A DEEP LEARNING BASED MULTILINGUAL HATE SPEECH DETECTION FOR RESOURCE SCARCE LANGUAGES

by

Nabil Beshay

A project report submitted in conformity with the requirements
for the degree of Master of Science, Information Technology

Department of Mathematical and Physical Sciences
Faculty of Graduate Studies
Concordia University of Edmonton

A DEEP LEARNING BASED MULTILINGUAL HATE SPEECH DETECTION FOR RESOURCE SCARCE LANGUAGES

NABIL BESHAY

Approved:

Supervisor: Dr. Baidya Saha	Date
Committee Member	Date
Dean of Graduate Studies: Dr. Alison Yacyshyn	Date

A DEEP LEARNING BASED MULTILINGUAL HATE SPEECH DETECTION FOR RESOURCE SCARCE LANGUAGES

Nabil Beshay
Master of Science, Information Technology

Department of Mathematical and Physical Sciences
Concordia University of Edmonton
2022

Abstract

Over the last decade, the increased use of social media has led to an increase in hateful activities in social networks. An international issue that weakens the cohesiveness of civil societies is hate speech on internet social networks. Due to the lack of restrictions set by these sites for its users to express their views as they like. Hate speech is one of the most dangerous of these activities, so users have to protect themselves from these activities from social media sites such as YouTube, Facebook, Twitter, etc.Large-scale social platforms are currently investing important resources into automatically detecting and classifying hateful content, without much success.this research introduces a method for using deep learning algorithms to predict hate speech from social media websites. We implement proposed algorithms to detect hate speech in five different language: Arabic, English, and Urdu. this study employs a variety of feature engineering techniques and a comparative study among different machine and deep learning algorithms to automatically detect hate speech messages on many datasets. after hate speech data is collected, as a part of the preprocessing steaming, token splitting, character removal, and inflection elimination are carried out before performing the hate speech recognition process through deep learning algorithms.in future, we would like to deploy proposed algorithms to other low resource languages and explore more language specific features in deep learning framework.

Keywords: heat speech, Arabic, English, French, Urdu, Spanish, Sentiment analysis, Social media, Facebook, Twitter data, Data Preprocessing, machine learning classifiers, Supervised learning, Semi-supervised learning, performance metrics

Disclaimer: *Due to the nature of this work, some examples have offensiveness, hate speech and profanity. This doesn't reflect author opinions by any mean. We aim this work can help in detecting and preventing spread of such harmful content.*

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

There has been important debate over freedom of speech, hate speech, and hate speech legislation.The laws of some countries describe hate speech as speech, gestures, conduct, writing, or displays that incite violence or prejudicial conduct against a group or individualities on the base of their class in the group, or that disparage or intimidate a group or individualities on the base of their class in the group. The law may identify a group grounded on certain characteristics. In some countries, detest speech isn't a legal term. Also, in some countries, including the United States, much of what falls under the classification of" hate speech"is constitutionally covered. In other countries, a victim of hate speech may seek requital under civil law, guilty law, or both. [1].

The Internet is changing the face of communication and culture. In the world, the Internet has drastically altered the way we get our news, talk to our friends and generally live our lives. Its decentralized nature makes it a perfect place for amateurs and professionals likewise to produce and participate ideas, information, images, videos, art, music and further. In malignancy of, or maybe because of, its popular nature, the Internet is also peopled with Web spots devoted to inciting abomination against particular ethical, religious, racial or sexually- acquainted groups similar as women, Jews, African-Americans, Muslims and members of the lesbian, gay,bi-sexual and transgender (LGBT) community.[2].

Online hate speech is a type of speech that takes place online with the purpose of attacking a person or a group based on their race, religion, ethnical origin, sexual orientation, disability, and/ or gender. Online hate speech isn't easy defined, but can be recognized by the demeaning or dehumanizing function it serves.[3]

Multilateral covenants similar as the International Covenant on Civil and Political Rights (ICCPR) have sought to define its silhouettes. Multi-stakeholders processes (e.g. the Rabat Plan of Action) have tried to bring major clarity and suggested mechanisms to identify virulent messages. Yet, hate speech is still a general term in everyday converse, mixing concrete impendence to individualities and/ or groups with cases in which people may be simply venting their wrathfulness against authority. Internet interposers associations and social networks that intervene online communication like as Facebook, Twitter, and Google have advanced their own definitions of hate speech that bind users to a set of rules and allow companies to limit certain forms of expression. National and indigenous bodies have sought to promote understandings of the term that are more embedded in original traditions.[3]

The Internet's speed and reach makes it difficult for governments to enforce national legislation in the virtual world. Social media is a private space for public expression, which makes it difficult for regulators. Some of the companies owning these spaces have become more responsive towards tackling the problem of online hate speech.[3]

Politicians, activists, and academics discuss the character of online hate speech and its relation to offline speech and action, but the debates tend to be removed from systematic empirical evidence. The character of perceived hate speech and its possible consequences has led to placing much emphasis on the solutions to the problem and on how they should be grounded in international human rights law. Yet this very focus has also limited deeper attempts to understand the causes underlying the phenomenon and the dynamics through which certain types of content emerge, diffuse and lead—or not—to actual discrimination, hostility, or violence.[3]

Internet culture often categorizes hate speech as "trolling," but the severity and viciousness of these comments has evolved into something much more sinister in recent years, said Whitney Phillips, an assistant professor of communications at Syracuse University. Frequently, the targets of these comments are people of color, women and religious minorities, who have spoken out about online harassment and hateful attacks for as long as the social media platforms have existed, calling for tech companies to take action to curb them.[4] Unlike hate movements of the past, extremist groups are able to quickly normalize their messages by delivering a never-ending stream of hateful propaganda to the masses.

In Canada 2020, The number of police-reported hate crimes increased 37% during the first year of the pandemic. This increase was mostly due to more incidents targeting race or ethnicity, which nearly doubled,62% of all police-reported hate crimes were motivated by race or ethnicity and 20% of hate crimes were motivated by religion according to Statistic Canada. Of the 2,669 hate crimes reported to police in Canada in 2020, 1,594 (59.7 percent) of them were motivated by race or ethnicity. Nearly 20 percent were motivated by religion and almost 10 percent by sexual orientation. Other motivations including language, disability, sex, and age. [5].

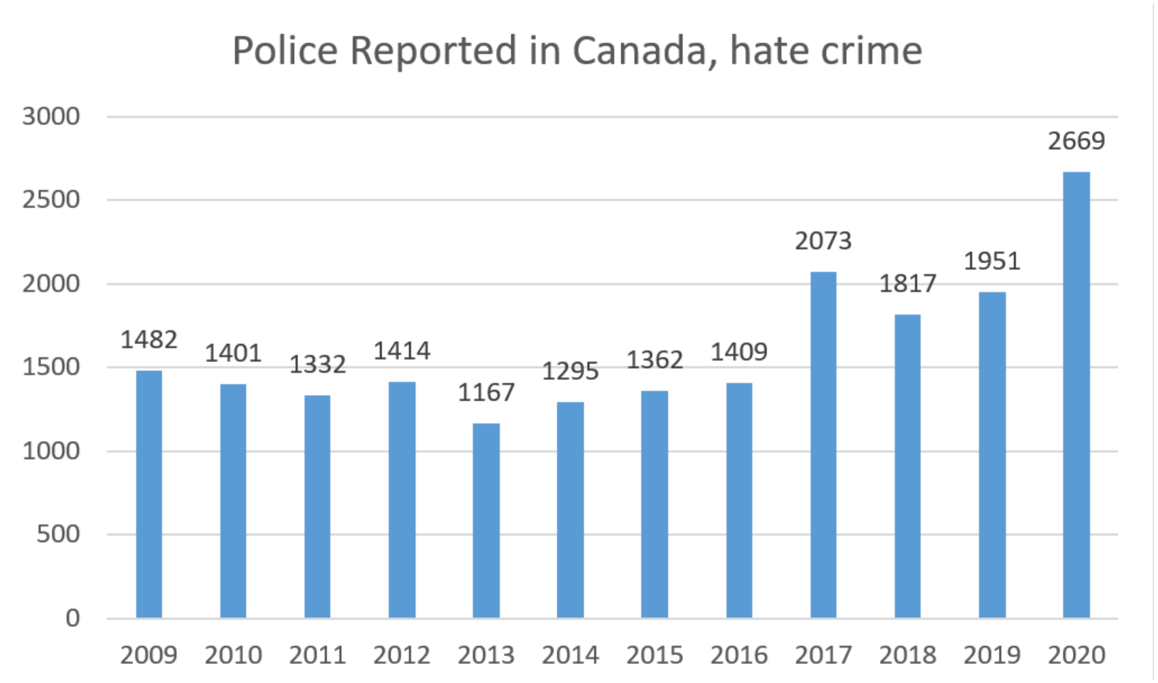


Figure 1.1: police-reported in Canada, hate crime 2020

It is not easy to comprehend hate speech. However, each culture has different characteristics that can be distinguished and recognized. These characteristics are debatable. Gelashvili and Nowak [6] say that it is difficult to regulate hate speech since many questions will be raised, such as: which kind of hate need to be dealt with? For studying hate speech, some common terminologies have been agreed on by a number of researchers, for example, some researchers [7] have surveyed general rules for hate speech recognition. In brief, it can be recognized when stereotyping group of people together or individuals by using racial and sexist slurs with intent to harm. In addition, indecently speaking about religion or specific country. Each social media company has its own policy regarding what content is or is not permitted online and Twitter's approach is among the most permissive.

Besides hate speech, there are many other related concepts, like hate , cyberbullying , abusive language, discrimination , toxicity , flaming . All of these concepts are slightly distinct from but still related to hate speech. By exploring those concepts can give insight into how to automatically detect hate speech.

Based on Silva et al [8], the hate speech can be grouped into ten categories: Race, Behavior, Physical, Sexual Orientation, Class, Gender, Ethnicity, Disability, Religion, and Others. Table 1.1 shows the categories and their corresponding examples of possible targets.

Table 1.1: Types of hate speech and examples (Table from Silva et al [8])

Categories	Example of possible targets
Race	black people, white people
Behavior	insecure people, sensitive people
Physical	obese people, beautiful people
Sexual	orientation gay people, straight people
Class	ghetto people, rich people
Gender	pregnant people, cunt, sexist people
Ethnicity	chinese people, indian people, paki
Disability	retard, bipolar people
Religion	religious people, jewish people
Other	drunk people, shallow people

The examples are to illustrate the severity of the hate speech problem. They are taken from the Twitter dataset [9] that was used in our experiments and in no way reflect the opinion of the authors.

1.2 Problem Statement

The debate around the regulation of hate speech is still ongoing . It is still not clear whether the best response to it is through legal measures, or other methods (such as counter-speech and education). Regardless of the means of countering it, the evident harm of hate speech makes its detection crucial. Both the volume of content generated online, particularly in social media, and the psychological burden of manual moderation [10] supports the need for the automatic detection of offensive and hateful content.
There are many layers to the difficulty of automatically detecting hateful and/or offensive speech, particularly in social media. Some of these difficulties being closely related to the shortcomings of keyword-based approaches. For one, words can be obfuscated in many different ways, both in an intentional attempt to avoid automatic content moderation [11], or as a consequence of the use of social media for communication . Furthermore, there are many expressions that are not inherently offensive, however they can be so in the right context. But even in the case of slurs, not only different

slurs hold a different degree of offense , the offense can also vary based on different time (as previously innocuous words may become slurs in time), as well as different use of the same word, different users, and different audience members [12].
when we speak about low resources Languages like (Arabic, Urdu,..etc) the detection problem will be more complicated , Because many of Machine learning Algorithm and Deep Learning Algorithm can't work with these Languages , low resources Languages have challenges with Machine Learning because : complexity, richness and Ambiguous structure.

1.3 Contribution of the thesis

- In general, hate speech detection is a text classification task. Following the typical procedure for the text classification, we first need to extract the features from text data and then apply the classification models to detect the hate speech.
- The research work involves the collection, cleaning, and analysis of data for the extraction of useful insights/information. The contribution to this project work involved the collection of data from Social Media especially Twitter. The data collected is different Datasets that present hate and offensive speech in five language(English,French,Spanish,Arabic and Urdu)
- The collected data is preprocessed which involved the removal of special characters, missing values, stopwords and clean data.
- With machine learning algorithm, feature selection was performed on the data to utilized only features (columns) that give more accurate analysis to the data. Not all features of the data are relevant for data analysis. Adding irrelevant features to the analysis may result in less accurate analysis.
- Machine Learning classifers were performed on a portion of the data, the train data, to create a model that can be used to test the remaining data. About seventeen (17) classifiers were performed to create the models and the accuracies of each model was tested to determine which model has the best performance. The classifiers utilized includes: KNeighborsClassifier,Support Vector Machine, Gaussian Process Classifier, Decision Tree Classifier, Random Forest Classifier, MLP Classifier, Ada Boost Classifier, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Gradient Boosting Classifier, Logistic Regression, Multinomial NB, SDG Classifier, LGBM Classifier and XGB Classisifer.

- For each classifier, the accuracy and precision were calculated for both supervised learning and semi-supervised learning. For supervised learning, the target feature is known which helps to train the remaining data. For semi-supervised learning, some of the data have the target variable while some don't. The data with the target variable serves as the train data with which the model is created. The model is tested on the test data, which automatically generated the target for the test (unlabelled) data.

- Deep learning methods have achieved notable performance in many classification tasks [13]. Unlike traditional machine learning methods, deep learning methods can automatically learn latent representations of the input data to perform classification [14]. Deep learning approaches have been widely applied to various natural language processing tasks, including text classification [15] [16] . Many recent studies adopt deep learning methods to detect hate speech in social media .

1.4 Organization of the thesis

For this research work, there are seven (7) chapters.

- In Chapter 1 is the Introduction which provides a detailed background of the research work, explaining the reasons for research work, the problem statement, how the hate speech consider big problem for people and communites.

- In Chapter 2 ,carries the Literature Review and is sectioned into three. The first section discusses related works of sentimental analysis of Hate speech data. Section two discusses the related work of Previous work in this area has focused on different aspects of hate speech . Section Three discusses monolingual, multilingual, and cross-lingual hate speech and offensive language detection models along with the few-shot learning problem in this domain.

- In Chapter 3, which is the methodology discusses in detail the three main methods for the research work which include feature selection, supervised learning, and semi-supervised learning.

- In Chapter 4 , is the Data collection and preprocessing which discussed in detail how data was collected, preprocessed to ensure it is cleaned enough for machine learning analysis.

- In Chapter 5, which is the Feature Selection and Visualization,identify meaningful coordinate projections for low dimensional data visualization.

- In Chapter 6, talked about the results and discussions of the analysis. Every step of the analysis is broken down here to understand different insights that were extracted from the data.
- In Chapter 7 , concludes this research work and also discussed the future works for further analysis.

Chapter 2

Literature Review

This section presents a review of the recently proposed methods for offensive and hate speech detection from user-generated content on social media for English ,Urdu and Arabic languages.

There are relatively several researches that focuses on social media analysis. People's opinions and perspectives on social media are relatively important due to the volume of information that can be extracted and anatomized to help give applicable insights. Starting with the World Wide Web, Warner and Hirschberg [17],are the first to research how to identify hate speech in the world wide web. Their work is targeted to specific type of hate which is anti-Semitic. For Twitter, Watanabe etal. [18] proposed a supervised approach for hate-speech detection Their approach proved that supervised classifier performs better in the binary classification when compared with ternary classification. Another binary classifier is developed by Burnap and Williams [19] that detects hateful and non-hateful tweets from labelled dataset.

The multilingual feature of hate speech has only recently been studied. For research, data sets have been made available for languages including Arabic and French [20], Indonesian [21], Italian [22], Polish [23], Portuguese [7], and Spanish [24]. As far as we are aware, very few studies have attempted to use these datasets to create multilingual classifiers. In order to annotate the hate speech on Twitter, Huang et al. [25] employed a corpus of tweets in five different languages. They investigate the demographic bias in the classification of hate speech using this new dataset. Corazza et al. [26] ,Three datasets from three languages (English, Italian, and German) were utilised by Corazza et al. [26] to explore the multilingual hate speech. To create models for detecting hate speech, the authors employed SVM and Bi-LSTM. Because we conduct the experiment on a considerably larger variety of languages [26] utilising more datasets, our study differs from these earlier studies [27]. Our research aims to make use of the already available resources for detecting hate speech in order to create models that may be applied universally to other languages.

2.1 Hate Speech Detection in English

While most approaches to hate speech detection are proposed for English, other systems are developed to handle the task in Arabic, Urdu,French and Spanish, due to recent shared tasks. Concerning Spanish, the IberEval 2018 edition 5 has proposed the Aggressiveness Detection task [28] applied to Spanish, aiming at providing a classification of aggressive/non- aggressive tweets. a range of systems is proposed, exploiting content-based (bag of words, word n-grams, term vectors, dictionary words, slang words) and stylistic-based features (frequencies, punctuation, POS, Twitter-specific elements). Most of the systems depend upon neural networks (CNN,LSTM, and others). The top-ranked team was INGEOTEC [29]: The system relies on MicroT, atext classification approach supported by a lexicon-based model that takes into consideration the presence of aggressive and effective words, and a model based on the Fast-text representation of texts.

More recently, a task for the detection of hate speech against immigrants and women on Twitter has been organised at Semeval 2019 [30], providing an English and Spanish dataset annotated consistent with the identical guidelines. While for both languages variety of neural network approaches has been proposed, the most effective systems for hateful content detection still depend on SVM and embedding-based features.

Waseem et al [31] .A logistic regression classifier was employed by Waseem et al. to identify hate speech (HS) tweets. They pinpointed the key characteristics that offer the greatest identification performance. Additionally, using non-linguistic variables like gender or geography can increase performance but they are always inaccessible or unreliable on social media. They examined the suggested method using 16 K tweets and received an F1-score of 73.93 %. Using four models from [31]—character 4-grams, word2vec, randomly generated word vectors, and character n-grams merged with the word2vec model—a convolutional neural network (CNN) model was suggested in [32] to detect HS. The analysis demonstrates that the Word2vec model produced the best performance. Pitsilis et al. [33] achieved 0.87 and 0.88 in recall and precision, respectively, using Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) models with word frequency vectorization for HS classification and the prior dataset gathered by [31]. When Watanabe et al. [34] employed text patterns and unigrams as features to train a J48graft machine learning algorithm, they were able to identify hateful or not-hateful tweets with an accuracy of 87.4 % .

2.2 Hate Speech Detection in Arabic

Albadi ,About 6.6 K Arabic HS tweets made up the initial HS dataset that Albadi et al [35]. collected. For the classification task, [36]the support vector machine (SVM) classifier and a GRU (Gated Recurrent Unit) trained on AraVec embeddings delivered the best performance with a 79 percent accuracy rate.[20] A multilingual HS dataset made up of tweets in English, French, and Arabic was created by Ousidhoum et al. 13 K tweets were categorized into many categories using Amazon Mechanical Turk, including target qualities, target groups, directness, and hostile types. In the majority of the multi-label classification tasks, BiLSTM and Sluice networks outperformed conventional bag-of-words models [37].

Abu Farha et al. [38] developed a multitask learning architecture based on CNN-BiLSTM, which is trained to detect HS and offensive language. The model incorporates more data through adding sentiment information using the Mazajak Sentiment Analyser [39]. The proposed model achieved a 90.4% F1-score in OFF and 73.7% in the HS task.

Mulki et al,A dataset of 6 K tweets for the Tunisian dialect from Twitter that contained hatred and offensive speech was produced by Mulki et al. [40]. Using Term Frequency (TF) weighting, the authors retrieved various n-gram features from each tweet. With the use of the retrieved features, SVM and Naive Bayes (NB) classifiers were created, yielding an F1-score of 83.6 percent. This study is dialect-specific and has poor performance on short datasets. In the shared goal of OFF Detection in the 4th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT4 (https://edinburghnlp.inf.ed.ac.uk/workshops/OSACT4/) (accessed on 10 May 2021)) [41], Djandji et al. [42] suggested a model based on AraBERT [41] with MTL. Utilizing data from several jobs at once, their proposed model addressed the issue of data imbalance and achieved the best performance with a 90% macro-F1 score.

Hassan et al. [43] implemented various classical ML and DL approaches, such as SVM, CNN BiLSTM, and Multilingual BERT, for the HS subtask. The stacked SVMs achieved an 80.6% macro F1-score.

Ghosh-Chowdhury et al. [44], have worked on the detection of religious hate speech in the Arabic language. They have put out a method made up of Social Network Graphs and Arabic word embeddings [45]. Additionally, they have placed a strong emphasis on neighbourhood characteristics for the detection of hate speech. They were able to find 3950 tweets in Arabic, 1685 with the hashtags "HS" and 2265 with "NHS." They have employed a variety of features for preprocessing to get rid of things like HTML links and hashtags. Last but not least, to train the classification model, they combined LSTM and CNN with 600d word embeddings.

2.3 Hate Speech Detection in Urdu

Hammad Rizwan et al [46].Hate-Speech and Offensive Language Detection in Roman Urdu,this research used a dataset in Roman Urdu for the task of hate speech detection in social media content, annotated with five fine-grained labels. implemented various classical that complex ensemble models yield a higher F1-score. For instance, SVM+RF+AB shows an F1-score of 0.90, which is the highest amongst all the baseline approaches.

MUHAMMAD et al [47] Automatic Detection of Offensive Language for Urdu and Roman Urdu, in this research performed automatic detection of offensive language from YouTube comments of Roman Urdu and Urdu. the major contribution is to provide the first dataset of the Urdu language to detect offensive language automatically from the text. implemented various classical of machine learning, regression-based technique outperforms the other six techniques but these models take longer time to build the model. LogitBoost shows superior performance on Roman Urdu using character tri-gram and achieved 99.2 % score of F-measure. SimpleLogistic outperforms the others classifiers using character tri-gram on Urdu dataset and achieved 95.8 % F-measure value. k-NN takes less time to build the model.

As has been mentioned before, when it comes to the identification of hate speech, a lot of the study is concentrated on the English language. All feature extraction methods, pattern recognition systems, and models are consequently built for that language. Furthermore, there is virtually no organised data available for study in low resource languages. For English, there are some quite large annotated publically available datasets, but the same cannot be stated for low resource languages. Additionally, to the best of our knowledge, there isn't any extensive published research on the identification of hate speech in Arabic and Urdu at this time. Because of this, it is currently challenging to determine which mechanism would work best for a given data set or how the data set should be modified to extract the best predictions feasible. We would definitely be interested in looking into the effectiveness of transfer learning when combined with different transformer models for our dataset.

2.4 hate speech Tweets Sentiments

Hate speech online content has become a major issue in today's world due to an exponential increase in the use of the internet by people of different cultures and educational backgrounds. Differentiating hate speech and offensive language is a key challenge in the automatic detection of toxic text content. In this report, we propose an approach to classify tweets on Twitter into two classes: hate speech and non-hate speech with Urdu dataset. in Arabic and English dataset classify tweets on Twitter into six classes: (Normal ,offensive , disrespectful , fearful,hateful and abusive) . Using the Twitter dataset, we perform experiments by leveraging bag of words and the term frequency-inverse document frequency (TFIDF) values to multiple machine learning models. We perform comparative analysis of the models considering both of these approaches.

Chapter 3

Hate Speech Prediction

Analyzing data to extract using insights requires some methods and this research work is not an exception. The methodologies utilized can have a huge impact on the outcome or performance of the project work. Figure 3.1 illustrates the suggested architecture for machine learning and Deep learning based hate speech prediction that uses these techniques.

3.1 Methodologies for Hate Speech Detection

Hate speech detection is, in general, a text classification task. In accordance with standard practise, we must first extract the features from text data before applying classification algorithms to detect hate speech.

3.2 Text Features

The data are typically in text data forms, which machine learning models cannot condense. Only numeric values can be compacted into machine learning models. It is possible to translate these text data into numerical features using procedures without distorting the meaning of the data. The raw text data cannot be supplied directly to machine learning algorithms when using those techniques. The majority of algorithms can process raw text data with variable length but only take numerical feature vectors with a fixed size. We must extract numerical aspects from text content in order to solve this issue. In conventional machine learning, we typically use Bag of Words or Term Frequency-Inverse Document Frequency to transform a set of text documents into a matrix of token counts.

Some of the feature extraction techniques are mentioned below:

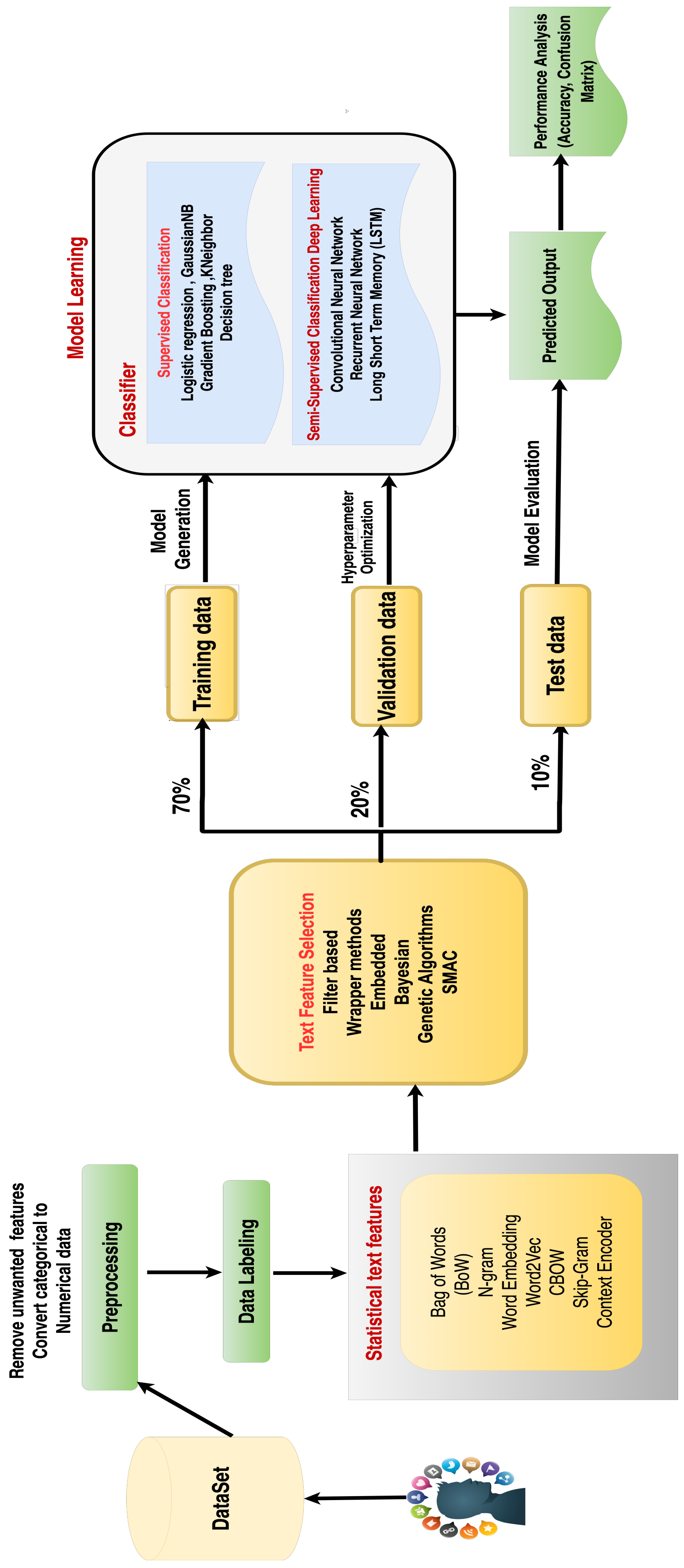


Figure 3.1: Proposed Architecture for Mental Health Prediction With Machine Learning

3.2.1 N-gram

A sentence's N-gram is made up of a series of words. The n-gram is likely the most basic machine learning notion. There are many different ways that N-grams might be useful. It can be utilised for automatic word correction, automatic spell checking, and grammar checks [48]. Checking the relationships between words is also helpful, particularly when attempting to predict what someone will say in order to infer their feelings or sentiments from the words they use. N-grams are word combinations that are used in combination. With N = 1, unigrams are N-grams. These are known as bigrams for N = 2 and trigrams for N = 3. The language's structure is captured by n-grams, which make it possible to predict which word would likely come after a given word.

3.2.2 The Word2Vec Model

Word2vec is a technique for quickly producing word embeddings. It is a predictive deep learning-based model developed by Google in 2013 that computes and produces high quality, distributed, and continuous dense vector representations of words that reflect similarity in both context and meaning [49]. It is a kind of unsupervised model that extracts a vocabulary of words from a large corpus of words and produces dense word embeddings for each word in the vocabulary. To enable machine learning algorithms to conduct algebra operations on numbers rather than words, the words are converted into vectors. The term "word embedding" refers to this change [50]. With dispersed Hypothesis in Word2Vec, a word's lexicon can be located in words that are close by. By examining these nearby words, one can guess a word.

3.2.2.1 Continuous Bag of Words

Bag of Words (BOW). A technique called BOW is used to extract numerical information from textual content. The first step is tokenizing every sentence and storing all of the unique tokens that resulted from tokenizing every sentence in a dictionary (a big bag) [49], [51]. The second step is counting the instances of tokens in every sentence. Here, we turn the raw phrases we have into feature vectors. As a result, a corpus of sentences can be represented as a matrix with one row for each phrase and one column for each token that appears in the corpus. We will have a feature column for each token; this process is known as text vectorization .

According to the CBOW model, the architecture tries to anticipate the current target word, which is typically the middle word, using the source context words, or the words immediately around it. The corpus is designed such that it is possible to extract every distinct word from the dictionary and map it to a special number identification. The Kera preprocessing package is the main Python module used. Context and target are the next two variables used to construct the CBOW generator. The CBOW model's deep learning architecture is created using Keras and Tensorflow. With fewer datasets, this model typically performs better. The model may be trained quickly and with greater precision.

3.2.2.2 Continuous Skip-Gram Model

This is the inverse of CBOW as it predicts the surrounding words from the current target words. With a larger dataset, this model performs better. Predicting the contexts of a given word is the goal. To implement this model, the corpus dictionary is built such that each unique word can be extracted from the dictionary and assigned a unique identifier. Additionally, mappings that convert words into and out of their distinctive identifiers are kept up to date. The skip-gram generator, which will give the pair of words and their significance, is then developed. To build the skip-gram model, Keras on top of TensorFlow is taken advantage of to build it. The embedded words are retrieved after the model has been trained. [49].

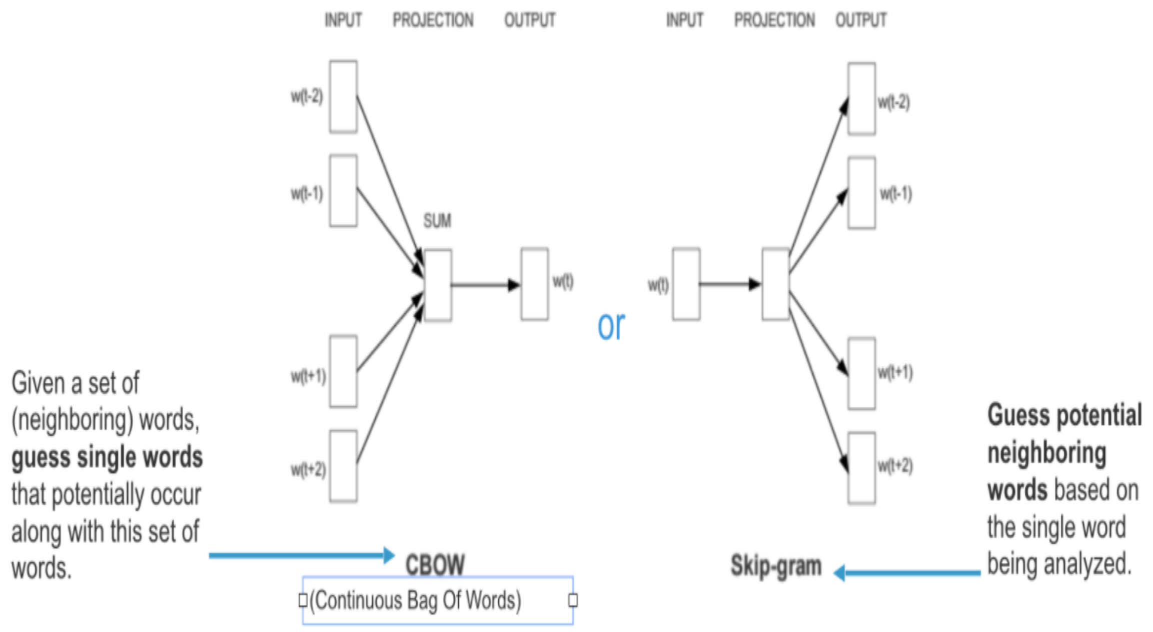

INPUT
PROJECTION
OUTPUT
w(t-2)
w(t-1)
SUM
w(t)
w(t+1)
w(t+2)
or
INPUT
PROJECTION
OUTPUT
w(t)
w(t-2)
w(t-1)
w(t+1)
w(t+2)
Given a set of (neighboring) words, guess single words that potentially occur along with this set of words.
CBOW
(Continuous Bag Of Words)
Skip-gram
Guess potential neighboring words based on the single word being analyzed.

Figure 3.2: CBOW and Continuous Skip-Gram Model Architectures [50]

3.2.3 Bag of Words

A sort of feature extraction or feature encoding called "bag of words" is used to extract features from text. Because the sequence in which the text appears is unimportant and ignored, it is known as a "bag of words." It doesn't care where in the text the known word appears; it merely cares that it does. Every piece of free text in a document is transformed into a vector that may be fed into or produced from a machine learning model. The BOW module in the Python programming language is called CountVectorizer. It transforms a given text into a vector-based on the frequency (count) of each word occurs in the entire text. Information is available **here**. The text input is first pre-processed by CountVectorizer before it generates a vector representation of the words. Each unique word is represented by a column in the matrix that is created by CountVectorizer, and each sample of text from the document is represented by a row in the matrix [52].

3.2.4 hashing vectorizer

hashing vectorizer is a vectorizer which uses the hashing trick to find the token string name to feature integer index mapping. This vectorizer converts text documents into matrices by creating sparse matrices out of the collection of documents that contain the token occurrence counts. The following benefits of hashing vectorizer: It is highly low memory scalable for huge data sets because the vocabulary dictionary does not need to be kept in memory. It can be utilised in a parallel or streaming pipeline because there isn't any state during the fit. The hashing trick is a machine learning technique used to encode categorical features into a numerical vector representation of pre-defined fixed length. It works by using the categorical hash values as vector indices, and updating the vector values at those indices. [53].

3.3 Supervised Learning

This is a type of Machine learning where a dependent variable can be predicted based on one or more independent variables. As an illustration, let's say we want to predict if a bank customer would repay a loan based on the loan amount, length, and demography. In this case, the status variable is the dependent variable and the loan amount, duration, and demography are the independent factors. The independent variables, usually represented with X are also known as input variables while the dependent variable, usually represented by Y is also known as output variables. A dataset of several animals, including dogs, cats, horses, and lions, is provided as an example. The machine learning model is given a portion of the correctly labelled data to interpret. The model is then given the remaining data (test data) to sort. The model can reliably identify which animal is which using the test data because it is already familiar with the traits of the many animals. Supervised machine learning can forecast the dependent variable from the independent variables.Supervised learning typically involves a classification problem, where the dependent variable and target variable are both categorical data types used to identify a data's category. Logistic regression, Random Forest, Decision Tree, KNN, Linear Support Vector Machines, Non-Linear Support Vector Machines, Naive Bayes Theorem, and many others are examples of the algorithms used in supervised machine learning to develop supervised learning models.

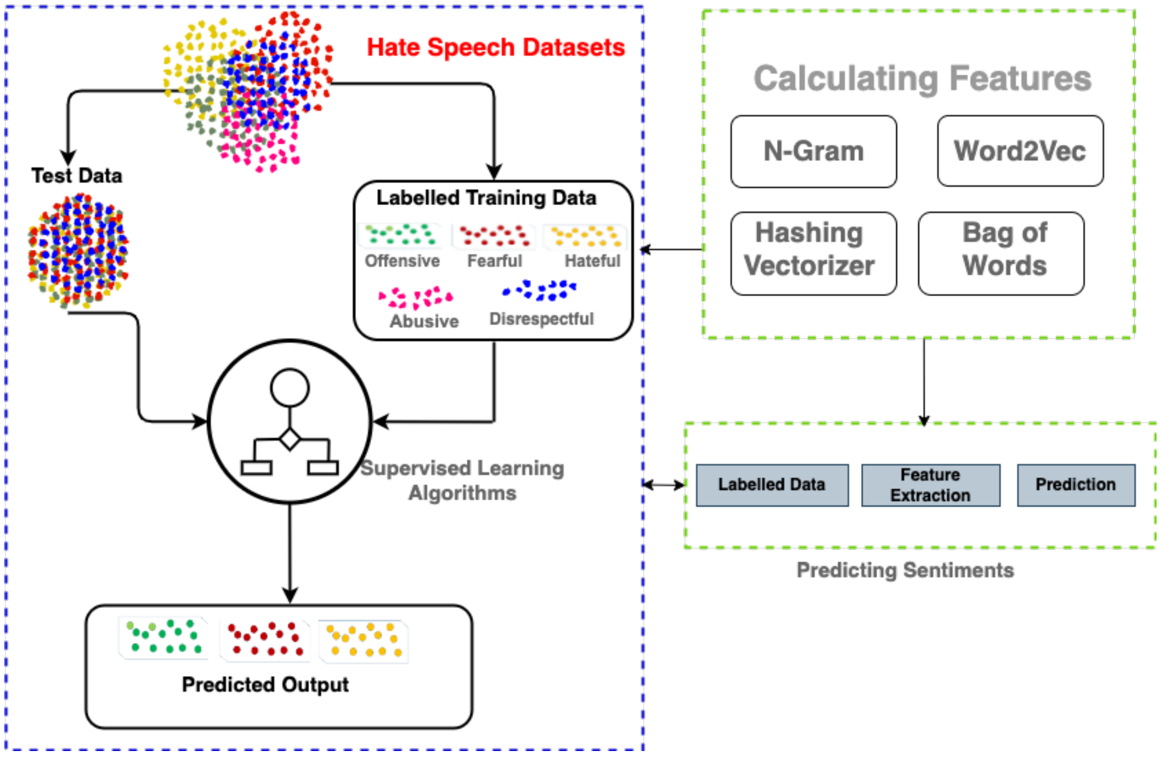


Figure 3.3: Supervised Learning

3.4 Semi-supervised Learning

Semi-Supervised In machine learning, the majority of the data (input data) are unlabeled and the amount of labelled data is very minimal (output data). The model is trained using these data. Between supervised and unsupervised machine learning, this form of learning occurs. This is where most real-world data is classified. It uses pseudo labelling to train its model, which combines a variety of neural network models and training techniques [54]. Similar to supervised learning, the model is trained using a limited subset of labelled data until promising results are attained. The outputs are predicted using the unlabelled training data, which are faux labels. This might not be true. The pseudo labels are connected to the labels of the labelled training data. Both the labelled training data and the unlabeled data that they enter are related. To reduce mistakes and increase model accuracy, the model is re-trained once more. The illustration of semi-supervised learning is shown in Figure 3.4.

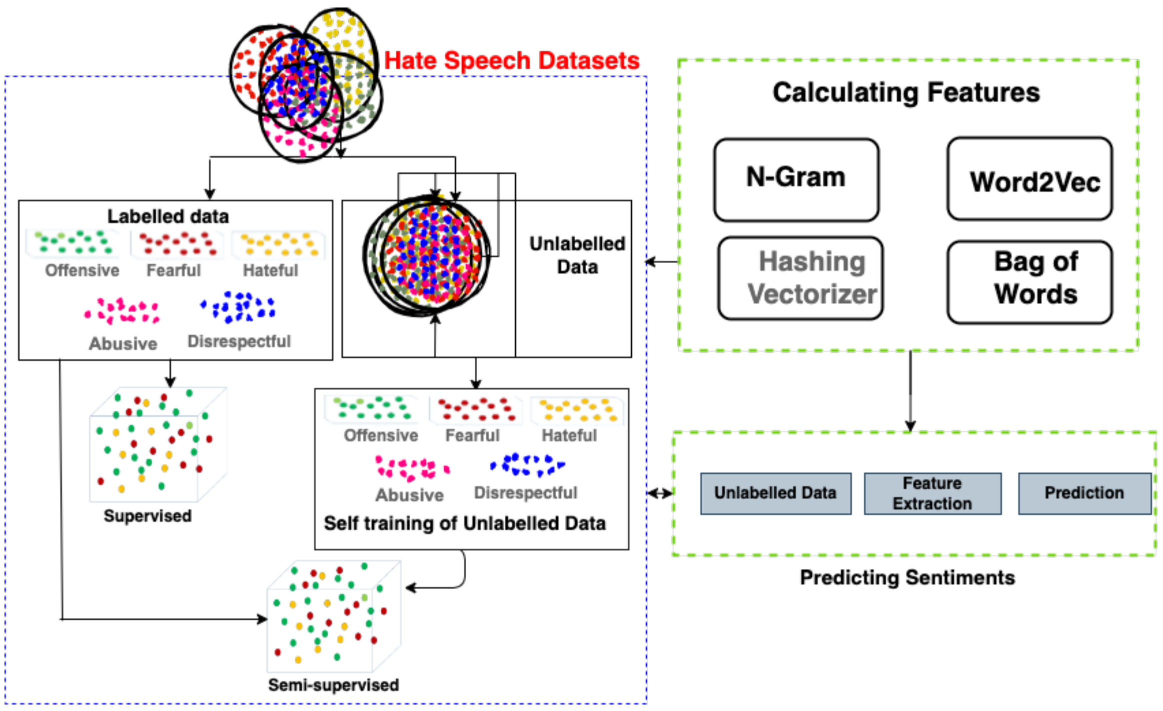


Figure 3.4: Semi-Supervised Learning

Chapter 4

Data Collection and Preprocessing

4.1 Data Collection

In general, CS researchers have two alternatives for obtaining the information they need to propose or evaluate their solutions: (a) using already-existing datasets; or (b) developing and labelling their own dataset. It is important to note that there isn't a commonly accepted benchmark dataset for the task of detecting hate speech. In light of the favorable results from other data-science jobs, the availability of such benchmark datasets can help advance the field and make performance comparisons between particular solutions simpler and more accurate.
We chose against combining datasets with hate and offence samples because to the different definitions of hate speech and offensive language content in publicly available datasets. As a result, we take into account two distinct tasks, hate speech recognition and offensive language detection, with various datasets in various languages. The offensive datasets include any form of unacceptable language or a targeted offence, including insults, threats, and posts containing profane language or swear words [55]. The hateful datasets consist of insults targeted toward a group based on some protected characteristics, such as sexual orientation, religion, misogyny, nationality, gender, ethnicity, etc.

We make use of three publically accessible sources supplied by the scholarly community in the languages of English, Arabic, and Urdu. The majority of the statistics are chosen in accordance with Twitter data on hate speech and vulgarity.

The statistics of these datasets are shown in Table 4.1, where the datasets from various languages in the Hate Speech and Offensive Language categories are represented in the second column. The final column lists the total number of samples in each language.

Table 4.1: Statistics of the three datasets

Dataset	Total	Classes Distribution
English	24784	Offensive 54.33 % Hateful 11.44 % Abusive 5.67 % Fearful 6.59 % Disrespectful 10.27 % Normal 11.71 %
Arabic	8845	Offensive 54.33 % Hateful 15.72 % Abusive 16.49 % Fearful 1.22 % Disrespectful 9.04 % Normal 27.29 %
Urdu	2400	Hateful 49.46 % Normal 5.45 %

4.1.1 English dataset

Use publicly available Twitter Dataset for Hate Speech and abusive language with 24784 rows provided by CrowdFlower[56].Dataset using Twitter data, is was used to research hate-speech detection. The text is categorised as either offensive language, hate speech,Abusive,Fearful,Disrespectful or normal speech. It's critical to be aware that this dataset includes content that can be construed as racist, sexist, homophobic, or otherwise inappropriate given the nature of the study.

4.1.2 Arabic dataset

Use publicly available Twitter Dataset for Hate Speech and Abusive Language with 8845 rows from Harvard University and GitHub [57].Dataset using Twitter data, is was used to research hate-speech detection. The text is categorised as either offensive language, hate speech,Abusive,Fearful,Disrespectful or normal speech. It's critical to be aware that this dataset includes content that can be construed as racist, sexist, homophobic, or otherwise inappropriate given the nature of the study.

4.1.3 Urdu dataset

Use publicly available Twitter Dataset for Hate Speech and Abusive Language with 2400 rows provided by CICLing 2021 track @ FIRE 2021 co-hosted with ODS SoC

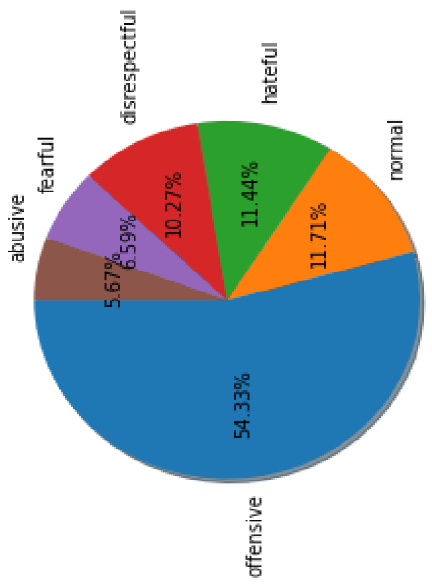


Figure 4.1: English dataset classification

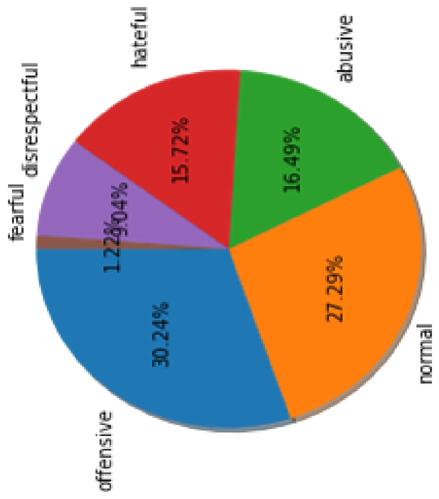


Figure 4.2: Arabic dataset classification

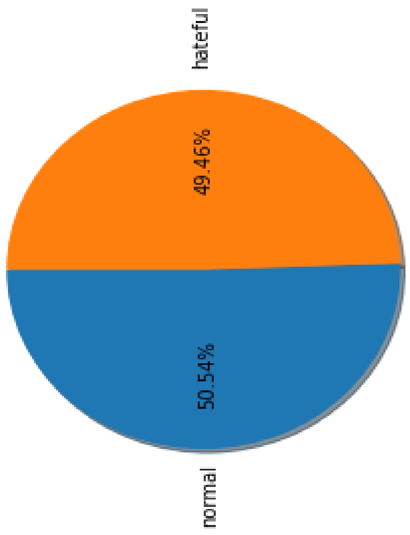


Figure 4.3: Urdu dataset classification

2021 [58] .Dataset using Twitter data, is was used to research hate-speech detection. The text is categorised as either hate speech, or normal speech. It's critical to be aware that this dataset includes content that can be construed as racist, sexist, homophobic, or otherwise inappropriate given the nature of the study.

4.2 Data Preprocessing

Data Preprocessing is the most important step of data mining which deals with the transformation and preparation of datasets for knowledge extraction [reference 2 data preprocessing]. The process of preprocessing involves a number of strategies. The dataset is being cleaned, integrated, transformed, and reduced by some of them. This produces organised, clean data that can be used for modelling. The majority of the time, the raw format of the data collected or retrieved from various sources makes it impractical to analyse it; as a result, the raw data must first be cleaned before analysis. Data cleaning accounts for roughly 70 % of all analysis project work. The text is the subject of analysis, thus that is where the focus lies. To make sure the data is clean enough for the model to accept, the uncleaned data was preprocessed. The preprocessed data is saved in a new column called text-str. Text from columns is preprocessed. The following list includes some of the preprocessing done on our raw data:

4.2.1 Remove the urls from the text

To remove the urls from a text, a function as shown below is written in python and applied to the text.

```
def remove_url(row):
txt = str(row['text-str']).split('https')[0]
return txt
data['text-str'] = data.apply(remove_url, axis = 1)
```

4.2.2 Remove the special characters

Some regular expressions were expressed to remove the special characters as stated in the code below

```
data['text-str'] = data.text-str.str.replace("?!,\ &:;%()", " ", regex=True)
```

4.2.3 Remove all usernames with @

The following line of code can be used to remove and replace usernames.

```
data['text-str'] = data['text'].str.replace('@[\w:]*','')
```

4.2.4 Removal of noise, URLs, hashtags user mentions

Noise, URLs, hashtags, and user mentions are removed. Unwanted strings and Unicode, which are regarded as crawling byproducts, add noise to the data. Additionally, practically all tweets that users publish include URLs that point to extra information, @username mentions, and the hashtag symbol (sometrendingtopic) to link their tweet to a certain subject. These hashtags can also be used to convey emotion. These hints provide supplementary information that is helpful for humans, but they offer no information to machines and can be viewed as noise that needs to be dealt with. Different approaches have been proposed by researchers to deal with this additional data provided by users, such as URLs; a study by Agarwal et al.[59] replaced them with tags whereas in another study by Khan et al. [60] removed user mentions (@username).

4.2.5 Word segmentation

is the process of dividing the words, phrases, or keywords used in a hashtag; for example, the hashtag sometrendingtopic is divided into the three words some, trend, and topic. This phase can assist in making it simple for machines to understand and categorise the content of tweets without any human involvement. As was already established, practically all tweets on Twitter contain hashtags that link them to a specific current topic.

4.2.6 Replacing emoticons and emojis

To convey emotion and opinion, Twitter users can use a variety of emoticons and emojis, including:-), ;-),:-(, etc. In order to effectively classify tweets, it is crucial to record this helpful information. These expressions and emoticons were changed in a study by Gimpel et al. [10] to their corresponding word meanings, for example,:-) is changed to cheerful, and:-(to sad.

4.2.7 Remove numbers from characters

Here, all numeric values are removed with the regular expression (regex) pattern \d+. The addition sign ensures multiple numbers such as 10 are interpreted as such and not interpreted as two separate numbers. The line of code below removed all numeric values from the text.
data['text-str'] = data['text-str'].str.replace('\d+',"").

4.2.8 Drop Null Values

Some unstructured/raw data, there may be missing values. These values are sometimes called null values. They are usually filled up with the most common words or are dropped completely.If null values are not dealt with, it will affect our models are machine models don't accept null values. In our preprocessing, we dropped the null values with the line of code stated below:

data['text-str'] = data['text-str'].dropna(inplace = True)

4.2.9 Removal of stopwords

Stopwords are words that do not add meaning to a sentence and therefore can be ignored or removed without tampering with the meaning of the sentence.[reference stopwords]. Stopwords are found in most languages but for the purpose of this project work, stopwords in English are utilized. For sentimental analysis, Stop words are to be removed from the data to keep only the root words. Common stopwords include [i,me,my,myself,we,our,ours,ourselves,you,your]. A list of common stopwords (in english) can be found **here**. With these data, stopwords are removed by importing stopwords from the corpus of nltk, the python module for natural language processing.

4.2.10 Lowercasing

Lowercasing all the texts, which avoids capitalized versions of words being treated as separate features to lowercase versions of the same word.

4.2.11 Tokenization

Tokenization is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as individual words or terms. Each

4.2.12 Removing Punctuation

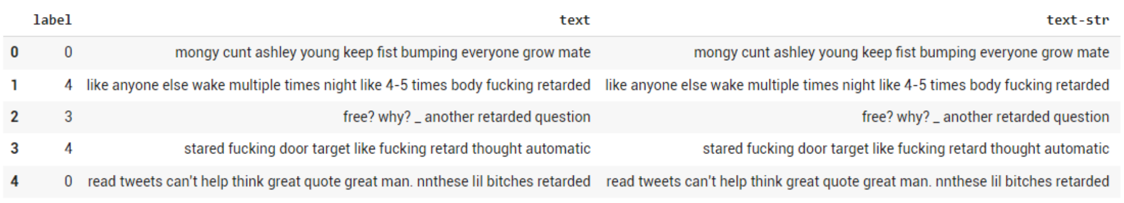
Punctuation is frequently used in social media to convey sentiment and emotion. While this is simple for people to understand, it is less helpful for brief texts to be automatically classified. Because of this, removing punctuation from text before preprocessing it for automated classification tasks like sentiment analysis is a popular technique. But occasionally, emotive punctuation marks like! and? are used. Punctuation was eliminated in the study by Lin et al. [61], whereas Balahur [62] explored an alternate strategy in which question marks or exclamation points were substituted with appropriate tags, such as!, which is frequently used to show astonishment.

```
import nltk
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
```

Using the lambda function, apply the stopwords to the text to remove them. Also, extract words that have 4 characters and above in order to filter words that are meaningful to the research work.

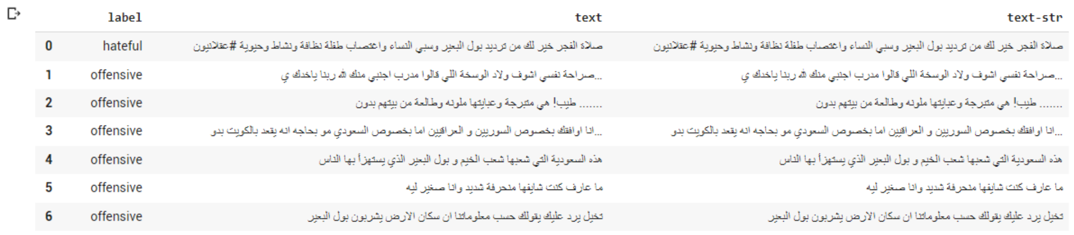
```
data['text-str'] = data['text-str'].apply(lambda x: ' '.join([w for w in x.split() if w not
in stop_words]))
data['text-str'] = data['text-str'].apply(lambda x: ' '.join([w for w in x.split() if
len(w)> 4]))
```

Below is the top 10 data showing the id, month, the uncleaned text and the tidy text
    The complete python code for data preprocessing can be accessed **here**.



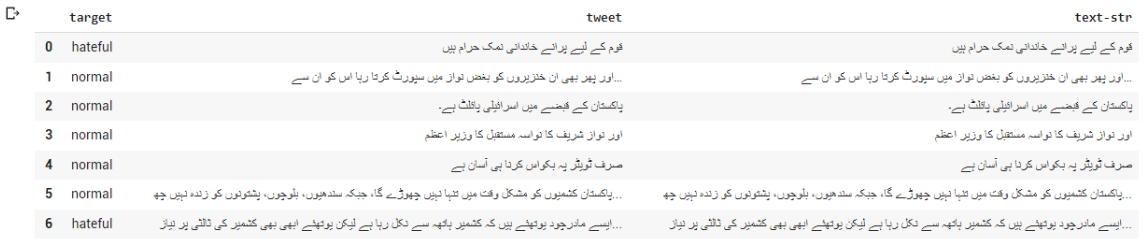
	label	text	text-str
0	0	mongy cunt ashley young keep fist bumping everyone grow mate	mongy cunt ashley young keep fist bumping everyone grow mate
1	4	like anyone else wake multiple times night like 4-5 times body fucking retarded	like anyone else wake multiple times night like 4-5 times body fucking retarded
2	3	free? why? _ another retarded question	free? why? _ another retarded question
3	4	stared fucking door target like fucking retard thought automatic	stared fucking door target like fucking retard thought automatic
4	0	read tweets can't help think great quote great man. nnthese lil bitches retarded	read tweets can't help think great quote great man. nnthese lil bitches retarded

Figure 4.4: The Unstructured and Pre-processed data for English



	label	text	text-str
0	hateful	صلاة الفجر خير لك من ترديد بول البعير وسبي النساء واغتصاب طفلة نظافة ونشاط وحيوية #عقلانيون	صلاة الفجر خير لك من ترديد بول البعير وسبي النساء واغتصاب طفلة نظافة ونشاط وحيوية #عقلانيون
1	offensive	صراحة نفسي اشوف ولاد الوسخة اللي قالوا مدرب اجنبي منك لله ربنا ياخدك ي...	صراحة نفسي اشوف ولاد الوسخة اللي قالوا مدرب اجنبي منك لله ربنا ياخدك ي...
2	offensive	طيب! هي متبرجة وعبايتها ملونه وطالعة من بيتهم بدون	طيب! هي متبرجة وعبايتها ملونه وطالعة من بيتهم بدون ......
3	offensive	انا اوافقك بخصوص السوريين و العراقيين اما بخصوص السعودي مو بحاجه انه يقعد بالكويت بدو...	انا اوافقك بخصوص السوريين و العراقيين اما بخصوص السعودي مو بحاجه انه يقعد بالكويت بدو...
4	offensive	هذه السعودية التي شعبها شعب الخيم و بول البعير الذي يستهزأ بها الناس	هذه السعودية التي شعبها شعب الخيم و بول البعير الذي يستهزأ بها الناس
5	offensive	ما عارف كنت شايفها متحرقة شديد وانا صغير ليه	ما عارف كنت شايفها متحرقة شديد وانا صغير ليه
6	offensive	تخيل يرد عليك يقولك حسب معلوماتنا ان سكان الارض يشربون بول البعير	تخيل يرد عليك يقولك حسب معلوماتنا ان سكان الارض يشربون بول البعير

Figure 4.5: The Unstructured and Pre-processed data for Arabic



	target	tweet	text-str
0	hateful	قوم کے لیے پرانے خاندانی نمک حرام ہیں	قوم کے لیے پرانے خاندانی نمک حرام ہیں
1	normal	اور پھر بھی ان خنزیروں کو بغض نواز میں سپورٹ کرتا رہا اس کو ان سے...	اور پھر بھی ان خنزیروں کو بغض نواز میں سپورٹ کرتا رہا اس کو ان سے...
2	normal	پاکستان کے قبضے میں اسرائیلی پائلٹ ہے.	پاکستان کے قبضے میں اسرائیلی پائلٹ ہے.
3	normal	اور نواز شریف کا نواسہ مستقبل کا وزیر اعظم	اور نواز شریف کا نواسہ مستقبل کا وزیر اعظم
4	normal	صرف ٹویٹر پہ بکواس کرنا ہی آسان ہے	صرف ٹویٹر پہ بکواس کرنا ہی آسان ہے
5	normal	پاکستان کشمیوں کو مشکل وقت میں تنہا نہیں چھوڑے گا، جبکہ سندھیوں، بلوچوں، پشتونوں کو زندہ نہیں چھ...	پاکستان کشمیوں کو مشکل وقت میں تنہا نہیں چھوڑے گا، جبکہ سندھیوں، بلوچوں، پشتونوں کو زندہ نہیں چھ...
6	hateful	ایسے مادرچود بوتھئے ہیں کہ کشمیر ہاتھ سے نکل رہا ہے لیکن بوتھئے ابھی بھی کشمیر کی ثالثی پر نیاز...	ایسے مادرچود بوتھئے ہیں کہ کشمیر ہاتھ سے نکل رہا ہے لیکن بوتھئے ابھی بھی کشمیر کی ثالثی پر نیاز...

Figure 4.6: The Unstructured and Pre-processed data for Urdu

Chapter 5

Feature Selection and Visualization

Understanding the most significant characteristics to employ is essential for creating a successful model. Experimentation is required to determine which qualities to examine, and appropriate presentation of the data can assist in clarifying the first choices.

5.1 Feature Selection

This is the process of selecting variables that are beneficial in predicting a response in machine learning. When creating predictive models, it is a good idea to figure out which attributes are significant. There are three major methods for selecting features. Filter methods, Embedded methods, and Wrapper methods. The differences between the 3 major feature selection techniques are summarized in the Table 5.1 [63].

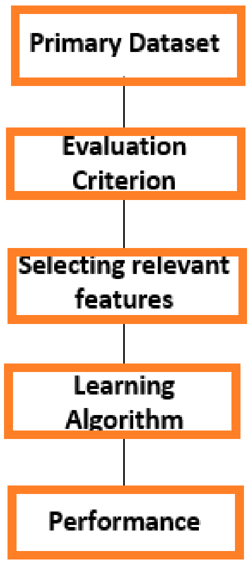


Figure 5.1: Filter-based method of feature selection

Table 5.1: Major differences between 3 feature selection techniques

Filter	**Wrapper**	**Embedded**
A collection of approaches that do not use a particular machine learning algorithm.	To discover the best characteristics, it evaluates a certain machine learning algorithm.	During the model construction phase, embeds features. During the model training phase, each iteration of the feature selection is observed.
In terms of time complexity, it is far quicker than wrapper techniques.	For a dataset with many features, the calculation time is long.	In terms of time complexity, it falls between filter methods and wrapper methods.
Overfitting is less likely.	Because it requires training machine learning models with diverse combinations of features, there is a high risk of overfitting.	Used to generally minimize overfitting by penalizing models with very high coefficients.
Examples include ANOVA, Variance Threshold, Mutual Information, Correlation etc.	Examples include Forward selection, backward selection, Bi-directional etc.	Examples include LASSO, Ridge etc.

5.1.1 Filter method

Filter methods compute the relationship between features and the target variable using mathematical techniques. To rate the significance of individual functions, filter approaches often use statistical test scores and variances. Filter methods are not dependent of the machine learning algorithms. Fig 4.1 illustrates this method. As a result, they may be utilized as input to any machine learning model and are extremely quick. Instead of cross-validation performance, it measures the intrinsic qualities of the features using univariate statistics. These approaches are more efficient and cost less to compute.

Five examples of filtering methods adopted for this DDoS attack detection model:

- **ANOVA F-value:** This calculates the degree of linearity between the input features (independent features) and output (dependent feature). A high F-value implies that the degree of linearity is strong, whereas a low F-value suggests that the degree of linearity is low. However, ANOVA F-value only captures the linear relationships between the two categories of feature.
- **Variance Threshold:** This assign threshold values to the features and gets rid of the features whose variance is less than the set threshold. It removes all zero-variance features by default, i.e., features with the same value across all

samples. It can be used for both supervised and unsupervised learning. It simply looks at the relationships between the features, not the relationships between the input and output features. The features with a greater variance contain more important information, but the drawback is that it does not take into account the link between the feature variables and the target variables.

- **Mutual information:** It measures the quantity of information gained about one feature through the other feature to measure the dependency of one variable on another. It is symmetric and non-negative, and it equals zero if and only if two random variables are independent; higher values indicate greater dependence. It can handle non-linear relationships between input and output features. Mutual information in machine learning refers to the amount of information that the presence or absence of a feature provides to making the right prediction on Y. Their MI is zero if X and Y are independent. MI is the entropy of X, which is a concept in information theory that assesses or quantifies the amount of information within a variable if X is deterministic of Y. It can be mathematically represented as shown in the equation below **0038younes**.

$$I\left(X;Y\right)=\sum_{x,y}P_{XY}\left(x,y\right)log\left[\frac{P_{XY}\left(x,y\right)}{P_{X}\left(x\right)P_{Y}\left(y\right)}\right]$$

- **SelectKBest:** This "removes everything but the k highest scoring features" after selecting the features with a function (this case, ANOVA F-value).

- **Pearson's Correlation:** A statistic that calculates the linear correlation between two continuous variables. It ranges from -1 to +1, with +1 indicating positive linear correlation, 0 indicating no linear correlation, and 1 indicating negative linear correlation. It is a well-known metric in the field of machine learning. It is a metric for expressing the strength of a linear relationship between two variables. It can be mathematically represented as shown in the equation below

$$r_{xy}=\frac{\sum_{i=1}^{n}\left(x_i-\bar{x}\right)\left(y_i-\bar{y}\right)}{\sqrt{\sum_{i=1}^{n}\left(x_i-\bar{x}\right)^2}\sqrt{\sum_{i=1}^{n}\left(y_i-\bar{y}\right)^2}}$$

  Where x_i are features, y_i are labels, $\bar{x}$ and $\bar{y}$ are the mean.

5.1.2 Feature Selection Techniques

When building a machine learning model in real-life, it's almost rare that all the variables in the dataset are useful to build a model. Repetitive factors decrease a classifier's capacity to generalise and may also lower the classifier's overall accuracy. Additionally, a model's total complexity rises as more variables are added to it [64].

5.1.2.1 Filter methods

Filter methods pick up the intrinsic properties of the features measured via univariate statistics instead of cross-validation performance. Compared to wrapper methods, these techniques are quicker and more computationally efficient. Using filter methods while working with high-dimensional data is computationally more affordable.

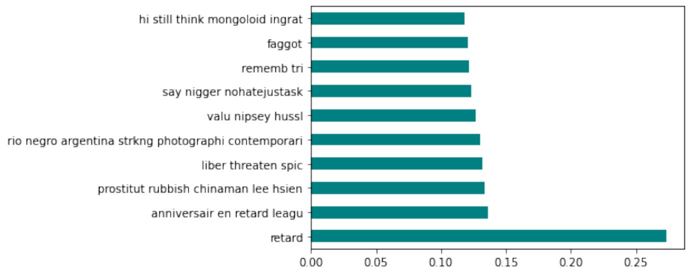
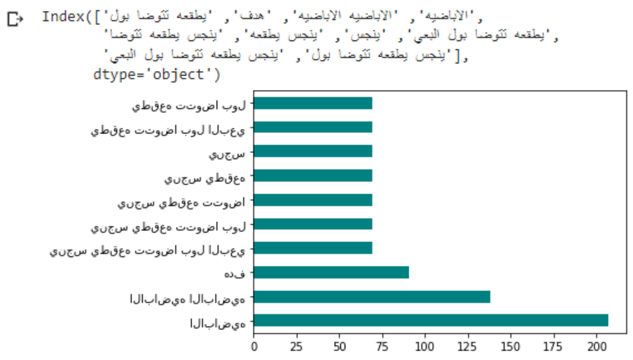
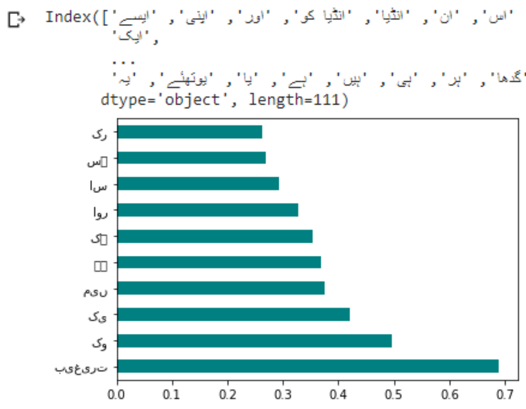


Figure 5.2: English Filter methods



```
Index(['الاباضيه', 'الاباضيه الاباضيه', 'هدف', 'يطقعه تتوضا بول',
       'يطقعه تتوضا بول البعي', 'ينجس', 'ينجس يطقعه', 'ينجس يطقعه تتوضا',
       'ينجس يطقعه تتوضا بول', 'ينجس يطقعه تتوضا بول البعي'],
      dtype='object')
```

Figure 5.3: Arabic Filter methods



```
Index(['ایسے', 'اپنی', 'اور', 'انڈیا کو', 'انڈیا', 'ان', 'اس', 'آنسو', 'آئے',
       'ایک',
       ...
       'یہ', 'یوتھئے', 'یا', 'ہے', 'ہیں', 'ہی', 'ہر', 'گدھا', 'گا', 'کے آنسو'],
      dtype='object', length=111)
```

Figure 5.4: Urdu Filter methods

5.1.2.2 Chi-square Test

The Chi-square test is used for categorical features in a dataset. We calculate Chi-square between each feature and the target and select the desired number of features with the best Chi-square scores.

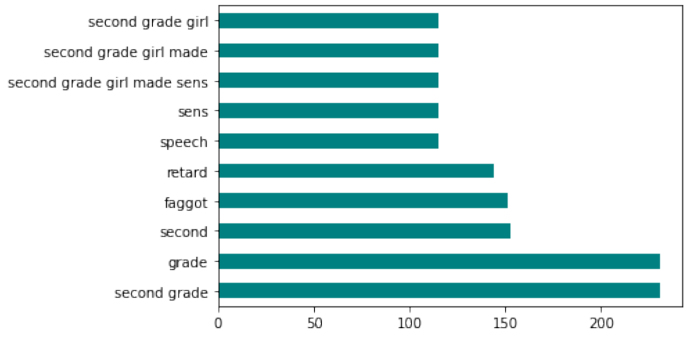
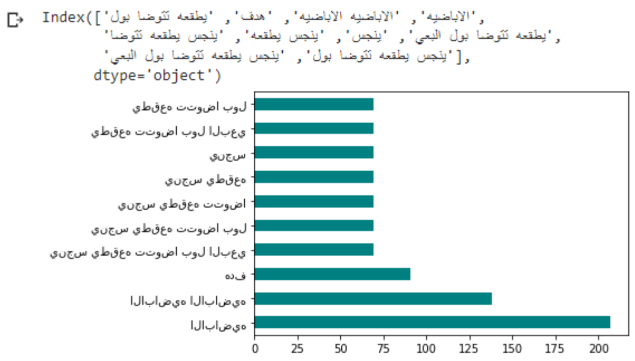
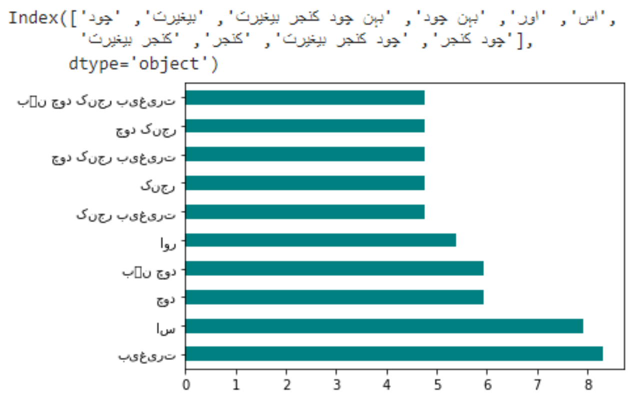


Figure 5.5: English Chi-square Test



```
Index(['الاباضيه', 'الاباضيه الاباضيه', 'هدف', 'يطقعه تتوضا بول',
       'يطقعه تتوضا بول البعي', 'ينجس', 'ينجس يطقعه', 'ينجس يطقعه تتوضا',
       'ينجس يطقعه تتوضا بول', 'ينجس يطقعه تتوضا بول البعي'],
      dtype='object')
```

Figure 5.6: Arabic Chi-square Test



```
Index(['اس', 'اور', 'بہن چود', 'بہن چود کنجر بیغیرت', 'بیغیرت', 'چود',
       'چود کنجر', 'چود کنجر بیغیرت', 'کنجر', 'کنجر بیغیرت'],
      dtype='object')
```

Figure 5.7: Urdu Chi-square Test

5.1.2.3 Fisher's Score

Fisher score is one of the most widely used supervised feature selection methods. The algorithm we'll employ returns, in descending order, the ranks of the variables based on the fisher score. The variables can then be chosen based on the situation.

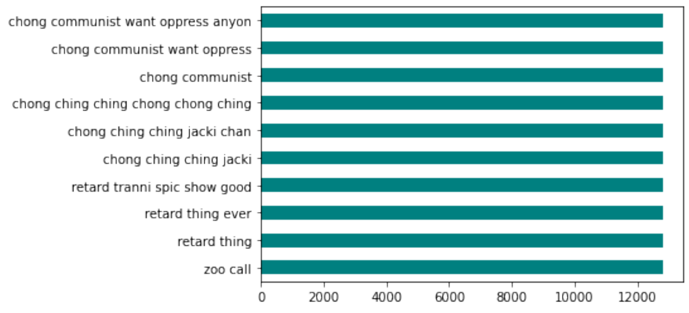


Figure 5.8: English Fisher's Score

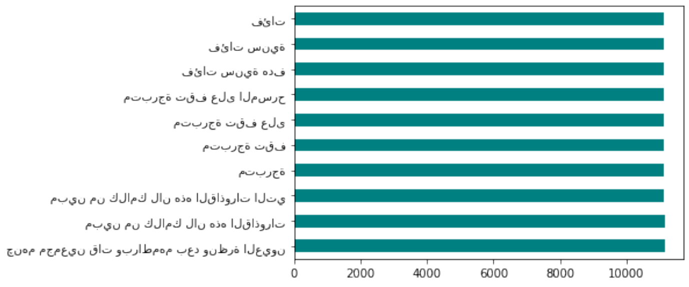


Figure 5.9: Arabic Fisher's Score

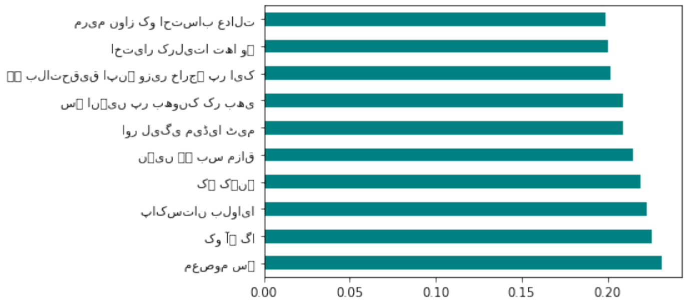


Figure 5.10: Urdu Fisher's Score

5.1.2.4 Variance Threshold

This method of text feature selection eliminate features with low importance, that is, feature with not so much relevant information. Utilizing this feature on our sample data, the relevant information is stated.

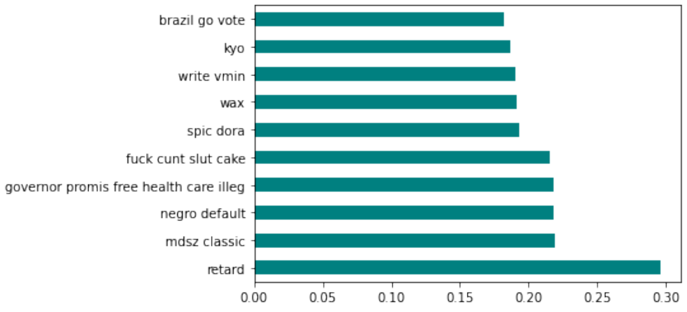


Figure 5.11: ext Feature Selection Utilizing Variance Threshold

5.1.2.5 Mean Absolute Difference (MAD)

This method is similar to the variance threshold method, with the exception that it excludes and square. This is a scaled variant that computes the mean absolute difference from a given feature's mean value. This is shown in figure below, where this method is used on a data sample.

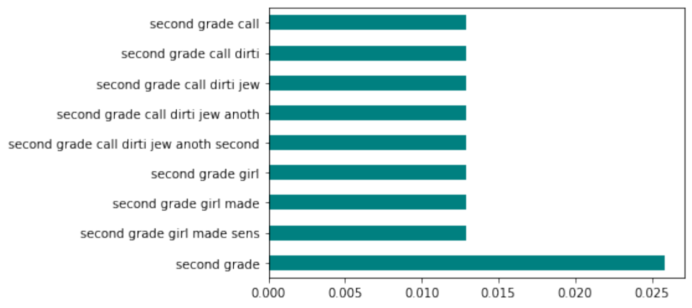


Figure 5.12: Text Feature Selection Utilizing Mean Absolute Difference

5.1.2.6 Dispersion Ratio

Higher dispersion corresponds to more relevant features. Divide the arithmetic mean and geometric mean for the given feature to get the dispersion ratio. Using this method on our sample data, the word with the highest ratio, as shown in the graph, is "hate speech."

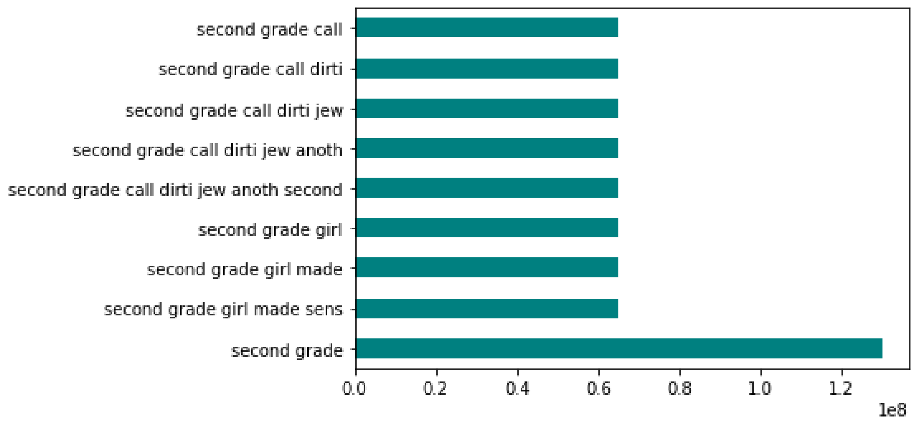


Figure 5.13: Text Feature Selection Utilizing Dispersion Ratio

5.1.2.7 Forward Feature Selection

The algorithm commences the feature selection process with empty features and then gradually adds features that improve the model. It continues this process for every iteration until the addition of a new feature does not improve the performance of the model

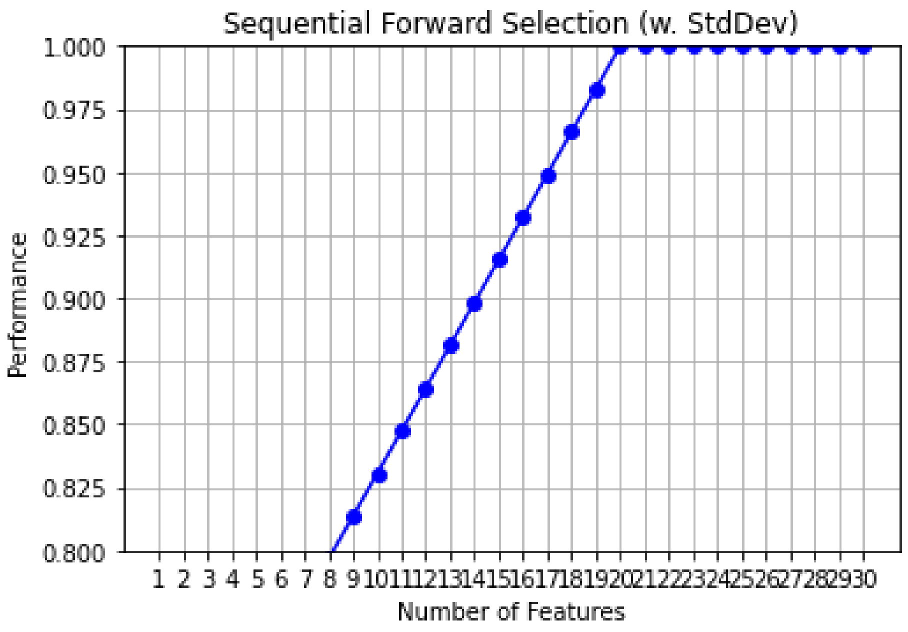


Figure 5.14: Forward Feature Selection

5.1.2.8 Random Forest Importance

Random Forests is a kind of a Bagging Algorithm that aggregates a specified number of decision trees. The tree-based strategies used by random forests naturally rank by how well they improve the purity of the node, or in other words a decrease in the impurity (Gini impurity) over all trees. Nodes with the greatest decrease in impurity happen at the start of the trees, while notes with the least decrease in impurity occur at the end of trees. Thus, by pruning trees below a particular node, we can create a subset of the most important features.

5.1.2.9 Tree based feature selection and random forest classification

In random forest classification method there is a feature importance attributes that is the feature importance (the higher, the more important the feature). !!! To use feature importance method, in training data there should not be correlated features. Random forest choose randomly at each iteration, therefore sequence of feature importance list can change.

Tree-based algorithms and models (random forest) are well-known algorithms that can give us with what we term feature importance as a technique to choose features in addition to high prediction performance. Random forests provide us with feature relevance by utilising straightforward methods like mean decrease accuracy and mean decrease impurity. One tree cannot see all the features or have access to all the observations, hence a random forest is a group of decision trees that are formed using a random sample and feature extraction from the dataset. The nodes of a decision tree are dependent on a single attribute. The dataset is supposed to be split in half by these nodes. Similar observation values will be included in one set, while dissimilar observation values will be included in another. Therefore, the "purity" of each collection determines the value of each trait.

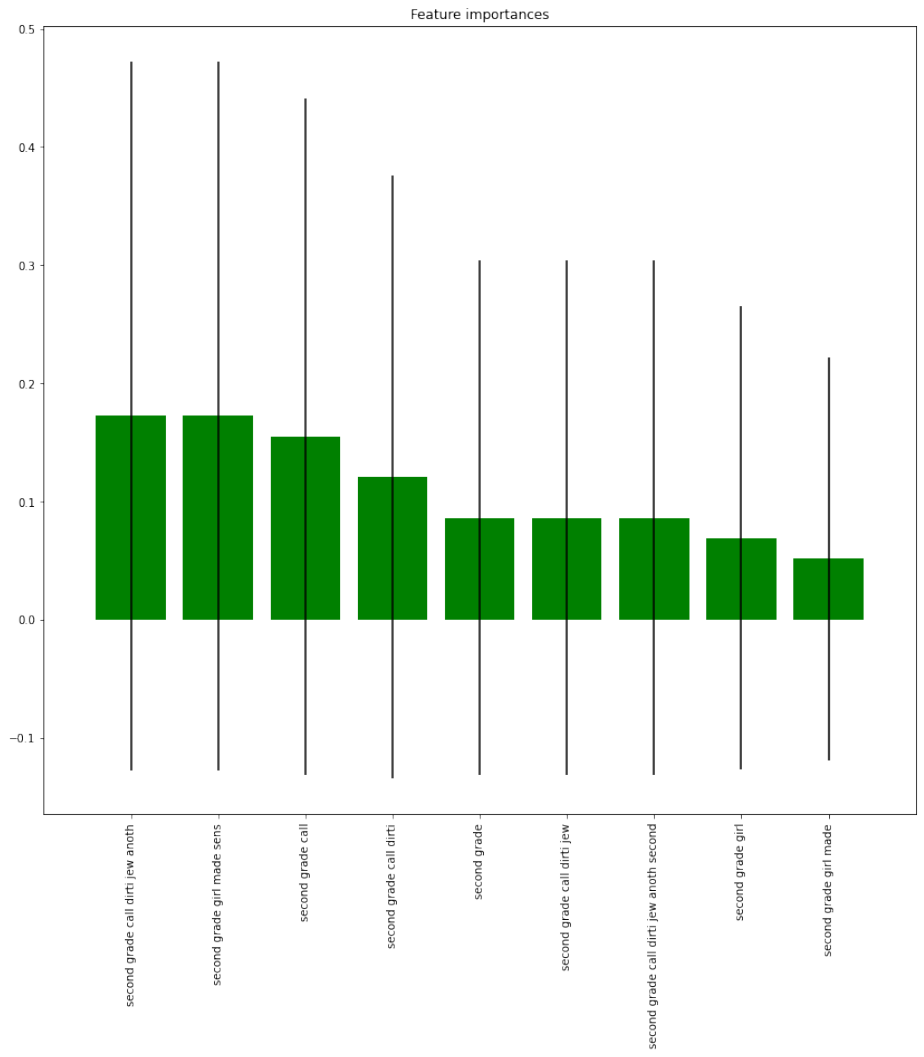


Figure 5.15: Tree based feature selection and random forest classification

5.2 Feature Engineering

Following the selection of relevant features from the datasets, some feature engineering is performed to convert the dataset's features into **Vectors** and to create new features from the dataset. Some of these engineering features are discussed in more detail below:

5.2.1 Count Vectors as features

Text data is represented by numbers 1 or 0, depending on its position. If a feature contains specific text, it is represented by a 1, otherwise by a 0. Every time the word appears, the count is increased, leaving a 0 everywhere else. This is also referred to as One-Hot Encoding. Human efforts will be time-consuming, so a Python package called CountVectorizer is available in the Scikit Learning module to assist with this feature engineering method.

5.2.2 Word Embeddings

This is a Natural Language Processing Approach in which words are encoded in vector formats so that the word closest to that vector has a similar meaning. Consider the words fruit and man: pineapple and apple, respectively. Because of our shared language, it is simple to identify similar fruits: pineapple and not man: apple. The goal of word embedding is to provide the machine with an instant understanding of words that are close to them. To calculate the accuracies of this method with various classification models, see the table below, which shows the results for each feature engineering method to determine which is best for our dataset.

5.3 Visualization

A lot of information is hidden behind data, as well as issues with determining the structure of the data. Understanding data trends and patterns, assessing data frequency and other features, determining the distribution of variables in the data, and finally displaying the link that may exist between various variables are all necessary.

5.3.1 RadViz

This is a multivariate data visualisation technique that depicts points on the inside of a circle, normalising their values on the axes from the centre to each arc, then depicts each feature dimension evenly around the circumference of the circle. This technique allows for as many dimensions as can fit on a circle, significantly increasing the dimensionality of the visualisation. RadViz charts are widely used to depict multidimensional data because they use the familiar concept of 2D points for storing data components and displaying the original data dimensions that function as springs for setting the x and y coordinates [65].
Radviz provides an easy way to visualise an N-dimensional data set by projecting it into a simple 2D environment where the effect of each dimension can be read as a balance between the influence of all dimensions. Each dimension in the dataset is represented by a dimensional anchor, which is uniformly distributed over a unit circle. Each dimensional anchor is connected by a spring to each line in the data set, which corresponds to a point in the projection. Classes of 'normal,' 'udp,' 'tcp,' and 'icmp' are created when using Radviz to visualise DDoS data. These classes are the data points inside the circle, with different colours for each class.

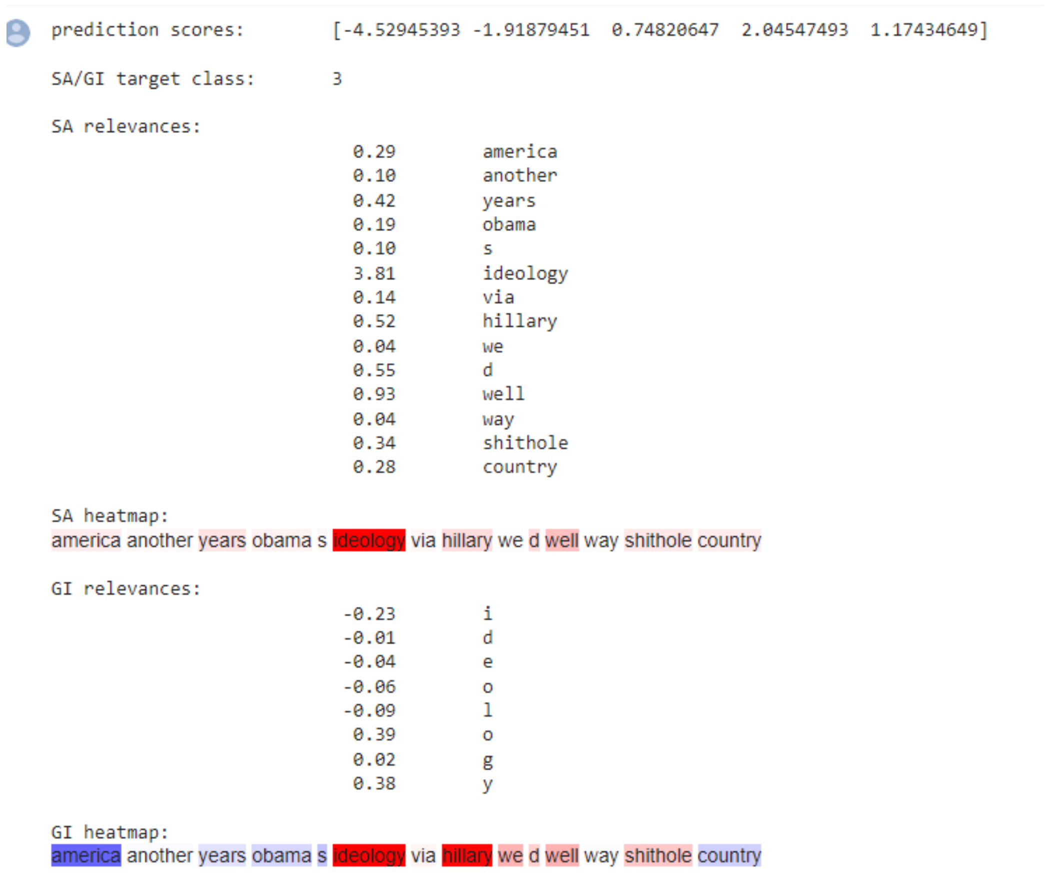
5.3.2 Parallel Coordinates

Parallel coordinates are a popular method for viewing and analysing large datasets. To represent a group of points in an n-dimensional space, a background is drawn consisting of n parallel lines, usually vertical and evenly spaced. This type of visualisation is used to plot multivariate numerical data. Parallel Coordinates Plots are useful for comparing and visualising the relationships between multiple variables.

Parallel coordinates are a typical technique of displaying and interpreting multivariate data in high-dimensional geometry. In a Parallel Coordinates Plot, each variable has its own axis, and all of the axes are parallel to each other. Because each variable has its own unit of measurement, each axis can have its own scale, or all of the axes can be normalised to keep all of the scales uniform. A design with polylines depicting multivariate items intersecting with parallel axes representing variables can be used to analyse many features of a multivariate data collection. [66].

5.3.3 Layer-wise Relevance Propagation (LRP) for LSTM

Parallel coordinates are a typical technique of displaying and interpreting multivariate data in high-dimensional geometry. In a Parallel Coordinates Plot, each variable has its own axis, and all of the axes are parallel to each other. Because each variable has its own unit of measurement, each axis can have its own scale, or all of the axes can be normalized to keep all of the scales uniform. A design with polylines depicting multivariate items intersecting with parallel axes representing variables can be used to analyze many features of a multivariate data collection [67].



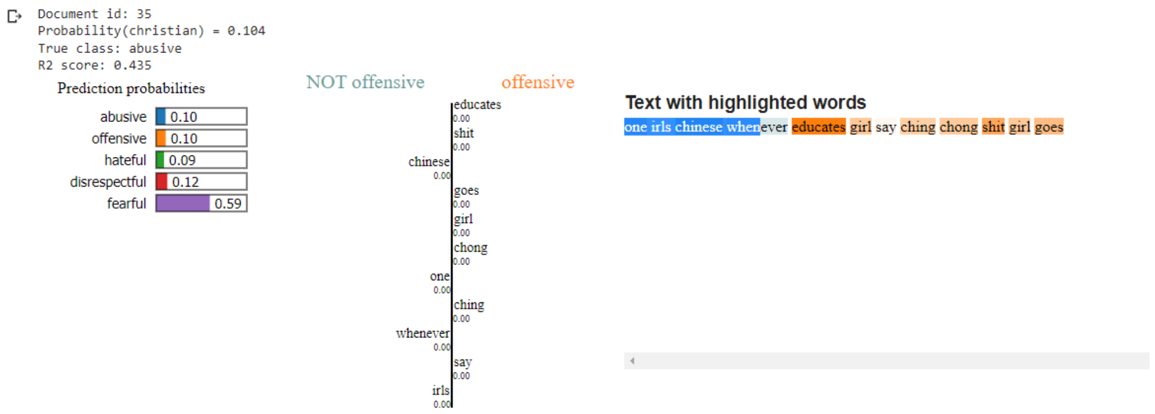
```
prediction scores:        [-4.52945393 -1.91879451  0.74820647  2.04547493  1.17434649]

SA/GI target class:       3

SA relevances:
                             0.29       america
                             0.10       another
                             0.42       years
                             0.19       obama
                             0.10       s
                             3.81       ideology
                             0.14       via
                             0.52       hillary
                             0.04       we
                             0.55       d
                             0.93       well
                             0.04       way
                             0.34       shithole
                             0.28       country

SA heatmap:
america another years obama s ideology via hillary we d well way shithole country

GI relevances:
                            -0.23       i
                            -0.01       d
                            -0.04       e
                            -0.06       o
                            -0.09       l
                             0.39       o
                             0.02       g
                             0.38       y

GI heatmap:
america another years obama s ideology via hillary we d well way shithole country
```

Figure 5.16: example of Layer-wise Relevance Propagation (LRP) for LSTM

5.3.4 Lime and Shap

Both SHAP and LIME are well-known Python libraries for model explainability. SHAP (SHapley Additive exPlanation) uses Shapley values to score model feature influence. A Shapley value is defined as the "average marginal contribution of a feature value across all possible coalitions." In other words, Shapley values take into account all possible predictions for an instance using all possible input combinations. SHAP can guarantee properties such as consistency and local accuracy because of this thorough approach.

LIME (Local Interpretable Model-agnostic Explanations) constructs sparse linear models around each prediction in order to explain how the black box model works in that specific area. The authors of SHAP demonstrate in their NIPS paper that Shapley values are the only guarantee of accuracy and consistency, and that LIME is a subset of SHAP but lacks the same properties.



```
Document id: 35
Probability(christian) = 0.104
True class: abusive
R2 score: 0.435
```

Figure 5.17: Lime Value outPut

The concept of LIME is visually attractive (especially for text), and indeed accepts any classifier, as long as it can probe it's .Although LIME promises to optimize between interpretability/simplicity and faithfulness (in an elegant equation in the paper), the algorithm does not do this for us. The user specifies the number of coefficients (simplicity), and a linear regression is fitted to the samples. Furthermore, as we will see, sampling, selecting a kernel size that defines locality, and regularising the linear model can be difficult.

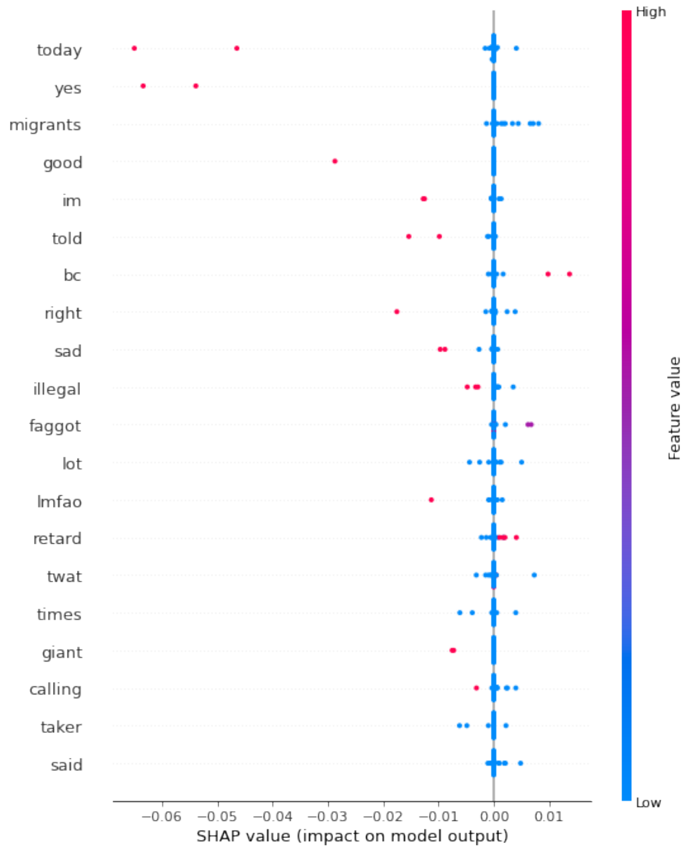


Figure 5.18: SHAP Value outPut

We are using SHAP's KernelExplainer because it is model-agnostic and can explain the same NLP logistic regression model that we implemented above.

Because the KernelExplainer is a particularly time-consuming algorithm, it was necessary for this implementation to generate smaller samples from the training and test sets in order to run the model.

5.3.5 TF-IDF (Term Frequency - Inverse Document Frequency)

TF-IDF stands for term frequency-inverse document frequency and it is a measure, used in the fields of information retrieval (IR) and machine learning, that can quantify the importance or relevance of string representations (words, phrases, lemmas, etc) in a document amongst a collection of documents (also known as a corpus).

We will need a vectorizer because we will be using bag-of-words classifiers: an object that generates a vector for each text instance, indicating the presence/absence or counts of each word in the vocabulary. The Tf-idf vectorizer also gives less frequently occurring words more weight.

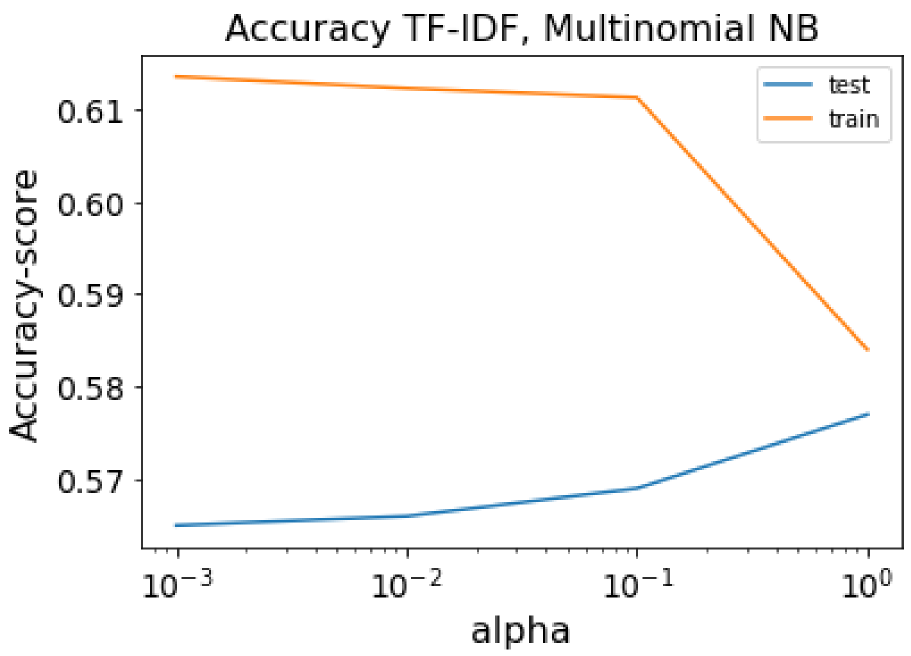


Figure 5.19: Accuracy TF-IDF, Multinomial NB

An accuracy, of over 55 % on the test data. Only a minimal smoothing of the counts using pseudocounts is needed: An alpha of 1E-2 will do.

Chapter 6

Results and Discussions

Collect Data from public resource for three Language (English , Arabic and Urdu) and the data is preprocessed as discussed in Chapter 4 and Chapter 5 . The preprocessed data is fed into various natural language processing models for statistical text features, as described in Chapter 3. The data is then fed into various text feature selection methods, which select the best features from the datasets to provide relevant and accurate results. The process of removing irrelevant data from a dataset is known as feature selection. If the correct subset is chosen, the model's accuracy improves and the model can train its data faster. Filter Methods, Wrapper Methods, and Embedded Methods are some of the feature selection methods used.

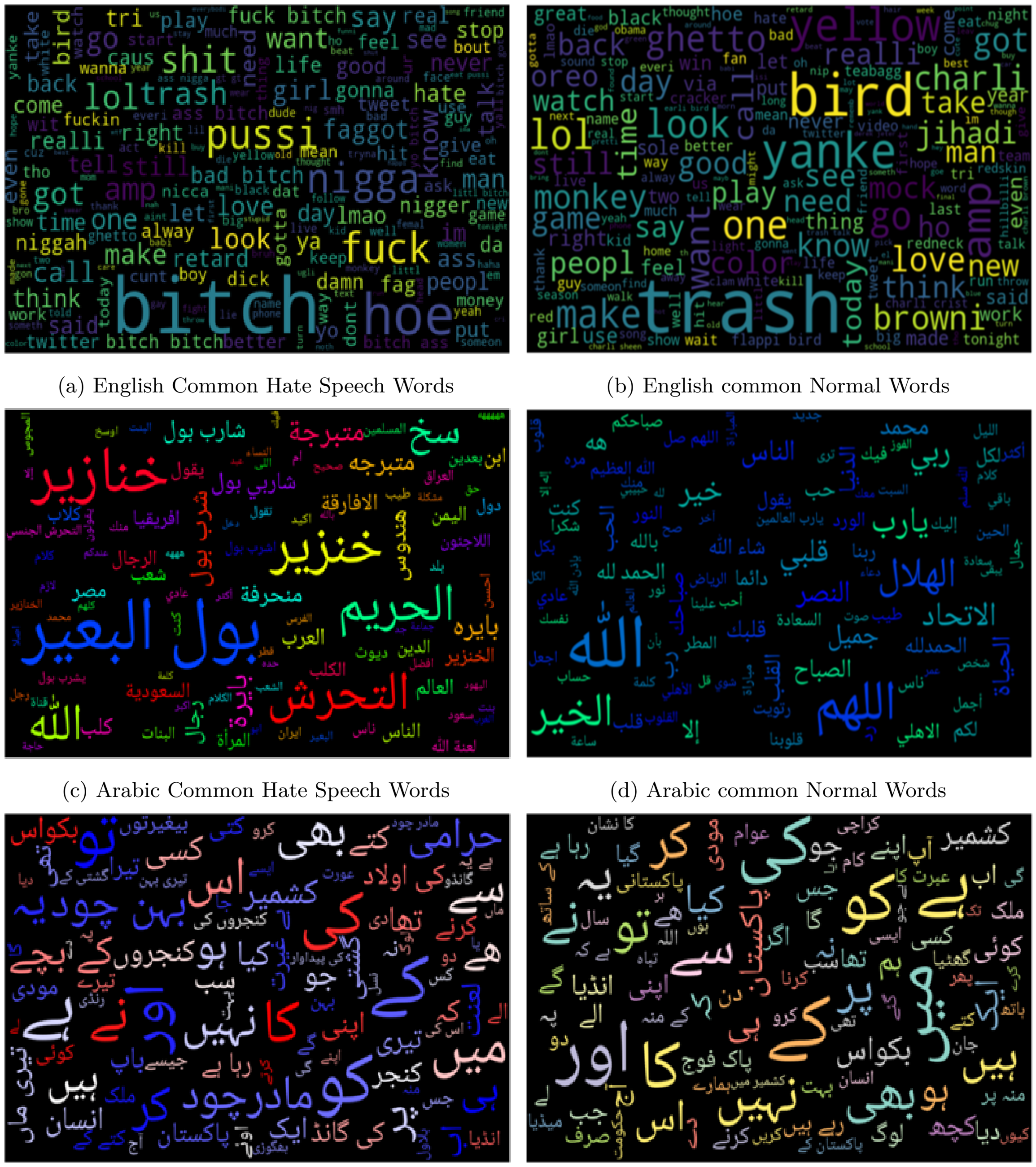
6.1 Data Exploration

6.1.1 Word Cloud

A word cloud is a powerful visual representation object for text processing that shows the frequency and importance of specific words based on their size. The larger the word, the more important it is. Figure 6.1 depicts the word clouds of the three datasets' positive classes. We can see from the three word clouds that the English dataset, Arabic dataset, and Urdu dataset have common hate speech and normal speech.

6.1.2 pyLDAvis

pyLDAvis is a Python library for interactive LDA visualisation. Each circle represents a distinct topic, the size of the circle represents the topic's importance, and the distance between each circle represents how similar the topics are to one another. When you select a topic/circle, you'll see a horizontal bar chart with the 30 most relevant words for the topic, as well as the frequency of each word appearing in the



(a) English Common Hate Speech Words (b) English common Normal Words

(c) Arabic Common Hate Speech Words (d) Arabic common Normal Words

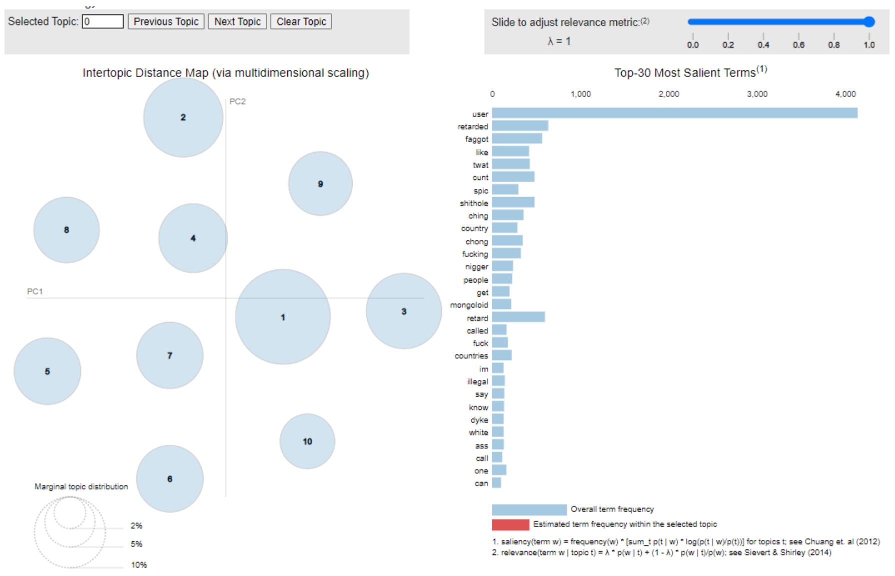
(e) Urdu Common Hate Speech Words (f) Urdu common Normal Words

Figure 6.1: The Wordcloud of frequent words classified three Languages

topic and the overall corpus.

The relevance metric distinguishes between words that are distinct/exclusive to the topic ($\lambda\lambda$ closer to 1.0).and words that are likely to be included in the selected topic ($\lambda\lambda$ closer to 1.0).

Figure 6.2 illustrates the topic keywords from our LDA model of user timeline tweets from a user that authored a sample hateful tweet from Arabic dataset. We see that our largest topic, includes terms like 'retarded', 'faggot', 'twat', 'shithole', 'fuck', 'nigger', 'fucking', 'cunt' and 'dyke' implying that this user frequently tweets about Hate Speech.



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 6.2: Topic LDA Modeling of User's Timeline from English Hateful Tweet

Figure 6.3 illustrates the topic keywords from our LAD model of user timeline tweets from a user that authored a sample hateful tweet from Arabic dataset.

Figure 6.4 illustrates the topic keywords from our LAD model of user timeline tweets from a user that authored a sample hateful tweet from Urdu dataset.

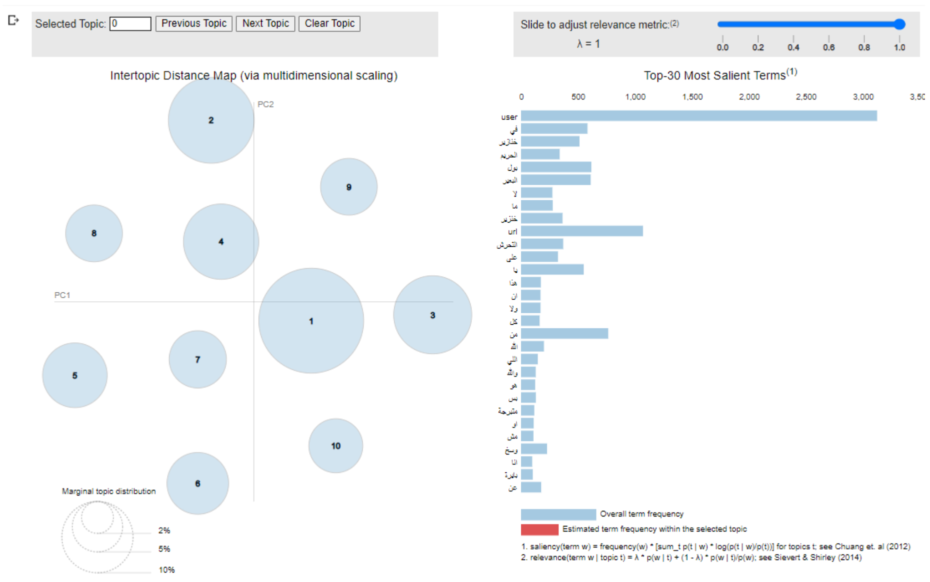

Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0 0.2 0.4 0.6 0.8 1.0
Intertopic Distance Map (via multidimensional scaling)
PC1
PC2
1 2 3 4 5 6 7 8 9 10
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms(1)
0 500 1,000 1,500 2,000 2,500 3,000 3,500
user
في
خنازير
الحريم
بول
البعير
لا
ما
خنزير
url
التحرش
على
يا
هذا
ان
ولا
كل
من
الله
اللي
والله
هو
بس
متبرجة
او
مش
وسخ
انا
بايرة
عن
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 6.3: Topic LDA Modeling of User's Timeline from Arabic Hateful Tweet

6.1.3 Sentiment and Polarity Distribution

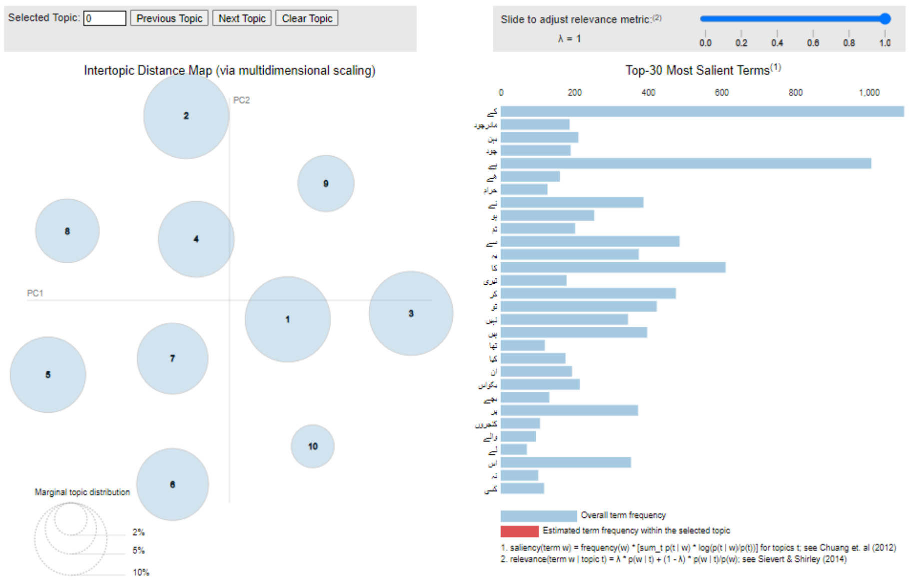
Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative, or neutral.For calculating polarity of a text, polarity score of each word of the text, if present in the dictionary, is added to get an 'overall polarity score' , Figure 6.5 and Figure 6.6 and Figure 6.7 illustrates Sentiment polarity for an element defines the orientation of the expressed sentiment it determines if the text expresses the positive, negative or neutral sentiment of the user about the entity in consideration.

6.1.4 Frequency of Common Words of All Languages

Figure 6.8 illustrates the diversity of annotations in the original datasets (English , Arabic , Urdu). This diversity of annotations translates the variety of hate speech Frequency of Common Words of All Languages .

6.2 Sentiment Classification

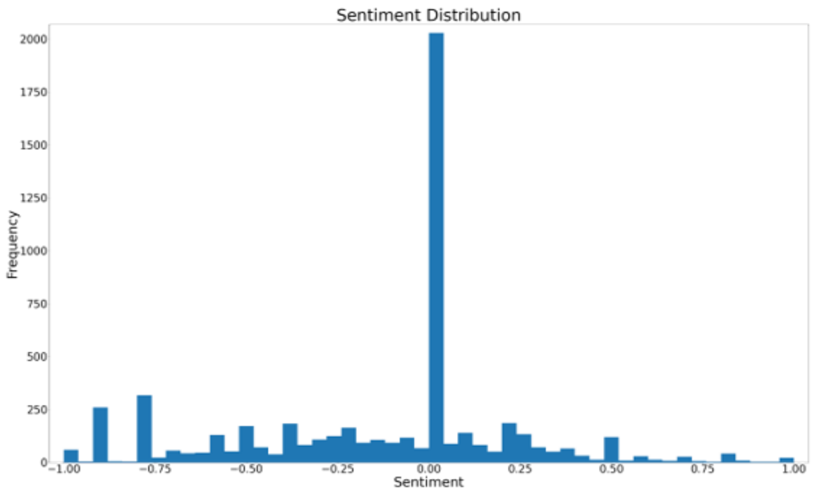
We used a publicly available Twitter dataset. The dataset has been preprocessed. Natural language processing has removed stop words, special characters, and irrelevant words. Text feature extraction has been performed in order to select the most relevant



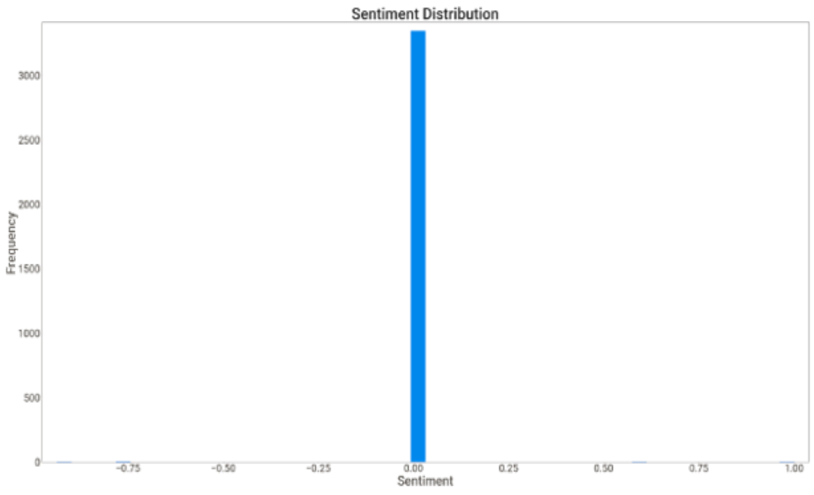
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 6.4: Topic LAD Modeling of User's Timeline from Urdu Hateful Tweet

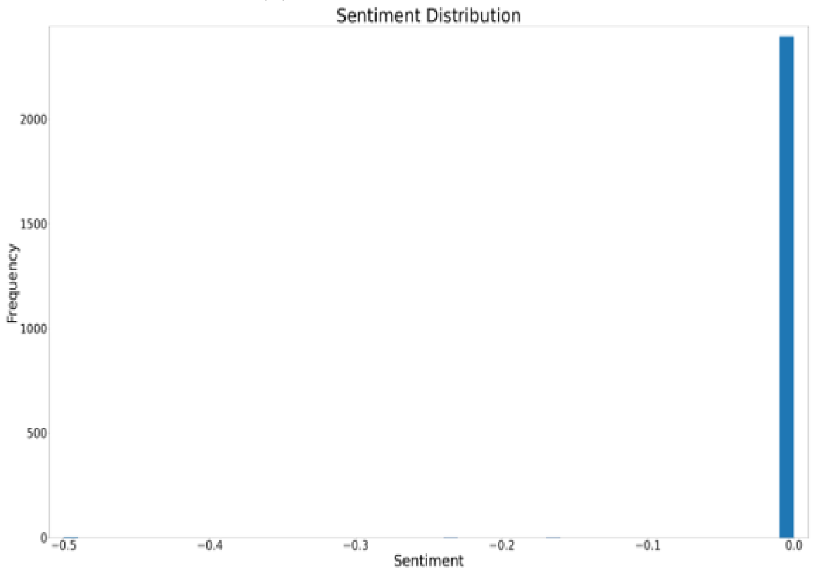
features that will result in accurate model performance. Feature engineering was also used to convert the features into vectors. The accuracy, F1 score, precision, recall, and kappa parameters of the scikit learn metrics were evaluated. To test these parameters, 13 models were created. Table 6.1 , Table 6.2, Table 6.3 and Table 6.4 displays the metrics for the models.



(a) English Sentiment Score

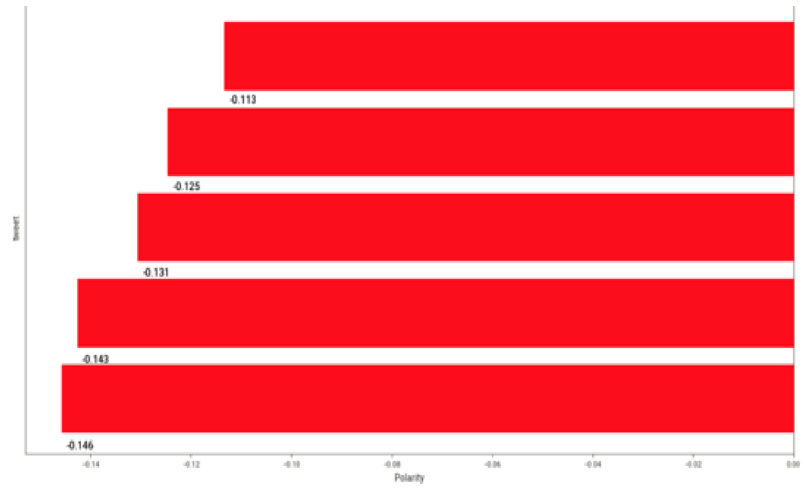


(b) Arabic Sentiment Score

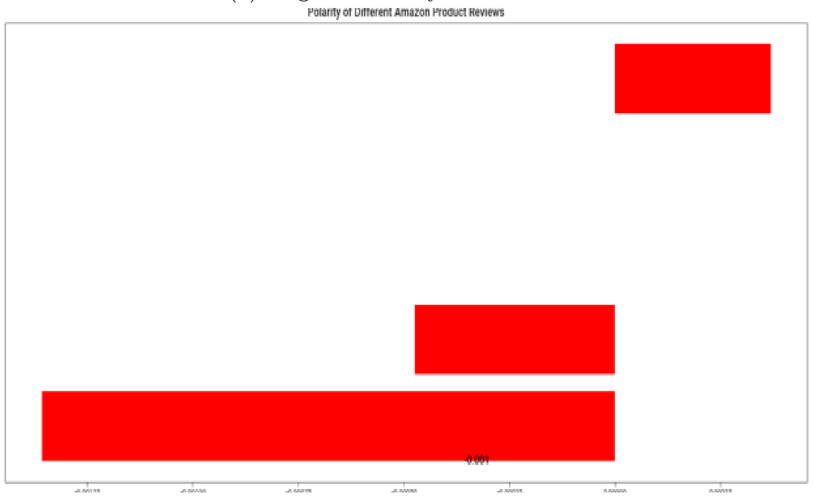


(c) Urdu Sentiment Score

Figure 6.5: Sentiment Score



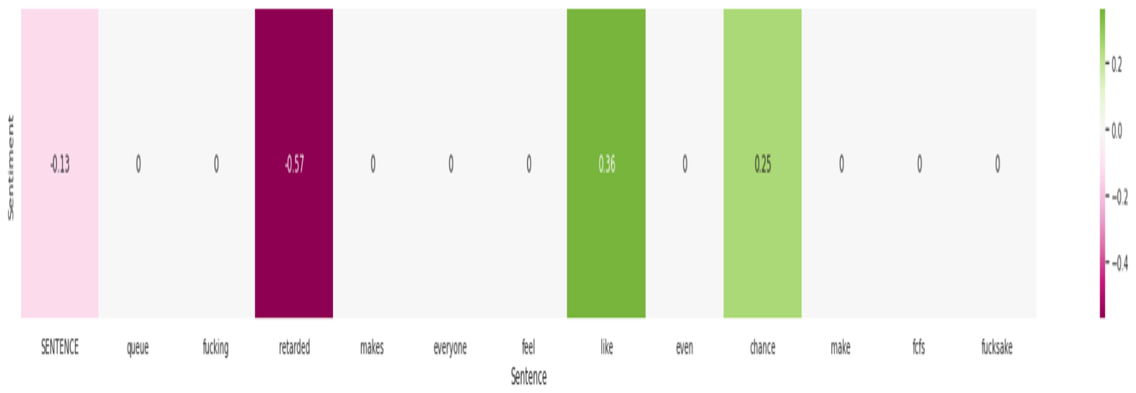
(a) English Polarity Distribution



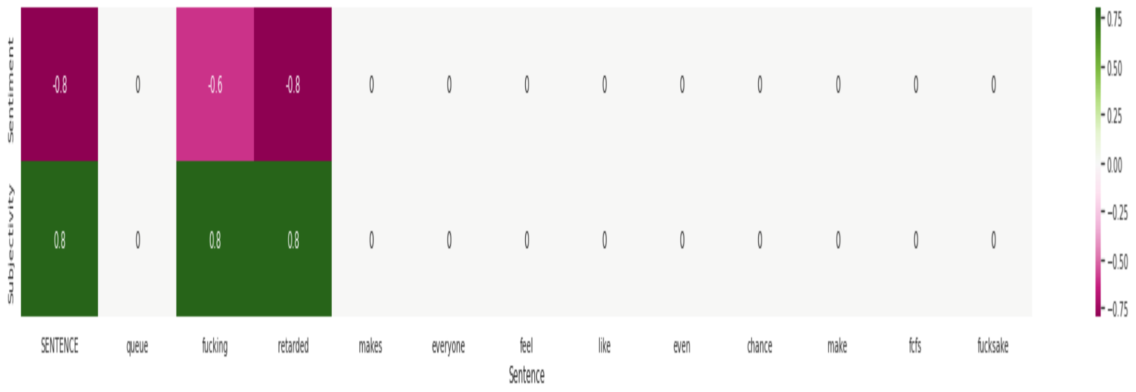
(b) Arabic Polarity Distribution

(c) Urdu Polarity Distribution

Figure 6.6: Polarity Distribution

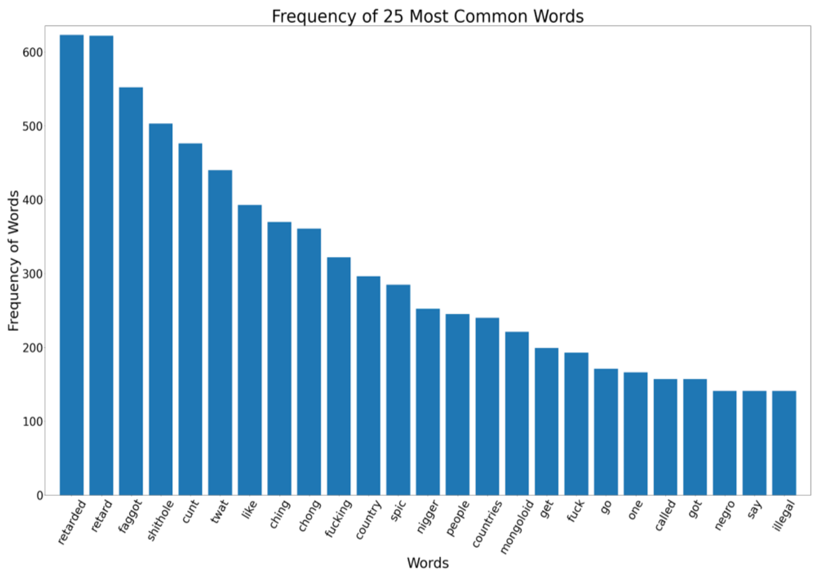


(a) English Sentiment Score

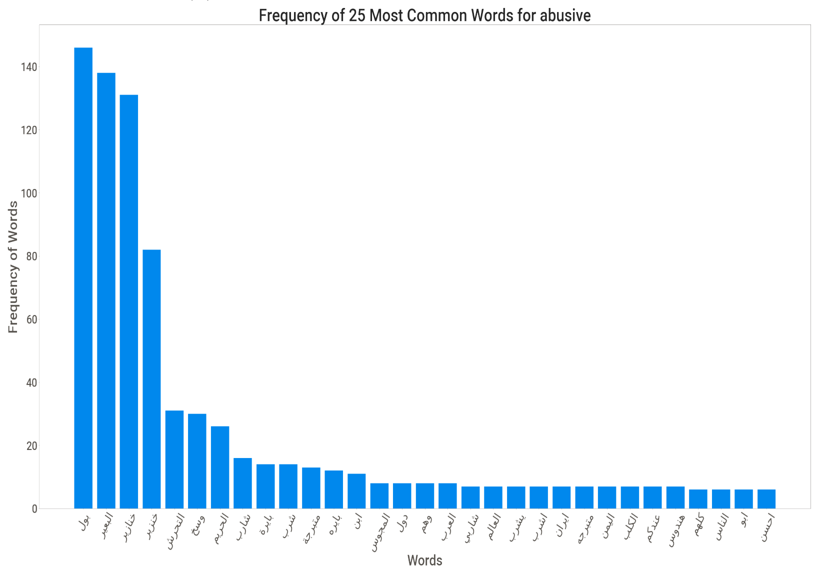


(b) Arabic Sentiment Score

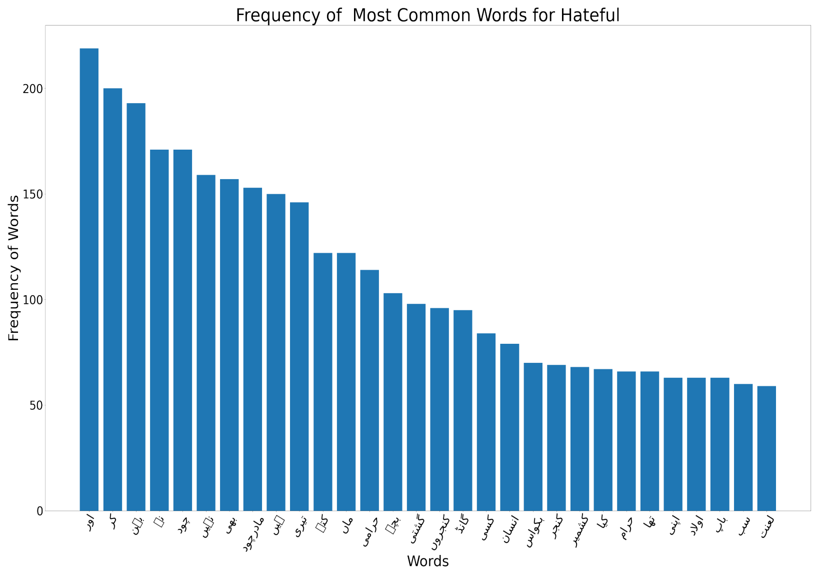
Figure 6.7: Sentiment Score



(a) Frequency of Common English Words



(b) Frequency of Common Arabic Words



(c) Frequency of Common Urdu Words

Figure 6.8: Frequency of Common Words of All Languages

Table 6.1: Twitter Text Classification Results

Classifier	Accuracy	Precision	Recall	F1	Kappa
AdaBoost	0.567568	0.474223	0.567568	0.419477	0.028911
Decision Tree	0.560811	0.316648	0.560811	0.404759	0.006198
Random Forest	0.560811	0.314509	0.560811	0.403007	0.000000
GradientBoost	0.560811	0.440411	0.560811	0.429602	0.044497
Gaussian Process Classifier	0.560811	0.314509	0.560811	0.403007	0.000000
Support Vector Machine 1	0.560811	0.314509	0.560811	0.403007	0.000000
Support Vector Machine 2	0.560811	0.318817	0.560811	0.406527	0.013637
MLP	0.547297	0.315456	0.547297	0.400226	-0.005272
LGBM	0.547297	0.319899	0.547297	0.403784	0.007904
LogisticRegression	0.540541	0.315950	0.540541	0.398799	-0.004391
SGD	0.500000	0.325787	0.500000	0.387455	-0.039484
LinearDiscriminant	0.493243	0.305516	0.493243	0.377320	-0.050142
QuadraticDiscriminant	0.493243	0.370319	0.493243	0.408572	0.006711
Nearest Neighbors	0.331081	0.361296	0.331081	0.338472	-0.040108
GaussianNB	0.243243	0.457335	0.243243	0.297337	0.040241

6.3 Supervised and Unsupervised

6.3.1 word to sentence embedding

Because we have a large number of high-quality word vectors, we can create document vectors from them.

6.3.2 Unsupervised Evaluation

On the hate speech Multilanguage dataset, unsupervised evaluation of the learned sentence embeddings is performed using sentence cosine similarity. These similarity scores are compared to the gold-standard human judgements using Pearson's correlation scores.

6.3.3 Supervised evaluation

Sentence embeddings are tested for use in a variety of supervised classification tasks. We test classification of movie review sentiment (MR) (Pang Lee, 2005), subjectivity (SUBJ) (Pang Lee, 2004), and question type (TREC) (Voorhees, 2002). Model embeddings are calculated from input sentences and fed directly to a logistic regression

Table 6.2: Count Vectorizer Classification Results

Classifier	Accuracy	Precision	Recall	F1	Kappa
Nearest Neighbors	0.486364	0.556550	0.486364	0.495014	0.199098
LogisticRegression	0.972727	0.973913	0.972727	0.972239	0.953635
Decision Tree	0.627273	0.644558	0.627273	0.507823	0.105425
Random Forest	0.600000	0.360000	0.600000	0.450000	0.000000
MLP	0.995455	0.995489	0.995455	0.995288	0.992422
AdaBoost	0.622727	0.500656	0.622727	0.500824	0.119236
GaussianNB	0.954545	0.977557	0.954545	0.961827	0.927347
LinearDiscriminant	0.981818	0.981887	0.981818	0.981669	0.969688
QuadraticDiscriminant	0.940909	0.957021	0.940909	0.943747	0.902575
GradientBoost	0.740909	0.819048	0.740909	0.673504	0.453857
MultinomialNB	0.936364	0.940530	0.936364	0.935185	0.889332
SGD	1.000000	1.000000	1.000000	1.000000	1.000000
LGBM	0.609091	0.454212	0.609091	0.468968	0.034891
Gaussian Process Classifier	0.600000	0.360000	0.600000	0.450000	0.000000
Support Vector Machine 1	0.759091	0.828108	0.759091	0.724814	0.499033
Support Vector Machine 2	1.000000	1.000000	1.000000	1.000000	1.000000

Table 6.3: Doc2Vec Classification Results

Classifier	Accuracy	Precision	Recall	F1	Kappa
Nearest Neighbors	0.640909	0.654188	0.640909	0.632596	0.382988
LogisticRegression	0.600000	0.360000	0.600000	0.450000	0.000000
Decision Tree	0.618182	0.576335	0.618182	0.545948	0.199515
Random Forest	0.618182	0.677463	0.618182	0.490242	0.072382
MLP	0.600000	0.360000	0.600000	0.450000	0.000000
AdaBoost	0.563636	0.451619	0.563636	0.466275	0.039083
GaussianNB	0.122727	0.663272	0.122727	0.074176	0.056633
LinearDiscriminant	0.750000	0.755165	0.750000	0.750944	0.592332
QuadraticDiscriminant	0.404545	0.551461	0.404545	0.441245	0.194094
GradientBoost	0.763636	0.798167	0.763636	0.732460	0.525232
SGD	0.650000	0.599476	0.650000	0.576403	0.258319
LGBM	0.845455	0.868979	0.845455	0.833718	0.709323
Gradient Process Classifier	0.600000	0.360000	0.600000	0.450000	0.000000
Support Vector Machine 1	0.600000	0.360000	0.600000	0.450000	0.000000
Support Vector Machine 2	0.600000	0.360000	0.600000	0.450000	0.000000

Table 6.4: Hashing Vectorizer Classification Results

Classifier	Accuracy	Precision	Recall	F1	Kappa
Nearest Neighbors	0.668182	0.685673	0.668182	0.636810	0.403528
LogisticRegression	0.600000	0.360000	0.600000	0.450000	0.000000
Decision Tree	0.618182	0.548485	0.618182	0.487066	0.067326
Random Forest	0.600000	0.360000	0.600000	0.450000	0.000000
MLP	0.600000	0.360000	0.600000	0.450000	0.000000
AdaBoost	0.600000	0.360000	0.600000	0.450000	0.000000
GaussianNB	0.595455	0.389138	0.595455	0.462134	0.028722
LinearDiscriminant	0.600000	0.360000	0.600000	0.450000	0.000000
QuadraticDiscriminant	0.609091	0.442499	0.609091	0.478413	0.060902
GradientBoost	0.631818	0.644039	0.631818	0.517358	0.119086
SGD	0.600000	0.360000	0.600000	0.450000	0.000000
LGBM	0.645455	0.745325	0.645455	0.542662	0.161741
Gaussian Process Classifier	0.600000	0.360000	0.600000	0.450000	0.000000
Support Vector Machine 1	0.600000	0.360000	0.600000	0.450000	0.000000
Support Vector Machine 2	0.609091	0.595121	0.609091	0.470061	0.033362

classifier. For the MR and SUBJ datasets, accuracy scores are calculated using 10-fold cross-validation. Nested cross-validation is used to tune the L2 penalty for those datasets. The accuracy is computed on the test set for the TREC dataset.

6.3.4 Evaluation Result

Table 6.5 displays the Evaluation Result, Higer means better with an exception in MSE.

Table 6.5: Evaluation Result

S.No.	Model Name	Pearson	Spearman	MSE	SUBJ	MR	TREC
1.	Doc2Vec	0.35	0.34	4.54	0.797	0.715	0.542
2.	Sent2Vec	0.52	0.45	1.62	0.890	0.772	0.642
3.	Word2Vec with simple average	0.60	0.51	1.88	0.925	0.757	0.688
4.	Word2Vec with TF-IDF	0.63	0.51	0.97	0.913	0.758	0.684
5.	word2Vec with SIF	0.59	0.50	1.71	0.904	0.703	0.484

6.4 Semi-Supervised Performance

Recently, the machine learning community has paid a lot of attention to semi-supervised learning (SSL). When labels are few or expensive to get, semi-supervised learning (SSL) offers a potent framework for utilising unlabeled data. We begin by evaluating multiple heterogeneous pairs of text vectorization algorithms (such as N-Grams, World2Vec Skip-Gram, AraBert, and DistilBert) and machine learning algorithms to determine the best classifier for semi-supervised learning (SSL) (such as SVM, CNN and BiLSTM). Semi-supervised learning (SSL) is the appropriate machine learning to create the appropriate label predictions with a dataset that contains both labelled and unlabeled data, such as our different language datasets. To create different models for accurate predictions, different semi-supervised learning classifiers were used.

The performance evaluation for the different models with English dataset is displayed in Table 6.6 , From the performance evaluation of the semi-supervised learning models, three different models have result more than of 0.9 while the GaussianNB model has the best performance of .99 Comparing the performance results of semi-supervised learning , while KNeighbors Classifier model have low performance with .076. Semi-supervised learning is appropriate for data with a combination of some labeled data and large unlabeled data.

Table 6.6: Semi Supervised Evaluation Result for English data set

.	classifier	Train Score	Test Score	SS Train Score	SS Test Score
2	GaussianNB()	0.022309	0.019837	0.410392	0.988197
3	Logistic Regression()	0.433782	0.419412	0.656840	0.987124
1	DecisionTree (max$_depth = 5$)	0.561261	0.537726	0.729799	0.976395
0	KNeighbors (n$_neighbors = 3$)	0.596671	0.324478	0.717073	0.785408

The performance evaluation for the different models with Arabic dataset is displayed in Table 6.7 , From the performance evaluation of the semi-supervised learning models, three different models have result more than of 0.9 while the Decision Tree Classifier model has the best performance of 0.99 Comparing the performance results of semi-supervised learning , while KNeighbors Classifier model have low performance with 0.79 . Semi-supervised learning is appropriate for data with a combination of some labeled data and large unlabeled data.

Table 6.7: Semi Supervised Evaluation Result for Arabic data set

.	classifier	Train Score	Test Score	SS Train Score	SS Test Score
1	DecisionTree(max$_depth = 5$)	0.569469	0.518496	0.741336	0.989170
3	LogisticRegression()	0.539058	0.498210	0.716685	0.976534
2	GaussianNB()	0.066786	0.048329	0.430154	0.963899
0	KNeighbors(n$_neighbors = 3$)	0.671437	0.414678	0.769918	0.790614

The performance evaluation for the different models with Urdu dataset is displayed in Table 6.8, , From the performance evaluation of the semi-supervised learning models, three different models have result more than of 0.9 while the GaussianNB model has the best performance of 1.0 Comparing the performance results of semi-supervised learning , while KNeighbors Classifier model have low performance with 0.74 . Semi-supervised learning is appropriate for data with a combination of some labeled data and large unlabeled data.

Table 6.8: Semi Supervised Evaluation Result for Urdu data set

.	classifier	Train Score	Test Score	SS Train Score	SS Test Score
2	GaussianNB()	0.511667	0.500833	0.707585	1.000000
3	LogisticRegression()	0.542500	0.493333	0.704591	0.964646
1	DecisionTree(max$_depth = 5$)	0.686667	0.570833	0.786427	0.924242
0	KNeighbors(n$_neighbors = 3$)	0.777500	0.559167	0.836327	0.843434

6.5 Performance evaluation by classifying sample document using LDA TF-IDF model

Performance evaluation by classifying sample document using LDA Bag of Words model ,We checked where our test document would be classified.explore the words occuring in that topic and its relative weight.
lda_model_tfidf = gensim.models.LdaMulticore(corpus_tfidf, num_topics=10, id2word=dictionary, passes=2, workers=4)
for idx, topic in lda_model_tfidf.print_topics(-1):
print('Topic: Word: '.format(idx, topic))

Table 6.9: Performance evaluation by classifying sample document using LDA TF-IDF model

Score	Topic
Score: 0.7749735116958618	Topic: 0.112*"cunt" + 0.050*"retard" + 0.032*"faggot" + 0.029*"come" + 0.027*"fuck" + 0.027*"look" + 0.022*"trump" + 0.020*"terrorist" + 0.018*"make" + 0.018*"shithol"
Score: 0.025014081969857216	Topic: 0.134*"faggot" + 0.070*"fuck" + 0.057*"retard" + 0.049*"mongoloid" + 0.026*"tell" + 0.019*"call" + 0.018*"know" + 0.017*"say" + 0.015*"peopl" + 0.014*"shut"
Score: 0.02500251494348049	Topic: 0.033*"countri" + 0.033*"retard" + 0.028*"shithol" + 0.027*"faggot" + 0.025*"twat" + 0.024*"think" + 0.021*"world" + 0.020*"shit" + 0.019*"spic" + 0.018*"like"
Score: 0.025002513080835342	Topic: 0.117*"ching" + 0.115*"chong" + 0.051*"negro" + 0.023*"spic" + 0.021*"fuck" + 0.020*"cunt" +0.019 *"faggot" + 0.017*"hate" + 0.016*"like" + 0.014*"racist"
Score: 0.02500232122838497	Topic: 0.201*"retard" + 0.132*"twat" + 0.021*"fuck" + 0.021*"feminazi" + 0.019*"like" + 0.016*"dont" + 0.015*"time" + 0.014*"good" + 0.013*"know" + 0.013*"idiot"
Score: 0.025001665577292442	Topic: 0.067*"dyke" + 0.032*"mongoloid" + 0.028*"negro" + 0.026*"love" + 0.025*"tweet" + 0.022*"talk" + 0.021*"raghead" + 0.020*"like" + 0.019*"faggot" + 0.017*"retard"
Score: 0.025001173838973045	Topic: 0.081*"shithol" + 0.078*"countri" + 0.076*"nigger" + 0.032*"peopl" + 0.032*"retard" + 0.024*"refuge" + 0.022*"right" + 0.021*"leav" + 0.019*"like" + 0.016*"lmao"
Score: 0.02500097267329693	Topic: 0.030*"bitch" + 0.029*"retard" + 0.025*"get" + 0.021*"call" + 0.020*"imagin" + 0.020*"nigger" + 0.020*"twat" + 0.018*"chinaman" + 0.017*"send" + 0.015*"word"
Score: 0.025000806897878647	Topic: 0.064*"spic" + 0.050*"mongi" + 0.038*"immigr" + 0.034*"go" + 0.033*"cunt" + 0.023*"retard" + 0.023*"live" + 0.019*"countri" + 0.019*"love" + 0.018*"today"
Score: 0.025000423192977905	Topic: 0.055*"mongol" + 0.030*"illeg" + 0.029*"alien" + 0.028*"cunt" + 0.028*"retard" + 0.025*"fuck" + 0.025*"like" + 0.023*"spic" + 0.022*"migrant" + 0.019*"thank"

Chapter 7

Conclusion and Future Works

On social media, conflicts and violent behaviors become more explicit with every posted hate tweet or abusive content, affecting people's lives . the research analyzed three hate speech datasets (English , Arabic and Urdu), This includes the scarcity of good open-source code that is regularly maintained and used by the society, the lack of comparative studies that evaluate the existing approaches, and the absence of resources in non-English experiments.

These sentiments were categorized into positive and negative sentiments. With machine learning, different models were designed under semi-supervised and supervised learning. Different metrics were utilized to determine the model with the best performance. Also, we concluded that semi-supervised models outperformed supervised models and the best performance under the supervised model predicted 0.77 for the best model performance and semi-supervised prediction for the same model was 0.96.

To build a general framework using deep learning models, the training dataset must have sufficient samples, in the future, the current dataset may be extended to achieve better accuracy.

The performance of the three generative models: Bert, Gradient boosting, and ML, suggest sample opportunity for improvement. We intend to make our dataset freely available to facilitate further exploration of hate speech intervention and better models for generative intervention.

We developed a Deep Learning Based Multilingual Hate Speech Detection for Resource Scarce Languages such as Arabic and Urdu. Text feature selection techniques allow to find important N-grams (consecutive words) responsible for hate speech detection .

Sentiment score and polarity reflects the emotion in a sentence and helps in hate speech detection. In future we would like to investigate language specific features for hate speech detection and other resource-scarce languages such as Persian.

Bibliography

[1] wikipedia.org, *Hate speech def*, 2022. [Online]. Available: `https://en.wikipedia.org/wiki/Hate_speech#cite_note-3`.

[2] *Hate speech in social media: An exploration of the problem and its proposed solutions*, 2014. [Online]. Available: `https://scholar.colorado.edu/downloads/n870zr009`.

[3] Gagliardone, *Countering online hate speech (pdf). paris: Unesco publishing*, 2015. [Online]. Available: `https://unesdoc.unesco.org/ark:/48223/pf0000233231`.

[4] RachelHatzipanagos, *How online hate turns into real-life violence*, 2018. [Online]. Available: `https://www.washingtonpost.com/nation/2018/11/30/how-online-hate-speech-is-fueling-real-life-violence/`.

[5] statcan, *Police-reported hate crime in canada, 2020*, 2020. [Online]. Available: `https://www150.statcan.gc.ca/n1/pub/11-627-m/11-627-m2022022-eng.htm`.

[6] Gelashvili, *Gelashvili, t., nowak, k.a.: Hate speech on social media. lund university (2018)*, 2018.

[7] N. Fortuna P., *A survey on automatic detection of hate speech in text. acm comput. surv. 51(4), 1–30 (2018)*, 2018.

[8] B. Mondal Correa, "Analyzing the targets of hate in online social media," *AAAI*, vol. 10, p. 1, 2016.

[9] M. Warmsley, "Utomated hate speech detection and the problem of offensive language," *AAAI*, vol. 10, p. 1, 2017.

[10] Hern, *Catastrophic effects of working as a facebook moderator. the guardian*, 2020. [Online]. Available: `https://www.theguardian.com/technology/2019/sep/17/revealed-catastrophic-effects-working-facebook-moderator`.

[11] T. Tetreault, *Abusive language detection in online user content. in: Proceedings of the 25th international conference on world wide web, www '16, p. 145–153. international world wide web conferences steering committee, republic and canton of geneva, che*, 2016. [Online]. Available: `https://doi.org/10.1145/2872427.2883062`.

[12] Hom, *Hom c. a puyyle about pejoratives. philos. stud. 2012;159:383–405.* 2012. [Online]. Available: `https://doi.org/10.1007/s11098-011-9749-7`.

[13] H. Lee, *R. cao, r. k.-w. lee, and t.-a. hoang, "deephate: Hate speech detection via multifaceted text representations," in 12th acm conference on web science, 2020, pp.11–20.* 2020.

[14] G. Yoshua, *L. c. yan, b. yoshua, and h. geoffrey, "deep learning," nature, vol. 521, no. 7553,pp. 436–444, 2015.* 2015.

[15] Y.Goldberg, *"a primer on neural network models for natural language processing" journal of artificial intelligence research, vol. 57, pp. 345–420*, 2016.

[16] H. Dyer Smola, *"hierarchical attention networks for document classification," in proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies, 2016, pp. 1480–1489.* 2016.

[17] W. W, *Hirschberg, j.: Detecting hate speech on the world wide web. in: Proceedings of the second workshop on language in social media, pp. 19–26*, 2012.

[18] Watanabe, *H., bouazizi, m., ohtsuki, t.: Hate speech on twitter: A pragmatic approach to collect hateful and ofensive expressions and perform hate speech detection. ieee access 6, 13825–13835*, 2018.

[19] Burnap, *P., williams, m.l.: Us and them: Identifying cyber hate on twitter across multiple protected characteristics. epj data sci*, 2016. [Online]. Available: `https://doi.org/10.1140/epjds/s13688-016-0072-6`.

[20] *Ousidhoum, n.; lin, z.; zhang, h.; song, y.; yeung, d.y. multilingual and multi-aspect hate speech analysis. in proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp); association for computational linguistics: Hong kong, china, 2019; pp. 4675–4684.*

[21] *Ibrohim, m.o., budi, i.: Multi-label hate speech and abusive language detection in indonesian twitter. in: Proceedings of the third workshop on abusive language online. pp. 46–57 (2019).*

[22] *Sanguinetti, m., poletto, f., bosco, c., patti, v., stranisci, m.: An italian twitter corpus of hate speech against immigrants. in: Proceedings of the eleventh international conference on language resources and evaluation (lrec 2018) (2018).*

[23] *Ptaszynski, m., pieciukiewicz, a., dyba la, p.: Results of the poleval 2019 shared task 6: First dataset and open shared task for automatic cyberbullying detection in polish twitter. proceedings of the poleval2019workshop p. 89 (2019).*

[24] *Basile, v., bosco, c., fersini, e., nozza, d., patti, v., pardo, f.m.r., rosso, p., sanguinetti, m.: Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. in: Proceedings of the 13th international workshop on semantic evaluation. pp. 54–63 (2019).*

[25] *Huang, x., xing, l., dernoncourt, f., paul, m.j.: Multilingual twitter corpus and baselines for evaluating demographic bias in hate speech recognition. arxiv preprint arxiv:2002.10361 (2020).*

[26] *Corazza, m., menini, s., cabrio, e., tonelli, s., villata, s.: A multilingual evaluation for online hate speech detection. acm transactions on internet technology (toit) 20(2), 1–22 (2020).*

[27] *Founta, a.m., djouvas, c., chatzakou, d., leontiadis, i., blackburn, j., stringhini, g., vakali, a., sirivianos, m., kourtellis, n.: Large scale crowdsourcing and characterization of twitter abusive behavior. in: Twelfth international aaai conference on web and social media (2018).*

[28] *Miguel ángel álvarez carmona, estefanía guzmán-falcón, manuel montes-y-gómez, hugo jair escalante, luis villasenor pineda, verónica reyes-meza, and antonio rico sulayes. 2018. authorship and aggressiveness analysis in mexican spanish tweets. in proceedings of the 3rd workshop on evaluation of human language technologies for iberian languages (ibereval 18) colocated with the 34th conference of the spanish society for natural language processing (sepln 18).* 2018.

[29] M. Graff, *Sabino miranda-jiménez, eric sadit tellez, daniela moctezuma, vladimir salgado, josé ortiz-bejar, and claudia n. sánchez. 2018. author profiling and aggressiveness analysis in twitter using utc and evomsa. in proceedings of the 3rd workshop on evaluation of human language technologies for iberian languages (ibereval'18) colocated with the 34th conference of the spanish society for natural language processing (sepln'18). 128–133.* 2018.

[30] V. Basile, *Cristina bosco, elisabetta fersini, debora nozza, viviana patti, francisco manuel rangel pardo, paolo rosso, and manuela sanguinetti. 2019. multilingual detection of hate speech against immigrants and women in twitter. in proceedings of the 13th international workshop on semantic evaluation. association for computational linguistics. 54–63.* 2019. [Online]. Available: `https://doi.org/10.18653/v1/S19-2007`.

[31] *Waseem, z.; hovy, d. hateful symbols or hateful people? predictive features for hate speech detection on twitter. in proceedings of the naacl student research workshop, san diego, ca, usa, 12–17 june 2016; pp. 88–93.*

[32] *Gambäck, b.; sikdar, u.k. using convolutional neural networks to classify hate-speech. in proceedings of the first workshop on abusive language online, vancouver, bc, canada, 4 august 2017; pp. 85–90.*

[33] *Pitsilis, g.k.; ramampiaro, h.; langseth, h. effective hate-speech detection in twitter data using recurrent neural networks. appl. intell. 2018, 48, 4730–4742.*

[34] *Watanabe, h.; bouazizi, m.; ohtsuki, t. hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. ieee access 2018, 6, 13825–13835.*

[35] *Albadi, n.; kurdi, m.; mishra, s. are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere. in proceedings of the 2018 ieee/acm international conference on advances in social networks analysis and mining (asonam), barcelona, spain, 28–31 august 2018; pp. 69–76.*

[36] *Ashi, m.; siddiqui, m.; nadeem, f. pre-trained word embeddings for arabic aspect-based sentiment analysis of airline tweets. in advances in intelligent systems and computing; springer international publishing: Cham, switzerland, 2019; pp. 241–251. [crossref].*

[37] *Ruder, s.; bingel, j.; augenstein, i.; søgaard, a. latent multi-task architecture learning. in proceedings of the aaai conference on artificial intelligence, honolulu, hi, usa, 27 january–1 february 2019; volume 33, pp. 4822–4829.*

[38] *Abu farha, i.; magdy, w. multitask learning for arabic offensive language and hate-speech detection. in proceedings of the 4th workshop on open-source arabic corpora and processing tools, with a shared task on offensive language detection; european language resource association: Marseille, france, 2020; pp. 86–90.*

[39] *Abu farha, i.; magdy, w. mazajak: An online arabic sentiment analyser. in proceedings of the fourth arabic natural language processing workshop; association for computational linguistics: Florence, italy, 2019; pp. 192–198.*

[40] *Mulki, h.; haddad, h.; ali, c.b.; alshabani, h. l-hsab: A levantine twitter dataset for hate speech and abusive language. in proceedings of the third workshop on abusive language online, florence, italy, 1 august 2019; pp. 111–118.*

[41] *Mubarak, h.; darwish, k.; magdy, w.; elsayed, t.; al-khalifa, h. overview of osact4 arabic offensive language detection shared task. in proceedings of the 4th workshop on open-source arabic corpora and processing tools, with a shared task on offensive language detection; european language resource association: Marseille, france, 2020; pp. 48–52.*

[42] *Djandji, m.; baly, f.; antoun, w.; hajj, h. multi-task learning using arabert for offensive language detection. in proceedings of the 4th workshop on open-source arabic corpora and processing tools, with a shared task on offensive language detection; european language resource association: Marseille, france, 2020; pp. 97–101.*

[43] *Hassan, s.; samih, y.; mubarak, h.; abdelali, a.; rashed, a.; chowdhury, s.a. alt submission for osact shared task on offensive language detection. in proceedings of the 4th workshop on open-source arabic corpora and processing tools, with a shared task on offensive language detection; european language resource association: Marseille, france, 2020; pp. 61–65.*

[44] *Ghosh chowdhury, a. didolkar, r. sawhney, r.r. shah arhnet - leveraging community interaction for detection of religious hate speech in arabic proceedings of the 57th annual meeting of the association for computational linguistics: Student research workshop, association for computational linguistics, florence, italy (2019), pp. 273-280, 10.18653/v1/p19-2038.*

[45] *D. truong, t. dkaki, q.-.b. truong graph methods for social network analysis, 168 (2016), pp. 276-286, 10.1007/978-3-319-46909-6*$_2$5.

[46] *Hate-speech and offensive language detection in roman urdu ,hammad rizwan, muhammad haroon shakeel, asim karim , publisher: Association for computational linguistics,pages: 2512–2522.*

[47] *Automatic detection of offensive language for urdu and roman urdu, muhammad pervez akhter , zheng jiangbin , irfan raza naqvi , mohammed abdelmajeed , and muhammad tariq sadiq https://ieeexplore.ieee.org/ielx7/6287639/8948470/09094176.pdf.*

[48] *Understanding n-grams.* [Online]. Available: `https://towardsdatascience.com/understanding-word-n-grams-and-n-gram-probability-in-natural-language-processing`.

[49] *Implementing deep learning methods.* [Online]. Available: `https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-feature-engineering-text-data-cbow.html`.

[50] *An implementation guide to word2vec using numpy and google sheets*, 2018. [Online]. Available: `https://towardsdatascience.com/an-implementation-guide-to-word2vec-using-numpy-and-google-sheets-13445eebd281`.

[51] *Implementing deep learning methods.* [Online]. Available: `https://towardsdatascience.com/nlp-101-word2vec-skip-gram-and-cbow`.

[52] *Using countvectorizer to extracting features from text, https://www.geeksforgeeks.org/using-countvectorizer-to-extracting-features-from-text2020.*

[53] *Explain what is a hashing vectorizer in nlp*, 2022. [Online]. Available: `https://www.projectpro.io/recipes/explain-what-is-hashing-vectorizer`.

[54] *Semi-supervised learning*, 2020. [Online]. Available: `https://algorithmia.com/blog/semi-supervised-learning`.

[55] *M. zampieri, p. nakov, s. rosenthal, p. atanasova, g. karadzhov, h. mubarak, l. derczynski, z. pitenis, and ç. çöltekin, "semeval2020 task 12: Multilingual offensive language identification in social media (offenseval 2020)," in proc. 14th workshop semantic eval., barcelona, spain, 2020, pp. 1425–1447. [online]. available: Https://aclanthology.org/2020.semeval-1.188.*

[56] *Hate speech and offensive language dataset,https://data.world/crowdflower/hate-speech-identification.*

[57] *Hate speech and offensive language dataset, https://github.com/hala-mulki/l-hsab-first-arabic-levantine-hatespeech-dataset , https://github.com/hkust-knowcomp/mlma$_h$ate$_s$peech.*

[58] *Abusive and threatening language detection task in urdu @fire 2021, https://www.urduthreat2021.cicling.org/.*

[59] *Agarwal a, xie b, vovsha i, rambow o, rebecca j (2011) passonneau. sentiment analysis of twitter data.*

[60] *Khan fh, bashir s, qamar u (2014) tom: Twitter opinion mining framework using hybrid classification scheme. decis support syst 57:245–257.*

[61] *Lin c, he y (2009) joint sentiment/topic model for sentiment analysis. in: Proceedings of the 18th acm conference on information and knowledge management, cikm '09, new york, ny, usa, acm, pp 375–384.*

[62] *Balahur a (2013) sentiment analysis in social media texts. in: Wassa@naacl-hlt.*

[63] V. Luhaniwal, *Feature selection using wrapper method - python implementation*, Dec. 2020. [Online]. Available: `https://www.analyticsvidhya.com/blog/2020/10/a-comprehensive-guide-to-feature-selection-using-wrapper-methods-in-python/`.

[64] *Feature selection techniques in machine learning, https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning/.*

[65] M. Angelini, G. Blasilli, S. Lenti, A. Palleschi, and G. Santucci, "Towards enhancing radviz analysis and interpretation," in *2019 IEEE Visualization Conference (VIS)*, 2019, pp. 226–230. DOI: `10.1109/VISUAL.2019.8933775`.

[66] J. Johansson and C. Forsell, "Evaluation of parallel coordinates: Overview categorization and guidelines for future research," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, pp. 1–1, Nov. 2015. DOI: `10.1109/TVCG.2015.2466992`.

[67] *Explaining recurrent neural network predictions in sentiment analysis, https://aclanthology.org/w17-5221/.*