	0.000, 0.000
0.000, 0.022	0.000, 0.024	0.000, 0.024	0.000, 0.024	0.000, 0.000
0.000, 0.000	0.000, 0.012	0.000, 0.019	0.000, 0.000	0.000, 0.032
104.000,	6.017, 0.023	23.000, 0.019	0.000, 0.012	0.000, 0.020
0.405	0.000, 0.024	0.000, 0.027	2.000, -0.195	0.000, 0.000
0.000, 0.000	0.000, 0.036	0.000, 0.014	0.000, 0.000	0.000, 0.028
0.000, 0.023	0.000, 0.031	0.000, -0.005	0.000, 0.023	0.000, 0.024
0.000, 0.016	0.000, 0.000	0.000, 0.021	0.000, 0.000	0.000, 0.024
0.000, 0.000	0.000, -0.012	0.000, 0.000	0.000, 0.021	0.000, 0.021
0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.010	0.000, 0.038
0.000, 0.000	0.000, 0.024	0.000, 0.001	0.000, 0.027	0.000, 0.011
0.000, 0.000	0.000, 0.005	0.000, 0.018	0.000, 0.020	0.000, 0.000
0.000, 0.000	0.000, 0.017	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.028	0.000, 0.024	0.000, 0.022	0.000, 0.034	0.000, 0.001
0.000, 0.031	0.000, 0.022	0.000, 0.015	0.000, 0.030	0.000, 0.021
0.000, 0.024	0.000, 0.000	0.000, 0.020	0.000, 0.028	0.000, 0.027
0.000, 0.027	0.000, 0.028	0.000, 0.022	0.000, 0.000	0.000, 0.033
0.000, 0.031	0.000, -0.002	0.000, 0.027	0.000, 0.031	0.000, 0.015
0.000, 0.040	0.000, 0.000	0.000, 0.022	0.000, 0.017	0.000, 0.000
0.000, 0.020	0.000, 0.000	0.000, 0.031	0.000, 0.000	1.000, 0.242
0.000, 0.000	0.000, 0.024	0.000, 0.000	0.000, 0.000	0.000, 0.012
0.000, -0.015	0.000, 0.000	0.000, 0.000	0.000, 0.018	0.000, 0.000
0.000, 0.021	0.709, 0.017	0.000, 0.032	0.000, 0.000	0.000, 0.000
0.000, 0.027	0.000, 0.020	0.000, 0.020	0.000, 0.027	0.000, 0.000
0.000, 0.013	0.000, -0.027	0.000, 0.024	0.000, 0.020	0.000, 0.029
0.000, 0.000	0.000, 0.000	0.000, 0.009	0.000, 0.045	0.000, 0.019
0.000, 0.000	0.000, 0.000	0.000, 0.014	0.000, 0.000	0.000, 0.012
0.000, 0.000	0.000, 0.013	0.000, 0.000	0.000, 0.000	0.000, 0.017
0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.008	0.000, 0.027
0.000, 0.026	2.523, 0.024	0.000, 0.030	0.000, 0.022	0.000, 0.013
0.000, 0.000	14.000, 0.014	0.000, 0.030	0.000, 0.009	0.000, 0.000

0.000, 0.000	0.000, 0.021	0.000, 0.017	0.000, 0.016	0.000, 0.028
0.000, 0.000	0.000, 0.016	0.000, 0.000	0.000, 0.027	0.000, 0.000
0.000, 0.000	0.000, 0.029	0.000, 0.026	0.000, 0.019	0.000, 0.000
0.000, 0.020	0.000, 0.044	0.000, 0.024	0.000, 0.018	0.000, 0.024
0.000, 0.018	0.000, 0.015	0.000, 0.019	0.000, 0.000	0.000, 0.021
0.000, 0.024	0.000, 0.026	0.000, 0.020	0.000, 0.018	0.000, 0.028
0.000, 0.016	0.000, 0.000	0.000, 0.002	0.000, 0.004	0.000, 0.000
0.000, 0.000	0.000, 0.000	0.000, -0.007	0.000, 0.022	0.000, 0.014
0.000, 0.000	0.000, 0.021	0.000, 0.000	0.000, 0.003	0.000, 0.000
0.000, 0.026	0.000, 0.000	1.500, 0.015	0.440, 0.027	0.000, 0.000
0.000, 0.000	0.000, 0.022	0.000, 0.000	0.000, 0.012	0.000, 0.022
0.000, 0.026	0.000, 0.013	0.000, 0.026	0.000, 0.000	0.000, 0.000
0.000, 0.020	0.000, 0.000	0.000, 0.013	0.000, 0.039	0.000, 0.022
0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, -0.008	0.000, -0.018
0.000, 0.000	0.000, 0.027	0.000, 0.000	0.000, 0.025	0.000, 0.000
0.316, 0.019	0.000, 0.002	0.000, 0.019	0.000, 0.009	0.000, 0.000
0.000, 0.019	0.000, 0.021	0.000, 0.020	0.000, 0.014	0.000, 0.024
0.000, 0.024	0.000, 0.000	0.000, 0.000	0.000, 0.028	0.000, 0.003
0.000, 0.000	0.000, 0.000	0.000, 0.019	0.000, 0.021	0.000, 0.013
0.000, 0.022	0.000, 0.000	0.000, 0.018	0.000, 0.022	0.000, 0.029
0.000, 0.000	0.000, 0.022	0.000, 0.022	0.000, 0.021	0.000, 0.000
0.000, 0.015	0.000, 0.021	0.000, 0.027	0.000, 0.024	0.000, 0.020
0.000, 0.020	0.000, 0.007	0.000, 0.024	0.000, 0.002	0.000, 0.010
0.000, 0.008	0.000, 0.000	12.727, -	0.000, 0.017	0.000, 0.000
0.000, 0.000	0.000, 0.031	0.882	0.000, 0.028	0.000, 0.000
0.000, 0.000	0.000, 0.035	0.000, 0.000	0.000, 0.000	0.000, 0.026
0.000, 0.000	0.000, 0.021	0.000, 0.019	0.000, 0.034	0.000, 0.013
0.000, 0.020	0.862, 0.526	0.000, 0.029	0.000, 0.001	0.000, 0.020
0.000, 0.000	0.000, 0.018	0.000, 0.022	0.000, 0.012	0.368, 0.016
0.000, 0.000	0.000, 0.000	0.000, 0.032	0.000, 0.000	0.000, 0.013
0.000, 0.021	0.000, 0.024	0.922, 0.024	0.000, 0.012	0.000, 0.000
0.000, 0.003	0.000, 0.000	0.000, 0.025	0.000, 0.013	0.000, 0.010
0.000, 0.013	0.000, 0.021	0.000, 0.030	0.000, 0.122	0.000, 0.013
0.000, 0.000	0.000, 0.014	0.000, 0.000	0.000, 0.038	0.000, 0.000
0.000, 0.024	0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.019
0.000, 0.026	0.000, 0.000	0.000, 0.015	0.000, 0.000	0.000, 0.000
0.000, 0.013	0.000, 0.006	0.000, 0.000	0.000, 0.015	0.000, 0.035
0.000, 0.021	0.000, 0.000	0.000, 0.000	0.000, 0.018	0.000, 0.019
0.000, 0.027	0.273, 0.020	0.000, 0.027	0.000, 0.011	0.000, 0.030
0.000, 0.000	0.000, 0.026	0.000, 0.026	0.000, 0.000	0.000, 0.019
0.000, 0.000	0.000, 0.022	4.000, 0.012	0.000, 0.027	0.000, 0.000
0.000, 0.019	0.000, 0.000	0.000, 0.000	0.056, 0.021	0.000, 0.000
0.000, 0.022	0.000, 0.027	1.000, 0.015	0.000, 0.000	0.000, 0.024
0.000, 0.000	0.000, 0.000	0.000, 0.018	0.000, 0.000	0.000, 0.019
0.000, 0.026	0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.000
1.527, 0.021	0.210, 0.018	0.304, 0.012	0.000, 0.020	0.000, -0.003

0.000, 0.042	0.000, 0.016	0.000, 0.024	0.000, 0.035	0.000, 0.022
0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.178	0.000, 0.000
0.000, 0.006	0.000, 0.022	0.000, 0.019	0.000, 0.021	0.000, 0.023
0.000, -0.004	0.000, 0.015	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.015	0.000, 0.028	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.029	0.000, 0.069	0.000, 0.020	0.000, 0.023
0.000, -0.002	0.000, 0.000	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.033	0.000, 0.000	0.000, 0.000	0.000, 0.024
0.000, 0.000	0.858, 0.011	0.000, 0.022	0.000, 0.026	0.000, 0.001
0.000, 0.021	0.000, 0.000	0.000, 0.000	0.000, 0.000	1.132, 0.021
0.000, 0.000	0.000, 0.000	0.000, 0.022	0.000, 0.019	0.000, 0.061
0.000, 0.000	0.000, 0.000	0.000, 0.031	0.000, -0.025	0.000, 0.033
0.000, 0.000	0.000, 0.020	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.000	0.000, 0.030	0.000, 0.000	0.000, -0.008
0.000, 0.028	0.000, 0.000	0.000, 0.017	0.000, 0.022	0.000, 0.008
0.000, 0.026	0.000, 0.028	0.000, 0.021	0.000, 0.026	0.000, 0.029
0.000, 0.029	0.000, 0.825	0.000, 0.000	0.000, -0.012	0.000, 0.012
0.000, 0.000	0.000, 0.000	0.000, 0.019	0.000, 0.029	0.000, 0.005
0.000, 0.019	0.000, 0.030	0.000, 0.000	0.472, 0.011	0.000, 0.021
0.000, -0.119	0.000, 0.021	0.334, 0.013	0.000, 0.023	0.000, 0.000
0.000, 0.019	0.000, 0.000	0.000, 0.027	0.000, 0.025	0.000, 0.019
0.000, 0.023	0.000, -0.003	0.000, 0.008	0.000, 0.026	0.000, 0.000
0.000, 0.009	0.000, 0.000	0.000, 0.019	0.000, 0.000	0.000, 0.016
0.000, 0.000	0.000, 1.151	0.000, 0.000	0.000, 0.025	0.000, 0.028
0.000, 0.019	0.000, 0.000	0.000, 0.000	0.000, 0.027	0.000, 0.001
0.000, 0.027	0.000, 0.000	0.000, 0.028	0.000, 0.016	0.000, 0.000
0.000, 0.029	0.000, 0.000	0.000, 0.024	0.000, 0.026	0.000, 0.011
0.000, 0.000	0.000, 0.000	0.000, 0.020	0.000, 0.000	0.000, 0.000
0.000, 0.000	0.000, 0.000	0.000, 0.024	0.000, 0.014	0.000, 0.000
0.000, 0.000	0.000, 0.012	0.000, 0.019	0.000, 0.021	0.000, 0.029
0.000, 0.000	0.000, 0.032	0.000, 0.000	0.000, 0.017	0.000, -0.009
0.000, 0.027	0.000, 0.009	0.000, 0.001	0.000, 0.030	0.000, 0.000
0.000, 0.021	0.000, 0.029	0.000, 0.000	0.000, 0.024	0.000, 0.028
0.000, 0.018	0.000, 0.021	0.000, 0.000	0.000, 0.000	0.000, 0.022
0.000, 0.019	0.000, 0.015	0.000, 0.000	0.000, 0.000	0.000, 0.000
0.000, 0.022	0.000, 0.021	0.000, 0.009	0.000, 0.025	0.000, 0.016
0.187, 0.021	0.000, 0.045	0.000, 0.025	0.000, 0.000	0.000, 0.000
0.000, 0.023	0.000, 0.000	0.000, 0.022	0.000, 0.015	0.000, 0.017
0.000, 0.000	0.000, 0.022	0.000, 0.000	0.000, 0.000	0.000, 0.019
0.000, 0.000	0.000, 0.000	0.000, 0.014	0.000, 0.023	0.000, 0.019
0.000, 0.024	0.000, 0.019	0.000, 0.000	0.000, 0.000	0.000, 0.019
0.000, 0.000	0.000, 0.021	0.000, 0.000	0.000, 0.014	0.000, 0.000
0.000, 0.015	0.000, 0.000	0.000, 0.014	0.000, 0.026	0.000, 0.029
0.000, 0.019	0.000, 0.022	0.000, 0.020	0.000, 0.000	0.000, 0.026
0.000, 0.008	0.000, 0.000	0.000, 0.287	0.000, 0.024	0.000, 0.038
0.000, 0.000	0.000, 0.000	0.000, 0.027	0.000, 0.000	0.000, 0.064

0.000, 0.006	0.000, 0.000	0.000, 0.001	0.000, -0.031
0.000, 0.028	0.000, 0.016	0.000, 0.011	0.000, 0.029
0.000, 0.028	0.000, 0.025	0.000, -0.006	0.000, 0.000
0.000, 0.009	0.000, 0.024	0.000, 0.000	0.000, 0.028
0.000, 0.010	0.233, 0.000	1.000, 0.026	0.000, 0.000
0.000, 0.027	0.000, 0.027	0.000, 0.000	0.000, 0.022
0.000, 0.020	1.000, 0.063	0.000, 0.001	0.000, 0.000
0.000, -0.012	0.000, 0.037	0.000, 0.035	0.000, 0.009
0.000, 0.015	0.000, 0.020	0.000, 0.026	0.000, 0.020
0.000, 0.044	0.000, 0.000	0.000, 0.003	0.270, 0.004
0.000, 0.023	0.000, 0.022	0.000, 0.000	0.000, 0.000
0.000, 0.030	0.000, 0.000	0.000, 0.030	
0.000, 0.019	0.000, 0.000	0.000, 0.000	
0.000, 0.000	0.000, 0.062	0.000, 0.017	
0.000, 0.021	0.000, -0.222	0.000, 0.000	
0.000, 0.000	0.000, 0.028	0.000, 0.000	
0.000, 0.024	0.000, 0.023	0.000, 0.000	
0.000, 0.028	0.000, 0.023	0.000, 0.000	
3.370, 0.022	0.000, 0.020	0.000, 0.019	
0.000, 0.021	0.000, 0.000	0.000, 0.000	
0.000, 0.019	0.000, 0.000	0.000, 0.023	
0.000, -0.017	0.000, 0.018	0.000, 0.017	
0.000, 0.015	0.000, -0.022	0.000, 0.021	
0.000, 0.022	4.000, 0.000	0.000, 0.000	
0.000, 0.016	0.000, 0.019	0.000, -0.028	
0.000, -0.041	0.000, 0.000	0.000, 0.024	
0.000, 0.000	18.000, 0.028	0.000, 0.022	
0.000, 0.000	0.000, 0.027	0.000, 0.018	
0.000, 0.028	0.000, 0.000	0.000, 0.020	
0.000, 0.019	0.000, 0.008	0.000, 0.000	
0.000, 0.000	0.000, 0.000	0.000, 0.027	
0.000, 0.022	0.000, 0.000	0.000, 0.000	
0.000, 0.002	0.000, 0.010	0.000, 0.015	
0.000, 0.006	0.000, 0.000	0.000, 0.000	
0.000, 0.000	0.000, 0.019	0.000, 0.011	
0.000, 0.000	0.000, 0.029	0.000, 0.022	
0.000, 0.022	0.000, 0.020	0.000, -0.004	
0.000, 0.025	0.000, 0.002	0.000, 0.021	
0.000, 0.001	0.000, -0.027	0.000, 0.029	
0.000, 0.000	0.000, 0.008	0.000, 0.027	
0.000, 0.000	0.000, 0.000	0.000, 0.029	
0.000, 0.000	0.000, 0.044	0.000, 0.018	
0.000, 0.022	0.000, 0.004	0.000, 0.000	
0.000, 0.019	0.000, 0.017	0.000, 0.011	
0.000, 0.000	0.000, 0.000	0.000, 0.000	
0.000, 0.019	0.000, 0.014	0.000, 0.000	

Fold 3	0.183, 0.105	0.000, 0.093	0.000, 0.104	0.000, 0.117
	0.000, 0.159	0.000, 0.177	0.000, 0.125	0.000, 0.104
0.000, 0.156	0.000, 0.144	0.000, 0.104	0.000, 0.081	0.000, 0.117
0.000, 0.129	0.000, 0.104	0.000, 0.104	0.000, 0.136	0.000, 0.104
0.000, 0.154	0.000, 0.050	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.115	0.000, 0.104	0.000, 0.104	0.000, 0.087
0.000, 0.104	0.000, 0.159	0.000, 0.018	0.000, 0.104	0.000, 0.104
0.000, 0.125	0.000, 0.142	0.000, 0.162	0.000, 0.104	0.000, 0.135
0.000, 0.117	0.000, 0.104	0.000, 0.117	0.000, 0.104	0.000, -0.014
3.270, 0.117	0.000, 0.104	0.000, -0.346	0.000, 0.102	0.000, 0.137
0.000, 0.155	0.000, -0.008	0.000, 0.024	0.000, 0.104	0.000, 0.104
0.000, 0.123	0.000, 0.068	1.916, 0.134	0.000, 0.118	0.000, 0.104
0.000, 0.091	0.000, 0.104	0.000, 0.078	0.000, 0.104	0.000, 0.104
0.000, 0.155	0.000, 0.130	0.000, 0.108	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.102	0.000, -0.135	0.000, 0.121	0.000, 0.104
0.000, 0.137	0.000, 0.115	0.000, 0.113	0.000, 0.078	0.000, 0.003
0.000, 0.168	0.000, 0.133	0.000, 0.104	0.000, 0.114	0.000, 0.104
0.000, 0.104	0.000, 0.146	0.000, 0.104	0.000, 0.133	0.000, 0.122
0.000, 0.093	0.000, 0.163	0.000, 0.081	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.054	0.000, -0.008	0.000, 0.165
0.000, 5.221	0.000, 0.104	0.000, 0.160	0.000, 0.104	6.000, 0.106
0.014, 0.165	0.000, 0.104	0.000, 0.104	0.000, 0.127	0.000, 0.104
0.000, 0.075	0.000, 0.104	0.000, 0.104	0.000, 0.120	0.000, 0.104
0.000, 0.118	0.000, 0.104	0.000, -0.006	0.000, 0.104	0.000, 0.173
0.000, 0.104	0.000, 0.159	0.000, 0.094	0.000, 0.159	0.000, 0.104
0.000, 0.719	0.000, 0.128	0.000, -0.033	0.110, 0.131	0.000, 0.078
0.000, 0.121	0.000, 0.104	0.000, 0.113	0.000, -0.193	0.000, 0.162
0.000, 0.104	0.000, 0.143	0.000, 0.117	0.000, 0.122	0.000, 0.147
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.078
1.000, 0.133	0.000, 0.078	0.000, 0.104	0.000, 0.102	0.000, 0.104
0.000, -0.616	0.000, 0.104	0.000, 0.104	0.000, 0.158	0.000, 0.104
0.000, 0.104	0.000, 0.130	0.000, 0.104	0.000, 0.128	0.000, 0.159
0.000, 0.104	0.000, 0.136	0.000, 0.104	0.000, 0.155	0.000, 0.119
0.000, 0.117	0.000, 0.149	0.000, 0.104	0.000, 0.104	0.000, 0.126
0.000, 0.093	0.000, 0.099	0.000, 0.153	6.000, 0.139	0.000, -0.188
0.000, 0.104	0.219, 0.133	0.000, 0.100	0.000, 0.133	0.000, 0.134
0.000, 0.100	0.000, 0.104	0.000, 0.146	0.000, -0.578	0.000, 0.113
0.021, 0.108	0.000, 0.144	0.000, 0.081	0.551, 0.168	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.094	1.480, 0.131	0.000, 0.125
0.000, 0.102	0.000, 0.190	0.000, 0.104	16.080, 0.057	0.000, -0.059
0.000, 0.059	0.000, 0.141	0.000, 0.164	0.000, 0.098	0.000, 0.040
0.000, 0.043	0.000, 0.118	1.400, 0.145	0.000, 0.104	0.000, 0.108
1.000, 0.116	0.000, 0.018	0.000, 0.104	0.000, 0.150	0.000, 0.163
0.000, -0.093	0.000, 0.104	0.000, 0.135	0.000, 0.108	0.000, 0.104
0.000, 0.096	0.000, 0.120	0.000, 0.141	0.000, 0.116	0.000, 0.104
0.000, 0.104	0.000, 0.598	0.000, 0.104	0.000, 0.104	0.000, 0.050

0.000, 0.132	0.000, 0.152	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.154	0.000, 0.203	0.000, -0.016	0.000, 0.135
0.000, 0.108	0.000, 0.124	0.000, -0.038	0.000, 0.171	0.000, 0.030
0.000, 0.123	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.133
0.000, 0.104	0.000, 0.124	0.000, 0.104	0.000, 0.163	0.000, 0.181
0.000, -0.045	0.000, 0.859	0.000, 0.104	0.000, 0.104	0.000, 0.099
0.000, 0.104	0.193, 0.146	0.540, 0.134	0.000, 0.104	0.000, 0.104
0.000, 0.149	0.000, 0.104	0.000, 0.061	0.000, 0.136	0.000, -0.124
0.000, 0.104	0.000, 0.104	0.000, 0.165	0.000, 0.104	0.000, 0.097
0.000, 0.132	0.214, 0.133	0.000, 0.104	0.000, -0.029	0.000, 0.104
0.000, 0.141	0.000, 0.133	0.000, 0.104	0.000, 0.135	0.000, 0.104
0.000, 0.104	0.000, 0.116	0.000, 0.104	0.000, 0.141	0.000, 0.045
0.000, 0.133	0.000, 0.104	0.000, 0.149	0.000, 0.128	0.000, 0.104
0.000, 0.119	0.000, 0.124	0.000, 0.104	0.000, 0.156	0.000, 0.093
0.000, 0.150	0.000, 0.104	0.000, 0.154	0.000, 0.136	0.000, 0.104
0.000, 0.133	0.000, 0.168	0.000, 0.118	0.000, 0.128	0.000, 0.104
0.000, 0.133	0.000, -0.049	0.000, 0.104	0.000, 0.135	0.000, 0.040
0.000, 0.104	0.000, 0.123	0.000, 0.104	0.000, 0.113	0.000, 0.104
2.000, 0.117	0.000, 0.104	0.000, 0.104	0.000, 0.112	0.000, 0.121
0.000, 4.734	0.000, 0.143	0.000, 0.163	0.000, 0.136	0.976, 0.101
0.630, 0.165	0.507, 0.141	0.000, 0.071	0.000, 0.146	0.000, 0.104
0.000, 0.153	0.000, 0.142	0.000, 0.104	0.000, 0.104	0.000, 0.130
0.000, 0.104	0.000, 0.003	0.000, 0.104	0.000, 0.104	0.000, 0.104
1.176, 0.044	0.000, 0.104	0.000, 0.135	0.000, 0.112	0.000, 0.104
0.000, 1.625	0.000, 0.134	0.000, 0.173	0.000, 0.078	0.000, 0.104
0.000, 0.104	0.000, 0.091	0.000, 0.124	0.000, 0.104	0.000, 0.150
0.000, 0.151	0.000, 0.039	0.000, 0.117	0.000, 0.055	0.000, 0.073
0.000, 0.104	0.000, 0.022	0.000, 0.117	0.000, 0.096	0.000, 0.159
0.000, 0.149	0.000, 0.095	0.000, 0.060	0.000, 0.104	0.000, 0.117
0.000, 0.142	0.000, 0.104	0.000, 0.136	0.000, 0.140	0.000, 0.163
0.000, 0.104	0.000, 0.104	0.000, 0.117	0.000, 0.104	0.000, 0.104
0.000, 0.132	0.000, 0.155	0.000, 0.098	0.000, 0.244	0.000, 0.141
0.000, -0.327	0.000, 0.104	0.000, 0.174	1.022, 0.148	0.000, 0.104
0.000, 0.068	0.000, 0.119	0.000, 0.104	0.000, 0.104	0.000, 0.140
0.000, 0.134	0.000, 0.145	0.000, 0.104	0.000, 0.104	0.000, 0.023
0.000, 0.160	0.000, 0.117	0.000, 0.104	0.000, 0.104	0.637, 0.163
0.000, 0.104	0.000, 0.105	0.000, 0.104	0.000, 0.109	0.000, 0.135
0.000, 0.134	0.000, 0.079	0.000, -0.040	0.000, 0.140	0.000, 0.104
0.000, -0.018	0.000, 0.118	0.000, 0.137	0.000, -0.027	0.000, 0.121
0.000, 0.152	0.000, 0.104	0.000, 0.116	0.000, 0.104	5.150, 0.095
0.000, 0.104	0.000, 0.148	0.000, 0.152	0.000, 0.159	0.000, 0.142
0.000, 0.122	0.000, 0.104	0.000, -0.099	0.000, 0.104	0.000, -0.116
1.066, 0.126	0.000, 0.069	0.000, 0.104	0.000, 0.104	1.000, 0.156
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.106	0.000, 0.159
0.000, 0.134	0.000, 0.107	0.000, 0.127	0.000, 0.130	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.128	0.000, 0.137

0.000, 0.170	0.000, 0.129	0.000, 0.104	0.000, 0.166	0.000, 0.122
0.000, 0.133	1.000, 0.107	0.000, 0.123	0.000, 0.123	0.000, 0.172
0.000, 0.156	0.000, 0.104	0.000, 0.128	0.000, 0.143	0.000, 0.104
0.000, 0.002	0.000, 0.090	0.000, 0.104	0.000, 0.104	0.000, 0.141
0.000, 0.155	0.000, 0.104	0.000, 0.117	0.000, 0.148	0.000, 0.104
0.000, 0.145	0.000, 0.130	0.000, 0.104	0.000, 0.173	0.000, 0.104
0.000, 0.133	0.000, 0.126	0.000, 0.122	0.000, 0.117	0.000, 0.100
0.000, 0.104	0.000, 0.111	0.000, 0.104	0.000, 0.130	0.000, 0.123
0.000, 0.069	0.000, 0.143	0.000, 0.153	0.000, 0.104	0.000, 0.104
0.000, 0.122	0.000, 0.173	0.000, 0.117	0.000, 0.076	0.000, 0.209
0.000, 0.104	0.000, 0.147	0.000, 0.150	0.000, 0.094	0.000, 0.104
0.000, 0.140	0.000, 0.088	0.000, 0.120	0.000, 0.104	0.000, 0.106
0.000, 0.123	0.000, 0.140	0.000, 0.104	3.000, -0.135	0.000, 0.164
0.000, 0.215	0.000, 0.104	0.000, 0.104	1.000, 0.117	0.000, 0.104
0.000, 0.104	0.000, 0.060	0.000, 0.115	0.000, 0.136	0.000, 0.104
0.000, -0.012	0.000, 0.066	0.000, 0.110	0.000, 0.117	0.000, 0.104
0.000, 0.155	0.000, 0.104	0.000, 0.104	0.000, 0.114	0.000, 0.115
0.000, 0.171	0.000, 0.141	0.000, 0.104	0.000, 0.125	0.000, 0.104
0.000, 0.074	0.000, 0.104	0.000, 0.104	0.000, 0.104	4.000, 0.149
0.000, 0.104	0.000, 0.099	0.000, 0.133	0.000, 0.104	0.000, 0.038
0.000, 0.095	0.000, 0.151	0.000, 0.104	0.000, 0.135	0.000, 0.154
0.000, 0.153	0.000, 0.119	0.000, 0.164	0.000, 0.104	0.000, 0.117
0.000, 0.151	0.000, 0.104	0.000, 0.062	0.000, 0.122	0.000, 0.121
0.000, 0.104	0.000, 0.112	0.000, 0.117	0.000, 0.159	0.000, 0.104
0.000, 0.144	0.000, 0.154	0.000, 0.104	0.000, 0.127	0.000, 0.163
0.000, -0.132	1.294, 0.117	0.000, 0.145	1.856, -0.312	0.000, 0.113
0.000, 0.157	0.000, 0.104	0.000, 0.101	0.000, 0.133	0.000, 0.105
0.000, 0.149	0.000, -0.127	0.000, 0.038	0.000, 0.104	0.000, 0.104
0.000, 0.120	0.000, 0.094	0.000, 0.133	0.000, 0.104	0.000, 0.124
0.000, 0.104	0.000, 0.128	0.000, 0.120	0.000, 0.140	0.000, 0.104
0.000, 0.003	0.000, 0.104	0.000, 0.168	0.000, 0.104	0.000, 0.143
0.000, 0.361	2.000, 0.116	0.000, 0.151	0.000, 0.070	0.000, 0.128
0.000, 0.104	0.000, 0.120	0.000, 0.055	0.000, 0.104	0.000, 0.308
0.000, 0.104	0.000, 0.135	0.000, 0.104	0.000, -0.126	0.000, 0.104
0.000, 0.104	0.000, 0.122	0.000, 0.114	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.259	0.000, 0.127	0.000, 0.104	0.000, 0.112	0.000, 0.133
0.000, 0.104	0.000, 0.145	4.000, 0.111	0.000, 0.136	0.000, 0.090
0.000, 0.121	0.000, 0.107	0.000, 0.133	0.000, 0.104	0.000, 0.104
0.000, 0.149	0.000, 0.144	0.000, 0.096	0.000, 0.160	0.000, 0.104
0.000, 0.121	0.000, 0.104	0.000, 0.137	0.000, 0.117	0.000, 0.137
0.000, 0.141	0.000, 0.099	0.000, 0.104	0.000, 0.113	0.000, 0.111
0.000, 0.057	0.000, 0.181	0.000, 0.089	0.000, 0.104	0.000, 0.133
0.000, 0.159	0.000, 0.116	0.000, 0.104	0.000, 0.085	0.000, -0.027
0.000, 0.069	0.000, 0.120	0.000, 0.104	0.000, 0.120	0.000, 0.104
0.000, -0.049	0.000, 0.149	0.000, 0.104	0.000, 0.104	0.000, 0.144

1.023, 0.138	0.000, 0.166	0.000, 0.149	0.000, 0.125
0.000, 0.141	0.000, 0.060	0.000, -0.008	0.000, 0.125
0.000, 0.163	0.000, 0.065	0.000, 0.150	0.864, 0.123
6.990, 0.159	0.000, 0.191	2.000, 0.065	0.000, 0.133
0.000, 0.120	0.000, 0.165	0.000, 0.104	0.000, 0.125
0.000, 0.417	0.000, 0.104	0.000, 0.104	0.000, 0.076
0.000, 0.102	0.000, 0.104	0.310, 0.114	0.000, 0.104
0.000, 0.104	0.000, 0.130	0.000, 0.104	0.000, 0.163
0.000, 0.110	0.000, -0.103	0.972, 0.069	0.000, 0.104
0.000, 0.038	0.000, 0.104	0.000, 0.119	
0.000, 0.104	0.000, 0.104	0.000, 0.104	
0.000, 1.001	0.000, 0.111	0.070, 0.156	
0.000, 0.104	0.000, 0.127	0.000, 0.104	
0.000, 0.004	0.000, 0.150	0.000, 0.104	
0.000, 0.121	0.000, -0.303	0.000, 0.125	
0.726, 0.126	0.000, 0.150	0.000, 0.112	
0.000, 0.240	0.000, 0.154	0.000, 0.133	
0.000, 0.060	0.000, 0.104	0.000, 0.104	
0.000, 0.156	0.000, 0.006	0.000, 0.103	
0.000, 0.104	0.000, 0.042	0.000, 0.112	
0.000, 0.127	0.000, 0.104	0.000, 0.104	
0.000, 0.104	0.000, 0.120	0.000, 1.229	
0.000, 0.104	0.000, 0.116	0.000, 0.151	
0.000, 0.104	0.000, 0.105	0.000, 0.104	
0.000, 0.122	0.000, 0.163	0.000, 0.167	
1.500, 0.139	0.000, 0.104	0.000, 0.162	
0.000, 0.034	0.000, 0.166	0.000, 0.080	
0.000, 0.146	0.000, 0.104	0.000, 0.043	
0.000, 0.094	0.000, 0.128	0.000, 0.104	
0.000, 0.104	0.000, 0.122	0.000, 0.104	
0.000, 0.104	0.000, 0.104	0.000, 0.160	
0.000, 0.124	0.000, 0.104	0.000, 0.152	
0.000, 0.104	0.000, 0.133	0.000, 0.137	
0.000, 0.143	0.000, 0.133	0.000, 0.115	
0.000, 0.139	0.000, 0.104	0.000, 0.101	
0.000, 0.104	0.000, 0.119	0.000, 0.104	
0.000, 0.083	0.000, 0.104	0.000, 0.129	
0.000, 0.497	0.130, 0.095	0.000, 0.104	
0.000, 0.104	0.000, 0.104	0.000, 0.116	
0.000, 0.104	0.400, 0.103	0.000, 0.133	
0.000, -0.007	0.000, 0.089	0.000, 0.117	
9.475, 0.134	0.000, 0.111	0.000, 0.104	
0.000, 0.155	0.000, 0.144	0.000, 0.003	
0.000, 0.104	0.000, 0.085	0.000, 0.104	
0.000, 0.091	0.000, 0.104	0.000, -0.049	
0.000, 0.072	0.000, -0.076	0.000, 0.104	

Fold 4	0.000, 0.100	0.000, 0.123	0.000, 0.100	0.000, 0.153
	0.000, 0.140	0.000, 0.100	0.000, 0.114	0.000, 0.100
0.000, 0.123	0.000, 0.124	0.000, 0.120	0.000, 0.168	0.000, 0.100
0.000, 0.111	0.000, 0.100	0.070, 0.170	0.000, 0.156	0.000, 0.166
0.000, 0.162	0.000, 0.137	0.000, -0.150	0.000, 0.100	0.000, -0.417
0.000, 0.100	0.000, 0.129	0.000, 0.134	0.000, 0.166	0.000, 0.100
0.000, -0.039	0.000, 0.086	0.000, 0.116	0.000, 0.124	0.000, 0.165
0.000, 0.100	0.000, 0.100	0.000, 0.109	0.027, 0.160	0.000, 0.100
0.000, 0.113	0.000, 0.112	0.000, 0.133	0.000, 0.145	0.000, 0.134
3.000, 0.188	0.000, 0.022	0.000, 0.100	0.000, 0.100	0.000, 0.160
0.000, 0.163	0.000, 0.161	0.000, 0.143	0.000, 0.116	1.000, 0.112
0.000, 0.100	0.000, 0.135	0.000, 0.100	0.825, 0.192	0.000, 0.100
0.183, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.069	0.000, 0.041
0.000, 0.116	0.000, 0.116	0.000, 0.100	0.000, 0.100	0.000, 0.100
0.000, 0.121	0.000, 0.100	0.000, 0.200	0.000, 0.100	0.000, 0.152
0.000, 0.100	0.000, 0.026	0.000, 0.132	13.410, 0.146	0.000, 0.104
0.000, 0.100	0.000, 0.100	0.000, 0.124	0.000, 0.161	0.000, 0.100
0.000, 0.111	0.000, 0.148	0.000, -0.421	0.000, 0.129	0.000, 0.101
0.000, 0.147	0.000, 0.147	0.000, 0.100	0.000, 0.100	0.000, 0.100
0.000, 0.149	0.000, 0.124	0.000, 0.143	1.000, 0.116	0.000, 0.149
0.000, 0.183	0.000, 0.100	0.000, 0.152	2.000, -0.210	10.164, 0.754
0.000, 0.100	0.000, -0.058	0.000, 0.100	0.786, 0.114	0.000, 0.100
0.000, 0.100	0.000, 0.154	0.000, 0.100	0.000, 0.100	0.000, 0.060
0.000, 0.100	0.000, 0.124	0.000, 0.100	0.828, 0.123	0.000, 0.162
0.000, 0.100	0.000, 0.100	0.000, 0.121	0.000, 0.085	0.000, 0.140
2.000, -0.104	0.000, 0.155	0.000, 0.100	0.000, 0.100	0.000, 0.000
0.000, 0.100	0.000, 0.047	0.000, 0.141	0.000, -0.121	0.000, 0.130
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.137	0.000, 0.154
0.000, 0.116	0.602, 0.105	0.000, 0.100	0.000, 0.100	0.000, 0.423
0.000, 0.140	0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.167
0.000, 0.124	0.000, 0.100	0.000, 0.148	0.000, 0.100	0.000, 0.185
0.000, 0.108	0.000, 0.124	0.000, 0.100	0.000, 0.157	0.000, 0.059
0.000, 0.110	0.000, 0.100	0.000, 0.108	0.000, 0.117	0.000, 0.295
0.000, 0.110	0.000, 0.156	0.000, 0.124	2.000, 0.114	0.000, 0.100
0.000, 0.100	0.000, 0.086	1.000, 0.083	0.000, 0.100	0.000, 0.161
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.150	0.000, 0.108
0.000, 0.115	0.000, 0.100	0.000, 0.109	0.000, 0.100	0.000, 0.124
0.000, -0.016	0.000, 0.100	0.000, 0.106	0.210, -0.010	0.000, 0.151
0.000, 0.100	0.000, 0.100	3.666, -0.036	0.000, 0.100	0.000, 0.100
0.000, 0.110	0.000, 0.155	0.000, 0.100	1.000, 0.090	0.000, 0.116
0.000, 0.108	0.000, 0.103	0.000, 0.100	0.000, 0.100	0.000, 0.076
0.000, 0.100	0.000, 0.159	0.000, 0.099	0.000, 0.156	0.000, 0.148
0.000, 0.115	0.000, 0.072	0.000, 0.108	0.000, 0.134	0.000, 0.122
0.000, 0.107	2.000, 0.222	0.000, 0.109	0.000, 0.100	0.000, 0.130
0.000, 0.141	0.000, -0.052	0.000, 0.135	0.000, 0.100	0.000, 0.132
0.000, 0.147	0.000, 0.153	0.000, 0.109	0.509, 0.089	0.000, 0.147

0.000, 0.100	4.000, 0.621	0.000, 0.100	0.000, 0.126	0.000, 0.100
0.000, 0.114	0.000, 0.170	0.000, 0.124	0.000, 0.106	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.112	0.000, 0.100	2.457, 0.213
0.000, 0.138	0.000, 0.133	0.000, 0.139	0.000, 0.033	0.000, 0.100
0.000, 0.134	0.000, 0.110	0.000, 0.100	0.000, 0.144	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.154	0.000, 0.089	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.042	0.000, 0.100	0.000, 0.138
0.000, 0.133	0.000, 0.153	0.000, 0.158	0.000, 0.139	0.000, 0.038
0.000, 0.147	0.000, 0.100	0.000, 0.172	4.000, 0.357	0.000, 0.100
0.000, 0.115	0.000, 0.123	0.000, 0.116	0.000, 0.100	0.000, 0.141
0.000, 0.100	0.000, 0.100	0.000, 0.136	0.000, 0.141	0.000, 0.143
0.000, 0.138	0.000, 0.100	0.000, 0.114	0.090, 0.159	0.000, 0.016
1.000, 0.042	0.000, 0.114	0.000, 0.118	0.000, 0.080	0.000, 0.099
0.000, 0.147	0.000, 0.174	0.000, 0.112	0.000, 0.100	0.000, 0.107
0.365, 0.167	0.000, 0.100	0.000, -0.009	0.000, 0.114	0.000, 0.130
0.000, 0.126	0.000, 0.129	0.000, 0.074	0.000, 0.044	0.000, 0.070
0.000, 0.140	0.000, 0.124	0.000, 0.442	0.000, 0.100	0.000, 0.144
2.215, 0.161	0.000, 0.100	0.000, 0.100	0.000, 0.023	0.000, 0.100
0.000, 0.100	0.000, 0.109	0.000, 0.100	0.000, 0.106	0.000, 0.226
0.000, 0.100	0.000, 0.146	0.000, 0.161	0.000, 0.146	0.000, 0.069
0.000, 0.147	0.000, 0.132	0.000, 0.124	0.000, 0.100	0.000, 0.152
0.000, 0.152	0.000, 0.100	0.000, 0.099	0.000, 0.108	0.000, 0.134
0.000, 0.100	0.000, 0.130	0.000, 0.100	0.000, 0.100	0.000, 0.069
0.000, 0.136	0.000, 0.071	0.000, 0.100	0.000, 0.109	0.474, 0.149
0.000, 0.147	0.000, 0.145	0.000, 0.131	0.000, 0.102	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.108	0.000, 0.100	0.000, 0.100
0.000, 0.114	0.000, 0.114	0.000, 0.152	0.000, 0.136	0.000, 0.100
0.000, 0.168	20.000, 0.172	0.000, 0.100	0.000, 0.124	0.000, 0.109
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.108
0.896, 0.129	0.000, -0.074	0.000, 0.100	0.000, 0.158	0.000, 0.100
0.000, 0.100	0.000, 0.144	0.000, 0.161	0.000, 0.140	0.281, 0.147
0.000, 0.116	0.000, 0.100	0.000, 0.100	0.000, 0.046	0.000, 0.100
0.000, 0.166	0.000, 0.113	0.000, 0.103	0.000, 0.100	0.000, 0.118
0.048, 0.109	0.000, 0.026	0.000, 0.129	0.000, 0.100	1.000, 0.151
0.000, 0.100	0.000, 0.114	0.000, 0.168	0.000, 0.100	0.000, 0.117
0.000, 0.140	2.000, 0.182	0.000, 0.100	22.000, 0.064	0.000, 0.119
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.147
1.000, 0.249	0.000, 0.109	0.000, 0.128	0.000, 0.042	0.000, 0.118
0.000, 0.104	0.000, 0.100	0.000, 0.110	0.000, 0.143	0.000, 0.100
0.000, 0.111	0.000, 0.215	0.000, 0.108	0.000, 0.117	0.000, 0.163
0.000, 0.100	1.086, 0.142	0.000, -0.074	0.000, 0.100	0.150, 0.048
0.000, 0.159	0.000, 0.094	0.000, 0.145	0.000, 0.100	0.000, 0.100
0.000, 0.159	0.000, 0.112	0.000, 0.100	0.000, 0.153	0.000, 0.124
0.000, 0.062	0.000, 0.153	0.000, 0.153	0.000, 0.100	0.000, 0.123
0.000, 0.130	0.000, -0.006	0.000, 0.100	0.000, 0.142	0.000, 0.100
0.000, 0.100	0.000, 0.133	0.000, 0.100	0.000, 0.106	0.000, 0.120

0.000, 0.100	0.000, 0.156	0.110, 0.096	0.000, 0.100	0.000, 0.137
0.000, 0.120	0.000, 0.105	0.000, 0.130	0.000, 0.100	0.000, 0.100
0.000, 0.109	0.000, 0.157	0.000, 0.176	0.000, 0.100	0.000, 0.100
0.000, 0.106	0.000, 0.123	0.000, 0.100	0.000, 0.029	0.000, 0.100
0.000, 0.099	0.000, 0.100	0.000, 0.100	0.000, 0.116	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.114	0.000, 0.113	0.000, 0.134
0.000, 0.141	0.000, 0.043	0.000, 0.100	0.000, -0.041	0.000, 0.100
0.000, 0.109	0.000, 0.100	0.000, 0.021	0.000, 0.100	0.000, 0.124
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.139	0.000, 0.100
0.000, 0.099	0.000, 0.137	0.000, 0.102	0.000, 0.100	0.000, 0.100
0.000, 0.153	0.000, -0.043	0.000, 0.116	0.000, 0.108	0.000, 0.172
0.000, 0.100	1.225, 0.134	0.000, 0.100	0.000, 0.100	0.000, 0.154
0.000, 0.100	0.000, 0.144	0.000, 0.161	0.000, 0.100	0.000, 0.112
0.000, 0.100	0.000, 0.136	0.000, 0.100	0.000, 0.100	0.000, 0.150
0.000, 0.100	0.000, 0.145	0.000, 0.100	0.000, 0.100	0.000, 0.039
0.000, 0.100	0.000, 0.099	0.000, 0.047	0.000, 0.137	0.000, 0.062
0.000, 0.171	0.000, 0.110	0.000, 0.125	0.000, 0.091	0.142, 0.111
0.000, -0.041	0.000, 0.105	0.000, 0.100	0.000, 0.100	0.000, 0.002
0.000, 0.100	0.000, -0.016	0.000, 0.056	0.000, 0.100	0.000, 0.147
0.000, 0.102	5.000, 1.019	0.000, 0.100	0.000, 0.118	0.000, 0.100
0.000, 0.100	0.000, 0.114	0.000, 0.099	0.000, 0.109	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.146	0.000, 0.132	0.000, 0.134
0.000, 0.100	0.000, 0.100	0.000, 0.136	0.000, 0.111	0.000, 0.104
0.000, 0.116	0.000, 0.115	0.000, 0.124	0.014, 0.170	0.000, 0.127
0.000, 0.100	0.130, -0.136	0.000, 0.049	0.000, 0.147	0.000, 0.100
0.000, 0.100	0.000, 0.100	0.000, 0.100	0.000, 0.306	0.000, 0.100
0.000, 0.100	0.000, 0.108	0.000, 0.100	0.000, 0.123	0.000, 0.114
0.000, 0.159	0.000, 0.114	0.000, 0.100	0.000, 0.150	0.000, 0.167
0.000, 0.107	0.350, 0.059	0.000, 0.134	0.000, 0.100	0.000, 0.100
0.000, 0.099	0.000, 0.100	0.000, -0.177	0.000, 0.100	0.000, 0.100
0.000, 0.100	0.000, 0.161	0.000, 0.108	0.000, 0.100	0.000, 0.184
0.000, 0.131	0.000, 0.100	0.000, 0.100	0.000, 0.100	0.333, 0.147
0.000, 0.153	0.000, 0.100	0.000, 0.105	0.000, 0.100	0.000, 0.161
0.000, 0.100	0.000, 0.100	0.000, 0.132	0.000, 0.100	0.000, 0.100
0.000, 0.129	0.000, 0.144	0.000, 0.100	0.000, 0.006	0.175, 0.076
0.000, 0.100	0.000, 0.100	0.000, 0.108	0.000, 0.129	0.000, 0.100
0.000, 0.082	0.000, 0.100	0.000, 0.100	0.000, 0.100	0.456, 0.298
0.400, 0.152	0.000, 0.108	0.000, 0.130	0.000, 0.102	0.000, 0.109
0.000, 0.100	0.000, 0.108	0.000, 0.100	0.000, 0.113	0.000, 0.100
0.000, 0.100	0.000, 0.119	0.000, 0.134	1.116, 0.124	0.000, 0.013
0.000, 0.138	0.000, 0.152	0.000, 0.160	0.000, 0.131	0.000, 0.100
0.000, 0.100	0.000, 0.103	0.000, 0.100	0.000, 0.100	0.000, 0.166
0.000, 0.121	0.000, 0.116	0.000, 0.148	0.000, 0.100	0.000, 0.100
0.000, 0.100	1.000, 0.086	0.000, 0.093	0.000, 0.043	1.210, 0.145
0.000, 0.147	0.000, 0.155	0.000, -0.055	0.514, 0.141	0.000, 0.109
0.000, -0.080	0.000, 0.124	0.000, 0.100	0.000, 0.100	0.000, 0.147

0.000, 0.099	0.000, 0.143	0.000, 0.114	0.000, 0.100
0.000, 0.100	0.000, 0.115	0.000, 0.006	0.000, 0.100
0.000, 0.100	0.000, 0.131	0.000, 0.129	0.000, 0.100
0.000, 0.069	0.000, 0.133	0.000, 0.100	0.000, 0.100
0.000, 0.139	0.000, 0.142	0.000, 0.449	0.000, 0.110
0.000, 0.100	0.000, 0.158	0.000, 0.152	0.000, 0.075
0.000, 0.108	0.000, 0.100	0.000, -0.118	0.000, 0.081
0.000, 0.100	0.000, 0.141	0.000, 0.100	0.000, 1.061
0.000, 0.100	0.000, 0.100	0.000, 0.092	0.000, 0.109
0.000, 0.100	0.000, 0.176	2.000, 0.171	
0.000, 0.100	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.107	0.000, 0.160	
0.000, 0.100	0.000, 0.061	0.000, 0.130	
0.000, 0.122	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	3.000, 0.029	
0.000, 0.100	0.000, 0.064	0.000, 0.100	
0.000, 0.131	0.000, 0.100	0.000, 0.100	
0.000, 0.163	0.000, 0.108	0.000, 0.100	
0.000, 0.112	0.000, 0.189	0.000, 0.100	
0.000, 0.148	0.000, 0.100	0.000, 0.145	
0.000, 0.114	0.000, 0.100	0.000, -0.002	
0.000, 0.274	0.000, 0.100	0.000, -0.272	
0.000, 0.100	0.000, 0.100	0.000, 0.124	
0.000, -0.068	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.100	
0.000, 0.097	0.000, 0.100	0.000, 0.118	
0.000, 0.158	0.000, 0.100	0.000, 0.122	
0.000, 0.100	0.000, 0.047	0.000, 0.139	
0.000, 0.100	0.000, 0.100	0.000, 0.171	
4.000, 0.149	0.000, 0.144	0.000, 0.105	
0.000, 0.100	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, -0.043	
0.100, 0.196	0.000, 0.117	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.108	
4.000, 0.213	0.000, 0.100	0.000, 0.152	
0.000, 0.124	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.119	
0.000, 0.099	0.000, 0.160	0.000, 0.155	
0.000, 0.105	0.000, 0.100	0.000, 0.156	
0.000, 0.133	0.000, 0.100	0.000, 0.141	
0.000, 0.100	0.000, 0.100	0.976, 0.108	
0.000, 0.133	0.000, 0.100	0.000, 0.100	
0.000, 0.100	0.000, 0.100	0.000, 0.113	
0.000, 0.127	0.000, 0.124	7.000, 0.115	

Fold 5	0.000, 0.080	0.000, -1.303	0.000, 0.141	0.000, 0.080
	0.000, 0.113	0.000, 0.082	0.000, -0.235	0.000, 0.124
0.000, 0.139	0.000, 0.114	0.000, 0.080	0.000, 0.140	5.500, 0.115
0.000, 0.080	0.000, 0.157	0.000, 0.080	0.000, 0.080	0.000, 0.110
0.000, 0.117	0.000, 0.166	0.000, 0.012	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.148	0.000, 0.124	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.081	0.000, 0.109	0.000, 0.090	0.000, 0.148
0.000, 0.163	0.000, 0.080	3.000, 0.478	0.000, 0.101	0.000, 0.161
0.000, 0.080	0.000, 0.082	0.000, 0.125	0.000, 0.080	0.000, 0.117
0.000, 0.080	0.000, 0.157	0.000, 0.080	0.000, 0.080	0.000, 0.132
0.000, 0.080	0.000, 0.147	0.000, 0.151	0.000, 0.130	0.000, -0.004
0.000, 0.108	0.000, 0.080	0.000, 0.133	0.000, 0.080	0.000, 0.129
0.000, 0.136	0.000, 0.101	0.000, 0.109	0.000, -0.091	0.000, 0.080
0.000, 0.080	0.000, 0.108	0.000, 0.148	0.000, 0.153	0.000, 0.080
0.000, 0.134	0.000, 0.080	0.000, 0.080	0.000, -0.039	0.000, 0.141
0.000, 0.080	0.000, 0.146	0.000, 0.110	0.000, 0.158	0.000, 0.080
0.000, 0.117	0.000, 0.133	0.000, 0.080	0.000, 0.080	0.000, 0.135
0.000, 0.080	0.333, 0.148	0.000, 0.169	0.000, 0.151	0.000, 0.162
0.000, 0.158	0.000, 0.080	0.000, 0.080	1.000, 0.100	0.000, 0.140
0.000, 0.080	0.000, 0.029	0.000, 0.080	0.000, 0.080	0.000, 0.080
0.000, 0.093	0.000, 0.153	0.000, 0.041	0.000, 0.153	0.000, 0.079
0.000, 0.144	0.000, 0.070	0.000, 0.080	8.047, 5.249	0.000, 0.115
0.000, 0.080	0.000, 0.088	0.000, 0.080	0.000, 0.080	0.000, 0.140
0.000, 0.090	3.220, 0.097	0.000, 0.080	0.000, 0.080	0.000, 0.080
0.456, 0.142	0.000, 0.080	0.000, 0.073	0.000, 0.032	0.000, 0.080
0.000, 0.080	0.000, 0.143	0.000, 0.147	0.000, 0.122	0.996, 0.134
0.000, 0.114	0.000, 0.156	0.000, -0.138	0.000, 0.131	0.000, 0.080
0.000, 0.080	0.000, 0.138	1.000, 0.178	0.000, -0.096	1.000, 0.160
1.000, 0.157	0.000, 0.127	0.000, 0.101	0.000, 0.080	0.000, 0.080
0.000, 0.130	0.000, -0.011	0.000, 0.080	0.000, 0.163	0.000, 0.007
0.000, 0.116	0.000, 0.080	0.000, 0.104	0.000, 0.143	0.000, 0.107
0.000, 0.125	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.148
0.000, 0.080	0.000, 0.071	9.170, 0.148	0.000, 0.150	0.000, 0.080
0.000, 0.158	0.000, 0.080	0.000, 0.080	0.000, 0.139	0.000, 0.164
0.000, 0.114	0.000, 0.121	0.000, 0.094	0.000, 0.080	0.000, 0.080
1.000, 0.110	0.000, 0.080	0.000, -0.016	0.000, 0.133	0.000, 0.080
0.000, 0.129	0.000, 0.162	0.000, 0.080	0.000, 0.080	7.392, -0.063
0.000, 0.098	0.000, -0.111	0.000, 0.108	0.000, 0.143	0.000, 0.082
0.000, 0.152	0.000, 0.151	5.000, 0.138	0.000, -0.133	0.000, 0.080
0.000, 0.098	0.000, 0.080	0.000, 0.133	0.000, -0.004	0.000, 0.695
0.000, 0.117	0.000, 0.006	0.000, 0.157	0.000, 0.155	0.000, 0.124
0.000, 0.080	0.000, 0.141	0.000, 0.149	0.000, 0.138	0.000, 0.080
0.000, 0.148	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.137
0.000, 0.133	0.000, 0.080	0.000, 0.080	0.000, 0.121	0.000, -0.138
0.000, 0.104	0.000, 0.076	0.110, 0.114	0.000, 0.080	0.000, 0.075
0.000, 0.112	0.000, 0.137	0.477, 0.133	0.000, 0.108	0.000, 0.175

0.000, 0.251	0.000, 0.080	0.000, 0.150	0.000, 0.122	0.000, 0.080
0.000, 0.168	0.000, 0.052	0.000, 0.102	0.000, 0.135	0.000, 0.114
0.000, 0.080	0.000, -0.009	0.000, 0.172	0.000, 0.156	0.000, 0.142
0.000, 0.140	0.000, 0.116	0.000, 0.088	0.000, 0.148	0.000, 0.104
0.000, 0.080	0.000, 0.141	0.000, 0.114	0.000, 0.080	0.000, 0.133
0.000, 0.113	0.000, 0.172	0.000, 0.133	0.000, 0.080	0.000, 0.129
0.000, 0.096	0.000, 0.143	0.000, 0.126	0.000, 0.148	0.000, 0.080
0.000, 0.070	0.000, 0.080	0.000, 0.123	0.000, 0.080	0.000, 0.156
0.000, 0.133	0.000, 0.085	0.000, 0.080	0.000, 0.141	0.000, 0.012
0.000, 0.080	0.000, 0.116	0.000, 0.080	0.000, 0.080	0.000, -0.103
0.000, 0.149	0.000, 0.179	0.000, 0.084	0.000, 0.151	0.000, -0.018
0.000, 0.158	0.000, 0.123	0.000, 0.128	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.114	0.000, 0.137	0.000, 0.048
0.000, 0.080	0.000, 0.166	0.000, 0.170	0.000, 0.146	0.000, 0.136
0.000, 0.080	0.000, 0.100	0.000, 0.094	0.000, 0.080	0.044, 0.121
0.000, 0.080	0.000, 0.080	0.000, -0.016	0.000, 0.133	1.630, 0.138
0.000, 0.133	0.000, 0.080	0.000, 0.153	0.000, -0.088	0.000, 0.041
0.000, 0.080	0.000, 0.119	0.000, 0.088	0.000, 0.077	0.000, 0.133
0.000, 0.090	0.000, 0.080	0.000, 0.113	0.000, -0.107	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.128	0.000, 0.124
0.000, 0.177	0.000, 0.080	0.000, 0.080	0.000, 0.113	0.000, 0.141
0.000, 0.089	0.000, 0.159	0.000, 0.158	0.000, 0.080	0.000, 0.161
0.000, 0.080	0.000, 0.095	0.000, 0.116	0.000, 0.057	2.000, -0.068
0.000, 0.080	0.000, 0.126	0.000, 0.080	0.000, 0.075	0.000, 0.119
0.000, 0.102	0.000, 0.131	0.000, 1.323	0.000, 0.141	0.000, 0.144
0.000, 0.133	0.000, 0.155	0.000, 0.143	0.000, 0.080	0.000, 0.091
0.000, 0.125	0.000, 0.094	0.000, 0.080	0.000, 0.118	0.000, 0.123
0.000, 0.068	0.000, 0.062	0.000, 0.128	0.000, 0.080	0.000, 0.111
0.000, 0.109	3.858, 0.138	0.000, 0.080	0.000, 0.124	0.000, 0.133
1.000, 0.146	0.000, 0.067	0.000, 0.058	0.000, 0.053	0.000, 0.080
0.000, 0.092	0.000, 0.080	2.000, 0.137	0.000, 0.156	0.000, 0.127
0.000, 0.111	0.000, 0.124	0.000, 0.108	0.000, 0.080	0.000, 0.080
0.000, 0.150	0.000, 0.149	0.000, 0.080	0.000, 0.080	3.388, -1.551
0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.069	0.000, 0.080
0.000, 0.137	0.000, 0.148	0.000, 0.080	0.000, 0.172	0.000, 0.117
0.000, 0.080	0.000, 0.126	0.000, 0.080	0.000, 0.127	0.000, 0.143
0.000, 0.080	0.000, 0.149	0.000, 0.080	0.000, 0.145	0.000, 0.080
0.000, 0.069	0.000, 0.115	0.000, 0.080	0.000, 0.080	0.000, 0.131
0.000, -0.079	1.984, 0.167	0.000, 0.045	0.000, 0.080	1.000, 0.113
0.000, 0.121	0.000, 0.080	0.000, 0.162	0.000, 0.147	0.000, 0.104
0.000, 0.115	0.000, 0.051	0.000, 0.080	0.000, 0.071	0.000, 0.174
0.000, 0.080	0.000, 0.151	0.000, 0.115	0.000, 0.080	0.000, 0.080
0.000, 0.131	0.000, 0.123	0.000, 0.080	0.000, 0.080	0.000, 0.029
0.000, 0.082	0.000, 0.080	0.000, 0.087	0.000, 0.081	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.116	0.000, 0.080	0.000, 0.145
0.000, 0.170	0.000, 0.113	0.000, 0.080	0.000, 0.080	0.000, 0.080

0.000, 0.130	0.000, 0.127	0.000, 0.080	0.000, 0.080	0.000, 0.094
0.000, 0.100	0.000, 0.143	0.000, 0.152	0.000, 0.127	0.000, 0.165
0.000, 0.157	0.000, 0.162	0.000, 0.113	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.120	0.000, 0.003	0.000, 0.087	0.000, 0.154
0.000, 0.142	0.000, 0.142	0.000, 0.080	0.000, -0.040	0.000, 0.047
0.000, 0.177	0.000, 0.080	0.000, 0.004	0.000, 0.140	0.000, 0.089
0.000, 0.158	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.116
0.000, 0.058	0.000, 0.113	0.000, 0.132	0.000, 0.176	0.000, 0.140
0.000, 0.117	0.110, 0.148	0.000, 0.143	1.590, 0.172	0.000, 0.080
0.000, 0.080	0.000, -0.028	0.000, 0.080	0.000, 0.080	0.000, 0.027
0.000, 0.140	1.000, 0.115	0.000, 0.050	0.000, 0.116	0.000, 0.134
0.000, 0.104	0.000, 0.134	0.000, 1.337	0.000, 0.117	0.000, 0.062
0.000, 0.165	0.000, 0.143	0.000, 0.080	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.152	0.000, 0.107	0.000, 0.146	0.000, 0.106
0.000, 0.126	0.000, 0.167	0.000, 0.080	0.000, 0.133	0.000, 0.109
0.000, 0.127	0.000, 0.080	0.000, 0.022	0.000, 0.137	0.000, 0.143
0.000, 0.118	0.000, 0.080	0.000, 0.098	0.000, 0.080	0.000, 0.114
0.000, 0.122	0.000, 0.146	0.000, 0.140	0.000, 0.126	0.000, 0.143
0.000, 0.113	0.000, 0.133	0.000, 0.116	0.000, 0.137	0.000, 0.080
0.000, 0.080	0.000, 0.163	0.000, 0.218	0.000, -0.004	0.000, 0.140
0.000, 0.156	0.000, 0.114	0.000, 0.065	0.000, 0.080	0.000, 0.118
0.000, 0.156	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.113	0.000, 0.080	0.000, 0.080
0.000, 0.124	0.000, -0.036	0.000, -0.084	7.925, 0.102	0.000, 0.117
0.000, 0.080	0.000, 0.108	0.000, 0.124	0.203, 0.135	0.000, 0.136
0.000, 0.080	0.000, 0.124	0.000, 0.139	0.000, 0.138	0.000, 0.122
0.000, 0.124	0.000, 0.108	0.000, 0.151	0.000, 0.080	0.000, 0.110
0.000, 0.145	0.000, 0.127	0.000, 0.080	0.000, 0.114	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.169	0.000, 0.153	0.000, 0.080
0.000, 0.162	0.000, 0.142	0.000, 0.089	0.000, 0.137	0.000, 0.073
0.000, 0.080	0.000, 0.133	0.000, 0.080	0.000, 0.160	0.000, 0.138
0.000, 0.080	0.000, 0.080	0.000, 0.125	0.032, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.080	0.000, 0.036	0.000, 0.140	0.000, 0.168
1.078, 0.156	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.004
0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.138
0.000, 0.080	0.000, 0.194	0.000, 0.149	0.000, 0.104	0.000, 0.115
0.000, 0.080	1.404, 0.170	0.000, 0.136	0.000, 0.080	0.000, 0.121
0.000, 0.140	0.000, 0.080	0.000, 0.080	0.000, 0.080	0.000, 0.143
0.000, 0.080	0.000, 0.135	0.000, 0.482	0.000, 0.080	0.000, 0.080
0.000, 0.148	0.000, 0.080	0.000, 0.127	0.000, 0.123	0.000, 0.099
0.000, 0.146	0.000, -0.588	5.460, 0.120	0.000, 0.125	0.000, 0.080
0.000, 4.443	0.000, 0.105	0.000, 0.080	0.000, 0.080	0.000, 0.275
0.000, 0.124	0.160, 0.098	1.000, 0.133	0.000, 0.080	0.000, 0.116
0.000, 0.084	0.000, 0.034	0.000, -1.484	0.380, 0.069	0.000, 0.155
0.000, 0.139	0.000, 0.128	0.000, 0.080	0.000, 0.683	0.000, 0.118
0.000, 0.080	0.000, 0.136	0.000, 0.080	0.000, -0.104	0.000, 0.079

0.000, 0.149	0.000, 0.080	0.000, 0.080	0.000, 0.113
0.000, 0.065	0.000, 0.161	0.000, 0.160	0.000, 0.092
0.000, 0.113	0.000, 0.080	0.000, 0.170	0.000, 0.080
0.000, 0.082	0.000, 0.112	0.000, 0.080	0.000, 0.080
0.000, 0.080	0.000, 0.115	0.000, 0.080	0.000, 0.133
0.000, 0.080	0.000, 0.119	0.000, 0.122	0.000, 0.080
0.000, 0.112	0.456, 0.293	0.000, 0.171	3.000, 0.146
0.000, 0.021	0.000, 0.115	0.000, 0.001	0.000, 0.149
0.000, 0.124	0.000, 0.172	0.000, 0.116	3.000, 0.159
0.000, 0.080	0.000, 0.133	0.000, 0.080	
0.000, 0.080	0.000, 0.146	0.000, 0.148	
0.000, -0.039	0.000, -0.141	0.000, 0.108	
0.000, 0.115	0.000, 0.080	0.000, 0.080	
0.000, 0.116	0.000, 0.122	0.000, 0.163	
0.000, 0.080	0.000, 0.152	0.000, 0.150	
0.000, 0.139	0.000, 0.083	0.000, 0.080	
0.000, 0.080	0.000, 0.078	0.000, 0.080	
0.000, 0.080	0.000, 0.080	0.000, 0.133	
0.000, 0.116	0.000, 0.158	0.000, 0.114	
0.000, 0.113	0.000, 0.137	0.000, 0.100	
0.000, 0.152	0.000, 0.123	1.000, 0.114	
0.000, 0.080	0.000, 0.020	0.000, 0.154	
0.000, 0.027	0.000, 0.145	0.000, 0.080	
0.000, 0.080	0.000, 0.118	0.000, -0.104	
0.000, 0.162	0.000, 0.126	0.000, 0.133	
0.000, 0.080	2.455, -0.015	0.000, 0.080	
0.000, 0.080	0.000, 0.113	0.000, 0.080	
0.000, 0.112	0.000, 0.103	0.000, 0.171	
0.000, 0.116	0.000, 0.130	0.000, 0.041	
0.000, 0.080	0.000, 0.116	0.000, 0.104	
0.583, -0.098	0.000, 0.065	0.000, 0.080	
0.130, 0.082	0.000, 0.080	0.000, 0.124	
0.000, 0.080	0.000, 0.133	0.000, 0.080	
0.000, 0.137	0.000, 0.080	0.000, 0.080	
0.000, 0.108	0.000, 0.127	0.000, 0.080	
0.000, 0.133	0.000, 0.065	0.000, 0.131	
0.000, 0.098	0.000, 0.153	0.000, 0.133	
0.000, 0.157	0.000, 0.151	0.000, 0.080	
0.000, 0.162	0.000, 0.080	0.000, 0.080	
0.000, 0.120	0.000, 0.138	0.000, 0.080	
0.000, 0.080	0.000, 0.132	0.000, 0.119	
0.000, 0.128	0.000, 0.113	0.056, 0.113	
0.000, 0.080	0.000, 0.080	0.000, 0.080	
0.000, 0.080	0.000, 0.013	0.000, 0.080	
0.000, 0.137	0.000, -0.020	0.000, 0.080	
0.000, 0.080	0.000, 0.080	0.000, 0.139	

Fold 6	0.000, 0.141	0.000, 0.107	0.000, 0.104	0.000, 0.104
	0.000, 0.104	0.000, 0.134	0.000, 0.114	0.000, 0.148
0.000, 0.105	0.000, 0.109	0.612, 0.117	0.000, 0.142	0.000, 0.104
0.000, 0.179	0.000, 0.148	0.000, 0.108	0.000, 0.107	0.000, 0.557
0.000, 0.104	0.000, 0.130	0.000, 0.109	0.000, 0.124	0.000, 0.104
0.000, 0.104	0.000, 0.110	0.000, 0.104	0.000, 0.009	0.000, 0.104
0.000, 0.110	0.000, 0.118	0.000, 0.119	0.000, -0.042	0.000, 0.116
0.000, 0.104	0.000, 0.137	0.000, 0.104	2.000, 0.118	0.000, 0.024
0.000, 0.176	0.000, 0.101	0.000, 0.141	0.000, 0.114	0.000, 0.168
0.000, 0.104	0.000, 0.125	0.000, 0.008	0.000, 0.105	0.000, 0.143
0.000, 0.036	0.000, 0.150	0.000, 0.071	0.000, 0.104	0.000, 0.136
0.000, 0.104	0.000, 0.120	0.000, 0.087	0.000, 0.104	0.000, 0.074
0.000, 0.104	0.000, 0.104	0.000, -0.019	0.000, 0.127	0.000, -0.554
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.143	0.000, 0.121
8.200, 0.130	0.000, 0.047	0.000, 0.116	0.000, 0.008	0.000, 0.135
0.000, 0.194	0.000, 0.105	0.000, 0.124	0.000, -0.021	0.000, 0.104
0.000, 0.104	0.000, 0.118	0.000, -0.381	0.000, 0.110	0.000, 0.096
0.000, 0.121	0.000, 0.121	0.000, -0.042	0.000, 0.104	2.160, 0.159
0.000, 0.149	0.000, 0.104	0.000, 0.108	0.000, 0.113	0.000, 0.142
0.000, 0.159	0.000, 0.104	0.130, 0.140	0.000, 0.104	0.000, 0.104
1.036, 0.137	0.000, 0.131	0.064, 0.127	0.000, 0.104	0.000, 0.133
0.000, 0.123	0.000, 0.151	0.000, 0.104	0.000, 0.102	0.000, 0.104
0.000, 0.134	0.023, 0.109	0.000, 0.118	0.680, 0.126	0.000, 0.104
0.000, 0.038	0.000, 0.132	0.000, 0.141	0.000, 0.104	0.000, 0.105
0.000, 0.104	0.973, 0.027	0.000, 0.104	0.000, 0.104	0.000, 0.128
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.100
0.000, 0.147	0.000, 0.113	0.000, -0.101	0.000, 0.104	0.000, 0.116
0.000, 0.136	0.000, 0.083	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.181	0.000, 0.104	0.000, 0.078
0.000, 0.110	0.000, 0.146	0.000, 0.104	0.000, 0.122	0.000, 6.001
0.000, 0.104	0.000, -0.038	0.000, 0.131	0.000, 0.079	0.000, 0.137
1.000, 0.054	0.000, 0.112	0.000, 0.104	0.000, 0.119	0.000, 0.126
0.000, 0.130	0.000, 0.106	0.000, 0.126	0.000, 0.115	0.000, 0.113
0.000, 0.143	0.000, 0.135	0.000, 0.118	0.000, 0.125	0.000, 0.001
0.000, -0.059	0.000, 0.146	0.000, 0.102	0.000, 0.104	0.000, 0.104
0.000, 0.108	0.000, 0.143	0.000, 0.104	4.000, 0.163	0.000, 0.181
0.000, 0.115	0.000, 0.118	0.000, -0.014	0.000, 0.116	0.000, 0.103
0.000, 0.104	0.000, 0.059	0.000, 0.107	0.000, 2.239	0.000, 0.114
0.000, 0.124	0.000, 0.104	0.000, 0.110	0.000, 0.104	0.000, 0.073
0.000, 0.083	0.000, 0.144	0.000, 0.109	0.000, 0.156	0.000, 0.115
0.000, 0.181	0.000, 0.198	0.000, 0.121	0.000, 0.113	0.000, 0.141
0.000, 0.113	0.000, 0.133	0.610, 0.149	0.000, 0.095	0.000, -0.008
0.000, 0.133	0.364, 0.133	0.000, 0.109	0.000, 0.120	0.000, 0.100
3.000, 0.151	0.000, 0.083	0.000, 0.104	0.000, 0.463	0.000, 0.104
0.000, 0.116	2.000, 0.117	0.000, 0.120	0.000, 0.102	0.000, 0.099
0.000, 0.147	0.000, 0.133	0.000, 0.148	0.000, 0.095	0.000, 0.104

0.000, 0.154	0.000, 0.094	0.000, 0.082	0.000, 0.130	0.000, 0.137
0.000, 0.122	0.000, 0.117	0.000, 0.104	0.000, 0.130	0.000, 0.135
0.000, 0.145	0.000, 0.104	0.000, 0.133	0.000, 0.127	2.442, 0.098
0.000, 0.129	0.000, 0.113	0.000, 0.104	0.000, 0.142	0.000, 0.167
0.000, 0.104	0.500, 0.181	0.000, 0.121	0.000, 0.116	0.000, 0.149
0.000, -0.008	0.000, 0.137	0.000, 0.104	0.000, 0.130	0.000, 0.130
0.248, 0.047	0.000, 0.104	18.240, 0.149	0.000, 0.181	0.000, 0.104
0.000, 0.127	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.025	1.000, -0.076	0.000, 0.104	0.000, 0.104	1.000, 0.141
0.000, 0.104	0.000, -0.008	0.000, 0.127	0.000, 0.104	0.000, 0.105
0.000, 0.104	0.000, 0.104	0.000, 0.126	0.000, 0.104	0.000, 0.134
0.000, 0.104	0.000, 0.112	0.000, 0.101	0.000, 0.441	0.000, 0.104
0.000, 0.071	0.000, 0.065	0.000, 0.104	0.000, 0.104	0.000, 0.106
0.000, 0.104	0.000, 0.090	0.000, 0.130	0.150, 0.122	0.000, 0.108
0.000, 0.085	0.000, -0.140	0.000, 0.104	0.000, 0.104	0.000, 0.116
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.104
0.000, 0.133	0.000, 0.001	0.000, 0.067	0.000, 0.082	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.146	0.000, 0.109	0.000, -0.002
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.000, 0.086	0.000, 0.104
0.000, 0.672	0.000, 0.104	0.000, 0.104	0.000, 0.119	0.000, 0.104
0.000, 0.181	0.000, 0.371	0.000, 0.058	0.000, 0.140	0.000, 0.110
0.000, 0.112	0.000, 0.104	0.000, 0.127	0.000, 0.153	0.000, 0.079
0.000, 0.104	0.760, 0.112	0.000, 0.104	0.000, 0.154	0.000, 0.134
0.000, 0.116	0.000, 0.104	0.000, 0.000	0.000, 0.181	0.000, 0.112
0.000, 0.147	0.000, 0.110	0.000, 0.141	0.000, 0.106	0.000, 0.105
0.000, 0.106	0.000, 0.104	0.000, 0.101	0.000, 0.139	0.000, 0.141
0.000, 0.104	0.000, 0.124	0.000, 0.118	0.000, 0.090	0.000, 0.112
0.000, 0.167	0.000, 0.110	0.000, 0.148	0.000, 0.132	0.000, 0.147
0.000, 0.152	0.000, 0.120	0.000, 0.085	0.000, 0.131	1.118, 0.091
0.000, 0.116	0.000, 0.104	0.000, 0.104	0.000, 0.115	0.000, 0.086
0.000, 0.104	0.000, 0.181	0.000, 0.104	0.337, 0.121	0.000, 0.114
0.000, 0.113	0.000, 0.139	0.000, 0.104	0.000, 0.150	0.000, 0.104
0.000, 0.104	0.000, 0.116	0.000, 0.132	0.000, 0.131	0.000, 0.127
2.000, 0.031	0.000, 0.131	0.000, 0.030	0.000, 0.107	0.000, 0.104
0.000, 0.137	0.000, 0.118	0.000, -0.131	2.000, 0.250	0.000, 0.085
0.000, 0.104	0.000, 0.104	0.000, 0.104	0.640, 0.115	0.000, 0.104
0.000, -0.015	0.000, 0.104	0.000, 0.112	0.000, 0.181	0.000, 0.104
0.000, 0.181	0.000, 0.127	0.000, 0.104	0.000, 0.147	0.000, 0.116
0.000, 0.106	0.000, 0.109	0.000, 0.104	0.000, 0.104	0.000, 0.080
0.000, 0.113	0.000, -0.133	0.000, 0.009	0.000, 0.108	0.000, 0.104
0.000, 0.105	0.000, 0.104	0.000, 0.135	0.000, 0.121	0.000, 0.104
0.000, 0.106	0.000, 0.169	0.000, 0.104	0.000, 0.104	0.000, 0.115
0.000, 0.104	0.000, 0.104	0.000, 0.115	0.000, 0.097	0.000, 0.145
0.000, 0.109	0.000, 0.099	0.000, 0.107	0.000, 0.104	0.000, 0.104
0.000, 0.136	0.000, 0.119	0.000, 0.127	0.000, 0.161	0.000, 0.104
0.000, 0.104	0.000, 0.120	0.000, 0.104	0.000, 0.104	0.000, 0.104

0.000, 0.104	0.000, 0.897	0.000, 0.104	0.000, 0.104	2.010, 0.146
0.000, 0.147	3.000, 0.148	0.000, 0.104	0.000, 0.057	0.000, 0.104
0.000, 0.283	0.000, 0.115	0.000, 0.104	0.000, 0.104	0.000, 0.111
0.000, 0.104	0.000, 0.085	0.000, 0.104	0.000, 0.104	0.000, 0.142
0.000, 0.101	0.000, 0.164	0.000, 0.003	0.000, 0.181	0.000, 0.104
0.000, 0.136	0.000, 0.128	0.000, 0.104	0.000, 0.104	0.668, 0.010
4.000, 0.231	0.000, 0.117	0.000, 0.077	0.000, -0.058	0.000, 0.121
0.375, 0.181	0.465, 0.096	0.000, 0.160	0.567, 0.096	0.000, 0.152
0.000, 0.087	0.000, 0.104	0.000, 0.135	0.000, 0.104	0.000, 0.114
0.000, 0.104	0.000, 0.181	5.004, 0.128	0.000, 0.104	0.000, 0.107
0.000, 0.116	0.000, 0.127	0.000, 0.104	0.000, 0.181	0.000, 0.058
0.000, 0.125	0.000, 0.104	0.000, 0.139	0.050, 0.181	0.000, 0.104
0.000, 0.104	0.000, 0.117	0.000, 0.104	0.000, 0.077	0.000, 0.159
0.000, 0.104	0.500, 0.183	0.000, 0.076	0.000, 0.181	0.000, -0.007
0.000, 0.104	0.000, 0.104	0.000, 0.133	0.000, 0.104	0.000, 0.104
0.000, 0.104	0.000, 0.109	0.000, 0.038	0.000, 0.110	0.000, 0.104
0.000, 0.104	0.000, 0.105	0.000, 0.163	0.000, 0.078	0.000, 0.141
0.000, 0.120	0.000, 0.104	0.000, -0.050	0.000, 0.105	0.000, 0.104
0.000, 0.104	0.000, 0.136	0.000, 0.110	0.295, 0.110	0.000, 0.137
0.000, 0.104	0.000, 0.072	0.000, 0.104	0.000, 0.108	0.000, 0.136
0.000, 0.104	4.000, 0.375	0.000, 0.138	0.000, 0.104	0.000, 0.147
0.000, 0.131	0.000, 0.085	0.000, 0.110	0.000, 0.104	0.545, 0.167
0.000, 0.104	0.000, 0.126	0.000, 0.104	0.000, 0.136	0.000, 0.104
0.000, 0.104	0.000, 0.151	0.000, 0.100	0.000, 0.143	1.203, 0.147
0.000, 0.727	0.000, 0.104	0.000, 0.135	0.000, 0.144	0.000, 0.104
0.000, 0.160	0.000, 0.104	0.000, 0.137	0.000, 0.105	0.000, 0.088
0.000, 0.143	0.000, 0.134	0.000, 0.104	0.000, 0.181	0.000, 0.104
0.000, 0.087	0.000, 0.114	0.000, 0.102	0.000, 0.123	0.000, 0.104
0.000, 0.104	0.000, 0.124	0.000, 0.104	0.000, 0.104	0.000, 0.127
0.000, -0.038	0.000, 0.087	0.000, -0.031	0.000, 0.128	0.000, -0.104
0.000, 0.110	0.000, 0.095	0.000, 0.151	0.000, 0.104	0.000, 0.114
0.000, 0.115	0.000, 0.107	0.000, 0.113	0.000, 0.096	0.000, 0.104
0.000, 0.104	1.076, 0.113	0.000, 0.123	0.000, 0.144	0.000, 0.104
0.000, 0.104	0.000, 0.104	0.000, 0.132	0.000, 0.146	0.000, -0.013
0.000, 0.146	0.000, 0.113	1.448, 0.144	0.000, 0.140	0.000, 0.105
0.000, 0.104	0.000, 0.143	0.100, 0.084	0.000, 0.131	0.000, -0.055
0.000, 0.079	0.000, 0.104	0.000, 0.124	0.000, 0.090	0.000, 0.104
0.000, 0.016	0.000, 0.063	0.934, 0.149	0.000, 0.092	0.000, 0.119
0.000, 0.104	0.000, 0.141	0.000, 0.104	0.000, 0.104	0.000, 0.127
0.000, 0.099	2.000, 0.082	0.000, 0.104	0.000, 0.104	0.000, 0.070
0.000, 0.163	0.000, 0.104	0.000, 0.140	0.000, 0.122	0.000, -0.060
0.000, 0.129	0.000, 0.104	0.000, 0.104	0.000, 0.105	0.000, 0.086
0.000, 0.127	0.000, 0.104	0.000, 0.181	0.000, 0.104	0.000, 0.409
0.000, 0.104	0.000, 0.051	0.000, 0.116	0.000, 0.111	0.000, 0.129
0.000, 0.117	0.000, 0.122	0.000, 0.104	0.000, 0.104	0.000, 0.171
0.000, 0.109	2.080, 0.121	0.000, 0.072	0.000, 0.181	0.000, 0.104

0.000, 0.116	0.000, 0.163	0.000, 0.088	0.000, 0.104
0.000, 0.108	0.000, 0.154	0.000, 0.104	0.000, 0.106
0.000, 0.186	0.000, 0.104	0.000, 0.181	0.000, 0.140
0.000, 0.105	0.000, 0.104	0.000, 0.111	0.000, 0.104
0.000, 0.141	0.000, 0.104	10.000, 4.875	0.000, 0.104
0.000, 0.145	0.000, 0.112	0.000, 0.104	0.000, 0.154
0.000, 0.113	1.000, 0.140	0.000, 0.080	0.000, 0.146
0.000, 0.104	0.000, 0.104	0.000, 0.155	0.000, 0.120
0.000, 0.160	0.000, 0.112	0.000, 0.065	1.000, 0.152
0.000, 0.104	0.000, 0.131	0.000, 0.078	
0.750, 0.108	0.000, 0.018	0.000, 0.104	
0.000, 0.132	0.036, 0.118	0.000, 0.119	
0.070, 0.125	0.000, 0.104	0.456, 0.156	
0.000, 0.104	0.000, 0.109	0.000, 0.181	
0.000, 0.136	0.000, 0.104	0.000, 0.104	
0.000, 0.096	0.000, 0.134	0.000, 0.104	
0.000, 0.127	0.000, 0.104	0.000, 0.104	
0.000, 0.117	0.000, 0.116	0.000, 0.104	
0.000, 0.108	0.000, 0.104	0.000, 0.129	
0.000, 0.122	0.000, 0.181	0.000, 0.027	
0.000, 0.116	0.000, 0.121	0.000, 0.181	
0.000, 0.087	0.000, 0.087	0.000, 0.166	
0.000, 0.110	0.000, -0.070	6.000, 0.048	
0.000, 0.104	0.000, 0.147	0.000, 0.108	
0.000, 0.104	0.000, 0.104	0.000, 0.108	
0.000, 0.104	0.000, 0.104	0.000, 0.181	
0.000, 0.136	0.000, 0.104	0.000, 0.111	
0.000, 0.104	0.000, 0.181	0.000, 0.104	
0.000, 0.104	0.000, 0.014	0.000, 0.062	
0.523, 0.164	0.000, 0.104	0.000, 0.104	
0.000, 0.104	0.000, 0.152	0.000, 0.091	
0.000, 0.105	0.000, 0.086	0.000, -0.032	
0.000, 0.111	0.000, 0.104	0.000, 0.113	
0.000, 0.138	0.000, 0.104	0.000, 0.078	
0.000, 0.161	0.000, 0.131	0.000, 0.111	
0.000, 0.104	0.000, 0.171	0.000, 0.089	
0.000, 0.118	0.000, 0.157	0.000, 0.173	
0.000, 0.098	0.000, 0.104	0.000, 0.133	
1.000, 0.093	0.000, 0.075	0.000, 0.104	
0.000, 0.108	0.000, 0.094	0.000, 0.114	
0.000, 0.073	0.000, 0.112	0.000, 0.059	
0.000, 0.147	0.000, 0.104	0.000, 0.155	
0.000, 0.039	0.000, 0.104	0.000, 0.138	
0.000, 0.104	0.000, 0.110	0.000, 0.123	
0.000, 0.104	0.000, 0.112	0.000, 0.115	
0.000, 0.107	0.000, 0.128	0.000, 0.104	

Fold 7	0.000, 0.157	0.000, 0.122	0.000, 0.138	0.000, 0.199
	0.000, 0.421	0.000, 0.122	0.000, 0.131	0.000, 0.130
0.000, 0.122	0.000, 0.122	0.000, 0.134	0.000, 0.145	0.000, 0.030
0.000, 0.107	0.000, 0.164	0.000, 0.122	0.000, 0.122	0.000, 0.107
0.000, 0.122	2.048, 0.162	0.000, 0.131	0.000, 0.139	0.000, 0.083
0.000, 0.142	0.000, 0.051	0.000, 0.131	0.000, 0.122	0.000, 0.141
0.000, 0.149	0.000, 0.028	0.000, 0.122	0.000, 0.031	0.000, 0.139
0.000, 0.176	0.000, 0.139	0.000, -0.465	0.000, 0.050	0.000, 0.041
0.000, 0.095	0.000, 0.049	0.000, 0.122	0.000, 0.139	0.000, 0.143
0.000, 0.167	0.000, 0.136	0.180, 0.150	0.000, 0.437	0.000, 0.122
0.000, 0.122	0.000, 0.147	2.000, 0.156	0.000, 0.099	0.000, 0.122
0.000, 0.185	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.676, -0.169	0.000, 0.122	0.000, 0.175	0.000, 0.131	0.000, 0.176
0.000, 0.139	0.000, 0.129	0.000, -0.128	0.000, 0.132	0.000, 0.135
0.000, 0.167	0.000, 0.122	2.000, 0.143	0.000, 0.122	0.000, 0.170
0.000, 0.122	0.000, 0.164	0.000, 0.122	0.210, 0.098	0.000, 0.065
3.000, 0.167	0.000, 0.159	0.000, 0.122	0.000, 0.159	0.000, -0.023
0.000, 0.124	0.000, 0.146	0.000, 0.122	0.000, 0.102	0.000, 0.181
0.000, 0.080	0.000, 0.104	0.000, 0.122	0.000, 0.134	0.000, 0.122
0.000, 0.062	0.000, 0.101	0.000, 0.122	0.000, 0.152	0.000, 0.039
2.000, 0.119	0.000, 0.099	0.000, 0.164	0.000, 0.085	0.000, 0.166
0.000, 0.119	0.000, 0.111	0.000, 0.122	0.000, 0.150	0.000, 0.166
0.000, 0.122	0.400, 0.177	0.000, -0.673	0.000, 0.147	0.000, 0.030
0.000, 0.122	0.000, 0.133	0.000, 0.122	0.000, 0.180	0.000, 0.132
0.000, 0.122	0.000, 0.116	0.000, 0.123	0.000, 0.122	0.000, -0.800
0.000, 0.122	0.000, 0.122	0.000, 0.136	0.000, 0.122	0.000, 0.104
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.062	0.000, 0.122
0.000, 0.133	0.000, -0.040	0.000, 0.035	0.000, 0.122	0.000, 0.125
0.000, 0.122	0.000, 0.142	0.000, 0.131	0.000, 0.122	0.000, 0.147
0.000, -0.140	0.000, 0.122	0.000, 0.129	0.000, 0.122	0.000, 0.144
0.000, -0.032	0.000, 0.124	0.000, 0.122	0.000, 0.122	0.000, 0.122
2.448, 0.167	0.000, 0.059	0.000, 0.122	0.000, 0.140	0.000, 0.152
0.000, 0.040	0.000, 0.149	0.000, 0.128	0.000, 0.165	0.000, 0.099
0.000, 0.481	0.000, 0.099	0.000, 0.182	0.000, 0.144	0.000, 0.104
0.000, 0.043	0.000, -0.020	0.000, 0.171	0.000, 0.171	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.059	0.000, 0.127	0.000, 0.122
0.000, 0.174	0.000, 0.149	0.000, 0.137	0.000, 0.108	0.000, 0.122
0.000, 0.148	0.000, 0.122	0.000, 0.201	0.000, 0.122	0.000, 0.122
0.000, 0.091	0.000, 0.109	0.000, 0.134	2.200, 0.143	0.000, 0.123
0.000, 0.171	0.000, 0.144	0.000, 0.168	0.000, 0.122	0.000, 0.161
0.000, 0.122	0.000, 0.122	0.000, 0.095	0.000, 0.065	0.000, 0.167
0.000, 0.147	0.000, 0.122	0.000, 0.129	0.000, 0.068	0.000, 0.134
0.000, 0.191	0.000, 0.122	0.000, 0.099	0.000, 0.146	0.000, 0.122
0.000, 0.146	0.000, 0.122	0.000, 0.199	0.000, 0.172	0.000, 0.176
0.000, 0.134	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.131
0.000, 0.122	0.000, 0.143	0.162, 0.168	0.000, 0.145	0.000, 0.122

0.000, 0.122	0.000, 0.129	0.000, 0.121	0.000, 0.158	0.130, 0.155
0.000, 0.196	0.000, 0.099	0.000, 0.157	0.000, 0.169	0.000, 0.122
5.000, 0.151	0.000, 0.138	0.000, 0.146	0.000, 0.118	0.000, 0.139
0.000, 0.122	0.000, 0.122	0.000, 0.134	0.000, 0.155	0.000, 0.122
0.000, 0.159	0.000, 0.122	0.000, 0.600	0.000, 0.090	0.000, 0.148
0.000, 0.161	0.000, 0.099	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, -0.001	0.000, 0.120	0.000, 0.149
1.000, 0.144	0.000, 0.122	0.000, 0.129	0.000, 0.163	0.000, 0.122
0.000, 0.157	0.000, 0.159	0.000, 0.162	0.000, 0.101	0.000, 0.155
0.000, 0.139	0.000, 0.127	0.000, 0.075	0.000, 0.147	0.000, 0.174
0.000, 1.520	0.000, 0.116	0.000, 0.122	0.000, 0.122	0.000, 0.040
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.114	0.000, -0.138
0.000, 0.175	0.000, 0.185	0.000, 0.122	0.000, 0.077	0.000, 0.143
0.000, 0.122	0.000, 0.062	0.000, 0.037	0.000, 0.128	0.000, 0.139
0.000, 0.122	0.000, 0.122	0.000, 0.147	0.000, 0.156	0.000, 0.122
0.000, 0.122	0.000, 0.128	0.000, 0.155	0.000, 0.133	0.040, 0.099
0.000, 0.164	0.000, 0.122	0.000, 0.123	0.000, 0.122	0.000, 0.122
0.000, 0.159	1.000, 0.151	0.000, -0.174	0.000, 0.160	0.000, 0.122
0.000, 0.172	0.000, 0.122	0.000, 0.122	0.000, 0.145	0.000, 0.054
0.433, 0.098	0.000, 0.070	0.116, 0.154	0.000, 0.170	0.000, 0.169
0.000, 0.151	0.000, 0.122	0.000, 0.121	0.000, 0.122	0.000, 0.163
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.149	0.000, 0.128	0.000, 0.114	0.000, 0.135	0.000, 0.122
0.000, 0.142	0.000, 0.156	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.134	0.000, 0.122	0.000, 0.140	0.000, 0.122	0.000, 0.166
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.091	0.000, 0.106
0.000, 0.150	0.000, 0.053	0.000, 0.159	0.000, 0.122	0.000, 0.135
0.000, 0.122	0.000, 0.072	0.000, 0.088	0.000, 0.160	0.000, 0.014
0.000, 0.185	0.000, 0.051	0.000, 0.138	0.000, 0.163	0.000, 0.151
0.000, 0.122	0.000, 0.085	0.303, 0.121	0.000, 0.135	12.000, 0.165
0.000, 0.160	0.000, 0.127	0.000, 0.122	0.000, 0.165	0.000, 0.122
0.000, 0.084	0.000, 0.146	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.167	0.000, 0.122	0.000, 0.167	0.000, 0.098	0.000, 0.122
0.000, 0.135	0.000, 0.174	0.000, 0.053	0.000, 0.183	0.000, 0.035
0.000, 0.122	0.000, 0.165	0.000, 0.127	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.170	0.000, 0.165	0.000, 0.098	0.000, 0.145
0.000, 0.067	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.167
0.000, 0.122	0.000, 0.185	0.000, 0.122	0.000, 0.197	0.000, 0.052
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.137	0.000, 0.122
0.000, 0.109	0.000, 0.110	0.000, 0.104	0.000, 0.164	0.000, 0.086
1.000, 0.147	0.000, 0.122	0.000, 0.138	0.000, 0.151	0.000, 0.152
0.000, 0.159	0.000, -0.210	0.530, 0.150	0.000, 0.128	0.000, 0.122
0.000, 0.122	0.000, 0.143	0.000, 0.160	0.000, 0.083	0.000, 0.074
0.465, 0.092	0.000, 0.132	0.000, 0.149	0.000, 0.122	0.000, 0.144
0.000, 0.122	0.000, 0.136	0.000, 0.110	0.000, 0.168	0.000, 0.122
0.000, 0.131	0.000, 0.154	2.568, -0.064	0.000, 0.122	0.000, 0.122

0.000, 0.148	0.000, 0.122	0.000, 0.129	0.000, 0.122	0.000, 0.127
0.000, 0.122	0.000, 0.170	0.000, 0.143	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.165	0.000, 0.136	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.132	0.000, 0.088	0.000, 0.085
0.000, 0.117	0.000, 0.122	0.000, 0.139	0.000, 0.206	0.000, 0.078
0.000, 0.135	0.000, 0.028	0.000, 0.107	0.000, 0.122	0.000, 0.145
0.000, 0.099	0.000, 0.150	0.000, 0.142	0.000, 0.109	0.000, 0.150
0.000, 0.030	0.000, 0.112	0.000, 0.099	0.000, 0.096	0.000, 0.122
0.000, 0.122	0.000, 0.087	0.000, 0.122	0.000, 0.160	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.179	0.000, 0.122	0.000, 0.099
0.000, 0.122	0.000, -0.292	0.000, 0.159	0.000, 0.122	0.000, 0.141
0.000, 0.157	0.000, 0.121	0.000, 0.126	0.000, 0.137	0.000, 0.122
0.000, 0.150	0.000, 0.160	0.000, 0.120	0.000, 0.122	0.000, 0.048
0.000, 0.160	0.000, 0.099	0.000, 0.041	0.000, 0.049	0.000, 0.122
0.603, 0.175	0.000, 0.126	0.000, 0.008	0.000, 0.122	0.000, 0.131
0.000, -0.030	0.000, 0.133	0.000, 0.122	0.000, 0.107	0.000, 0.121
0.000, 0.122	0.000, 0.177	0.000, 0.016	0.000, 0.139	1.000, 0.574
0.000, 0.151	4.000, 0.110	0.000, 0.122	0.000, 0.135	0.000, 0.120
0.000, 0.122	0.000, 0.122	0.328, 0.133	0.000, 0.089	0.000, 0.122
0.000, 0.157	0.000, 0.127	0.000, 0.122	0.000, 0.157	0.000, 0.122
0.000, 0.165	0.000, 0.122	0.000, 0.152	0.000, 0.122	0.000, 0.083
0.000, 0.178	0.000, 0.143	0.000, 0.152	0.000, 0.122	2.000, 0.148
0.000, 0.142	0.000, 0.122	0.000, 0.122	0.000, 0.087	0.000, 0.161
0.000, 0.122	0.000, 0.078	0.000, 0.156	0.000, 0.122	0.000, 0.122
0.000, 0.138	0.000, 0.136	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.122	0.000, 0.131	0.000, 0.143	0.000, 0.160	3.122, 0.171
0.000, 0.122	0.000, 0.099	0.000, 0.122	0.000, 0.108	0.000, 0.122
0.000, 0.061	0.000, 0.163	0.000, -0.018	0.000, 0.092	0.000, 0.147
0.000, 0.169	0.000, 0.122	0.000, 0.120	0.000, 0.122	0.000, 0.122
0.000, 0.167	0.000, 0.122	0.000, 0.122	0.000, 0.166	0.000, 0.123
0.000, 0.122	0.000, 0.132	0.000, 0.131	0.000, 0.151	0.000, 0.147
0.000, 0.134	0.000, 0.168	0.000, 0.155	0.000, 0.172	0.000, 0.051
0.000, 0.132	0.000, 0.152	0.000, 0.145	0.000, 0.164	0.000, 0.133
0.000, 0.122	0.000, 0.122	0.000, 0.057	0.000, 0.122	0.000, 0.122
0.000, 0.102	0.000, 0.130	0.000, 0.148	0.000, 0.118	0.000, 0.262
0.000, 0.122	0.000, 0.135	0.000, 0.077	0.000, 0.167	0.000, 0.147
0.459, 0.058	0.000, 0.064	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.137	0.000, 0.104	0.000, 0.167	0.000, 5.632	0.650, 0.120
0.665, 0.126	0.000, 0.041	0.250, 0.149	0.000, 0.029	0.000, 0.122
0.500, 0.105	0.000, 0.058	0.000, 0.166	0.000, 0.110	0.000, 0.122
0.000, 0.136	0.000, 0.122	0.000, 0.145	0.000, 0.132	0.000, 0.122
0.243, 0.142	0.000, 0.107	0.000, 0.146	0.000, 0.122	0.000, 0.168
0.000, 0.081	0.000, 0.044	0.000, 0.136	0.000, 0.145	0.000, 0.161
0.000, 0.169	0.000, 0.093	0.000, 0.122	0.000, 0.167	0.000, 0.152
0.000, 0.058	0.000, 0.077	0.000, 0.490	0.000, 0.122	0.000, 0.078

0.000, 0.122	0.000, 0.142	0.084, 0.138	0.000, 0.154
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.122
0.000, 0.135	0.000, 0.163	0.000, 0.099	0.000, 0.122
0.000, 0.122	0.000, 0.122	0.000, 0.122	0.000, 0.423
0.000, 0.149	0.000, 0.119	0.000, 0.044	0.000, 0.022
0.000, 0.174	0.000, 0.147	0.000, 0.051	0.000, -0.113
0.000, 0.146	0.000, 0.132	0.000, 0.140	0.000, 0.146
0.000, 0.131	0.000, 0.087	0.000, 0.154	0.330, 0.131
1.000, 0.139	0.000, 0.122	0.000, 0.122	0.000, 0.150
0.000, 0.175	0.000, 0.099	0.000, 0.157	
0.618, 0.140	0.000, 0.122	0.000, 0.122	
0.000, 0.122	0.000, 0.122	0.000, 0.133	
0.000, 0.026	0.000, 0.122	0.000, 0.158	
0.000, 0.158	0.000, 0.117	0.041, 0.157	
0.000, 0.131	0.000, 0.122	0.000, 0.143	
0.000, 0.122	0.000, 0.122	0.000, 0.131	
0.000, 0.157	0.000, 0.122	0.000, 0.584	
0.000, 0.167	0.000, 0.145	0.000, 0.089	
0.000, 0.122	0.000, 0.095	0.000, 0.158	
0.000, 0.122	0.000, 0.071	0.000, 0.122	
0.000, 0.153	0.000, 0.122	0.000, 0.162	
0.000, 0.058	0.000, 0.156	0.000, 0.124	
0.000, 0.122	0.000, 0.022	0.000, 0.122	
0.000, 0.037	0.000, 0.125	0.000, 0.135	
0.000, -0.126	0.000, 0.169	0.000, 0.122	
0.000, 0.122	0.000, 0.137	0.000, 0.158	
0.000, 0.170	0.000, 0.174	0.000, 0.122	
0.000, 0.148	0.000, 0.117	0.000, 0.129	
0.000, 0.122	0.000, 0.143	5.000, 0.137	
0.000, 0.121	0.000, 0.100	0.000, 0.099	
0.000, 0.127	0.000, 0.014	1.672, 1.030	
0.000, 0.112	0.000, 0.132	0.000, 0.122	
0.000, 0.082	0.000, 0.099	0.000, 0.153	
0.000, 0.122	0.000, 0.122	0.000, -0.135	
0.000, 0.147	0.000, 0.117	0.000, 0.132	
0.000, 0.122	0.000, 0.137	0.000, 0.016	
0.000, 0.122	0.000, 0.170	0.000, 0.121	
0.000, 0.140	0.000, 0.012	0.000, 0.122	
0.000, -0.548	0.000, 0.158	0.000, -0.007	
0.000, 0.037	0.000, 0.093	0.000, 0.134	
0.000, 0.144	0.000, 0.149	0.000, 0.095	
0.000, 0.168	0.706, 0.144	0.000, 0.122	
0.000, 0.122	0.000, 0.169	0.000, 0.146	
0.000, 0.122	0.000, 0.042	0.000, 0.122	
0.000, 0.159	0.000, 0.134	0.000, 0.146	
0.000, -0.066	0.000, 0.131	0.000, 0.084	

Fold 8	0.000, 0.507	0.000, 0.124	0.000, 0.102	0.000, 0.102
	0.000, 0.168	0.000, 0.102	0.000, 0.102	6.000, -0.056
0.000, 0.101	0.000, 0.144	0.000, 0.094	0.000, 0.102	0.000, 0.113
0.000, 0.137	0.000, 0.117	0.000, 0.156	0.000, 0.117	0.000, 0.151
0.644, 0.120	0.000, 0.135	4.030, 0.013	0.000, 0.102	0.000, 0.102
0.000, 0.109	0.000, 0.109	2.247, 0.087	0.000, 0.105	0.000, 0.138
1.000, 0.126	0.000, 0.096	0.000, 0.102	0.000, -0.050	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.109	0.000, 0.157	0.000, 0.102
0.000, 0.112	0.000, 0.102	0.000, 0.121	0.000, 0.102	0.504, 0.132
0.000, 0.082	0.000, 0.101	0.000, 0.102	0.040, 0.148	0.000, 0.102
0.000, 0.109	0.000, 0.058	0.000, 0.160	0.000, -0.106	0.000, 0.102
0.000, 0.168	0.000, 0.102	0.000, 0.160	1.000, -0.201	0.000, 0.164
0.000, 0.178	0.000, 0.106	0.000, 0.157	0.000, 0.170	0.000, 0.124
0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.094	0.000, 0.146
0.000, 0.120	0.000, 0.115	0.000, 0.150	0.000, 0.007	0.000, 0.163
0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.139	0.000, 0.141
0.000, 0.102	0.000, 2.158	0.000, -0.019	0.000, 0.102	0.000, 0.087
0.000, 0.082	0.000, 0.092	0.000, 0.102	0.000, 0.093	0.000, 0.149
0.000, 0.102	0.000, 0.150	0.000, -0.002	0.000, 0.117	0.000, 0.125
0.000, 0.148	0.000, 0.099	0.000, 0.148	0.000, 0.102	0.000, 0.148
0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.132	0.000, 0.036
0.000, 0.061	0.000, 0.102	0.000, 0.125	0.000, 0.102	0.000, 0.150
0.000, 0.102	0.000, 0.090	0.000, -0.045	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.141	0.000, 0.131	0.000, 0.097
0.000, 0.512	0.000, 0.110	0.000, 0.126	0.000, 0.121	0.000, 0.132
0.000, 0.112	0.000, 0.080	0.000, 0.165	0.000, 0.130	0.000, 0.114
0.000, 0.092	0.000, 0.102	0.000, 0.035	0.000, 0.102	0.000, 0.070
0.000, 0.097	0.000, 0.102	0.000, 0.103	0.455, 0.131	0.000, 0.102
0.000, 0.043	1.313, 0.114	0.000, 0.092	0.000, 0.101	0.000, 0.115
0.000, 0.132	0.000, 0.102	0.964, 0.131	0.000, 0.102	0.000, 0.160
0.000, 0.448	0.000, 0.088	0.000, 0.102	0.000, 0.144	0.000, 0.115
0.000, 0.102	0.000, 0.148	0.000, 0.101	0.000, 0.102	0.000, 0.150
0.000, 0.130	0.000, 0.109	0.000, 0.102	0.000, 0.104	0.000, -0.038
0.000, 0.137	0.000, 0.088	0.000, 0.102	0.000, 0.112	0.197, 0.183
0.000, 0.132	0.000, 0.148	0.000, 0.102	0.000, 0.126	0.000, 0.118
0.000, 0.161	0.000, 0.111	0.000, 0.102	0.000, 0.102	0.000, 0.128
0.000, 0.285	0.000, 0.102	0.000, 0.102	0.000, 0.164	0.000, 0.031
0.000, 0.112	7.325, 0.116	0.000, 0.102	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.141	0.000, 0.118	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 1.935	0.640, 0.157	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.083	0.000, 0.102	0.000, 0.104	0.000, 0.137
0.000, 0.133	0.000, 0.102	0.000, 0.112	0.000, 0.102	0.000, 0.102
0.000, 0.165	0.000, 0.102	0.000, 0.117	0.000, 0.144	0.000, 0.130
0.000, -0.055	0.000, 0.102	0.000, 0.127	0.000, 0.102	0.000, 0.102
0.000, 0.115	0.000, 0.079	0.000, 0.069	0.000, 0.064	0.000, 0.155
0.000, 0.146	0.000, 0.147	0.000, 0.102	0.846, 0.081	0.000, 0.120

1.000, 0.091	0.000, 0.102	0.000, -0.018	0.000, 0.102	0.000, 0.102
0.000, 0.129	0.000, 0.102	0.000, 0.102	0.000, 0.123	0.000, 0.173
0.000, 0.068	0.000, 0.005	0.000, -0.018	0.000, 0.125	0.000, 0.142
0.000, 0.102	0.000, 0.102	0.000, 0.092	0.000, -0.269	0.000, 0.102
0.000, 0.095	0.000, 0.133	0.288, 0.101	0.000, 0.102	0.000, 0.108
0.000, 0.163	0.000, 0.080	0.180, 0.160	0.000, 0.111	0.000, 0.119
0.000, 0.102	0.000, -0.014	0.000, 0.102	0.000, 0.121	0.000, 0.102
0.000, 0.118	0.000, 0.027	0.000, 0.136	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.156	0.000, 0.067	0.000, 0.102	0.000, 0.102
0.000, -0.006	0.000, 0.156	0.000, 0.148	0.000, 0.136	0.000, 0.138
0.000, 0.082	0.000, 0.102	0.000, -0.032	0.000, 0.148	5.136, 0.054
0.000, 0.102	0.000, 0.116	0.000, 0.102	0.180, 0.151	0.000, 0.115
0.000, 0.126	0.000, 0.138	0.000, 0.102	0.000, 0.129	0.000, 0.089
0.000, 0.102	0.000, 0.148	0.000, 0.096	0.000, 0.118	0.000, 0.114
0.000, 0.155	0.456, 0.300	0.000, 0.109	0.000, 0.150	0.000, 0.102
0.000, 0.102	0.000, 0.116	0.000, 0.022	0.000, 0.102	0.000, 0.102
1.000, 0.135	0.000, 0.040	0.000, 0.103	2.000, 0.115	0.000, 0.150
0.000, 0.102	0.000, 0.148	0.000, 0.152	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.195	0.000, 0.021	0.000, 0.102	0.000, 0.076
0.000, 0.103	0.000, 0.102	0.130, 0.142	0.000, 0.108	0.000, 0.137
0.000, 0.102	0.000, 0.102	0.000, 0.015	0.000, 0.102	0.000, 0.136
0.000, 0.164	0.000, 0.109	0.000, 0.120	0.000, 0.102	0.000, 0.176
0.000, 0.138	0.000, 0.129	0.000, 0.148	0.000, 0.116	0.000, 0.101
0.000, 0.118	0.000, 0.116	0.000, 0.095	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.123	0.000, 0.102	4.000, 0.120	0.000, 0.145
0.000, 0.148	0.000, 0.156	0.000, 0.102	0.000, 0.121	0.000, 0.086
0.000, 0.104	0.000, 0.153	0.000, 0.154	0.000, 0.102	0.000, 0.129
0.000, 0.141	0.000, 0.100	0.000, 0.125	0.000, 0.119	0.000, 0.148
0.000, 0.102	0.000, 0.102	0.000, 0.098	0.000, 0.102	0.000, 0.102
0.000, 0.054	0.000, 0.103	0.000, 0.102	6.027, 0.015	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.101	0.000, 0.102	0.000, 0.084
0.000, 0.106	0.160, 0.112	0.000, -0.003	0.000, 0.102	0.000, 0.132
0.000, 0.115	0.000, 0.122	0.000, 0.148	0.000, 0.102	0.000, 0.102
0.000, 0.058	0.000, 0.102	0.000, 0.147	0.000, 0.102	0.000, 0.148
0.000, 0.102	0.000, 0.109	11.660, 0.134	0.000, 0.083	0.000, 0.148
0.000, 0.111	0.000, 0.201	0.000, 0.102	0.000, 0.148	0.000, 2.159
0.000, 0.102	0.000, 0.178	0.000, 0.148	0.000, 0.107	0.000, 0.109
0.000, 0.102	0.000, 0.094	0.000, 0.102	0.000, 0.102	0.000, 0.129
0.000, 0.108	0.000, 0.101	0.000, -0.032	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.081	0.000, 0.003	0.000, 0.127	0.000, 0.143
0.000, 0.076	2.000, 0.105	0.000, 0.102	0.000, 0.061	0.000, 0.158
0.000, 0.117	0.000, 0.102	0.000, 0.102	0.000, -0.057	0.000, 0.102
0.000, 0.102	0.000, 0.150	0.000, 0.128	0.000, 0.114	0.000, 0.102
0.000, 0.102	0.000, 0.140	0.000, 0.150	1.076, 0.120	0.000, 0.032
0.000, 0.102	0.000, 0.132	0.000, 0.102	0.000, 0.102	0.360, 0.143
0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.102

0.000, 0.086	0.000, 0.166	0.000, 0.152	0.000, 0.102	0.000, 0.118
0.000, 0.102	0.000, -0.010	0.000, 0.159	0.000, 0.097	0.000, 0.170
0.000, 0.102	0.333, 0.156	0.000, 0.135	0.000, 0.147	2.850, 0.069
0.000, 0.138	0.000, 0.150	0.000, 0.078	0.000, 0.089	0.000, 0.102
0.000, 0.139	0.000, -0.348	0.000, 0.095	0.000, 0.109	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.146	0.000, 0.128	1.970, 0.172
0.000, 0.139	0.000, 0.102	0.000, 0.102	0.000, -0.057	0.000, 0.102
0.000, 0.102	0.000, 0.149	0.000, 0.102	0.000, 0.146	0.000, 0.104
0.000, 0.102	0.000, -0.031	0.000, 0.167	0.000, 0.110	0.000, 0.102
0.000, 0.146	0.000, 0.050	0.000, 0.026	0.000, 0.102	0.000, 0.089
0.000, 0.075	0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.102
0.000, 0.109	0.000, 0.141	0.000, 0.102	0.000, 0.091	0.000, 0.148
0.000, 0.076	0.000, 0.148	0.000, 0.101	0.000, 0.096	0.000, 0.102
0.000, 0.109	0.000, 0.102	0.000, 0.119	0.000, 0.103	0.000, 0.102
0.000, 0.139	0.000, 0.117	0.000, 0.139	0.000, 0.035	0.000, 0.102
0.000, 0.114	0.000, 0.102	0.280, 0.105	0.000, 0.102	0.000, 0.126
0.000, 0.128	0.000, 0.120	0.000, 0.102	0.000, 0.086	0.000, 0.102
0.000, 0.102	0.000, 0.137	0.000, 1.511	0.000, 0.102	0.000, 0.120
0.000, 0.106	0.000, 0.127	0.000, 0.118	0.000, 0.137	0.000, 0.148
0.000, 0.102	0.000, 0.036	0.000, 0.102	0.000, 0.102	0.000, 0.100
0.000, 0.184	0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.155
2.000, 0.097	0.171, 0.098	0.000, 0.160	6.000, -0.864	0.000, 0.125
0.000, 0.102	0.000, 0.079	0.000, 0.121	1.000, 0.079	0.000, 0.124
0.000, 0.103	0.000, 0.113	3.000, 0.126	0.000, 0.117	0.000, 0.102
0.000, 0.094	0.000, 0.108	0.000, 0.160	0.000, 0.112	0.000, 0.094
0.000, 0.188	0.000, 0.148	2.000, -0.101	0.791, 0.102	0.000, 0.155
0.000, 0.102	0.000, -0.014	0.000, 0.132	0.000, 0.105	0.000, 0.117
0.000, 0.112	0.000, 0.081	0.000, 0.102	0.000, -0.007	0.000, 0.133
0.000, 0.135	0.000, 0.160	0.000, 0.153	0.000, 0.149	0.000, 0.102
0.000, 0.093	0.000, 0.132	0.000, 0.121	0.000, 0.114	0.000, 0.124
0.000, 0.102	0.000, 0.122	0.000, 0.079	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.102	0.000, 0.128	0.000, 0.102
0.000, -0.026	0.000, 0.036	0.000, -0.036	0.000, 0.102	0.000, -0.014
0.000, 0.102	0.000, 0.102	0.000, 0.157	0.000, 0.102	0.000, 0.110
0.000, 0.169	0.000, 1.940	0.000, 0.111	0.000, 0.134	0.000, 0.151
0.000, 0.143	0.000, 0.146	0.000, 0.102	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.118	0.000, 0.126	0.000, -0.193	0.000, 0.102
0.000, 0.111	0.000, 0.148	0.000, 0.102	0.000, 0.151	0.000, 0.102
0.000, 0.113	0.000, 0.111	0.000, 0.102	0.000, 0.125	0.000, 0.102
0.000, 0.088	0.000, 0.102	0.000, 0.118	0.000, 0.077	0.000, 0.124
0.000, 0.102	0.000, 0.127	0.000, 0.084	0.000, 0.160	0.000, 0.148
0.000, 0.112	0.000, 0.285	0.000, 0.134	0.000, 0.105	0.000, 0.087
0.000, 0.114	0.000, 0.130	0.000, 0.137	0.000, 0.131	0.000, 0.102
0.000, 0.115	0.000, 0.102	0.000, -0.085	0.000, 0.102	0.000, 0.014
0.000, 0.102	0.572, 0.107	0.000, 0.026	0.000, 0.102	8.000, 0.150
0.000, 0.102	0.000, 0.110	0.000, 0.102	0.000, 0.113	0.000, 0.102

0.000, 0.079	0.000, 0.102	0.000, 0.102	0.000, 0.102
0.000, 0.102	2.000, 0.074	0.000, 0.090	0.000, 0.102
0.000, 0.098	0.000, 0.112	0.000, 0.148	0.000, -0.154
0.000, 0.115	0.000, 0.147	0.000, 0.102	0.000, 0.102
0.110, 0.068	0.000, 0.117	0.000, 0.054	0.000, 0.122
0.000, 0.102	0.000, 0.102	0.000, 0.127	0.000, 0.120
0.000, 0.168	0.000, 0.129	0.000, 0.102	0.000, 0.102
0.000, 0.102	0.000, 0.102	0.000, 0.127	0.000, 0.102
0.000, 0.102	0.000, -0.065	1.000, 0.142	0.000, 0.161
0.000, 0.015	0.000, 0.159	0.000, 0.140	
0.000, 0.102	0.000, 0.148	0.000, 0.102	
0.000, 0.108	0.000, 0.109	0.000, 0.102	
0.000, 0.102	0.000, 0.165	0.000, 0.154	
0.000, 0.102	0.000, 0.116	0.000, 0.113	
0.000, 0.102	0.000, 0.102	0.000, 0.143	
0.000, 0.111	0.000, 0.102	0.000, 0.146	
5.000, 0.124	0.000, 0.115	0.000, 0.050	
0.000, 0.011	0.000, 0.102	0.000, 0.155	
0.000, 0.132	0.000, 0.131	0.000, 0.117	
0.000, 0.026	0.000, 0.156	0.000, 0.113	
0.000, 0.148	0.000, 0.148	0.000, 2.804	
5.400, 0.120	0.000, 0.111	0.000, 0.148	
0.000, 0.100	0.000, 0.102	0.000, 0.111	
0.000, 0.123	0.000, 0.116	0.000, 0.093	
0.000, 0.109	0.000, 0.102	0.000, -0.037	
0.000, 0.114	0.000, 0.110	0.000, 0.102	
0.000, 0.154	0.000, 0.101	0.000, 0.102	
0.000, 0.101	0.000, 0.102	0.000, 0.143	
0.000, 0.120	0.000, 0.102	0.000, 0.102	
0.000, 0.130	0.000, 0.070	0.000, -0.099	
0.000, 0.102	0.000, 0.161	0.000, 0.157	
0.000, 0.139	0.000, 0.121	0.000, 0.149	
0.000, 0.119	0.465, 0.150	0.000, 0.072	
0.000, 0.102	0.000, 0.095	0.000, 0.738	
0.000, 0.130	0.000, 0.102	0.000, 0.012	
0.000, 0.102	0.476, 0.101	0.000, 0.102	
0.000, 0.100	0.000, 0.004	0.000, 0.102	
0.000, 0.102	0.000, 0.102	0.000, 0.102	
0.000, 0.103	0.000, 0.102	0.000, 0.102	
0.000, 0.153	0.000, 0.102	0.000, 0.102	
0.000, -0.016	0.000, 0.157	0.000, 0.102	
0.000, 0.159	0.000, 0.102	0.000, 0.102	
0.000, 0.116	0.000, 0.166	0.000, 0.102	
0.000, 0.170	0.000, 0.098	0.000, 0.109	
0.000, 0.058	0.000, 0.118	0.000, 0.129	
1.581, 1.397	0.000, 0.102	0.000, -0.055	

Fold 9	0.000, 0.141	2.000, 0.080	0.000, 0.073	0.000, 0.073
	0.000, 0.115	0.000, 0.073	0.000, 0.073	0.000, 0.124
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.046
2.000, 0.139	0.000, -0.029	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.089	0.000, 0.081	0.000, 0.143	0.000, 0.116	0.000, 0.072
0.000, 0.028	0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.085
0.270, 0.080	0.000, 0.128	0.000, 0.081	0.000, 0.144	0.000, 0.111
0.303, 0.071	0.000, 0.073	0.000, -0.059	0.000, 0.085	0.000, 0.139
0.000, 0.071	0.108, 0.081	0.000, -0.479	0.000, 0.092	0.000, 0.073
0.000, 0.104	0.000, 0.108	0.000, 0.073	0.000, 0.073	0.000, 0.157
0.000, 0.107	0.000, 0.037	0.000, 0.073	0.000, 0.091	0.000, 0.083
0.000, 0.073	0.000, 0.073	0.000, 0.105	0.000, 0.068	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.106	0.000, 0.136	0.000, -0.020
0.458, 0.071	0.000, 0.135	0.000, 0.057	0.000, 0.073	0.000, 0.111
0.000, 0.073	0.000, 0.096	0.000, 0.073	0.000, 0.205	0.000, 0.155
0.000, 0.076	0.000, 0.086	0.000, 0.100	0.000, 0.081	0.000, 0.073
0.000, 0.117	0.000, 0.081	0.000, 0.073	0.000, 0.088	0.000, 0.111
0.000, 0.099	0.000, 0.156	0.000, 0.074	0.000, 0.053	0.000, 0.073
0.000, 0.136	0.000, 0.114	0.000, 0.086	0.000, 0.073	0.000, 0.073
0.000, 0.130	0.000, 0.073	0.000, 0.127	0.000, 0.158	0.000, 0.082
0.000, 0.139	0.000, 0.073	0.000, 0.117	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.063
0.000, 0.073	0.000, 0.073	0.000, 0.097	0.000, 0.084	0.000, 0.097
0.000, 0.147	0.000, 0.138	0.000, 0.068	0.000, 0.079	0.000, 0.058
0.000, 0.073	0.000, 0.153	0.000, 0.127	0.000, 0.073	0.000, 0.073
0.000, 0.139	0.000, 0.090	5.005, 0.115	0.000, 0.073	0.000, 0.073
0.000, 0.082	0.000, 0.143	0.000, 0.073	0.000, 0.116	0.000, 0.141
0.000, 0.089	0.000, 0.100	0.000, 0.073	0.000, 0.139	0.000, 0.073
0.000, 0.147	0.000, 0.420	0.000, 0.081	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.075	0.000, 0.045	0.000, 0.073
0.000, 0.056	0.000, 0.139	0.000, 0.064	0.000, 0.123	0.000, 0.048
0.000, 0.135	0.000, 0.096	0.000, 0.112	0.000, 0.073	0.000, 0.133
0.470, 0.064	0.000, 0.073	0.000, 0.073	0.000, 0.128	0.000, 0.073
0.000, 0.073	0.000, 0.157	0.000, 0.124	0.000, 0.073	10.000, 0.089
0.000, 0.035	0.000, 0.073	2.656, 0.065	0.000, 0.086	0.000, 0.073
0.000, 0.075	0.000, 0.151	0.000, 0.073	0.000, 0.081	0.000, 0.073
0.000, 0.094	0.000, 0.073	0.000, 0.073	0.000, 0.129	0.000, 0.159
0.000, 0.080	0.000, 0.073	0.000, 0.079	0.000, 0.103	0.000, 0.073
0.000, -0.016	0.000, 0.139	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.131	0.000, 0.086	0.000, 0.073	0.000, 0.073
0.000, 0.115	0.000, 0.073	0.000, 0.139	0.000, -0.081	0.000, 0.136
0.000, 0.085	0.000, 0.142	0.000, 0.139	0.000, 0.116	0.000, 0.081
0.000, 0.103	0.000, 0.073	1.000, 0.021	0.000, 0.132	0.000, 0.080
0.000, 0.073	0.000, 0.073	7.610, 0.108	0.000, 0.072	0.000, 0.073
0.000, 0.137	0.000, 0.073	0.000, 0.073	0.000, 0.085	0.000, 0.083
0.000, 0.145	1.000, 0.071	0.000, 0.135	0.000, 0.146	0.000, 0.110

0.000, 0.073	0.000, 0.789	0.000, 0.062	0.000, 0.117	0.000, 0.116
0.000, 0.081	0.819, 0.138	0.000, 0.073	3.000, 0.053	0.000, 0.073
0.000, 0.187	0.000, 0.090	0.000, 0.156	0.000, 0.061	0.000, 0.096
0.000, 0.073	0.000, 0.081	0.000, 0.211	0.000, 0.086	0.000, 0.073
0.048, 0.082	0.000, 0.073	0.000, 0.122	0.000, -0.179	0.000, 0.109
0.000, 0.149	0.000, 0.081	0.000, 0.159	0.000, 0.084	0.000, 0.123
0.000, 0.073	0.000, 0.073	0.000, 0.081	0.000, 0.066	0.000, 0.114
0.000, 0.073	0.000, 0.025	0.000, 0.086	0.000, 0.129	0.000, 0.103
0.000, 0.065	0.000, 0.145	0.000, -0.118	0.000, 0.073	0.000, 0.142
0.000, 0.073	0.000, 0.082	0.000, 0.076	0.000, 0.104	0.000, 0.049
0.000, 0.085	0.000, 0.132	0.000, 0.179	0.000, 0.073	0.000, 0.120
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.252
0.000, 0.073	0.000, 0.067	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.154	0.000, 0.090	0.000, 0.073	0.000, 0.073
0.000, 0.079	0.000, 0.082	0.000, 0.073	0.000, 0.051	4.983, 0.128
0.000, 0.071	0.000, 0.079	0.000, 0.086	0.000, -0.005	0.000, 0.073
0.000, 0.137	0.000, 0.130	0.000, 0.070	0.000, 0.116	0.000, 0.073
0.000, 0.152	0.500, 0.131	0.000, 0.124	0.000, 0.073	0.000, -0.036
2.100, 0.147	0.000, 0.062	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.121	0.000, 0.073	0.000, 0.073	0.000, 0.107	0.000, 0.150
0.000, 0.082	0.000, 0.034	0.000, 0.124	0.000, 0.098	0.000, 0.073
0.000, 0.073	2.000, -0.012	0.000, 0.094	8.000, 0.816	0.000, 0.081
0.000, 0.093	0.000, 0.073	0.000, 0.073	0.000, 0.121	0.000, 0.073
0.000, 0.127	0.000, 0.150	0.000, 0.106	0.000, 0.215	0.000, 0.073
0.000, 0.073	0.000, 0.033	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.395	0.000, 0.073	0.000, 0.081	0.000, 0.139	0.000, 0.081
3.360, 0.095	0.000, 0.073	0.000, -0.081	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.137	0.000, 0.081
0.000, 0.115	0.000, 0.073	0.000, 0.436	0.000, 0.107	0.000, 0.073
0.322, 0.125	0.000, 0.139	0.000, 0.081	0.000, 0.073	0.000, 0.129
0.000, 0.073	0.000, 0.073	0.000, 0.121	0.000, 0.073	1.624, 0.115
0.000, 0.073	0.000, -1.155	0.000, 0.073	10.590, 0.151	0.000, 0.439
0.000, 0.073	0.000, -0.106	0.000, 0.073	0.000, 0.148	0.000, 0.045
0.000, 0.086	0.000, 0.081	0.000, 0.151	0.000, 0.125	0.000, 0.145
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.382, 0.122	0.000, 0.073
0.000, 0.060	0.000, 0.107	0.000, 0.099	0.000, 0.070	0.000, 0.073
0.000, 0.141	0.000, 0.068	0.000, 0.158	0.000, 0.128	0.000, 0.082
0.000, 0.105	0.000, 0.073	0.000, 0.082	0.000, 0.155	0.000, 0.032
0.000, 0.073	0.000, 0.087	22.000, 0.127	0.000, 0.063	0.000, 0.125
0.000, 0.073	0.000, 0.096	0.086, 0.096	0.000, 0.073	0.000, 0.073
0.000, 0.128	0.000, 0.087	0.000, 0.073	0.000, -0.037	0.000, 0.110
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.112	0.000, 0.106
0.000, 0.073	0.000, 0.028	0.000, -0.017	0.000, 0.119	0.000, 0.119
0.000, 0.153	0.000, 0.073	0.000, 0.122	0.000, 0.126	0.000, 0.120
0.000, 0.063	0.000, 0.073	0.000, 0.116	0.000, 0.073	0.000, 0.085
0.000, 0.073	0.000, 0.081	0.000, 0.073	0.000, 0.200	9.000, 0.063

0.000, 0.063	0.000, 0.082	0.000, 0.088	0.000, 0.139	4.000, 0.095
0.000, 0.106	0.000, 0.125	0.000, 0.073	0.000, 0.096	0.000, 0.080
0.000, 0.095	0.000, 0.017	0.000, 0.073	0.000, 0.092	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.139	0.000, 0.073
0.000, 0.073	0.000, 0.091	0.000, -0.046	0.000, 0.073	0.000, 0.018
0.000, 0.135	0.000, 0.073	0.000, 0.073	0.000, 0.081	0.000, 0.073
0.000, 0.117	0.000, 0.083	0.000, 0.102	0.000, 0.055	0.000, 0.081
0.000, 0.105	0.000, 0.087	0.000, 0.081	0.000, 0.073	0.000, 0.138
0.000, 0.073	0.000, 0.060	0.000, 0.149	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.073	0.000, 0.146	0.000, 0.073	0.000, 0.089
0.000, 0.073	0.000, 0.073	0.000, 0.098	0.000, 0.072	0.000, 0.102
0.000, 0.155	0.000, 0.091	0.000, 0.114	0.130, 0.114	0.000, 0.126
0.000, 0.073	0.000, 0.073	0.000, 0.133	0.000, 0.073	0.000, 0.073
0.000, 0.116	0.000, 0.118	0.000, 0.129	0.000, 0.073	0.000, 0.159
0.000, -0.120	0.000, 0.073	0.000, 0.128	0.000, 0.158	0.000, 0.141
0.000, 0.073	0.489, 0.126	0.000, 0.143	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.127	0.000, 0.073	0.000, 0.073	0.000, 0.121
0.000, 0.081	0.000, 0.073	0.000, 0.064	1.000, 0.081	0.000, 0.109
0.168, 0.117	0.000, -0.338	0.000, 0.073	0.000, 0.073	1.000, 0.137
0.000, 0.118	0.000, 0.160	0.000, 0.160	0.000, 0.073	0.000, 0.129
0.000, 0.121	0.000, 0.136	0.000, 0.073	0.000, 0.073	0.000, 0.118
0.000, 0.073	0.000, 0.122	0.000, 0.074	0.000, 0.099	0.000, 0.139
0.000, 0.107	0.000, 0.141	0.000, 0.071	0.000, -0.134	0.000, 0.073
0.000, 0.139	0.000, 0.090	0.000, 0.093	0.000, 0.121	0.000, 0.139
0.000, 0.120	0.000, 0.073	0.000, 0.073	0.000, 0.139	0.000, 0.073
0.000, 0.112	0.000, 0.073	0.000, 0.134	0.000, 0.081	0.000, 0.073
0.000, 0.073	0.000, 0.018	0.000, 0.111	0.000, 1.871	0.000, 0.089
0.000, 0.139	0.000, 0.121	0.000, 0.139	0.000, 0.073	0.000, 0.136
0.000, 0.071	0.000, 0.074	0.000, 0.073	0.000, 0.073	0.000, 0.048
0.000, 0.053	0.000, 0.073	0.000, 0.093	0.000, 0.068	0.000, 0.121
0.000, 0.102	0.000, 0.095	0.000, 0.073	0.000, 0.181	0.000, 0.095
0.000, 0.133	0.000, 0.096	0.000, 0.073	0.000, 0.145	0.000, 0.159
1.625, 0.147	0.000, 0.071	0.000, 0.142	0.000, 0.102	0.000, 0.073
0.000, 0.083	0.000, 0.082	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.090	0.000, 0.111	0.000, 0.073	0.000, 0.110	3.100, 0.073
0.000, 0.141	0.000, 0.077	0.000, 0.073	0.000, 0.073	0.000, 0.133
0.000, 0.119	0.000, 0.141	0.000, 0.084	0.000, 0.098	0.000, 0.073
0.000, 0.096	0.000, 0.073	0.000, 0.128	0.000, 0.073	0.000, 0.073
0.000, 0.068	0.000, 0.073	0.000, 0.082	0.000, 0.152	0.000, 0.122
0.000, 0.136	0.000, 0.096	0.000, 0.158	0.000, 0.019	0.000, -0.280
0.000, 0.100	0.000, 0.091	0.000, 0.075	0.000, 0.126	0.000, 0.075
0.000, 0.129	0.000, 0.090	0.000, 0.213	0.000, 0.071	0.000, 0.073
0.000, 0.080	0.000, -0.001	0.000, 0.073	0.000, 0.073	0.000, 0.215
0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.073	0.000, 0.107	0.000, 0.073	0.000, 0.130	0.000, 0.065
0.000, 0.073	0.000, 0.073	0.000, 0.086	0.000, 0.107	0.000, 0.073

0.000, 0.120	0.000, 0.073	0.000, 0.073	0.000, 0.073
0.000, 0.073	2.000, 0.131	0.000, 0.102	0.000, 0.047
0.000, 0.090	0.000, 0.109	0.000, 0.136	0.000, -0.008
0.000, 0.073	0.000, 0.073	0.000, 0.101	0.000, 0.073
0.000, 0.073	0.000, 0.138	0.000, -0.048	0.000, 0.139
0.000, 0.149	0.000, 0.073	0.000, 0.141	0.000, 0.088
0.000, 0.073	0.000, 0.122	0.000, 0.073	0.000, 0.164
0.000, 0.105	0.000, 0.167	0.000, 0.073	0.000, 0.073
0.000, 0.070	0.000, 0.106	0.000, 0.089	0.000, 0.104
0.000, 0.073	0.000, 0.173	0.000, 0.073	
0.000, 0.073	0.000, 0.073	3.775, 0.097	
0.000, 0.073	0.000, 0.105	0.157, 0.123	
0.000, 0.073	0.000, 0.113	0.810, 0.107	
0.000, 0.071	0.000, 0.081	0.000, 0.073	
0.000, -0.067	0.000, 0.051	0.000, 0.006	
0.000, 0.098	0.000, 0.080	0.000, 0.111	
0.000, 0.081	0.000, 0.128	0.000, 0.073	
0.000, 0.105	0.000, -0.254	0.000, 0.073	
0.000, -0.124	0.000, 0.073	0.000, 0.073	
0.000, 0.107	0.000, 0.073	0.000, 2.741	
0.000, 0.073	0.000, 0.082	0.000, 0.069	
0.000, 0.139	0.000, 0.108	0.000, 0.069	
0.000, 0.073	0.000, 0.083	0.000, 0.119	
0.000, 0.073	0.000, 0.073	0.000, 0.084	
0.430, 0.731	0.000, -0.016	0.000, 0.150	
0.000, 0.073	0.000, 0.073	0.000, 0.073	
0.000, 0.073	0.000, 0.086	0.110, 0.114	
0.000, 0.105	0.000, 0.073	0.000, 0.089	
0.000, 0.139	0.000, 0.073	0.000, 0.073	
0.000, 0.122	0.000, 0.073	0.000, 0.256	
0.000, 0.073	0.000, 0.073	0.000, 0.152	
0.000, 0.042	0.000, 0.048	0.000, 0.057	
0.000, 0.129	0.000, 0.145	0.000, 0.098	
0.000, 0.073	0.000, 0.095	0.000, 0.071	
0.000, -0.152	0.000, 0.106	0.000, 0.073	
0.000, 0.132	0.000, 0.073	0.000, 0.073	
1.009, 0.137	0.000, 0.154	0.000, 0.073	
0.000, 0.147	0.000, 0.115	0.000, 0.116	
0.000, 0.073	0.000, 0.124	0.000, 0.108	
0.000, 0.068	0.000, 0.082	0.000, 0.098	
0.000, 0.073	0.000, 0.073	0.000, 0.136	
0.000, 0.103	0.000, 0.072	0.000, -0.003	
0.000, 0.058	9.977, 0.176	0.000, 0.143	
0.000, 0.138	0.000, 0.073	0.000, 0.073	
2.000, 0.111	0.000, 0.073	0.000, -0.122	
0.000, -0.115	0.000, 0.132	0.000, 0.073	

Fold 10	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081
	0.000, 0.066	0.000, 0.103	0.000, -0.001	0.000, 0.081
0.000, 0.154	0.000, 0.081	0.000, 0.142	0.000, 0.054	0.000, 0.081
0.000, 0.108	0.000, 0.085	0.000, 0.081	0.000, 0.079	0.000, 0.081
0.000, 0.081	0.500, 0.096	0.000, 0.131	0.000, 0.081	0.000, 0.081
0.000, 0.066	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.087
0.000, 0.055	0.000, 0.081	0.000, 0.073	0.000, 0.103	0.000, 0.141
0.000, 0.090	0.000, 0.147	0.000, 0.093	0.000, 0.097	0.000, 0.081
0.000, 0.146	0.000, 0.081	0.000, 0.114	0.090, 0.081	0.000, 0.019
0.000, 0.133	0.000, 0.081	0.000, 0.127	1.522, 0.104	5.496, 0.183
0.000, 0.143	0.000, 0.142	0.000, 0.127	0.000, 0.165	0.000, 0.109
0.000, 0.090	0.000, 0.144	0.000, 0.081	0.000, 0.081	0.000, 0.123
0.000, -0.080	0.000, 0.087	0.000, 0.055	0.000, 0.085	0.000, 0.089
0.000, 0.081	0.000, 0.081	0.000, 0.152	0.000, 0.103	0.000, 0.128
0.000, 0.105	0.000, 0.142	0.000, 0.081	0.000, 0.081	0.000, 0.081
0.000, 0.150	0.000, 0.105	0.000, 0.103	0.000, 0.081	0.000, -0.039
0.000, 0.081	0.000, 0.156	0.000, 0.102	0.000, 0.081	10.945, 0.102
0.000, 0.112	0.000, 0.081	0.000, 0.081	0.000, 0.094	0.000, 0.143
0.000, 0.437	0.000, 0.041	0.000, 0.136	0.000, 0.269	0.000, 0.129
0.000, 0.137	0.000, 0.141	0.000, 1.383	1.000, 0.126	0.000, 0.081
0.000, 0.081	0.094, 0.086	0.000, 0.076	0.000, 0.103	0.000, 0.086
0.000, 0.078	0.000, 0.143	0.000, 0.113	0.000, 0.081	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.090
0.000, 0.143	0.000, 0.120	0.000, 0.140	0.000, 0.103	0.000, 0.081
0.000, 0.120	0.000, 0.081	0.000, 0.094	0.000, 0.081	0.000, 0.089
0.000, 0.134	0.000, 0.119	0.000, 0.081	0.000, 0.088	0.000, 0.064
0.000, 0.113	0.000, 0.081	0.000, 0.005	0.000, 0.147	0.000, 0.081
0.000, 0.095	0.000, 0.083	0.000, 0.081	0.000, 0.102	0.000, 0.120
0.000, 0.137	0.000, 0.103	0.000, 0.081	0.000, 0.128	0.000, 0.168
0.000, 0.135	0.000, 0.084	0.000, 0.081	0.000, 0.081	0.000, -0.387
0.000, 0.092	0.000, 0.097	0.000, 0.083	0.000, 0.084	0.000, 0.023
0.000, 0.081	0.000, 0.081	0.000, 0.081	1.206, 0.202	0.000, 0.109
0.000, 0.100	0.000, 0.148	0.000, 0.081	0.695, 0.156	0.000, 0.044
0.000, 0.112	0.000, 0.103	0.000, 0.125	0.000, 0.083	0.000, 0.107
0.991, 0.058	0.000, 0.081	0.000, 0.081	0.132, 0.139	0.000, 0.150
0.000, 0.081	0.000, 0.147	0.000, 0.138	0.000, 0.081	0.000, -0.376
0.000, 0.206	0.000, 0.081	0.000, 0.117	0.000, 0.108	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.171	0.000, 0.094
0.000, 0.081	0.000, 0.081	0.000, 0.011	0.000, 0.140	0.000, 0.086
4.830, 0.136	0.000, 0.113	0.000, 0.130	0.000, 0.116	0.000, 0.153
0.000, 0.081	0.000, 0.114	0.000, 0.139	0.000, 0.037	0.000, 0.081
0.000, 0.125	0.000, 0.143	0.000, 0.114	0.000, 0.088	0.000, 0.117
0.000, 0.104	0.000, 0.085	0.000, 0.081	0.000, 0.081	0.000, 0.156
0.000, 0.081	0.000, 0.111	0.000, -0.006	0.000, 0.081	0.000, 0.078
0.000, 0.098	0.000, 0.147	0.000, 0.048	0.000, 0.103	12.208, 1.554
0.000, 0.150	0.000, 0.056	0.000, 0.085	0.000, 0.137	0.000, 0.081

0.000, 0.093	0.000, 0.139	0.000, 0.081	0.000, 0.144	0.000, 0.029
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	2.000, 0.083
0.000, -0.025	0.000, 0.119	3.858, 0.238	0.000, 0.129	0.000, 0.139
0.000, 0.120	0.000, 0.103	0.000, 0.110	0.000, 0.081	0.000, 0.081
0.000, 0.116	0.000, 0.136	0.000, 0.106	0.000, 0.134	0.000, 0.081
0.000, 0.115	0.000, 0.772	0.000, 0.122	0.000, 0.086	0.000, 0.108
0.000, 0.081	0.000, 0.081	0.000, 0.133	0.000, 0.079	0.000, 0.081
0.000, 0.084	0.000, 0.114	0.000, 0.081	0.000, 0.121	0.000, 0.708
0.000, 0.088	0.000, 0.082	0.000, 0.081	0.000, 0.051	0.000, 0.077
0.000, 0.081	0.000, 0.037	0.136, 0.124	0.000, 0.081	0.000, 0.144
0.000, 0.109	0.000, 0.081	0.000, 0.081	0.000, 0.109	0.000, 0.104
0.000, -0.012	2.000, 0.116	0.000, 0.120	0.000, 0.138	0.000, 0.114
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081
0.000, 0.072	0.000, 0.081	0.000, 0.257	0.000, 0.081	0.000, 0.103
0.000, 0.123	0.000, 0.091	0.000, 0.081	0.000, 0.081	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.117	0.000, 0.086	0.000, 0.081
0.000, 0.146	0.000, 0.127	0.000, 0.083	0.000, 0.128	0.000, 0.103
0.000, 0.081	0.000, 0.138	0.000, 0.081	0.000, 0.081	0.000, 0.113
0.000, 0.148	0.000, 0.086	0.000, 0.013	0.000, 0.167	0.000, 0.133
0.000, 0.142	0.000, 0.145	0.000, 0.082	0.000, 0.102	0.596, 0.091
0.000, 0.081	0.000, 0.103	0.000, 0.081	0.000, 0.117	0.000, 0.081
0.000, 0.141	0.000, 0.081	0.000, 0.086	0.000, 0.131	0.000, 0.081
0.000, 0.123	0.000, 0.117	0.000, -0.059	0.000, 0.103	0.000, 0.081
0.000, 0.081	0.000, 0.081	1.190, 0.108	0.000, 0.125	0.000, 0.127
0.000, 0.063	0.706, 0.084	0.000, 0.109	0.000, 0.004	0.000, 0.081
1.000, 0.103	0.000, 0.092	0.000, 0.114	0.000, 0.081	0.000, 0.165
0.000, 0.081	0.000, 0.081	0.000, 0.096	0.000, 0.082	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.145	0.000, 0.081	0.000, 0.041
0.000, 0.081	0.000, 0.081	0.000, 0.132	0.000, 0.086	0.000, 0.087
2.758, 0.099	0.000, 0.103	0.000, 0.087	0.000, 0.081	0.000, 0.081
0.000, 0.103	0.000, 0.105	0.000, 0.071	0.000, 0.121	0.000, 0.134
0.000, 0.081	0.000, 0.128	0.000, 0.081	0.000, 0.083	0.000, 0.081
0.000, 0.142	0.000, 0.127	0.000, 0.164	0.000, 0.150	0.000, 0.096
0.000, 0.114	0.303, 0.079	0.000, 0.121	0.000, 0.081	0.000, 0.089
0.000, 0.084	0.000, 0.129	0.000, 0.050	0.820, 0.013	0.000, 0.147
0.000, 0.126	0.000, 0.115	0.000, 0.081	0.000, 0.081	0.000, -1.309
0.000, 0.151	0.000, 0.094	0.000, 0.106	0.000, 0.132	1.970, 0.182
0.000, 0.081	0.000, -0.015	0.000, 0.103	0.000, 0.109	0.000, 0.081
0.000, 0.081	0.000, 0.087	0.000, -0.047	0.000, 0.121	0.000, 0.081
0.000, 0.162	0.000, 0.081	0.000, 0.094	0.000, 0.081	0.000, 0.103
0.000, 0.081	0.000, 0.081	0.186, 0.108	0.000, -0.064	0.000, 0.081
0.000, 0.081	0.000, 0.122	0.000, 0.132	0.000, 0.087	0.000, 0.131
0.000, 0.081	0.000, -0.006	0.000, 0.128	0.000, 0.104	0.000, 0.081
0.000, 0.089	0.000, 0.076	0.000, 0.081	1.000, 0.085	0.000, 0.042
0.000, 0.109	0.000, 0.110	0.000, 0.150	0.000, 0.103	0.000, 0.081
0.000, 0.123	0.000, -0.035	0.000, 0.084	0.465, 0.131	0.000, 0.150

0.000, 0.192	0.000, 0.081	0.000, 0.081	0.000, 0.103	0.000, 0.081
0.000, 0.128	0.000, 0.081	0.000, 0.081	0.000, 0.103	0.045, 0.155
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.115	0.000, 0.081
0.000, 0.081	0.000, -0.078	0.000, 0.081	0.000, 0.132	0.000, 0.121
0.000, 0.134	0.000, 0.081	0.000, 0.091	0.000, -0.011	0.000, 0.131
0.000, 0.081	0.000, 0.112	0.000, 0.081	0.000, 0.017	0.000, 0.081
0.000, 0.084	0.000, 0.114	0.000, 0.081	0.000, -0.062	0.000, 0.084
0.000, 0.103	0.000, 0.081	0.000, 0.081	0.000, 0.122	0.000, -0.074
0.000, 0.081	0.000, -0.036	0.000, 0.081	0.000, 0.094	0.000, -0.017
0.000, 0.081	0.000, -0.017	0.000, 0.081	0.000, 0.149	0.000, 0.081
0.000, 0.081	0.000, 0.086	0.000, -0.252	0.000, 0.125	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.089	0.000, 0.081	0.000, 0.087
0.000, 0.081	0.000, 0.189	0.000, 0.145	0.000, 0.052	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.134	0.000, 0.081
0.000, 0.089	1.000, 0.104	0.000, 0.081	0.000, 0.103	0.000, 0.081
0.000, 0.109	0.000, 0.150	0.000, 0.081	0.000, 0.096	0.000, 0.131
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.141
0.000, 0.119	0.000, 0.069	0.000, 0.105	0.000, 0.124	1.324, 0.000
6.000, -0.012	0.000, 0.103	0.000, 0.139	0.000, 0.081	0.000, 0.015
0.000, 0.143	0.000, 0.088	0.000, 0.142	0.027, 0.133	0.000, 0.120
0.000, 0.081	0.000, 0.138	0.000, 0.089	0.000, 0.137	0.000, 0.107
0.000, 0.081	0.000, 0.123	0.000, 0.081	0.000, 0.137	0.000, 0.121
0.000, 0.081	0.000, 0.081	0.000, 0.163	0.000, 0.081	0.000, 0.081
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081
0.000, 0.148	0.000, 0.080	0.000, 0.081	0.000, 0.081	0.000, 0.081
2.000, 0.093	0.000, 0.101	0.000, 0.081	0.000, 0.130	0.000, 0.039
0.000, -0.025	0.000, 0.081	0.000, 0.251	0.000, 0.103	0.000, 0.081
0.000, 0.081	0.000, 0.092	0.000, 0.081	0.000, 0.103	0.000, 0.144
0.000, 0.082	0.000, 0.096	0.000, 0.093	0.000, 0.130	0.000, 0.085
0.000, 0.155	0.000, 0.004	0.000, 0.114	0.000, 0.081	0.000, 0.095
0.000, 0.114	0.000, 0.081	0.000, 0.039	0.000, 0.084	0.000, 0.081
0.000, 0.081	0.000, 0.146	0.000, 0.108	0.000, 0.081	0.000, 0.098
0.000, 0.082	0.000, 0.147	0.000, 0.151	0.000, 0.103	0.000, 0.116
0.000, 0.130	0.000, 0.081	0.000, 0.122	0.000, 0.115	0.000, 0.081
0.000, -3.378	0.000, 0.081	4.000, 0.068	0.000, 0.089	0.000, 0.106
0.000, 0.135	0.000, 0.081	0.000, 0.083	0.000, 0.147	0.000, 0.099
0.000, 0.351	0.000, 0.081	0.000, -0.058	0.000, 0.136	0.000, 0.081
0.000, 0.103	0.000, 0.081	0.000, 0.114	0.000, 0.081	0.000, 0.540
0.000, 0.116	0.000, 0.087	0.000, 0.146	0.000, 0.104	0.000, 0.081
0.000, 0.081	0.000, 0.110	0.000, 0.168	0.000, 0.103	0.000, 0.178
0.000, 0.132	0.000, 0.197	0.000, 0.149	0.000, 0.110	0.000, 0.088
0.000, 0.087	0.000, 0.102	0.000, 0.125	0.000, 0.081	0.000, 0.023
0.000, 0.067	0.000, 0.036	0.000, 0.081	0.000, 0.040	0.000, 0.109
0.000, 0.103	0.000, -0.021	0.000, 0.187	0.000, 0.114	0.000, 0.132
0.000, 0.079	0.000, 0.081	0.000, 0.081	0.000, 0.118	0.000, 0.081
0.000, 0.015	0.000, 0.148	0.000, 0.105	0.000, 0.081	0.000, 0.103

0.000, 0.132	0.000, 0.110	0.000, 0.084	0.000, 0.081	0.896, 0.087
0.000, 0.081	0.000, 0.094	0.000, 0.134	0.000, 0.081	0.000, 0.141
0.000, 0.082	0.000, 0.154	0.000, 0.098	0.000, 0.036	0.000, 0.105
0.000, 0.142	0.000, 0.126	0.000, 0.100	0.111, 0.108	0.000, 0.063
0.000, 0.081	0.000, 0.154	0.000, 0.121	0.000, 0.111	0.000, 0.081
0.000, 0.084	0.000, 0.103	1.000, 0.102	0.000, 0.137	0.000, 0.105
0.000, 0.104	0.000, 0.143	0.000, 0.127	0.000, 0.081	2.000, 0.081
0.000, 0.170	0.000, 0.124	0.250, 0.129	0.000, 0.109	0.000, 0.117
0.000, 0.037	0.000, 0.098	0.000, 0.140	0.000, 0.184	0.000, 0.002
0.000, 0.121	0.313, 0.103	0.000, 0.133	0.000, -0.585	0.000, 0.081
3.000, -0.136	0.000, 0.092	0.000, 0.043	0.000, 0.103	0.000, 0.146
0.000, 0.130	0.000, 0.130	0.000, 0.081	0.000, 0.081	0.000, 0.156
0.000, 0.081	0.000, 0.081	0.000, 0.092	0.000, 0.081	0.000, 0.088
0.000, 0.155	0.000, 0.105	0.000, 0.135	0.000, 0.114	0.000, 0.081
0.000, 0.081	0.000, 0.146	0.000, 0.135	0.000, 0.103	0.000, 0.081
0.000, 0.119	0.000, 0.155	0.000, -0.052	0.000, 0.123	0.000, -0.024
0.000, 0.006	0.000, 0.081	0.000, 0.103	0.000, 0.204	0.000, 0.092
0.000, 0.079	0.000, 0.136	0.000, -0.068	0.000, 0.081	0.000, -0.167
0.000, 0.081	0.000, 0.108	0.000, 0.135	0.000, 0.081	0.000, 0.121
0.000, 0.114	0.000, 0.105	0.000, 0.081	1.000, 0.127	0.000, 0.107
0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.081	0.000, 0.016
0.070, 0.103	0.000, 0.121	0.000, 0.145	1.000, 0.134	2.000, 0.079
0.000, 0.081	0.000, 0.057	0.000, 0.139	0.000, 0.081	0.000, 0.108
0.000, 0.081	0.000, 0.081	0.000, 0.668	0.000, 0.081	0.000, 0.081
0.202, 0.121	0.000, 0.157	0.000, 0.131	0.000, 0.147	0.000, 0.134
0.000, 0.069	0.180, 0.168	0.000, 0.171	0.000, 0.216	0.000, -0.085
0.000, 0.107	0.000, 0.247	0.000, 0.081	0.000, 0.132	
0.417, 0.052	0.000, 0.081	0.000, 0.094	0.000, 0.088	
0.000, 0.125	0.293, 0.100	0.000, 0.108	0.000, 0.100	
0.000, 0.138	0.000, -0.012	0.000, 0.103	0.080, 0.130	

Classification results

This section presents the confusion matrices from the classification algorithms.

Seven classification algorithms were used: J48, J48 Unprunned, RIPPER, KStar, Radial Basis Functions, Random Forests, and Support Vector Machines. Each classifier has thirty resampling strategies sourced from a combination of six levels for over-sampling (Over: 0, 100, 200, 300, 400, 500) and five levels for under-sampling (Under: 0, 20, 40, 60, 80).

For each resampling strategy, there are ten confusion matrices corresponding to the 10-fold cross validation procedure. These are visualized as ten boxes, each consisting of two rows and two columns.

All the confusion matrices have the following structure:

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Classifier: J48

Over: 0 Under: 0		Over: 0 Under: 20		Over: 0 Under: 40		Over: 0 Under: 60	
785	0	784	1	778	7	763	22
50	0	47	3	47	3	44	6
785	0	782	3	785	0	769	16
50	0	49	1	49	1	46	4
784	1	782	3	785	0	782	3
49	1	49	1	50	0	49	1
784	1	784	1	784	1	753	32
49	1	49	1	49	1	42	8
785	0	785	0	784	1	784	1
50	0	50	0	48	2	48	2
785	0	781	4	782	3	773	12
50	0	50	0	50	0	50	0
785	0	785	0	784	1	772	13
50	0	50	0	50	0	49	1
785	0	782	3	785	0	781	4
50	0	49	1	49	1	48	2
784	1	784	1	785	0	767	18
50	0	49	1	50	0	45	5
785	0	782	3	784	1	769	16
49	0	48	1	48	1	44	5

Over: 0
Under: 80

702	83
28	22
701	84
38	12
739	46
42	8
768	17
45	5
690	95
35	15
729	56
39	11
697	88
33	17
764	21
45	5
757	28
43	7
725	60
39	10

Over:100
Under:20

785	0
50	0
785	0
50	0
779	6
49	1
784	1
49	1
784	1
50	0
783	2
50	0
785	0
49	1
783	2
50	0
784	1
49	1
784	1
48	1

Over: 100
Under: 60

783	2
49	1
773	12
49	1
771	14
49	1
762	23
46	4
764	21
48	2
774	11
50	0
769	16
47	3
760	25
44	6
782	3
48	2
765	20
46	3

Over: 200
Under: 0

765	20
46	4
763	22
45	5
764	21
47	3
765	20
46	4
750	35
45	5
759	26
45	5
757	28
44	6
759	26
46	4
767	18
45	5
774	11
47	2

Over: 100
Under: 0

785	0
50	0
785	0
50	0
784	1
49	1
784	1
49	1
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
784	1
50	0
785	0
49	0

Over: 100
Under: 40

782	3
48	2
783	2
49	1
784	1
49	1
782	3
49	1
785	0
50	0
777	8
50	0
785	0
48	2
782	3
49	1
785	0
50	0
782	3
48	1

Over: 100
Under: 80

747	38
39	11
732	53
45	5
720	65
37	13
703	82
36	14
721	64
42	8
696	89
35	15
690	95
40	10
708	77
37	13
755	30
47	3
721	64
36	13

Over: 200
Under: 20

758	27
42	8
770	15
45	5
758	27
46	4
760	25
41	9
745	40
43	7
731	54
42	8
745	40
45	5
745	40
44	6
753	32
40	10
760	25
45	4

Over: 200
Under: 40

755	30
40	10
734	51
41	9
738	47
43	7
735	50
43	7
728	57
43	7
714	71
40	10
730	55
44	6
756	29
47	3
737	48
36	14
739	46
40	9

Over: 200
Under: 80

559	226
17	33
578	207
25	25
596	189
17	33
466	319
16	34
567	218
22	28
565	220
23	27
557	228
21	29
591	194
26	24
574	211
20	30
593	192
27	22

Over: 300
Under: 20

743	42
37	13
747	38
42	8
741	44
43	7
734	51
42	8
739	46
45	5
741	44
40	10
727	58
42	8
747	38
37	13
756	29
39	11
758	27
43	6

Over: 300
Under: 60

664	121
25	25
659	126
31	19
702	83
32	18
648	137
35	15
669	116
36	14
669	116
41	9
672	113
29	21
641	144
33	17
691	94
36	14
645	140
34	15

Over: 200
Under: 60

725	60
34	16
680	105
36	14
728	57
34	16
710	75
39	11
671	114
37	13
673	112
31	19
732	53
43	7
712	73
39	11
726	59
39	11
699	86
33	16

Over: 300
Under: 0

764	21
41	9
760	25
43	7
757	28
48	2
745	40
43	7
743	42
45	5
750	35
41	9
747	38
41	9
770	15
48	2
755	30
43	7
773	12
48	1

Over: 300
Under: 40

743	42
36	14
707	78
38	12
726	59
44	6
723	62
40	10
724	61
43	7
710	75
42	8
712	73
39	11
720	65
40	10
728	57
37	13
727	58
39	10

Over: 300
Under: 80

559	226
14	36
547	238
28	22
487	298
14	36
552	233
21	29
609	176
27	23
436	349
8	42
533	252
20	30
544	241
24	26
537	248
14	36
563	222
22	27

Over: 400
Under: 0

766	19
44	6
754	31
44	6
747	38
43	7
737	48
42	8
738	47
42	8
742	43
41	9
747	38
43	7
755	30
49	1
758	27
41	9
757	28
46	3

Over: 400
Under: 40

704	81
30	20
725	60
37	13
734	51
38	12
731	54
44	6
728	57
42	8
697	88
40	10
701	84
38	12
731	54
41	9
740	45
40	10
731	54
39	10

Over: 400
Under: 80

550	235
17	33
534	251
18	32
579	206
14	36
488	297
17	33
543	242
22	28
549	236
19	31
425	360
14	36
501	284
16	34
549	236
18	32
565	220
24	25

Over: 500
Under: 20

728	57
36	14
728	57
34	16
719	66
34	16
738	47
43	7
726	59
42	8
704	81
39	11
740	45
46	4
738	47
37	13
746	39
40	10
740	45
38	11

Over: 400
Under: 20

734	51
31	19
742	43
40	10
759	26
45	5
727	58
39	11
725	60
41	9
713	72
41	9
721	64
42	8
722	63
41	9
754	31
43	7
734	51
41	8

Over: 400
Under: 60

670	115
27	23
655	130
29	21
664	121
25	25
641	144
29	21
639	146
35	15
641	144
34	16
676	109
37	13
636	149
32	18
654	131
27	23
654	131
33	16

Over: 500
Under: 0

736	49
29	21
753	32
44	6
744	41
39	11
725	60
35	15
730	55
49	1
729	56
39	11
741	44
47	3
760	25
48	2
749	36
40	10
738	47
41	8

Over: 500
Under: 40

691	94
25	25
718	67
40	10
729	56
35	15
670	115
36	14
703	82
43	7
683	102
35	15
710	75
41	9
697	88
33	17
708	77
31	19
731	54
41	8

Over: 500
Under: 60

623	162
18	32
638	147
28	22
654	131
29	21
663	122
37	13
666	119
35	15
640	145
29	21
627	158
29	21
622	163
27	23
669	116
27	23
672	113
30	19

Over: 500
Under: 80

525	260
11	39
537	248
23	27
544	241
14	36
556	229
20	30
536	249
20	30
514	271
19	31
506	279
19	31
554	231
22	28
549	236
20	30
582	203
25	24

Classifier: J48 Unprunned

Over: 0
Under: 0

778	7
46	4
779	6
48	2
777	8
46	4
776	9
48	2
775	10
48	2
777	8
44	6
768	17
48	2
775	10
47	3
773	12
48	2
774	11
44	5

Over: 0
Under: 40

774	11
45	5
768	17
45	5
774	11
48	2
777	8
48	2
766	19
45	5
770	15
47	3
765	20
48	2
772	13
47	3
766	19
46	4
771	14
45	4

Over: 0
Under: 80

697	88
29	21
673	112
34	16
721	64
41	9
753	32
46	4
690	95
34	16
700	85
33	17
687	98
31	19
747	38
45	5
742	43
43	7
710	75
38	11

Over: 100
Under: 20

766	19
44	6
777	8
46	4
775	10
46	4
770	15
47	3
773	12
50	0
776	9
47	3
764	21
47	3
769	16
48	2
772	13
46	4
767	18
44	5

Over: 0
Under: 20

777	8
43	7
774	11
44	6
776	9
45	5
777	8
46	4
768	17
46	4
774	11
48	2
777	8
48	2
764	21
47	3
770	15
47	3
766	19
43	6

Over: 0
Under: 60

763	22
43	7
745	40
42	8
760	25
46	4
745	40
40	10
771	14
46	4
742	43
42	8
763	22
47	3
765	20
44	6
753	32
44	6
758	27
39	10

Over: 100
Under: 0

778	7
46	4
779	6
48	2
777	8
46	4
776	9
48	2
775	10
48	2
777	8
44	6
768	17
48	2
775	10
47	3
773	12
48	2
774	11
44	5

Over: 100
Under: 40

751	34
42	8
772	13
45	5
772	13
46	4
765	20
43	7
766	19
47	3
761	24
45	5
769	16
48	2
760	25
47	3
765	20
43	7
768	17
45	4

Over: 100
Under: 60

753	32
40	10
749	36
43	7
761	24
48	2
751	34
44	6
715	70
45	5
749	36
46	4
743	42
42	8
745	40
43	7
758	27
41	9
759	26
43	6

Over: 200
Under: 0

759	26
45	5
758	27
43	7
751	34
45	5
758	27
43	7
743	42
45	5
752	33
44	6
748	37
44	6
754	31
47	3
762	23
44	6
771	14
46	3

Over: 200
Under: 40

747	38
37	13
729	56
42	8
731	54
42	8
725	60
40	10
724	61
43	7
694	91
38	12
719	66
44	6
752	33
47	3
734	51
36	14
732	53
36	13

Over: 200
Under: 80

567	218
19	31
582	203
25	25
597	188
17	33
463	322
15	35
569	216
23	27
562	223
22	28
550	235
21	29
591	194
26	24
579	206
21	29
592	193
26	23

Over: 100
Under: 80

736	49
38	12
733	52
44	6
690	95
32	18
692	93
35	15
726	59
42	8
691	94
34	16
658	127
37	13
696	89
36	14
740	45
46	4
697	88
33	16

Over: 200
Under: 20

757	28
40	10
766	19
44	6
750	35
47	3
758	27
42	8
736	49
41	9
728	57
42	8
735	50
45	5
738	47
43	7
747	38
39	11
754	31
44	5

Over: 200
Under: 60

726	59
33	17
692	93
36	14
709	76
32	18
701	84
37	13
670	115
34	16
671	114
31	19
714	71
41	9
707	78
37	13
726	59
36	14
674	111
31	18

Over: 300
Under: 0

761	24
40	10
768	17
42	8
754	31
45	5
743	42
43	7
740	45
44	6
749	36
40	10
742	43
41	9
769	16
47	3
753	32
42	8
771	14
48	1

Over: 300
Under: 20

737	48
37	13
748	37
41	9
747	38
46	4
742	43
43	7
739	46
45	5
734	51
40	10
728	57
42	8
728	57
38	12
740	45
34	16
751	34
43	6

Over: 300
Under: 60

666	119
24	26
662	123
31	19
701	84
30	20
644	141
37	13
686	99
38	12
658	127
39	11
667	118
29	21
639	146
33	17
694	91
37	13
652	133
35	14

Over: 400
Under: 0

755	30
42	8
737	48
42	8
740	45
41	9
738	47
41	9
730	55
42	8
733	52
35	15
744	41
43	7
752	33
46	4
756	29
41	9
752	33
46	3

Over: 400
Under: 40

694	91
27	23
725	60
36	14
727	58
38	12
724	61
43	7
725	60
40	10
685	100
38	12
691	94
41	9
730	55
41	9
736	49
38	12
726	59
38	11

Over: 300
Under: 40

735	50
35	15
699	86
37	13
735	50
44	6
711	74
37	13
720	65
42	8
697	88
41	9
722	63
40	10
724	61
37	13
727	58
38	12
719	66
38	11

Over: 300
Under: 80

549	236
13	37
556	229
30	20
498	287
16	34
550	235
21	29
616	169
28	22
435	350
7	43
550	235
21	29
554	231
23	27
540	245
17	33
580	205
23	26

Over: 400
Under: 20

728	57
30	20
739	46
40	10
746	39
40	10
726	59
40	10
716	69
41	9
701	84
37	13
724	61
42	8
719	66
40	10
747	38
43	7
727	58
41	8

Over: 400
Under: 60

667	118
27	23
659	126
29	21
676	109
26	24
640	145
30	20
629	156
33	17
647	138
34	16
672	113
37	13
634	151
31	19
646	139
26	24
650	135
33	16

Over: 400
Under: 80

553	232
16	34
541	244
21	29
584	201
14	36
491	294
17	33
544	241
22	28
559	226
22	28
427	358
15	35
503	282
16	34
555	230
20	30
566	219
25	24

Over: 500
Under: 20

724	61
34	16
724	61
34	16
720	65
35	15
734	51
43	7
720	65
40	10
699	86
38	12
733	52
46	4
738	47
37	13
742	43
39	11
733	52
37	12

Over: 500
Under: 60

610	175
17	33
629	156
28	22
646	139
29	21
656	129
35	15
665	120
34	16
641	144
29	21
633	152
27	23
627	158
26	24
671	114
27	23
666	119
31	18

Over: 500
Under: 0

726	59
31	19
755	30
44	6
742	43
36	14
727	58
35	15
729	56
46	4
725	60
40	10
741	44
46	4
753	32
47	3
746	39
40	10
741	44
41	8

Over: 500
Under: 40

679	106
21	29
714	71
40	10
725	60
34	16
664	121
33	17
697	88
41	9
686	99
34	16
704	81
41	9
696	89
33	17
702	83
29	21
730	55
40	9

Over: 500
Under: 80

522	263
10	40
532	253
22	28
552	233
14	36
594	191
25	25
552	233
22	28
523	262
19	31
518	267
19	31
557	228
22	28
557	228
20	30
579	206
24	25

Classifier: RIPPER

Over: 0
Under: 0

785	0
50	0
785	0
50	0
779	6
49	1
785	0
50	0
784	1
50	0
784	1
50	0
784	1
49	1
785	0
50	0
781	4
50	0
785	0
49	0

Over: 0
Under: 40

785	0
50	0
785	0
49	1
783	2
50	0
784	1
48	2
784	1
50	0
782	3
50	0
782	3
48	2
784	1
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 80

731	54
42	8
742	43
41	9
747	38
44	6
771	14
45	5
775	10
50	0
723	62
41	9
704	81
41	9
785	0
50	0
769	16
46	4
749	36
40	9

Over: 100
Under: 20

785	0
50	0
783	2
49	1
781	4
50	0
785	0
50	0
785	0
49	1
783	2
50	0
781	4
50	0
784	1
50	0
785	0
50	0
783	2
49	0

Over: 0
Under: 20

784	1
48	2
782	3
47	3
785	0
50	0
785	0
50	0
785	0
50	0
779	6
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 60

785	0
50	0
781	4
50	0
782	3
49	1
785	0
50	0
783	2
48	2
772	13
50	0
777	8
47	3
769	16
47	3
785	0
50	0
769	16
45	4

Over: 100
Under: 0

785	0
50	0
784	1
48	2
785	0
50	0
785	0
50	0
785	0
50	0
783	2
50	0
785	0
50	0
784	1
49	1
785	0
50	0
783	2
48	1

Over: 100
Under: 40

785	0
50	0
780	5
49	1
783	2
50	0
785	0
50	0
785	0
50	0
777	8
50	0
785	0
50	0
785	0
50	0
785	0
50	0
784	1
48	1

Over: 100
Under: 60

785	0
50	0
785	0
50	0
784	1
50	0
765	20
44	6
779	6
47	3
767	18
49	1
775	10
47	3
777	8
48	2
781	4
50	0
782	3
47	2

Over: 200
Under: 0

782	3
45	5
781	4
49	1
781	4
49	1
785	0
49	1
780	5
49	1
778	7
48	2
783	2
48	2
783	2
50	0
780	5
48	2
782	3
49	0

Over: 200
Under: 40

783	2
48	2
768	17
46	4
752	33
45	5
752	33
43	7
754	31
48	2
736	49
44	6
739	46
46	4
778	7
46	4
781	4
48	2
771	14
47	2

Over: 200
Under: 80

533	252
13	37
531	254
23	27
582	203
20	30
580	205
21	29
570	215
20	30
529	256
21	29
514	271
24	26
553	232
24	26
560	225
18	32
542	243
16	33

Over: 100
Under: 80

742	43
36	14
731	54
42	8
739	46
44	6
747	38
41	9
743	42
41	9
734	51
47	3
728	57
42	8
757	28
43	7
716	69
36	14
769	16
45	4

Over: 200
Under: 20

784	1
48	2
778	7
47	3
785	0
48	2
780	5
48	2
775	10
48	2
778	7
50	0
783	2
49	1
781	4
49	1
776	9
50	0
779	6
49	0

Over: 200
Under: 60

666	119
28	22
672	113
35	15
709	76
39	11
704	81
36	14
714	71
37	13
665	120
35	15
693	92
43	7
676	109
33	17
696	89
32	18
738	47
43	6

Over: 300
Under: 0

783	2
48	2
779	6
48	2
782	3
48	2
778	7
50	0
773	12
49	1
779	6
50	0
779	6
49	1
782	3
49	1
782	3
49	1
783	2
48	1

Over: 300
Under: 20

782	3
46	4
773	12
47	3
780	5
49	1
783	2
50	0
784	1
49	1
771	14
47	3
777	8
47	3
778	7
49	1
767	18
46	4
779	6
49	0

Over: 300
Under: 60

741	44
36	14
667	118
36	14
695	90
33	17
675	110
32	18
673	112
35	15
687	98
42	8
698	87
36	14
659	126
22	28
682	103
37	13
739	46
43	6

Over: 400
Under: 0

785	0
48	2
775	10
48	2
783	2
49	1
784	1
50	0
781	4
50	0
775	10
48	2
780	5
48	2
777	8
49	1
778	7
49	1
778	7
46	3

Over: 400
Under: 40

772	13
45	5
742	43
45	5
775	10
48	2
769	16
49	1
739	46
46	4
758	27
48	2
762	23
46	4
765	20
46	4
769	16
48	2
777	8
48	1

Over: 300
Under: 40

783	2
46	4
754	31
43	7
762	23
47	3
765	20
45	5
754	31
47	3
776	9
48	2
762	23
49	1
737	48
42	8
736	49
38	12
758	27
42	7

Over: 300
Under: 80

490	295
10	40
460	325
18	32
451	334
10	40
438	347
8	42
454	331
15	35
413	372
14	36
433	352
12	38
470	315
14	36
428	357
10	40
481	304
14	35

Over: 400
Under: 20

780	5
42	8
779	6
48	2
777	8
49	1
777	8
49	1
768	17
48	2
773	12
47	3
776	9
48	2
779	6
50	0
771	14
48	2
778	7
49	0

Over: 400
Under: 60

765	20
43	7
653	132
33	17
686	99
35	15
637	148
28	22
655	130
34	16
658	127
38	12
660	125
36	14
659	126
33	17
638	147
28	22
644	141
34	15

Over: 400
Under: 80

400	385
3	47
439	346
14	36
411	374
10	40
507	278
14	36
471	314
16	34
431	354
12	38
394	391
11	39
448	337
14	36
499	286
15	35
429	356
6	43

Over: 500
Under: 20

782	3
47	3
779	6
48	2
774	11
48	2
776	9
48	2
765	20
49	1
767	18
48	2
755	30
47	3
778	7
49	1
771	14
48	2
781	4
48	1

Over: 500
Under: 60

623	162
25	25
636	149
30	20
691	94
37	13
604	181
29	21
706	79
33	17
659	126
37	13
645	140
33	17
634	151
34	16
688	97
38	12
695	90
40	9

Over: 500
Under: 0

783	2
49	1
773	12
48	2
780	5
48	2
784	1
50	0
782	3
50	0
776	9
49	1
775	10
49	1
781	4
49	1
783	2
49	1
784	1
48	1

Over: 500
Under: 40

733	52
33	17
644	141
30	20
761	24
47	3
769	16
46	4
778	7
48	2
730	55
43	7
743	42
42	8
764	21
46	4
761	24
45	5
762	23
48	1

Over: 500
Under: 80

408	377
4	46
417	368
11	39
434	351
8	42
486	299
18	32
417	368
7	43
403	382
12	38
429	356
16	34
491	294
15	35
420	365
6	44
505	280
13	36

Classifier: KStar

Over: 0
Under: 0

766	19
41	9
768	17
43	7
762	23
44	6
766	19
47	3
760	25
44	6
761	24
40	10
765	20
46	4
755	30
43	7
769	16
45	5
755	30
41	8

Over: 0
Under: 40

752	33
39	11
756	29
39	11
751	34
43	7
747	38
44	6
744	41
42	8
737	48
40	10
745	40
42	8
738	47
42	8
752	33
39	11
743	42
40	9

Over: 0
Under: 80

670	115
21	29
687	98
27	23
683	102
26	24
668	117
39	11
655	130
35	15
668	117
32	18
669	116
34	16
672	113
32	18
681	104
30	20
681	104
34	15

Over: 100
Under: 20

764	21
39	11
763	22
43	7
758	27
42	8
759	26
46	4
753	32
43	7
754	31
39	11
763	22
46	4
747	38
41	9
764	21
44	6
752	33
41	8

Over: 0
Under: 20

758	27
40	10
760	25
42	8
757	28
43	7
761	24
47	3
750	35
43	7
755	30
40	10
763	22
46	4
752	33
41	9
767	18
45	5
750	35
39	10

Over: 0
Under: 60

739	46
35	15
742	43
34	16
735	50
38	12
723	62
42	8
713	72
40	10
723	62
37	13
729	56
39	11
724	61
36	14
737	48
39	11
718	67
35	14

Over: 100
Under: 0

766	19
41	9
768	17
43	7
762	23
44	6
766	19
47	3
760	25
44	6
761	24
40	10
765	20
46	4
755	30
43	7
769	16
45	5
755	30
41	8

Over: 100
Under: 40

749	36
37	13
759	26
40	10
750	35
43	7
750	35
46	4
744	41
43	7
741	44
38	12
748	37
43	7
739	46
39	11
746	39
43	7
740	45
37	12

Over: 100
Under: 60

725	60
33	17
740	45
36	14
738	47
37	13
729	56
44	6
724	61
39	11
722	63
37	13
728	57
39	11
724	61
33	17
732	53
33	17
723	62
34	15

Over: 200
Under: 0

756	29
37	13
758	27
40	10
750	35
43	7
755	30
45	5
748	37
44	6
747	38
38	12
754	31
45	5
749	36
41	9
757	28
43	7
747	38
38	11

Over: 200
Under: 40

725	60
35	15
738	47
38	12
725	60
36	14
731	54
42	8
720	65
40	10
715	70
37	13
733	52
43	7
706	79
35	15
738	47
39	11
727	58
37	12

Over: 200
Under: 80

580	205
18	32
614	171
26	24
602	183
16	34
619	166
26	24
593	192
30	20
605	180
29	21
584	201
26	24
584	201
21	29
609	176
28	22
598	187
26	23

Over: 100
Under: 80

669	116
29	21
687	98
35	15
688	97
26	24
667	118
32	18
660	125
33	17
651	134
33	17
667	118
34	16
668	117
30	20
685	100
33	17
684	101
34	15

Over: 200
Under: 20

751	34
34	16
757	28
40	10
748	37
41	9
744	41
42	8
736	49
44	6
740	45
37	13
748	37
44	6
732	53
39	11
749	36
39	11
742	43
39	10

Over: 200
Under: 60

693	92
24	26
705	80
33	17
704	81
32	18
705	80
40	10
685	100
38	12
684	101
34	16
696	89
34	16
678	107
34	16
694	91
30	20
697	88
35	14

Over: 300
Under: 0

752	33
37	13
758	27
42	8
751	34
40	10
750	35
44	6
735	50
43	7
746	39
38	12
754	31
45	5
742	43
41	9
754	31
43	7
743	42
38	11

Over: 300
Under: 20

738	47
34	16
744	41
39	11
743	42
36	14
738	47
42	8
726	59
42	8
733	52
38	12
739	46
44	6
736	49
40	10
740	45
41	9
735	50
36	13

Over: 300
Under: 60

680	105
28	22
685	100
31	19
686	99
26	24
682	103
35	15
668	117
36	14
678	107
34	16
686	99
32	18
655	130
36	14
695	90
32	18
688	97
31	18

Over: 400
Under: 0

749	36
34	16
749	36
40	10
747	38
40	10
740	45
42	8
734	51
44	6
744	41
38	12
742	43
44	6
743	42
42	8
752	33
41	9
738	47
39	10

Over: 400
Under: 40

713	72
30	20
726	59
34	16
713	72
32	18
713	72
36	14
702	83
38	12
694	91
36	14
711	74
39	11
706	79
36	14
723	62
34	16
721	64
36	13

Over: 300
Under: 40

721	64
31	19
727	58
33	17
721	64
33	17
714	71
39	11
702	83
40	10
709	76
36	14
716	69
43	7
706	79
40	10
723	62
37	13
709	76
36	13

Over: 300
Under: 80

589	196
20	30
587	198
20	30
600	185
18	32
580	205
28	22
596	189
26	24
564	221
24	26
575	210
21	29
579	206
21	29
581	204
23	27
574	211
23	26

Over: 400
Under: 20

729	56
34	16
739	46
35	15
732	53
34	16
733	52
40	10
715	70
42	8
724	61
37	13
735	50
42	8
725	60
38	12
737	48
40	10
732	53
39	10

Over: 400
Under: 60

650	135
30	20
685	100
31	19
661	124
26	24
654	131
34	16
646	139
34	16
646	139
28	22
658	127
31	19
661	124
29	21
680	105
28	22
660	125
35	14

Over: 400
Under: 80

561	224
16	34
589	196
20	30
576	209
19	31
545	240
25	25
551	234
25	25
563	222
22	28
565	220
22	28
575	210
19	31
580	205
23	27
581	204
28	21

Over: 500
Under: 20

731	54
32	18
725	60
36	14
719	66
32	18
717	68
39	11
715	70
41	9
722	63
37	13
728	57
42	8
720	65
37	13
732	53
35	15
721	64
38	11

Over: 500
Under: 60

657	128
24	26
670	115
33	17
669	116
27	23
653	132
35	15
633	152
37	13
639	146
28	22
637	148
27	23
648	137
28	22
661	124
33	17
652	133
30	19

Over: 500
Under: 0

740	45
33	17
741	44
39	11
745	40
40	10
742	43
42	8
728	57
41	9
732	53
37	13
738	47
44	6
732	53
40	10
748	37
40	10
733	52
39	10

Over: 500
Under: 40

686	99
27	23
713	72
34	16
710	75
28	22
701	84
36	14
684	101
36	14
693	92
35	15
706	79
41	9
695	90
34	16
708	77
34	16
698	87
33	16

Over: 500
Under: 80

572	213
18	32
586	199
25	25
562	223
14	36
559	226
23	27
540	245
22	28
547	238
18	32
550	235
24	26
559	226
23	27
563	222
21	29
578	207
22	27

Classifier: Radial Basis Functions

Over: 0
Under: 0

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 40

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 80

764	21
42	8
785	0
50	0
775	10
48	2
770	15
49	1
771	14
49	1
785	0
50	0
785	0
50	0
785	0
50	0
765	20
47	3
776	9
48	1

Over: 100
Under: 20

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 20

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 0
Under: 60

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 100
Under: 0

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 100
Under: 40

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 100
Under: 60

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 200
Under: 0

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 200
Under: 40

765	20
41	9
785	0
50	0
785	0
50	0
768	17
49	1
785	0
50	0
768	17
49	1
767	18
47	3
772	13
41	9
757	28
46	4
785	0
49	0

Over: 200
Under: 80

544	241
24	26
606	179
32	18
679	106
44	6
725	60
40	10
596	189
30	20
681	104
40	10
532	253
25	25
530	255
20	30
602	183
27	23
711	74
38	11

Over: 100
Under: 80

767	18
42	8
785	0
50	0
774	11
48	2
773	12
49	1
767	18
47	3
785	0
50	0
785	0
50	0
785	0
50	0
768	17
47	3
775	10
47	2

Over: 200
Under: 20

785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
771	14
48	2
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
785	0
49	0

Over: 200
Under: 60

770	15
41	9
785	0
50	0
730	55
46	4
738	47
41	9
740	45
44	6
736	49
46	4
752	33
43	7
773	12
43	7
758	27
48	2
762	23
46	3

Over: 300
Under: 0

785	0
50	0
785	0
50	0
785	0
50	0
780	5
49	1
767	18
47	3
779	6
49	1
785	0
50	0
772	13
42	8
781	4
47	3
775	10
47	2

Over: 300
Under: 20

785	0
50	0
785	0
50	0
785	0
50	0
774	11
49	1
766	19
47	3
785	0
50	0
785	0
50	0
773	12
43	7
785	0
50	0
774	11
47	2

Over: 300
Under: 60

706	79
39	11
631	154
37	13
683	102
43	7
656	129
34	16
700	85
43	7
721	64
45	5
715	70
43	7
730	55
36	14
710	75
44	6
722	63
43	6

Over: 400
Under: 0

785	0
50	0
772	13
48	2
774	11
48	2
775	10
49	1
766	19
47	3
775	10
49	1
770	15
47	3
785	0
50	0
776	9
47	3
785	0
49	0

Over: 400
Under: 40

687	98
35	15
715	70
45	5
742	43
46	4
733	52
40	10
763	22
46	4
726	59
44	6
768	17
47	3
772	13
42	8
742	43
47	3
733	52
44	5

Over: 300
Under: 40

749	36
45	5
767	18
48	2
734	51
46	4
776	9
44	6
769	16
48	2
740	45
47	3
761	24
50	0
772	13
42	8
757	28
49	1
771	14
46	3

Over: 300
Under: 80

721	64
45	5
631	154
37	13
710	75
44	6
683	102
43	7
730	55
36	14
656	129
34	16
700	85
43	7
706	79
39	11
715	70
43	7
722	62
40	10

Over: 400
Under: 20

732	53
39	11
752	33
49	1
771	14
48	2
773	12
49	1
769	16
48	2
738	47
45	5
752	33
44	6
785	0
50	0
764	21
49	1
771	14
47	2

Over: 400
Under: 60

597	188
30	20
640	145
36	14
664	121
40	10
692	93
34	16
606	179
32	18
716	69
44	6
555	230
23	27
674	111
32	18
627	158
35	15
576	209
31	18

Over: 400
Under: 80

555	230
23	27
597	188
30	20
627	158
33	17
716	69
44	6
664	121
19	11
576	209
31	18
692	93
34	16
606	179
32	18
640	145
36	14
674	111
32	18

Over: 500
Under: 20

767	18
41	9
767	18
48	2
756	29
47	3
739	46
40	10
736	49
42	8
739	46
47	3
750	35
44	6
772	13
42	8
756	29
48	2
773	12
45	4

Over: 500
Under: 60

640	145
30	20
547	238
25	25
590	195
30	20
652	133
33	17
654	131
37	13
678	107
42	8
545	240
27	23
594	191
26	24
601	184
31	19
529	256
23	26

Over: 500
Under: 0

785	0
50	0
745	40
47	3
761	24
47	3
775	10
49	1
744	41
44	6
745	40
46	4
768	17
47	3
771	14
42	8
773	12
47	3
775	10
47	2

Over: 500
Under: 40

612	173
31	19
627	158
36	14
718	67
47	3
724	61
39	11
716	69
43	7
719	66
43	7
619	166
35	15
769	16
40	10
669	116
36	14
713	72
42	7

Over: 500
Under: 80

545	240
26	24
590	195
30	20
529	256
22	27
594	191
26	24
678	107
41	9
640	145
29	21
547	238
25	25
601	184
30	20
652	133
33	17
654	131
36	14

Classifier: Random Forest

Over: 0
Under: 0

777	8
47	3
778	7
42	8
776	9
46	4
778	7
48	2
771	14
47	3
776	9
46	4
777	8
50	0
775	10
46	4
775	10
46	4
774	11
44	5

Over: 0
Under: 40

771	14
46	4
766	19
42	8
771	14
43	7
770	15
48	2
762	23
46	4
764	21
45	5
768	17
48	2
770	15
46	4
765	20
45	5
767	18
45	4

Over: 0
Under: 80

707	78
29	21
718	67
35	15
713	72
31	19
686	99
36	14
716	69
41	9
694	91
33	17
691	94
41	9
693	92
34	16
691	94
36	14
710	75
34	15

Over: 100
Under: 20

779	6
45	5
772	13
41	9
774	11
44	6
775	10
47	3
773	12
47	3
778	7
45	5
776	9
50	0
769	16
47	3
767	18
44	6
771	14
46	3

Over: 0
Under: 20

777	8
43	7
776	9
43	7
778	7
44	6
767	18
47	3
771	14
45	5
774	11
46	4
771	14
49	1
778	7
46	4
772	13
45	5
774	11
45	4

Over: 0
Under: 60

768	17
35	15
758	27
39	11
759	26
41	9
750	35
47	3
747	38
42	8
749	36
42	8
747	38
48	2
750	35
45	5
757	28
44	6
757	28
40	9

Over: 100
Under: 0

780	5
47	3
780	5
42	8
776	9
47	3
776	9
49	1
774	11
48	2
774	11
46	4
778	7
48	2
777	8
46	4
777	8
44	6
774	11
47	2

Over: 100
Under: 40

777	8
40	10
764	21
39	11
772	13
44	6
765	20
48	2
766	19
47	3
765	20
42	8
757	28
46	4
769	16
44	6
769	16
45	5
764	21
44	5

Over: 100
Under: 60

766	19
40	10
757	28
39	11
760	25
39	11
763	22
46	4
756	29
45	5
755	30
42	8
747	38
44	6
745	40
43	7
757	28
42	8
751	34
41	8

Over: 200
Under: 0

777	8
42	8
775	10
40	10
773	12
43	7
772	13
49	1
764	21
46	4
772	13
44	6
769	16
47	3
769	16
49	1
769	16
47	3
771	14
43	6

Over: 200
Under: 40

768	17
38	12
752	33
40	10
757	28
37	13
758	27
46	4
738	47
46	4
749	36
40	10
749	36
45	5
755	30
45	5
763	22
42	8
754	31
44	5

Over: 200
Under: 80

637	148
25	25
645	140
29	21
638	147
21	29
655	130
34	16
655	130
31	19
632	153
30	20
628	157
32	18
626	159
30	20
643	142
27	23
634	151
29	20

Over: 100
Under: 80

706	79
34	16
703	82
33	17
710	75
33	17
712	73
36	14
716	69
32	18
689	96
38	12
689	96
33	17
699	86
36	14
705	80
36	14
713	72
34	15

Over: 200
Under: 20

772	13
42	8
776	9
44	6
774	11
42	8
770	15
45	5
758	27
45	5
761	24
45	5
762	23
47	3
764	21
47	3
770	15
44	6
770	15
43	6

Over: 200
Under: 60

719	66
27	23
728	57
38	12
733	52
34	16
722	63
43	7
724	61
43	7
709	76
40	10
710	75
36	14
723	62
40	10
732	53
37	13
730	55
40	9

Over: 300
Under: 0

776	9
43	7
758	27
43	7
776	9
47	3
770	15
46	4
759	26
44	6
771	14
44	6
768	17
47	3
765	20
46	4
767	18
42	8
773	12
44	5

Over: 300
Under: 20

770	15
39	11
759	26
39	11
761	24
40	10
767	18
48	2
743	42
44	6
762	23
43	7
755	30
47	3
760	25
44	6
764	21
42	8
769	16
45	4

Over: 300
Under: 60

723	62
31	19
695	90
33	17
715	70
35	15
722	63
36	14
705	80
38	12
713	72
37	13
720	65
36	14
704	81
36	14
729	56
38	12
716	69
37	12

Over: 400
Under: 0

771	14
46	4
757	28
43	7
772	13
44	6
769	16
47	3
751	34
48	2
758	27
41	9
761	24
45	5
758	27
44	6
769	16
46	4
765	20
44	5

Over: 400
Under: 40

748	37
34	16
752	33
40	10
743	42
40	10
750	35
41	9
729	56
45	5
736	49
43	7
733	52
46	4
748	37
45	5
750	35
41	9
753	32
42	7

Over: 300
Under: 40

753	32
38	12
743	42
40	10
749	36
41	9
746	39
42	8
733	52
45	5
750	35
39	11
737	48
45	5
752	33
44	6
745	40
40	10
752	33
42	7

Over: 300
Under: 80

623	162
19	31
632	153
24	26
630	155
21	29
622	163
29	21
633	152
27	23
623	162
26	24
615	170
27	23
643	142
25	25
632	153
23	27
646	139
29	20

Over: 400
Under: 20

767	18
40	10
753	32
40	10
758	27
43	7
762	23
46	4
741	44
45	5
749	36
43	7
751	34
47	3
756	29
46	4
767	18
41	9
758	27
45	4

Over: 400
Under: 60

716	69
35	15
700	85
36	14
713	72
30	20
704	81
32	18
697	88
39	11
697	88
35	15
702	83
37	13
698	87
39	11
702	83
34	16
709	76
39	10

Over: 400
Under: 80

638	147
21	29
624	161
30	20
619	166
20	30
619	166
32	18
630	155
28	22
624	161
23	27
605	180
29	21
622	163
27	23
643	142
27	23
629	156
25	24

Over: 500
Under: 20

765	20
36	14
746	39
41	9
758	27
41	9
760	25
43	7
742	43
47	3
756	29
41	9
748	37
45	5
756	29
44	6
768	17
39	11
755	30
41	8

Over: 500
Under: 60

699	86
27	23
691	94
36	14
718	67
32	18
704	81
40	10
708	77
38	12
691	94
33	17
684	101
37	13
706	79
36	14
712	73
33	17
707	78
35	14

Over: 500
Under: 0

772	13
38	12
764	21
43	7
762	23
41	9
763	22
45	5
754	31
45	5
755	30
43	7
759	26
47	3
767	18
45	5
771	14
42	8
765	20
45	4

Over: 500
Under: 40

731	54
32	18
741	44
38	12
751	34
34	16
749	36
42	8
726	59
43	7
734	51
40	10
740	45
41	9
722	63
47	3
742	43
37	13
732	53
40	9

Over: 500
Under: 80

615	170
21	29
616	169
25	25
615	170
19	31
622	163
31	19
628	157
26	24
589	196
22	28
604	181
29	21
629	156
26	24
628	157
22	28
634	151
32	17

Classifier: Support Vector Machines

Over: 0
Under: 0

772	13
45	5
778	7
45	5
767	18
44	6
767	18
48	2
771	14
43	7
769	16
45	5
772	13
46	4
770	15
45	5
766	19
45	5
772	13
45	4

Over: 0
Under: 40

766	19
44	6
771	14
45	5
757	28
42	8
750	35
47	3
764	21
42	8
761	24
45	5
755	30
45	5
763	22
46	4
756	29
42	8
762	23
45	4

Over: 0
Under: 80

718	67
36	14
730	55
34	16
716	69
26	24
706	79
35	15
716	69
40	10
705	80
34	16
698	87
38	12
706	79
38	12
718	67
38	12
727	58
41	8

Over: 100
Under: 20

770	15
45	5
774	11
43	7
768	17
42	8
760	25
47	3
768	17
43	7
768	17
45	5
766	19
45	5
764	21
44	6
766	19
44	6
771	14
44	5

Over: 0
Under: 20

769	16
45	5
776	9
43	7
760	25
43	7
756	29
48	2
771	14
44	6
766	19
45	5
772	13
45	5
765	20
42	8
765	20
44	6
772	13
46	3

Over: 0
Under: 60

750	35
42	8
761	24
40	10
749	36
42	8
742	43
43	7
754	31
43	7
749	36
41	9
755	30
42	8
745	40
42	8
745	40
41	9
749	36
44	5

Over: 100
Under: 0

772	13
45	5
778	7
45	5
767	18
44	6
767	18
48	2
771	14
43	7
769	16
45	5
772	13
46	4
770	15
45	5
766	19
45	5
772	13
45	4

Over: 100
Under: 40

764	21
42	8
768	17
40	10
758	27
40	10
752	33
48	2
763	22
44	6
756	29
45	5
754	31
43	7
759	26
44	6
762	23
44	6
761	24
44	5

Over: 100
Under: 60

752	33
41	9
755	30
40	10
759	26
40	10
745	40
45	5
754	31
44	6
753	32
40	10
744	41
44	6
743	42
42	8
749	36
42	8
750	35
42	7

Over: 200
Under: 0

769	16
45	5
763	22
41	9
754	31
40	10
748	37
47	3
757	28
44	6
762	23
42	8
753	32
44	6
758	27
43	7
751	34
40	10
764	21
43	6

Over: 200
Under: 40

746	39
44	6
737	48
39	11
742	43
37	13
735	50
44	6
735	50
41	9
729	56
39	11
733	52
41	9
731	54
41	9
736	49
40	10
741	44
38	11

Over: 200
Under: 80

527	258
8	42
555	230
18	32
516	269
17	33
557	228
19	31
541	244
18	32
519	266
20	30
542	243
15	35
529	256
15	35
520	265
19	31
556	229
17	32

Over: 100
Under: 80

737	48
39	11
727	58
36	14
720	65
33	17
721	64
41	9
724	61
37	13
710	75
36	14
720	65
39	11
714	71
38	12
712	73
37	13
735	50
42	7

Over: 200
Under: 20

756	29
43	7
761	24
39	11
752	33
38	12
740	45
43	7
753	32
43	7
750	35
41	9
750	35
40	10
750	35
43	7
752	33
39	11
760	25
40	9

Over: 200
Under: 60

716	69
30	20
714	71
39	11
696	89
29	21
713	72
38	12
701	84
35	15
681	104
36	14
691	94
37	13
699	86
34	16
709	76
35	15
714	71
37	12

Over: 300
Under: 0

761	24
44	6
760	25
39	11
752	33
38	12
741	44
46	4
751	34
45	5
747	38
43	7
743	42
43	7
746	39
44	6
748	37
40	10
761	24
40	9

Over: 300
Under: 20

750	35
39	11
758	27
40	10
733	52
37	13
731	54
41	9
745	40
41	9
739	46
41	9
724	61
41	9
737	48
41	9
723	62
39	11
744	41
38	11

Over: 300
Under: 60

687	98
31	19
660	125
26	24
674	111
28	22
679	106
33	17
675	110
33	17
653	132
32	18
632	153
24	26
660	125
31	19
680	105
35	15
690	95
35	14

Over: 400
Under: 0

757	28
43	7
760	25
40	10
740	45
38	12
739	46
42	8
744	41
42	8
744	41
40	10
743	42
43	7
739	46
41	9
743	42
39	11
753	32
37	12

Over: 400
Under: 40

717	68
35	15
728	57
36	14
694	91
31	19
708	77
37	13
715	70
39	11
706	79
34	16
682	103
37	13
708	77
37	13
707	78
37	13
722	63
39	10

Over: 300
Under: 40

727	58
34	16
732	53
39	11
726	59
36	14
719	66
40	10
716	69
40	10
708	77
40	10
704	81
43	7
721	64
39	11
714	71
36	14
727	58
40	9

Over: 300
Under: 80

510	275
10	40
522	263
16	34
520	265
17	33
522	263
12	38
522	263
17	33
511	274
13	37
496	289
13	37
513	272
12	38
500	285
18	32
548	237
20	29

Over: 400
Under: 20

737	48
41	9
742	43
40	10
729	56
35	15
727	58
42	8
726	59
40	10
726	59
41	9
723	62
40	10
729	56
41	9
734	51
38	12
738	47
39	10

Over: 400
Under: 60

607	178
19	31
586	199
23	27
598	187
25	25
600	185
20	30
601	184
23	27
599	186
26	24
586	199
23	27
605	180
22	28
580	205
23	27
629	156
33	16

Over: 400
Under: 80

488	297
8	42
515	270
18	32
506	279
15	35
513	272
14	36
510	275
16	34
465	320
13	37
481	304
15	35
511	274
12	38
493	292
15	35
538	247
17	32

Over: 500
Under: 20

729	56
35	15
728	57
37	13
708	77
32	18
719	66
40	10
728	57
41	9
725	60
35	15
702	83
41	9
726	59
38	12
724	61
37	13
725	60
40	9

Over: 500
Under: 60

585	200
12	38
610	175
26	24
568	217
15	35
581	204
23	27
566	219
21	29
582	203
20	30
565	220
18	32
579	206
18	32
572	213
22	28
600	185
27	22

Over: 500
Under: 0

750	35
39	11
746	39
39	11
739	46
41	9
733	52
41	9
745	40
43	7
736	49
38	12
724	61
42	8
727	58
40	10
740	45
40	10
744	41
40	9

Over: 500
Under: 40

669	116
21	29
667	118
33	17
656	129
26	24
672	113
25	25
673	112
35	15
659	126
31	19
652	133
32	18
671	114
26	24
672	113
29	21
676	109
32	17

Over: 500
Under: 80

494	291
6	44
504	281
15	35
483	302
11	39
510	275
9	41
496	289
12	38
482	303
10	40
490	295
14	36
505	280
9	41
475	310
11	39
510	275
16	33

Appendix 2

Chapter 4 raw data

Appendix 2 contains the raw data from the confusion matrices of the experiment results for Chapter 4.

The results for SMOTE are shown first, followed by the results for SMOTE Surrounding Neighbors (SN). Seven datasets were used: CM1, KC1, KC2, JM1, PC1, KC1-Class and Mozilla. For each dataset there are 25 resampling strategies sourced from the combination of five levels for over-sampling (Over: 0, 100, 200, 300, 400) and five levels for under-sampling (Under: 0, 20, 40 , 60, 80). Each resampling strategy has ten confusion matrices corresponding to the 10-fold cross validation procedure. These are visualized as ten boxes with two rows and two columns each.

All confusion matrices are shown in the following format:

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Resampling algorithm: SMOTE

Dataset: CM1

Over: 0
Under: 0

44	0
6	0
44	1
4	1
45	0
5	0
44	1
5	0
43	2
5	0
42	3
4	1
45	0
5	0
44	1
5	0
44	1
3	1
45	0
4	0

Over: 0
Under: 20

44	0
6	0
43	2
4	1
45	0
5	0
44	1
5	0
43	2
5	0
40	5
5	0
44	1
5	0
45	0
5	0
42	3
3	1
45	0
4	0

Over: 0
Under: 40

43	1
3	3
45	0
5	0
43	2
4	1
40	5
5	0
42	3
4	1
36	9
3	2
45	0
5	0
44	1
5	0
42	3
2	2
45	0
4	0

Over: 0
Under: 60

43	1
5	1
40	5
5	0
44	1
5	0
38	7
5	0
37	8
2	3
29	16
1	4
44	1
4	1
45	0
5	0
41	4
4	0
44	1
4	0

Over: 0
Under: 80

26	18
0	6
26	19
1	4
35	10
3	2
33	12
4	1
27	18
2	3
27	18
1	4
41	4
5	0
33	12
4	1
30	15
2	2
42	3
4	0

Over: 100
Under: 20

44	0
5	1
39	6
3	2
42	3
3	2
39	6
5	0
40	5
4	1
37	8
4	1
44	1
4	1
41	4
3	2
39	6
1	3
41	4
3	1

Over: 100
Under: 60

36	8
2	4
36	9
2	3
36	9
3	2
30	15
3	2
37	8
2	3
35	10
3	2
36	9
3	2
41	4
3	2
36	9
3	1
40	5
2	2

Over: 200
Under: 0

37	7
1	5
42	3
5	0
42	3
2	3
41	4
5	0
39	6
3	2
37	8
3	2
44	1
5	0
41	4
3	2
40	5
4	0
42	3
4	0

Over: 100
Under: 0

41	3
6	0
44	1
3	2
43	2
4	1
40	5
5	0
43	2
4	1
41	4
5	0
40	5
4	1
40	5
4	1
42	3
3	1
43	2
4	0

Over: 100
Under: 40

31	13
3	3
38	7
3	2
40	5
4	1
38	7
5	0
39	6
4	1
34	11
4	1
39	6
4	1
40	5
3	2
35	10
1	3
44	1
3	1

Over: 100
Under: 80

27	17
3	3
20	25
2	3
33	12
2	3
27	18
2	3
31	14
2	3
29	16
2	3
25	20
1	4
32	13
3	2
33	12
1	3
15	30
2	2

Over: 200
Under: 20

39	5
4	2
40	5
4	1
39	6
3	2
38	7
3	2
41	4
3	2
37	8
4	1
36	9
3	2
41	4
4	1
39	6
2	2
44	1
4	0

Over: 200
Under: 40

35	9
5	1
34	11
4	1
40	5
4	1
37	8
5	0
37	8
3	2
32	13
3	2
38	7
3	2
41	4
4	1
35	10
2	2
37	8
3	1

Over: 200
Under: 80

24	20
0	6
30	15
3	2
32	13
3	2
18	27
2	3
36	9
3	2
25	20
2	3
32	13
1	4
30	15
2	3
30	15
1	3
13	32
0	4

Over: 300
Under: 20

40	4
3	3
36	9
3	2
43	2
5	0
37	8
5	0
39	6
2	3
33	12
3	2
43	2
3	2
41	4
4	1
37	8
2	2
40	5
2	2

Over: 300
Under: 60

31	13
2	4
35	10
4	1
37	8
3	2
27	18
2	3
37	8
1	4
31	14
3	2
37	8
4	1
38	7
4	1
33	12
2	2
41	4
2	2

Over: 200
Under: 60

40	4
3	3
37	8
1	4
34	11
2	3
39	6
4	1
37	8
2	3
31	14
1	4
36	9
2	3
37	8
2	3
30	15
2	2
42	3
4	0

Over: 300
Under: 0

38	6
2	4
39	6
4	1
42	3
3	2
38	7
5	0
40	5
2	3
38	7
4	1
43	2
4	1
41	4
4	1
38	7
3	1
40	5
2	2

Over: 300
Under: 40

37	7
2	4
38	7
5	0
43	2
5	0
38	7
4	1
35	10
1	4
32	13
2	3
37	8
4	1
39	6
3	2
36	9
3	1
36	9
3	1

Over: 300
Under: 80

26	18
0	6
30	15
0	5
32	13
2	3
39	6
4	1
26	19
0	5
25	20
2	3
35	10
2	3
41	4
4	1
26	19
1	3
35	10
2	2

Over: 400
Under: 0

40	4
0	6
41	4
3	2
43	2
3	2
40	5
4	1
40	5
2	3
37	8
4	1
42	3
4	1
40	5
4	1
42	3
3	1
44	1
2	2

Over: 400
Under: 40

35	9
2	4
37	8
3	2
40	5
3	2
37	8
5	0
40	5
3	2
36	9
4	1
40	5
3	2
39	6
4	1
33	12
3	1
39	6
2	2

Over: 400
Under: 80

29	15
1	5
32	13
0	5
31	14
3	2
23	22
2	3
30	15
2	3
11	34
1	4
32	13
2	3
33	12
2	3
30	15
2	2
26	19
0	4

Over: 400
Under: 20

38	6
3	3
36	9
4	1
39	6
4	1
36	9
4	1
38	7
2	3
35	10
4	1
39	6
3	2
38	7
4	1
40	5
3	1
43	2
3	1

Over: 400
Under: 60

31	13
1	5
35	10
2	3
36	9
4	1
36	9
5	0
37	8
1	4
32	13
2	3
35	10
3	2
39	6
3	2
33	12
3	1
30	15
0	4

Dataset: KC1

Over: 0
Under: 0

166	12
24	9
167	11
19	14
163	15
23	10
174	4
26	7
163	15
19	14
171	7
21	12
164	14
20	13
168	11
24	8
176	3
25	7
168	11
22	9

Over: 0
Under: 40

163	15
21	12
148	30
15	18
154	24
24	9
169	9
21	12
151	27
13	20
161	17
24	9
151	27
18	15
159	20
19	13
167	12
19	13
169	10
18	13

Over: 0
Under: 80

125	53
13	20
123	55
7	26
129	49
13	20
130	48
9	24
133	45
8	25
112	66
10	23
133	45
12	21
130	49
12	20
136	43
6	26
111	68
5	26

Over: 100
Under: 20

164	14
19	14
157	21
17	16
158	20
24	9
170	8
18	15
170	8
28	5
168	10
20	13
150	28
14	19
169	10
26	6
171	8
18	14
163	16
15	16

Over: 0
Under: 20

165	13
21	12
150	28
14	19
158	20
22	11
166	12
19	14
169	9
18	15
163	15
23	10
162	16
21	12
168	11
24	8
170	9
20	12
165	14
20	11

Over: 0
Under: 60

164	14
18	15
147	31
12	21
148	30
14	19
156	22
19	14
161	17
17	16
149	29
20	13
146	32
19	14
153	26
15	17
158	21
14	18
161	18
18	13

Over: 100
Under: 0

160	18
22	11
154	24
15	18
155	23
19	14
167	11
17	16
163	15
22	11
161	17
19	14
166	12
22	11
165	14
26	6
164	15
16	16
168	11
17	14

Over: 100
Under: 40

157	21
21	12
153	25
16	17
150	28
19	14
148	30
10	23
148	30
10	23
152	26
16	17
156	22
17	16
160	19
18	14
162	17
10	22
146	33
12	19

Over: 100
Under: 60

155	23
16	17
122	56
6	27
110	68
7	26
141	37
12	21
128	50
6	27
136	42
13	20
132	46
15	18
153	26
16	16
153	26
13	19
152	27
12	19

Over: 200
Under: 0

162	16
25	8
155	23
18	15
165	13
25	8
167	11
14	19
158	20
16	17
165	13
21	12
152	26
15	18
163	16
21	11
168	11
22	10
165	14
15	16

Over: 200
Under: 40

153	25
16	17
141	37
13	20
138	40
19	14
156	22
11	22
150	28
12	21
151	27
16	17
149	29
16	17
156	23
19	13
153	26
7	25
142	37
8	23

Over: 200
Under: 80

87	91
4	29
100	78
2	31
94	84
5	28
105	73
5	28
103	75
3	30
119	59
9	24
123	55
8	25
108	71
7	25
130	49
8	24
114	65
6	25

Over: 100
Under: 80

134	44
7	26
86	92
3	30
106	72
8	25
119	59
7	26
106	72
2	31
96	82
9	24
106	72
5	28
137	42
11	21
106	73
1	31
122	57
6	25

Over: 200
Under: 20

157	21
18	15
152	26
16	17
158	20
26	7
154	24
12	21
159	19
17	16
159	19
24	9
161	17
14	19
158	21
21	11
165	14
15	17
165	14
16	15

Over: 200
Under: 60

146	32
10	23
124	54
7	26
139	39
13	20
143	35
11	22
127	51
10	23
142	36
16	17
128	50
14	19
151	28
15	17
151	28
11	21
143	36
11	20

Over: 300
Under: 0

165	13
25	8
147	31
17	16
158	20
25	8
171	7
16	17
161	17
22	11
160	18
21	12
153	25
16	17
158	21
20	12
163	16
13	19
166	13
16	15

Over: 300
Under: 20

156	22
24	9
145	33
11	22
134	44
17	16
162	16
15	18
145	33
12	21
166	12
20	13
148	30
14	19
156	23
18	14
160	19
13	19
151	28
14	17

Over: 300
Under: 60

142	36
10	23
113	65
6	27
124	54
17	16
143	35
11	22
142	36
7	26
143	35
14	19
138	40
9	24
126	53
19	13
131	48
7	25
149	30
13	18

Over: 400
Under: 0

159	19
20	13
155	23
17	16
156	22
26	7
165	13
15	18
156	22
12	21
162	16
16	17
153	25
15	18
156	23
21	11
158	21
11	21
161	18
17	14

Over: 400
Under: 40

163	15
18	15
138	40
15	18
149	29
18	15
157	21
11	22
141	37
10	23
143	35
16	17
142	36
10	23
148	31
15	17
144	35
7	25
148	31
10	21

Over: 300
Under: 40

140	38
16	17
136	42
10	23
141	37
13	20
151	27
15	18
141	37
11	22
142	36
14	19
145	33
10	23
152	27
20	12
154	25
11	21
145	34
11	20

Over: 300
Under: 80

123	55
7	26
92	86
6	27
85	93
4	29
104	74
6	27
106	72
3	30
112	66
7	26
89	89
5	28
120	59
10	22
101	78
5	27
93	86
4	27

Over: 400
Under: 20

167	11
21	12
153	25
16	17
158	20
17	16
157	21
13	20
154	24
16	17
157	21
15	18
141	37
14	19
155	24
17	15
164	15
10	22
156	23
13	18

Over: 400
Under: 60

142	36
15	18
122	56
9	24
122	56
14	19
138	40
10	23
134	44
10	23
137	41
11	22
134	44
10	23
143	36
14	18
139	40
5	27
145	34
7	24

Over: 400
Under: 80

94	84
3	30
76	102
2	31
111	67
8	25
116	62
6	27
97	81
2	31
89	89
6	27
109	69
9	24
121	58
9	23
104	75
1	31
111	68
5	26

Dataset: KC2

Over: 0
Under: 0

40	3
6	4
39	4
5	5
39	3
6	4
39	2
2	9
36	5
8	3
39	2
7	4
35	6
5	6
35	6
7	4
38	3
5	6
41	0
9	2

Over: 0
Under: 40

35	8
2	8
36	7
6	4
37	5
5	5
36	5
4	7
39	2
4	7
35	6
2	9
33	8
4	7
34	7
3	8
33	8
3	8
34	7
4	7

Over: 0
Under: 80

35	8
1	9
36	7
2	8
35	7
5	5
35	6
1	10
35	6
3	8
34	7
3	8
28	13
2	9
25	16
4	7
33	8
1	10
35	6
5	6

Over: 100
Under: 20

33	10
2	8
37	6
4	6
37	5
5	5
37	4
4	7
33	8
3	8
38	3
6	5
33	8
5	6
30	11
4	7
32	9
1	10
34	7
5	6

Over: 0
Under: 20

37	6
5	5
39	4
5	5
41	1
5	5
33	8
4	7
34	7
4	7
36	5
6	5
39	2
7	4
33	8
4	7
38	3
3	8
38	3
9	2

Over: 0
Under: 60

34	9
2	8
35	8
6	4
37	5
5	5
33	8
2	9
35	6
4	7
31	10
2	9
32	9
3	8
29	12
5	6
36	5
2	9
32	9
3	8

Over: 100
Under: 0

36	7
3	7
39	4
4	6
40	2
6	4
39	2
4	7
38	3
4	7
39	2
6	5
34	7
5	6
31	10
3	8
34	7
1	10
37	4
8	3

Over: 100
Under: 40

32	11
0	10
35	8
2	8
37	5
3	7
35	6
4	7
37	4
3	8
36	5
3	8
31	10
3	8
34	7
3	8
30	11
4	7
33	8
6	5

Over: 100
Under: 60

36	7
0	10
29	14
2	8
34	8
2	8
31	10
0	11
32	9
3	8
32	9
0	11
26	15
2	9
33	8
4	7
31	10
1	10
32	9
3	8

Over: 200
Under: 0

38	5
8	2
37	6
5	5
36	6
2	8
33	8
3	8
39	2
8	3
38	3
6	5
35	6
5	6
30	11
5	6
37	4
3	8
34	7
3	8

Over: 200
Under: 40

33	10
2	8
31	12
4	6
32	10
2	8
30	11
0	11
32	9
5	6
34	7
4	7
36	5
4	7
32	9
4	7
34	7
1	10
33	8
6	5

Over: 200
Under: 80

25	18
1	9
27	16
2	8
27	15
1	9
27	14
0	11
30	11
3	8
31	10
3	8
23	18
3	8
24	17
4	7
22	19
1	10
26	15
3	8

Over: 100
Under: 80

30	13
1	9
32	11
4	6
29	13
1	9
27	14
0	11
31	10
4	7
21	20
2	9
28	13
2	9
24	17
3	8
23	18
1	10
15	26
1	10

Over: 200
Under: 20

35	8
6	4
36	7
5	5
32	10
3	7
33	8
0	11
36	5
5	6
36	5
3	8
33	8
6	5
32	9
4	7
35	6
2	9
32	9
4	7

Over: 200
Under: 60

28	15
1	9
35	8
5	5
34	8
4	6
28	13
0	11
32	9
3	8
38	3
2	9
29	12
3	8
28	13
4	7
28	13
1	10
31	10
2	9

Over: 300
Under: 0

34	9
3	7
37	6
5	5
39	3
4	6
32	9
4	7
38	3
8	3
38	3
6	5
35	6
5	6
35	6
7	4
36	5
3	8
34	7
3	8

Over: 300
Under: 20

33	10
1	9
34	9
5	5
35	7
4	6
29	12
2	9
36	5
5	6
35	6
2	9
32	9
4	7
31	10
5	6
35	6
5	6
34	7
3	8

Over: 300
Under: 60

32	11
1	9
32	11
4	6
33	9
5	5
24	17
1	10
33	8
3	8
34	7
4	7
29	12
4	7
30	11
5	6
26	15
1	10
31	10
3	8

Over: 400
Under: 0

32	11
4	6
36	7
4	6
39	3
4	6
30	11
4	7
37	4
8	3
37	4
5	6
36	5
6	5
31	10
3	8
35	6
4	7
33	8
3	8

Over: 400
Under: 40

29	14
1	9
33	10
5	5
34	8
3	7
29	12
1	10
37	4
5	6
37	4
4	7
32	9
4	7
28	13
4	7
33	8
2	9
33	8
2	9

Over: 300
Under: 40

37	6
3	7
34	9
4	6
36	6
4	6
32	9
0	11
35	6
3	8
34	7
4	7
28	13
3	8
29	12
4	7
34	7
3	8
28	13
3	8

Over: 300
Under: 80

24	19
0	10
32	11
2	8
24	18
0	10
22	19
1	10
28	13
2	9
31	10
2	9
20	21
2	9
23	18
3	8
25	16
1	10
34	7
3	8

Over: 400
Under: 20

34	9
3	7
39	4
6	4
34	8
2	8
28	13
3	8
34	7
6	5
36	5
5	6
31	10
6	5
30	11
5	6
36	5
2	9
35	6
6	5

Over: 400
Under: 60

28	15
0	10
32	11
3	7
35	7
1	9
25	16
0	11
31	10
3	8
32	9
3	8
28	13
4	7
29	12
4	7
30	11
1	10
29	12
3	8

Over: 400
Under: 80

15 0	28 10
32 2	11 8
21 0	21 10
20 1	21 10
25 3	16 8
30 0	11 11
14 2	27 9
22 3	19 8
21 2	20 9
20 3	21 8

Dataset: JM1

Over: 0
Under: 0

816	61
160	51
806	71
158	53
795	82
143	68
820	58
161	49
841	37
177	33
825	53
162	48
825	53
156	54
821	57
146	64
815	63
163	47
821	57
173	37

Over: 0
Under: 40

801	76
157	54
730	147
121	90
738	139
114	97
751	127
127	83
780	98
143	67
726	152
119	91
811	67
153	57
733	145
115	95
719	159
117	93
788	90
154	56

Over: 0
Under: 80

492	385
45	166
568	309
73	138
512	365
41	170
496	382
60	150
524	354
54	156
524	354
67	143
484	394
45	165
554	324
51	159
541	337
75	135
480	398
55	155

Over: 100
Under: 20

765	112
133	78
761	116
126	85
744	133
119	92
737	141
116	94
731	147
139	71
713	165
115	95
739	139
117	93
723	155
109	101
718	160
119	91
754	124
138	72

Over: 0
Under: 20

809	68
150	61
777	100
132	79
784	93
136	75
793	85
149	61
811	67
152	58
790	88
149	61
789	89
126	84
824	54
154	56
780	98
139	71
788	90
156	54

Over: 0
Under: 60

784	93
138	73
650	227
86	125
687	190
98	113
711	167
119	91
757	121
144	66
667	211
108	102
709	169
94	116
717	161
107	103
673	205
89	121
696	182
120	90

Over: 100
Under: 0

801	76
136	75
768	109
129	82
732	145
108	103
800	78
158	52
780	98
143	67
771	107
135	75
802	76
163	47
806	72
140	70
789	89
131	79
754	124
146	64

Over: 100
Under: 40

710	167
98	113
682	195
95	116
666	211
84	127
692	186
118	92
729	149
120	90
677	201
110	100
718	160
100	110
740	138
115	95
729	149
104	106
681	197
107	103

Over: 100
Under: 60

652	225
89	122
593	284
86	125
637	240
73	138
605	273
81	129
651	227
95	115
591	287
82	128
595	283
74	136
584	294
59	151
604	274
93	117
613	265
80	130

Over: 200
Under: 0

776	101
124	87
760	117
126	85
731	146
121	90
748	130
137	73
749	129
133	77
755	123
132	78
775	103
135	75
763	115
128	82
774	104
141	69
745	133
123	87

Over: 200
Under: 40

687	190
100	111
649	228
91	120
653	224
77	134
695	183
104	106
678	200
110	100
658	220
106	104
657	221
79	131
684	194
90	120
655	223
95	115
658	220
96	114

Over: 200
Under: 80

380	497
40	171
334	543
19	192
445	432
32	179
447	431
48	162
415	463
46	164
424	454
44	166
427	451
40	170
463	415
36	174
379	499
37	173
467	411
46	164

Over: 100
Under: 80

344	533
32	179
385	492
45	166
393	484
31	180
449	429
46	164
405	473
29	181
440	438
58	152
421	457
36	174
488	390
43	167
402	476
42	168
451	427
43	167

Over: 200
Under: 20

703	174
101	110
718	159
120	91
719	158
114	97
742	136
127	83
721	157
125	85
725	153
114	96
738	140
128	82
733	145
111	99
728	150
114	96
716	162
122	88

Over: 200
Under: 60

593	284
70	141
564	313
62	149
594	283
72	139
574	304
92	118
581	297
78	132
571	307
90	120
572	306
68	142
585	293
69	141
577	301
77	133
582	296
74	136

Over: 300
Under: 0

768	109
136	75
734	143
113	98
718	159
105	106
720	158
123	87
782	96
140	70
741	137
126	84
744	134
134	76
757	121
121	89
739	139
124	86
734	144
127	83

Over: 300
Under: 20

684	193
109	102
696	181
108	103
715	162
114	97
719	159
116	94
686	192
112	98
712	166
121	89
717	161
117	93
738	140
128	82
693	185
100	110
703	175
119	91

Over: 300
Under: 60

578	299
78	133
576	301
68	143
551	326
61	150
544	334
77	133
592	286
84	126
549	329
72	138
578	300
76	134
569	309
59	151
593	285
76	134
585	293
87	123

Over: 400
Under: 0

738	139
120	91
733	144
111	100
729	148
106	105
729	149
131	79
730	148
120	90
713	165
128	82
764	114
127	83
738	140
130	80
719	159
126	84
776	102
126	84

Over: 400
Under: 40

657	220
102	109
641	236
94	117
635	242
76	135
622	256
88	122
669	209
91	119
632	246
96	114
625	253
79	131
666	212
103	107
643	235
99	111
637	241
106	104

Over: 300
Under: 40

662	215
95	116
648	229
98	113
671	206
96	115
650	228
101	109
637	241
90	120
623	255
92	118
652	226
96	114
691	187
110	100
677	201
99	111
653	225
104	106

Over: 300
Under: 80

408	469
39	172
365	512
33	178
435	442
37	174
449	429
49	161
447	431
36	174
436	442
55	155
371	507
39	171
352	526
22	188
422	456
44	166
430	448
50	160

Over: 400
Under: 20

712	165
113	98
707	170
114	97
721	156
106	105
705	173
120	90
665	213
97	113
721	157
118	92
704	174
115	95
724	154
108	102
705	173
112	98
695	183
108	102

Over: 400
Under: 60

578	299
64	147
530	347
65	146
578	299
64	147
563	315
85	125
570	308
76	134
552	326
75	135
543	335
63	147
552	326
66	144
542	336
63	147
569	309
72	138

Over: 400
Under: 80

387	490
37	174
434	443
51	160
418	459
27	184
404	474
46	164
394	484
41	169
433	445
51	159
422	456
37	173
350	528
32	178
399	479
52	158
411	467
39	171

Dataset: PC1

Over: 0
Under: 0

102	1
6	2
98	5
5	3
101	2
5	3
100	3
4	4
101	2
7	1
102	1
6	2
97	6
6	2
101	2
7	1
104	0
6	1
103	1
5	1

Over: 0
Under: 40

100	3
5	3
97	6
4	4
97	6
3	5
103	0
3	5
98	5
6	2
102	1
6	2
97	6
7	1
100	3
5	3
95	9
6	1
101	3
4	2

Over: 0
Under: 80

97	6
4	4
95	8
4	4
71	32
1	7
90	13
3	5
95	8
6	2
74	29
2	6
89	14
6	2
86	17
3	5
93	11
4	3
85	19
4	2

Over: 100
Under: 20

96	7
6	2
98	5
5	3
96	7
4	4
97	6
2	6
101	2
6	2
100	3
4	4
98	5
7	1
98	5
4	4
98	6
5	2
98	6
4	2

Over: 0
Under: 20

103	0
6	2
102	1
6	2
99	4
6	2
100	3
3	5
101	2
6	2
102	1
6	2
98	5
7	1
103	0
7	1
102	2
6	1
104	0
5	1

Over: 0
Under: 60

103	0
5	3
97	6
6	2
94	9
2	6
91	12
3	5
96	7
2	6
100	3
4	4
101	2
7	1
89	14
5	3
94	10
5	2
101	3
3	3

Over: 100
Under: 0

101	2
5	3
100	3
4	4
96	7
5	3
100	3
4	4
101	2
7	1
100	3
5	3
98	5
7	1
96	7
7	1
99	5
6	1
100	4
4	2

Over: 100
Under: 40

97	6
5	3
93	10
5	3
89	14
2	6
93	10
4	4
98	5
3	5
99	4
4	4
95	8
6	2
95	8
3	5
91	13
5	2
99	5
5	1

Over: 100
Under: 60

93	10
3	5
79	24
2	6
86	17
3	5
94	9
4	4
90	13
6	2
92	11
4	4
92	11
6	2
94	9
4	4
89	15
4	3
96	8
5	1

Over: 200
Under: 0

100	3
5	3
94	9
6	2
93	10
4	4
94	9
3	5
98	5
6	2
101	2
6	2
96	7
7	1
99	4
7	1
101	3
7	0
98	6
4	2

Over: 200
Under: 40

99	4
4	4
95	8
4	4
88	15
4	4
94	9
1	7
95	8
4	4
95	8
4	4
88	15
5	3
98	5
7	1
89	15
3	4
95	9
5	1

Over: 200
Under: 80

75	28
1	7
89	14
4	4
77	26
0	8
76	27
2	6
86	17
4	4
81	22
4	4
79	24
4	4
64	39
1	7
85	19
2	5
85	19
2	4

Over: 100
Under: 80

89	14
2	6
79	24
2	6
84	19
2	6
81	22
5	3
89	14
4	4
79	24
1	7
86	17
4	4
76	27
2	6
82	22
5	2
92	12
3	3

Over: 200
Under: 20

101	2
5	3
101	2
6	2
97	6
3	5
98	5
4	4
98	5
6	2
100	3
4	4
96	7
5	3
99	4
6	2
102	2
6	1
97	7
5	1

Over: 200
Under: 60

94	9
4	4
95	8
2	6
87	16
2	6
83	20
3	5
92	11
4	4
92	11
3	5
88	15
6	2
92	11
4	4
91	13
5	2
93	11
3	3

Over: 300
Under: 0

100	3
5	3
99	4
6	2
96	7
4	4
91	12
3	5
99	4
6	2
98	5
5	3
96	7
7	1
97	6
5	3
99	5
5	2
97	7
4	2

Over: 300
Under: 20

94	9
5	3
93	10
6	2
92	11
5	3
83	20
5	3
95	8
5	3
99	4
3	5
95	8
7	1
98	5
4	4
99	5
6	1
94	10
3	3

Over: 300
Under: 60

88	15
3	5
89	14
5	3
94	9
4	4
88	15
2	6
91	12
2	6
85	18
2	6
88	15
5	3
90	13
3	5
93	11
2	5
90	14
3	3

Over: 400
Under: 0

98	5
4	4
97	6
6	2
95	8
4	4
96	7
3	5
99	4
6	2
97	6
5	3
95	8
7	1
99	4
6	2
99	5
5	2
97	7
5	1

Over: 400
Under: 40

94	9
4	4
90	13
6	2
87	16
2	6
92	11
3	5
96	7
4	4
95	8
4	4
97	6
6	2
95	8
3	5
96	8
6	1
94	10
3	3

Over: 300
Under: 40

93	10
3	5
101	2
3	5
94	9
2	6
96	7
2	6
97	6
6	2
96	7
3	5
97	6
7	1
92	11
5	3
94	10
5	2
100	4
5	1

Over: 300
Under: 80

82	21
4	4
75	28
2	6
79	24
0	8
79	24
2	6
88	15
4	4
78	25
1	7
87	16
3	5
86	17
1	7
74	30
2	5
91	13
5	1

Over: 400
Under: 20

97	6
4	4
99	4
6	2
90	13
4	4
96	7
4	4
100	3
6	2
92	11
4	4
95	8
6	2
90	13
4	4
101	3
6	1
97	7
3	3

Over: 400
Under: 60

93	10
4	4
79	24
5	3
79	24
2	6
90	13
1	7
88	15
5	3
84	19
1	7
89	14
5	3
83	20
3	5
97	7
3	4
93	11
2	4

Over: 400
Under: 80

81 2	22 6
83 1	20 7
74 0	29 8
82 0	21 8
84 2	19 6
72 2	31 6
80 4	23 4
82 4	21 4
86 4	18 3
87 4	17 2

Dataset: KC1-Class

Over: 0
Under: 0

7	1
4	3
7	1
1	6
7	1
3	4
5	3
5	2
7	1
2	5
5	4
2	3
9	0
2	3
6	3
1	4
6	3
1	4
6	3
4	1

Over: 0
Under: 40

7	1
1	6
7	1
2	5
5	3
2	5
6	2
4	3
7	1
1	6
5	4
2	3
6	3
0	5
5	4
2	3
4	5
0	5
7	2
1	4

Over: 0
Under: 80

4	4
0	7
7	1
1	6
3	5
0	7
3	5
1	6
2	6
0	7
3	6
0	5
4	5
0	5
4	5
0	5
4	5
0	5
6	3
0	5

Over: 100
Under: 20

7	1
1	6
7	1
1	6
3	5
3	4
5	3
1	6
8	0
3	4
4	5
1	4
8	1
2	3
5	4
1	4
6	3
2	3
5	4
1	4

Over: 0
Under: 20

7	1
1	6
7	1
1	6
8	0
3	4
5	3
5	2
4	4
2	5
4	5
1	4
6	3
1	4
6	3
1	4
5	4
0	5
7	2
0	5

Over: 0
Under: 60

6	2
2	5
6	2
1	6
7	1
4	3
4	4
1	6
4	4
2	5
4	5
2	3
7	2
1	4
6	3
0	5
5	4
1	4
6	3
2	3

Over: 100
Under: 0

7	1
1	6
7	1
3	4
8	0
3	4
7	1
1	6
6	2
2	5
4	5
2	3
7	2
3	2
4	5
1	4
5	4
0	5
6	3
1	4

Over: 100
Under: 40

3	5
0	7
7	1
2	5
6	2
4	3
5	3
6	1
5	3
3	4
5	4
1	4
6	3
2	3
4	5
0	5
5	4
0	5
7	2
1	4

Over: 100
Under: 60

7	1
2	5
7	1
0	7
8	0
2	5
2	6
3	4
4	4
1	6
3	6
0	5
3	6
0	5
4	5
0	5
5	4
0	5
6	3
1	4

Over: 200
Under: 0

6	2
2	5
7	1
3	4
5	3
2	5
5	3
3	4
6	2
3	4
5	4
1	4
7	2
3	2
5	4
1	4
7	2
0	5
5	4
1	4

Over: 200
Under: 40

7	1
1	6
7	1
2	5
6	2
2	5
3	5
2	5
5	3
3	4
7	2
1	4
6	3
1	4
5	4
1	4
5	4
0	5
6	3
2	3

Over: 200
Under: 80

3	5
0	7
4	4
0	7
4	4
0	7
2	6
0	7
4	4
1	6
3	6
0	5
3	6
0	5
4	5
0	5
4	5
0	5
5	4
0	5

Over: 100
Under: 80

3	5
0	7
4	4
0	7
4	4
0	7
4	4
0	7
4	4
1	6
3	6
1	4
4	5
1	4
3	6
0	5
5	4
0	5
7	2
0	5

Over: 200
Under: 20

7	1
1	6
7	1
3	4
5	3
2	5
4	4
1	6
7	1
2	5
4	5
1	4
7	2
1	4
3	6
0	5
8	1
1	4
7	2
1	4

Over: 200
Under: 60

3	5
0	7
7	1
2	5
7	1
2	5
3	5
0	7
5	3
1	6
4	5
1	4
1	8
0	5
5	4
1	4
4	5
0	5
6	3
1	4

Over: 300
Under: 0

6	2
1	6
7	1
3	4
6	2
2	5
5	3
1	6
6	2
3	4
5	4
1	4
6	3
2	3
6	3
1	4
7	2
1	4
7	2
2	3

Over: 300
Under: 20

6	2
1	6
7	1
3	4
7	1
3	4
3	5
2	5
6	2
2	5
4	5
1	4
7	2
1	4
6	3
1	4
6	3
0	5
5	4
1	4

Over: 300
Under: 60

4	4
1	6
8	0
0	7
4	4
0	7
3	5
0	7
6	2
2	5
4	5
1	4
3	6
1	4
4	5
0	5
5	4
0	5
6	3
1	4

Over: 400
Under: 0

5	3
1	6
7	1
3	4
7	1
3	4
4	4
2	5
7	1
2	5
7	2
1	4
6	3
1	4
6	3
1	4
8	1
1	4
5	4
1	4

Over: 400
Under: 40

4	4
0	7
6	2
2	5
7	1
2	5
4	4
2	5
6	2
3	4
4	5
0	5
5	4
0	5
3	6
1	4
7	2
1	4
5	4
1	4

Over: 300
Under: 40

3	5
0	7
6	2
0	7
4	4
1	6
3	5
0	7
5	3
4	3
6	3
3	2
4	5
0	5
6	3
0	5
6	3
0	5
7	2
0	5

Over: 300
Under: 80

3	5
0	7
5	3
1	6
4	4
0	7
3	5
1	6
4	4
1	6
3	6
0	5
1	8
0	5
3	6
0	5
5	4
1	4
5	4
0	5

Over: 400
Under: 20

5	3
2	5
7	1
2	5
6	2
6	1
3	5
2	5
5	3
2	5
5	4
1	4
5	4
1	4
4	5
1	4
6	3
1	4
6	3
0	5

Over: 400
Under: 60

3	5
0	7
5	3
0	7
8	0
3	4
4	4
0	7
7	1
1	6
4	5
1	4
4	5
0	5
3	6
0	5
4	5
0	5
5	4
1	4

Over: 400
Under: 80

3	5
0	7
4	4
0	7
3	5
0	7
2	6
0	7
5	3
2	5
4	5
1	4
1	8
0	5
5	4
0	5
4	5
0	5
5	4
1	4

Dataset: Mozilla

Over: 0
Under: 0

785	0
50	0
785	0
50	0
784	1
49	1
784	1
49	1
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
784	1
50	0
785	0
49	0

Over: 0
Under: 40

784	1
46	4
785	0
48	2
782	3
49	1
784	1
49	1
785	0
49	1
784	1
50	0
785	0
50	0
782	3
49	1
782	3
48	2
783	2
48	1

Over: 0
Under: 80

745	40
42	8
685	100
33	17
752	33
45	5
720	65
42	8
726	59
42	8
686	99
40	10
744	41
45	5
746	39
43	7
748	37
46	4
727	58
38	11

Over: 100
Under: 20

765	20
40	10
764	21
47	3
767	18
46	4
751	34
43	7
769	16
46	4
741	44
44	6
760	25
43	7
754	31
46	4
766	19
43	7
762	23
47	2

Over: 0
Under: 20

784	1
46	4
785	0
50	0
782	3
49	1
784	1
49	1
785	0
50	0
784	1
50	0
785	0
50	0
784	1
50	0
785	0
50	0
785	0
48	1

Over: 0
Under: 60

784	1
48	2
761	24
46	4
781	4
49	1
768	17
46	4
785	0
48	2
760	25
46	4
770	15
48	2
763	22
47	3
784	1
49	1
768	17
44	5

Over: 100
Under: 0

768	17
42	8
772	13
43	7
776	9
47	3
753	32
44	6
769	16
48	2
765	20
46	4
771	14
48	2
771	14
49	1
769	16
48	2
778	7
48	1

Over: 100
Under: 40

768	17
41	9
746	39
38	12
761	24
44	6
751	34
44	6
739	46
45	5
739	46
45	5
735	50
47	3
765	20
48	2
762	23
44	6
762	23
46	3

Over: 100
Under: 60

741	44
37	13
712	73
35	15
727	58
38	12
719	66
44	6
713	72
35	15
675	110
38	12
707	78
33	17
712	73
37	13
731	54
42	8
742	43
37	12

Over: 200
Under: 0

773	12
44	6
751	34
43	7
763	22
45	5
760	25
47	3
759	26
46	4
761	24
42	8
766	19
48	2
774	11
45	5
761	24
43	7
766	19
45	4

Over: 200
Under: 40

727	58
37	13
749	36
43	7
738	47
41	9
723	62
39	11
718	67
38	12
736	49
37	13
751	34
47	3
736	49
44	6
750	35
41	9
739	46
38	11

Over: 200
Under: 80

613	172
24	26
570	215
23	27
483	302
9	41
570	215
21	29
619	166
30	20
570	215
27	23
460	325
13	37
563	222
22	28
579	206
18	32
574	211
22	27

Over: 100
Under: 80

601	184
19	31
631	154
27	23
624	161
28	22
637	148
31	19
608	177
24	26
565	220
21	29
633	152
34	16
596	189
29	21
589	196
22	28
641	144
32	17

Over: 200
Under: 20

756	29
43	7
752	33
45	5
760	25
42	8
742	43
45	5
755	30
46	4
740	45
43	7
743	42
39	11
764	21
40	10
746	39
45	5
766	19
45	4

Over: 200
Under: 60

693	92
34	16
715	70
42	8
694	91
37	13
731	54
45	5
679	106
36	14
699	86
36	14
673	112
35	15
720	65
38	12
712	73
33	17
727	58
35	14

Over: 300
Under: 0

762	23
41	9
758	27
40	10
763	22
46	4
741	44
43	7
739	46
45	5
753	32
43	7
752	33
47	3
764	21
48	2
764	21
46	4
756	29
44	5

Over: 300
Under: 20

755	30
31	19
734	51
42	8
757	28
43	7
728	57
42	8
743	42
43	7
736	49
40	10
746	39
45	5
751	34
47	3
747	38
41	9
749	36
43	6

Over: 300
Under: 60

655	130
26	24
664	121
34	16
639	146
25	25
650	135
33	17
683	102
35	15
656	129
31	19
716	69
39	11
644	141
30	20
647	138
32	18
660	125
32	17

Over: 400
Under: 0

762	23
40	10
748	37
43	7
763	22
45	5
742	43
43	7
751	34
43	7
738	47
40	10
760	25
46	4
738	47
39	11
760	25
43	7
758	27
43	6

Over: 400
Under: 40

734	51
32	18
723	62
38	12
721	64
37	13
697	88
37	13
703	82
41	9
714	71
44	6
705	80
41	9
690	95
37	13
746	39
38	12
740	45
41	8

Over: 300
Under: 40

726	59
34	16
719	66
38	12
732	53
42	8
716	69
35	15
716	69
40	10
720	65
46	4
733	52
45	5
723	62
41	9
722	63
33	17
732	53
40	9

Over: 300
Under: 80

519	266
10	40
546	239
23	27
455	330
10	40
585	200
30	20
564	221
24	26
532	253
21	29
554	231
20	30
495	290
21	29
505	280
15	35
553	232
22	27

Over: 400
Under: 20

728	57
30	20
726	59
40	10
736	49
39	11
729	56
38	12
753	32
41	9
727	58
41	9
731	54
44	6
741	44
41	9
757	28
41	9
748	37
43	6

Over: 400
Under: 60

622	163
16	34
641	144
33	17
692	93
28	22
648	137
34	16
657	128
32	18
696	89
38	12
608	177
34	16
656	129
32	18
624	161
27	23
637	148
32	17

Over: 400
Under: 80

491	294
5	45
547	238
19	31
528	257
12	38
541	244
21	29
531	254
22	28
549	236
25	25
520	265
26	24
538	247
25	25
557	228
17	33
576	209
18	31

SMOTE Surrounding Neighbors (SN)

Dataset: CM1

Over: 0
Under: 0

44	0
6	0
44	1
4	1
45	0
5	0
44	1
5	0
43	2
5	0
42	3
4	1
45	0
5	0
44	1
5	0
44	1
3	1
45	0
4	0

Over: 0
Under: 20

44	0
6	0
45	0
5	0
45	0
5	0
42	3
5	0
42	3
4	1
42	3
5	0
45	0
5	0
44	1
5	0
43	2
4	0
44	1
4	0

Over: 0
Under: 40

44	0
6	0
45	0
5	0
45	0
5	0
43	2
5	0
43	2
5	0
38	7
3	2
45	0
5	0
45	0
5	0
43	2
2	2
45	0
4	0

Over: 0
Under: 60

44	0
6	0
45	0
5	0
45	0
5	0
43	2
5	0
37	8
4	1
43	2
4	1
45	0
5	0
39	6
3	2
45	0
4	0
42	3
2	2

Over: 0
Under: 80

32	12
1	5
25	20
1	4
45	0
5	0
28	17
3	2
36	9
3	2
30	15
3	2
30	15
1	4
36	9
3	2
29	16
0	4
42	3
1	3

Over: 100
Under: 20

36	8
2	4
41	4
3	2
43	2
4	1
38	7
5	0
39	6
4	1
37	8
5	0
40	5
4	1
43	2
4	1
40	5
2	2
41	4
4	0

Over: 100
Under: 60

34	10
2	4
41	4
4	1
39	6
5	0
39	6
4	1
34	11
2	3
31	14
3	2
38	7
4	1
34	11
2	3
37	8
3	1
36	9
3	1

Over: 200
Under: 0

39	5
3	3
43	2
3	2
42	3
5	0
39	6
5	0
40	5
3	2
39	6
4	1
44	1
5	0
44	1
4	1
36	9
3	1
42	3
2	2

Over: 100
Under: 0

38	6
3	3
41	4
3	2
44	1
5	0
37	8
4	1
37	8
5	0
37	8
4	1
44	1
4	1
44	1
5	0
40	5
2	2
43	2
4	0

Over: 100
Under: 40

40	4
3	3
36	9
4	1
42	3
3	2
37	8
4	1
35	10
3	2
35	10
3	2
39	6
3	2
40	5
4	1
37	8
3	1
38	7
2	2

Over: 100
Under: 80

30	14
1	5
34	11
0	5
28	17
2	3
34	11
3	2
31	14
0	5
24	21
2	3
32	13
2	3
36	9
3	2
27	18
1	3
26	19
2	2

Over: 200
Under: 20

35	9
1	5
41	4
3	2
39	6
4	1
39	6
4	1
36	9
3	2
36	9
3	2
43	2
3	2
41	4
4	1
39	6
3	1
38	7
2	2

Over: 200
Under: 40

37	7
4	2
40	5
3	2
38	7
3	2
37	8
5	0
37	8
2	3
35	10
2	3
41	4
3	2
41	4
4	1
32	13
3	1
39	6
4	0

Over: 200
Under: 80

27	17
2	4
36	9
1	4
26	19
2	3
33	12
2	3
36	9
2	3
24	21
0	5
31	14
1	4
33	12
1	4
28	17
2	2
26	19
2	2

Over: 300
Under: 20

33	11
1	5
38	7
3	2
41	4
3	2
38	7
4	1
40	5
4	1
38	7
2	3
42	3
4	1
42	3
4	1
38	7
3	1
43	2
4	0

Over: 300
Under: 60

39	5
3	3
30	15
2	3
33	12
3	2
37	8
3	2
36	9
4	1
32	13
4	1
34	11
3	2
41	4
4	1
31	14
4	0
43	2
3	1

Over: 200
Under: 60

36	8
0	6
39	6
2	3
34	11
4	1
32	13
4	1
36	9
1	4
30	15
3	2
33	12
2	3
36	9
3	2
35	10
1	3
34	11
4	0

Over: 300
Under: 0

39	5
2	4
39	6
3	2
40	5
3	2
41	4
4	1
38	7
3	2
35	10
4	1
40	5
5	0
39	6
4	1
36	9
3	1
40	5
2	2

Over: 300
Under: 40

35	9
2	4
34	11
2	3
35	10
4	1
38	7
4	1
40	5
1	4
39	6
4	1
41	4
5	0
38	7
5	0
35	10
3	1
40	5
2	2

Over: 300
Under: 80

30	14
1	5
27	18
2	3
33	12
1	4
33	12
3	2
26	19
1	4
25	20
0	5
35	10
3	2
25	20
1	4
26	19
0	4
24	21
0	4

Over: 400
Under: 0

38	6
4	2
36	9
3	2
40	5
3	2
36	9
5	0
39	6
3	2
39	6
4	1
41	4
3	2
42	3
5	0
39	6
3	1
40	5
3	1

Over: 400
Under: 40

37	7
3	3
38	7
2	3
40	5
3	2
37	8
5	0
37	8
2	3
32	13
3	2
37	8
2	3
39	6
3	2
31	14
2	2
38	7
3	1

Over: 400
Under: 80

27	17
3	3
24	21
0	5
28	17
2	3
25	20
2	3
27	18
0	5
17	28
0	5
25	20
0	5
35	10
2	3
26	19
2	2
33	12
0	4

Over: 400
Under: 20

36	8
3	3
44	1
3	2
40	5
3	2
36	9
5	0
38	7
4	1
34	11
3	2
42	3
5	0
40	5
3	2
38	7
3	1
39	6
3	1

Over: 400
Under: 60

33	11
2	4
37	8
0	5
37	8
2	3
36	9
5	0
38	7
3	2
34	11
3	2
35	10
4	1
39	6
4	1
32	13
4	0
34	11
2	2

Dataset: KC1

Over: 0
Under: 0

166	12
24	9
167	11
19	14
163	15
23	10
174	4
26	7
163	15
19	14
171	7
21	12
164	14
20	13
168	11
24	8
176	3
25	7
168	11
22	9

Over: 0
Under: 40

162	16
24	9
148	30
15	18
159	19
22	11
162	16
18	15
152	26
19	14
152	26
22	11
153	25
21	12
164	15
21	11
160	19
16	16
160	19
16	15

Over: 0
Under: 80

136	42
12	21
119	59
7	26
128	50
12	21
137	41
9	24
131	47
4	29
136	42
17	16
158	20
17	16
132	47
13	19
121	58
7	25
149	30
12	19

Over: 100
Under: 20

154	24
19	14
146	32
15	18
149	29
20	13
163	15
12	21
146	32
14	19
149	29
21	12
151	27
21	12
156	23
18	14
158	21
8	24
157	22
17	14

Over: 0
Under: 20

167	11
24	9
155	23
17	16
165	13
26	7
173	5
28	5
164	14
15	18
163	15
22	11
168	10
20	13
169	10
26	6
172	7
23	9
172	7
21	10

Over: 0
Under: 60

156	22
18	15
123	55
12	21
147	31
18	15
154	24
14	19
143	35
14	19
151	27
16	17
154	24
20	13
157	22
19	13
169	10
19	13
145	34
16	15

Over: 100
Under: 0

165	13
21	12
155	23
16	17
149	29
20	13
163	15
17	16
155	23
19	14
164	14
20	13
158	20
17	16
162	17
20	12
163	16
18	14
158	21
16	15

Over: 100
Under: 40

157	21
20	13
140	38
12	21
145	33
15	18
152	26
16	17
142	36
13	20
149	29
20	13
151	27
16	17
161	18
21	11
162	17
14	18
151	28
14	17

Over: 100
Under: 60

146	32
14	19
119	59
7	26
137	41
18	15
145	33
14	19
134	44
9	24
150	28
17	16
144	34
14	19
145	34
19	13
146	33
12	20
140	39
10	21

Over: 200
Under: 0

165	13
20	13
154	24
16	17
154	24
22	11
158	20
16	17
160	18
19	14
162	16
24	9
168	10
18	15
159	20
19	13
168	11
17	15
161	18
13	18

Over: 200
Under: 40

150	28
18	15
143	35
13	20
144	34
21	12
154	24
13	20
151	27
14	19
156	22
16	17
153	25
15	18
152	27
20	12
160	19
15	17
141	38
11	20

Over: 200
Under: 80

111	67
7	26
98	80
3	30
101	77
7	26
124	54
9	24
119	59
5	28
105	73
10	23
126	52
12	21
136	43
12	20
97	82
0	32
127	52
4	27

Over: 100
Under: 80

125	53
11	22
110	68
6	27
121	57
10	23
118	60
9	24
119	59
6	27
125	53
11	22
119	59
8	25
100	79
9	23
117	62
13	19
102	77
2	29

Over: 200
Under: 20

167	11
21	12
146	32
14	19
152	26
20	13
152	26
11	22
146	32
15	18
163	15
20	13
156	22
15	18
159	20
23	9
159	20
14	18
157	22
16	15

Over: 200
Under: 60

143	35
17	16
133	45
16	17
139	39
18	15
138	40
6	27
133	45
12	21
138	40
16	17
136	42
10	23
147	32
17	15
148	31
12	20
133	46
5	26

Over: 300
Under: 0

165	13
20	13
151	27
15	18
151	27
20	13
164	14
16	17
157	21
20	13
162	16
18	15
153	25
14	19
165	14
24	8
172	7
15	17
163	16
13	18

Over: 300
Under: 20

165	13
20	13
142	36
13	20
156	22
23	10
161	17
14	19
150	28
16	17
153	25
18	15
151	27
17	16
145	34
17	15
165	14
11	21
155	24
15	16

Over: 300
Under: 60

142	36
14	19
137	41
11	22
134	44
14	19
151	27
10	23
121	57
4	29
137	41
13	20
141	37
12	21
147	32
15	17
140	39
5	27
135	44
11	20

Over: 400
Under: 0

164	14
21	12
148	30
12	21
146	32
20	13
164	14
15	18
161	17
13	20
167	11
21	12
161	17
20	13
158	21
20	12
167	12
17	15
163	16
16	15

Over: 400
Under: 40

143	35
15	18
133	45
9	24
139	39
17	16
156	22
11	22
134	44
12	21
146	32
16	17
155	23
12	21
148	31
18	14
156	23
9	23
150	29
11	20

Over: 300
Under: 40

152	26
16	17
138	40
9	24
143	35
17	16
150	28
10	23
147	31
12	21
151	27
13	20
147	31
17	16
144	35
17	15
146	33
10	22
150	29
11	20

Over: 300
Under: 80

101	77
6	27
105	73
7	26
104	74
7	26
114	64
6	27
93	85
3	30
113	65
10	23
120	58
9	24
123	56
9	23
116	63
5	27
124	55
5	26

Over: 400
Under: 20

158	20
17	16
140	38
16	17
150	28
19	14
161	17
21	12
155	23
12	21
152	26
18	15
156	22
15	18
159	20
16	16
166	13
15	17
160	19
16	15

Over: 400
Under: 60

140	38
14	19
131	47
11	22
140	38
16	17
128	50
10	23
137	41
11	22
125	53
10	23
130	48
8	25
146	33
14	18
126	53
4	28
141	38
12	19

Over: 400
Under: 80

130	48
10	23
98	80
4	29
112	66
9	24
123	55
7	26
110	68
6	27
117	61
11	22
117	61
7	26
89	90
6	26
108	71
5	27
103	76
7	24

Dataset: KC2

Over: 0
Under: 0

40	3
6	4
39	4
5	5
39	3
6	4
39	2
2	9
36	5
8	3
39	2
7	4
35	6
5	6
35	6
7	4
38	3
5	6
41	0
9	2

Over: 0
Under: 40

35	8
4	6
37	6
7	3
39	3
6	4
35	6
3	8
34	7
5	6
36	5
3	8
37	4
4	7
32	9
4	7
34	7
1	10
36	5
6	5

Over: 0
Under: 80

31	12
0	10
34	9
1	9
29	13
2	8
28	13
2	9
33	8
3	8
32	9
4	7
28	13
2	9
25	16
3	8
28	13
2	9
33	8
3	8

Over: 100
Under: 20

37	6
6	4
36	7
5	5
37	5
4	6
29	12
4	7
33	8
7	4
34	7
2	9
29	12
3	8
33	8
5	6
33	8
1	10
37	4
6	5

Over: 0
Under: 20

38	5
7	3
38	5
4	6
39	3
6	4
35	6
4	7
38	3
7	4
39	2
6	5
34	7
4	7
30	11
4	7
36	5
1	10
40	1
9	2

Over: 0
Under: 60

33	10
2	8
37	6
3	7
39	3
4	6
29	12
2	9
38	3
7	4
34	7
3	8
35	6
4	7
31	10
4	7
33	8
1	10
33	8
3	8

Over: 100
Under: 0

36	7
6	4
39	4
6	4
38	4
3	7
33	8
5	6
35	6
7	4
39	2
5	6
36	5
5	6
34	7
6	5
35	6
4	7
35	6
5	6

Over: 100
Under: 40

31	12
1	9
38	5
6	4
34	8
4	6
30	11
2	9
36	5
7	4
38	3
4	7
36	5
6	5
33	8
4	7
32	9
2	9
37	4
3	8

Over: 100
Under: 60

37	6
2	8
37	6
4	6
38	4
4	6
31	10
3	8
32	9
3	8
35	6
3	8
35	6
6	5
33	8
4	7
30	11
3	8
37	4
4	7

Over: 200
Under: 0

35	8
1	9
38	5
6	4
37	5
3	7
34	7
5	6
35	6
6	5
36	5
3	8
35	6
4	7
30	11
4	7
34	7
5	6
34	7
3	8

Over: 200
Under: 40

34	9
3	7
35	8
5	5
35	7
4	6
32	9
0	11
34	7
6	5
35	6
2	9
29	12
4	7
32	9
5	6
28	13
2	9
31	10
3	8

Over: 200
Under: 80

24	19
0	10
26	17
3	7
26	16
2	8
27	14
0	11
28	13
4	7
24	17
2	9
25	16
3	8
28	13
3	8
19	22
2	9
25	16
1	10

Over: 100
Under: 80

29	14
2	8
34	9
4	6
25	17
0	10
28	13
2	9
31	10
3	8
28	13
1	10
27	14
2	9
25	16
3	8
29	12
2	9
21	20
1	10

Over: 200
Under: 20

38	5
5	5
36	7
5	5
38	4
6	4
30	11
1	10
37	4
5	6
36	5
5	6
36	5
7	4
30	11
4	7
33	8
5	6
37	4
5	6

Over: 200
Under: 60

29	14
1	9
36	7
4	6
33	9
3	7
28	13
0	11
33	8
5	6
32	9
2	9
31	10
3	8
34	7
4	7
30	11
3	8
28	13
3	8

Over: 300
Under: 0

36	7
5	5
35	8
5	5
35	7
5	5
33	8
2	9
38	3
7	4
35	6
3	8
35	6
5	6
31	10
5	6
31	10
1	10
34	7
4	7

Over: 300
Under: 20

35	8
4	6
37	6
7	3
30	12
1	9
32	9
2	9
38	3
9	2
36	5
3	8
36	5
5	6
31	10
4	7
35	6
4	7
33	8
4	7

Over: 300
Under: 60

26	17
0	10
29	14
2	8
34	8
2	8
31	10
0	11
33	8
4	7
29	12
4	7
27	14
3	8
27	14
5	6
27	14
1	10
29	12
6	5

Over: 400
Under: 0

37	6
5	5
36	7
5	5
33	9
3	7
31	10
1	10
38	3
6	5
37	4
3	8
34	7
5	6
29	12
5	6
33	8
4	7
36	5
6	5

Over: 400
Under: 40

33	10
2	8
38	5
3	7
34	8
3	7
30	11
1	10
32	9
5	6
33	8
4	7
31	10
5	6
29	12
4	7
33	8
1	10
32	9
4	7

Over: 300
Under: 40

29	14
3	7
38	5
5	5
31	11
4	6
30	11
1	10
34	7
3	8
31	10
5	6
32	9
3	8
26	15
3	8
31	10
1	10
30	11
4	7

Over: 300
Under: 80

26	17
0	10
22	21
0	10
20	22
0	10
24	17
0	11
28	13
2	9
30	11
2	9
22	19
2	9
24	17
5	6
15	26
1	10
24	17
1	10

Over: 400
Under: 20

35	8
4	6
36	7
5	5
31	11
1	9
30	11
1	10
33	8
3	8
37	4
5	6
33	8
5	6
28	13
4	7
34	7
1	10
30	11
2	9

Over: 400
Under: 60

29	14
1	9
35	8
3	7
27	15
1	9
30	11
1	10
36	5
3	8
33	8
3	8
29	12
2	9
28	13
3	8
25	16
2	9
23	18
2	9

Over: 400
Under: 80

27	16
1	9
29	14
1	9
30	12
3	7
22	19
0	11
23	18
3	8
30	11
0	11
19	22
2	9
25	16
4	7
21	20
1	10
25	16
3	8

Dataset: JM1

Over: 0
Under: 0

816	61
160	51
806	71
158	53
795	82
143	68
820	58
161	49
841	37
177	33
825	53
162	48
825	53
156	54
821	57
146	64
815	63
163	47
821	57
173	37

Over: 0
Under: 40

798	79
151	60
772	105
130	81
750	127
127	84
781	97
158	52
793	85
142	68
735	143
132	78
779	99
121	89
755	123
114	96
720	158
119	91
755	123
146	64

Over: 0
Under: 80

553	324
65	146
499	378
56	155
562	315
55	156
526	352
81	129
546	332
60	150
484	394
60	150
533	345
55	155
530	348
51	159
497	381
59	151
565	313
74	136

Over: 100
Under: 20

730	147
119	92
672	205
109	102
696	181
98	113
679	199
108	102
682	196
108	102
685	193
105	105
698	180
98	112
687	191
106	104
674	204
116	94
695	183
120	90

Over: 0
Under: 20

785	92
147	64
801	76
145	66
785	92
143	68
807	71
156	54
795	83
145	65
796	82
150	60
781	97
135	75
810	68
144	66
781	97
149	61
808	70
172	38

Over: 0
Under: 60

695	182
106	105
656	221
88	123
619	258
69	142
695	183
117	93
666	212
90	120
672	206
99	111
693	185
89	121
706	172
101	109
667	211
99	111
664	214
113	97

Over: 100
Under: 0

750	127
132	79
739	138
117	94
765	112
131	80
731	147
120	90
756	122
128	82
715	163
102	108
730	148
131	79
738	140
128	82
756	122
130	80
717	161
131	79

Over: 100
Under: 40

679	198
93	118
629	248
94	117
635	242
87	124
628	250
95	115
659	219
97	113
640	238
97	113
608	270
82	128
671	207
106	104
631	247
89	121
656	222
92	118

Over: 100
Under: 60

599	278
79	132
552	325
80	131
565	312
71	140
570	308
80	130
608	270
86	124
579	299
80	130
562	316
74	136
612	266
77	133
564	314
73	137
576	302
91	119

Over: 200
Under: 0

737	140
120	91
756	121
134	77
767	110
129	82
765	113
137	73
763	115
138	72
738	140
126	84
766	112
128	82
758	120
139	71
766	112
130	80
764	114
131	79

Over: 200
Under: 40

650	227
94	117
649	228
83	128
694	183
97	114
637	241
100	110
679	199
113	97
683	195
103	107
663	215
89	121
682	196
85	125
692	186
106	104
663	215
100	110

Over: 200
Under: 80

422	455
50	161
419	458
41	170
427	450
29	182
434	444
54	156
414	464
42	168
418	460
43	167
407	471
40	170
398	480
37	173
411	467
46	164
434	444
39	171

Over: 100
Under: 80

410	467
43	168
420	457
58	153
408	469
39	172
407	471
48	162
417	461
33	177
421	457
49	161
410	468
33	177
389	489
42	168
442	436
51	159
427	451
37	173

Over: 200
Under: 20

713	164
110	101
700	177
111	100
719	158
101	110
745	133
121	89
706	172
107	103
712	166
113	97
713	165
103	107
736	142
122	88
716	162
123	87
709	169
129	81

Over: 200
Under: 60

595	282
78	133
562	315
69	142
573	304
69	142
592	286
86	124
628	250
96	114
587	291
85	125
595	283
87	123
586	292
59	151
582	296
74	136
622	256
94	116

Over: 300
Under: 0

772	105
132	79
734	143
113	98
733	144
111	100
760	118
136	74
750	128
126	84
739	139
124	86
760	118
150	60
749	129
129	81
737	141
132	78
752	126
127	83

Over: 300
Under: 20

737	140
119	92
715	162
111	100
714	163
114	97
713	165
117	93
700	178
115	95
685	193
100	110
729	149
110	100
704	174
102	108
720	158
118	92
706	172
128	82

Over: 300
Under: 60

601	276
72	139
578	299
73	138
541	336
58	153
528	350
68	142
596	282
83	127
577	301
79	131
560	318
71	139
569	309
75	135
544	334
60	150
552	326
76	134

Over: 400
Under: 0

762	115
132	79
725	152
116	95
723	154
113	98
740	138
127	83
768	110
142	68
759	119
135	75
758	120
138	72
762	116
125	85
702	176
131	79
727	151
114	96

Over: 400
Under: 40

644	233
85	126
633	244
80	131
644	233
81	130
624	254
96	114
615	263
86	124
666	212
100	110
625	253
90	120
639	239
90	120
652	226
94	116
651	227
100	110

Over: 300
Under: 40

693	184
101	110
638	239
83	128
666	211
87	124
674	204
108	102
660	218
94	116
626	252
88	122
667	211
99	111
663	215
98	112
698	180
113	97
653	225
92	118

Over: 300
Under: 80

400	477
46	165
388	489
45	166
442	435
39	172
470	408
58	152
402	476
36	174
305	573
20	190
395	483
34	176
360	518
20	190
413	465
39	171
372	506
35	175

Over: 400
Under: 20

696	181
114	97
694	183
105	106
727	150
121	90
680	198
108	102
708	170
109	101
688	190
111	99
701	177
111	99
709	169
102	108
706	172
106	104
735	143
139	71

Over: 400
Under: 60

590	287
70	141
549	328
64	147
567	310
58	153
525	353
83	127
552	326
69	141
544	334
72	138
548	330
71	139
564	314
60	150
513	365
59	151
575	303
68	142

Over: 400
Under: 80

391	486
38	173
403	474
33	178
395	482
30	181
376	502
40	170
397	481
34	176
382	496
37	173
437	441
46	164
375	503
26	184
404	474
43	167
420	458
34	176

Dataset: PC1

Over: 0
Under: 0

102	1
6	2
98	5
5	3
101	2
5	3
100	3
4	4
101	2
7	1
102	1
6	2
97	6
6	2
101	2
7	1
104	0
6	1
103	1
5	1

Over: 0
Under: 40

102	1
6	2
98	5
4	4
96	7
3	5
98	5
5	3
100	3
5	3
99	4
4	4
98	5
7	1
102	1
8	0
101	3
6	1
96	8
5	1

Over: 0
Under: 80

98	5
6	2
84	19
0	8
85	18
0	8
79	24
1	7
91	12
3	5
79	24
1	7
98	5
6	2
90	13
6	2
81	23
3	4
95	9
4	2

Over: 100
Under: 20

98	5
6	2
91	12
6	2
91	12
1	7
92	11
5	3
96	7
4	4
98	5
6	2
97	6
8	0
96	7
3	5
96	8
3	4
101	3
4	2

Over: 0
Under: 20

101	2
6	2
99	4
5	3
101	2
5	3
101	2
4	4
101	2
7	1
102	1
6	2
99	4
5	3
100	3
6	2
101	3
7	0
104	0
5	1

Over: 0
Under: 60

98	5
5	3
92	11
4	4
93	10
3	5
89	14
3	5
95	8
6	2
96	7
2	6
93	10
5	3
97	6
6	2
93	11
4	3
100	4
4	2

Over: 100
Under: 0

101	2
6	2
98	5
5	3
96	7
4	4
96	7
2	6
99	4
4	4
99	4
4	4
97	6
7	1
97	6
6	2
99	5
7	0
98	6
4	2

Over: 100
Under: 40

100	3
4	4
93	10
5	3
91	12
3	5
95	8
4	4
98	5
5	3
97	6
3	5
95	8
7	1
96	7
4	4
94	10
6	1
92	12
5	1

Over: 100
Under: 60

98	5
4	4
89	14
5	3
85	18
1	7
85	18
4	4
97	6
6	2
91	12
1	7
97	6
5	3
90	13
2	6
95	9
5	2
91	13
4	2

Over: 200
Under: 0

102	1
5	3
98	5
5	3
98	5
5	3
97	6
4	4
99	4
5	3
99	4
4	4
96	7
6	2
100	3
7	1
99	5
7	0
98	6
4	2

Over: 200
Under: 40

96	7
4	4
98	5
6	2
93	10
4	4
92	11
3	5
98	5
5	3
96	7
6	2
94	9
6	2
91	12
4	4
96	8
6	1
97	7
4	2

Over: 200
Under:80

76	27
1	7
58	45
0	8
76	27
1	7
74	29
1	7
91	12
6	2
80	23
2	6
91	12
5	3
78	25
3	5
77	27
1	6
84	20
3	3

Over: 100
Under: 80

86	17
2	6
72	31
2	6
81	22
2	6
81	22
2	6
83	20
4	4
85	18
1	7
82	21
5	3
84	19
2	6
85	19
3	4
85	19
2	4

Over: 200
Under: 20

102	1
5	3
95	8
5	3
95	8
5	3
95	8
3	5
98	5
6	2
96	7
3	5
93	10
6	2
96	7
6	2
99	5
7	0
99	5
4	2

Over: 200
Under: 60

92	11
4	4
93	10
3	5
89	14
1	7
90	13
2	6
91	12
3	5
91	12
3	5
93	10
5	3
95	8
6	2
94	10
6	1
91	13
5	1

Over: 300
Under: 0

96	7
5	3
100	3
6	2
97	6
4	4
95	8
5	3
100	3
6	2
100	3
1	7
97	6
7	1
102	1
7	1
101	3
7	0
99	5
4	2

Over: 300
Under: 20

101	2
4	4
96	7
6	2
95	8
5	3
93	10
4	4
98	5
5	3
99	4
6	2
95	8
7	1
98	5
4	4
98	6
6	1
101	3
5	1

Over: 300
Under: 60

95	8
3	5
88	15
1	7
86	17
3	5
86	17
2	6
93	10
6	2
92	11
2	6
89	14
4	4
89	14
4	4
96	8
6	1
92	12
4	2

Over: 400
Under: 0

101	2
5	3
97	6
6	2
96	7
5	3
92	11
4	4
98	5
3	5
99	4
5	3
98	5
8	0
100	3
6	2
100	4
5	2
102	2
5	1

Over: 400
Under: 40

100	3
4	4
97	6
4	4
96	7
4	4
86	17
3	5
96	7
4	4
94	9
4	4
91	12
8	0
89	14
3	5
100	4
7	0
96	8
4	2

Over: 300
Under: 40

97	6
4	4
95	8
5	3
94	9
4	4
99	4
3	5
98	5
5	3
98	5
4	4
95	8
7	1
86	17
3	5
102	2
7	0
93	11
5	1

Over: 300
Under: 80

84	19
1	7
61	42
0	8
61	42
0	8
68	35
2	6
84	19
5	3
85	18
2	6
88	15
6	2
93	10
6	2
88	16
4	3
79	25
3	3

Over: 400
Under: 20

102	1
4	4
95	8
6	2
97	6
5	3
96	7
2	6
99	4
6	2
101	2
4	4
95	8
7	1
96	7
4	4
97	7
6	1
96	8
4	2

Over: 400
Under: 60

90	13
3	5
94	9
4	4
79	24
1	7
91	12
3	5
94	9
6	2
90	13
3	5
90	13
4	4
92	11
3	5
95	9
4	3
90	14
6	0

Over: 400
Under: 80

85 2	18 6
76 3	27 5
76 1	27 7
67 1	36 7
85 3	18 5
84 1	19 7
84 5	19 3
83 4	20 4
67 0	37 7
89 4	15 2

Dataset: KC1-Class

Over: 0
Under: 0

7	1
4	3
7	1
1	6
7	1
3	4
5	3
5	2
7	1
2	5
5	4
2	3
9	0
2	3
6	3
1	4
6	3
1	4
6	3
4	1

Over: 0
Under: 40

7	1
4	3
7	1
0	7
6	2
3	4
4	4
4	3
6	2
1	6
2	7
1	4
6	3
2	3
6	3
1	4
8	1
1	4
6	3
0	5

Over: 0
Under: 80

3	5
0	7
6	2
0	7
2	6
0	7
2	6
1	6
3	5
0	7
4	5
1	4
6	3
1	4
4	5
0	5
6	3
0	5
5	4
0	5

Over: 100
Under: 20

6	2
2	5
7	1
0	7
4	4
3	4
5	3
4	3
8	0
2	5
6	3
2	3
6	3
2	3
7	2
1	4
6	3
1	4
6	3
2	3

Over: 0
Under: 20

7	1
4	3
8	0
2	5
5	3
2	5
4	4
4	3
5	3
1	6
5	4
1	4
9	0
2	3
6	3
1	4
5	4
0	5
6	3
4	1

Over: 0
Under: 60

7	1
2	5
7	1
1	6
3	5
4	3
5	3
3	4
4	4
1	6
2	7
0	5
6	3
1	4
5	4
1	4
5	4
1	4
7	2
2	3

Over: 100
Under: 0

5	3
2	5
7	1
2	5
7	1
3	4
7	1
6	1
5	3
1	6
6	3
1	4
8	1
1	4
5	4
0	5
5	4
1	4
6	3
4	1

Over: 100
Under: 40

5	3
2	5
6	2
0	7
8	0
3	4
5	3
4	3
4	4
2	5
5	4
0	5
6	3
1	4
4	5
0	5
6	3
1	4
6	3
1	4

Over: 100
Under: 60

4	4
1	6
6	2
0	7
6	2
2	5
3	5
4	3
4	4
1	6
5	4
1	4
7	2
1	4
6	3
1	4
6	3
1	4
6	3
1	4

Over: 200
Under: 0

6	2
1	6
7	1
1	6
8	0
3	4
5	3
2	5
8	0
2	5
5	4
1	4
6	3
3	2
5	4
0	5
6	3
0	5
7	2
2	3

Over: 200
Under: 40

6	2
1	6
5	3
1	6
4	4
2	5
4	4
3	4
3	5
1	6
4	5
1	4
5	4
3	2
4	5
0	5
5	4
0	5
6	3
0	5

Over: 200
Under: 80

3	5
0	7
6	2
0	7
3	5
0	7
3	5
0	7
4	4
1	6
3	6
0	5
3	6
0	5
3	6
0	5
4	5
0	5
3	6
0	5

Over: 100
Under: 80

3	5
0	7
7	1
0	7
3	5
0	7
4	4
1	6
3	5
0	7
3	6
0	5
3	6
0	5
4	5
0	5
4	5
0	5
5	4
0	5

Over: 200
Under: 20

7	1
1	6
7	1
0	7
8	0
3	4
3	5
3	4
6	2
2	5
5	4
1	4
5	4
0	5
4	5
0	5
7	2
0	5
6	3
2	3

Over: 200
Under: 60

3	5
0	7
7	1
0	7
4	4
0	7
5	3
1	6
4	4
1	6
4	5
1	4
3	6
0	5
4	5
0	5
4	5
0	5
6	3
0	5

Over: 300
Under: 0

5	3
1	6
7	1
1	6
5	3
3	4
4	4
2	5
7	1
3	4
5	4
2	3
7	2
1	4
4	5
1	4
6	3
0	5
7	2
2	3

Over: 300
Under: 20

7	1
1	6
7	1
3	4
7	1
1	6
3	5
2	5
5	3
2	5
4	5
1	4
4	5
2	3
6	3
1	4
6	3
1	4
6	3
1	4

Over: 300
Under: 60

3	5
0	7
6	2
1	6
3	5
1	6
5	3
4	3
4	4
1	6
4	5
1	4
2	7
0	5
3	6
0	5
4	5
0	5
5	4
1	4

Over: 400
Under: 0

5	3
2	5
7	1
3	4
7	1
4	3
5	3
0	7
7	1
2	5
7	2
1	4
6	3
1	4
6	3
1	4
6	3
0	5
7	2
2	3

Over: 400
Under: 40

5	3
1	6
5	3
3	4
3	5
2	5
4	4
0	7
4	4
1	6
4	5
1	4
4	5
0	5
4	5
2	3
6	3
0	5
5	4
0	5

Over: 300
Under: 40

3	5
0	7
7	1
0	7
5	3
2	5
3	5
1	6
3	5
2	5
5	4
1	4
4	5
1	4
5	4
1	4
6	3
0	5
6	3
0	5

Over: 300
Under: 80

3	5
0	7
7	1
0	7
4	4
0	7
1	7
1	6
4	4
2	5
4	5
1	4
2	7
0	5
3	6
0	5
3	6
0	5
6	3
1	4

Over: 400
Under: 20

7	1
1	6
7	1
1	6
6	2
3	4
4	4
2	5
7	1
2	5
5	4
1	4
5	4
1	4
7	2
1	4
6	3
0	5
7	2
0	5

Over: 400
Under: 60

3	5
0	7
6	2
0	7
4	4
1	6
3	5
0	7
3	5
1	6
5	4
1	4
5	4
1	4
5	4
0	5
5	4
0	5
6	3
1	4

Over: 400
Under: 80

1	7
0	7
5	3
0	7
1	7
0	7
2	6
1	6
2	6
0	7
4	5
0	5
4	5
1	4
1	8
0	5
4	5
0	5
4	5
0	5

Dataset: Mozilla

Over: 0
Under: 0

785	0
50	0
785	0
50	0
784	1
49	1
784	1
49	1
785	0
50	0
785	0
50	0
785	0
50	0
785	0
50	0
784	1
50	0
785	0
49	0

Over: 0
Under: 40

780	5
48	2
782	3
48	2
785	0
50	0
782	3
48	2
780	5
49	1
771	14
47	3
781	4
49	1
783	2
48	2
783	2
48	2
783	2
48	1

Over: 0
Under: 80

753	32
45	5
701	84
41	9
718	67
36	14
701	84
39	11
729	56
41	9
727	58
43	7
756	29
45	5
743	42
43	7
722	63
39	11
715	70
32	17

Over: 100
Under: 20

762	23
43	7
750	35
41	9
754	31
42	8
741	44
43	7
739	46
43	7
750	35
41	9
752	33
42	8
738	47
42	8
752	33
41	9
747	38
40	9

Over: 0
Under: 20

785	0
49	1
784	1
50	0
782	3
49	1
784	1
49	1
785	0
49	1
782	3
50	0
785	0
49	1
785	0
50	0
783	2
49	1
784	1
48	1

Over: 0
Under: 60

772	13
43	7
778	7
48	2
782	3
49	1
757	28
45	5
757	28
45	5
764	21
47	3
768	17
49	1
771	14
46	4
770	15
48	2
767	18
45	4

Over: 100
Under: 0

763	22
41	9
762	23
44	6
766	19
46	4
752	33
43	7
760	25
44	6
752	33
41	9
752	33
45	5
742	43
41	9
756	29
43	7
763	22
40	9

Over: 100
Under: 40

736	49
37	13
737	48
39	11
738	47
43	7
741	44
42	8
735	50
40	10
717	68
37	13
721	64
38	12
709	76
42	8
734	51
38	12
735	50
36	13

Over: 100
Under: 60

698	87
28	22
703	82
36	14
693	92
32	18
688	97
40	10
715	70
40	10
670	115
33	17
695	90
39	11
700	85
32	18
700	85
37	13
712	73
35	14

Over: 200
Under: 0

763	22
40	10
762	23
45	5
751	34
47	3
758	27
45	5
762	23
47	3
764	21
47	3
759	26
44	6
773	12
47	3
767	18
45	5
771	14
48	1

Over: 200
Under: 40

751	34
43	7
732	53
41	9
738	47
39	11
732	53
43	7
732	53
42	8
738	47
45	5
761	24
42	8
741	44
42	8
756	29
40	10
734	51
40	9

Over: 200
Under: 80

510	275
7	43
525	260
15	35
523	262
12	38
524	261
17	33
485	300
13	37
588	197
28	22
532	253
20	30
576	209
28	22
613	172
27	23
611	174
29	20

Over: 100
Under: 80

586	199
25	25
623	162
26	24
641	144
21	29
635	150
28	22
591	194
19	31
615	170
31	19
609	176
34	16
582	203
31	19
611	174
24	26
643	142
30	19

Over: 200
Under: 20

761	24
39	11
756	29
43	7
755	30
46	4
761	24
45	5
742	43
44	6
764	21
45	5
758	27
43	7
745	40
46	4
775	10
45	5
762	23
44	5

Over: 200
Under: 60

735	50
34	16
706	79
37	13
728	57
42	8
693	92
38	12
654	131
31	19
670	115
35	15
683	102
39	11
699	86
39	11
742	43
43	7
717	68
36	13

Over: 300
Under: 0

760	25
40	10
754	31
43	7
760	25
45	5
749	36
42	8
740	45
44	6
744	41
43	7
772	13
46	4
764	21
46	4
770	15
44	6
764	21
45	4

Over: 300
Under: 20

759	26
43	7
759	26
44	6
753	32
45	5
743	42
44	6
729	56
37	13
729	56
43	7
744	41
43	7
749	36
47	3
753	32
40	10
754	31
44	5

Over: 300
Under: 60

682	103
31	19
669	116
38	12
722	63
39	11
701	84
38	12
669	116
37	13
625	160
29	21
662	123
33	17
629	156
30	20
715	70
36	14
670	115
36	13

Over: 400
Under: 0

764	21
46	4
743	42
39	11
749	36
42	8
769	16
49	1
752	33
46	4
757	28
43	7
754	31
43	7
745	40
44	6
757	28
40	10
764	21
45	4

Over: 400
Under: 40

721	64
33	17
721	64
40	10
735	50
39	11
726	59
43	7
725	60
38	12
711	74
32	18
719	66
43	7
719	66
41	9
704	81
31	19
739	46
45	4

Over: 300
Under: 40

728	57
39	11
724	61
36	14
731	54
37	13
718	67
42	8
713	72
42	8
707	78
37	13
738	47
45	5
718	67
36	14
707	78
35	15
751	34
44	5

Over: 300
Under: 80

560	225
8	42
546	239
15	35
523	262
16	34
433	352
14	36
521	264
16	34
504	281
14	36
573	212
27	23
509	276
15	35
562	223
19	31
585	200
25	24

Over: 400
Under: 20

743	42
31	19
728	57
40	10
747	38
43	7
732	53
39	11
727	58
39	11
738	47
41	9
738	47
41	9
727	58
43	7
749	36
42	8
756	29
43	6

Over: 400
Under: 60

688	97
29	21
679	106
32	18
688	97
36	14
648	137
38	12
679	106
38	12
680	105
35	15
681	104
36	14
674	111
31	19
665	120
32	18
684	101
41	8

Over: 400
Under: 80

585	200
18	32
547	238
17	33
572	213
16	34
575	210
22	28
568	217
23	27
478	307
19	31
529	256
25	25
552	233
14	36
541	244
16	34
570	215
26	23