Building a Competitive Associative Classification Model

by

Nitakshi Sood

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

The power of associative classifiers is to determine patterns from the data and perform classification based on the features that are most indicative for prediction. Although they have emerged as competitive classification systems, however, they suffer limitations such as without prior knowledge it would be cumbersome to state the proper support and confidence threshold values which vary with the dataset. Most of the existing rule-based classifiers also suffer from the production of a large number of classification rules, affecting the model readability. This hampers the classification accuracy as noisy rules might not add any useful information for classification and also lead to longer classification time. In this study, we further propose SigD2 which uses a novel, two-stage pruning strategy which prunes most of the noisy, redundant and uninteresting rules and makes the classification model more accurate and readable.

Furthermore, deciding a heuristic for associative classification system such as sum, average, minimum, maximum of confidence of the rules is yet another challenging task. In our study, we propose BiLevCSS (**Bi-Lev**el **C**lassification using **S**tatistically **S**ignificant Rules), a two stage classification model which implements automatic learning on the rules. In the first stage of learning, statistically significant classification association rules are derived through association rule mining. Further in the second stage of learning, we employ a machine learning based algorithm which automatically learns the weights of the rules for classification. We use the p-value obtained from the Fisher's ex-

act test to determine the statistical significance of rules. The rules obtained from the first stage form meaningful features to be used in the second stage of learning. Therefore, in this study, the supervised learning classifiers like Neural Network, SVM and rule based classifiers like RIPPER help in classifying the rules automatically in the second stage of learning, instead of forcing the use of a specific heuristic for the same.

Further, it has been noticed that due to the huge success of deep learning, other machine learning paradigms have had to take back seat. Yet other models, particularly rule-based, are more readable and explainable and can even be competitive when labeled data is not abundant. To make SigDirect more competitive with the most prevalent but uninterpretable machine learning-based classifiers like neural networks and support vector machines, we further propose bagging and boosting on the ensemble of the Sigdirect classifier. The results of the proposed algorithms are quite promising and we are able to obtain a minimal set of statistically significant rules for classification without jeopardizing the classification accuracy.

Another challenge faced by associative classifiers is their inability to deal with very high dimensional data sets. In order to address this problem, we divide the high dimensional feature space into smaller subspaces, to be given as an input to the ensemble of SigD2. Our proposed algorithm, Diverse SubSpace for Ensemble (DSAFE) ensures diversity among each subspace while ensuring the coverage of the total feature space. This strategy although compensates on the explainability factor of the complete model, however, each subspace still remains a white-box and there is a possibility of getting the explanation of obtained results. We have also tested all our models on the UCI datasets and were found to outperform various state-of-the-art classifiers not only in terms of classification accuracy but also in terms of the number of rules. We have also tested our classification model on the COVID-19 Kaggle dataset for

prediction problem and the results obtained are quite promising. Thus, our study highlights the fact that the association based classification models can be quite competitive with various other existing approaches. Lastly, we also show that designing a multi-layered architecture for feature transformation on SigD2, like deep neural networks, can potentially give good results.

Preface

The following papers are a part of this manuscript and were accepted in DaWaK 2020 and DEXA 2020 respectively.

1. Nitakshi Sood, Osmar Zaiane, "Building a Competitive Associative Classifier", accepted for publication in The 22nd International Conference on Big Data Analytics and Knowledge Discovery - DaWaK2020, Bratislava, Slovakia, September 14-17, 2020. [48]
2. Nitakshi Sood, Leepakshi Bindra, Osmar Zaiane, "Bi-Level Associative Classifier using Automatic Learning on Rules", accepted for publication in the 31st International Conference on Database and Expert Systems Applications - DEXA2020, Bratislava, Slovakia, September 14-17, 2020. [47]

In the list mentioned above, paper 1 is included in Chapter 3 and 5.
Paper 2 is covered in Chapter 4. I was the primary contributor and my co-author Leepakshi Bindra helped constructively in completing the experiments and writing.
Dr. Osmar Zaiane provided critical inputs and was the primary instructor and co-author for both the publications.

To my family

For their constant support and motivation.

Arise, awake, and stop not until the goal is achieved.

–Swami Vivekananda

Acknowledgements

I want to express my profound gratitude to my supervisor, Dr. Osmar Zaiane for his valuable guidance, stimulating suggestions, constant support and encouragement that guided me in exploring this area of research. His valuable suggestions and time spent in discussions has added greatly to my knowledge.

I am grateful to my supervisory committee for taking out time to review my research work. I would like to thank Mohammad Hossein Motallebi Shabestari, for helping me with the original SigDirect code and Parnian Yousefi for introducing me to the RIPPER's pruning strategy. I would also like to thank Leepakshi Bindra for collaborating with us on the topic 'Bi-level Associative Classifier using Automatic Learning on Rules'.

Lastly, I would like to express my deepest gratitude to my family and friends for entrusting me and providing me with their unfailing support and continuous encouragement all the time. It has been a long journey up until finishing the thesis and this is just a beginning for more adventure ahead.

Thank you.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Thesis Statement . . . 9
- 1.3 Thesis Contribution . . . 10
- 1.4 Thesis Outline . . . 12

2 Related Work **14**
- 2.1 Classification . . . 14
  - 2.1.1 Associative Classification . . . 16
  - 2.1.2 Statistically Significant Classification . . . 22
- 2.2 Ensemble Learning Models . . . 24
- 2.3 Application of Associative Classifiers . . . 27

3 SigD2 **28**
- 3.1 Methodology . . . 28
  - 3.1.1 Notations and Definitions . . . 28
  - 3.1.2 Rule Generation Phase . . . 30
  - 3.1.3 Rule Pruning Phase . . . 31
  - 3.1.4 Classification Phase . . . 34
- 3.2 Experimental Results . . . 35
  - 3.2.1 Classification Accuracy . . . 35
  - 3.2.2 Number of Rules . . . 36
  - 3.2.3 Statistical Analysis . . . 37

4 Bi-Level Associative Classifier using Automatic Learning on Rules **41**
- 4.1 Methodology . . . 41
  - 4.1.1 Method 1 . . . 41
  - 4.1.2 Method 2 . . . 44
- 4.2 Experimental Results . . . 48
  - 4.2.1 Classification Accuracy . . . 48
  - 4.2.2 Statistical Analysis . . . 52

5 Ensemble Learning on the Associative Classifiers **55**
- 5.1 Bagging and Boosting on wSigDirect . . . 55
  - 5.1.1 Methodology . . . 55
  - 5.1.2 Experiments . . . 58
- 5.2 DSAFE for SigD2 on High Dimensional Data . . . 62
  - 5.2.1 Methodology . . . 62
  - 5.2.2 Experiments . . . 68
- 5.3 Application of Associative Classification model . . . 74

6 Multilayered framework for SigD2 **76**
6.1 Background 76
6.2 Methodology 78
6.3 Experimental Results 80

7 Conclusion **85**
7.1 Summary 85
7.2 Future Work 87

References **89**

List of Tables

3.1 Comparison of classification accuracy of SigD2 with other association and rule-based classifiers and SVM 39
3.2 SigD2 compared with other algorithms on the basis of number of rules 40
3.3 Best and runner-up counts comparison from Table 3.1 basis of classification accuracy 40
3.4 Statistical analysis of results obtained in Table 3.1 40

4.1 Example for transformed set of features in Class based 43
4.2 Example for transformed set of features in Rule based 43
4.3 Example for Transformed Set of Features for Method 2 47
4.4 Comparison of classification accuracy using Rule-based and Class-based Features extraction in Method 1 48
4.5 Comparison of classification accuracy of BiLevCSS with other state-of-the-art classifiers 53
4.6 BiLevCSS(NN) compared to the rest of the algorithms on 10 UCI datasets 54

5.1 Comparison of classification accuracy of ACboost with ACbag, SigD2, SigDirect, ANN and SVM 60
5.2 Best and runner-up counts comparison from Table 5.1 on the basis of classification accuracy 60
5.3 Statistical analysis of Table 5.1 61
5.4 Comparison of classification accuracy using different values of overlap rate threshold 70
5.5 Description of Datasets 71
5.6 Comparison of classification accuracy using Random Sampling vs DSAFE for SigD2 71
5.7 Comparison of classification accuracy using SigD2, DSAFE for SigD2, RIPPER, SVM and MLP 75

6.1 Comparison of the proposed multi-layered SigD2 with other contenders on the basis of the classification accuracy on two UCI Datasets 81

List of Figures

2.1 General Approach for Associative Classification 17

3.1 An example illustrating the two stage pruning process 34

4.1 Flowchart Illustration for training and testing phases of 2SARC 44
4.2 Flowchart Illustration for training and testing phases of BiLevCSS 46
4.3 Comparison of classification accuracy for BiLevCSS(NN) with vanilla Neural Network, 2SARC1(NN) and 2SARC2(NN). . . . 51
4.4 Comparison of classification accuracy for BiLevCSS(NN) with BiLevCSS(RIPPER), BiLevCSS(SVM) and SigDirect. 51

5.1 Bagging on wSigDirect 57
5.2 Boosting on wSigDirect 58
5.3 Visual Illustration of the basic approach followed in DSAFE method 62
5.4 Comparison of classification accuracy on MLP, SigDirect, Random Sampling on Ensemble of SigD2, DSAFE with Bootstrap Sampling for SigD2, DSAFE for SigD2 on different UCI datasets. 72
5.5 Number of Estimators comparison with/without early stop on DSAFE for T_over= 0.5 and RF=0.5. 73
5.6 Comparison of Classification Accuracy on Different Datasets at different Overlap Rate. 74

6.1 Multi-Layered framework for SigD2 based Classification Model. Note - For the purpose of illustration, the window size {10,20,30} is chosen, it is assumed that their are 3 output classes for the considered data and each ESigDirect is an ensemble of 100 SigDirects formed using random sampling on data. 84

Acronyms

2SARC	Two Stage Associative Classification System.
ANN	Artificial Neural Networks.
ARC	Association Rule Classification.
BiLevCSS	Bi-Level Classification using Statistically Significant Rules.
CAR	Classification Association Rules.
CBA	Classification Based on Associations.
CMAR	Classification based on Multiple Association Rules.
CPAR	Classification based on Predictive Association Rules.
CSARS	Cost Sensitive Adaptive Random Subspace ensemble.
DNN	Deep Neural Networks.
DSAFE	Diverse SubSpace For Ensemble.
MDL	Minimum Description Length.
MLP	Multi-Layer Perceptron.
PSS	Potentially Statistically Significant.
SVM	Support Vector Machines.

Chapter 1

Introduction

1.1 Motivation

Classification is the process of organizing and categorizing data into distinct classes. It involves various tasks like building a model based on the distribution of the data in consideration and further using this model for identification of the class label of new data. Machine learning is being widely used today for various classification problems like text categorization, fraud detection, spam filtering, medical applications etc. The association rule mining is a rule-based approach that helps in identifying patterns in the data in the form of association rules, by finding the relationships between the items in the dataset. The association rules are in the form X $\rightarrow$Y, where X is the antecedent and Y is the consequent [1].

Associative classifiers combine the concept of association rule mining and classification to build a classification model. In an associative classifier, we choose the consequent of the rule to be the class label and the antecedent set is a set of attribute-value pairs for the associated class label. Associative classification has found its immense usage in market basket analysis, where the retailers try to analyse the association among the items bought by the buyers and consequently find patterns among the items frequently brought together. This helps the retailer in building their vending strategies in order to increase

their profitability. The application of the associative classification can also be extended to text classification, recommendation systems etc.

In the literature, various associative classifiers have been proposed till now namely, CBA [43], CMAR [41], CPAR [53], ARC[4] etc. Although these classifiers are easily understandable, flexible and do not assume independence among the attributes, they require prior knowledge for choosing appropriate parameter values (support and confidence). Another huge limitation of the previously proposed association rule based classification models is that, the rules generated may include noisy and meaningless rules, which might hinder the classification. A rule is said to be *noisy* if it does not add any new information for prediction and instead misleads the classification model. In other terms, a noisy rule would participate more often in miss-classifications than in correct classifications.

The authors in [39] proposed SigDirect, an associative classifier which mines statistically significant rules without the need for the support and confidence values. In our study, we first propose SigD2 where we introduce a more effective two stage pruning strategy to obtain a more accurate classification model. The proposed method reduces the number of rules to be used for classification without compromising on the prediction performance. In fact, the performance is improved. Most of the prevalent supervised classification techniques like ANN, SVM etc, although provide very high classification accuracy, they act as a black box. The models produced by such classifiers are not straight forwardly explainable. However, the proposed associative classifier makes the model more explainable by producing only a minimal set of classification association rules CAR. The proposed technique finds its immense usage in various health-care related applications, where the explanation of proposed models along with the classification accuracy are highly significant [55]. In health-care, incorrect predictions may have catastrophic effect, so doctors find it hard to trust AI

unless they can validate the obtained results.

Most of the previously proposed associative classification algorithms in literature have different rule discovery, rule pruning, rule prediction and evaluation methods. However, a predefined weighting scheme is required, for each of these methods in order to predict the class from the association rules. Heuristics like maximum/minimum of confidence, average of confidence or sum of confidence of the rules for the classes can be used to decide the predicted value for the new samples. However, the weighting scheme may differ for various applications when using associative classifiers. Deciding the heuristics to select rules to apply during inference and therefore to predict the class from the derived classification rules is a challenging task, and is typically fixed as part of the algorithm.

This form of classification offered by associative classifiers is easily understandable, flexible and does not assume independence between attributes, however, it requires prior knowledge to choose appropriate support and confidence threshold values for rule mining. Moreover, they contain a large number of noisy rules which are redundant, uninteresting and lead to longer classification time. Various pruning techniques have been designed to deal with this limitation, for instance, removing the low ranked specialized rules, removing conflicting rules or using database coverage based pruning strategy. A two level classification method was initially proposed by Antonie and Zaiane in [3], where the first stage used Apriori-based approach [1] to generates associative rule classification model which is followed by a stage of machine learning classifiers to learn the weights for classification in the second stage. We extended their work and compare the performance of SVM [16], Neural Networks [7] and RIPPER [15] in the second stage of learning. Although, this automatic approach of learning to use the rules is expected to give better classification results, it suffers with certain limitations. Firstly, the setting up of an optimal

support and confidence threshold values to mine the rules in the first stage is a cumbersome task. Secondly, the rules generated using the former approach may contain noisy, non statistically significant rules and may not cover all the important features in the selected rules.

So, in order to address the above given limitations, we propose BiLevCSS (**Bi-Lev**el **C**lassification using **S**tatistically **S**ignificant Rules), which uses statistically significant rules generated from a first stage, to form features that are made full use of, for classification in the second stage of learning. We follow the approach proposed by Li and Zaiane in [39] for generation of statistically significant CARs. We also use Fisher's exact test to obtain the p-value which is used to determine the statistical significance of the association rules. We further extract features from these significant association rules and then train the supervised learning classifiers like Neural Network, SVM and RIPPER on them. Finally, the trained model from the second stage is used to find the class label for a new data point.

Traditional association rules mining methods mostly prune the infrequent items on the basis of frequency of the itemset and thereafter calculate the strength of the rule in the form of its confidence values. This also ignores the statistically significant rules. Although most of the associative classifiers deal with this limitation by setting up small minimum threshold values, however, this leads to the generation of a huge number of insignificant rules. Therefore, in our proposed model, we use the instance-centric pruning strategy as used in SigDirect[39] to find globally optimal CAR (Class Association Rules) for each instance in the training dataset without compromising the classification accuracy.

Furthermore, we use Neural Networks [7] and Support Vector Machines [16] in our approach as they are strong machine learning classifiers, that have proved their worth in various applications. With the aim to build an efficient

classification strategy, we train them using meaningful features obtained from the first stage of learning. However, many real time applications specifically in healthcare and medicine require explainable models in order to interpret the results post classification. In our proposed strategy, although the statistically significant rules and derived features obtained in the first stage form an explainable model, Neural Network and SVM used in the second stage for classification might make the results un-explainable for such applications. Therefore, in order to make our approach interpretable, we explored the applicability of a rule-based classifier like RIPPER in the second stage for classification of derived features. Ripper [15] is a rule-based classifier which was found to produce a minimal set of explainable classification rules when given meaningful features in the second stage of our proposed approach, without compromising on the classification accuracy.

Therefore, in our study we propose a novel bi-level classification model, which uses the association rule mining to produce statistically significant rules. Further these rules are used to form more meaningful and non redundant features to be given as input in the second stage of learning comprised of a second classifier. The proposed algorithm helps in automatic learning of non noisy, statistically significant rules and further, it leads to a higher classification accuracy. The above proposed BiLevCSS classification model, overcomes the limitation of various associative classifiers described in literature for setting up the appropriate parameter values for support and confidence and also don't require any predefined weighting scheme for classification.

Due to the immense success of ensemble learning models in past, we further explore ensemble learning on associative classifiers. Therefore, we propose ACboost algorithm, which uses an ensemble of classification models obtained from the weak version of SigDirect, for boosting. Our goal is to strengthen the classifier using less number of rules for prediction. Since, SigDirect is a strong

learner and produces already a lesser number of rules for prediction, we form a weak version of SigDirect called wSigDirect, by further reducing the number of rules to be used for classification as explained later in Chapter 6. Moreover, in the proposed approach we use Adaboost [23] based boosting strategy over the ensemble of wSigDirect. The wSigDirect's classification model is learnt by running it multiple times on a re-weighted data, thereafter performs voting over the learned classifiers. We also propose ACbag which is defined as bagging on an ensemble of wSigDirect classifiers. Motivated by the approach proposed by Breiman in [8], we use an ensemble model of wSigDirect classifiers trained in parallel over different training datasets, and perform a majority voting over the ensemble for prediction. With the use of this strategy of combining weak learners, the goal is to decrease the variance in the prediction and improve the classification performance henceforth.

It was found that, ACboost performs better for most of the datasets than SigD2, ACbag, SVM, ANN which performs similar to deep neural network on these reasonably sized datasets. Deep learning has garnered all the attention lately, but their inability to produce transparent explanations for the decisions motivates us towards the domain of explainable artificial intelligence using rule-based models which have fallen out of favour of late. The main aim of this study is to make associative classifiers more competitive and to highlight their significance as opposed to the other machine learning based classifiers like neural networks which do not produce interpretable predictions.

However, for most of the association based classification models discussed before namely, SigD2, ACboost, ACbag and BiLevCSS, handling high dimensional data has been a huge concern. Essentially, handling large datasets has always been a big challenge for associative classifiers. While on the other hand, DNNs are generally found to perform better on large datasets.

So, one possible solution to handle a high dimensional dataset, could be,

to form an ensemble of feature subspaces in order to make the prediction. Motivated by the feature sampling strategy called CSARS proposed by Cao et al. in [11], we define a new strategy called **D**iverse **S**ubSp**a**ce **F**or **E**nsemble (DSAFE), to be used to build the ensemble of SigD2 using diverse feature subspaces. In our study, we have made an attempt to optimize the CSARS strategy, by using an efficient approach for searching the diverse feature subspaces and have been able to achieve encouraging results without doing bootstrap sampling on each of the sub-datasets so formed. Traditional approaches like Random Sampling have certain limitations of setting up the hyper-parameter for number of estimators and doesn't ensure the complete coverage and diversity among different estimators. In the proposed approach, we randomly select the features to build a feature subspace, while ensuring diversity among different subspaces. We also ensure that given the total number of features, there goes no feature that hasn't been selected at least once in this process, hence complete coverage of features in the ensemble. Next, these low dimensional feature subspaces are given to the ensemble of SigD2 for classification. DSAFE feature sampling approach automatically determines the number of estimators (i.e. different feature subspaces) to be used for classification. Thus, overcoming the need to determine the number of feature subspaces explicitly as a parameter while ensuring complete coverage of the total feature subspace and diversity among each of the subspaces. This makes the classification results obtained from so formed ensemble model, to be more authentic. It was found that using DSAFE for SigD2, is faster and gives higher accuracy than using SigD2 alone or using Random sampling on the ensemble of SigD2. It was even found to be competitive with MLP classifier on high dimensional dataset and has proved to be an efficient approach.

We have also tested the application of our model, on one of the classification problems related to COVID-19 pandemic. The data contains medical

information of some anonymous patients who visited the hospital to get tested. The goal here is to predict whether the patient should be admitted to intensive care unit, semi-sensitive ward, general ward or does not require admission to the hospital at all. This would help the practitioners to predict the number of hospitalizations that may occur, on the basis of early symptoms and medical tests. The results obtained are encouraging and highlight the significance of the explainable nature of associative classifiers.

Further, we also hypothesize that one of the many reasons for the excellent performance of the neural networks is their layered complex model that has feature transformation ability. With the aim to achieve high performance like deep networks, we build a multi-layered architecture for SigDirect, which is a kind of feed forward network, without back propagation. Therefore, it is a non differential model and has inherent feature transformation ability. Zhou et al. proposed the deep forest model in [57], which is a layered architecture consisting of ensemble of random forests, that transforms features within the model and has high model complexity to mimic the deep network style of classification but using lesser hyper-parameters. However, random forests involve a cumbersome process of tuning the number of trees in the forest as a hyper parameter for every dataset. Improper tuning of this parameter might overfit the model. The average probability distribution for each class from each tree in the forest forms the features to be cascaded in the further layers. Furthermore, our previous results from SigD2 have shown that, performance of SigD2 is competitive to that of random forest both in terms of accuracy and run time.

Motivated by the deep forest which uses random forests to build a deep architecture, we use SigD2 which produces statistically significant rules for classification, as the base classifier in this layered architecture. The parameters used for the proposed classification model are very less and do not require

much tuning as compared to the random forests and neural networks. Thus, avoiding the problem of over-fitting the model. This kind of architecture although compromises the explainability factor, however here we intend to build a classification model, which provides high classification accuracy irrespective of the size of data and requires less hyper-parameters for tuning at the same time.

Therefore, inspired by the deep architecture of neural networks and encouraging results from ensemble learning algorithms, we further build a layered architecture for learning an ensemble of SigD2. This architecture is an amalgamation of multi-grained scanning and cascading of information obtained from each SigD2 model. Initially raw features are given as input to the SigD2 using a sliding window in the multi-grained scanning step. SigDirect's classification model outputs the probability score for each class. Further, these probability scores are concatenated to form feature vectors for the next phase. These features are then transformed layer by layer based on the information obtained from the classification model of SigD2 at each layer.

We intend to build a system where the number of layers to be used in a network would be learnt automatically by the model. Hence, the number of layers would not be a pre-defined parameter and would not require tuning. This complex layered model is found to give better performance both in terms of speed and accuracy for large datasets. Furthermore, this architecture can also be extended to be used for high dimensional image datasets, as model explainability is not a huge concern in such classification problems.

1.2 Thesis Statement

In this study, we intend to address various challenges encountered with the use of associative classifiers. And we hypothesize the following statements for the same.

1. The most significant advantage of an associative classifier is its ability to provide an explanatory classification model. The goal is to obtain an accurate, readable classification model with minimal rules. We hypothesize that the pruning strategy used in the current associative classifier, SigDirect, can be optimized further to produce lesser number of rules for classification without jeopardizing the prediction performance.
2. We believe that statistically significant classification rules can form meaningful features to be used as an input for other machine learning classifiers. This would overcome the limitation of selecting any weighting scheme for associative classification while using very few hyper-parameters at the same time.
3. We next hypothesize that the ensemble learning on associative classifiers can help enhance the prediction performance and make associative classification models more competitive. A prudent feature subspace selection approach, can help in building an efficient ensemble model.
4. We hypothesize that one of the many reasons for the success of deep networks is their complex, multi-layered model which has ability to perform complex feature transformations. This idea could also be used to build a deep multi-layered framework for an associative classifier to make it more competitive.

1.3 Thesis Contribution

Our contribution in this study is as follows:

1. We propose SigD2, an associative classifier, which uses an effective two stage pruning strategy for pruning the rules to be used for classification. Using the proposed approach, the number of rules used for classification are reduced notably, without compromising on the classification perfor-

mance. We evaluate the proposed algorithm, on the UCI datasets and compare it with other commonly used classifiers which shows that our classifier gives better classification accuracy than other state-of-the-art rule-based and associative classifiers.

2. We propose BiLevCSS, (**Bi**-**Lev**el **C**lassification using **S**tatistically **S**ignificant Rules), which is an effective two stage learning model. In the first stage of learning, we build an associative rules classifier (ARC) model based on statistically significant rules, followed by a supervised learning classifier in the second stage of learning for classification. We evaluate the performance of machine learning classifier like Neural Network and SVM against rule based classifier RIPPER to compare their accuracy and suitability for different datasets.
3. We propose ensemble learning on associative classification. We introduce ACbag, an ensemble based classifier founded on wSigDirect and ACboost, which is boosting the wSigDirect classifier, to improve the classification accuracy with an explainable base model. Therefore, making SigDirect more competitive for classification tasks.
4. We propose Diverse SubSpace for Ensemble for SigD2, to perform prediction over the ensemble of diverse yet complete feature subspaces. The proposed strategy is found to give promising results for classification tasks on high dimensional data.
5. We propose multi-layered framework for SigD2, a deep layered architecture with feature transformation ability, for an associative classification based learning model. The proposed model uses very few parameters for tuning and gets encouraging results.

1.4 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2 reviews the work done in literature in the area of associative classification, statistically significant classification, ensemble learning model and applications of associative classifiers. Section 2.1.1 provides detailed explanation on associative classification including the rule generation phase, rule pruning phase and classification and also discusses various variants of associative classifiers proposed so far in the literature. Next, sub section 2.1.2 provides information on work done in past on statistically significant classification followed by detailed review on the ensemble learning models in Section 2.2. Finally we review some works that have applied associative classifiers in various healthcare and fraud detection systems in Section 2.3.
- Chapter 3 contains describes the novel optimized pruning strategy defined for SigD2. In this chapter, Section 3.1 explains the complete methodology, including rule generation, rule pruning and classification phase for SigD2. Further, Section 3.2 contains the contains the promising experimental results for SigD2 in comparison with other classifiers.
- Chapter 4 addresses some limitations of the prevalent association based classification models and provides details of the different novel associative classification models. In this chapter, Section 4.1 discusses the proposed algorithm BiLevCSS which is a novel Bi-level associative classifier using the automatic learning on rules. In Section 4.2 we compare the results obtained using the proposed BiLevCSS algorithm versus the other state of the art associative based and rule based classifiers, along with some statistical analysis on the results obtained.
- Chapter 5 contains detailed methodology and evaluation of the proposed

ACbag and ACboost ensemble algorithms introduced in Section 5.1. Further, Section 5.2 addresses the limitation of high dimensional data on associative classifiers and contains detailed information and evaluation of the novel algorithm called DSAFE used for the diverse feature sub-space selection used to form the ensemble learning model. Lastly, Section 5.3 describes the experimental results obtained from application of proposed classification models on a COVID-19 dataset.

- Chapter 6 introduces a multi-layered framework for an associative classifier, so as to make it more competitive with deep networks. Some background and motivation is described in Section 6.1, further Section 6.2 contains the basic methodology involved and Section 6.3 contains experimental results.
- Chapter 7 concludes the thesis with a short summary of the proposed work and the significant conclusions drawn from this study. It also describes the future work and the possible areas that can be explored based on the study performed in this thesis.

Chapter 2

Related Work

In this section, we describe related work on classification, statistically significant classification and ensemble learning. We also review some related work on the applications of associative classifiers.

2.1 Classification

Classification can be is defined as a process of predicting the class label of new data points, given a set of labeled data points in the training set. It is a three step process of model construction, model evaluation and finally using the generated model for classification. There are various techniques of classification commonly used machine learning namely, Support Vector Machines, Neural Networks, Decision trees, Logistic regression, Naive bayes classifier, associative classification etc. We would discuss a few of them below.

Cortes et al. proposed Support Vector Machines in [16]. It is a widely used linear classification model, that has the ability to classify non-linearly separable data. They can be applied to both classification and regression tasks. Given an N-dimensional space, where N is the number of attributes, the goal here is to find a hyperplane that classifies the data points. In SVM, kernels play a significant role in separating the non-linearly separable data points. The SVMs are not found to perform well when dataset is very large and seem to

take long time for classification. Also the prediction model produced difficult to understand and the outcome is not explainable.

Another much prevalent machine learning classifier, Artificial Neural Network was proposed by Beale et al. in [7]. A Neural network is a layered framework inspired by the architecture and intelligence of the biological brain. The network consists of many interconnected layers, each of which contains nodes through which information is passed. The nodes inside the network emulate the behaviour of the neurons inside the brain. Some non linearity in the form of activation function, is applied to the cumulative weighted sum of the inputs, to get the output from that node. The output can thereafter be given as the input to other nodes. The number of nodes in the input layer are equal to the number of attributes in the data set. For a classification problem, the number of nodes in the output layer are as the number of classes of the target variable. Although neural networks are robust and have been able to obtain high prediction accuracy, they are difficult to understand. They might not suffice the requirement of some applications where explainability of results is expected. Also, they are not found to perform very well when the size of training data is not very high.

Quinlan proposed Decision trees [44] which are interpretable classification algorithm which performs tests on the attributes so as to divide the data into various partitions. Each leaf node would represent the associated class label or class label distribution. It includes a two step process of tree construction and tree pruning. Tree construction begins with the root containing all training examples, and recursively the best attribute for split is found using different measures like Information gain and Gini index. Further in tree pruning phase all the noisy tree branches are chopped off that do not increase the accuracy on validation set. Tree pruning helps in overcoming the problem of over-fitting in decision trees. Decision trees can work well irrespective of the data type as

they can handle all types like numerical, categorical and so on. However, they are found to be more prone to over fitting and become unstable if not used properly.

Quinlan proposed the C4.5 algorithm was proposed in [45] to generate decision trees. Further, Ho et al. proposed the algorithm for Random forests which is an ensemble of the decision trees in [32].

Another form of explainable and intuitive classifiers in machine learning are associative classifiers. In the following section, we review some literature work on associative classification.

2.1.1 Associative Classification

Association rule mining aims to find patterns in the data, in order to determine the association and the correlation between the items in the transaction data set. It is widely used in market basket analysis, such that if a customer buys say product 'A', then how likely is it for that customer to buy product 'B'. This helps the sellers in building their selling strategies to increase their profit. Other applications of association rule mining could be for catalogue design or cross marketing. In fact, this can further be extended to any classification task where model analysis is required. For example, in a mushroom dataset consisting of two classes 'poisonous' and 'non-poisonous', then an obvious question can be, what are the attributes-value pairs that led to the prediction of a specific instance of a mushroom to be 'poisonous' or 'non-poisonous'. Associative classification forms association based classification rules that help in interpreting answers to such questions. Stemming from association rule mining, associative classifiers have been extensively studied in the last three decades.

The idea of associative classifiers was first presented by Liu et al. [43], while the concept of using association rules as CARs was proposed earlier by

Bayardo Jr. [46]. Bayardo used Apriori algorithm and proposed few pruning strategies to improve the performance by efficiently extracting the high confidence rules. Liu et al. first proposed the classification based on association (CBA) technique in [43] and showed that the association rule mining techniques could be applicable for classification tasks. Associative classification consists of three major steps, namely rule generation, rule pruning and rule selection as shown in Figure 2.1.

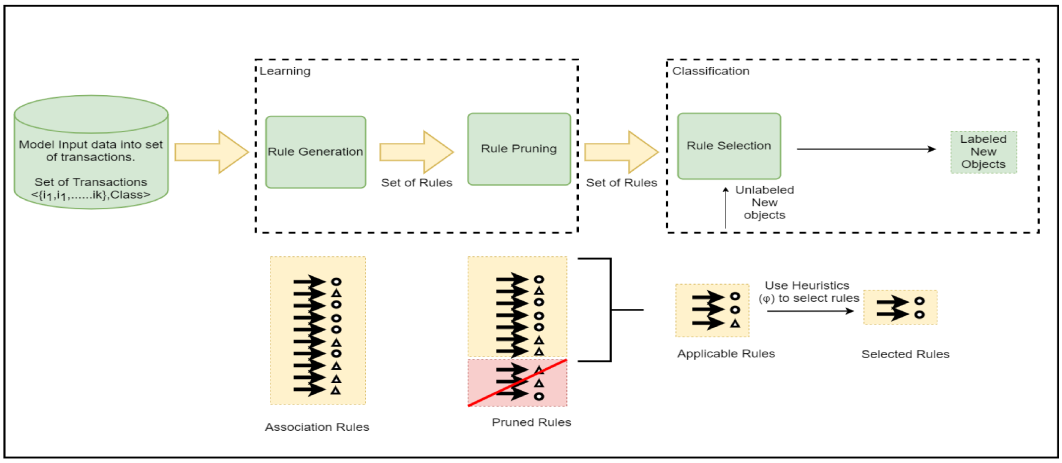

Figure 2.1: General Approach for Associative Classification

Rule Generation

The approach of association rule mining is used to find the frequent patterns and associations from a given dataset. A general association rule is represented as (X $\rightarrow$Y), where both antecedent and consequent are the itemsets found in the data, such that the consequent is found in combination with the antecendent. Support and Confidence are important measures used for determining these relationships. Support is measured by the proportion of transactions in which an itemset appears. Confidence is defined as the proportion of transactions having item X, in which the item Y also appears. Frequent itemsets are those itemsets which have support value larger than the support threshold

value. These frequent items are then used for the generation of rules that help in the prediction.

In this phase, transaction items are used to form association rules such that the antecedent form the list of attribute-value pairs while the consequent is the class label. Different association rule mining strategies have been proposed in past like Apriori [1], FP-growth [29] etc. Aggarwal et al. proposed Apriori which is one of the most widely algorithm for generation of frequent itemsets and class association rules(CARs). It majorly involves two steps for rule generation. In the first step all the frequent itemsets are generated from the given transaction set and further these frequent itemsets are used to generate the association rules. This algorithm requires to set the minimum support and confidence hyper parameter values to it to function well. It uses breadth first search strategy and requires very large memory due to large number of candidate itemsets generated in the process. Furthermore, another algorithm called FP-growth proposed by Han et al. in [29] was used for frequent itemset and rule generation. It uses the FP-Tree type data structure to obtain the frequent itemsets and further extracts the association rules. The algorithm uses depth first search strategy and requires less space comparatively. Most of the associative classifiers like CBA, ARC [4] use Apriori based rule generation algorithm while CMAR [41] uses FP-growth based rule generation algorithm, which would be discussed in detail later in this section.

Rule Pruning

Rules generated so far, may contain noisy rules, which are redundant and meaningless. Such rules do not help in the process of classification and should be removed. Liu et al. proposed database coverage based pruning strategy for CBA [43]. This pruning approach uses the generated CARs, such that each rule is matched and further evaluated based on the given training dataset. If

the rule has the ability of classifying even a single training example, then it is kept for classification otherwise it is removed. At the same time, all the transactions that matched the previous rule are removed as well. This process is repeated until all the rules have been tested similarly, thereafter the set of non-noisy rules is obtained.

Furthermore, another study done by Zaiane and Antonie in [54] focuses on the significance of obtaining a minimal set of CAR's without jeopardising the performance of the classifier. They propose a pruning strategy to reduce the number of rules in order to build an effective classification model without seriously compromising on the classification accuracy. The authors also propose heuristics to select rules which obtain high accuracy on the plot of correct/incorrect classification for each rule on the training set for effective rule pruning combined with the database coverage technique based on the given dataset.

Rule Selection

Before using this set of rules of classification, it is worth noting that the number of rules obtained so far are still large enough. The goal of classification is to select the relevant rule for each transaction in the test dataset. Since, the number of relevant rules would be more than one, consequently leading to large number of possible class labels. Therefore, it becomes highly important to select a heuristics which would lead to high classification accuracy. Among all the applicable rules, heuristics like maximum of confidence or average of confidence or sum of confidence of the rules for the classes, can be used to select the high quality rules. For each test transaction, the rules are selected on the basis of the chosen heuristics, to determine the class label effectively. Different associative classifiers [41, 43, 53] have been proposed in literature and each of them have chosen different heuristics for rule selection which would be discussed in next section.

Associative Classifiers

Liu et al. proposed CBA, an approach to perform classification using the class association rules in [43]. The proposed work used Apriori based rule generation algorithm, involving the cumbersome process of tuning support and confidence values. Furthermore, CBA applies the paradigm of "database coverage" for rule pruning and uses highest ranked matching rules as the heuristic for classification. This study paved the way for the associative classification.

CPAR proposed by Yin and Han in [53], uses a dynamic programming based greedy strategy that generates association rules from the training dataset. It prevents repeated calculation in rule generation and also selects best k rules in prediction.

Li et al. proposed another associative classifier called CMAR in [41]. CMAR uses FP growth which is a frequent pattern mining based approach to produce a set of association rules. The authors also use a novel data structure called CR-tree to store the CARs. Furthermore, CMAR determines the class label based on the set of matching rules using weighted chi-square measure, unlike the previous approaches. Antonie and Zaiane propose an associative rule-based classifier for automatic text categorization called ARC-BC [4]. ARC-BC forms association rules grouped by the category for each set of documents. The ARC model takes all the rules which lie within the confidence range, then the rules are grouped by the class labels (consequent) and finally the average confidence is calculated for all the rules in each category. The class label of the group with the highest value for confidence average is considered as the predicted category. The proposed algorithm works for both single and multi class classification.

The CCCS algorithm in [6] is proposed for imbalanced dataset classification problems where using support/confidence framework would not be suffi-

cient. The classification using complement class support algorithm mines the positively correlated CARs by using a novel measure called complement class support(CCS) conjointly with top-down row enumeration algorithm. However, CCCS does not assure the statistical significance of the mined association rules, which defeats the purpose we would like to achieve. Moreover, they only compare their approach with the original CBA. Later, Verhein and Chawla also proposed Significant, Positively Associated and Relatively Class Correlated Classification (SPARCCC) algorithm [49], which is an amalgamation of the statistical significance rules with a novel measure called CCR for calculation of relative class correlation of a rule. The proposed associative classifier does not use the support confidence threshold framework for classification and is stated to perform well on imbalanced datasets. Furthermore, the novel CCR measure helps in using only those rules for classification, where the antecedent is more positively correlated with the classes they predict as compared to their correlation with other classes.

Antonie and Zaiane [5] proposed the first associative classifier that uses both the positive and negative CARs. They use the Pearson's coefficient as the interestingness measure to mine positively and negatively correlated CARs. They were able to prove that a much smaller set of positive and negative CARs was efficient enough to compete and outperform various other categorization systems. The classification is made by using average confidence heuristic.

Coenen and Leng have reviewed three case satisfaction mechanisms namely, Best First Rule, Best K Rules and All Rules in [14] and various alternative rule ordering strategies. The authors have evaluated these case satisfactions as they have been commonly used in numerous Classification Association Rule Mining (CARM) algorithms to use the classifier thus formed, for the prediction task.

Antonie et al. proposed a two stage classification model called 2SARC in

[3] which automatically learns to select appropriate rules for classification. In the first stage, association rule mining is used to determine the classification rules. These rules are further used to determine meaningful features to be used for predicting which rules are to be selected for induction. In the second stage, these multiple features are given as input to another learning algorithm that is, a neural network, in order to obtain a more accurate classification model with high prediction accuracy. However, the model produced by the proposed approach does not seem to give explainable results. The main aim of the work proposed by Antonie et al. was to overcome the cumbersome task of tuning support and confidence values for every dataset. Although, the results obtained are interesting, however they are not convincing as they tend to ignore the statistical significance of the rules. Noisy and meaningless rules produced in the first stage might mislead the classification in the second phase. This forms the baseline of our work as discussed later.

2.1.2 Statistically Significant Classification

Hamalainen proposed StatApriori algorithm [27, 28] which explores statistically significant association rules using z-score measure and without using the frequency threshold framework. The proposed algorithm provides non redundant globally optimal association rules. However, the concept of StatApriori algorithm was found to be more constrictive than the traditional definition. Therefore, further in [25, 26], Hamalainen proposed the Kingfisher algorithm. The algorithm uses Fisher's exact test and new pruning strategies for searching the globally optimal dependency association rules. The algorithm performs a scalable search using branch and bound including the positive or negative dependencies over the attributes. Hamalainen's work has been a fundamental input to the statically significant rule mining.

Further, Li and Zaiane presented a novel associative classifier in [40] which

is built upon both positive and negative association classification rules. They improvised the kingfisher algorithm for rule generation, and also proposed a novel pruning strategy for both positive and negative rules simultaneously. The authors state that a generated rule can be pruned if it is found to incorrectly classify at least one training instance. Finally in the classification stage, they concluded that summing up the confidence values of all matching rules and accordingly making the class label prediction proves to be the best classification method.

Moreover, tuning values for support and confidence parameters is an arduous task as it varies with the change in dataset. Li and Zaiane in [39] overcome this limitation by proposing SigDirect that tunes only one parameter that is the p-value, which computes the statistical significance of rules using Fisher's exact test. The SigDirect is found to be immune to the problem of missing data, and handles datasets with the missing information quite efficiently.

The novel associative classifier SigDirect, involves rule generation, rule pruning and classification phase. First, the authors use Apriori based strategy for rule generation. The tree is enumerated starting from one item in the antecedent to generate class association rules. These association rules are further checked, if they are potentially statistically significant(PSS). Only the PSS rules are then further tested for statistical significance. Next using PSS 1-itemset rules, PSS 2-itemset rules are generated using the property that if a rule is PSS, then its parent rule would also be PSS. The process is repeated until no further PSS could be generated at that level. The algorithm also keeps a check whether the rule is non redundant and minimal or not (Also defined later in Section 3.1). The tree enumeration is also said to stop when a rule is marked as minimal as all its children rule would not be able to get a lower p-value. Further noisy and non statistically significant rules are removed from the set of generated rules using the proposed instance centric rule pruning

strategy. Li et al. have also evaluated various heuristics for the classification and infer that SigDirect, with a specific heuristic, gives high classification accuracy using a minimum set of globally optimal class association rules. Although SigDirect has proved to be quite competitive in terms of prediction, there are still noisy rules that can compromise the accuracy and has the potential of improvement and would be discussed in detail in Chapter 3. We would also design a multi-layered framework for SigD2 in Chapter 6, motivated by the architecture of Deep Forest, proposed by Zhou et al in [57], discussed in detail later in Section 6.1.

2.2 Ensemble Learning Models

Dasarathy and Sheela first presented the concept of composite classifier in [17]. The authors determined an optimality criterion for partitioning the features space using multiple classifiers. They proved that the composite system leads to better accuracy and improves the system performance unlike the case when each component from this composition is considered individually. Hansen et al. later proposed the idea of ensemble of similarly configured neural networks in [30], and laid emphasis on the reduction of generalized error using this ensemble system. The authors obtain the prediction output from each of the networks in the ensemble and do prediction based on the consensus scheme of voting. Different ways of creating diverse ensemble models have been used in literature. Most of them involve various sampling strategies, for example, one approach is to do bootstrapping and creating bags of data with replacement called bagging [8]. Another approach can be to split the features using random sampling without replacement approach [34]. The results from each of these different subsets are then given to the classifiers and final result is achieved through majority voting. Another approach can be to use ensemble of different kinds of classifiers such as MLPs, decision trees, support vector machines etc.

performing the classification on the same given dataset, as per the requirement of the application and later using some combining strategy like majority voting of the results from each classifier, to give the final prediction [33, 38, 51]. This approach can be slightly computationally expensive as different classifiers required different hyper-parameter tuning. Yet another less used approach is to use ensemble of one classifier, but having different parameters for each of the classification models.

Different approaches for training each member of the ensemble model are used in bagging [8], boosting [23] and their variants.

Ensemble models are widely used for enhancing the accuracy of the classification models using a combination of classification models. Freund and Schapire proposed Adaboost based boosting strategy in [23]. It is one of the most famous boosting algorithm which works on the principle of learning in series. Adaboost performs learning over an aggregation of various weak classification models. For the defined number of estimators, the weight is maintained over each training sample and for each base classifiers as well. After each iteration the miss-classified observations are given more weight after each iteration in the sequential model while lesser weight is given to the ones classified correctly. The process continues until the preset number of weak estimators are added or until low training error is achieved. The boosting strategy helps to reduce bias however, it is susceptible to the problem of overfitting. Later for the prediction phase, voting is done based on the weight of each classifier. Promising results obtained by authors in [23] motivated us to apply boosting over the associative classifier. Furthermore, the SAMME algorithm proposed by Zhu et al. in [31] is a multi-class extension of the binary Adaboost algorithm. Wolpert proposed the method of stacked generalization in [50], which is a stacked framework wherein at the first level classification models are trained on the given input dataset and the predictions from this layer are best blended

together to be used in the second layer for better output prediction and also to reduce the generalization error.

Data selection for creating subspaces to be used for the base models, play a very significant role in an ensemble system. The goal is to obtain diverse results from each of the base classifiers in the system, so as to highlight the significance of their amalgamation.

Brown et al. have presented a study on how negative correlation learning can be used over the ensemble of neural network [9]. This work describes that the significance of an ensemble model lies on the fact that the output from the base classifiers should be diverse enough such that either it is independent or might be negatively correlated. Brown et al. further present a detailed study in [10] on diversity in ensemble models with respect to both classification and regression contexts. The basic intuition here is that, the error rates produced by the base classifiers in the ensemble system should be diverse enough to emphasis the significance of the strategic combination of the classifiers in that system [56].

Cao et al. have proposed a novel Cost Sensitive Adaptive Random Subspace ensemble (CSARS) in [11]. In general random sampling method suffers limitation for imbalanced dataset classification and might lack diversity of instances in the ensemble model as it might involve overlapping of features in each subspace. The CSARS provides a cost sensitive framework ensuring diversity among the ensemble of subspaces and works efficiently with better accuracy for imbalanced datasets as well. CSARS automatically determines the number of estimators to be used in the ensemble model for classification.

Finally, there are various approaches in literature, for combining the class labels of the ensemble system by majority voting [37], weighted majority voting[42] and Borda count given by Jean-Charles de Borda [21]. In our study we extensively use ensemble modeling to highlight the significance of associative

classifiers.

2.3 Application of Associative Classifiers

Associative classifiers play a very significant role in health-care. Maria et al. proposed two associative classification strategies namely ARC-AC and ARC-BC to classify digital mammograms in [2]. Their study also emphasis the importance of data cleaning when using data mining approaches on image data. Since mammograms are of the trusted methods of cancer detection, the authors use the computer aided diagnostic system, which achieves high classification accuracy and the results are explainable at the same time. Exarchos et al. in [22] have also used the application of associative classifiers in the domain of health care. The authors propose the use of association classification rules for the ECG analysis, moreover the study shows how various classification tree algorithms can be used to discretize the continuous values features. The proposed model uses a three stage framework, given a raw ECG recording as input, the system performs feature extraction, followed by feature discretization, rule mining, and finally the best classification. The authors have laid attention on the ability of their proposed model to provide explanations on the decisions made, which is highly important in ECG analysis.

Furthermore, Ye et al. have proposed a novel CIDCPF system, to detect malware in [52]. The authors also use post processing approaches of associative classification like Chi square testing, use database coverage based rule pruning strategy on chi-square measure of ranking rules and pessimistic error estimation, and used best first rule heuristic for classification of new data file. The proposed CIMDS system [52] uses association rules to build CIDCPF classifier for detection of malware, and it is found to outperform various other antivirus detection systems like McAfee and Norton AntiVirus.

Chapter 3

SigD2

In this section, we first introduce the details about the proposed effective two-stage pruning technique as used in SigD2. Further, we evaluate the proposed classifier and compare it with various other associative and rule based classifiers on the UCI datasets.

3.1 Methodology

SigD2 processes the learning of rules in rule generation and rule pruning phases. It further uses these rules for prediction in the classification phase. The aim of an associative classifier is to find knowledge from data in the form of association rules associating conjunctions of attribute-value pairs with class labels, and then using these rules for prediction. The next sub section contains some useful notations and definitions and further we would discuss each of the phases processed by SigD2 below in detail.

3.1.1 Notations and Definitions

Definition 1. Dependency of a CAR [39]

If a transaction database T consists of a set of items $I = \{i_1, i_2, ..., i_m\}$ and a set of class labels $C = \{c_1, c_2, ..., c_L\}$, a transaction X in T consists of a set of items $A = \{a_1, a_2, ..., a_n\}$and a particular class label c_k such that $A \subseteq I$ and

$c_k \in C$. A CAR R in the form of A $\rightarrow c_k$ is called dependent if the antecedent part and the consequent class label of the CAR satisfy $P(A, c_k) \neq P(A)P(c_k)$, where P(A) denotes the probability of occurrence of itemset A.

Definition 2. Fisher's exact test [40]

Consider a null hypothesis in which A and c_k are assumed to be independent of each other. The dependency of the CAR A $\rightarrow c_k$ is said to be statistically significant at level α, if the probability p of obtaining an equal or stronger dependency in a dataset complying with a null hypothesis is not greater than α. The probability p, i.e., p-value, can be calculated by Fisher's exact test:

$$p_f(A \rightarrow c_k) = \sum_{i=0}^{min\{\sigma(A,\neg c_k)\sigma(\neg A,c_k)\}} \frac{\binom{\sigma(A)}{\sigma(A,c_k)+i}\binom{\sigma(\neg A)}{\sigma(\neg A,\neg c_k)+i}}{\binom{|T|}{\sigma(c_k)}} \tag{3.1}$$

where σ(A) denotes the support count of A and c_k represents the consequent class label. Note that $\sigma(A)$ and $\sigma(\neg A)$ represent presence and absence of conjunction of itemsets A in the transaction dataset T, respectively. The significance level α is usually set to be 0.05.

Definition 3. Potentially Statistically Significant [39]

*The CAR A $\rightarrow c_k$ is defined as "Potentially Statistically Significant" (**PSS**), if it meets either of the following conditions:*

(1) $\sigma(A) \leq \sigma(c_k)$ holds, and the lower bound $\frac{\sigma(\neg A)!\sigma(c_k)!}{|T|!(\sigma(c_k)-\sigma(A))!}$ is smaller than or equal to α;

(2) $\sigma(A) > \sigma(c_k)$ holds.

where $A \subseteq I_{Remaining}$ and $c_k \in \{c_1, c_2, ..., c_L\}$

If a CAR is **PSS**, we need to calculate the exact p-value to see if it is indeed statistically significant.

3.1.2 Rule Generation Phase

In this phase, we use the approach proposed by Li and Zaïane for SigDirect in [39]. SigD2 generates statistically significant CARs, such that the p-value from Fisher's exact test [25] of the rule in the form A $\rightarrow c_k$ is small. The rule generation process includes the enumeration of complete search space and applying certain pruning approaches to deal with this large space, which is found to increase exponentially as the number of items in the antecedent increase. Initially all the impossible antecedent itemsets are removed using the corollary defined in [39], which states that, the items whose support value is below $\gamma|T|$, where T is the transaction database and $\gamma \leq 0.5$, are termed as impossible itemsets. It is impossible for such items to occur in any statistically significant classification association rules and so they are removed.

The remaining items ($I_{Remaining}$) are sorted and arranged in the ascending order of their support values. The enumeration tree is built over the remaining items. For the first level, all the CARs with one antecedent values are listed and checked if they are potentially statistically significant (PSS) [39]. The Definition 3 describes two cases to make a rule as PSS. The lower bound for p(A $\rightarrow c_k$) is found to give the best-value which is $\frac{\sigma(\neg A)!\sigma(c_k)!}{|T|!(\sigma(c_k)-\sigma(A))!}$. If the value of lower bound is found to be smaller than the significance level α , then it is defined as PSS, otherwise it is called as non PSS.

All the non PSS CARs are pruned from the tree while, for the CARs which are PSS, exact p-value is calculated to find out if it is statistically significant. Using 1-itemset PSS CARs from the first level, 2-itemset PSS CARs are generated and this process is repeated until a certain level is reached where no PSS CARs can be generated. This is derived from another significant property considered here that is, a child CAR would be potentially statistically significant iff all of its parent CARs are PSS as well (Detailed proof can be

found in [39]). To ensure this, the algorithm uses the breadth first search strategy.

Furthermore, it is also checked if the CAR suffice the non redundant and minimal property [39]. The CAR in the form $A \rightarrow c_k$ is said to be non redundant if there does not exist any CARs in the form of $y \rightarrow c_k$, such that $p(y \rightarrow c_k) < p(A \rightarrow c_k)$ and y is proper subset of A, where p is the p-value of the rule calculated from Fisher's exact test, A is the set of items in the database and C_k is the class label [39]. While a CAR is termed as minimal if $A \rightarrow c_k$ is non redundant and there does not exist any CARs in the form $Z \rightarrow c_k$ such that A is a proper subset of Z and $p(Z \rightarrow c_k) < p(A \rightarrow c_k)$. So, if a CAR is found to be minimal then it is impossible for all its children in the subtree to get a lower p-value [39]. Therefore, the tree is not enumerated further.

3.1.3 Rule Pruning Phase

The rule generation phase may produce many CARs which are noisy and would not only slow down the process of classification but also lead to incorrect classification. Originally, SigDirect only performs instance based rule pruning on generated rules. It was observed that, although the previous strategy produces globally best CARs, the rules were still noisy and could be further reduced. So the question is, how can we prune more rules without actually jeopardising the accuracy of the associative classifier?

We propose a two stage strategy for pruning, wherein we randomly divide the training set into train set and prune set in the ratio of 2:1. The rules are generated in the rule generation phase using the train set. However, for pruning, only the prune set is used. We arrange the CARs in the order from highest confidence values to the least. The proposed pruning process, consists of matching the CAR with highest confidence and scanning over all the transactions in the pruning dataset to see if they match. If the rule applies correctly

Algorithm 1: Algorithm for Two-Stage Pruning Strategy used in SigD2

Input: **T**: Pruning transaction database, **R**: Initial rule list from rule generation phase, $\mathbf{R_{mid}}$: Rule list being formed after pruning the insignificant rules from R, **conf_threshold**: Confidence threshold value.
Result: $\mathbf{R_{new}}$: Classification association rules to be used for prediction
while *rules exist in R* **do**
    Sort the rules in R in descending order of their confidence values
    Select the rule r_i with highest confidence from R and add to the R_{mid}
    if *conf(r_i) < conf_threshold* **then**
        break
    Find all applicable instances in T that match the antecedent of rule r_i
    if *r_i correctly classifies a pruning instance in T* **then**
        Mark r_i as a candidate rule in the classifier
        Remove all instances in T covered by r_i
    Update the confidence values, based on the remaining transactions
    Remove the rule r_i from the R
end
for *each instance t in the original transaction database T* **do**
    Scan the CARs from R_{mid} to find the matching CAR r_i, with highest confidence value
    if *$r_i \notin R_{new}$* **then**
        R_{new}.add(r_i)
        r_i.count=1
    else
        r_i.count+=1
    end
end

on the transactions, it is marked and is selected to be used for classification and subsequently the matching transactions are removed from the pruning set. We re-calculate the confidence values of the remaining rules, each time using the remaining transactions in the pruning set and arrange them in the descending order. This process is repeated until either the rules or transactions have been covered or until the confidence threshold is reached. It is assumed that for a

rule, if the confidence value in each iteration is less than the threshold, then that rule can be pruned as it is not able to cover at least few instances in the prune set.

After this step, we obtain the rules which might be useful for classification. However, we still need to find the globally best CARs. So, further we apply the instance based pruning step as proposed in SigDirect [39]. For every instance in the pruning transaction database, the complete set of CARs generated from the previous step are scanned. The aim here is to find the matching CARs with the highest confidence value, such that, the class label of the rule and the transaction matches and the antecedent of the rule is the subset of the transaction. Furthermore, the count of how many times the rule has been selected in the pruning instances is maintained. This is later used in order to perform weighted classification using the number of times the CAR was selected in the pruning phase. Using the proposed approach only high quality rules with high confidence values are kept. High quality rules are the non noisy rules which do not make mistakes on the pruning set. Figure 3.1 shows an example of two stage pruning process on the Iris dataset. The first block in the figure shows all the rules obtained from the rule generation phase. All the rules highlighted in red, are the ones that are pruned in the first stage of pruning itself for being noisy and meaningless. Further the rules highlighted in orange, are the ones that are pruned in the second stage of pruning, for not being globally optimal. Finally, after pruning out all the noisy CARs, the remaining rules are the high quality rules, that are highlighted in the green color and are further used in the classification phase. This pruning strategy also avoids over-fitting on the data.

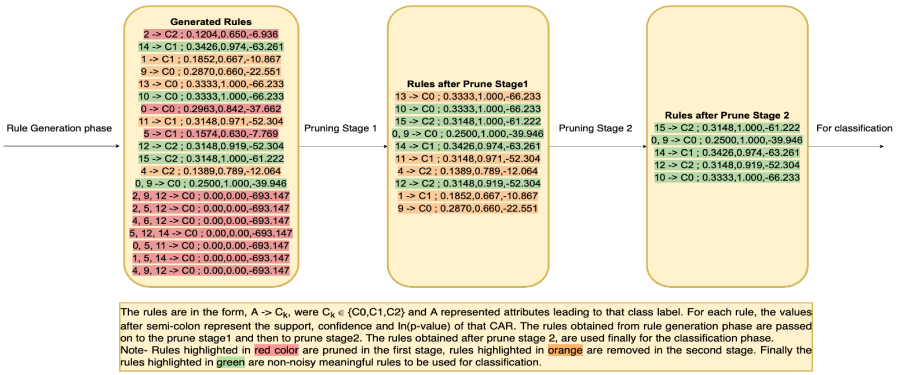

Figure 3.1: An example illustrating the two stage pruning process

3.1.4 Classification Phase

After the pruning phase, the minimal set of statistically significant rules are obtained. Further we make predictions on the new instances from the test set. For a given new instance, the classification process would search the subset of the CARs that match the new instance in order to predict its class label. All the matching CARs are then divided into groups based on their class labels. Then we use the three heuristics proposed in [39], which state that each class's group should be ordered on the basis of sum of ln(p-value), sum of confidence value and the sum of ln(p-value).confidence. It should be noted here that lower the p-value, the better the rule, and adding up the p-value is not a suitable heuristic for a set of rules. So, we take the log value of p-value as proposed by Li et al., in order to generalize it with the confidence measure.

The class label for the group sorted on the values of sum of ln(p-value) is obtained from the class of the matching CAR with the lowest ln(p-value). Moreover, the class label for the group sorted on values of sum of ln(p-value).confidence is obtained from the class of the matching CAR with lowest ln(p-value).confidence value. Finally, the class label for the group sorted on the values of sum of confidence value is determined from the class of the CAR with the highest confidence value. Furthermore, we can also use two stage clas-

sification as proposed by Antonie et al. in [3], to learn with a neural network in a second phase to predict the the classification rules to use.

3.2 Experimental Results

We evaluate our SigD2 associative classifier on 15 UCI datasets [20]. We discretize the datasets as proposed in [13], so the classification accuracy might be marginally different from the previously reported results. We report the results after performing the average over 10 fold cross validation on each dataset. We use 90% of the total data as the train set and further divide the train set into train set and prune set in the ratio of 2:1.

3.2.1 Classification Accuracy

We compare the performance of the proposed classifiers on 15 UCI datasets, with other rule-based classifiers like CBA, CMAR, CPAR, RIPPER, C4.5 and the original SigDirect, in terms of classification accuracy and number of classification rules in the final model. We also compare performance of SigD2 with SVM. We use the default parameters as stated by the authors in original respective papers for most of the contenders. In CBA and CMAR the parameters are tuned such that the minimum confidence values is set to be 50% , minimum value of support is set as 1%, the maximum number of CARs are limited to 80,000 and the size of number of antecedent items are limited to 6. In CPAR, the minimum gain threshold is set to 0.7, decay factor to 2/3 while the threshold for the total weight is set to be 0.05. For RIPPER[15], we use default JRip from WEKA [35]. The default parameters as stated in the respective papers, are used for SVM [16] and SigDirect[39]. For C4.5, the confidence factor (CF) was set to the default value of 25% and the minimum number of split off cases set equal to 2 [45]. For SigD2, we have performed a sensitivity analysis on the confidence threshold and it was found that threshold

value lower than 30% or higher than 50%, does not lead to best results for all the considered datasets. Hence, we chose to vary the confidence threshold in the range of 30-50% depending on the dataset.

Table 3.1 shows that SigD2 performs quite well as compared to other rule-based and associative classifiers. The average performance over 15 datasets of SigD2 is better than all the other rule-based classifiers. Although, the difference between SigDirect and SigD2 on the basis of classification accuracy is marginal, when we compare the number of rules, we show that SigD2 outperforms SigDirect. In order to have a fair comparison, among different algorithms on various datasets, we analyse how many times did an algorithm win and how many times it was a runner up as shown in Table 3.3. The proposed pruning strategy is found to give quite promising results as compared to the other rule-based and associative classifiers. SigD2 outperforms RIPPER on 10 out of 15 datasets. SigDirect also outperforms SVM for most of the considered datasets. Please note that, we would also compare SigD2 with classifiers other than associative and rule based classifier like ANN and DNN later in Section 5.1.2.

3.2.2 Number of Rules

The main advantage of the associative classifiers over the other machine learning supervised classifiers is its ability to build a model which is human readable. Noisy, redundant and uninteresting rules lead to longer classification time, reduce the performance of the classifier and also make it tedious for humans to analyse the model. Ideally, we want to achieve maximum accuracy with a minimum possible set of rules. Table 3.2 shows the comparison among different classifiers on the basis of number of rules generated. The two stage pruning technique is found to give a minimum number of rules without compromising the classification performance. Table 3.4 clearly shows that out of 15 datasets,

on average SigD2 outperforms most of the other rule-based and associative classifiers for at least 10 datasets with some ties in few cases as well.

SigD2 gets a smaller number of rules on 9 datasets when compared with CBA. Although, CBA is found to have less rules for some datasets, it is unable to provide a high accuracy in such cases. Our proposed strategy outperforms CMAR on all datasets, the original SigDirect on all but one dataset and CPAR, C4.5 on 8 datasets. The number of rules is found to be appropriate enough to provide information about the classification model without compromising on the performance. In Table 3.2, we take the difference of the average of number of rules over all the other classifiers and the proposed classifier in the last column. It is found that the difference is substantial which essentially shows the significance of the proposed pruning strategy. We also compute the percentage decrease of the number of rules on average in Table 3.2. Furthermore, SigD2 is found to outperform RIPPER in terms of accuracy for most of the datasets, however, RIPPER obtains less rules comparatively. This is majorly because RIPPER greedily modifies the generated rules using the Minimum Discription Length MDL principle. RIPPER produces a kind of superset of rules covering all information required for classification in the form of intervals. This indicates that there is potential for further improvements.

3.2.3 Statistical Analysis

For better understanding the performance over various datasets, we use Demsar's method [18] to perform statistical tests in order to compare different algorithms over different datasets. We perform non parametric Friedman's test for comparing the contenders with the proposed approaches. Friedman's test is generally used to compare more than two samples that are related. The default assumption is that the samples have the same distribution. The assumption is rejected if the probability of observing the data samples given

the base assumption(p-value) exceeds the significance threshold value (alpha). The Friedman's test is said to give significant results when the p-value is less than alpha. After analysis of Friedman's test with algorithms in Table 1, we obtained a p-value which is less than alpha (=0.05), which shows that at least one of the samples is significantly different from other samples. Hence, the results are found to be statistically significant.

Furthermore, we also perform Wilcoxon's signed-ranks test which is another non-parametric statistical hypothesis test to compare the performances of proposed algorithms and the contenders in a pairwise manner. In this test, initially the differences in the results obtained from the considered pair of algorithms is evaluated for all the datasets to calculate the absolute differences and further to sign each rank. For all the cases where tie in ranks occurs, the average rank is calculated. All the cases with the difference value of zero are ignored. If the original difference is positive then the rank remains positive, however, if the difference is found to be less than zero, then the rank is multiplied by -1. Further, the sum of positive and negative ranks are used to calculate the z-score values as defined in [18]. With the z-score value smaller than -1.96, the corresponding p-value is less than 0.05 which leads to the rejection of the null hypothesis. The results in Table 3.4 show that, SigD2 is significantly better than C4.5, CBA, CMAR, CPAR and SVM. However, the performance when compared with the original SigDirect seems to be quite similar and the p-value comes out to be greater than 0.05. We assume that, although there might not be difference in terms of classification accuracy, however, the new pruning strategy of SigD2 is more substantial and promising as it has reduced the number of rules to a small number as compared to the original SigDirect. SigD2's performance is found to be as good as RIPPER, however, with more wins and a p-value of 0.07.

Table 3.1: Comparison of classification accuracy of SigD2 with other association and rule-based classifiers and SVM

Datasets	**#cls**	**#rec**	**C4.5**	**CBA**	**CMAR**	**CPAR**	**RIPPER**	**SVM**	**SigDirect**	**SigD2**
Adult	2	48842	78.8	**84.2**	81.3	77.3	**84.1**	75.8	**84.1**	83.59
Anneal	6	898	76.7	94.5	90.7	95.1	**98.32**	85	96.99	**97.21**
Breast	2	699	91.5	94.1	89.9	93	**95.42**	**95.7**	91.7	92.7
Flare	9	1389	82.1	84.2	**84.3**	63.9	72.13	73.8	**84.23**	**84.3**
Glass	7	214	65.9	68.4	**71.1**	64.9	68.69	68.6	**70.56**	69.17
Heart	5	303	**61.5**	57.8	56.2	53.8	53.97	55.4	58.49	**59.81**
Hepatitis	2	155	84.1	42.2	79.6	75.5	78.06	79.3	**85.83**	**86**
Horse	2	368	70.9	78.8	82.3	81.2	**84.23**	72.5	81.23	**85.03**
Iris	3	150	91.3	93.3	94	94.7	**95.33**	94.6	94	**96**
Led7	10	3200	**73.9**	73.1	73.2	71.3	69.15	73.6	73.78	**73.81**
Mushroom	2	8124	92.5	46.5	**100**	**98.5**	**100**	99.8	**100**	**100**
PageBlocks	5	5473	92	90.9	90.1	**92.5**	**96.83**	91.2	91.21	92.18
Pima	2	768	70.5	74.6	74.4	74	66.36	74	**75.25**	**74.86**
Wine	3	178	71.7	49.6	92.7	88.2	91.57	**94.9**	92.71	**93.2**
Zoo	7	101	91	40.7	**93**	**94.1**	87.12	92.2	91	89.18
Average			79.62	71.52	83.52	81.2	82.75	81.76	**84.73**	**85.13**

Note- #cls indicates number of class labels and #rec indicate the number of records in dataset.

Table 3.2: SigD2 compared with other algorithms on the basis of number of rules

Datasets	**C4.5**	**CBA**	**CMAR**	**CPAR**	**SigDirect**	**SigD2**	**Difference with Average # of rules**
Adult	1176.5	691.8	2982.5	84.6	91.2	53.62	951.7 (94.67%)
Anneal	17	27.3	208.4	25.2	41.7	29.2	34.72 (54.31%)
Breast	8.8	13.5	69.4	6	10.9	7	14.72 (67.65%)
Flare	54.4	115.1	347.1	48.1	75.8	25.7	102.4 (79.93%)
Glass	14.8	63.7	274.5	34.8	55.6	23.1	65.58 (73.9%)
Heart	23.9	78.4	464.2	44	80.2	27.7	110.44 (77.3%)
Hepatitis	8.1	2.3	165.7	14.3	33.3	16	28.74 (64.23%)
Horse	25.6	116.4	499.9	19	90.4	41.5	108.76 (72.38%)
Iris	8.4	12.3	63.4	7.4	6.2	4.8	14.74 (75.43%)
Led7	63.2	71.2	206.3	31.7	104.3	54.4	40.94 (42.94%)
Mushroom	121.2	2	102.6	11.1	106.4	48.9	19.76 (28.77%)
PageBlocks	16.3	7.6	80.6	29.9	31.1	13.2	19.9 (60.12%)
Pima	24.4	43.2	203.3	21.7	36.6	11.3	54.54 (82.83%)
Wine	12.8	4.7	122.7	15.2	29.3	16.3	20.64 (55.87%)
Zoo	5.3	2	35	16.9	16.2	9	6.08 (40.31%)

Table 3.3: Best and runner-up counts comparison from Table 3.1 basis of classification accuracy

Classifiers	**Best**	**Runner-up**
C4.5	2	0
RIPPER	3	4
SVM	2	0
CBA	1	0
CMAR	3	1
CPAR	1	2
SigDirect	2	4
SigD2	5	5

Table 3.4: Statistical analysis of results obtained in Table 3.1

Classifiers	**Wins**	**Losses**	**Ties**	**p-value**
SigD2 vs C4.5*	12	3	0	0.005
SigD2 vs RIPPER	10	4	1	0.074
SigD2 vs CBA*	13	2	0	0.005
SigD2 vs CMAR*	11	2	2	0.033
SigD2 vs CPAR*	12	3	0	0.008
SigD2 vs SigDirect	10	4	1	0.272
SigD2 vs SVM*	12	3	0	0.041

(*) indicates statistically significant results with a p-value of 0.05.

Chapter 4

Bi-Level Associative Classifier using Automatic Learning on Rules

In this section, we introduce the details about the proposed Bi-Level classification technique. We initially describe Method 1, which is the baseline technique of developing a two level classifier by using the Apriori algorithm for building the ARC model in the first level. However, this technique was found to suffer limitations with regard to selecting the optimum support and confidence threshold values for different datasets. Therefore, we extended our baseline to include the approach proposed by Li et al. [39] in Method 2. In our proposed method we use statistically significant CARs to obtain rule features that are used in the second stage of learning.

4.1 Methodology

4.1.1 Method 1

The aim of associative classification is to find knowledge from data in the form of association rules associating features and class labels. During inference one or a set of rules are selected and used to predict the class label. This selection is typically based on heuristics for ranking rules.

Using the proposed approach of two stage classification in [3], we have im-

plemented the same technique for building a model which would learn to select and use the discovered rules automatically rather than relying on heuristics to select them. The Figure 4.1 shows the flowchart illustration of Method 1. In brief, the first stage is to learn an associative classifier and the second stage is to extract features from the learned rules to learn a second predictor predicting which rule is best to use during inference. The initial training dataset is split into two parts, one used to derive rules with association rule mining and the second part to extract features for the second training level. These two sets are disjoint in order to avoid overfitting. On the TrainSet 1, the first stage of learning is performed. Here, our algorithm uses a constrained form of Apriori[1] to perform association rule mining to obtain a set of rules that have features on the left and class labels on the right side of the rule and that are above the minimum threshold values for support and confidence. This ARC Model is used to collect a set of features from the samples present in TrainSet 2. As proposed in by Antonie et al. in [3], we have used two approaches namely, the class based (2SARC1) and the rules based (2SARC2) feature extraction , to get the set of features and class labels from the ARC model.

Class Based Features

For the class based feature extraction technique, we derive rules from TrainSet 1 and we match the features from our TrainSet 2 with the antecedents of the rules in the ARC Model. A rule is said to be *applicable* to a new instance of TrainSet2 if the antecedent of the rule is a subset of the features of the instance. Using the set of rules that apply to the instances in TrainSet 2, we count the number of rules that match for each class. Using this approach we derive a transformed feature set as shown in Table 4.1, where we state the average confidence and the count of all the matching rules for an example of

three given class labels. This dataset of *class-based features* is given to the next level of learning in order to train a classification model that selects rules.

Table 4.1: Example for transformed set of features in Class based

Class1		**Class2**		**Class3**	
Avg Conf	#Rules	Avg Conf	#Rules	Avg Conf	#Rules
85	1	81.6	3	80	2

Table 4.2: Example for transformed set of features in Rule based

R1			**R2**			**R3**		
Conf	Sup	Match	Conf	Sup	Match	Conf	Sup	Match
80	10	0	90	10	1	85	15	1

Rule Based Features

For the rule based approach, we use the characteristics of the rules derived from TrainSet 1 to create a new feature space. For each instance in the dataset TrainSet 2, we check if each of the rules in the ARC model apply or not, that is we match the features from the sample with the antecedents of the rule. This feature is denoted by a boolean value 1 to represent a match, 0 for absent. Along with this, information of support and confidence is added as features in the new set. An example is shown in Table 4.2, where one row in the dataset is taken and a new feature is generated for 3 rules of the ARC Model.

The features derived using the ARC model are further given as a training input to the next level, consisting of the classifier, which learns how to use the rules in the prediction process. In the second level, machine learning based classifiers like Neural network (NN) and Support Vector Machine(SVM) or rule based classifier like RIPPER, are used to automatically learn on rules to determine the weighting scheme for classification and obtain the final model.

For testing, we use the ARC model to derive the set of features for the Test dataset. Further, these features are given to the trained model of Neural network, SVM or RIPPER to classify the new samples. The ARC model and

the trained model in the second level together predict the class for any new sample given for classification.

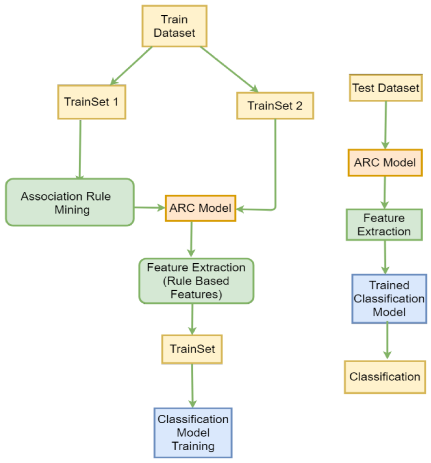

Figure 4.1: Flowchart Illustration for training and testing phases of 2SARC

4.1.2 Method 2

In our second approach, we propose BiLevCSS (**Bi**-**Lev**el **C**lassification using **S**tatistically **S**ignificant Rules).

For this purpose, we extend the bi-level classification technique by using statistically significant class association rules [39]. Li et al. proposed to use Fisher's exact test to extract the statistical significance of rules. The proposed algorithm determines non-redundant association rules for classification which show statistical dependency between the antecedent items and the consequent labels by using the p-value.

The process is illustrated in Algorithm 2. We split our training dataset into two parts as shown in Figure 4.2. On the TrainSet 1, the first level of learning is performed. The association rule mining is done by building Apriori like tree to form the ARC model, which gives us the set of association classification rules. The rules described by this ARC model are statistically significant, giving us the p-value for each rule. The rules obtained in the first level are

Algorithm 2: Algorithm for BiLevCSS

Data: **Train Dataset**: Initial training dataset. **Test Dataset**: Initial testing dataset. **TransformedTestSet**: Testing dataset for classification model. **TrainSet1**: Training set used to build the ARC Model. **TrainSet2**: Training set used to build features using the ARC model and train the classication model.

Result: Predict class label of each instance in **TestSet**.

```
1  Use TrainSet1 to generate all statistically significant CARs A →c_k. ;
     ▷ Follow the Algorithm 1 and 2 in [39]
2  classLabelsSet ⟵ Unique set of class labels in dataset
3  ARC Model = { CARs A →c_k | c_k ∈ classLabelSet}
4  for each instance T in TrainSet2 do
5  |  NewFeature=[]
6  |  for each rule R in ARC model do
7  |  |  match(T, R) ;      ▷ Determine if instance T matched the
   |  |   antecedent of rule R.
8  |  |  if match(T, R)==True then
9  |  |  |  NewFeature.append(Conf(R), Support(R), log(P-value(R)),
   |  |  |   1);
10 |  |  else
11 |  |  |  NewFeature.append(Conf(R), Support(R), log(P-value(R)),
   |  |  |   0);
12 |  |  end
13 |  end
14 |  TransformedTrainSet.append(NewFeature);
15 end
16 Train a supervised learning model using TransformedTrainSet dataset
     for classification.
17 Repeat steps 4 to 15, to extract features from Test Dataset using ARC
     model to build TransformedTestSet for second stage of learning.
18 Derive the accuracy of the classification model using the Test dataset.
```

used to extract the transformed feature set from the TrainSet 2. We used rule-based approach as described in Method 1 to extract features for this classifier as well. This is because the rule-based features in Method 1 showed better results than class-based features, as will be discussed in Section 4.2.

As proposed in [39], initially, all the impossible items are removed. An item is termed as impossible to appear in a statistically significant CAR if it has support value below $\gamma|T|$, where $\gamma \leq 0.5$ and T is the transaction database.

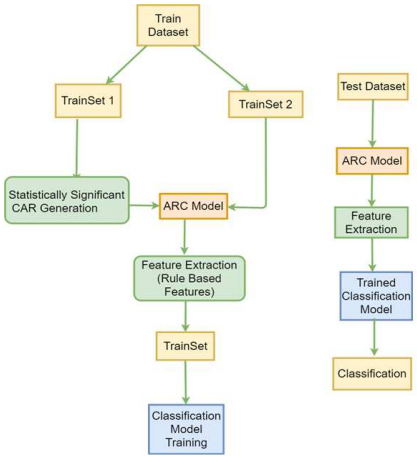

Figure 4.2: Flowchart Illustration for training and testing phases of BiLevCSS

These items are removed and thereafter all the left over items are sorted in the ascending order of their support values. Further the tree is enumerated to generate class association rules and only those with one antecedent are listed. These rules are then checked for their PSS value (Definition 3 described in Section 3.1.1). Rules that do not satisfy either of the PSS conditions are pruned and the other rules are checked for statistical significance. From PSS 1-itemset rules, PSS 2-itemset rules are generated considering the property that if a rule is PSS, then its parent rule will also be PSS, i.e. if CAR A $\rightarrow c_k$ is PSS, then any of its parent rule B $\rightarrow c_k$ is also PSS, where $B \subsetneq A$ and $|B| = |A| - 1$. The process repeats until no PSS rules are generated at a certain level. Also, if a rule is marked as minimal, the expansion from this rule is stopped because all of its children rules can not get a lower p-value.

The number of rules generated by the above approach may be large and might contain some unnecessary rules as well. In order to make the classification efficient and to obtain globally best rules from the training dataset, we use the proposed instance-centric rule pruning approach [39]. These pruned rules form the ARC model for this method. It should be noted here that, we only use instance based pruning strategy instead of two-level pruning ap-

proach proposed in the previous chapter. This is because the latter was found to reduce the number of rules by a large number. This does not suffice the requirement of BiLevCSS as it requires more rules to generate more data for machine learning classifier in the stage2.

We further apply the rule based approach to extract the features for the TrainSet 2 using this ARC model. An example for rule-based feature extraction for Method 2 is shown in Table 4.3 with just two rules. For each sample in TrainSet 2, we take the boolean value representing whether the rule matches the sample or not. Along with this, we take the characteristics of the rule as features in the transformed feature set. These include support value of the rule, its confidence and the log of the p-value. The lower the p-value, the better the rule, and summing up the p-value is not a suitable heuristic for a set of rules. Hence, we take the log value of p-value in order to generalize the process for rule-based and class based feature extraction. The features are extracted for each row in the testing dataset using the ARC model and the learnt classification model predicts the class label for each data point.

Table 4.3: Example for Transformed Set of Features for Method 2

R1				**R2**			
Conf	Sup	ln(p-value)	Match	Conf	Sup	ln(p-value)	Match
80	10	-10.6	0	90	10	-5.1	1

We also evaluate the BiLevCSS with SigDirect associative classifier in the second level. However, SigDirect is found to have a limitation of not being able to work well with very high dimensional datasets. For some datasets when using BiLevCSS, the features extracted for the second phase are found to have a large dimensionality due to a sizeable number of generated rules. This greatly increases the runtime of the SigDirect algorithm when used in the second phase. Therefore, we do not report the results of SigDirect as a predictor in the second stage.

4.2 Experimental Results

We have evaluated our algorithm on 10 UCI datasets to compare the classification accuracy with other rule based and machine learning based algorithms that exist in the literature. We report the average of the results obtained for every dataset on the 10 fold cross validation in our experiments. We compare the performance with common machine learning techniques like SVM and Neural networks, rule-based classifiers like C4.5 and RIPPER and previously proposed associative classifiers like CBA, CMAR and CPAR. We also compare our baseline approaches 2SARC1 (NN) [3], 2SARC2 (NN) [3], 2SARC1 (SVM), 2SARC2 (SVM), 2SARC1 (RIPPER) and 2SARC2 (RIPPER) with these classifiers.

Table 4.4: Comparison of classification accuracy using Rule-based and Class-based Features extraction in Method 1

Datasets	**#cls**	**#rec**	**2SARC2 (NN)**	**2SARC2 (SVM)**	**2SARC2 (RIPPER)**	**2SARC1 (NN)**	**2SARC1 (SVM)**	**2SARC1 (RIPPER)**
Iris	3	150	93.74	89.74	**94.28**	**94.11**	89.3	90.94
Glass	7	214	48.9	**52.2**	**69.17**	50	**52.2**	51.74
Heart	5	303	**63.5**	54.34	54.95	**62.34**	57.14	54.02
Hepati	2	155	**85**	**81.25**	79.97	70	75	80.48
Pima	2	768	66.45	65.2	**72.74**	64.39	67.53	**70.93**
Flare	9	1389	74.5	70.58	**84.35**	74.39	70.6	**83.96**
Anneal	6	989	77	82	**96.41**	79.5	78.04	**83.74**
Horse	2	368	67.6	63.3	**81.40**	**72.97**	70.96	63.75
Breast	2	699	89.7	93	93.14	93.75	**98.6**	**93.78**
Wine	3	178	**97.18**	77.97	85.15	**94.92**	72.02	53.84
Average			**76.35**	72.95	**81.15**	75.63	73.13	72.71

4.2.1 Classification Accuracy

We compare our proposed model BiLevCSS with the above stated contenders on the basis of classification accuracy. We evaluate the performance of BiLevCSS model with three different classifiers in the second level; Neural Network at the second stage (regarded as BiLevCSS (NN)), RIPPER in the second stage (regarded as BiLevCSS (RIPPER)) and SVM in the second stage (regarded as BiLevCSS (SVM)).

We follow the default parameter values for SVM [16], C4.5[45], CBA[43], CMAR[41], CPAR[53] as stated in the original papers. For RIPPER as a standalone rule based classifier, we have used default parameters from Weka [35] which are also stated to be the best by the authors in [15]. For vanilla Neural Network, we use a single hidden layer with the number of nodes to be the average of the number of input and output nodes and we also tune ReLU or sigmoid activation functions with a learning rate of 0.1.

For our baseline Method 1, we perform experiments using Apriori [1] based rule generation in the first level learning. Further, we test the accuracy of the rule-based feature extraction approach to build the bi-level classifier with Neural Network, SVM or RIPPER in the second stage. Similarly, we also measure the accuracy, of the bi-level classifier, which uses class-based features. For Apriori, we use a range of support values from 5% to 30% depending on the size of the dataset. The threshold value for confidence is set around 50%. In Table 4.4, we report the accuracy obtained for the 10 UCI datasets using our baseline approach. Along with the classification accuracy values, the name of the dataset and the number of records have also been reported. As can be seen from Table 4.4, the overall accuracy does not follow a pattern and nothing conclusive could be derived from the results aforementioned. However, the results from Method 1 showed that, for most of the UCI datasets, the rule-based feature extraction approach is found to give altogether a better average accuracy over the class-based feature extraction approach.

Therefore, in the second method, we adapt the rule-based feature extraction approach to build the bi-level classification model with statistically significant rules. For the following experiments, we discretize the numerical attributes of the datasets as stated in [13]. All the results reported in this section have been performed on the same discretized dataset for fair comparison.

Moreover, as suggested by Li et al. in [39], we use the Fisher exact test to

analyse the statistical significance of the class association rules. The threshold for p-value is set to be 0.05. The use of only statistically significant rules and the addition of p-value value along with support and confidence as a feature in the rule-based classification gives us much better results for Method 2 than the baseline Method 1. For the second layer of both the methods, we use Neural Network with single hidden layer, with 'ReLU' or 'sigmoid' as the activation functions and a learning rate of 0.1. We also tune the hyper parameter values of gamma, kernel and regularization parameters for the SVM classifier. We have performed 5 fold internal cross validation for SVM and NN to tune their respective hyper parameter values. For RIPPER at the second stage of learning, we use the default best parameters from Weka. It can be observed that, in Table 4.5, the BiLevCSS model gives the best overall classification accuracy for the considered datasets. Our algorithm BiLevCSS(NN) outperforms all the other classification algorithms in the 10 UCI datasets with highest average accuracy.

We further perform a comparison between BiLevCSS with Neural Network at the second level against the vanilla Neural Network with 1 hidden layer, to validate the efficiency of the model. The results show that the proposed algorithm outperforms the vanilla NN. Similarly, BiLevCSS(SVM) was found to outperform vanilla SVM and BiLevCSS(RIPPER) outperformed the vanilla RIPPER algorithm. Figure 4.3 illustrates the comparison of results given by the best model BiLevCSS(NN) with vanilla Neural Network.

The results shown in Table 4.5 highlight that the BiLevCSS model outperforms other rule based and associative classifiers on comparison. Next, we compared the three proposed strategies namely, BilevCSS (Ripper), BiLevCSS (NN) and BiLevCSS (SVM) with SigDirect. The results of this comparison are summarized graphically in Figure 4.4. The graph shows that BiLevCSS (NN) performs better than the rest, which proves that, when meaningful,

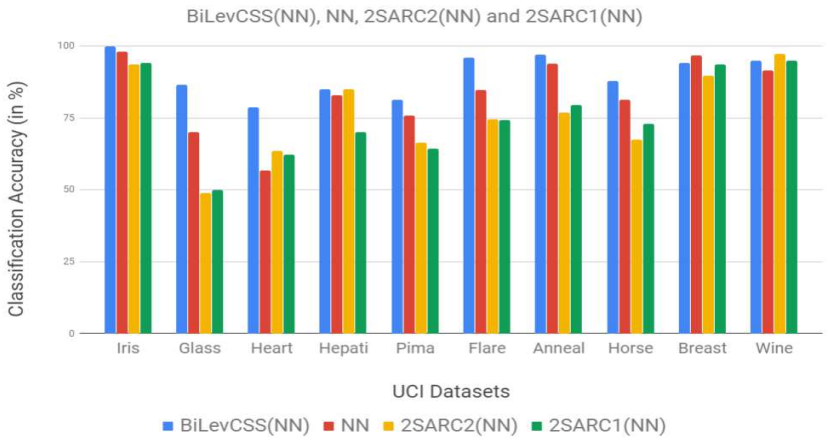

Figure 4.3: Comparison of classification accuracy for BiLevCSS(NN) with vanilla Neural Network, 2SARC1(NN) and 2SARC2(NN).

statistically significant and non-noisy rules are given to Neural Network, the classification accuracy of the classifier improves. The results obtained from BiLevCSS (Ripper) are motivating, however do not beat BiLevCSS (NN) in performance. Therefore, in the future we aim to evaluate more explanatory classification models in the second phase of learning, for a more explainable model since Neural Networks are more of a black box compared to Ripper.

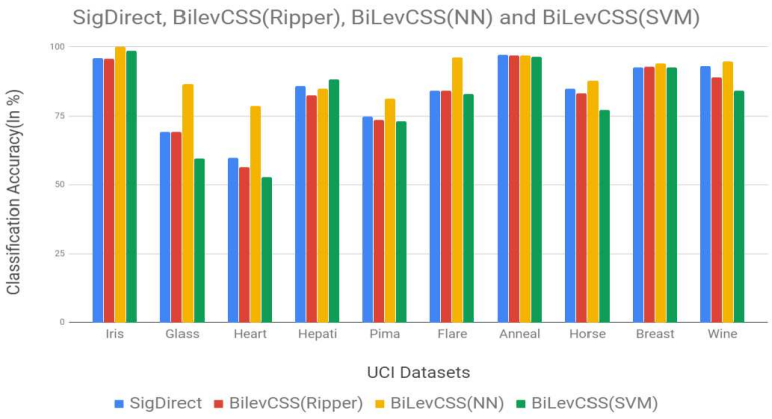

Figure 4.4: Comparison of classification accuracy for BiLevCSS(NN) with BiLevCSS(RIPPER), BiLevCSS(SVM) and SigDirect.

4.2.2 Statistical Analysis

The accuracy values report that BiLevCSS performs better for most of the datasets. To confirm this statement, we perform statistical analysis as shown in Table 4.6. We follow Demsar's study [18] and use Friedman's test to compare the statistical significance of the results obtained from the comparison of all the algorithms on the basis of classification accuracy. Since the p-value obtained from this test was less than the critical value (alpha) which is equal to 0.05, it proves that the results are statistically significant and the algorithms are significantly different from one another.

Furthermore, to investigate the statistical significant of the proposed algorithm with other contenders pair-wise, we perform another non-parametric test called Wilcoxon signed ranked test [18]. In this test, for every pair of algorithm in consideration, the difference of their classification accuracy, D_i is calculated to analyse the ranks based on the absolute values of these differences, $|D_i|$. Further, positive ranks R_i^+ and negative ranks R_i^- are calculated based on the original values of D_i for two algorithms. Adding up all the values of R_i^+ and R_i^-, $\mathrm{W_{stat}}$ is calculated as $min(\sum R_i^+, \sum R_i^-)$ which gives us the critical value Z. For alpha value equal to 0.05, the corresponding Z-value is -1.96, therefore, the null hypothesis is rejected if the obtained critical value Z is less than -1.96. Table 4.6 reports the p-values obtained by comparing the most accurate model, BiLevCSS(NN) against other classifiers using Wilcoxon test. We also compare the number of times the different algorithms win or lose against BiLevCSS(NN) and if there is a tie between them. The p-values obtained are less than 0.05 which show that BiLevCSS(NN) is statistically significantly better than all the contenders. The results show that the proposed BiLevCSS algorithm with Neural Network at the second stage of learning outperforms the rest of the algorithms by winning in at least 8 out of 10 instances.

Table 4.5: Comparison of classification accuracy of BiLevCSS with other state-of-the-art classifiers

Datasets	**BilevCSS (RIPPER)**	**BilevCSS (NN)**	**BilevCSS (SVM)**	**RIPPER**	**NN**	**SVM**	**C4.5**	**CBA**	**CMAR**	**CPAR**
Iris	95.72	**100**	**98.66**	94	98.09	94.6	94	94.67	94	94.7
glass	69.27	**86.60**	59.52	68.69	70.14	68.6	71.47	73.9	70.1	**74.4**
Heart	56.51	**78.64**	52.84	53.97	56.72	55.4	**61.5**	57.8	56.2	53.8
Hepati	82.57	**84.95**	**88.41**	78.06	82.89	79.3	79.25	81.82	80.5	79.4
Pima	73.64	**81.24**	73.2	66.36	**75.95**	74	73.7	72.9	75.1	73.8
Flare	84.27	**96.1**	83.1	72.13	**84.61**	73.8	82.1	84.2	84.3	63.9
Anneal	**96.93**	96.96	96.25	95.8	93.96	85	89.87	**97.91**	97.3	98.4
Horse	83.34	**87.78**	77.27	84.23	81.321	72.5	**85.04**	82.36	82.6	84.2
Breast	93.05	94.26	92.80	95.42	**96.83**	95.7	94.71	96.28	**96.4**	96
Wine	89	**94.94**	84.20	91.57	91.66	**94.9**	71.7	49.6	92.7	88.2
Average	82.43	**90.14**	80.62	80.02	**83.21**	79.38	80.33	79.14	82.92	80.68

Table 4.6: BiLevCSS(NN) compared to the rest of the algorithms on 10 UCI datasets

Classifiers	**Wins**	**Losses**	**Ties**	**P-value**
BiLevCSS(NN) vs BiLevCSS(SVM)	9	1	0	0.017
BiLevCSS(NN) vs RIPPER	9	1	0	0.007
BiLevCSS(NN) vs NN	9	1	0	0.013
BiLevCSS(NN) vs SVM	9	1	0	0.009
BiLevCSS(NN) vs 2SARC2(NN)	8	2	0	0.013
BiLevCSS(NN) vs 2SARC2(SVM)	10	0	0	0.005
BiLevCSS(NN) vs 2SARC2(RIPPER)	10	0	0	0.005
BiLevCSS(NN) vs 2SARC1(NN)	10	0	0	0.005
BiLevCSS(NN) vs 2SARC1(SVM)	9	1	0	0.007
BiLevCSS(NN) vs 2SARC1(RIPPER)	10	0	0	0.005
BiLevCSS(NN) vs BiLevCSS(RIPPER)	10	0	0	0.05
BiLevCSS(NN) vs C4.5	9	1	0	0.007
BiLevCSS(NN) vs CBA	8	2	0	0.013
BiLevCSS(NN) vs CPAR	8	2	0	0.013
BiLevCSS(NN) vs CMAR	8	2	0	0.013

Chapter 5

Ensemble Learning on the Associative Classifiers

In this section, we introduce the ensemble learning on the associative classifiers. Initially we describe the basic approach of bagging and boosting on wSigDirect and perform experiments to evaluate it. Further we introduce a novel **D**iverse **S**ubSp**a**ce **F**or **E**nsemble (DSAFE) approach to form a more efficient ensemble model with SigD2. Finally, we evaluate our approach and compare with other classifiers using the UCI datasets.

5.1 Bagging and Boosting on wSigDirect

5.1.1 Methodology

We perform bagging and boosting on the weak version of SigDirect, we call wSigDirect. While SigDirect is already a strong learner, we chose it over CBA as it gives a smaller number of rules. But we need to make it weaker to be used for ACbag and ACboost. We do this by further reducing the number of rules to be used for classification. The strategy for rule generation and rule pruning stays similar to that of the original SigDirect. However, for all the association rules obtained from the pruning phase for classification, we divide these rules as per the class label. Further, we chose the top η rules on the basis of highest confidence values from each class label group. The

classification model thus obtained is called weak as it does not involve all the significant rules. We perform bagging and boosting on the ensemble model of wSigDirect over different trained datasets for prediction.

Bagging:

ACbag is motivated by the approach proposed in [8]. The weak classifiers are learnt in parallel by picking instances randomly with replacement from the training data. Each wSigDirect model is learnt independent of each other. In bootstrap sampling, every observation has equal probability of appearing in the training dataset. Finally, we perform a majority voting over the results of the weak learners and predict the class label for each testing sample. The complete process has been shown in Figure 5.1. This approach helps in avoiding the problem of overfitting. Since, the base models are explainable, the ACbag can explain the responses of each learner, and the explanation of the ensemble would be the set of rules that were voted on by the ensemble. Furthermore, it was observed that the results obtained after performing bagging on wSigDirect are very comparable or slightly better than those achieved by bagging on the original SigDirect. Therefore, later in Section 5.1.2, we report the results of bagging on wSigDirect.

Boosting:

Boosting is a process of improving the performance of a weak learning algorithm. It is done under the assumption that, the performance of the weak learner is at least slightly better than random guessing on different observations. In this phase, we propose ACboost, which iteratively calls wSigDirect. This weak learner is converted to a strong learner either by weighted average of the predictions from weak learners or by considering prediction with majority voting. Given a training set, with features and class labels, we initialize the

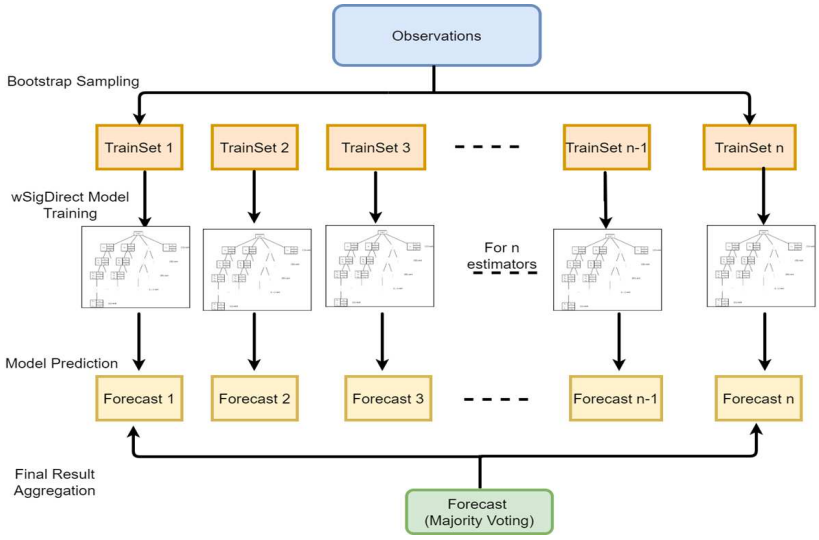

Figure 5.1: Bagging on wSigDirect

weights of our samples as one divided by the number of training instances. For the number of weak learners to be used sequentially, we train the first base learner using wSigDirect and obtain the misclassification error of the model. Further, the weight of the classifier is calculated based on its performance on the training data. Finally, the weight of each sample is updated, such that samples that were correctly classified are given less weights whereas the samples which were incorrectly predicted are given more weights. This would force the learner to pay more attention towards the incorrect predictions done by the previous learner. The iteration is continued till the maximum number of estimators (pre-set number of weak learners) are reached or a low training error is achieved. Finally, the prediction is done by using the weights of each classifier calculated previously to perform weighted prediction. This sequential learning of models helps in reducing the training error. Figure 5.2, shows the visual illustration of the followed process. We have used the methodology proposed for multi-class classification in SAMME algorithm [31], an extension of adaboost, which adds up a log term to the weight of the classifier making the boosting algorithm applicable for both two-class and multi-class classifi-

cation tasks. Furthermore, since the rules produced by the base classifier are explainable therefore, there is a possibility of interpretation of results.

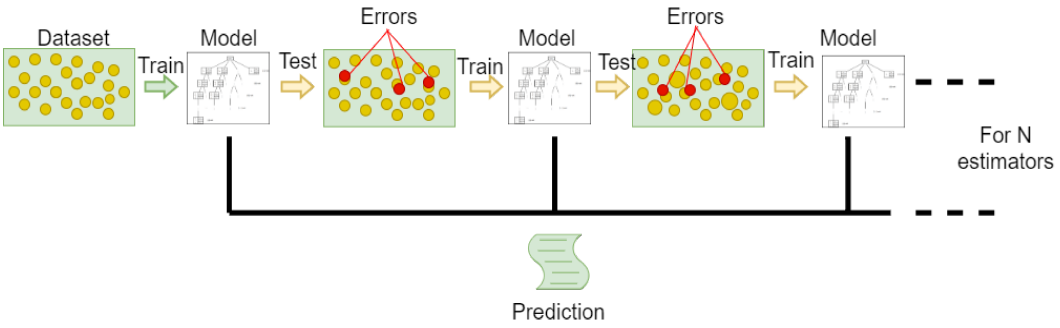

Figure 5.2: Boosting on wSigDirect

5.1.2 Experiments

We evaluate the performance of ACboost and ACbag on 15 UCI datasets [20]. We discretize the datasets as proposed in [13], so the classification accuracy might be marginally different from the previously reported results. We report the results after performing the average over 10 fold cross validation on each dataset. We use 90% of the total data as the train set and further divide the train set into train set and prune set in the ratio of 2:1.

Classification Accuracy

We compare the performance of the proposed classification models ACboost and ACbag with ANN, SVM, original SigDirect, SigD2 in Table 5.1. The best parameters as stated in [39] are used for SVM [16] and SigDirect, in order to have a fair comparison. The results for SigD2 are as obtained in Section 3.2. We vary the confidence threshold in the range of 30-50% for SigD2 depending on the dataset.

For ANN, we use a shallow network with one hidden layer. The number of nodes in the hidden layer are set as the average of number of input and

output nodes. The architecture may vary slightly with dataset, but we use ReLU (Rectified Linear Units) or sigmoid functions for activation and around 200 training epochs with a learning rate of 0.1. For ACboost and ACbag, the value of η is tuned in the range of 5-15 for every dataset. The number of estimators are varied in the range of 15-100 for each fold in every dataset and we report the best results. The value for parameters η and the number of estimators have been concluded after performing a sensitivity analysis on each of them.

Table 5.1 shows that ACboost outperforms all the classifiers including SigDirect, SigD2, ANN and SVM. Further, the performance of ACbag is better than SVM on 12 datasets with 3 loses, out of 15 datasets. However, in comparison with ANN, the ACbag wins 9 times with 1 tie out of 15 datasets. We have also tried to compare our approach with deep neural network (DNN) with 5 hidden layers. ACboost was found to perform better than DNN in 10 with 1 tie out of 15 datasets. However, since most of the considered datasets are not big enough to be used for DNN, the results might not be conclusive. Here we have also compared SigD2 with ANN and DNN and they are found to be at par with 7 wins and 7 loses with each of them on the basis of classification accuracy. This shows that associative classifiers can be quite competitive with the neural networks.

Number of Rules

As described in Section 3.2.2, SigD2 is found to obtain less number of rules for classification as compared to other state-of-the-art machine learning classifiers. On the similar lines, ACboost is said to be explainable as the base model called wSigDirect produces meaningful and readable rules. This leaves a room for the interpretability of the ensemble classification model. The ensemble model helps in determining the attributes which are of most indicative to determine a class.

Table 5.1: Comparison of classification accuracy of ACboost with ACbag, SigD2, SigDirect, ANN and SVM

Datasets	**SVM**	**ANN**	**DNN**	**SigDirect**	**SigD2**	**ACbag**	**ACboost**
Adult	75.8	75.66	**85.35**	84.1	83.59	84.74	**85.23**
Anneal	85	93.964	**97.6**	96.99	97.21	**97.43**	97.31
Breast	95.7	**96.83**	**96.48**	91.7	92.7	93.86	92.62
Flare	73.8	**84.61**	70.3	84.23	84.3	84.31	**85.35**
Glass	68.6	70.148	66.9	70.56	69.17	**72.01**	**76.96**
Heart	55.4	56.72	55.6	58.49	59.81	**61.33**	**63.74**
Hepatitis	79.3	82.89	83.07	85.83	**86**	85.18	**90.89**
Horse	72.5	81.321	80.9	81.23	85.03	**85.3**	**85.7**
Iris	94.6	**98.09**	95.8	94	96	94.66	**97.33**
Led7	73.6	69.64	68.63	73.78	73.81	**74.84**	**75.21**
Mushroom	**99.8**	**100**	**100**	**100**	**100**	**100**	**100**
PageBlocks	91.2	**95.42**	95.08	91.21	**92.18**	91.24	92.13
Pima	74	**75.95**	75.15	75.25	74.86	75.53	**75.55**
Wine	94.9	91.662	**97.62**	92.71	93.2	94.04	**98.85**
Zoo	92.2	93.192	89.94	91	89.18	**94.28**	**98.9**
Average	81.76	84.406	83.89	84.738	85.136	**85.91**	**87.71**

Table 5.2: Best and runner-up counts comparison from Table 5.1 on the basis of classification accuracy

Classifiers	**Best**	**Runner-up**
SVM	0	1
ANN	4	1
DNN	3	2
SigDirect	1	0
SigD2	1	2
ACboost	9	3
ACbag	1	6

Consider the example of mushroom dataset, the rule produced will be in the format -: (habitat = leaves) and (cap-color = white) $\rightarrow$ (class = poisonous), where feature name 'habitat' has value 'leaves' and feature name 'cap-color' has value equal to 'white'. This rule along with other similar rules can be further used in the classification phase to determine whether a mushroom is poisonous or not. Similarly for ACbag, the readable rules from the base classifiers can help in interpreting the results.

Statistical Analysis

In order to better evaluate the performance of proposed strategy with different algorithms over various datasets, we use Demsar's method [18]. The Demsar's method has been explained previously in section 3.2.3 as well. Firstly, we have performed Friedman's test to evaluate whether more than two samples

Table 5.3: Statistical analysis of Table 5.1

Classifiers	Wins	Losses	Ties	p-value
SigD2 vs ANN	7	7	1	0.510
SigD2 vs DNN	7	7	1	0.510
ACbag vs SigD2*	11	3	1	0.064
ACbag vs SVM*	12	3	0	0.005
ACbag vs ANN	9	5	1	0.140
ACbag vs DNN	8	6	1	0.140
ACboost vs SigD2*	12	2	1	0.002
ACboost vs SVM*	14	1	0	0.002
ACboost vs ANN*	10	4	1	0.016
ACboost vs DNN*	10	4	1	0.022

(*) indicates statistically significant results with a p-value of 0.05.

that are related. We obtained a p-value which is less than alpha (=0.05), for the algorithms mentioned in Table 5.2. This proves that the results are statistically significant and at least one of the samples is significantly different from other samples. Second, we have also performed Wilcoxon's signed-ranks test to evaluate the performances of proposed algorithms and the contenders in pairs.

The results in Table 5.3 show that, the results obtained from ACboost are found to be statistically significant than those of SigD2, ANN, DNN and SVM as p-value is less than the significance level of 0.05. Moreover, ACbag although performs better than SVM and SigD2, it has a comparable performance if not better than ANN, as the p-value is greater than 0.05. Thus, the results obtained in this section highlight the significance of the explainable models over the ones that are hard to interpret (ANN, DNN & SVM). Therefore, it can be concluded that SigD2 and ACboost are almost at par with other strong learners like neural network in terms of classification accuracy along with its ability to be interpreted using a limited number of rules.

5.2 DSAFE for SigD2 on High Dimensional Data

Previously proposed ensemble learning strategies, faced challenges on large data due to the inability of base learner that is SigDirect, for being unable to work on a high dimensional dataset. Therefore, in this section, we introduce the details about the proposed effective strategy for performing feature subspace selection in order to build a diverse ensemble model.

5.2.1 Methodology

SigD2 has the limitation of taking very long time for the rule generation on a large dataset. In order to deal with the high dimensional data, we propose a sampling method called as, **D**iverse **S**ubSp**a**ce **F**or **E**nsemble (DSAFE). The motivation behind DSAFE, is obtained from the cost sensitive adaptive random subspace ensemble approach given by Cao et al. in [11], which automatically finds the number of estimators for the ensemble model.

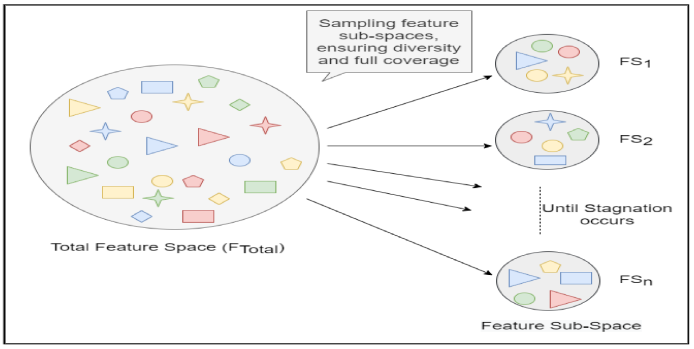

Figure 5.3: Visual Illustration of the basic approach followed in DSAFE method

Random Sampling is one of the most commonly used strategy for creating an ensemble of feature subspace. The raw input feature vector is divided randomly for a given size of feature vector defined as FS_k and the pre-determined number of estimators defined as η. The results from all the estimators are ag-

gregated in the end to obtain the prediction accuracy. Although such a model helps in preventing bias, however suffers limitation of setting up the parameter for number of estimators (η) to be present in the ensemble for classification. Even with the pre-defined number of estimators, the ensemble doesn't ensure the coverage of the complete dataset, and enough diversity among each of the sub-feature sample space. Infact, there might be some overlap between each of the feature sub-spaces which goes neglected, as there is no measure of how large or small it should be. The CSARS algorithm proposed by Cao overcomes some of these limitations, but calculates the overlap among the whole sub-dataset with in a certain subspace formed using bootstrap sampling. Also, there is scope of making the search of feature subspace more faster and efficient for large datasets.

We believe that, further careful selection of feature subsets can lead to better prediction performance both in terms of accuracy and speed. Also, we intend to use the DSAFE algorithm over the ensemble of SigD2.

Therefore, for a given large dimensional dataset, we split the data in a parallel manner, by randomly generating diverse feature subsets as described in Algorithm 4. DSAFE algorithm helps in overcoming the need of setting up the number of estimators to be given as a hyper parameter. In the proposed framework, space S randomly selects the sub features, based on the R_f (ratio of feature subspace), to form a new subspace FS_k, where k varies from 1 to the number of possible estimators. As shown in Figure 5.3, each feature sub space has a set of features selected randomly without internal repetition from the total available feature space. However, the attributes can be repeated externally in different sub-spaces. For the illustration purpose, only few features have been shown in each sub space, however in reality, this can be extended based on the required size of each sub space. Next, to determine whether the generated feature subsample is diverse enough or not, we use the threshold

T_over (Overlapping region threshold), whose value lies between 0 to 1 and is given as a hyper-parameter to the algorithm. The overlap rate of the each of the given feature subsample, can be determined using the equation given below, where $\mathrm{FS_i}$ and $\mathrm{FS_j}$ are feature subspaces and $\mathrm{N_{fea}}$ is the size of feature vector.

$$OverlapRate = \frac{FS_i \cap FS_j}{N_{fea}} \tag{5.1}$$

The hyper-parameter T_over, is a crucial parameter deciding the performance of the ensemble, if the value of T_over is too small, then the number of estimators produced would be too low, which kills the purpose of using the ensemble model. While if the T_over is too large then the feature subsets would lack diversity. Further as shown in Algorithm 6, if the overlap rate is greater than the overlapping threshold, then the feature subsample is not considered to be diverse enough, and hence is not considered for classification. On the other hand if the overlap rate is less than the overlapping threshold for all the subsamples in $DiverseFeatureSets$ then, we add the subsample to the list of samples and is considered further for classification.

We find the overlap only between the each of the feature set in the subspace, meaning only the overlap of the attributes to be selected and not the whole sub-dataset in a certain subspace.

Furthermore, in order to reduce the search space for feature subsamples, in DSAFE, we introduce a novel method called FSGen for generation of each subsample. As determined in Algorithm 5, given overlapping threshold T_over and length N_{fea} of subspace feature vector (say, $\mathrm{FS_k}$, where k is a variable) determined using the ratio of the feature subspace, we randomly generate attributes to be added to the feature subspace up to length equal to $N_{fea} * T\_over$, after this until the length of N_{fea} is obtained, we take two samples at a time and add them to the feature vector subset. The algorithm continues to add two samples to the feature subset and keeps checking for diversity up

till the complete feature subset $\mathrm{FS_k}$ is obtained. The motivation behind this approach is that, if the feature subset constructed up till the current length (which is greater $N_{fea} * T\_over$ but less than N_{fea}), is not found to be diverse, then the possibility of this feature vector being diverse on reaching the desired size of N_{fea} is very low, so we stop the search here itself. We keep checking for diversity every time we add two random attributes to our feature vector in order to reduce the searching time.

Next, we also perform an experiment to explore if using bagging on each sub dataset would help in classification process. So, we use the generated feature subsamples and create a bootstrap sample dataset with replacement $\mathrm{FS_n}$, based on the R_{S} (Ratio of bootstrap sample).

Further as described in Algorithm 6 , we introduce *sr* as another hyper-parameter called as the stagnation rate. Stagnation is said to occur, when feature space has been covered completely by the generated feature subsamples and no more diverse subsample of features can be created. This forms the breaking condition of our algorithm. Till all the diverse sets are generated, with maximum coverage on the given number of feature and while the stagnation has not occurred, this process continues. Further as shown in Algorithm 3 after the stagnation occurs, all the generated diverse sets are given to SigD2 for classification. For each diverse feature subset $\mathrm{FS_k}$ obtained from Algorithm 4, training sub-dataset FS_k^{train} is formed, and similarly for testing sub-dataset FS_k^{test} is generated. It is worth noting that the number of estimators for the ensemble of SigDirect are determined automatically by the above explained approach. Using the predictions from different estimators, we perform majority voting on them to aggregate the results. In the end, classification accuracy is determined using the results obtained from the majority of the models in the ensemble of SigD2.

The rudimentary principle involved in the above proposed framework that

is the wisdom of crowds, helps in building a strong and efficient classification model. Furthermore, we have also performed some experiments by using bootstrap sampling while forming the sub-datasets after the sub feature set has been defined.

Algorithm 3: Algorithm for DSAFE for SigDirect

Data: **Train Dataset**: Initial training dataset. **Test Dataset**: Initial testing dataset. $\mathbf{R_f}$: Ratio of the feature subspace. **T_over**: Threshold for the Overlapping region. **sr**: Stagnation Rate=100

Result: Predict class label of each instance in **Test Dataset**.

1 DiverseFeatureSets = GenerateDiverseFeatureSets(Train Dataset,R_f,T_over,sr);
2 **for** *each feature subset* FS_k *in DiverseFeatureSets* **do**
3 A training sample FS_k^{train} is created from the Train Dataset over feature subSpace FS_k.
4 Create subsample FS_k^{test} for TestSet using feature subspace FS_k.
5 ARCModel = SigD2(FS_k^{train})
6 Perform prediction on FS_k^{test} using the association rules in ARCModel
7 **end**
8 Generate the class label of each instance in Test Dataset by performing majority voting on the set of predictions, to aggregate the results from all the estimators.

Algorithm 4: Algorithm GenerateDiverseFeatureSets

Data: **Train Dataset**: Initial training dataset. $\mathbf{R_f}$: Ratio of the feature subspace. **T_over**: Threshold for the Overlapping region. **sr**: Stagnation Rate

Result: DiverseFeatureSets

```
change=0;
DiverseFeatureSets= {};
F_Total ⟵ Set of all features in the dataset.
while change<sr do
    FS_k = FS_Gen(F_Total, RF, T_over, DiverseFeatureSets) ;
     ▷ Generate a feature subset FS_k, using R_f, the given
     ratio of feature subspace.
    if IsDiverse(FS_k,DiverseFeatureSets, T_over )==True then
        Add FS_k to DiverseFeatureSets ;
        change = 0 ;
    else
        change+=1 ;
    end
end
```

Algorithm 5: FSGen - Algorithm for Feature subset generation

Data: $\mathbf{F_{Total}}$: Total feature space. $\mathbf{R_f}$: Ratio of the feature subspace.
T_over: Threshold for the Overlapping region.
Result: Returns FS_k - a feature vector subset.

```
1  N_fea = len(F_Total) * R_f
   ;    ▷ N_fea is the desired length of the FS_k feature subset.
2  FS_k=[]
3  while len(FS_k) != N_fea do
4  |  if len(FS_k) < (N_fea * T_over) then
5  |  |  Randomly select a feature from the available feature space
   |  |   F_Total and add it to the FS_k feature subset.
6  |  else
7  |  |  Randomly select two features, F_i and F_j from F_Total.
8  |  |  Check if adding these two features to FS_k would lead to a
   |  |   diverse feature subset or not. ;          ▷ One can keep the
   |  |   remaining feature values as dummy.
9  |  |  if IsDiverse( FS_k, DiverseFeatureSets, T_over )==True then
10 |  |  |  Append F_i and F_j it to the desired feature vector
11 |  |  else
12 |  |  |  return [] ;  ▷ This step is done to reduce the search
   |  |  |   space, as it is not worth adding elements to the
   |  |  |   feature subset which is not diverse enough after
   |  |  |   the length of N_fea * T_over have been
   |  |  |   discovered.
13 |  |  end
14 |  end
15 end
```

5.2.2 Experiments

In this section, we evaluate the performance of the cost adaptive feature selection for SigDirect, on 6 UCI datasets [20]. It should be noted that, among these datasets a few are those on which, SigDirect either could not give reasonable performance or faced time and memory issues. So, we test the proposed DSAFE for SigD2 on them as well.

As discussed before, we discretize the datasets as proposed in [13], so the classification accuracy might be marginally different from the previously reported results. We report the results after performing the average over 10 fold cross validation on each dataset. We use 90% of the total data as the train

Algorithm 6: Algorithm IsDiverse

Data: $\mathbf{FS_j}$: Feature subsample to be checked if diverse or not. **DiverseFeatureSets**: Dictionary containing all the feature subsets to be considered for classification. **T_over**: Threshold for the Overlapping region.

Result: Bool :- **True** or **False**

$N_{fea} = length(FS_j)$

for *each feature subset* FS_i *in DiverseFeatureSets* **do**

$$Overlap\_rate = \frac{FS_i \cap FS_j}{N_{fea}} \tag{5.2}$$

  if $Overlap\_rate > T\_over$ **then**

    return **False** ; ▷ `This means that the overlapping rate was greater than the threshold for at least one feature subset in the DiverseFeatureSets, and hence this should not be considered for classification`

  else

    Continue

  end

end

return **True**;

set and further divide the train set into train set and prune set in the ratio of 2:1. The hyper-parameter, such as the ratio of feature subspace R_F, has been varied based on the size of the dataset. Due to the limitation of SigDirect, for not being able to handle very large size of data, we extend it from 30% to 70% of the total feature size, while not extending maximum size of 25 attributes at a time. The maximum size has been concluded after performing the sensitivity analysis based on the experiments done in the previous sections. This is essentially done to obtain the maximum benefit out of the proposed explainable classifier, while accounting the limitation of the associative classifiers. Also, as described in previous section, we vary the confidence threshold hyper-parameter value for the pruning stage of SigD2, in the range of 30-50% selected after performing the sensitivity analysis.

Next, the overlapping region threshold defined as T_over, is varied in the

range of [0.10, 0.80], the stagnation rate is set to 100 in the algorithm. For this set of experiments, we conclude that the maximum of estimators are 100, we assume that a maximum of 100 estimators are reasonable enough to conclude the prediction. In order to perform a fair comparison, we use 100 estimators while using random sampling approach for the selection of feature subset for the ensemble of SigD2. Furthermore, we have explored two approaches, one without bootstrapping the samples, i.e we only split using feature selection strategy of DSAFE, without performing bootstrap sampling over the number of samples, while for the other one we perform experiments with feature selection and performing bootstrap sampling with replacements on the samples used to form the sub-datasets corresponding to those feature samples.

Table 5.4: Comparison of classification accuracy using different values of overlap rate threshold

T_OVER	IONOSPHERE	CYLBANDS	SOYABEAN	ADULT	MUSHROOM
0.1	88.05	84.63	69.07	78.5	98.19
0.2	89.16	79.42	67.78	79.3	95.91
0.3	89.72	81.88	62.77	79.2	97.6
0.4	91.11	87.1	68.88	79.3	97.78
0.5	91.66	90.44	78.7	**82.76**	**100**
0.6	92.58	87.31	**83.83**	81.07	99
0.7	**93.05**	**91.3**	82.59	81.52	98.73
0.8	91.94	84.6	83.7	80.75	98.67

Based on the results obtained from the considered UCI datasets and as shown in the graph 5.6, we hypothesize that the proposed approach leads to the best prediction performance, when T_over is in the range of [0.5,0.7]. It can be concluded that a very small value of T_over would produce very less number of estimators, while very large value of T_over would lead to lack of diversity among the large set of estimators produced.

Therefore, we use another UCI dataset, which is PenDigits, to evaluate its classification performance in the given range. It was found that, we obtain classification accuracy of 82.63%, which is greater than the original SigDirect.

Table 5.5: Description of Datasets

Dataset Name	#Attributes	#Instances	#Class
Soyabean	35	307	19
Ionosphere	34	351	2
Cylbands	39	512	2
Mushroom	22	8124	2
Adult	14	48842	2
PenDigits	16	10992	10
Arrhythmia	279	452	16

Also, it is worth noting that the given approach is well scalable on the large sized datasets such that the classification time has reduced a lot as compared to the original SigDirect, which is not found to perform very well on the large sized datasets in terms of runtime of the algorithm. We have also performed a sanity check to confirm our claim that, all the features would be covered at least once in the ensemble model. As described in Figure 5.5, it was found that, if we perform early stop on our algorithm once all the features in F_{total} are covered at least once, then the prediction performance is much less, hence proving the efficacy of our proposed approach.

Table 5.6: Comparison of classification accuracy using Random Sampling vs DSAFE for SigD2

Datasets	Random Sampling	DSAFE
Soyabean	87.75	91.3
Ionosphere	91.66	93.05
Cylbands	77.77	83.3
Mushroom	92.3	100
Adult	81.29	82.76
PenDigits	81.45	82.59

We further extend our approach and perform bootstrap sampling with replacement on FS_k^{train} on each FS_k obtained from $GenerateDiverseFeatureSets$ function. The results are as described in the Table 5.4. Next, we have also compared our approach DSAFE with the original SigDirect, MLP , SigD2 using Random Sampling, DSAFE for SigD2 with Booststrap sampling. It was found that, out of 6 datasets, there was 1 tie and 3 wins of DSAFE for SigD2 and 2 wins of MLP. The new sampling approach not only gives better accuracy than SigD2 but is also faster in terms of run time.

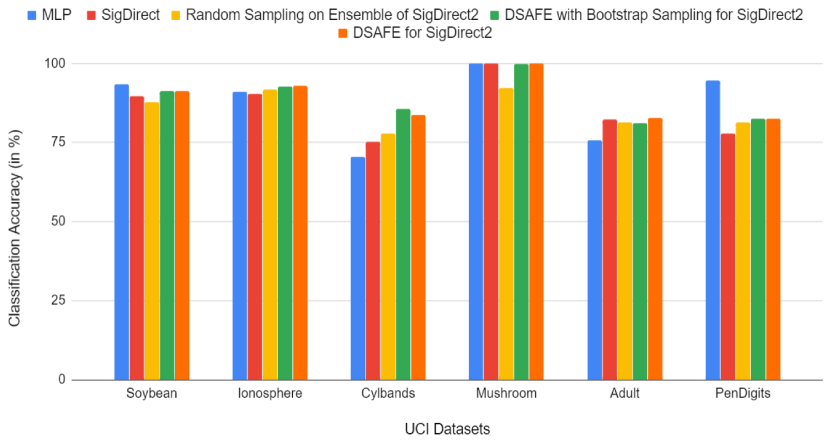

Figure 5.4: Comparison of classification accuracy on MLP, SigDirect, Random Sampling on Ensemble of SigD2, DSAFE with Bootstrap Sampling for SigD2, DSAFE for SigD2 on different UCI datasets.

On comparing the two approaches that is, DSAFE for SigD2 and DSAFE for SigD2 using Booststrap sampling, it was found that the former is slightly better in performance than the latter. We have been able to get better accuracy for DSAFE than on using Random Sampling for all the considered datasets, which proves the claim that careful selection of feature subsets can lead to better results. For few of the datasets used above, although we could use SigDirect alone for classification, but the algorithm takes fairly large running time as the tree generation takes large amount time and space. However, the proposed approach DSAFE for SigD2 is found to be much faster than using any version of SigDirect alone.

Furthermore, we have also performed an experiment on a high dimensional Arrhythmia Data Set, which has 279 attributes. This dataset could not be handled by SigDirect because of its limitations on high dimensional data, however, Diverse Subspace For Ensemble (DSAFE) on SigD2, could handle this data set and provided an accuracy of 65.21% which is very competitive with 67% obtained from MLP classifier used in Weka [35].

We have also evaluated the proposed approach and compared it with other

approaches using the Wilcoxon's exact test to perform statistical analysis as explained in previous sections. The results obtained using DSAFE for SigD2 are found to be statistically significant than original SigDirect alone, with a p-value of 0.043 which is less than the critical value of 0.05. Moreover, the results obtained from the proposed DSAFE approach are statistically significant than those obtained from Random Sampling approach for feature selection over SigD2 with p-value of 0.028. On comparison with MLP, although the results were not found to be statistically significant, they seem to be quite competitive.

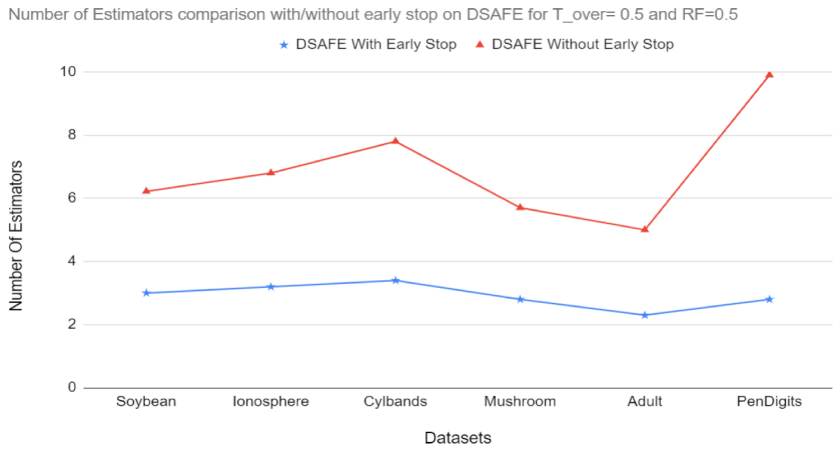

Figure 5.5: Number of Estimators comparison with/without early stop on DSAFE for T_over= 0.5 and RF=0.5.

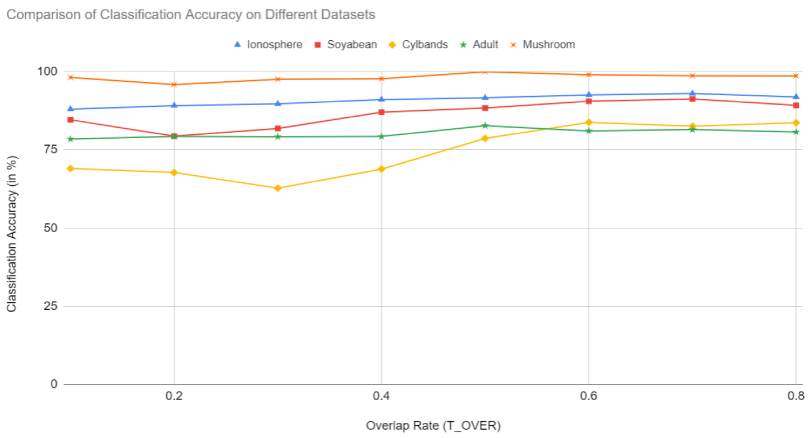

Figure 5.6: Comparison of Classification Accuracy on Different Datasets at different Overlap Rate.

5.3 Application of Associative Classification model

In this section, we discuss the application of associative classification models. Most of the industrial applications of machine learning, demand explainability of the results obtained. Specially in health-care, practitioners find it hard to trust artificial intelligence until they can verify the results obtained, as the result of the incorrect predictions can be catastrophic and is reprimanded.

Dealing with the COVID-19 pandemic, we have tried to use our proposed classification model for the diagnosis of COVID-19 using the clinical spectrum [19]. The data set has 5644 instances and 111 attributes. The goal here is to determine weather the patient should be admitted to the a general ward, semi-intensive unit or intensive care unit, based on the health conditions of anonymous patients reported by various medical tests.

The data provides details about the medical data of the patient, like Haemoglobin, Platelets, RBS count, Pneumonia etc, but it is found to be imbalanced and has a lot of missing values. Therefore, we have performed extensive cleaning and pre-processing on the available data. We have also

used SMOTE [12] to increase the samples and balance the dataset. Next, we apply our two classification models: SigD2 and DSAFE algorithm on SigD2 to perform the prediction task. The results as shown in Table 5.7 describes

Table 5.7: Comparison of classification accuracy using SigD2, DSAFE for SigD2, RIPPER, SVM and MLP

Classifier	**Accuracy**
SigD2	76.85
DSAFE for SigD2	**84.52**
Ripper	78.65
SVM	78.75
MLP	76.88

that our proposed strategies are quite competitive with the state-of-the-art classifiers. DSAFE for SigD2 gives 84.52% classification accuracy which the best among all the contenders. Promising results have proved that, associative classification is highly competitive with the deep learning models and also provides explanation of the obtained results at the same time. Association based classification models find their immense usage in medical applications. Note - Based on the community interest, for the greater good, the COVID-19 dataset was presented in Kaggle for embarking information to the interested research groups and not as any competition. However, based on one of the posted notebooks (as of 29 June 2020), our results were found to be very competitive and in fact better than the ones that claimed to obtain 80% accuracy using Grid Search Stratified KFold Cross Validation on SVM Classifier [36].

Chapter 6

Multilayered framework for SigD2

DNNs have introduced the world to the sophisticated deep layered architectures, that generally obtain good results. Researchers conjecture that replicating the idea of performing representation learning from neural networks and processing the raw features, layer by layer in a multi-layered architecture can help in building a deep model for other classifiers like decision trees and associative classifiers. In this Chapter, we first introduce the background of the approach used by Zhou et al. for gcForest. Later, in this section, we will introduce the architecture of multi-layered system for SigD2 and then discuss a few experimental results that show potential for future research.

6.1 Background

Deep Neural Networks have garnered all the attention due to its highly efficient performance, despite of certain limitations like it requires a cumbersome process of hyper-parameter tuning. Moreover, it is not found to perform well when training dataset isn't large enough and many a times it gets hard to get the large amount of labelled data. The complex deep networks also requires high computational machines which might not be available always due to monetary constraints.

Zhou et al. came up with a competitive alternative approach of using the ensemble of decision trees in [57]. They build a deep model called gcForest, using an ensemble of decision trees as an alternative to neural networks, and dealing few of the above mentioned limitations. The gcForest is found to have the key features of neural networks like, representation learning, model complexity and ability to deal with high dimensional data. The authors claim that the hyper parameters are quite robust and do not require much tuning unlike neural networks, where classification performance is significantly dependant on the chosen hyper-parameters. The framework works with a combination of Cascade Forest and Multi-grained scanning, that manifest representation learning like DNN. The cascade forest processes the features and transforms them layer by layer. The gcForest uses four decision forests each of different type, two completely random tree forest and two random forest. The number of trees in a forest are varied as a hyper-parameter.

If the size of features vector becomes very large, they use random sampling to create an ensemble of ensemble and consequently process the task of prediction. Each of the forest outputs the estimate of class distribution, which are concatenated to the original feature vector and are given as input to the next level in Cascade phase.

The process of multi-grained scanning is done prior to cascade forests, so as to determine the relationship among features. This step uses sliding window of a certain size, moving across the raw feature vector. For the purpose of experiments, authors have used three different window sizes. Each of the sub feature space from each of the sliding window are given as input to the ensemble of forests and the class vectors generated in the output are concatenated to form the transformed feature set. This in turn is used for training in cascade forest phase. Finally, the class label is determined by taking the maximum of the aggregated results from all the four forests in the last layer. The number

of layers in cascade forest are determined automatically, as training stops only when there is no improvement in classification performance. This eliminates the need to pre define the number of layers in Cascade phase, unlike deep neural networks, where it is pre-defined fixed parameter. There are certain experiments done with and without multi-grained scanning phase, to emphasis its significance. The authors have tested their approach for image categorization, face recognition, audio data classification, sentiment classification and a few experiments on the UCI-datasets.

The results obtained were found to be quite competitive with deep neural networks while using very less and robust hyper-parameters that do not require much tuning. In the following section, we have made an attempt to build a deep model for associative classifiers in the similar lines.

6.2 Methodology

We hypothesise that few of the many factors that help deep learning achieve high accuracy, is its complex layered architecture. Therefore, in order to increase the performance of the proposed associative classifier, we mimic the neural network style in terms of complexity and layered architecture, we have designed a multi-layered framework for SigD2. This study has been motivated by the architecture used by Zhou et al. in [57] for Deep Forest which opens the potential of alternatives to deep neural networks. However, we have modified the framework with using small number of learning models in the multi-grained phase and using ensemble of ensemble in the cascade phase. Moreover, we use an associative classifier instead of random forest in the framework. We believe that SigD2 would require very low tuning and can be competitive in performance.

Initially, the given raw input feature vector, is passed into the multi-grained scanning phase as shown in figure 6.1, where we use three different sliding

window to cover the whole high dimensional feature vector. We use three different windows each of different size ws_k, where k is a variable. As shown in Figure 6.1, sliding window of size ws_k generates feature subsets which are given as an input to the SigD2 classifier, in multi-grained scanning phase. The probability distribution for each of the class is obtained as the output from each SigD2 which are further concatenated to form vector $\mathrm{FS_{ws_k}}$. Similarly, the results from different window sizes ws_k are finally concatenated to form $\mathrm{FS_{total}}$. This $\mathrm{FS_{total}}$, is further given as an input to Cascade SigDirect phase. Please note that Figure 6.1 assumes there are three output classes, so therefore for a total of 100-dim raw input feature vector from a dataset of size $m * n$, where m is the number of instances and n is the number of attributes, we obtain $\mathrm{FS_{ws_k}}$ of size equal to 273 for $n = 100$ and window size $ws_k = 10$. This is calculated as, for a 100 sized vector, a overlapping sliding window moves one attribute ahead each time, therefore giving, $(100 - 10 + 1)$ which is equal to sliding 91 times. Next, since the number of output classes considered here are three, therefore the size of transformed vector would be $(91 * 3)$ which equals to 273 as stated before. Similarly, we get 243-dim vector for $ws_k = 20$ and 213-dim vector for $ws_k = 30$. All the sub-features are concatenated for form $\mathrm{FS_{total}}$ of size equal to 729. This transformed vector is then given to the Cascade phase as an input.

Within the Cascade phase there are four ensembles of SigD2. We have chosen four ensembles to prevent biased results. Moreover, since SigD2 has the limitation of not being able to work on high dimensional datasets, we use an ensemble of SigDirects and call it ESigDirect. For the initial set of experiments, we split the large dimensional data using random sampling on the features with replacement for the fixed number of estimators to generate the ensemble. Each ESigDirect is an ensemble of 100 SigD2 based learners.

We use four ESigDirects in the Cascade SigDirect phase, such that the

information obtained from the initial multi-grained phase is transformed layer by layer in the cascade phase. Next, we also perform the bootstrap sampling with replacement on the samples of each input data used for ESigDirect.

The average of class probabilities obtained from different SigDirect's in each of the four ESigDirects in that layer are concatenated with the input feature FS_{total}. Please note that each ESigDirect outputs the average of probabilities of each class, obtained from different SigDirects in the ensemble. As described in Figure 6.1, the FS_{total}, is concatenated to the output obtained at each cascade level. Since there are three output classes and four ESigDirect's used in the shown framework, the size of transformed vector turns out to be $729 + 12$ which is equal to 741.

The process is repeated for η layers using the validation set, until its performance stops improving or the set number of layers η, after which the training process stops. Now, for the test set, similar operations are performed in multi-grained scanning and cascade forest phases for η layers, and in the end, the majority voting is done based on the prediction probabilities obtained in the last layer. This approach seems promising, however, still needs to be explored more, especially with very high dimensional data. In future we would also like to study the possibility of interpreting the results even in a multi-layered framework.

6.3 Experimental Results

We evaluate the deep multi-layered framework for SigD2, designed so far on the UCI datasets. The classification accuracy might be slightly different from the previously reported results as we discretize the datasets [13]. We report the results after performing the average over 10 fold cross validation on each dataset. We use 90% of the total data as the train set and further divide the train set into train set and validation set in the ratio of 2:1.

We compare the performance of the proposed classification model with SigD2 and DNN. We also compare our results with the original gcForest as proposed by Zhou et al. We use different size of windows varying in the range of 7 to 30. We have chosen this range after performing the sensitivity analysis and also higher window size would slow down the performance of SigD2 while less than 7 might not be fruitful enough for associative classification. This set of values for ws was also chosen after performing sensitivity analysis and noting that the size of datasets considered so far are not very large. Please note that this range has been considered noting the inability of SigD2 to be able to run on very high datasets. The value of the maximum number of possible layers η, is set to 5. after performing sensitivity analysis for this experiment. Further in the cascade SigDirect phase, four ESigDirects are used with 100 SigD2 models in each of them. All the other hyper-parameters for the base classifier SigD2, remain the same as obtained from Section 3.2. For deep neural network, we have performed experiments with different architectures having different number of hidden layers. For DNN5, we use a deep network with five hidden layers. The number of nodes in the hidden layer are set as the average of number of input and output nodes. We use ReLU (Rectified Linear Units) or sigmoid functions for activation and around 200 training epochs with a learning rate of 0.1. For DNN1, DNN2, similarly we use a deep network with one and two hidden layers respectively. Please note that the number of hidden layers can further be extended based on the size of the dataset. The main aim here is to compare the performance of proposed approach with different architectures of the deep networks.

Table 6.1: Comparison of the proposed multi-layered SigD2 with other contenders on the basis of the classification accuracy on two UCI Datasets

	SigD2	**DSAFE for SigD2**	**Multi-layered SigD2**	**DNN1**	**DNN2**	**DNN5**
Adult	83.59	82.76	85.29	76.24	77.3	85.35
Ionosphere	90.3	93.05	94.28	92.95	94.36	95.77

For comparison with original SigD2 alone, we have performed the sensitivity analysis on the confidence threshold and have chosen to vary the threshold value in the range of 30-50% depending on the dataset. The configuration for gcForest remains the same as stated by Zhou et al. in [57].

We have used two UCI datasets for this set of experiments, which are Adult and Ionosphere dataset due to the following reasons. Firstly, because Ionosphere data set is found to have reasonable amount of attributes and the Adult dataset has reasonably high number of data instances. Secondly, DSAFE for SigD2 could not improve the classification performance when compared with original SigD2 for the Adult dataset and third, it has also been used by Zhou et al. for testing the performance of the gcForest algorithm.

For ionosphere dataset we obtained 94.28% classification accuracy which is very competitive to DNN5 which gets 95.77% accuracy. The performance is in fact found to be better than original SigD2 which obtains 90.3% accuracy. Furthermore, we have tested on another UCI dataset called the Adult dataset, and it was found that the proposed classification model gives 85.29% accuracy in comparison to DNN5 which gives 85.35% and SigD2 gives 83.59% accuracy on this dataset.

The gcForest achieves 86.40% accuracy for the Adult dataset as stated in [57], but we do no consider this for comparison with our approach, as the datasets that we have used are discretized [13], consequently their might be slight difference in the accuracy obtained. Since we were not able to reproduce the results for gcForest [24], we only consider the approximate idea from the results stated by the authors for gcForest in the paper.

In [57], the authors have only used a two layered network for comparison, which is is not fair as it is a weak contender and our experiments have shown that, the prediction performance tends to increase up till a certain number of hidden layers. So the advantage of the approach proposed by gcForest is

not as huge as claimed in the paper. In fact, we believe that the difference of performance of gcForest and DNN would actually be even lesser in reality as the layers of deep network would increase. Furthermore, the authors have not stated the number of layers of cascade phase obtained while testing on the UCI dataset. After working on the framework of multi-layered SigD2 similar to that proposed for gcForest, it has been observed that, they tend to require a lot of resources, both in terms of time and computing. Although these techniques may lead to slight increase in performance, but at the same time, they are found to require high computation as well. We conjecture that the architecture still has the scope of improvement.

As shown in Table 6.1, the multi-layered framework is performing better than both SigD2 and DSAFE for SigD2 and is also found to be quite competitive with the different deep network architectures. Although the results seem to be interesting for the small datasets, we still need to improve the algorithm and make it more efficient to be able to deal with very high dimensional data. As adding up the features in the proposed layer by layer model would increase the number of features to be transformed in each layer and finally to be used for classification.

For the purpose of this thesis, we leave the multi-layered framework for associative classifiers as an open problem. In future, we aim to rank the features in order of correlation before multi-grained scanning phase and apply the DSAFE approach to form an ensemble of ensemble on SigDirect in a layered cascade framework.

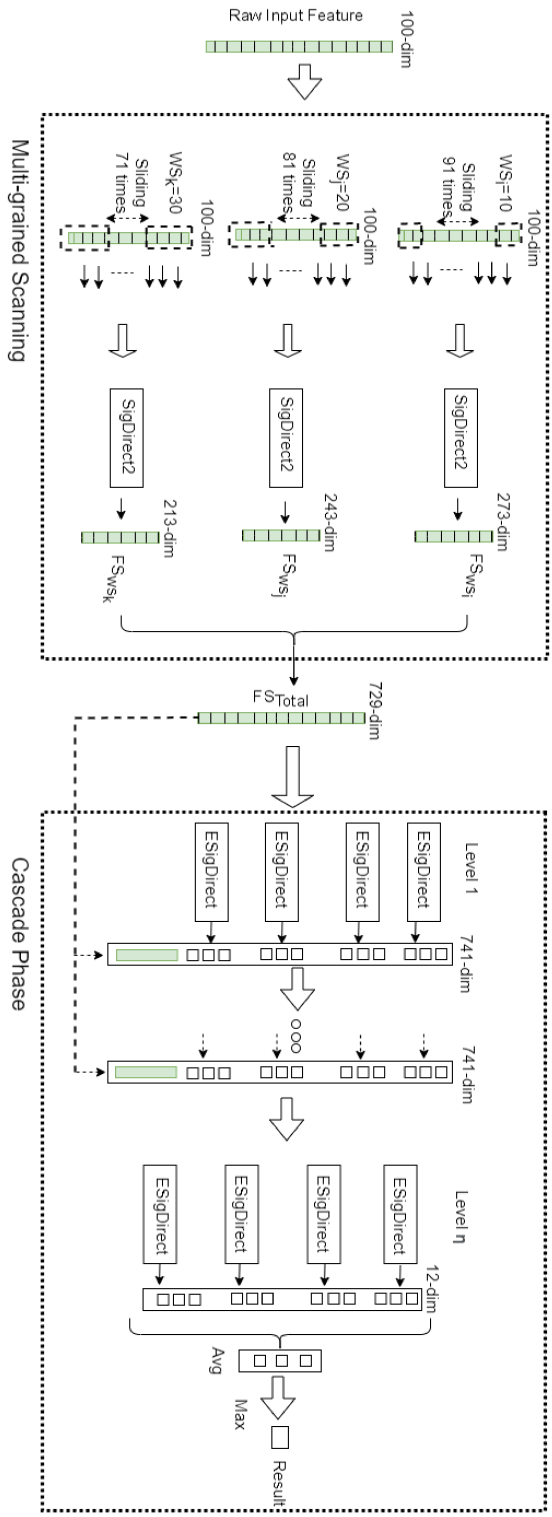

Figure 6.1: Multi-Layered framework for SigD2 based Classification Model. Note - For the purpose of illustration, the window size {10,20,30} is chosen, it is assumed that their are 3 output classes for the considered data and each ESigDirect is an ensemble of 100 SigDirects formed using random sampling on data.

Chapter 7

Conclusion

7.1 Summary

To summarize, in this study, we first propose an optimized two stage pruning strategy for SigDirect. It was found that the number of rules to be used for classification could be noisy, and the noisy rules would participate more often in misclassifications, adversely effecting the classification performance. Therefore, the proposed competitive associative classifier, SigD2 builds a rule-based model that is explainable, readable with minimal number of rules. The classifier initially performs a rule generation step to generate statistically significant class association rules. This is followed by a novel two phase rule pruning strategy used to obtain remove all noisy rules that would otherwise hamper the classification performance. These rules are further used for classification purpose. The proposed rule pruning strategy is found to reduce the rule set to a significantly small number, making it more useful for various applications especially in the field of bio-medicine where model interpretability is important. The proposed approaches outperforms other association and rule based classifiers and it is at par with the other supervised classifiers like ANN and SVM, which are black boxes and do not provide interpretable classification models. Unlike them, SigD2 is an explainable classifier.

Further in this study, we have introduced a novel approach BiLevCSS,

a two level classifier built on statistically significant dependent CARs. The proposed classification model consists of four steps of rule generation, rule pruning, transformed feature extraction for the next phase using the obtained rules and finally, the prediction on the learned model using Neural Network in the second stage. Rule generation leads to the generation of all statistically significant rules which are further used to train a second classification model to select appropriate rules. Since, these rules might be noisy with some irrelevant information, they are pruned using the instance-centric rule pruning strategy. Furthermore, the features are extracted using rule based or class based techniques. Finally, the classification is done by using the learned NN, SVM or RIPPER in the second level. The idea of using statistically significant rules has made our algorithm more efficient by selecting only valuable CAR and providing new features for the second stage. The experimental results obtained are very encouraging.

Furthermore, we also propose ensemble learning on SigDirect. We first introduced ACbag and ACboost which are the bagging and boosting algorithms on wSigDirect respectively. The ACboost algorithm which uses an ensemble of wSigDirect, to build a strong learner that boosts the prediction performance. While ACbag, produces an ensemble of wSigDirect using bootstrap sampling with replacement on the data points. The results obtained from ACboost were found to be the promising as compared to ACbag and were found to be competitive with ANN and DNN on a reasonable size of data.

Another huge challenge dealt in this study, is the inability of associative classifiers to deal with high dimensional data. We have tried to overcome this limitation by proposing DSAFE algorithm as a thoughtful approach to design an ensemble of SigD2. DSAFE sampling algorithm, vertically splits the feature space into diverse feature subspaces and use this to built an association based learning model. DSAFE ensures that there is minimum overlap between

various subspaces and verifies that the complete feature space is covered until no other diverse subspace could be generated. The DSAFE algorithm on SigD2 gives high classification performance and is competitive with other prevalent machine learning classifiers. We have also tried to test the application of our proposed approaches on the hospital data available in Kaggle for a prediction task related to the current COVID-19 pandemic. The results obtained seem very encouraging. The experimental results obtained, have highlighted the significance of associative classifiers and have proven them to be effective. We have also tested a multi-layered architecture for an associative classifier which although compromises on explainability, but can serve as an alternative approach to deep networks. The results seem to be encouraging, however, our study concludes that the multi-layered architecture still has some scope of improvement, in order to make it efficient for high dimensional datasets.

In this era of deep learning, where the models thus produced are black box, non intuitive and hard to interpret, the rule based association models can be used to build a more trustable model. The associative classifiers have an added advantage of being a white-box, such that the results obtained can be explained. Therefore, the proposed models, find their immense use in the industry, where practitioners find it hard to trust AI and demand for the validation of the obtained results.

7.2 Future Work

- In the future, we aim to experiment our algorithm by incorporating more features other than support, confidence, lift and p-value in Bi-level classification model. We would also like to evaluate the performance of our model with explainable associative classifiers in the second stage of learning. We would also extend our work for multi-label classification.

- Since, the results obtained are very encouraging, we would also like to work on making the SigD2 classifier more efficient in terms of rule generation phase. Although, we are able to compete with RIPPER in terms of accuracy, the number of rules for RIPPER are still smaller. In future, we also intend to identify rules that are noisy and can be potentially removed.
- In future, we also intend to use our proposed approach on various healthcare related applications where explanation of prediction is required. Furthermore, since SigD2 produces human readable rules, we would like to study the possibility of injecting human expert knowledge to the obtained rules in order to further improve the prediction performance.
- The multi-layered architecture of SigD2 can be a potential area of research and explored in future to build an alternative to deep networks. Finally, we believe that ensemble learning models with explainable base models have the scope for the explainability of results, which is left as open problem here and would be dealt future.

References

[1] Agrawal, R. and Srikant, R., "Fast algorithms for mining association rules," in *International conference on very large data bases (VLDB)*, 1994, pp. 487–499.

[2] Antonie, M. L., Zaiane, O., and Coman, A., "Associative classifiers for medical images," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, 2002, pp. 68–83.

[3] Antonie, M.L., Zaiane, O., and Holte, R.C., "Learning to use a learned model: A two-stage approach to classification," in *Sixth International Conference on Data Mining (ICDM'06)*, IEEE, 2006, pp. 33–42.

[4] Antonie, M.L. and Zaiane, O.R., "Text document categorization by term association," in *IEEE International Conference on Data Mining (ICDM)*, 2002, pp. 19–26.

[5] Antonie, M.L. and Zaïane, O.R., "An associative classifier based on positive and negative rules," in *Proceedings of the 9th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, ACM, 2004, pp. 64–69.

[6] Arunasalam, B. and Chawla, S., "CCCS: A top-down associative classifier for imbalanced class distribution," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006, pp. 517–522, ISBN: 1-59593-339-5.

[7] Beale, H. D., Demuth, H.B., and Hagan, MT, "Neural network design," *Pws, Boston*, 1996.

[8] Breiman, L., "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[9] Brown, G., "Diversity in neural network ensembles," PhD thesis, 2004.

[10] Brown, G., Wyatt, J., Harris, R., and Yao, X., "Diversity creation methods: A survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[11] Cao, P., Zhao, D., and Zaiane, O., "Cost sensitive adaptive random subspace ensemble for computer-aided nodule detection," in *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, 2013, pp. 173–178.

[12] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P., "Smote: Synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[13] Coenen, F., *The lucs-kdd software library*, 2004. [Online]. Available: `http://cgi.csc.liv.ac.uk/$%5Csim$frans/KDD/Software/`.

[14] Coenen, F. and Leng, P., "An evaluation of approaches to classification rule selection," in *Fourth IEEE International Conference on Data Mining (ICDM'04)*, 2004, pp. 359–362.

[15] Cohen, W., "Fast effective rule induction," in *International Conference on Machine Learning*, Elsevier, 1995, pp. 115–123.

[16] Cortes, C. and Vapnik, V., "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] Dasarathy, B. V. and Sheela, B. V., "A composite classifier system design: Concepts and methodology," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 708–713, 1979.

[18] Demšar, J., "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine learning research*, vol. 7, pp. 1–30, 2006.

[19] *Diagnosis of covid-19 and its clinical spectrum.* `https://www.kaggle.com/einsteindata4u/covid19`.

[20] Dua, D. and Graff, C., *UCI machine learning repository*, 2017. [Online]. Available: `http://archive.ics.uci.edu/ml`.

[21] Emerson, P., "The original borda count and partial voting," *Social Choice and Welfare*, vol. 40, no. 2, pp. 353–358, 2013.

[22] Exarchos, T. P., Papaloukas, C., Fotiadis, D. I., and Michalis, L. K., "An association rule mining-based methodology for automated detection of ischemic ecg beats," *IEEE transactions on biomedical engineering*, vol. 53, no. 8, pp. 1531–1540, 2006.

[23] Freund, Y. and Schapire, R.E., "Experiments with a new boosting algorithm," in *International Conference on Machine Learning*, vol. 96, 1996, pp. 148–156.

[24] *Gcforest v1.1.1*, `https://github.com/kingfengji/gcForest`.

[25] Hamalainen, W., "Efficient discovery of the top-k optimal dependency rules with fisher's exact test of significance," in *2010 IEEE International Conference on Data Mining*, 2010, pp. 196–205.

[26] Hämäläinen, W., "Kingfisher: An efficient algorithm for searching for both positive and negative dependency rules with statistical significance measures," *Knowledge and information systems*, vol. 32, no. 2, pp. 383–414, 2012.

[27] Hämäläinen, W. and Nykänen, M., "Efficient discovery of statistically significant association rules," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 203–212.

[28] W. Hämäläinen, "Statapriori: An efficient algorithm for searching statistically significant association rules," *Knowledge and information systems*, vol. 23, no. 3, pp. 373–399, 2010.

[29] Han, J. and Pei, J. and Yin, Y., "Mining frequent patterns without candidate generation," *In proceedings of ACM SIGMOD International Conference on Management of Data*, vol. 29, pp. 1–12, 2000.

[30] Hansen, L. K. and Salamon, P., "Neural network ensembles," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[31] Hastie, T., Rosset, S., and Zhu, J. and Zou, H., "Multi-class adaboost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[32] Ho, T. K., "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, IEEE, vol. 1, 1995, pp. 278–282.

[33] Ho, T. K., Hull, J., and Srihari, S. N., "Decision combination in multiple classifier systems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 16, no. 1, pp. 66–75, 1994.

[34] Ho, T.K., "The random subspace method for constructing decision forests," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[35] Holmes, G., Donkin, A., and Witten, I.H., "Weka: A machine learning workbench," *Proceedings of ANZIIS*, 1994.

[36] *Kaikewreis - covid-19 solution*, `https://www.kaggle.com/kaikewreis/a-second-end-to-end-solution-for-covid-19`.

[37] Kuncheva, L. I, *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2014.

[38] Lam, L. and Suen, C. Y., "Optimal combinations of pattern classifiers," *Pattern Recognition Letters*, vol. 16, no. 9, pp. 945–954, 1995.

[39] Li, J. and Zaiane, O.R., "Exploiting statistically significant dependent rules for associative classification," *In Intelligent Data Analysis*, vol. 21, no. 5, pp. 1155–1172, 2017.

[40] Li, J. and Zaiane, O.R., "Associative classification with statistically significant positive and negative rules," in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 633–642.

[41] Li, W. and Han, J. and Pei, J., "CMAR: Accurate and efficient classification based on multiple class-association rules," in *IEEE International Conference on Data Mining, ICDM*, 2001, pp. 369–376.

[42] Littlestone, N. and Warmuth, M. K. and others, *The weighted majority algorithm*. University of California, Santa Cruz, Computer Research Laboratory, 1989.

[43] Liu, B. and Hsu, W. and Y., "Integrating classification and association rule mining," in *International Conference on Knowledge Discovery and Data Mining*, 1998.

[44] Quinlan, J. Ross, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[45] Quinlan, J.R., "C4.5: Programs for machine learning," *Machine Learning*, vol. 16, no. 3, pp. 235–240, 1994.

[46] Roberto J and Bayardo Jr., "Brute-force mining of high-confidence classification rules," in *Proceedings of 3rd International Conference on Knowledge Discovery and Data Mining (KDD)*, 1997, pp. 123–126.

[47] N. Sood, L. Bindra, and O. Zaiane, "Bi-level associative classifier using automatic learning on rules," in *Intl. Conf. on Database and Expert Systems Applications*, 2020.

[48] N. Sood and O. Zaiane, "Building a competitive associative classifier," in *Intl. Conf. on Big Data Analytics and Knowledge Discovery*, 2020.

[49] Verhein, F and Chawla, S., "Using significant, positively associated and relatively class correlated rules for associative classification of imbalanced datasets," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, IEEE, 2007, pp. 679–684.

[50] Wolpert, D. H, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[51] Xu, L., Krzyzak, A., and Suen, C. Y., "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE transactions on systems, man, and cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[52] Ye, Y., Li, T., Jiang, Q., and Wang, Y., "Cimds: Adapting postprocessing techniques of associative classification for malware detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 3, pp. 298–307, 2010.

[53] Yin, X. and Han, J., "Cpar: Classification based on predictive association rules," in *SIAM International Conference on Data Mining*, 2003, pp. 331–335.

[54] Zaıane, O.R and Antonie, M.L., "On pruning and tuning rules for associative classifiers," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2005, pp. 966–973.

[55] Zhang, J., Cao, P., and Gross, D. and Zaıane, O., "On the application of multi-class classification in physical therapy recommendation," *Health Information Science and Systems*, vol. 1:15, 2013.

[56] Zhou, Z. H., "Ensemble learning.," *Encyclopedia of biometrics*, vol. 1, pp. 270–273, 2009.

[57] Zhou, Z. H. and Feng, J., "Deep forest: Towards an alternative to deep neural networks," in *IJCAI*, 2017.