

Social Media Hate Speech Detection Using Explainable AI

by

Harshkumar Mehta

A thesis submitted in partial fulfillment
of the requirements for the degree of
MSc Computational Sciences

The Office of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

© Harshkumar Mehta, 2022

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian University/Université Laurentienne
Office of Graduate Studies/Bureau des études supérieures

Title of Thesis Titre de la thèse	Social Media Hate Speech Detection Using Explainable AI	
Name of Candidate Nom du candidat	Mehta, Harshkumar	
Degree Diplôme	Maste of Science	
Department/Program Département/Programme	Computational Sciences	Date of Defence Date de la soutenance May 25, 2022

APPROVED/APPROUVÉ

Thesis Examiners/Examineurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Ramesh Subramanian
(Committee member/Membre du comité)

Dr. Mayuri Mehta
(External Examiner/Examineur externe)

Approved for the Office of Graduate Studies
Approuvé pour le Bureau des études supérieures
Tammy Eger, PhD
Vice-President Research (Office of Graduate Studies)
Vice-rectrice à la recherche (Bureau des études supérieures)
Laurentian University / Université Laurentienne

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

I, **Harshkumar Mehta**, hereby grant to Laurentian University and/or its agents the non-exclusive license to archive and make accessible my thesis, dissertation, or project report in whole or in part in all forms of media, now or for the duration of my copyright ownership. I retain all other ownership rights to the copyright of the thesis, dissertation or project report. I also reserve the right to use in future works (such as articles or books) all or part of this thesis, dissertation, or project report. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that this copy is being made available in this form by the authority of the copyright owner solely for the purpose of private study and research and may not be copied or reproduced except as permitted by the copyright laws without written authority from the copyright owner.

Abstract

Artificial Intelligence has invaded various fields in the present times. Be it science, education, finance, business or social media, Artificial Intelligence has found its applications everywhere. But currently, AI is limited to only its subset ‘Machine Learning’ and has not even realized its full potential. In machine learning, in contrast to traditional programming which requires writing algorithms, it is required to find the algorithm that learns patterns from a given dataset and builds a predictive model and the computer learns the patterns between input and output based on that. However, a key impediment of current AI-based systems is that they often lack transparency. The current AI systems have adopted a black box nature which allows powerful predictions, but these predictions cannot be explained directly. To gain human trust and increase transparency of AI-based systems, many researchers think that Explainable AI is the way forward. In today’s era, an enormous part of human communication takes place over digital platforms, for example, through social media platforms and so does hate speech, which are dangerous for an individual person as well as the society. These days automated hate speech detection is built on social media platforms such as Twitter, Facebook, etc. using machine learning approaches. Deep learning models attain a high performance has low transparency due to complex models, which leads to “trade-off” between performance and explainability. Explainable Artificial Intelligence (XAI) was used to create black box approaches interpretable, without giving up on performance. These XAI methods provide explanations that can be translated by humans without having a depth of knowledge in deep learning models. XAI characteristics have flexible and multifaceted potential in the hate speech detection by the deep learning models. XAI thus provides a strong interconnection between an individual moderator and hate speech detection framework, which is a pivot for the research study in interactive machine learning. In the case of Twitter, the main tweets are detected for hate

speech however retweets and replies are not detected for hate speech as there is no tool to handle the task to detect the hate speech for in progress conversations. Interpreting and explaining decisions made by complex AI models to understand the decision-making process of these model is the aim of this research. While machine learning models are being developed to detect the hate speech on social media, these models lack the interpretability and transparency on the decisions made. Traditional machine learning models achieve high performance at the cost of interpretability and explaining model decisions. The main objectives of this research are, to review and present a comparison of various techniques used in Explainable Artificial Intelligence (XAI), to present a novel approach for hate speech classification using Explainable Artificial Intelligence (XAI) and, to achieve a good trade-off between precision and recall for the method proposed. Explainable AI models for hate speech detection will help social media moderators and any other users for these models to not only see but also study and understand how the decisions are made and how the inputs are mapped to the output. As a part of this research study, two data sets were taken to demonstrate Hate Speech Detection using Explainable Artificial Intelligence (XAI). Data preprocessing was performed to remove any bias, clean data of any inconsistencies, clean the text of the tweets, tokenize, and lemmatize the text, etc. Categorical variables were also simplified in order to generate a clean dataset for training purposes. Exploratory data analysis was performed on the data sets to uncover various patterns and insights. Various pre-existent models were applied to the Google Jigsaw dataset such as Decision Trees, K-Nearest Neighbours, Multinomial Naïve Bayes, Random Forest, Logistic Regression, and Long Short-Term Memory (LSTM) out of which LSTM achieved an accuracy of 97.6%, which is an improvement compared to the studies of Risch et al. (2020). Explainable method like LIME (Local Interpretable Model-Agnostic Explanations) is applied on HateXplain dataset. Variants of BERT (Bidirectional Encoder Representations from

Transformers) model like BERT + ANN (Artificial Neural Networks) and BERT + MLP (Multilayer Perceptron) were created to achieve a good performance in terms of explainability using ERASER (Evaluating Rationales and Simple English Reasoning) benchmark by DeYoung et al. (2019) where in BERT + ANN achieved better performance in terms of explainability as compared to the study by Mathew et al. (2020).

Keywords: Explainable Artificial Intelligence, Hate Speech Detection, Offensive Languages, LIME, BERT, Neural Networks

Acknowledgements

The thesis is the outcome of the hard work and support of different people involved directly and indirectly throughout, so I cannot keep myself away from acknowledging them.

First and foremost, I would like to thank the almighty for the blessings because of which I have been able to successfully complete this thesis.

Secondly, I would like to especially acknowledge and thank my thesis supervisor, Dr. Kalpdrum Passi, for his kind supervision, expertise, comments and suggestions, assistance, and patience throughout researching and writing of this thesis. His constant support, belief and inspiration played a vital role in bringing this thesis to the edge.

I would like to thank my committee members for their generous and helpful suggestions and encouragement.

I would also like to thank my loving parents and family members, who have continuously supported and inspired me throughout my study, research, and thesis writing.

Lastly, I am much thankful to all my friends and everyone else who I might have missed to mention here in this thesis work but contributed in one or the other ways to support me during my studies.

Thank you all.

Table of Contents

<i>Abstract</i>	<i>iii</i>
<i>Acknowledgements</i>	<i>vi</i>
<i>Table of Contents</i>	<i>vii</i>
<i>List of Tables</i>	<i>x</i>
<i>List of Figures</i>	<i>xi</i>
Chapter 1	1
<i>Introduction</i>	1
1.1 Background	1
1.2 Machine Learning	2
1.3 Artificial Neural Networks	3
1.4 Deep Learning	5
1.5 Convolutional Neural Networks	6
1.6 Neural Network Frameworks	7
1.7 Trustworthiness of Artificial Intelligence	7
1.8 Need for Explainability	7
1.9 Algorithms	8
1.9.1 Naïve Bayes	8
1.9.2 Bidirectional Encoder Representations from Transformers model (BERT)	8
1.9.3 Local Interpretable Model Agnostic Explanations (LIME)	9
1.9.4 Layer-wise Relevance Propagation (LRP).....	9
1.9.5 Long Short-Term Memory (LSTM)	9
1.10 Motivation	10
1.11 Objectives	11
1.12 Proposed Methodology	11
1.13 Contributions	12
1.14 Outline of the thesis	13
Chapter 2 Literature Review	14
Chapter 3 Data and Preprocessing	23
3.1 Characteristics of the Dataset	23
3.1.1 Google Jigsaw Dataset.....	23
3.1.2 HateXplain Dataset	24
3.2 Extracting the Dataset	25
3.3 Data Preprocessing	25
3.3.1 Data Cleaning	26

3.3.2	Tokenization	27
3.3.3	Sentence Padding.....	28
3.3.4	Lemmatization	28
3.3.5	Simplification of Categorical Variables.....	29
3.4	Exploratory Data Analysis (EDA).....	29
Chapter 4		35
Feature Extraction and Classification Methods.....		35
4.1	Feature Extraction Methods.....	35
4.1.1	Count Vectorizer.....	35
4.1.2	TF-IDF.....	36
4.2	Classification Methods	37
4.2.1	BERT (Bidirectional Encoder Representation from Transformers).....	37
4.2.2	Artificial Neural Networks (ANN)	38
4.2.3	MLP (Multilayer Perceptron)	40
4.2.4	Decision Trees	40
4.2.5	KNN (K-Nearest Neighbors)	41
4.2.6	Random Forest.....	42
4.2.7	Logistic Regression.....	43
4.2.8	Naive Bayes.....	44
4.2.9	Local Interpretable Model Agnostic Explanations (LIME)	44
Chapter 5		47
Results and Discussion		47
5.1	Hate Speech and Offensive Language Detection for Google Jigsaw Dataset.....	47
5.1.1	Data set description.....	47
5.1.2	Exploratory Data Analysis.....	48
5.1.3	Data Preprocessing	51
5.1.4	Model Training and Evaluation	52
5.1.4.1	Deep Learning Model - Long Short-Term Memory (LSTM).....	52
5.1.4.2	Machine Learning Models - Decision Trees, KNN and Random Forest.....	53
5.1.4.3	Multinomial Naïve Bayes and Logistic Regression	56
5.1.5	Summary of Results for the Google Jigsaw Dataset	58
5.2	Hate Speech and Offensive Language Detection for the HateXplain data set	59
5.2.1	Dataset description.....	59
5.2.2	Exploratory Data Analysis.....	60
5.2.3	Model Training and Evaluation	63
5.2.3.1	BERT + MLP	63
5.2.3.2	BERT + ANN.....	65
5.2.3.3	LIME with other machine learning models.....	68
5.2.4	Summary of Results for the HateXplain Dataset	73
5.2.4.1	Explainability Metrics	73
5.2.4.2	Bias based Metrics	77
Chapter 6		79
Conclusions and Future Work		79
6.1	Conclusions of the study on the Google Jigsaw dataset	79
6.2	Conclusion of the study on the HateXplain dataset.....	80

6.3	Directions for future work	81
	<i>References</i>	<i>83</i>

List of Tables

Table 2.1. Literature Contributions	18
Table 3.1. Google Jigsaw Dataset Details	23
Table 3.2. Target Groups/Communities for Dataset Annotation	24
Table 3.3. HateXplain Dataset Details	25
Table 3.4. Statistical Analysis of the Data	32
Table 3.5. Dataset after Indexing	32
Table 3.6. Bias Handling.....	33
Table 5.1. Example of the Google jigsaw data set with six classes of hate speech.....	48
Table 5.2. Count of comments for each class label.....	49
Table 5.3. Count of sentences having different number of labels	50
Table 5.4. LSTM Model on the Google Jigsaw data set	52
Table 5.5. Results of classification models on the Google Jigsaw data set.....	59
Table 5.6. First five rows of the HateXplain data set.....	60
Table 5.7. Simplification of categorical variables.....	60
Table 5.8. Number of rows grouped by label.....	61
Table 5.9. Data distribution after train test split.....	62
Table 5.10. BERT + MLP Model Summary	64
Table 5.11. BERT + ANN Model Summary.....	66
Table 5.12. Accuracy of Linear Models on HatXplain Dataset	69
Table 5.13. Results of models on the HateXplain data set.....	76
Table 5.14. Explainability Metrics	76
Table 5.15. Bias Based Metrics.....	78

List of Figures

Figure 1.1. Explainable Artificial Intelligence model [1]	2
Figure 1.2. Artificial Intelligence and its relationship with machine learning and deep learning [2].....	6
Figure 1.3. Workflow for the methodology	11
Figure 3.1. Data Cleaning (ML Overview of Data Cleaning - GeeksforGeeks, 2018) [25]	26
Figure 3. 2. Tokenization (What is Tokenization Methods to Perform Tokenization, 2019) [26].....	28
Figure 3.3. Lemmatization (Kerem Kargin, 2021) [27].....	29
Figure 3.4. Exploratory Data Analysis (EDA) - Overview (Unit 1: Exploratory Data Analysis, 2021) [28]	30
Figure 4.1. Layers in Artificial Neural Networks [32].....	39
Figure 4.2. An Artificial Neuron [33]	39
Figure 4.3. MLP architecture [35].....	40
Figure 4.4. Decision Tree Structure [36]	41
Figure 4.5. K-Nearest Neighbors algorithm [37]	42
Figure 4.6. Working of Random Forest [38].....	43
Figure 4.7. Logistic Regression [39].....	44
Figure 4.8. LIME [41].....	45
Figure 5.1. Visualizing the number of comments per class label.....	49
Figure 5.2. Word Cloud for all six categories	50
Figure 5.3. Result summary of LSTM model on the Google Jigsaw data set.....	53
Figure 5.4. Result summary of DT, KNN and Random Forest classifiers on the Google Jigsaw data set	55
Figure 5.5. Plotting actual vs predicted data	56
Figure 5.6. Result summary of Multinomial Naïve Bayes and Logistic Regression models on the Google Jigsaw data set.....	57
Figure 5.7. Result summary of all Classification Models on the Google Jigsaw Dataset.....	58

Figure 5.8. Offensive Language Tweets Word Cloud	62
Figure 5.9. Hate Speech Tweets Word Cloud.....	63
Figure 5.10. Neither Category Tweets Word Cloud	63
Figure 5.11. BERT + MLP Model architecture	65
Figure 5.12. BERT + ANN model	67
Figure 5.13. Explainability with Random Forest	70
Figure 5.14. Explainability with Gaussian Naive Bayes.....	71
Figure 5.15. Explainability with Decision Tree	72
Figure 5.16. Explainability with Logistic Regression.....	73
Figure 5.17. Result summary of all Models on the HateXplain Dataset.....	75

Chapter 1

Introduction

1.1 Background

Artificial Intelligence has invaded various fields in the present times. Be it science, education, finance, business, Artificial Intelligence has found its applications everywhere. But currently, AI is limited to only its subset ‘Machine Learning’ and has not even realized its full potential. Machine Learning is the ability of computers to learn the relationship between input and output without being explicitly programmed. Thus, in machine learning, in contrast to traditional programming which requires writing algorithms, it is required to find the algorithm that learns patterns from a given dataset and builds a predictive model and the computer learns the patterns between input and output based on that. The Machine Learning model is now able to give predictions on new and unseen data. But these models do not provide an explanation as to how different features contribute to the output. So, the functioning of Artificial Intelligence is traditionally like a black box. This characteristic may not provide justifications in critical scenarios such as diagnosis of life-threatening diseases, defense etc. If there is an explanation given along with the output, combined with human reasoning, it may prove significantly useful. This forms the basis of Explainable Artificial Intelligence (XAI). XAI gives answers to many questions along with the output. It is an emerging area of research and has found applications in varied fields.

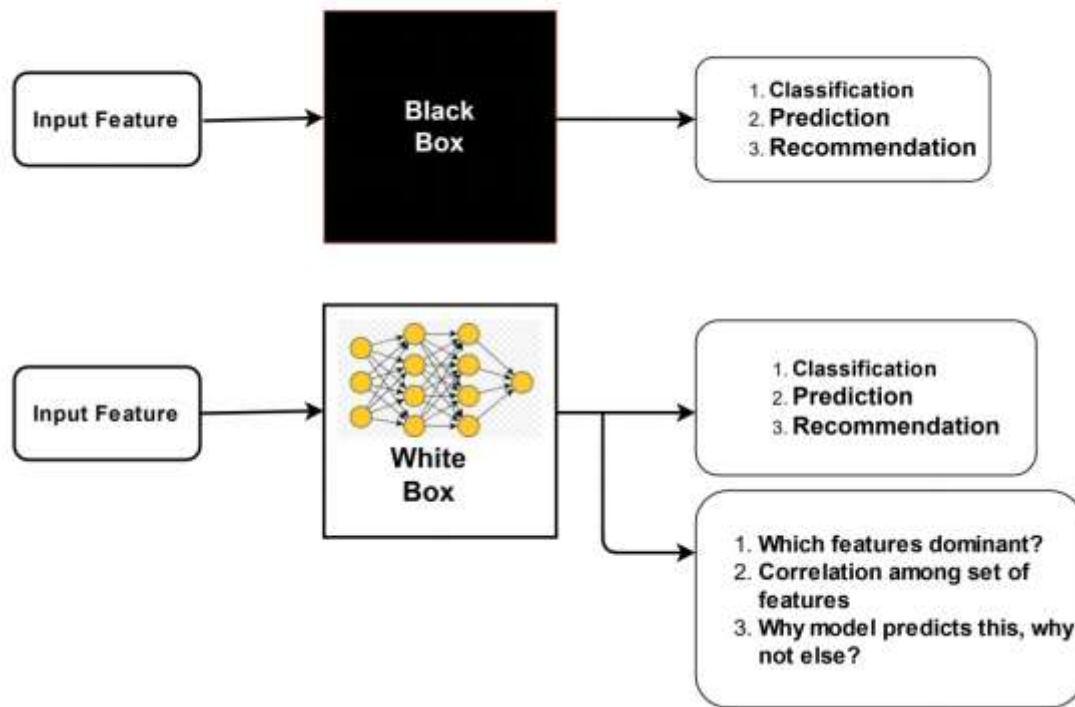


Figure 1.1. Explainable Artificial Intelligence model [1]

1.2 Machine Learning

Machine Learning is the ability of computers to learn the pattern and cause and effect relations between input and output without being fed the logic. Machine Learning recognizes the algorithm or logic from a given dataset of inputs and/or outputs and creates a model that can predict the output based on the given inputs. As it is suggestive, accuracy of prediction improves as training data improves. There are various types of machine learning, amongst which the most common classification is discussed below briefly:

- **Supervised Learning:**

Supervised machine learning is one in which the output of the training set has class labels. For example, if we are given some set of features about people and whether they are eligible for a loan or not, the eligibility is the class label or output which is not predicted.

- **Unsupervised Learning:**

Unsupervised learning is one in which the output doesn't have class labels. In unsupervised machine learning, prediction is done by finding similarities between data points and differences from other data points.

- **Semi-supervised Learning:**

This type of machine learning is a combination of supervised machine learning and unsupervised machine learning. Better predictions are formed when there is a mix of labelled and unlabeled data.

- **Reinforcement Learning:**

Reinforcement learning involves learning from feedback. In other words, the computer learns iteratively after training and predictions.

Some other types of machine learning are self-learning, transfer learning, adversarial learning, sparse dictionary learning, feature learning, etc.

1.3 Artificial Neural Networks

Artificial Neural Networks (ANN) are inspired by the neurons in the human brain. But it is different in the sense that artificial neural networks are static and symbolic while neurons of the brain are dynamic and analog. Artificial Neural Networks contain interconnected nodes which are called neurons and edges that connect these neurons. The rudimentary function of an artificial neural network is to receive a set of inputs and give an output after a set of operations to solve a given problem. ANN optimizes the weights of the neurons to provide the best prediction accuracy.

The various types of Artificial Neural Networks (ANN) are:

- **Feedforward Neural Network (FNN):**

This is the most basic neural network. Feedforward Neural network is a type of ANN that has connections between non-circular neural networks. Each connection has a weight assigned to it and provides output to one neuron and input to another. Weight denotes comparative significance.

- **Auto Encoder:**

Auto Encoder is a type of unsupervised neural network which is primarily used when there is a need to reduce the dimensionality of data. This is done by removing redundancy. Auto Encoder consists of layers of reduction and reconstruction.

- **Deep Belief Network (DBN):**

Deep Belief Network is a type of Deep Neural Network. In this type of neural network, there are multiple layers in which the first two layers correspond to associative memory and the subsequent layers receive inputs from the above layers. Thus, in DBN learning process takes place layer by layer.

- **Long Short-Term Memory Network (LSTM):**

This type of neural network consists of gates and explicitly defined memory neurons. Each neuron has three gates namely 'input', 'output', and 'forget' and a memory cell. LSTM is used for sequential data.

- **Gated Recurrent Unit (GRU):**

GRU is very similar to Long Short-Term Memory Network except for the fact that it has two gates: update gate and reset gate, instead of three gates. GRU works for small datasets and is faster and cost-effective as compared to LSTM.

- **Generative Adversarial Network (GAN):**

Generative Adversarial Network is made of two networks in which the first network creates content for the second network. The second network is fed either the training data or content from the first network. This is then evaluated and fed into the generating network. Both networks compete and learn better.

- **Recurrent Neural Network (RNN):**

In Recurrent Neural Networks, neurons receive input from the previous layers along with the information generated in the previous pass. RNN is used for autocompleting or advancing tasks.

- **Convolutional neural network (CNN):**

Convolutional Neural Networks is a type of Deep Neural Network mainly used in image analysis and classification of text-based data.

1.4 Deep Learning

Deep Learning is a type of machine learning that uses Artificial Neural Networks. Data is processed in a non-linear fashion in Deep Learning. There are multiple layers in the network of Deep Learning architecture. These are the input layer, the output layer and hidden layers between the input and the output layer. As the number of hidden layers is increased in the architecture, the accuracy of the model improves. Artificial Neural Networks (ANN) are included in Deep Learning.

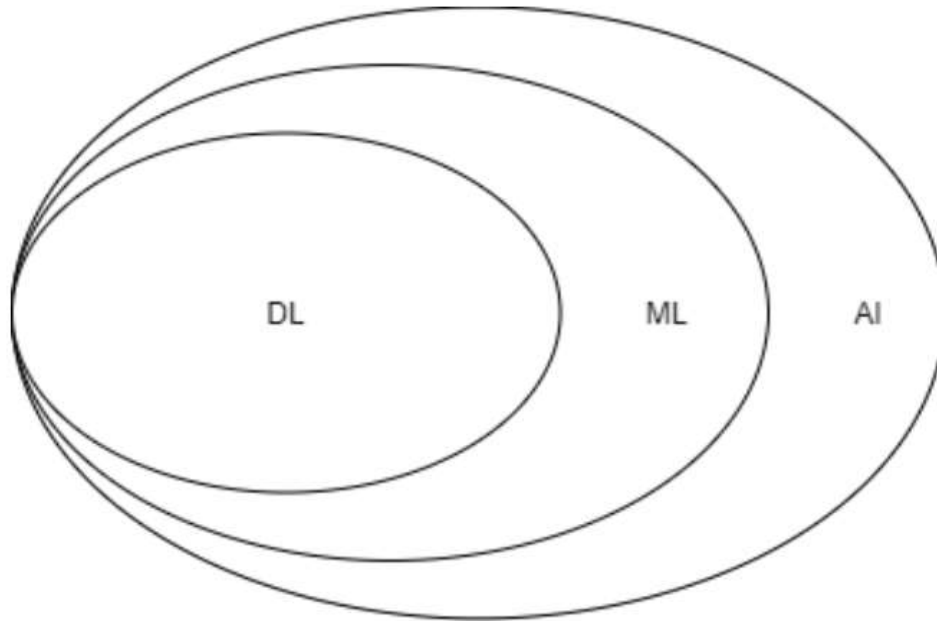


Figure 1.2. Artificial Intelligence and its relationship with machine learning and deep learning [2]

1.5 Convolutional Neural Networks

Convolutional Neural Networks (CNN) inputs raw image vectors and gives a single perspective store function as the output. The loss function is contained in the final layer of the Convolutional Neural Network. The architecture of CNN consists of three layers stacked together. These three layers are the convolutional layer, the pooling layer, and the fully connected layer. The convolutional layer is responsible for extracting low-level features from the image that is provided as input. As the number of convolutional layers is increased, it becomes possible to extract high-level features too. The Pooling layer is responsible for reducing the number of parameters. This is done to reduce the model's complexity. The fully connected layer has neurons that has full connections to the other layers of the CNN architecture.

1.6 Neural Network Frameworks

Machine Learning applications often use ANN frameworks. Models, optimizers, and metrics are implemented using ANN and then Machine Learning and Deep Learning applications are created using them. Some of the Neural Network Frameworks are:

- **TensorFlow:** TensorFlow is used for building Machine Learning and Deep Learning models.
- **Keras:** Keras is used for training, developing, and evaluating Deep Learning models. It allows the training of neural networks in very few lines of code.
- **PyTorch:** PyTorch is mainly used for applications involving Natural Language Processing and Computer Vision.

1.7 Trustworthiness of Artificial Intelligence

Artificial Intelligence is implemented as a ‘black box’ that just gives the output after a certain input but how it is done is not revealed. While it may not be necessary to get the reason behind the output in several cases, in fields such as medical research etc., the knowledge of answers to questions such as ‘how’, ‘why’, etc. becomes essential. If we do not know answers to when the model fails or succeeds, how to detect errors and correct them, etc., it may have serious implications. It may even raise questions on the efficacy of the model.

1.8 Need for Explainability

XAI is necessary if the users are to understand the AI results, trust the decisions of the algorithms, and manage the results in an organized manner. Regulatory considerations and ethics concern are important to incorporate AI in day-to-day human transactions. Explainable AI plays an important part in instilling trust in the AI regulators and business partners to make commercially beneficial and ethically viable decision making. Primarily, Artificial Intelligence is like a black box in which the input is provided to generate the output but there is no reasoning for the output. In critical

scenarios such as medical research, explainability may lead to better trust in the model. If there is an explanation of the result and how the model generates insights, this reasoning coupled with human knowledge and reasoning, may significantly improve results, and provide effective applications. This can also help prevent errors in situations in which there is no scope for them. Thus, explainable AI (or XAI) is this new dimension of Artificial Intelligence where we can seek answers to ‘why’ questions which is not possible traditionally. XAI has varied applications in healthcare, law and order, defense, etc. As it is suggestive, XAI is an emerging field of research.

1.9 Algorithms

1.9.1 Naïve Bayes

Naïve Bayes is a probabilistic based supervised machine learning model that is based on the famous Bayes rule of probability. It is called ‘Naïve’ Bayes as it works on an assumption that all features are independent of one another and contribute to the output independently. There are four types of Naïve Bayes: Optimal Naïve Bayes, Gaussian Naïve Bayes, Multinomial Naïve Bayes and Bernoulli Naïve Bayes.

1.9.2 Bidirectional Encoder Representations from Transformers model (BERT)

The bi-directional Encoder Representations from Transformers (BERT) Model is a deep learning model that was proposed in 2018 by researchers at Google Research. It has shown state-of-the-art accuracy on many Natural Language Processing and Natural Language Understanding tasks such as General Language Understanding Evaluation, Stanford Q/A dataset SQuAD v1.1 and v2.0 and Situation with Adversarial Generations.

BERT model was released in two sizes, BERT base and BERT large. The BERT base model is used for the measurement of performance of one architecture compared with another while the

BERT large model produces results reported in research papers. BERT model performs semi-supervised learning and has language processing capabilities. The BERT base model has 12 layers while the BERT large model has 24 layers in the Encoder stack. BERT model is known to have improved the F1-score of many Natural Language Processing tasks and Language modelling tasks.

1.9.3 Local Interpretable Model Agnostic Explanations (LIME)

Local Interpretable Model Agnostic Explanations (LIME) is an algorithm that can explain the results of a classifier or a regressor in a non-biased manner. The output of a LIME model is in the form of explanations where the contribution of each feature in the dataset towards the prediction of a data point is explained.

1.9.4 Layer-wise Relevance Propagation (LRP)

Layer-wise Relevance Propagation (LRP) can explain the output of a neural network. LRP propagates the output back through the network to the input layer by using weights of networks and activations of the neurons. This enables us to see which pixel has contributed to the output that is generated.

1.9.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory networks are a type of Recurrent Neural Networks that can handle long-term dependencies. Recurrent Neural Networks work on the principle of remembering previous information and utilizing it for processing the next input. However, RNNs cannot remember long term dependencies. LSTMs solve this shortcoming of RNNs. LSTMs can selectively remember or forget things.

1.10 Motivation

Artificial Intelligence is implemented as a ‘black box’ that just gives the output after a certain input but how it is done is not revealed. While it may not be necessary to get the reason behind the output in several cases, in fields such as medical research etc., the knowledge of answers to questions such as ‘how’, ‘why’, etc. becomes essential. If we do not know answers to when the model fails or succeeds, how to detect errors and correct them, etc., it may have serious implications. It may even raise questions on the efficacy of the model. Machine Learning has seen applications in various fields such as medical, research, business, education, industry, chatbots, recommendation systems and even self-driving cars. However, some Machine Learning models may not be intuitive, transparent and may be complex for people to understand. In such cases, these models may lose their effectiveness. In the past few years, Deep Learning models have also come up which present state-of-the-art results in many situations. But still, Deep Learning models are not being able to justify if they are taking the right decision or not. For instance, if a medical system is trained using Deep Learning models for the detection of cancer, it still requires the validation of doctors to verify whether the diagnosis by the model is correct or not. As the model becomes complex with an increased number of parameters, iterations, and optimization, it becomes even more difficult to validate the results by the model. If the model itself is able to explain how it functions and how all features are contributing to the output, the trust factor of the model increases, and the model becomes more reliable. This is the reason behind the need for Explainable models or Explainable Artificial Intelligence (XAI) that is the motivation behind this research.

1.11 Objectives

The objectives of this research are as follows:

1. To review and present a comparison of various techniques used in Explainable Artificial Intelligence (XAI)
2. To present a novel approach for Toxic Comments classification using Explainable Artificial Intelligence (XAI)
3. To achieve a good trade-off between precision and recall for the method proposed

1.12 Proposed Methodology

The methodology that we shall follow in this research is shown in Figure 1.2.

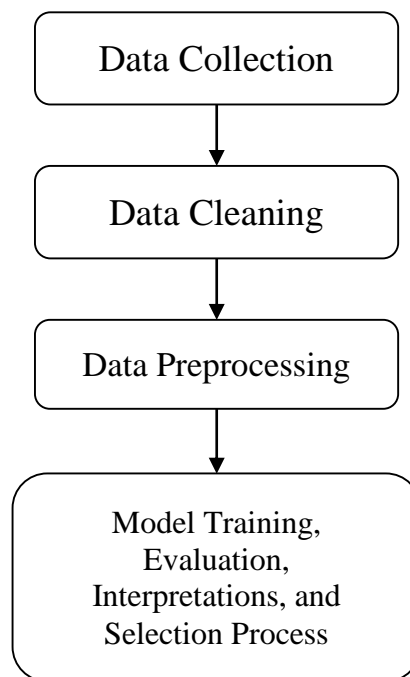


Figure 1.3. Workflow for the methodology

1. Data Collection:

We will consider two datasets for research on XAI. First, we will use a dataset containing 100,000 English tweets which are labelled abusive, hateful, and normal. Another dataset that

we will be using will consist of 9,925 unique posts in the English language. This dataset is an excel file with columns as 'posts' and 'labels'. Posts that include hate speech are labelled '1' and such posts are 1,155 in number whereas there are 8,770 posts that do not contain hate speech and are labelled '0'.

2. Data Cleaning:

After collecting the dataset, we will clean all the text contained in the dataset by removing hashtags, URLs, usernames, special characters etc. Stop words will be removed using lemmatization using the spaCy library in python.

3. Data Pre-processing:

Text data is converted into vectors using TF-IDF (Term Frequency - Inverse Document Frequency) vectorizer and pre-trained word embeddings.

4. Model Training, Evaluation, Interpretation and Selection:

We have trained Bidirectional Encoder Representations from Transformers model (BERT), Naïve Bayes, Long Short-Term Memory (LSTM), Layer-wise Relevance Propagation (LRP) along with Local Interpretable Model Agnostic Explanations (LIME). LIME explanations will be given with these models along with pre-processing and hyper-parameter tuning to get maximum accuracy, precision, and recall.

1.13 Contributions

In the area of Hate Speech Detection using Explainable Artificial Intelligence, the contributions are as follows:

1. A novel approach is presented in this thesis for hate speech classification using Explainable Artificial Intelligence (XAI). A crowd-sourced hate speech lexicon was used to collect hate speech words. Crowd sourcing was used to label a sample of words into different categories and then the classifiers were trained to predict different categories. A close and deep analysis of the text is performed and hate speech words are highlighted in the text.
2. The use of Explainable Artificial Intelligence (XAI) techniques for Hate Speech Detection on social media platforms is demonstrated.
3. An explainable model with combination of complex neural networks and BERT has been created to classify the rationalized hate speech dataset that performs very well on explainability benchmark called ERASER.

1.14 Outline of the thesis

Chapter 1 gives an introduction to the artificial intelligence and explainable artificial intelligence (EAI).

Chapter 2 reviews the literature related to explainable artificial intelligence and different applications of EAI.

Chapter 3 discusses the data sets considered for the research project and data cleaning and pre-processing applied on them.

Chapter 4 discusses the feature extraction methods used and the machine learning and deep learning methods used for implementation.

Chapter 5 discusses the results achieved in depth and Chapter 6 provides a concluding note on the research report and provides some future directions.

Chapter 2

Literature Review

Gohel et. al. [1] present an overview of Explainable AI techniques that are available and proposed in various research for multimedia processing. The paper also presents advantages and limitations of the techniques discussed in the paper. Lastly, they have also discussed and suggested some directions for future work in this area of research.

Arrieta et. al. [2] start their research by examining the existing literature and research contributions done in the research area of Explainable AI. The paper also gives a novel definition of Explainable Machine Learning. A classification and discussion of contributions to the field of Explainable Machine Learning and Deep Learning is also presented in this research. The literature review presented by the researchers encourages and motivates future work with respect to Responsible Artificial Intelligence that aims at integrity, model explainability and answerability.

Hrnjica et. al. [3] present a Predictive Maintenance (PdM) scenario in manufacturing using Explainable Artificial Intelligence (XAI). A machine learning model based on a highly efficient gradient boosting decision tree is proposed for the prediction of machine errors or any tool failures.

Singh et. al. [4] have proposed a novel explanation method based on LIME for the explanation of predictions made by a classifier. This is presented as a problem of submodular optimization. The model works for both text and image classification

Das et. al. [5] provide a comprehensive view of the landscape of the current Explainable Artificial Intelligence techniques etc. in Deep Learning. In addition, the paper supports them with mathematical summaries of the work mentioned. The researchers also present a categorization of the XAI techniques and methods based on factors such as their scope, methodology, algorithmic intuition, explanation capability. A historical timeline of various important landmarks in the study of XAI from the year 2007 to 2020 is presented. XAI algorithms and approaches are discussed. Finally, there is an evaluation of XAI algorithms generated explanation maps along with their limitations and directions for future work.

Founta et. al [6] have proposed a deep learning architecture for the detection of various types of abusive behavior online that may include hate speech, sexism, bullying, trolling, racism etc. The proposed method is tested against multiple datasets taken from Twitter. The results presented by this model are promising.

Davidson et. al. [7] have proposed a lexicon-based model for hate speech keyword identification. Crowdsourcing is used to label tweets from the dataset into hate speech, offensive language and neither of the two. A multi-classifier is trained for differentiating between these labels.

Chatzakou et. al. [8] propose a robust method for extracting text, user and network-based features and examining the characteristics of bullies. An approach for detecting bullying and aggressive behavior on Twitter has been proposed. The proposed method presents an AUC score of 0.90.

Chen et. al. [9] have proposed Lexical Syntactic Feature (LSF) architecture for the detection of offensive content on social media and identification of offensive users also. In Sentence offensive detection, the proposed framework achieves a precision of 98.24% and recall of 94.34% and a precision of 77.9% and 77.8% in the detection of offensive users.

Arras et. al. [10] provide an extension of the Layer-wise Relevance Propagation (LRP) technique to Recurrent Neural Networks (RNN). The researchers propose a propagation rule for growing connections in Recurrent Neural Networks (RNN) architectures. The proposed rule is applied to the word-based Bi-directional Long Short-Term Memory (LSTM) model for classification and presented significant results.

Montavon et. al. [11] present a discussion on the explainability of Deep Neural Network Models. The researchers also discuss the Layer-wise Relevance Propagation (LRP) technique.

Cordon et. al. [12] present a comprehensive analysis of Fuzzy systems for Explainable Artificial Intelligence (XAI). In this regard, the researchers have identified the need for fuzzy systems for Explainable Artificial Intelligence (XAI), when they were introduced, their applications, and directions for future work.

Gilpin et. al. [13] start their research with an overview of Explainability and a review of existing literature. Directions for future work are then presented based on the review of current approaches to XAI methods.

Samek et. al. [14] present an overview and theoretical basis of explainability and interpretability of Machine Learning models. The researchers also present an evaluation of interpretable Machine Learning algorithms. The best practices for the usage of these interpretable machine learning models and their applications are also discussed in this research. Finally, the challenges and some directions for future work in the field of explainable and interpretable machine learning models are discussed by the researchers.

Kanerva [15] examines the present state of Explainable Artificial Intelligence in their research. They also discuss the XAI models available out there and how they are deployed. Three Explainable Artificial Intelligence models namely LIME, Layer-wise Relevance Propagation and DeepLIFT are discussed at length in the thesis. Two proxy tasks namely pattern task and Gaussian blot task are introduced for the purpose of standardizing automated testing and evaluation of Explainable Artificial Intelligence (XAI) models. Finally, these proxy tasks and evaluation methods are applied to the three XAI models for evaluation.

Hastie et. al. [16] have discussed various terms that have been used to define Explainable Artificial Intelligence (XAI). They also present an analysis of research work done on Explainable Artificial Intelligence (XAI).

Kim et. al. [17] have laid down an approach to move towards defining interpretability of Machine Learning and Artificial Intelligence models. The researchers have also laid down the classification of approaches to evaluating the interpretability of these models. Next, the researchers have addressed some open questions and challenges in the three evaluation approaches discussed.

Hind et. al [18] propose a framework for the explanation of the results of an Artificial Intelligence System. The effectiveness and the generality of the proposed framework have also been discussed with examples.

Lundberg et. al. [19] have proposed a unified approach for the interpretation of ensemble or deep learning models. In addition to discussing various estimation methods for the proposed approach, the researchers have also presented their proofs and results.

Erion et. al. [20] have discussed three ways that can improve the explainability of tree-based Machine Learning models.

Miller [21] has presented a review on how social science research can be benefitted using Explainable Artificial Intelligence (XAI).

Nori et. Al. [22] discuss their open-source Python Package named ‘InterpretML’ that consists of explainable algorithms for Machine Learning. The package consists of two types of algorithms: glass-box and black-box models.

Table 2.1 gives a summary of the literature on explaining artificial intelligence (EAI).

Table 2.1. Literature Contributions

Ref.	Contribution	Key Findings	Limitation(s)
[1]	Discussion of various Explainable AI techniques	Need for XAI, key issues in XAI, Objectives and scope of XAI, Survey on various XAI techniques	The study emphasises on XAI and its importance but fails to discuss the limitations of conventional AI and its

		and methodologies	combination with XAI.
[2]	Explainable Artificial Intelligence (XAI): categorization, contributions, suggestions, and issues in responsible AI	Overview of Explainable Artificial Intelligence, Literature Review and taxonomy, Implications, vision, and future of XAI	Some functions are proprietary and are not exposed to public in this research. Explainable AI methods give explanations that are not aligned with what original method calculates.
[3]	Predictive Maintenance Case Study based on Explainable Artificial Intelligence (XAI)	A machine learning model based on a highly efficient gradient boosting decision tree is proposed for the prediction of machine errors or any tool failure.	Results of this research are presented using a generic dataset and not a real data, however the presented concept shows high maturity with promising results.
[4]	Explaining the predictions of any classifier	LIME model to explain the predictions of any classifier, SP-LIME model for selecting representative and non-redundant explanations	The method to perform pick up step for images is not addressed in this research.
[5]	Opportunities and Challenges in Explainable Artificial Intelligence (XAI)	Survey on seminal algorithms for explainable deep neural network algorithms, Evaluation of XAI methods and techniques	Human-attention is not able to arrive at XAI explanation maps for decision making. Quantitative measures of completeness and correctness of the explanation map are not available
[6]	A Unified Deep Learning Architecture for Abuse Detection	Deep learning architecture for detection of abuse online	Network-related metadata is not considered in the dataset due to time limitation as it takes significant amount of computation to crawl Twitter data due to Twitter API rate limits.
[7]	Automated Hate Speech Detection and the Problem of Offensive Language	Logistic Regression, Naive Bayes, Decision Trees, Random Forests, and SVM are tested using 5-fold cross-validation	The definition of hate speech in this research is limited to language that threatens or incites violence which excludes a large proportion of hate speech. Lexical

			methods used are inaccurate at identifying hate speech and only a small percentage of tweets flagged by Hate base lexicon were considered hate speech.
[8]	Detecting bullying and aggressive behavior on Twitter	Random Forest classifier using WEKA tool, 10-fold cross-validation	Results obtained with Random Forest classifier are only presented with respect to training time and performance due to limited space.
[9]	Detection of offensive content and identification of potential offensive users	Lexical Syntactical Feature (LSF) framework	Comparison of existing text-mining methods in detecting offensive contents with LSF framework in not detailed and lacks scientific validation.
[10]	Explanation of RNN predictions in Sentiment Analysis	Propagation rule for growing connections in Recurrent Neural Networks (RNN) architectures	Gradient-based Sensitivity Analysis used with approach is not able to get accurate relevance score when a sentiment is decomposed into words.
[11]	Explainability of Deep Neural Network Models	Key directions for moving towards transparency of Machine Learning models, Novel technological development for explainability	The studies in this research does not focus on exact choice of deep neural network for any particular domain instead is only focused on generalized conceptual developments.
[12]	Fuzzy systems for Explainable Artificial Intelligence	Need, timeline, applications, and future work of Fuzzy Systems for XAI	The research fails to address how to arrive at a solution to the problems that are not measurable in the evolutionary fuzzy systems (EFS) patterns.
[13]	Overview of Interpretability of Machine Learning models	Need for diverse metrics for targeted explanations, Suggestions for explainability of Deep	The study only focuses on abstract overview of explainability without diving deep into explanation

		Learning models	metrics.
[14]	Interpretable Machine Learning Models	Technical foundations of Explainable Artificial Intelligence, Presentation of practical XAI algorithms such as Occlusion, Integrated Gradients & LRP, Importance, Applications, Challenges and Directions for Future Work	The explanation revealed by model in this research are difficult to interpret by human observer due to limited accessibility of the data representation. Deeper understanding of relevance maps is not obtained by the model.
[15]	Evaluation of Explainable Artificial Intelligence Models for Convolutional Neural Networks (CNN) with proxy tasks	Proposed two 2 proxy tasks namely pattern task and Gaussian blot task which are then used to evaluate LIME, Layer-wise Relevance Propagation and Deep LIFT and results are discussed	The evaluation scheme discussed in the research has issues with cross model evaluation and is less comprehensive.
[16]	A literature survey on Explainable Artificial Intelligence (XAI) Terminology	Background, terminology, objectives of Explainable Artificial Intelligence (XAI), Natural Language Generation Approach	The survey does not explain how to evaluate Natural Language Generation (NLG).
[17]	Evaluation of Interpretability and Explainability in Machine Learning	Application-Grounded, Human-Grounded, and Functionally Grounded approaches for evaluation of interpretability, discussion of open questions related to these evaluation approaches	The research is focused only on the taxonomy to define and evaluate interpretability and not on methods to extract explanations.
[18]	Framework for the explanation of the results of an Artificial Intelligence System	Proposed framework named 'Teaching Explanations for Decisions (TED)' to provide explanations of an AI system	The proposed TED framework assumes a training dataset to be having explanation and applies Cartesian product using any machine learning algorithm to train classifier instead of

			using multitask setting.
[19]	A unified approach to explaining complex ML models	SHAP (SHapley Additive exPlanations) framework for the explanation of complex, ensemble and Deep Learning models	SHAP model is not consistent with human intuition in some cases which can lead to false positives or false negatives, a different approach is not considered in such cases.
[20]	Enhancing interpretability of tree-based machine learning models	Method for computation of the game theoretic SHapley values, local explanation method, tools for explainability using a combination of local explanation methods	Only local explanations are presented that focuses on single samples without considering global explanations.
[21]	Insights from Social Sciences related to Explainable Artificial Intelligence (XAI)	Wh-questions are diversified in Explainable AI, explanations are biased and social	Adopting the work of this research into explainable AI is not a straightforward step and the models discussed need to be refined and extended to provide good exploratory agent.
[22]	A unified framework for Machine Learning Interpretability	An open-source package InterpretML for glass-box and Blackbox explainability	Computational performance for models across datasets is not consistent for Explainable Boosting Machine (EBM) model discussed in this research.

Chapter 3

Data and Preprocessing

3.1 Characteristics of the Dataset

We have used two datasets for Hate Speech Detection using Explainable Artificial Intelligence and these datasets are discussed in this section.

3.1.1 Google Jigsaw Dataset

The first dataset that we have used is a dataset released by Google Jigsaw as part of a Kaggle Challenge. The dataset contains the following columns: Comment, toxic, severe_toxic, obscene, threat, insult, and identity_hate. The dataset comprises discussions from Wikipedia. The labels in the dataset can be multinomial i.e., a particular text can belong to two or more classes also.

Table 3.1. Google Jigsaw Dataset Details

Classification	Frequency
Clean	201,081
Toxic	21,384
Obscene	12,140
Insult	11,304
Identity Hate	2,117
Severe Toxic	1,962
Threat	689

3.1.2 HateXplain Dataset

The second dataset that we are using is the HateXplain dataset which contains posts from Twitter and Gab. Combining these two sources, we obtain a dataset that contains over 20,000 data containing hateful, offensive, and normal text as labels.

From Twitter, we take random 1% tweets from the period between January 2019 to June 2020. From Gab, we have taken the dataset provided in [24]. Reposts of the tweets have not been considered and the duplicates are removed. It is made sure that the tweets contain only textual data. However, emojis are kept as they contribute significantly to emotion detection. Also, all usernames were removed and instead a token <user> was inserted in their place.

Annotation of the Dataset:

Three different types of annotation for the posts have been considered:

- 1) Text of the posts classified as Hate Speech, Offensive or Normal
- 2) According to target groups/communities

Table 3.2. Target Groups/Communities for Dataset Annotation

Target Group/Community	Categories
Race	African, Arabs, Asians, Caucasian, Hispanic
Religion	Buddhism, Christian, Hindu, Islam, Jewish
Gender	Men, Women
Sexual Orientation	Heterosexual, LGBTQ+
Miscellaneous	Indigenous, Refugee/Immigrant, None, Others

- 3) Some parts of the posts considered as Hate Speech or Offensive

Table 3.3. HateXplain Dataset Details

	Twitter	Gab	Total
Hateful	708	5,227	5,935
Offensive	2,328	3,152	5,480
Normal	5,770	2,044	7,814
Undecided	249	670	919
Total	9,055	11,093	20,148

3.2 Extracting the Dataset

The data sets taken were in the form of CSV (Comma Separated Values) format. A CSV file stores tabular data in plain text separated by commas. Each line of a CSV file corresponds to one row of the data set, the first row of the file being the header row or the row that contains the column or attribute names. The CSV format files were loaded into a data frame using the Pandas library of Python. Pandas are used for data analysis and manipulation and are extensively used for Data Science and Machine Learning use cases.

3.3 Data Preprocessing

Pre-processing of data is a crucial step that impacts a model's performance. The data that is obtained from Twitter or online sources is noisy and can have null or missing values, outliers, images, audio, video, etc. Preprocessing ensures that the data is cleaned, free from noise and is meaningful. We have used Python's various libraries and functions for data preprocessing and cleaning for this research project.

A summary of the steps performed for preprocessing and cleaning of the dataset is given below:

1. Rows with missing labels were dropped as they do not contribute to the learning process.
2. Outliers were identified and removed lest they affect the training process.
3. Using the Natural Language Toolkit (NLTK) library, tokenization was done i.e., tokens of the sentences were created.
4. Stop words like if, then, the, and, etc. were removed to keep only the text that will contribute to the learning process.

3.3.1 Data Cleaning

Data cleaning is an essential step before training the model as it provides various benefits. Data cleaning removes any incorrect or inconsistent information that improves data quality. Outliers (extreme value due to incorrect information or the nature of the data set) are dealt with to normalize the data if it is not already. Data cleaning makes the data set error-free and makes the model training an efficient process. Figure 3.1 shows the steps in data cleaning.

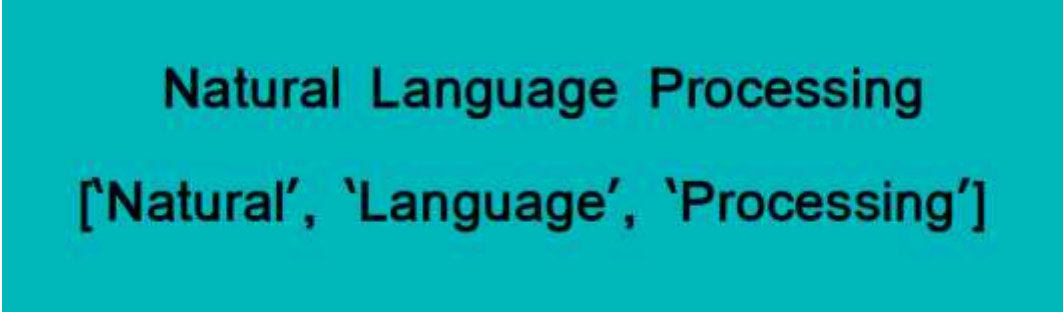


Figure 3.1. Data Cleaning (ML | Overview of Data Cleaning - GeeksforGeeks, 2018) [25]

1. Firstly, we import regular expressions module to help with data cleaning tasks. Regular expressions are sequences of characters that are used for matching with other strings in search. Patterns and strings of characters can be searched using regular expressions. Python has a “re” module that can help to find patterns and strings using regular expressions. Regular expressions can be used to remove or replace certain characters as part of data cleaning and pre-processing.
2. We remove any newline characters or additional spaces.
3. We also remove any URLs as they don’t contribute to the learning process.
4. Similarly, we also remove any other alphanumeric characters that include punctuation for the same reason. Punctuation that is removed includes the following strings:
`!"#$%&\'()*+,-./:;<=>@[\\]^_`{|}~`
Only uppercase and lowercase letters along with digits from 0 to 9 are kept.
5. Stopwords are words like “the”, “and”, “then”, “if”, etc. which are also removed as they are not a part of the learning process as such. Python’s NLTK library has stopwords of about 16 different languages. We have imported English stopwords to remove them from our dataset. These words are removed as they do not add any additional information to the learning process.
6. The outputs of these tasks are stored in a separate column and now we have word tokenized this column.

3.3.2 Tokenization

Tokenization is the process in which sentences are divided into smaller parts that are called tokens. These tokens serve as the basis for doing stemming and lemmatization and can aid in finding various patterns in the text. The Natural Language Toolkit (NLTK) library of Python provides functions to perform word tokenization. Figure 3.2 shows an example of tokenization.



Natural Language Processing

['Natural', 'Language', 'Processing']

Figure 3. 2. Tokenization (What is Tokenization | Methods to Perform Tokenization, 2019) [26]

Tokenization is the process by which a piece of text is broken into smaller units that are referred to as tokens. Specifically, word tokenization can be done either into characters or subwords. For example, the word “clearer” can be either tokenized into “clear” and “er” or “c-l-e-a-r-e-r”. In this, we have performed character tokenization that converts words into tokens that are an array of integers which makes the learning process efficient. We create a tokenizer object from a pre-trained model that is imported and then fit the tokenizer on the HateXplain dataset. This is done using the keras and TensorFlow library.

3.3.3 Sentence Padding

Padding is done so that all the input is of equal length. Neural networks require all input to be of same length. Originally, the raw text has words and sentences of different lengths. In the exploratory data analysis, we observed that mostly the maximum sentence length is 200. So, we trim sentences with lengths greater than 200 and pad the rest of the sentences.

3.3.4 Lemmatization

Using Natural Language Processing (NLP), word normalization is performed through lemmatization. In lemmatization, all words are reduced to their base/root forms as shown in Figure 3.3. For instance:

1. Go, going, gone, goes are reduced to go.

2. Read, reading, reads are reduced to read.
3. Hated, hating, hates are reduced to hate.

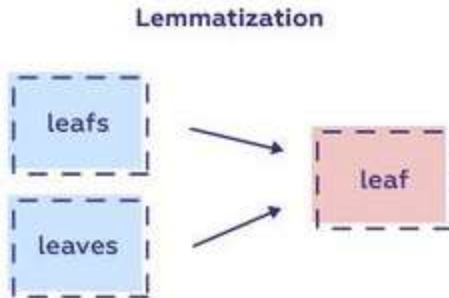


Figure 3.3. Lemmatization (Kerem Kargin, 2021) [27]

3.3.5 Simplification of Categorical Variables

The original data set has seven columns that are “unnamed”, “count”, “hate_speech”, “offensive language”, “neither”, and “tweet”. To simplify the data set for an efficient training and learning process, only three columns are kept: text, category, and label. The “tweet” column has been converted to a “text” column. The label has been derived from the class column in the original data set and the category label has values 0,1 and 2 encoded from the columns (hate_speech, offensive language, neither) in the existing dataset. In this column, 0 represents hate_speech, 1 represents offensive_language, and 2 represents neither. So, the new and final data set for the training and learning process has three columns: text, category, and label.

3.4 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is the process of investigating data and drawing out patterns and insights from it. Exploratory Data Analysis helps one understand the data better. It helps in understanding what are the various attributes in the data set, how various attributes contribute to

the target variable, identifying anomalies and outliers in the data. Exploratory data analysis also reveals any inconsistent or incomplete data. Exploratory Data Analysis serves as the basis of the Data Cleaning and Pre-processing step. EDA helps with matching assumptions and intuitions with reality. Thus, EDA is a crucial step to intelligently proceed with the following steps in the entire process of Machine Learning. Figure 3.4 rightly captures the essence of Exploratory Data Analysis.

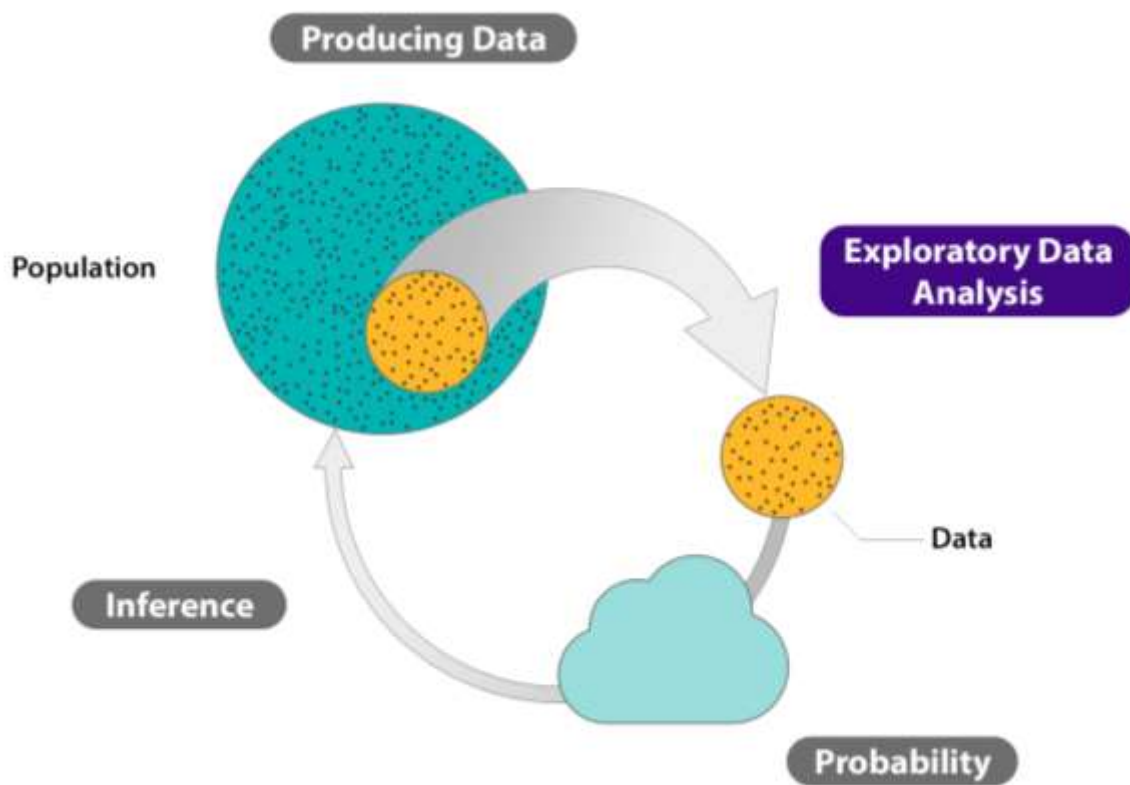


Figure 3.4. Exploratory Data Analysis (EDA) - Overview (Unit 1: Exploratory Data Analysis, 2021) [28]

Following are the steps performed as a part of Exploratory Data Analysis (EDA):

Step 1: Checking the Attributes of the Dataset

Using the `df.shape` function, we get to know the shape of the data frame i.e. the number of rows and columns in the data frame. Our data set contains 7 attributes or columns and 24,783 rows. The

df.info() function reveals certain information about the columns/attributes of the data set such as the data type of the attribute and the number of not null columns. There are no null values in our data set as the dataset has 24,783 rows and the number of not null rows in all the columns of the data set is also 24,783. All columns except for one i.e., tweet, are of the data type integer whereas the column tweet is of the data type object which is expected.

Step 2: Statistical Analysis of Data

As the next step of the Exploratory Data Analysis (EDA), statistical analysis of the data set is performed. The pandas function describe() presents a statistical summary of all the numerical attributes present in the data set. From the output of the info() function, it was observed that six of the seven attributes were numerical. So the describe() function provides a statistical summary on these six attributes which are “unnamed”, “count”, “hate_speech”, “offensive_language”, “neither” and “class”. The first aspect that the statistical summary reveals is the count of the rows i.e., not null rows of the respective columns which are 24,783 for all the columns as it was seen in the info() function as well. The other statistical measures that are provided in this summary are the mean, the standard deviation, the minimum value, the maximum value, the 25th percentile, the 50th percentile or the median and the 75th percentile. Table 3.4 shows the output of the described function applied on the data set.

Table 3.4. Statistical Analysis of the Data

	Count	Mean	Std	Min	25%	50%	75%	Max
Unnamed:0	24783.0	12681.192027	7299.553863	0.0	6372.5	12703.0	18995.5	26296.0
Count	24783.0	3.243473	0.883060	3.0	3.0	3.0	3.0	9.0
Hate Speech	24783.0	0.280515	0.631851	0.0	0.0	0.0	0.0	7.0
Offensive Language	24783.0	2.413711	1.399459	0.0	2.0	3.0	3.0	9.0
Neither	24783.0	0.549247	1.113299	0.0	0.0	0.0	0.0	9.0
Class	24783.0	1.110277	0.462089	0.0	1.0	1.0	1.0	2.0

Step 3. Indexing

Often, data sets contain several attributes and features and not all are relevant to the training process or the use case. Attributes often require some refinement to make the learning process more efficient. The data set considered for this research had labels in different columns. This data set is transformed on the attributes “class” and “tweet” and indexing is done as the index was not present in the original data set. Table 3.5 shows the head i.e., the first five rows of the transformed and indexed data set.

Table 3.5. Dataset after Indexing

	Class	Tweet
0	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	1	!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	1	!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	1	!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...

Step 4. Encoding of Categorical Variables

The labels or target variables of the data set are categorical variables that belong to either the “hate_speech” category, the “offensive_language” category or “neither” category. These labels are string values, but the Machine Learning models cannot interpret text or string values. So, these labels are encoded into numerical values. “Neither” category is labelled 2, the “offensive_language” category is labelled 1 and the “hate_speech” category is labelled 0.

Step 5: Bias Handling

Next, the data set is checked for its distribution. It is observed that the data set is highly imbalanced. Most of the tweets of the data set (about 77.43%) belonged to the “offensive_language” category. This unbalanced data set can generate bias in the training process and must be dealt with.

To deal with the bias, the data set is balanced while splitting it into training, test and validation sets. Table 3.6 shows the distribution of each category after the split.

Table 3.6. Bias Handling

Category	Label	Data Type	
Hate Speech	0	Test	143
		Train	1158
		Validate	129
Neither	2	Test	416
		Train	3372
		Validate	375
Offensive Language	1	Test	1920
		Train	15543
		Validate	1727

Step 6: Data Visualization using Word Cloud

Next, as a part of Exploratory Data Analysis (EDA), word clouds were generated to explore data using visualization. A word cloud is a data visualization technique which generates a cloud of words appearing in the text. The size of a particular word in the word cloud represents the frequency of the word in the data set. Thus, it reveals the most commonly used words in the text, in our case, in the tweets.

The tweets have been separated into different categories based on the labels: tweets containing offensive language, tweets containing hate speech and tweets containing neither of the two.

Chapter 4

Feature Extraction and Classification Methods

4.1 Feature Extraction Methods

After the data is cleaned and preprocessed, it should be converted into a form that the model can understand. For this, all variables have to be converted into numerical form. This process is called feature extraction or vectorization. This process also contributes to dimensionality reduction and hence, helps with feature extraction to keep only the features that improve the accuracy of the model. Feature Extraction can be done with a lot of methods. The importance of the words occurring in the dataset can be gauged and redundant data can be removed. New features can also be formed from existing ones. Through such methods, features that matter and new features can be generated to form a better version of the original dataset. We have used Count Vectorizer in this research which is described in the next subsection.

4.1.1 Count Vectorizer

Count Vectorizer is provided by sci-kit learn library of Python. Count Vectorizer is used for converting text into a vector [29]. This vector is in the form of token counts i.e., the frequency of the words in the text. Thus, in this way, a vocabulary of words available in the dataset is formed. This vocabulary is in the form of a matrix where the columns are the words, and the rows represent the count or the frequency of the words in the respective document. This aids in text analysis.

For instance, consider the following two sentences:

- (I) Krishna likes reading books.
- (II) The books that Radha likes reading are fictional.

The corresponding matrix would be:

	Krishna	likes	reading	books	the	that	Radha	are	fictional
Count	1	2	2	2	1	1	1	1	1

4.1.2 TF-IDF

The TF-IDF (term frequency-inverse document frequency) statistic examines the relevance of a word to a document in a collection of documents. This is accomplished by multiplying two metrics: the number of times a word appears in a document and the word's inverse document frequency over a collection of documents. It has a variety of applications, including automatic text analysis and scoring words in machine learning techniques for Natural Language Processing (NLP).

The TF-IDF format was created for document search and retrieval. It works by growing in proportion to the number of times a word appears in a document but offset by the number of documents containing the word. As a result, words like “this”, “what”, and “if”, which appear frequently in all documents, rank low since they don't mean much to that document in particular. However, if the word “Bug” appears frequently in one document but not in others, it is likely to be very relevant. If we're trying to figure out which themes some Next Sentence Prediction (NPS) replies belong to, the term “Bug”, for example, will almost certainly be associated with the topic “Reliability”, because most responses including that word will be about that topic. For each word in a document, the TF-IDF is calculated by multiplying two metrics:

The term for the number of times a word appears in a document. The simplest method for calculating this frequency is to simply count the number of times a word appears in a document. The frequency can then be adjusted based on the length of the document or the raw frequency of the most frequently used word in the document.

The word's inverse document frequency over a collection of documents. This refers to how common or uncommon a word is within the entire document set. The closer a term is to zero, the more common it is. The logarithm may be determined by taking the total number of documents, dividing it by the number of documents that contain a word, and then multiplying by the total number of documents. As a result, if the word is widely used and appears in a large number of publications, this value will be close to zero. Otherwise, it will be close to 1.

$$tfidf(t, d, D) = tf(t, d) * idf(t, D)$$

Where t denotes the terms; d denotes each document; D denotes the collection of documents.

4.2 Classification Methods

The classification methods that are used for explainability in this research are discussed below.

4.2.1 BERT (Bidirectional Encoder Representation from Transformers)

BERT model is a relatively new language model that was presented in a paper by Google in 2018 [30]. This model has presented state-of-the-art results in Natural Language Processing. The key feature of BERT is the bidirectionality of the model. BERT model makes use of the encoder component of the transformer to furnish the representation of words. BERT is used for the creation of language representation models that can serve various purposes.

BERT has undergone continual unsupervised learning and hence, continual improvement. BERT has a base layer of 'knowledge' that is derived from its pre-training. From this base layer of 'knowledge', BERT can further be trained to adapt to specifications provided. BERT uses a 'transformer' which is a part of the model responsible for providing BERT with increased capacity for understanding context and ambiguity in language. BERT's transformer processes any given word with respect to the word's relation to all other words in that particular sentence. This enables

BERT to understand the context of the word after looking at all surrounding words, unlike other models that understood the meaning of a word in one dimension only.

BERT uses the following two semi-supervised models for pre-training [31]:

- 1) **Masked Language Model (MLM):** In this task, BERT learns a featured representation for each of the words present in the vocabulary. About 85% of the words are used for training and the test are used for evaluation. The selection of the training and evaluation sets is done randomly and done in iterations. Through this process, the model learns featured representation in a bidirectional way i.e., learns both the left and right contexts of the words.
- 2) **Next Sentence Prediction (NSP):** In this task, BERT learns the relation between two different sentences. This task contributes to aspects like question answering . The model is trained to predict the next sentence.

4.2.2 Artificial Neural Networks (ANN)

ANN, also referred to as Deep Neural Networks are capable of learning high-level abstract representations in data. They can learn a hierarchy of concepts from data that range from narrow to broad concepts [32]. This helps the model to learn complex patterns from the given raw data. ANNs have a layered architecture with three layers namely, input, hidden and output layers. ANNs have a large number of hidden layers that enable them to learn deep and complex patterns from the given data, unlike the conventional machine learning algorithms. Figure 4.1 shows the architecture of the ANNs.

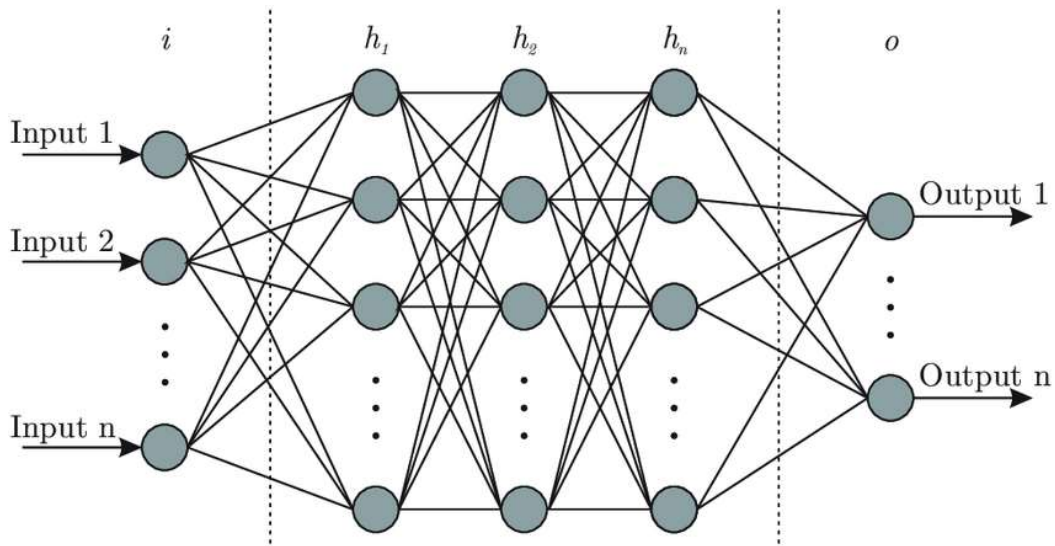


Figure 4.1. Layers in Artificial Neural Networks [32]

The basic working of an artificial neuron is similar to a biological neuron i.e., output is given on the basis of the given set of inputs. The output is optimized by the weights given to various inputs. The weights are applied using what is called the activation function. Figure 4.2 shows the structure of neurons and the use of activation function.

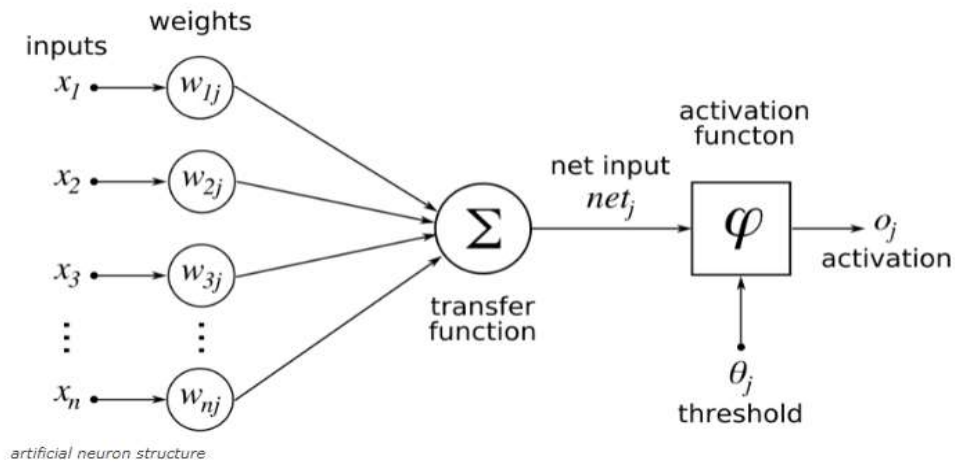


Figure 4.2. An Artificial Neuron [33]

For the explainability and interpretability, in this research, a combination of BERT and ANN has been used.

4.2.3 MLP (Multilayer Perceptron)

MLP is a type of Artificial Neural Network that is feedforward. It is composed of connected neurons which are called perceptrons. These perceptrons are composed as part of layered architecture [34]. There are three parts in the architecture of MLP namely the input layer, output layer and one or multiple hidden layers (as shown in the Figure 4.3). The hidden layer(s) aids in feature extraction and modelling.

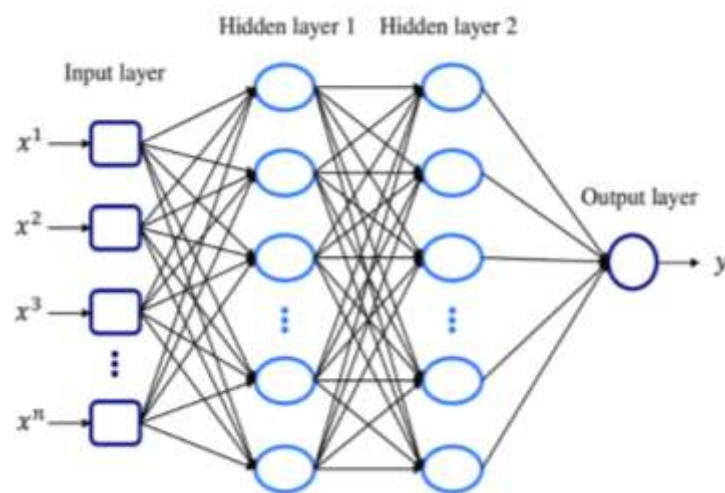


Figure 4.3. MLP architecture [35]

We have used BERT along with MLP in this research.

4.2.4 Decision Trees

As the name suggests, decision trees are tree-like structures in which each node is an attribute on the basis of which a decision is taken, and the data is further split. As shown in the Figure 4.4, a dataset is taken and it goes through a decision node, that is further split, and the dataset then goes to another decision node and so on. Each leaf node at the end of the decision nodes is a class label.

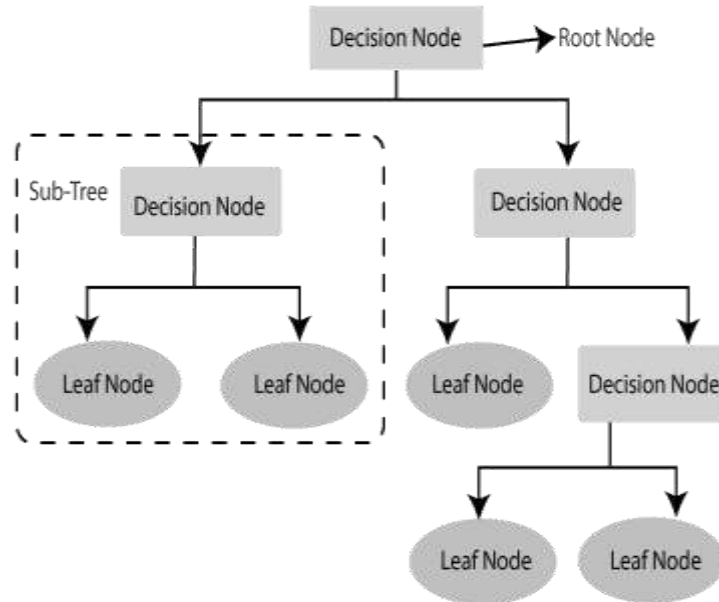


Figure 4.4. Decision Tree Structure [36]

A decision tree is a supervised learning technique that can be used for classification problems as well as regression problems. A decision tree closely mimics humans' way of thinking and decision making.

4.2.5 KNN (K-Nearest Neighbors)

KNN is a supervised machine learning algorithm in which data points are classified into categories based on their similarities to their k neighbors. It is also called a lazy learning algorithm as there is no prior learning and the algorithm just classifies the data points on the spot. Figure 4.5 depicts the working of the KNN algorithm. The green diamonds belong to category A and, the blue diamonds belong to category B. The new data point is taken and its distance from both the categories is measured. The category it is nearer to is the category that it belongs to.

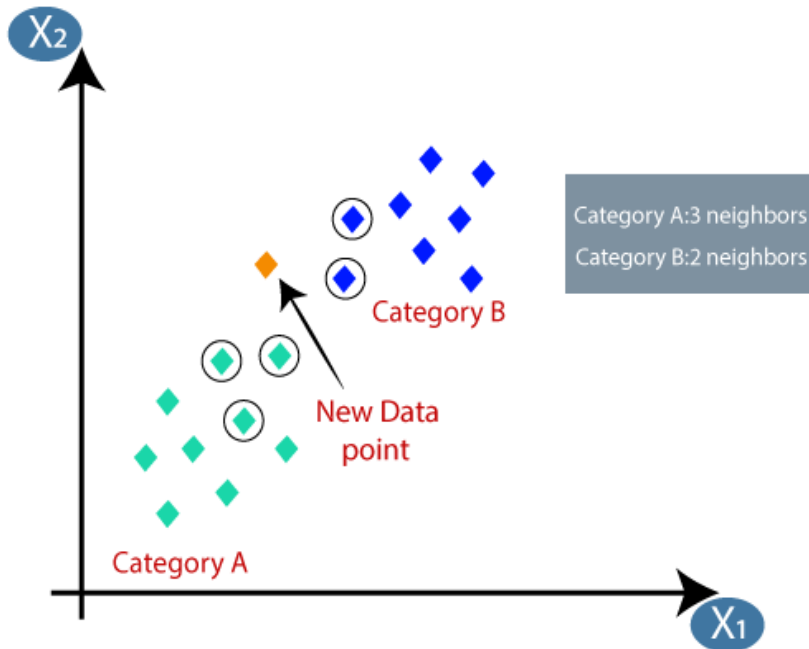


Figure 4.5. *K-Nearest Neighbors algorithm [37]*

4.2.6 Random Forest

In the simplest terms, Random Forest is a collection of decision trees. It was introduced to solve the decision trees' issue of overfitting [36]. Combining multiple trees reduces the variance and generalizes the model better. Each tree of the random forest is trained on different parts of the training dataset. Figure 4.6 depicts the working of the random forest algorithm.

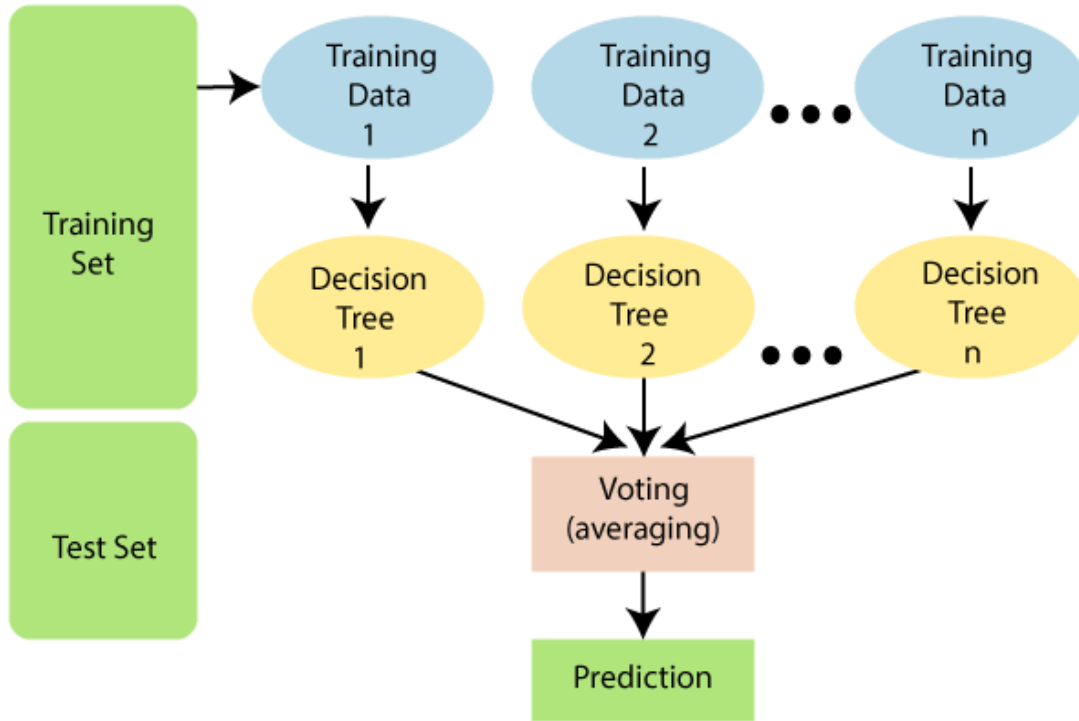


Figure 4.6. Working of Random Forest [38]

To enable explainability and interpretability using Random Forests, feature relevance, simplifying and extracting rules and techniques are used for tree ensembles.

4.2.7 Logistic Regression

Logistic Regression is a classification algorithm. The categorical dependent variable is predicted using various independent variables in Logistic Regression. Instead of a regression line as in the case of Linear Regression, an S-shaped curve is fit in Logistic Regression for the prediction of outputs. The function that is used for prediction is known as the sigmoid function. The sigmoid function gives a probabilistic value between 0 and 1 and according to this value, the output is predicted. Figure 4.7 shows a Logistic Regression curve.

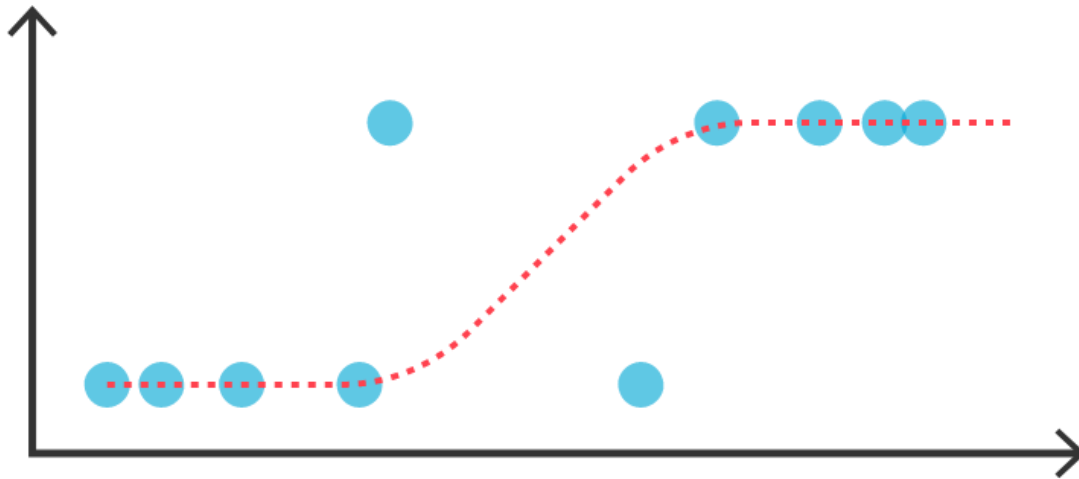


Figure 4.7. Logistic Regression [39]

4.2.8 Naive Bayes

Naive Bayes is a classification algorithm that is based on the Bayes theorem of probability.

Probabilities are calculated for all features independently (i.e., the assumption of Naive Bayes) [40]. Prediction is done based on the probabilities. Equation below gives the Naïve Bayes theorem.

$$P(B_j | A) = \frac{P(A | B_j) P(B_j)}{\sum_{i=1}^n P(A | B_i) P(B_i)}$$

Naive Bayes can be used for interpretability as Naive Bayes has an underlying assumption of independence. Thus, we can know how each feature contributes to the output.

4.2.9 Local Interpretable Model Agnostic Explanations (LIME)

LIME is an acronym for Local Interpretable Model-Agnostic Explanations. Each portion of the name represents something we want to be able to explain. Local fidelity refers to the need for the explanation to accurately reflect the classifier's behavior "around" the instance being predicted.

This explanation is pointless unless it is interpretable, that is, if it can be understood by a person. LIME is an agnostic model as it is capable of giving explanations for the predictions of a supervised learning model. LIME can be used with all types of data, be it text, images, or videos. LIME provides local interpretable explanations by computing important features and attributes for a given data point. It works by providing weights to the data rows and using feature selection techniques, it obtains the important features. LIME is especially successful in Explainable Artificial Intelligence (XAI). It can be applied to all types of data and in all domains as well.

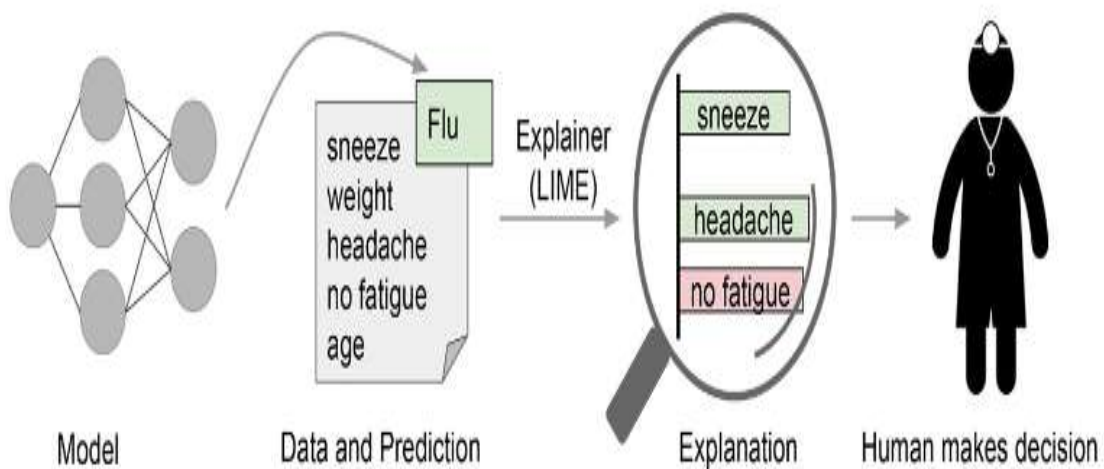


Figure 4.8. LIME [41]

First and foremost, a word on interpretability. Some classifiers employ representations that are completely unfamiliar to consumers (e.g., word embeddings). Lime describes those classifiers in terms of interpretable representations (words), even if that isn't the representation that the classifier actually uses. Furthermore, lime considers human constraints, such as the length of explanations. Model agnosticism refers to LIME's ability to provide justification for any form of supervised learning model prediction. This method can be used with any type of data, including images, text, and video. LIME can handle any supervised learning model and provide reasoning in this way.

LIME generates local optimal explanations by computing essential features in the immediate neighborhood of the instance to be explained. Lime can't peek inside the model in order to be model agnostic. We disrupt the interpretable input around its neighborhood to check how the model's predictions respond in order to figure out what sections of it are contributing to the prediction. The perturbed data points are then weighted according on their proximity to the original example, and an interpretable model is learned based on those and the related predictions. It generates 5000 samples of the feature vector by default, all of which follow normal distributions. It discovers the target variables for samples whose decisions are explained by LIME after producing normally distributed samples. It allocates weights to each of the rows based on how close they are to the original samples after getting the locally created dataset and their predictions. Then it extracts relevant features using a feature selection technique such as lasso or PCA (Principal Component Analysis). In the field of XAI, LIME has found a lot of success and support, and it's used for text, image, and tabular data. By tweaking the inputs, LIME observes the changes that happen in predictions. LIME generates a new data set having inputs with variations and their corresponding predictions generated through a black-box model. On this data set, LIME trains an explainable model with weights generated through the proximity of the instances generated. The model that is trained gets a good local approximation. Hence, the name local interpretable explanations. The explainable model trained for an instance minimizes loss and measures the proximity of the explanation to the prediction while keeping the model complexity low. LIME optimizes the loss part, and the user specifies the complexity of the model. LIME is applicable and expandable to all key machine learning fields, which is a noteworthy feature. Embeddings and vectorization of a given word or sentence can be considered a basic unit for sampling in the domain of text processing. In the case of Image, segmented chunks of the image are used as input samples.

Chapter 5

Results and Discussion

In Chapter 4, the feature extraction methods used for the two datasets were discussed, in addition to a brief discussion of the Machine Learning and Deep Learning algorithms that have been used to compare the performance enhancement in Explainable Artificial Intelligence (XAI). This chapter discusses the results from the implementation of the techniques and algorithms that have been discussed in Chapter 4.

5.1 Hate Speech and Offensive Language Detection for Google Jigsaw Dataset

5.1.1 Data set description

The first dataset that has been used in the study is taken from the Jigsaw Challenge on Kaggle. The training data set has 159,571 rows and 8 columns. The model is trained using a training-to-testing ratio of 70:30 . Five rows of the data set are shown in the Table 5.1. The columns in the data set are `comment_text`, `toxic`, `severe_toxic`, `obscene`, `threat`, `insult`, `identity_hate`. Based on whether or not the text in the given comment includes any labels from toxic, severely toxic, obscene, threatening, insulting, or identity hate, the respective column is populated with a value of 1 representing the presence of any of these features in the text. These represent the six classes. The presence of any of these labels indicate that the comment contains hate speech and is then classified accordingly.

Table 5.1. Example of the Google jigsaw data set with six classes of hate speech

Comment Text	Toxic	Severe toxic	Obscene	Threat	Insult	Identity hate
Stupid peace of shit stop deleting my stuff asshole go die and fall in a hole go to hell!	1	1	1	0	1	0
Tony Sidaway is obviously a fistfuckee. He loves an arm up his ass.	1	0	1	0	1	0
All of my edits are good. Cunts like you who revert good edits because you're too stupid to understand how to write well , and then revert other edits just because you've decided to bear a playground grudge, are the problem. Maybe one day you'll realise the damage you did to a noble project.	1	0	1	0	1	0
Atheism is full of bias shit	1	0	0	0	0	0
You sir are an imbecile, and a pervert.	1	0	0	0	1	0

5.1.2 Exploratory Data Analysis

Using the plotly library of python, the number of comments per label can be visualized in the form of a bar plot as shown in Figure 5.1. It can be seen in the following bar plot that a large number of comments are labelled as toxic.

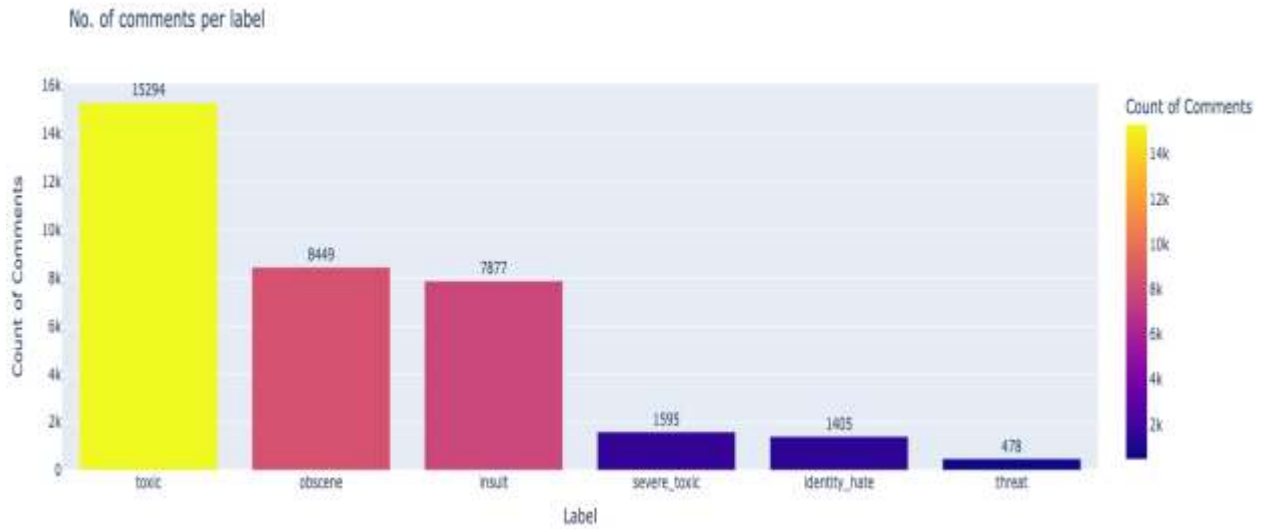


Figure 5.1. Visualizing the number of comments per class label

Extracting the information from the above bar graph in tabular form as shown in Table 5.2. It can be seen that a large number i.e., 15,294 comments are labelled as toxic, 8,449 are labelled as obscene, 7,877 comments are labelled as insult, 1,595 comments are labelled as severe toxic, and 1,405 comments are labelled as identity hate and 478 comments are labelled as Threat.

Table 5.2. Count of comments for each class label

Label	Count of comments
Toxic	15294
Obscene	8449
Insult	7877
Severe toxic	1595
Identity hate	1405
Threat	478

Next, the number of sentences containing different number of labels in a sentence are shown in Table 5.3. It can be observed that 143,346 sentences do not have any labels, 6,360 sentences have

5.1.4 Model Training and Evaluation

5.1.4.1 Deep Learning Model - Long Short-Term Memory (LSTM)

LSTM is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can process not only single data points, but also entire sequences of data.

The input layer of LSTM is designed with 30000 x 128 size (or 3840000 parameters) in order to incorporate the whole dataset comments as shown in Table 5.4. After lemmatizing, tokenizing, removing stop words and punctuation marks, top 30,000 words are taken for the processing. Alphabets and numbers can be represented uniquely using 7-bit ASCII code, and $27 = 128$. This layer will input the tokenized words and fetch 3,840,000 entities from it.

The function of dropout layers is to reduce the number of entities read, but to increase the number of features to be extracted from the input. The standard rate of dropout in LSTM is 0.2 (learning rate). The number of parameters 131,584 shows that after the recurrence layers, number of entities are reduced from 3,840,000 to 131584. The dense layer outputs 774 units (which roughly equals to 128×6) in order to tell which input word belongs to which class.

Table 5.4. LSTM Model on the Google Jigsaw data set

Layer Type	Output Shape	Param #
Embedding	(None, None,128)	3840000
LSTM 1	(None, None,128)	131584
LSTM 2	(None,128)	131584
Dense	(None,6)	774

The data is divided into 70-30 split where 70% of the data is utilized for training and 30% is utilized for testing purposes. After that the model defined above is compiled with the loss function

as binary cross-entropy and the Adam optimizer. Then, the model is fit on the training data with batch size of 128.

The accuracy obtained by the LSTM model is 97.6%, the precision is 0.85, the Recall is 0.83, the F1 score is 0.84, and the Specificity is 0.82. The result summary of LSTM model is shown in Figure 5.3.

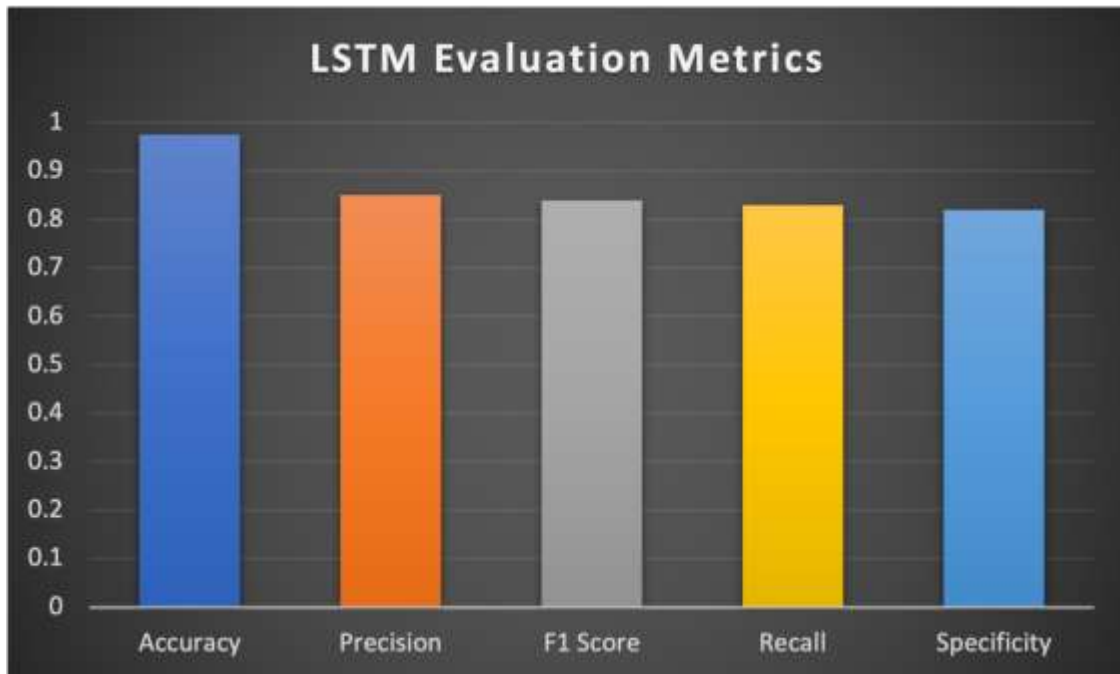


Figure 5.3. Result summary of LSTM model on the Google Jigsaw data set

5.1.4.2 Machine Learning Models - Decision Trees, KNN and Random Forest

In this section three machine learning models, Decision Tree, KNN and Random Forest are used for training and testing.

Decision Trees naturally represent the way we make decisions. Think of a machine learning model as a decision-making engine that takes a decision on any given input object (data point). The main strength of decision trees lies in classification and prediction tasks. A decision tree has a structure of a tree in which the non-leaf nodes represent test on some attribute, branch nodes specify the output for that test, while the terminal nodes represent the true class label. The process of learning

a decision tree involves splitting of the source sets into smaller subsets. This is done on the basis of an attribute value test. This event is recursively repeated and is also termed as recursive partitioning. The recursion process is terminated when further splitting of the nodes do not contribute into any additional predictions.

KNN is one of the linear algorithms that comes under the supervised machine learning algorithms. It can be used for solving classification as well as regression problems. Following are the basic steps of the algorithm.

1. Loading the data from the data set and initializing a variable named K.
2. The following two steps are repeated for each row of the data set
 - The distance between the questioned example and the current example of the data is calculated.
 - The calculated distance and an index is added to the example to form an ordered collection.
3. The ordered collection is sorted according to the distances and the index.
4. The initial K entries are picked from the sorted list, and are labelled in case of regression, or return the value of K in case of classification.

Random Forest is an ensemble collection of decision trees. An ensemble means a group of things viewed as a whole rather than individually. In ensembles, a collection of models is used to make predictions, rather than individual models. The most popular in the family of ensemble models is the random forest: an ensemble made by the combination of a large number of decision trees. Random forest are created using a special ensemble method called bagging. Bagging stands for Bootstrap Aggregation. Bootstrapping means creating bootstrap samples from a given data set. A bootstrap sample is created by sampling the given data set uniformly and with replacement. A

bootstrap sample typically contains about 30- 70% data from the data set. We have used Random Forest Classifier from sklearn.ensemble library.

As shown in Figure 5.4 the accuracy obtained by Decision Tree classifier is 89% whereas KNN classifier has 90% accuracy and Random Forest classifier has 91.2% accuracy. In terms of other measures such as precision, recall, F1 score, and specificity, it can be observed that Random Forest performs better than the other two classifiers. It can be seen from Figure 5.5 that all three linear classifiers perform very efficiently.

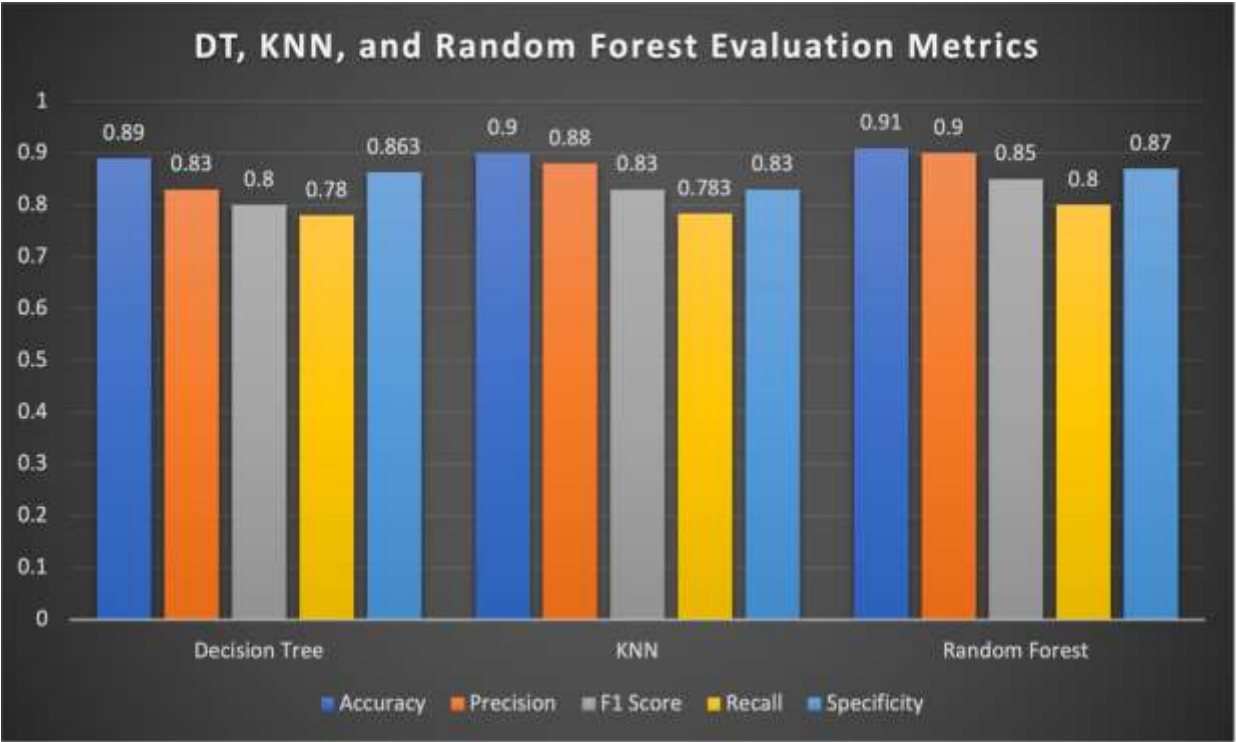


Figure 5.4. Result summary of DT, KNN and Random Forest classifiers on the Google Jigsaw data set

Figure 5.5 shows the actual values versus the predicted values for the best performing model, which is Random Forest model in this case in a bar chart. This indicates that the predicted values are closer to the actual values.

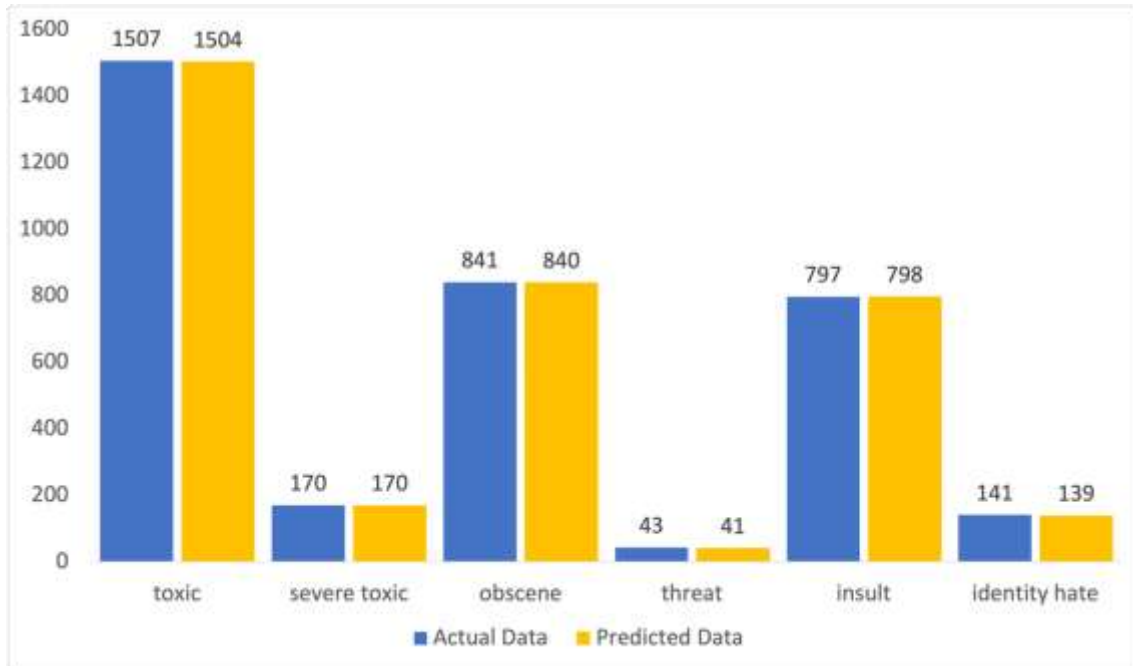


Figure 5.5. Plotting actual vs predicted data

5.1.4.3 Multinomial Naïve Bayes and Logistic Regression

This section demonstrates the implementation of the Multinomial Naïve Bayes and Logistic Regression. Multinomial Naïve Bayes is based on the popular mathematics theorem called Bayes theorem. Bayes theorem works on probabilities and is very popular in natural language processing. The main aim of this machine learning technique is to guess the tag of a piece of text using the Bayes theorem. The likelihood of the tag is calculated depending upon conditional probability. For analyzing the text inputs and the texts having large number of classes, Naïve Bayes method has proven to be a very strong method among the existing linear machine learning methods. When predictor B itself is available, we calculate the likelihood of class A. It is based on the formula below.

$$P(A|B) = P(A) * \frac{P(B|A)}{P(B)}$$

Logistic Regression predicts binary outcomes such as zero or one, yes or no, etc. using the statistical analysis and the previous observation of the instance. Logistic regression predicts the variable called as dependent variable using a logistic regression equation. In this particular problem, the logistic regression algorithm takes into consideration individual piece of text and returns the values true or false for each class that the piece of text can belong to.

A Naïve Bayes pipeline and Logistic Regression pipeline is created by supplying the English stop words from the NLTK library to the TF-IDF Vectorizer. As shown in Figure 5.6 the accuracy obtained by Multinomial Naïve Bayes classifier is 96% whereas Logistic Regression classifier has 97% accuracy, which are both very significant improvements as compared to previous tree classifiers. In terms of other measures such as precision, recall, F1 score, and specificity, it can be observed that Multinomial Naïve Bayes classifier performs better than the Logistic Regression classifier.

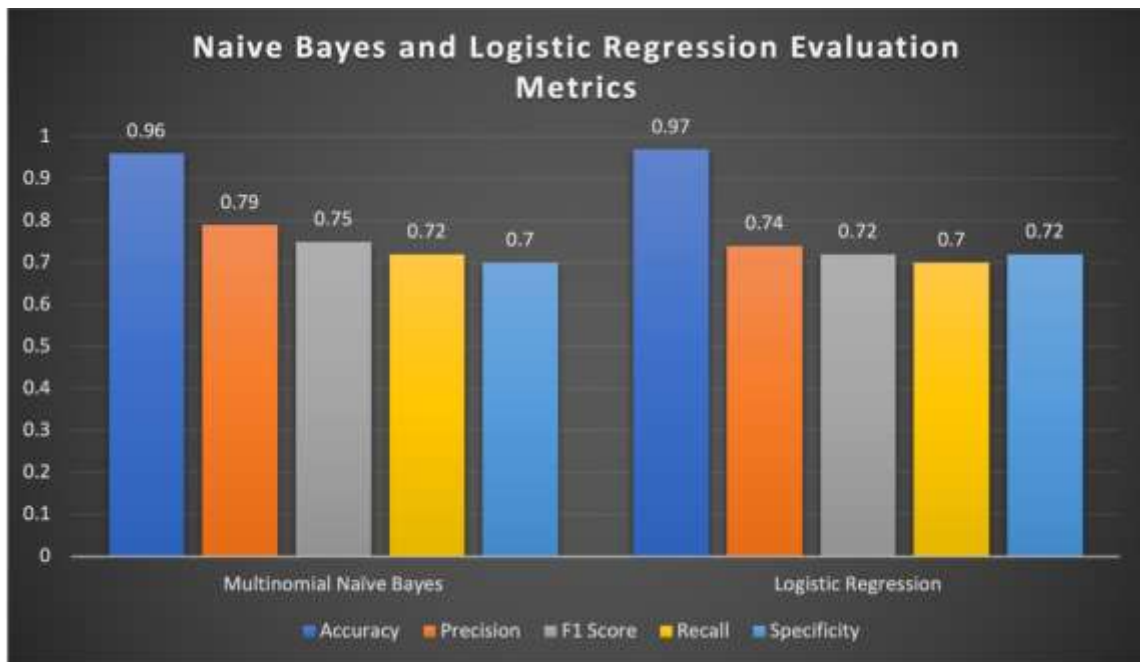


Figure 5.6. Result summary of Multinomial Naïve Bayes and Logistic Regression models on the Google Jigsaw data set

5.1.5 Summary of Results for the Google Jigsaw Dataset

The results of all the models on the Google Jigsaw dataset, evaluated in terms of their accuracy, precision and F1 score are shown in Figure 5.7. Table 5.5 gives the scores of the evaluation metrics. It can be observed that LSTM is the best performing model with an accuracy of 97.6%, closely followed by Multinomial Naive Bayes with an accuracy of 96% and Logistic Regression with an accuracy of 97%. Random Forest shows the highest precision of 90% and KNN classifier with a precision of 88%.

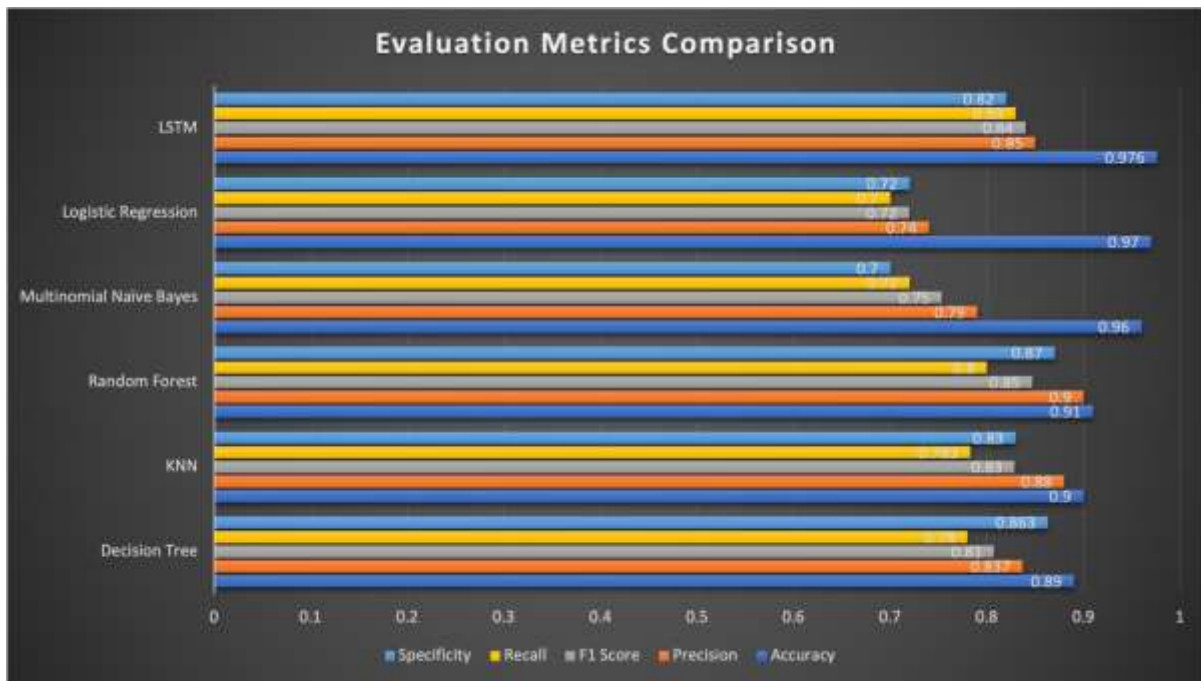


Figure 5.7. Result summary of all Classification Models on the Google Jigsaw Dataset

Table 5.5. Results of classification models on the Google Jigsaw data set

Classifier Name	Accuracy	Precision	F1-Score	Sensitivity/ Recall	Specificity
Decision Tree	0.89	0.837	0.81	0.78	0.863
K-Nearest Neighbours	0.90	0.88	0.83	0.783	0.83
Random Forest	0.91	0.90	0.85	0.80	0.87
Multinomial Naïve Bayes	0.96	0.79	0.75	0.72	0.70
Logistic Regression	0.97	0.74	0.72	0.70	0.72
Long Short-Term Memory (LSTM)	0.976	0.85	0.84	0.83	0.82

5.2 Hate Speech and Offensive Language Detection for the HateXplain data set

5.2.1 Dataset description

The HateXplain data set consists of tweets that contain either hate speech, offensive language or neither. This dataset is having a complex structure with tweet text classified into three categories by human annotators from different communities. Dataset has been imported and assigned to a pandas data frame. Printing the head (i.e., the first five rows) of the data frame shows the rows in the dataset in Table 5.6.

Table 5.6. First five rows of the HateXplain data set

	Count	Hate Speech	Offensive Language	Neither	Class	Tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. & as a man you should always take the trash out...
1	3	0	3	0	1	!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat hoe in the 1st place!!
2	3	0	3	0	1	!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fuck a bitch and she start to cry? You be confused as shit
3	3	0	2	1	1	!!!!!!! RT @C_G_Anderson: @viva_based she look like a tranny
4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you hear about me might be true or it might be faker than the bitch who told it to ya 

5.2.2 Exploratory Data Analysis

Next, as a part of data cleaning and pre-processing, the column tweet has been renamed as text and the column class has been renamed as category. Additionally, the category column has values 0, 1 and 2 that are mapped from hate_speech, offensive_language and neither column, respectively. Finally, the original class column along with the newly created text and category columns are concatenated to form a new data frame with three columns as shown in Table 5.7.

Table 5.7. Simplification of categorical variables

	Text	Category	Label
0	!!! RT @mayasolovely: As a woman you shouldn't...	Neither	2
1	!!!! RT @mleew17: boy dats cold...tyga dwn ba...	Offensive Language	1
2	!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...	Offensive Language	1
3	!!!!!!! RT @C_G_Anderson: @viva_based she lo...	Offensive Language	1

4	!!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...	Offensive Language	1
...
24778	you's a muthaf***in lie “@LifeAsKing: @2...	Offensive Language	1
24779	you've gone and broke the wrong heart baby, an...	Neither	2
24780	young buck wanna eat!!.. dat nigguh like I ain...	Offensive Language	1
24781	youu got wild bitches tellin you lies	Offensive Language	1
24782	~~Ruffled Ntac Eileen Dahlia - Beautiful col...	Neither	2

Grouping the dataset by label shows that the data set is highly imbalanced. We use the pandas function, groupby to obtain the results. The results contained by groupby is used to group the dataframe into three sections, one containing rows belonging to hate speech, another to offensive and the last to neither.

Table 5.8 shows the distribution that is obtained after grouping the dataset by the column label. Label 0 or Hate Speech has 1430 rows, Label 1 or Offensive Language has 19,190 rows and label 2 or neither has 4,163 rows. It is observed that most of the tweets of the given data set contain offensive language.

Table 5.8. Number of rows grouped by label

Label	Class	Text
0	Hate Speech	1430
1	Offensive	19190
2	Neither	4163

The percentage share of each label in the entire data set consists of 77.43% of the tweets having offensive language while only 5.77% containing hate speech. The entire data set is then split into

three parts i.e., 80% training, 10% validation, and 10% test datasets. Training data set is used to train the models. The validation set is used for hyperparameter tuning in order to avoid overfitting. Finally, the test data set is used for measuring the performance of the trained model on unlabeled data. Using scikit learn model selection, Train-Test split is performed in order to obtain the training, validation and test sets. Table 5.9 shows the distribution of the data after the data set is split into train, validation and test sets.

Table 5.9. Data distribution after train test split

Label	Class	Dataset Type	No of Rows
0	Hate Speech	Train (80%)	1158
		Validation (10%)	129
		Test (10%)	143
1	Offensive Language	Train (80%)	15543
		Validation (10%)	1727
		Test (10%)	1920
2	Neither	Train (80%)	3372
		Validation (10%)	375
		Test (10%)	416

Amongst the tweets that belong to the ‘Offensive Language’ category, Figure 5.8 shows the most popular words used in the tweets containing offensive language in the word cloud. Stop words are ignored as they don’t contribute anything significant to the analysis.



Figure 5.8. Offensive Language Tweets Word Cloud

Amongst the tweets that belong to the ‘hate speech’ category, Figure 5.9 shows the most popular words used in the tweets containing hate speech in the word cloud.



Figure 5.9. Hate Speech Tweets Word Cloud

Amongst the tweets that belong to the ‘neither’ category, Figure 5.10 shows the most popular words used in the tweets that do not contain any offensive language or hate speech in the word cloud.



Figure 5.10. Neither Category Tweets Word Cloud

5.2.3 Model Training and Evaluation

5.2.3.1 BERT + MLP

This section provides a discussion on the training of dataset using the BERT model used with other techniques to provide explainability. It is a machine learning framework for NLP tasks specially designed to help computational systems for understanding the complex structure of language in the given text by using the surrounding text to establish some meaning. From the TensorFlow hub, a BERT model (TensorFlow Hub, 2021) and a preprocessor model are selected. Unbalanced data is dealt with using weight optimization and bias is set. For doing weight

optimization, appropriate weights are calculated for each class, depending upon their proportion. These weight factors are then multiplied to individual class so that the bias between classes can be removed.

Next, BERT is trained with MLP model. Table 5.10 depicts the model summary for the BERT + MLP model where the first column indicates the type of the layer, second and third column indicates the output shape and number of parameters generated by processing of the layer, respectively, and the last column represents the previous layer it is connected to. There are a total 29,027,843 trainable parameters.

Table 5.10. BERT + MLP Model Summary

Layer (type)	Output Shape	Param #	Connected to
text (InputLayer)	[(None,)]	0	[]
preprocessing (KerasLayer)	{'input_word_ids':(None,128), 'input_mask': (None, 128), 'input_type_ids': (None, 128)}	0	['text [0][0]']
BERT_encoder (KerasLayer)	{'pooled_output': (28763649, None, 512), 'encoder_outputs': [(None, 128, 512), (None, 128, 512), (None, 128, 512), (None, 128, 512)], 'default': (None, 512), 'sequence_output': (None, 128, 512)}	28763649	['preprocessing [0][0]', 'preprocessing [0][1]', 'preprocessing [0][2].']
dense (Dense)	(None, 512)	262656	['BERT_encoder [0][5]']
dropout (Dropout)	(None, 512)	0	['dense [0][0]']
classifier (Dense)	(None, 3)	1539	['dropout [0][0]']
Total params: 29,027,844 Trainable params: 29,027,843 Non-trainable params: 1			

There are a total 29,027,844 parameters. Out of them 29,027,843 are trainable parameters and only 1 is the non-trainable parameter.

As shown in Figure 5.11, architecture of the BERT + MLP model is fine tuned in order to achieve the most efficient performance. The model contains one input and preprocessing layer along with the BERT encoder, which is a keras layer. Dense layer used after keras layer is used to reduce the parameters and increase the number of features being propagated to the next layer. The dropout

later is added, to avoid overfitting of the model followed by the one dense layer which is used to represent the results as a classification problem. After that the model defined above is compiled with the loss function as sparse categorical cross-entropy and the Adam optimizer.

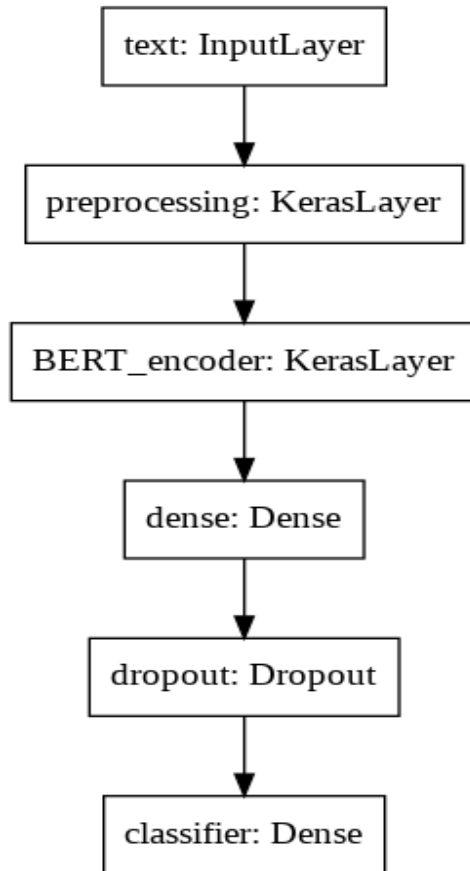


Figure 5.11. BERT + MLP Model architecture

5.2.3.2 BERT + ANN

Next, BERT is used with ANN to train the model and evaluate the performance. Table 5.11 depicts the model summary for the BERT + ANN model where the first column indicates the type of the layer, second and third column indicates the output shape and the number of parameters generated by processing of the layer, respectively, and the last column represents the previous layer it is connected to.

Table 5.11. BERT + ANN Model Summary

Layer (type)	Output Shape	Param #	Connected to
text (InputLayer)	[(None,)]	0	[]
preprocessing (KerasLayer)	{'input_word_ids':(None,128), 'input_mask': (None, 128), 'input_type_ids': (None, 128)}	0	['text [0][0]']
BERT_encoder (KerasLayer)	{'pooled_output': (28763649, None, 512), sequence_outputs': (None, 128, 512), 'encoder_outputs': [(None, 128, 512), (None, 128, 512), (None, 128, 512), (None, 128, 512)], 'default': (None, 512), 'sequence_output': (None, 128, 512)}	28763649	['preprocessing [0][0]', 'preprocessing [0][1]', 'preprocessing [0][2]',]
Conv1d (Conv1D)	(None, 127, 32)	32800	['BERT_encoder[0][6]']
Conv1d_1 (Conv1D)	(None, 126, 64)	4160	['conv1d[0][0]']
Global_max_pooling1d (GlobalMaxpooling1D)	(None, 64)	0	['conv1d_1[0][0]']
dense_1 (Dense)	(None, 512)	33280	['BERT_encoder [0][5]']
dropout_1 (Dropout)	(None, 512)	0	['dense_1 [0][0]']
classifier (Dense)	(None, 3)	1539	['dropout_1 [0][0]']
Total params: 28,835,428 Trainable params: 28,835,427 Non-trainable params: 1			

As shown in Figure 5.12, architecture of the BERT + ANN model is fine-tuned in order to achieve the most efficient performance. The model contains one input and preprocessing layer along with the BERT encoder, which is a keras layer. The BERT is combined, with convolution layers followed by 1D Global max-pooling layer, which computes the maximum of all the input sizes for each of the input channels. Dense layer used after 1D Global max-pooling layer is used to reduce the parameters and increase the number of features being propagated to the next layer. In the end, the dropout later is added to avoid overfitting followed by the dense layer. After that the model defined above is compiled with the loss function as sparse categorical cross-entropy and the Adam optimizer.

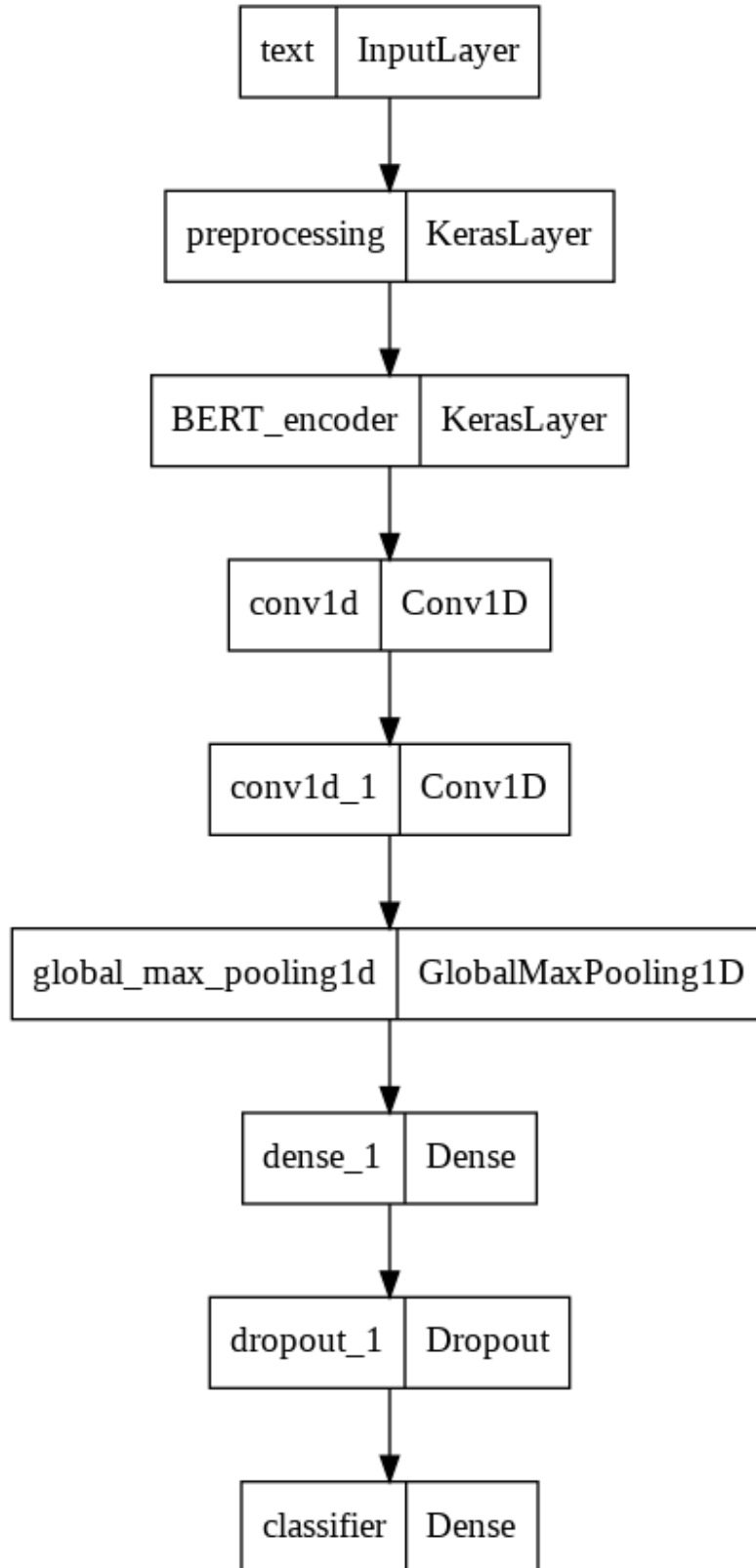


Figure 5.12. BERT + ANN model

The BERT+ANN and BERT+MLP models are trained for 50 epochs. As the number of epochs increases, the accuracy improves. Following is the formula used to find the number of training steps and number of warmup steps.

Number of Epochs = 50

Number of Training Steps = Steps Per Epoch * Number of Epochs

Number of Warmup Steps = 0.1 * Number of Training Steps

The accuracy obtained by the BERT + MLP model is 93.67% and BERT+ANN model is 93.55%, which indicates that the gap in conventional evaluation metrics is minimal but in terms of the explainability metrics BERT + ANN performs slightly better than the BERT+MLP model which is discussed later in the chapter.

5.2.3.3 LIME with other machine learning models

This section discusses the implementation of the LIME model with other linear machine learning models in order to provide explainability and interpretability. LIME is a novel explanation technique that explains the prediction of any classifier in an interpretable and faithful manner by learning an interpretable model locally around the prediction. It is a surrogate model, which means that it still uses the black-box machine learning models, but it tweaks the input slightly and tests the changes in prediction. There are two main properties of LIME explainer.

- Model agnostic refers to the property of LIME that can give explanations for any given supervised machine learning model by treating it as a ‘black-box’ separately.
- Local explanations mean that LIME gives explanations that are locally faithful within the surroundings or vicinity of the observation/sample being explained.

The same labelled data set that was used for BERT with ANN and MLP is used for training the LIME model. LIME is trained with the linear non-complex machine learning models like Random Forest, Naïve Bayes, Decision Tree, and Logistic Regression to extract the explanations. Table 5.12 summarizes the accuracy achieved by each of the models on the HateXplain dataset. It can be seen that Logistic Regression performed the best with an accuracy of 88.57%.

Table 5.12. Accuracy of Linear Models on HatXplain Dataset

Model	Accuracy
Random Forest	82.86%
Logistic Regression	88.57%
Decision Tree	82.86%
Naive Bayes	69%

In this section, LIME classification is demonstrated with an example. The comment text is "@ComedyPosts: Harlem shake is just an excus to go full retard for 30 seconds". After the preprocessing is performed on the text, the comment text is reduced to "comedypost harlem shake excus go full retard second". This comment text is obtained from the corpus of preprocessed pandas data frame and applied to LIME text explainer for each of the machine learning models. The same comment text is used for all the models for the purpose of comparison between each model.

Explainability for Random Forest

Figure 5.13 shows the explainability with LIME and Random Forest for a particular tweet. It can be observed that the LIME explainer has given weights to each useful word in the comment to indicate its' importance in the overall decision making. From Figure 5.13, we can see that words like "excus" and "retard" has the highest weights for contributing to the overall prediction probability at 0.10 and 0.07 respectively. The prediction probability of the tweet to be considered

as hate speech is reduced by the word “full”. Text that contributes in either way of prediction has been highlighted in the right side of the figure. The overall prediction probability for hate speech is 90% using the Random Forest classifier.

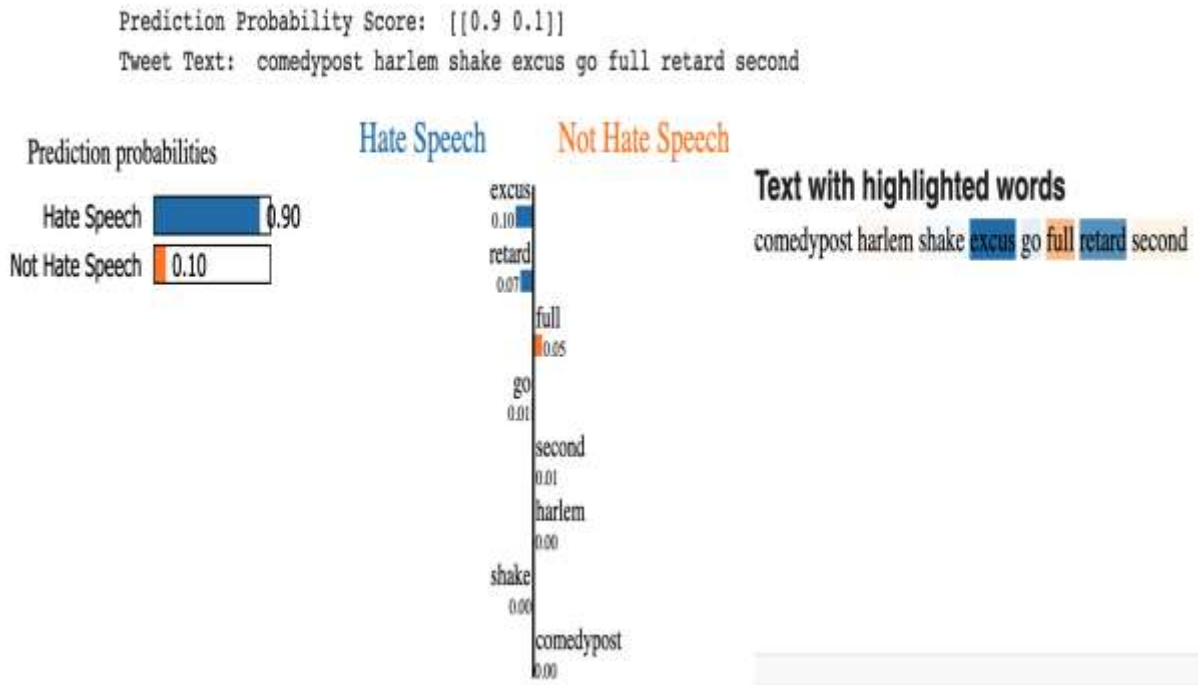


Figure 5.13. Explainability with Random Forest

Explainability for Gaussian Naive Bayes

Figure 5.14 shows the explainability with LIME and Gaussian Naïve Bayes for the example tweet. It can be observed that the LIME explainer has given weights to each useful word in the comment to indicate its’ importance in the overall decision making. From Figure 5.14, we can see that words like “full” and “excus” has the highest weights for contributing to the overall prediction probability at 0.08 and 0.07, respectively. Interestingly, the prediction probability of the tweet to be considered hate speech is reduced by the word “retard” in the case of Gaussian Naïve Bayes classifier. The word retard has the prediction probability of 0.14 which eventually increases the overall prediction probability of the text being not a hate speech by 20%. Text that contributes in either way of

prediction has been highlighted in the right side of the figure. The overall prediction probability for hate speech is 80% using the Gaussian Naïve Bayes classifier.

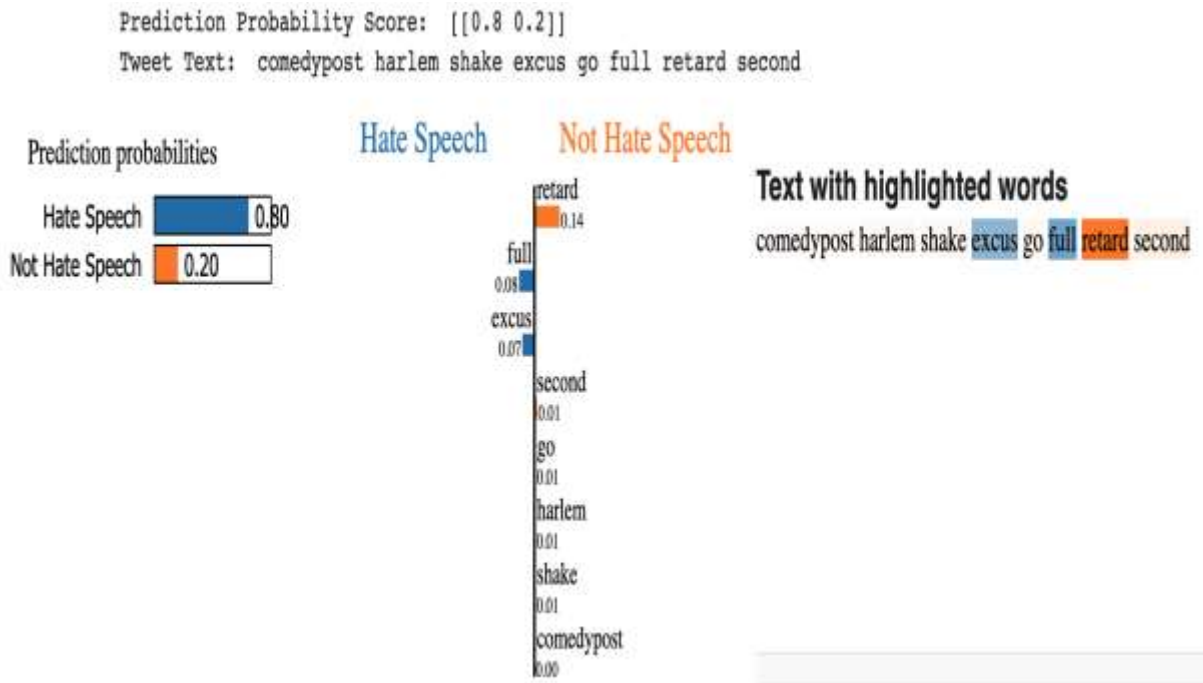


Figure 5.14. Explainability with Gaussian Naive Bayes

Explainability for Decision Tree

Figure 5.15 shows the explainability with LIME and Decision Tree for the example tweet. It can be observed that the LIME explainer has given weights to each useful word in the comment to indicate its' importance in the overall decision making. From Figure 5.15, we can see that words like “full”, “excus”, and “retard” have the highest weights for contributing to the overall prediction probability at 0.07, 0.06 and 0.06, respectively. The decision tree classifier has not given weight to predict the comment as non-hate speech for any of the words. We can see that trend in the text with highlighted words section. The overall prediction probability for hate speech is 100% using the Decision Tree classifier.

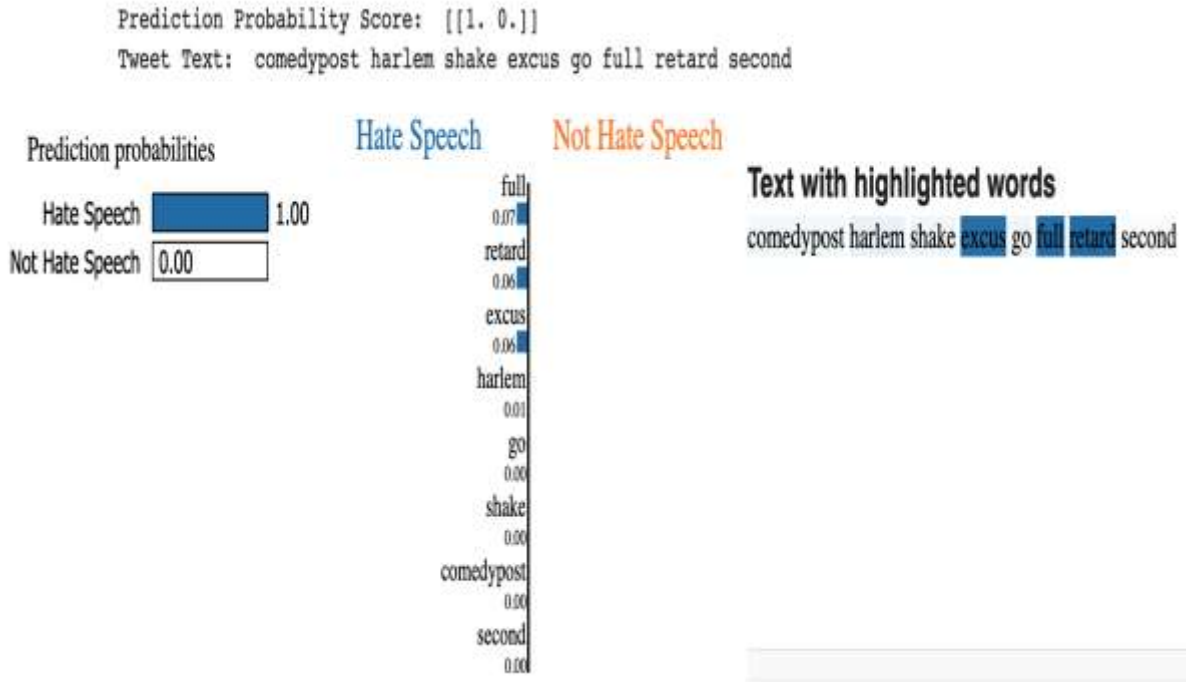


Figure 5.15. Explainability with Decision Tree

Explainability for Logistic Regression

Figure 5.16 shows the explainability with LIME and Logistic Regression for the example tweet. It can be observed that the LIME explainer has given weights to each useful word in the comment to indicate its' importance in the overall decision making. From Figure 5.16, we can see that words like "excus" and "second" has the highest weights for contributing to the overall prediction probability at 0.03 and 0.04 respectively. On the other hand, words like "retard" and "full" contribute to being not hate speech with the weights of 0.04 and 0.03, respectively. Text that contributes in either way of prediction has been highlighted in the right side of the figure. The overall prediction probability for hate speech is 95% using the Logistic Regression classifier.

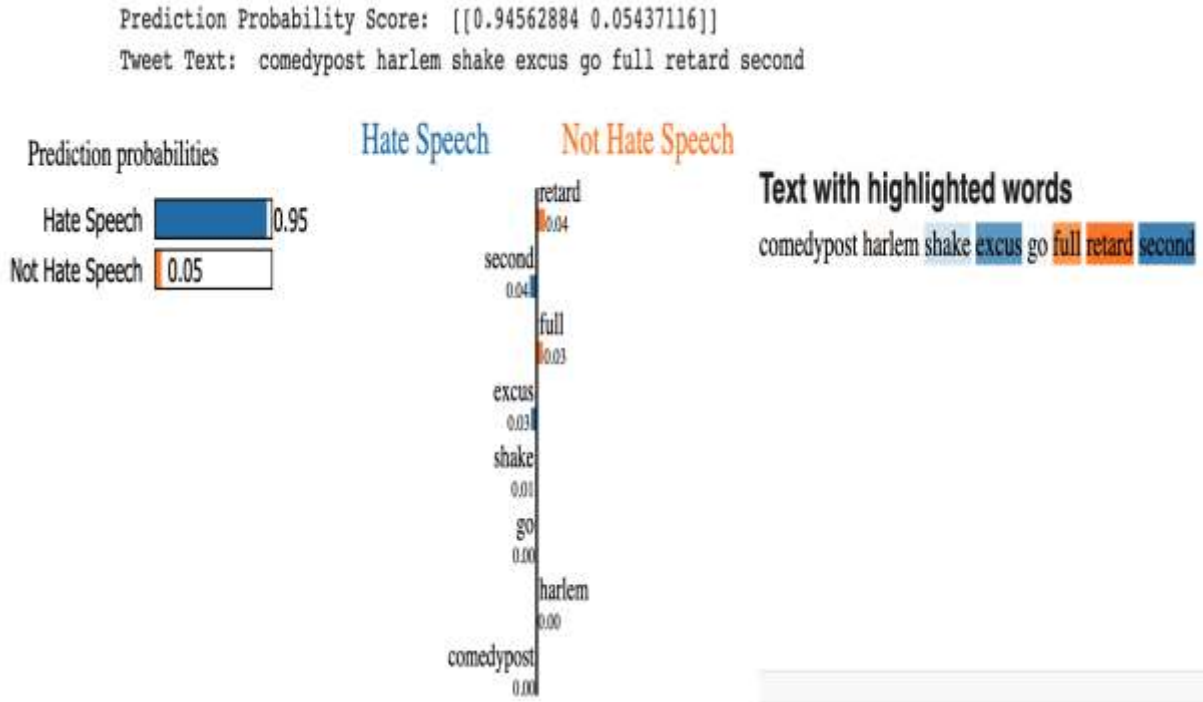


Figure 5.16. Explainability with Logistic Regression

5.2.4 Summary of Results for the HateXplain Dataset

The results of all the models on the HateXplain dataset, evaluated in terms of their accuracy, precision and F1 score are visualized in Figure 5.17. Table 5.13 gives the evaluation scores for all the models. It can be observed that BERT variants perform significantly better than the other linear explainable models with BERT + MLP having the highest accuracy of 93.67% closely followed by BERT + ANN with an accuracy of 93.55%. It can be observed that the measures such as precision, recall and F1 score as well, the BERT variants outperform the other linear models. Logistic Regression with LIME performs best among the linear models with an accuracy of 88.57% and F1 score of 93.75%. The results can be visualized in Figure 5.17 as a bar chart.

5.2.4.1 Explainability Metrics

We have used ERASER benchmark in order to measure the explainability of the trained models. ERASER (Evaluating Rationales and Simple English Reasoning) is a benchmark to evaluate

rationalized NLP models, which was proposed by DeYoung et al (2020). This is done by measuring the agreement with human rationales, measuring exact matches between predicted and reference rationales is likely too harsh, thus explainability has been accessed by measuring plausibility and faithfulness. The prediction is counted as a match if any of the words' prediction overlaps with the rationales annotated by humans. Token level calculations are done and compared with human annotations to derive the explainability. Various measures have been used from ERASER benchmark to calculate these comparisons.

Plausibility is the measure of how cogent the interpretation is to a human. To measure plausibility, the metrics IOU (Intersection-Over-Union) F1 score, Token F1 score and Area Under the Precision-Recall curve (AUPRC) score were calculated. IOU (Intersection-Over-Union) F1 score is calculated for token level. Partial matches where prediction overlaps more than 0.5 with either one of the ground truth rationales. Token-level F1 scores are measured from token-level precision and recall. AUPRC or Area Under the Precision-Recall curve measures soft token scoring. The higher the values of all these metrics, the greater the plausibility.

Faithfulness is the measure of the accuracy of the true reasoning process of the model. To measure the faithfulness of the models, comprehensiveness and sufficiency have been calculated. Comprehensiveness score is a measure of change in the probability of the output of the originally predicted class after eliminating significant tokens. A higher comprehensiveness score indicates a more faithful interpretation. Sufficiency measures the sufficiency of the important tokens to sustain the predictions. It captures the degree to which the snippets within the exact rationales are adequate for a model to make a prediction. The lower the sufficiency, the more faithful the model is.

Table 5.14 provides a summarised view of the explainability metrics calculated on all the models implemented. It can be observed that, BERT + MLP is the best performing model in terms of

plausibility. The model BERT + MLP shows the best values of IOU F1, Token F1 and AUPRC as compared to the other models. In terms of faithfulness, BERT + ANN model shows the best results with the highest comprehensiveness score of 0.4199. The achieved results improved over the ones in the base paper by Mathew et al. (2020). BERT variants have the best convincing interpretation to the humans. BERT + ANN shows a bit high comprehensiveness than BERT + MLP. The reason is the simple structure of ANN than MLP. The same trend in parameters of sufficiency have been observed in the base paper by Mathew et al. (2020) considered as well.

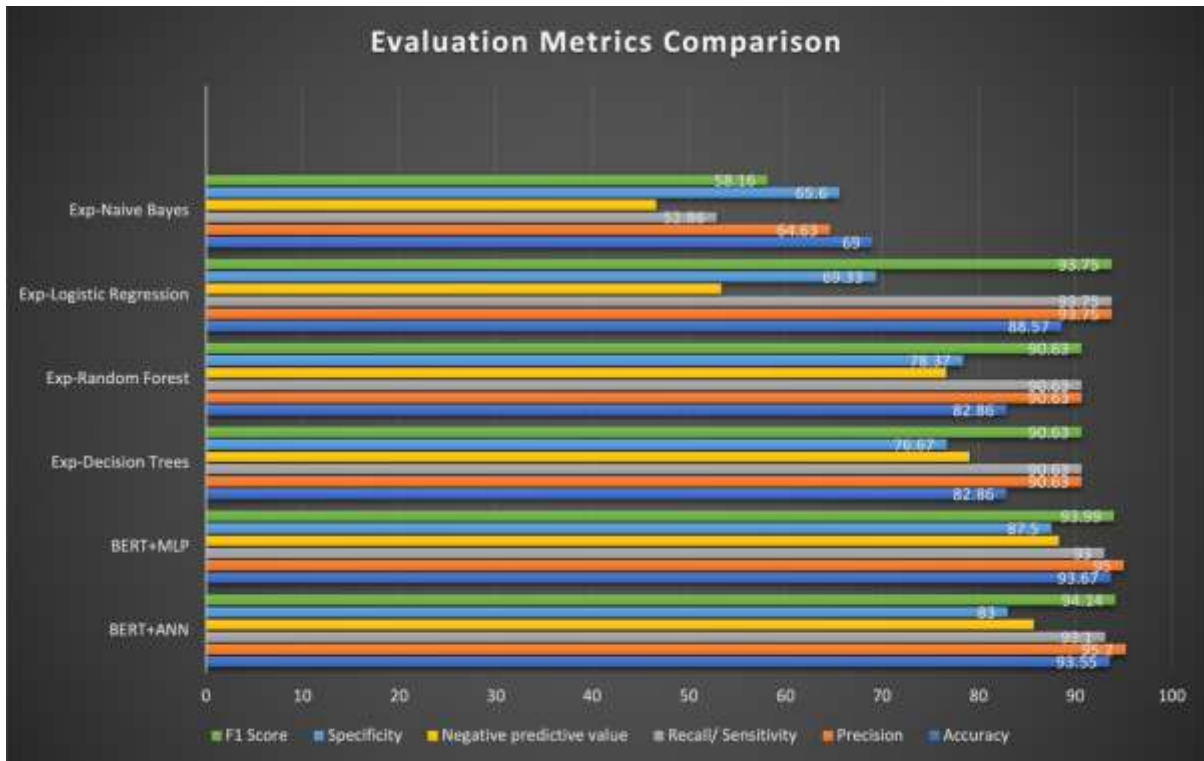


Figure 5.17. Result summary of all Models on the HateXplain Dataset

Table 5.13. Results of models on the HateXplain data set

S. No	Full Form	Features	Accuracy	Precision	Recall/ Sensitivity	Negative predictive value	Specificity	F1 Score
1	Bidirectional Encoder Representations from Transformers + Artificial Neural Network Layers (BERT+ANN)	Explainable, Layer Wise Propagation	93.55	95.2	93.1	85.7	83	94.14
2	Bidirectional Encoder Representations from Transformers + Multilayer Perceptron (BERT+MLP)	Explainable, Layer Wise Propagation	93.67	95	93	88.3	87.5	93.99
3	Exp-Decision Trees	Explainable (LIME)	82.86	90.63	90.63	79.03	76.67	90.63
4	Exp-Random Forest	Explainable (LIME)	82.86	90.63	90.63	76.65	78.37	90.63
5	Exp-Logistic Regression	Explainable (LIME)	88.57	93.75	93.75	53.33	69.33	93.75
6	Exp-Naive Bayes	Explainable (LIME)	69	64.63	52.86	46.6	65.6	58.16

Table 5.14. Explainability Metrics

Technique	Plausibility			Faithfulness	
	IOU F1	Token F1	AUPRC	Comprehensiveness	Sufficiency
BERT + ANN	0.1888	0.5074	0.8384	0.4199	0.0055
BERT + MLP	0.2980	0.5298	0.8589	0.3574	0.003
DT - LIME	0.1676	0.3887	0.7487	0.2993	0.0442
RF - LIME	0.2387	0.5118	0.8469	0.4132	0.0014
LR - LIME	0.1008	0.2271	0.5284	0.2132	0.0482
NB - LIME	0.1287	0.1818	0.5938	0.1999	0.0514

BERT + ANN: Bidirectional Encoder Representations from Transformers with Artificial Neural Networks

BERT + MLP: Bidirectional Encoder Representations from Transformers with Multilayer Perceptron

DT - LIME: Decision Tree with Local interpretable model-agnostic explanations

RF - LIME: Random Forest with Local interpretable model-agnostic explanations

LR - LIME: Logistic Regression with Local interpretable model-agnostic explanations

NB - LIME: Naive Bayes with Local interpretable model-agnostic explanations

5.2.4.2 Bias based Metrics

The hate speech detection models could make biased predictions for particular groups who are already the target of such abuse (Sap et al. 2019; Davidson, Bhattacharya, and Weber 2019). To measure these unintended model biases, the AUC based metrics by Borkan et al. (2019) have been used. We have computed Subgroup AUC (Area under the ROC curve), BPSN (Background Positive, Subgroup Negative) AUC and BSNP (Background Negative, Subgroup Positive) AUC. Subgroup AUC metric for this use case is a measure of the ability of the model to segregate the toxic and normal comments. The higher the value of subgroup AUC, the better the model at differentiating between the toxic and normal posts. BPSN (Background Positive, Subgroup Negative) AUC metric is the measure of false-positive rates of the model while BSNP (Background Negative, Subgroup Positive) AUC is a measure of false-negative rates of the model. The higher the value of BPSN, the less likely is the model to give false positives and higher value of BSNP indicates that the model is less likely to give false negatives. For this dataset, these metrics have been calculated with respect to a community.

Table 5.15 provides a summarised view of the bias-based metrics calculated on all the models implemented. We can see that bias-based metrics of BERT variants is significantly more accurate than the other linear models. BERT + MLP has highest value of subgroup AUC, BPSN AUC and

BSNP AUC with 0.8229, 0.7752 and 0.8077 respectively followed by BERT + ANN with the values of 0.7977, 0.7188 and 0.7391 respectively.

Table 5.15. Bias Based Metrics

Technique	Subgroup AUC	BPSN AUC	BSNP AUC
BERT + ANN	0.7977	0.7188	0.7391
BERT + MLP	0.8229	0.7752	0.8077
DT - LIME	0.6926	0.6578	0.6617
RF - LIME	0.7627	0.6977	0.5978
LR - LIME	0.5266	0.4522	0.4991
NB - LIME	0.6136	0.4812	0.5049

Chapter 6

Conclusions and Future Work

As a part of this research study, two data sets were taken to demonstrate Hate Speech Detection using Explainable Artificial Intelligence (XAI). Exploratory data analysis was performed on the data sets to uncover various patterns and insights and various explainable models were trained on both datasets to extract useful interpretable results. The conclusion of the study performed as a part of this research work is discussed in this chapter.

6.1 Conclusions of the study on the Google Jigsaw dataset

The Google Jigsaw dataset comprises of user discussions from talk pages of English Wikipedia, and it was released by Google Jigsaw. We trained various existing interpretable models like Decision Tree, KNN, Random Forest, Multinomial Naïve Bayes, Logistic Regression, and LSTM. We found that LSTM has outperformed the other models in terms of accuracy (97.6%) and Recall (83%) scores. The Random Forest model has the best performance in terms of precision (90%) and specificity (87%). KNN, Logistic Regression and Multinomial Naïve Bayes have comparatively low evaluation scores as compared to the other models, but they perform very well in terms of accuracy with 90%, 97% and 96% respectively. Decision trees and Random Forest also have significantly good performance with an accuracy of 89% and 91%, respectively. It was observed that LSTM model gives better overall performance in terms of accuracy, precision, recall and F1 measures as compared to the studies of Risch et al. (2020).

6.2 Conclusion of the study on the HateXplain dataset

The HateXplain dataset comprises of posts from Twitter and Gab and is annotated by human annotators. Several state-of-the-art models were tested on this dataset to perform evaluation on several aspects of the hate speech detection. These models contained explainability imbibed in them in some or the other way. LIME was used with interpretable models like Decision Tress, Random Forest, Logistic regression, and Naïve Bayes to extract weights of words that contributed significantly to model's decision making. Apart from that, variants of BERT were created to achieve best performance. The best performance has been observed for the BERT variants, BERT+ANN and BERT+MLP as compared to the other models. BERT+ANN have a slightly better overall performance than BERT+MLP. For appropriate comparisons, the evaluation metrics are divided into three subsets namely, Performance metrics (Accuracy, Precision, Recall, Negative Predicted Value, Specificity, F1 Score), Bias based metrics, and Explainability Metrics (Plausibility and Faithfulness) as in Mathew et al. (2020). The accuracy score of BERT+MLP is 93.67% and BERT+ANN is 93.55%, which has outperformed the simple BERT implementations done by Mathew et al. (2020) with an accuracy score 69.8%. The prime reason behind this difference is combining BERT with neural network models like MLP and ANN. Apart from that, our models are trained on 50 epochs which took around 11.5 and 8.3 hours, simultaneously on Google Colab Pro. The precision scores of BERT + ANN and BERT + MLP respectively are 95.2% and 95%, recall is 93.1% and 93%. This results of the F1 score are calculated to be 94.14% and 93.99%.

In terms of Bias based metrics, the performance of BERT variant models has been able to perform better in reducing the unintended model bias for all the bias metrics. We observe that presence of community terms within the rationales is effective in reducing the unintended bias. BERT + MLP

models handles this bias much better than other models in terms of Subgroup, BPSN (Background Positive, Subgroup Negative), and BNSP (Background Negative, Subgroup Positive) with the values of (0.8229, 0.7752, 0.8077) respectively which is an improvement over simple BERT implementation (0.807, 0.745, 0.763) by Mathew et al. (2020). Future research on hate speech, should consider the impact of the model performance on individual communities to have a clear understanding on the impact.

Taking into account the Explainability metrics using ERASER benchmark by DeYoung et al. (2019), there are two main factors called Plausibility (defined by IOU F1, Token F1 and AUPRC) and Faithfulness (defined by Comprehensiveness and Sufficiency). The best performing models, BERT ANN and BERT MLP have plausibility (IOU F1, Token F1, AUPRC) values of (0.188,0.507,0.8384) and (0.29,0.529, 0.8589), respectively when compared to the base BERT model (0.222,0.506,0.841) in the paper by Mathew et al. (2020). BERT + MLP has better performance than the simple BERT implementation. Similarly, the Faithfulness (Comprehensiveness, Sufficiency) values were found to be (0.419,0.0055) in BERT+ANN and (0.3574,0.003) in BERT+ MLP. BERT + ANN has better performance as compared to the BERT implementation in the paper by Mathew et al. (2020 (0.436, 0.008).

Hence, it can be derived that the variants of BERT used in the research work, BERT + ANN have best performance for explainability and BERT + MLP have the best performance metrics among all the traditional models like Logistic Regression, KNN, Naïve Bayes, decision trees and Random Forests.

6.3 Directions for future work

Traditional Artificial Intelligence (AI) is like a black box in which how the processing takes place is not visible, i.e., how an answer to a question is obtained, is not explained. Explainable artificial

intelligence (XAI) is a new area of artificial intelligence that aims to bridge this gap. Many applications require a transparent explanation of how an answer was obtained as a result of processing of a model and this alone ensures trust and reliability of the model and its results. In future, similar work can be carried out in more applications and fields and explainable artificial intelligence can be researched to be used in real-time and in critical applications like defense, medical healthcare, etc. Explainability or interpretability of Artificial Intelligence would also be useful in detecting bias and misconceptions that might go unnoticed with traditional Artificial Intelligence owing to the fact that it is like a black box. More hate speech datasets can be incorporated together to widen the scope of interpretations and reducing biases. Explainable Artificial Intelligence can also prove useful in making business decisions that are not just based on data analytics but also using explanations generated using XAI. Future research in developing interfaces that can provide the interpretability of the traditional black-box models can be done in which various other visualization methods can be used like SHAP (Shapely Additive Explanations) and counterfactual methods along with methods like LIME used in this research.

References

- [1] Gohel, P., Singh, P., & Mohanty, M. *Explainable AI: current status and future directions*.
<https://doi.org/10.1109/ACCESS.2017.DOI>
- [2] Barredo Arrieta, A., Díaz-Rodríguez, N., del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020a). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115.
<https://doi.org/10.1016/j.inffus.2019.12.012>
- [3] Hrnjica, B., & Softic, S. (2020). Explainable AI in Manufacturing: A Predictive Maintenance Case Study. *IFIP Advances in Information and Communication Technology*, 592 IFIP, 66–73. https://doi.org/10.1007/978-3-030-57997-5_8
- [4] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why Should I Trust You?”: Explaining the Predictions of Any Classifier. *NAACL-HLT 2016 - 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Demonstrations Session*, 97–101.
<https://doi.org/10.48550/arxiv.1602.04938>
- [5] A. Das and P. Rad, “Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey,” Jun. 2020, doi: [10.48550/arXiv.2006.11371](https://doi.org/10.48550/arXiv.2006.11371).
- [6] Founta, A. M., Chatzakou, D., Kourtellis, N., Blackburn, J., Vakali, A., & Leontiadis, I. (2018). A Unified Deep Learning Architecture for Abuse Detection. *WebSci 2019 - Proceedings of the 11th ACM Conference on Web Science*, 105–114.
<https://doi.org/10.48550/arxiv.1802.00385>

- [7] Davidson, T., Warmley, D., Macy, M., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. <http://arxiv.org/abs/1703.04009>
- [8] Chatzakou, D., Kourtellis, N., Blackburn, J., de Cristofaro, E., Stringhini, G., & Vakali, A. (2017). Mean birds: Detecting aggression and bullying on Twitter. WebSci 2017 - Proceedings of the 2017 ACM Web Science Conference, 13–22. <https://doi.org/10.1145/3091478.3091487>
- [9] Chen, Y., Zhou, Y., Zhu, S., & Xu, H. (2012). Detecting offensive language in social media to protect adolescent online safety. Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust and 2012 ASE/IEEE International Conference on Social Computing, SocialCom/PASSAT 2012, 71–80. <https://doi.org/10.1109/SocialCom-PASSAT.2012.55>
- [10] Arras, L., Montavon, G., Müller, K. R., & Samek, W. (2017). Explaining recurrent neural network predictions in sentiment analysis. *EMNLP 2017 - 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA 2017 - Proceedings of the Workshop*. <https://doi.org/10.18653/v1/w17-5221>
- [11] G. Montavon, W. Samek, and K.-R. Müller, “Methods for interpreting and understanding deep neural networks,” *Digital Signal Processing*, vol. 73, pp. 1–15, Feb. 2018, doi: [10.1016/j.dsp.2017.10.011](https://doi.org/10.1016/j.dsp.2017.10.011).
- [12] Fernandez, A., Herrera, F., Cordon, O., Jose Del Jesus, M., & Marcelloni, F. (2019). Evolutionary fuzzy systems for explainable artificial intelligence: Why, when, what for, and where to? *IEEE Computational Intelligence Magazine*, 14(1), 69–81. <https://doi.org/10.1109/MCI.2018.2881645>

- [13] Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2019). Explaining explanations: An overview of interpretability of machine learning. *Proceedings - 2018 IEEE 5th International Conference on Data Science and Advanced Analytics, DSAA 2018*. <https://doi.org/10.1109/DSAA.2018.00018>
- [14] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, “Toward Interpretable Machine Learning: Transparent Deep Neural Networks and Beyond,” Mar. 2020, doi: [10.48550/arXiv.2003.07631](https://doi.org/10.48550/arXiv.2003.07631).
- [15] O. Kanerva, “Evaluating explainable AI models for convolutional neural networks with proxy tasks,” 2019. https://www.semanticscholar.org/paper/Evaluating-explainable-AI-models-for-convolutional_Kanerva/d91062a3e13ee034af6807e1819a9ca3051daf13
- [16] M.-A. Clinciu and H. Hastie, “A Survey of Explainable AI Terminology,” in Proceedings of the 1st Workshop on Interactive Natural Language Technology for Explainable Artificial Intelligence (NL4XAI 2019), 2019, pp. 8–13. doi: 10.18653/v1/W19-8403.
- [17] F. Doshi-Velez and B. Kim, “Towards A Rigorous Science of Interpretable Machine Learning,” Feb. 2017, doi: [10.48550/arXiv.1702.08608](https://doi.org/10.48550/arXiv.1702.08608).
- [18] Hind, M., Wei, D., Campbell, M., Codella, N. C. F., Dhurandhar, A., Mojsilović, A., Natesan Ramamurthy, K., & Varshney, K. R. (2019). TED: Teaching AI to explain its decisions. *AIES 2019 - Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 123–129. <https://doi.org/10.1145/3306618.3314273>
- [19] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems, 2017-December*.

- [20] S. M. Lundberg et al., “Explainable AI for Trees: From Local Explanations to Global Understanding,” arXiv:1905.04610 [cs, stat], May 2019, Accessed: May 11, 2022. [Online]. Available: <http://arxiv.org/abs/1905.04610>
- [21] Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. In *Artificial Intelligence* (Vol. 267). <https://doi.org/10.1016/j.artint.2018.07.007>
- [22] H. Nori, S. Jenkins, P. Koch, and R. Caruana, “InterpretML: A Unified Framework for Machine Learning Interpretability,” Sep. 2019, doi: [10.48550/arXiv.1909.09223](https://doi.org/10.48550/arXiv.1909.09223).
- [23] DeYoung, J., Jain, S., Rajani, N. F., Lehman, E., Xiong, C., Socher, R., & Wallace, B. C. (2020). ERASER: A Benchmark to Evaluate Rationalized NLP Models. <https://doi.org/10.18653/v1/2020.acl-main.408>
- [24] B. Mathew, P. Saha, S. M. Yimam, C. Biemann, P. Goyal, and A. Mukherjee, “HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection,” arXiv:2012.10289 [cs], Dec. 2020, Accessed: Jun. 14, 2021. [Online]. Available: <http://arxiv.org/abs/2012.10289>
- [25] “ML | Overview of Data Cleaning,” GeeksforGeeks, May 15, 2018. <https://www.geeksforgeeks.org/data-cleansing-introduction/> (accessed Apr. 03, 2022).
- [26] “What is Tokenization | Methods to Perform Tokenization.” <https://www.analyticsvidhya.com/blog/2019/07/how-get-started-nlp-6-unique-ways-perform-tokenization/> (accessed Apr. 03, 2022).
- [27] K. Kargin, “NLP: Tokenization, Stemming, Lemmatization and Part of Speech Tagging,” *MLearning.ai*, Nov. 20, 2021. <https://medium.com/mllearning-ai/nlp-tokenization-stemming-lemmatization-and-part-of-speech-tagging-9088ac068768> (accessed May 30, 2022).

- [28] R. K. Pearson, "Exploratory Data Analysis: A First Look," in Exploratory Data Analysis Using R, Chapman and Hall/CRC, 2018.
- [29] "Using CountVectorizer to Extracting Features from Text," *GeeksforGeeks*, Jul. 15, 2020. <https://www.geeksforgeeks.org/using-countvectorizer-to-extracting-features-from-text/> (accessed Apr. 03, 2022).
- [30] Kamath, U., Graham, K. L., & Emara, W. (2022). Bidirectional encoder representations from transformers (BERT). *Transformers for Machine Learning*, 43-70. <https://doi.org/10.1201/9781003170082-3>
- [31] R. Nair, V. N. V. Prasad, A. Sreenadh and J. J. Nair, "Coreference Resolution for Ambiguous Pronoun with BERT and MLP," 2021 International Conference on Advances in Computing and Communications (ICACC), 2021, pp. 1-5, doi: 10.1109/ICACC-202152719.2021.9708203.
- [32] Gad, A. F., & Jarmouni, F. E. (2021). Introduction to artificial neural networks (ANN). *Introduction to Deep Learning and Neural Networks with Python™*, 15-32. <https://doi.org/10.1016/b978-0-323-90933-4.00007-3>
- [33] Neural network | Introduction to neural network | Neural network for DL. (2021, March 12). Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/03/basics-of-neural-network/>
- [34] R. Pramoditha, "The Concept of Artificial Neurons (Perceptrons) in Neural Networks," Medium, Dec. 29, 2021. <https://towardsdatascience.com/the-concept-of-artificial-neurons-perceptrons-in-neural-networks-fab22249cbfc> (accessed May 30, 2022).

- [35] Bisong, E. (2019). The Multilayer Perceptron (MLP). Building Machine Learning and Deep Learning Models on Google Cloud Platform, 401-405. https://doi.org/10.1007/978-1-4842-4470-8_31
- [36] Understanding the decision tree structure. (n.d.). scikit-learn. Retrieved May 28, 2022, from https://scikit-learn.org/stable/auto_examples/tree/plot_unveil_tree_structure.html
- [37] “K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint,” www.javatpoint.com. <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning> (accessed May 30, 2022).
- [38] Moore, C., Murphy, A. Random forest (machine learning). Reference article, Radiopaedia.org. (accessed on 30 May 2022) <https://doi.org/10.53347/rID-67772>
- [39] G, M. K. (2021). Accuracy analysis for logistic regression algorithm and random forest algorithm to detect frauds in mobile money transaction. *Revista Gestão Inovação e Tecnologias*, 11(4), 1228-1240. <https://doi.org/10.47059/revistageintec.v11i4.2182>
- [40] Seref, B., & Bostanci, E. (2018). Sentiment analysis using naive Bayes and complement naive Bayes classifier algorithms on Hadoop framework. 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). <https://doi.org/10.1109/ismsit.2018.8567243>
- [41] Biecek, P., & Burzykowski, T. (2021). Local interpretable model-agnostic explanations (LIME). *Explanatory Model Analysis*, 107-123. <https://doi.org/10.1201/9780429027192-11>