Toxicity Classification Model

Project Report

Submitted by
Hemang Sharma (Student ID: 24695785)

ABSTRACT

The prevalence of toxic and abusive language in online communication has become a significant problem in the digital age. To address this issue, researchers and developers have been working on creating automated tools and models that can detect and classify toxic language in text data. One such tool is the Toxicity Classification Model, which is designed to identify and flag abusive language in online communication.

Toxicity Classification Model is a type of machine learning model that is designed to identify and classify toxic or abusive language in text data. The model is trained on a dataset of labelled examples of toxic language and learns to recognise patterns and features in the text that are associated with toxic or abusive content.

The goal of a Toxicity Classification Model is to accurately predict whether a given piece of text contains toxic language, and to assign a score or label that indicates the degree of toxicity. There are many different approaches to building a Toxicity Classification Model, but most involve using a combination of natural language processing (NLP) techniques and machine learning algorithms.

This report provides a comprehensive overview of the Toxicity Classification Model and the challenges involved in building and deploying this type of model. Ultimately, this report highlights the importance of the Toxicity Classification Model in promoting healthy online interactions and creating a safer and more inclusive online environment.

ACKNOWLEDGEMENT

I would like to acknowledge the Gadigal people of the Eora Nation upon whose ancestral lands our City campus now stands. I would also like to pay respect to the Elders both past and present, acknowledging them as the traditional custodians of knowledge for this land. I would also like to convey my heartfelt thanks to, Dr. Shibani Aileen for her invaluable advice, constant encouragement, prompt assistance and provision of an ideal research environment. Throughout the project, despite her hectic schedule, she has been a cordial supporter of my efforts to complete this project.

Hemang Sharma (24695785)

List of Figures

TABLE OF CONTENTS

1. Introduction

Online harassment and hate speech are becoming increasingly prevalent in the digital age, creating a need for automated tools that can detect and restriction abusive behavior. We have developed a toxicity classification model that uses NLP techniques to identify and classify harmful comments. Our project goal is to develop tools to help moderators identify and remove such comments, improve online conversations, and create a safer environment for participating in online conversations.

We have used a dataset labelled for identity references and optimized a metric designed to measure unintended bias. Our model is designed to predict the toxicity level of user-submitted comments on a value between 0 and 1, inclusive. We classified the problem as a regression task using the mean squared error (MSE) as the evaluation metric.

The model is designed to rapidly predict toxicity scores and does not require interpretability. We have taken steps to minimize bias around identity references, ensuring our model is useful for a wide range of applications and conversations.

In this report, we have provided a detail description of data, NLP methods and techniques, insights, project outcomes, challenges, and solutions.

2. Data Description

The dataset for the project was obtained from the Jigsaw Toxic Comment Classification Challenge hosted on Kaggle. The dataset contains over 150,000 comments on Wikipedia talk pages, each rated on a scale of 0 to 1 for toxicity. Also, because the dataset is labelled with Identity mentions, the model can be optimized to minimize bias with respect to identity mentions.

This dataset was compiled by Jigsaw, a technology incubator founded by Google to address the Internet's most pressing problems, including online toxicity. The dataset was obtained from Kaggle website.

After obtaining the data set, several preprocessing steps were performed to prepare the text data for analysis, including removing redundant features, tagging, head word removal, and stop word deletion. We also divide the dataset into a training set and a test set respectively with a ratio of 80:20.

Overall, the Jigsaw Toxic Comment Classification Challenge dataset provided a diverse and complex set of comments to work with and allowed us to develop models to detect toxicity and reduce compliance alert bias.

3. Data Analysis

To begin our analysis, we first uploaded the training and testing datasets in Python using the Pandas package. We used functions like describe() and info() to get a sense of data structure. The training dataset has 1,804,874 rows and 44 columns, while the testing dataset has 97,320 rows and 1 column.

3.1 Target Feature

We have explored the training dataset to identify the patterns, its characteristics, and relationships, and investigate any potential issues like outliers or biased data that could impact the model and accuracy of model. To get a better understanding of the distribution of target variables in our dataset, we plotted graphs.

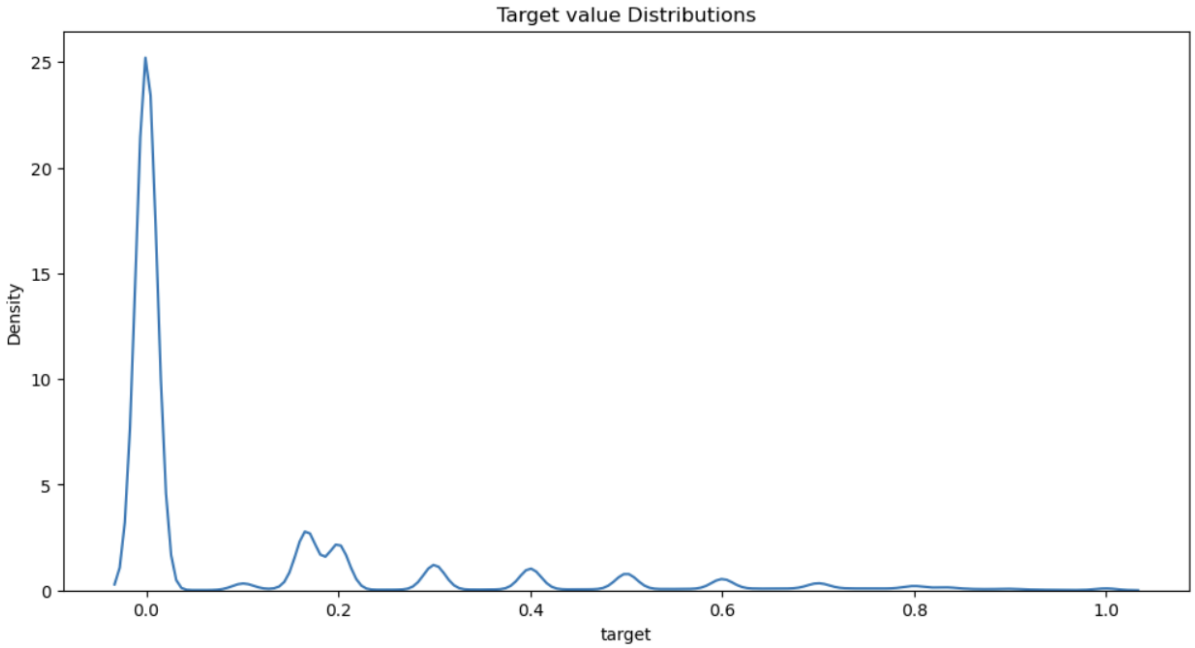


Figure1: Distribution of the Target value

As shown in Fig1, the range of the target variable is from 0 to 1. The distribution appears to be skewed towards the left, which means that a large number of the comments are non-toxic (<0.5). There can be considerable chances that our dataset may be imbalanced and could impact the model's performance.

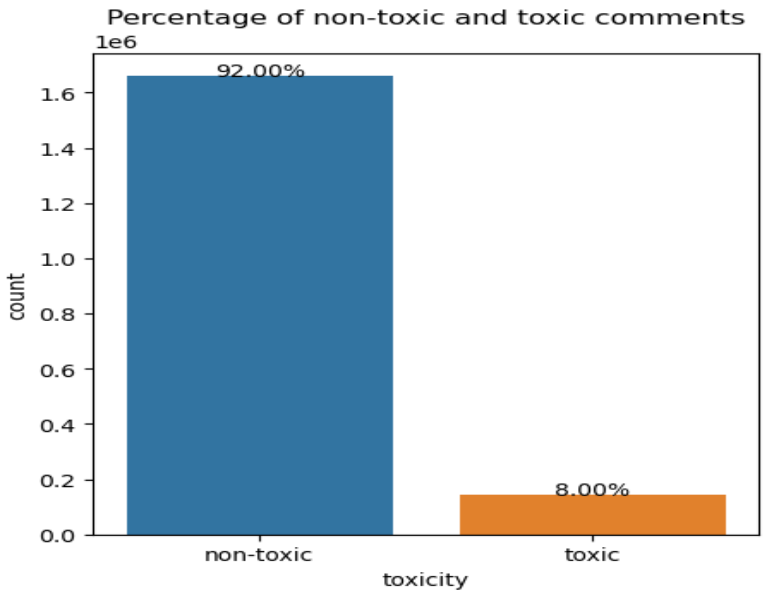


Figure2: Percentage of non-toxic and toxic comments

This graph reveals important information about the training dataset, the graph showed that a major part of the comments

had a low toxicity level. 92.00% of comments were non-toxic and 8.00% of comments were toxic, which means that the dataset is imbalanced.

3.2. Toxicity Subtype feature

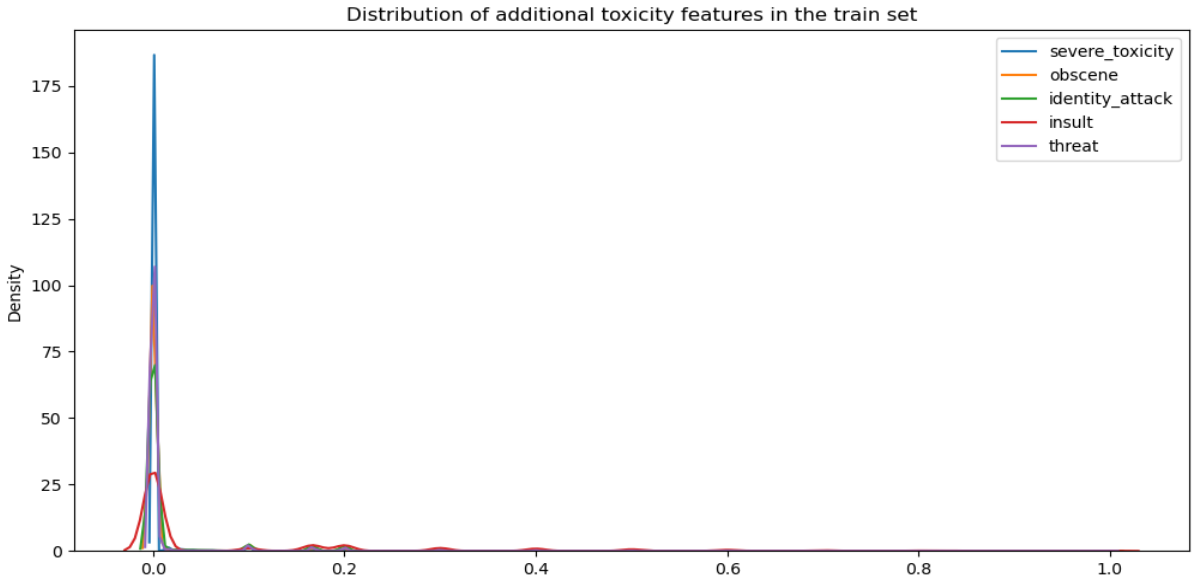


Figure3: Distribution of additional toxicity in the train set

We have generated the distribution plot for each Toxicity Subtype. The toxicity subtypes are ‘severe_toxicity’, ‘obscene’, ‘identity_attack’, ‘insult’, and ‘threat’. The main motive of this visualization is to explore the distribution of these features and help us identify any relationship between features and target variables.

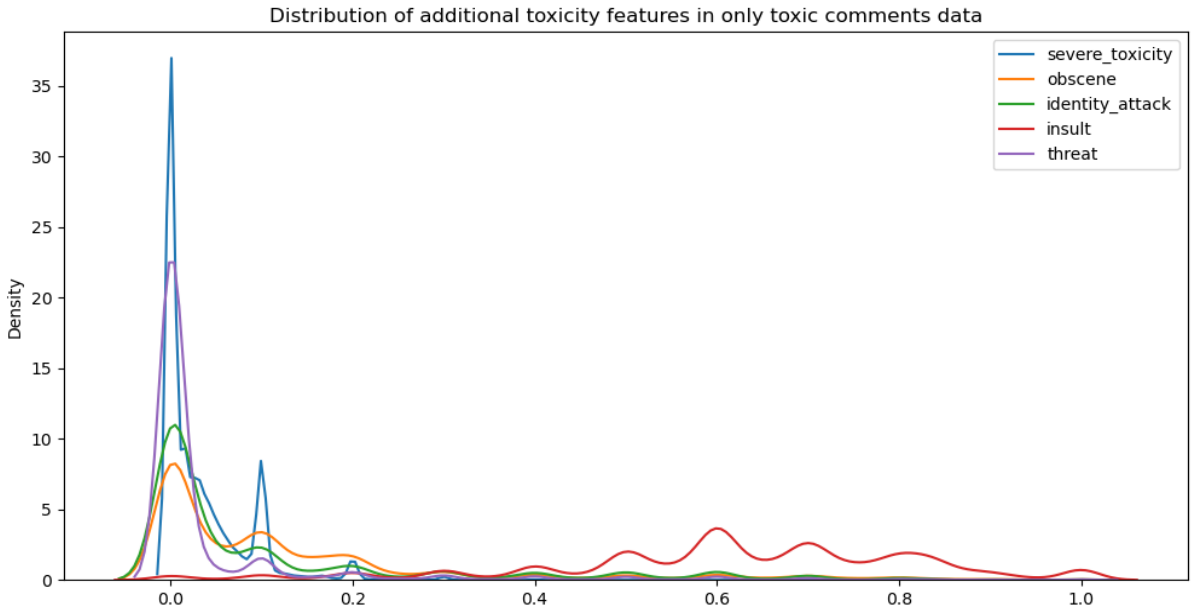


Figure4: Distribution of additional toxicity features in only toxic comments data

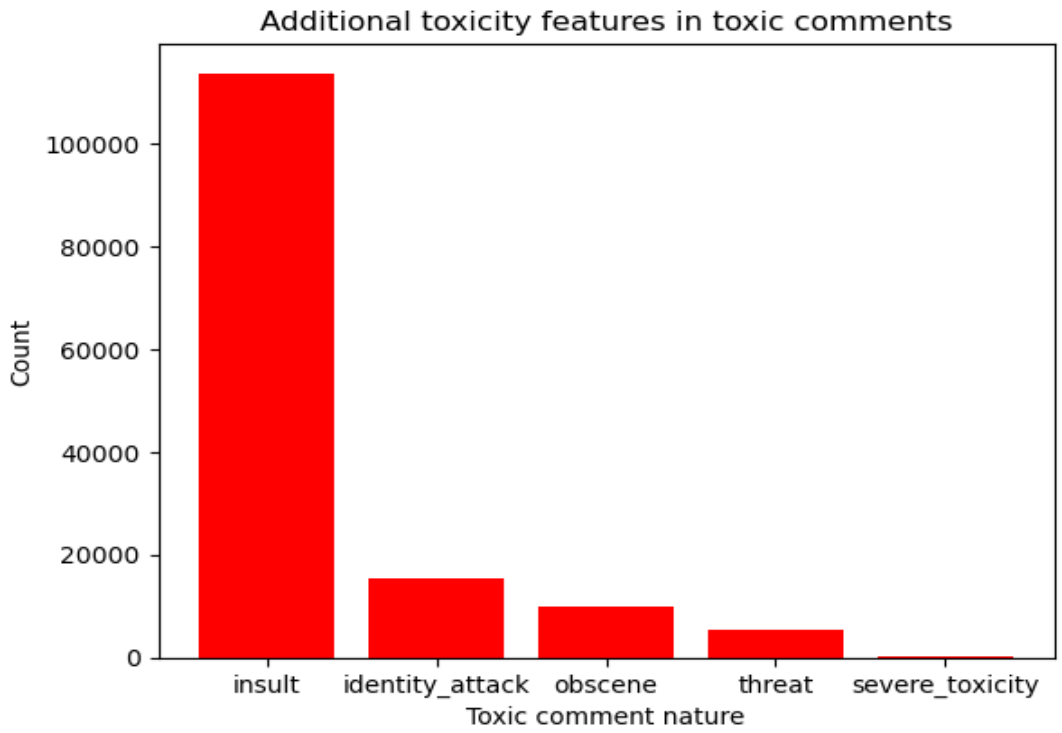


Figure5: Additional toxicity features in toxic comments

In our data set we had only 8% of data which had toxic comments. Out of that 8%, 81% of toxic comments are insults, 8.37% are identity attacks, 7.20 are obscene, 3.35% are threats and very small number of comments are severely toxic.

3.3. Identity Attributes

The identity attributes in the dataset include male, female, homosexual_gay_or_lesbian, bisexual, heterosexual, Christian, Jewish, Muslim, black, white, Asian, and Latino. We examined the distribution of these attributes in the dataset and explores the nature of toxic comments that involve these attributes. We first investigated the distribution of the identity features over the dataset, followed by race/ethnic references, sexual orientation references and gender references.

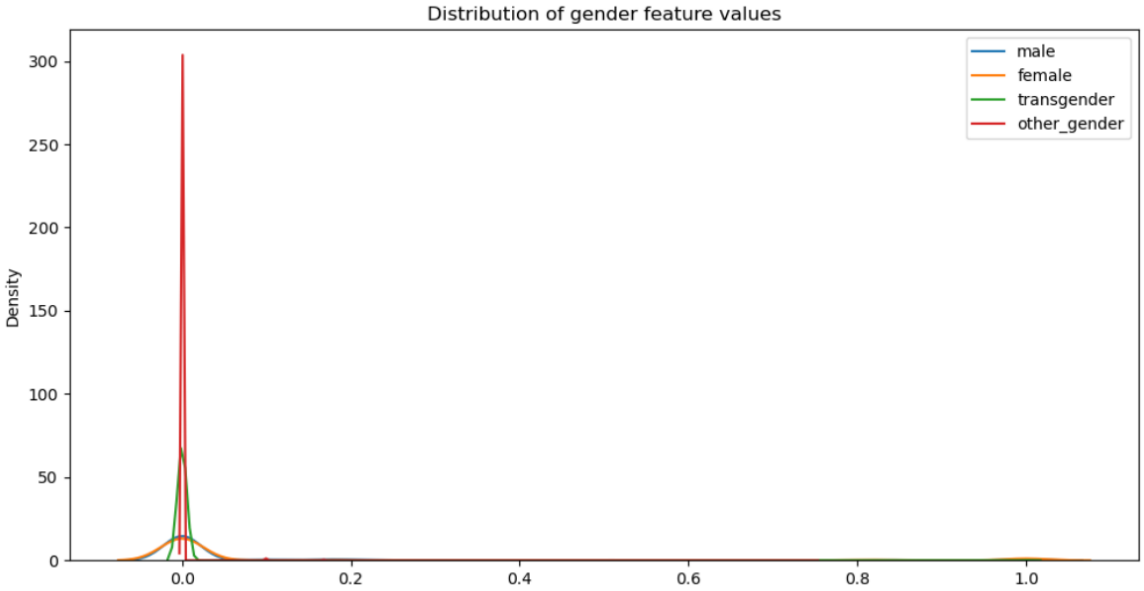


Figure6: Distribution of gender feature values in the train dataset

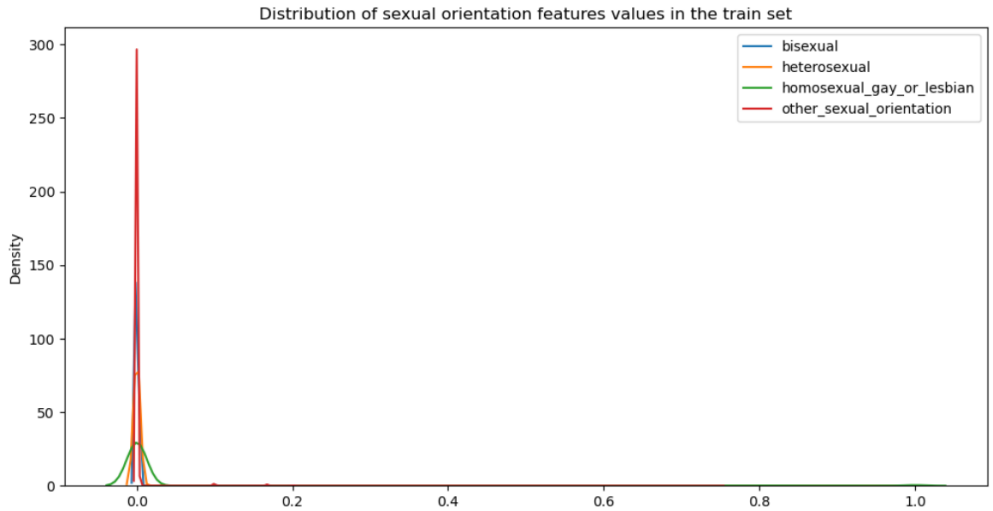


Figure 7: Distribution of sexual orientation features values in the train dataset

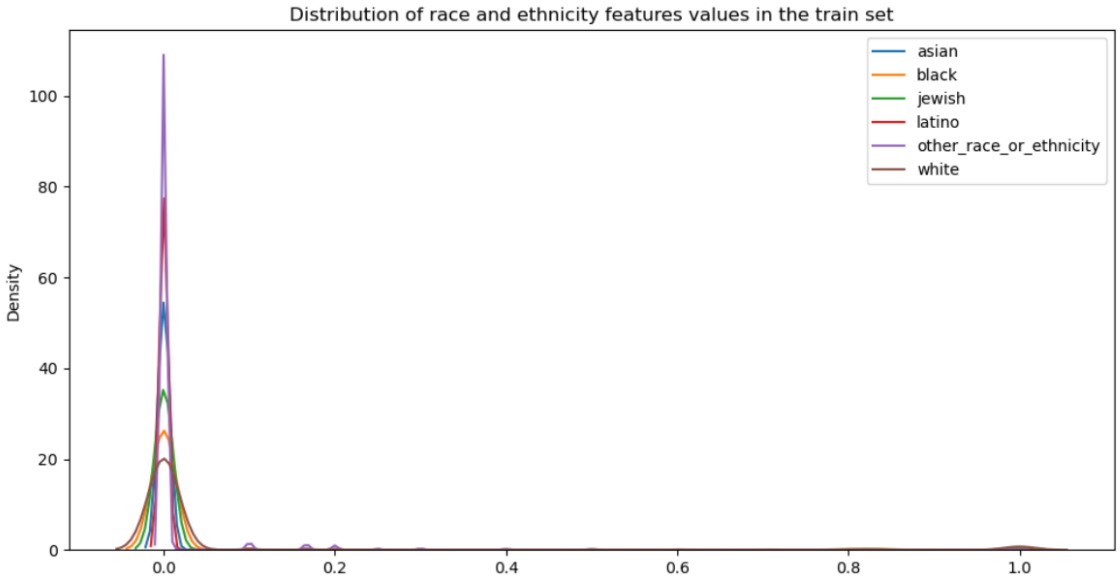


Figure 8: Distribution of race and ethnicity features values in the train dataset

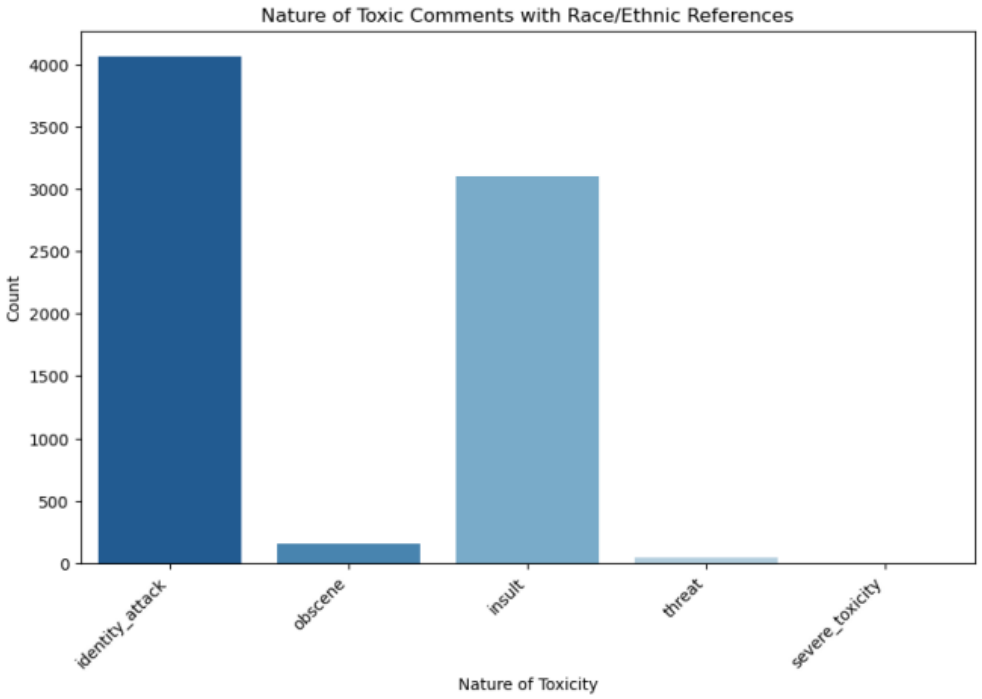


Figure 9: Nature of Toxic Comments with Race/Ethnic References

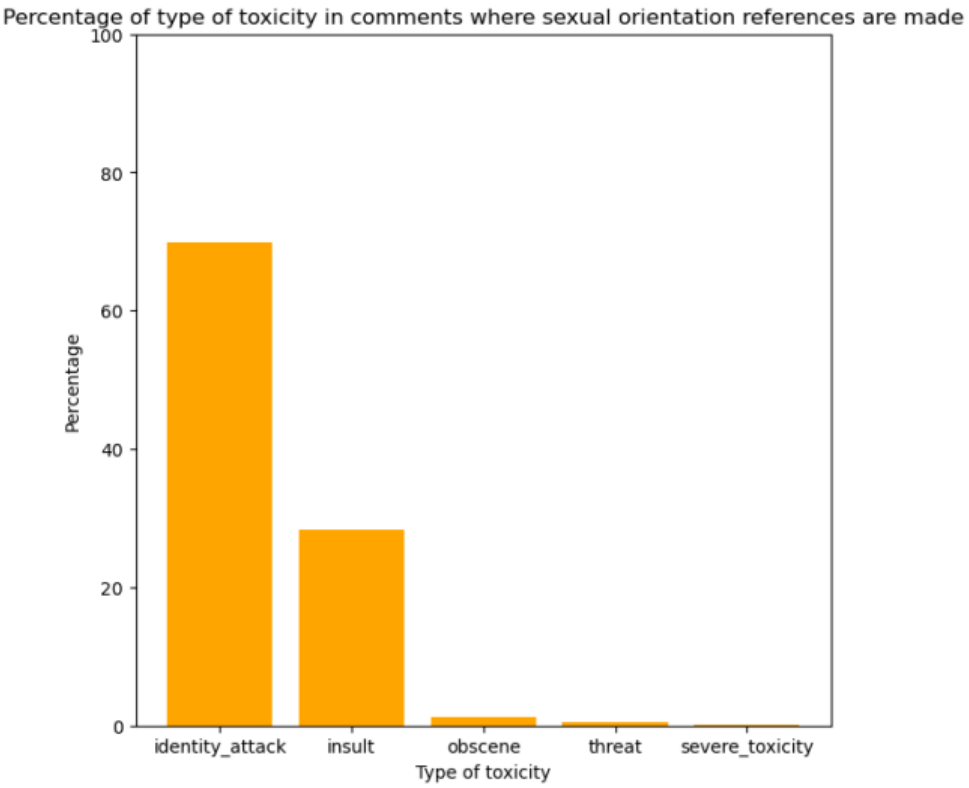


Figure 10: Percentage of type of toxicity in comments where sexual orientation references are made

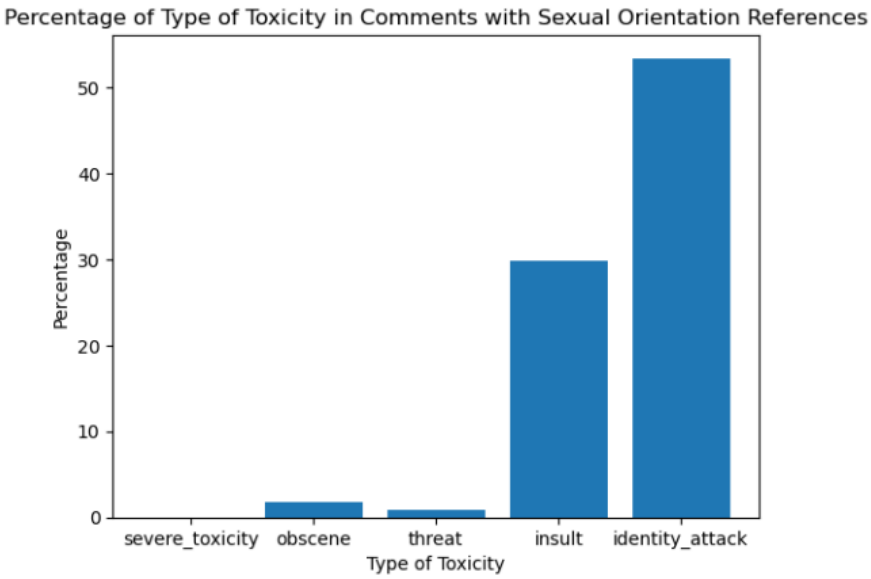


Figure 11: Percentage of Type of Toxicity in Comments with Sexual Orientation References

Overall, the analysis of identity attributes and their relation to toxic comments provided valuable insights into the nature of toxicity present in the train dataset.

3.4. Features generated by users' feedback

The features funny, sad, wow, likes, and disagree are generated by users' feedback and provide information about how users perceive and respond to a particular comment. We did an analysis of these features over the training dataset. These features generated by users' feedback provide important contextual information that can help in identifying and understanding the toxicity of comments.

4. Methodology

In our project, we have addressed the problem of classifying a given text as toxic or non-toxic using a regression algorithm, where any value less than 0.5 is non-toxic and a value greater than 0.5 is toxic.

To extract features from the text data, we have used two feature extraction techniques Bag of Words and Term Frequency-Inverse Document Frequency (TFIDF). We have used "snowballstemmer" function from the Natural Language Toolkit (NLTK) to prepare data for feature extraction. We removed non-alphanumeric characters, converted all text into lowercase, and removed common stop words. Then we used 'train_test_split()' from "sklearn.model_selection" (scikit-learn library) to create training and cross-validation data and then those files were saved, for re-usability to ensure consistent results.

For Regression, we used two algorithms: Stochastic Gradient Descent (SDG) Regressor and Decision Tree. We trained these models using pre-processed data and evaluated their performance on a cross-validation set. We used the mean squared error (MSE) as an evaluation metric.

4.1 Bag of words

Bag of words is a feature extraction technique used in Natural Language Processing. In our approach, we specified the ngram range from 1 to 2, which means that both single words and bigrams will be included in the vocabulary. We limited the maximum number of features to 30,000.

4.1.1 SGD regressor

In this step, we used the SGD Regressor algorithm to train our model. To tune hyperparameters we looped through different values of alpha and penalty. Alpha controls the regularization and the penalty specify the regularization penalty to be used (L1 or L2).

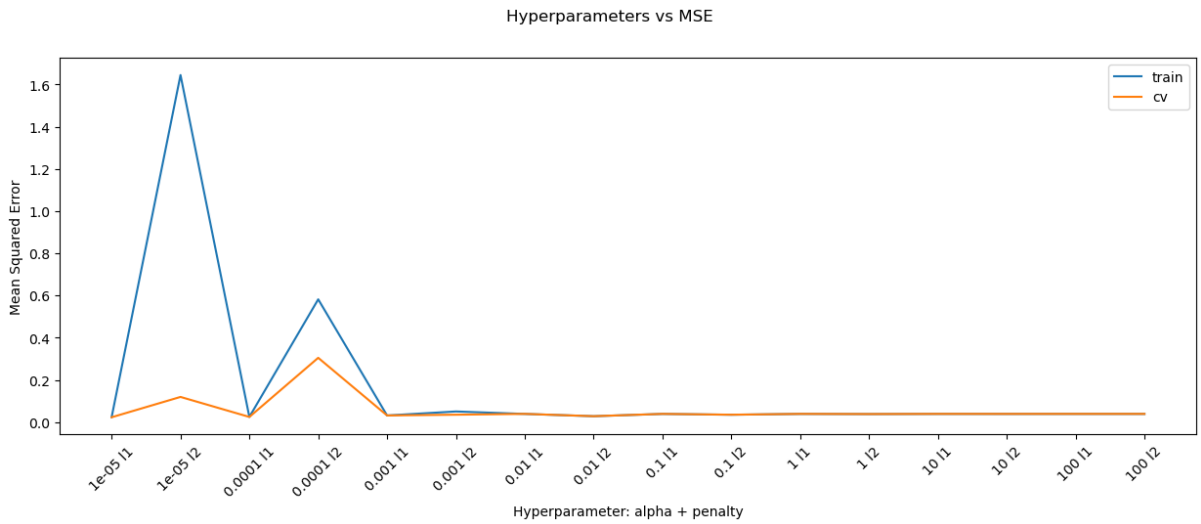


Figure12: Hyperparameters vs MSE

In our model, the alpha value of 0.00001, and penalty L1 have the lowest MSE value of 0.022995330405120723. The MSE value is calculated for each loop, and as the MSE value increases the model becomes less effective.

	weights
stupid stupid	-0.130182
knee jerk	-0.075628
black white	-0.049673
fool peopl	-0.047908
ignor fact	-0.037218

Figure13: MSE values

Then we printed the top 20 words which contribute to a comment being toxic. If we have words like stupid, jerk, and dumb in our comments, then that comments have a high chance of being toxic.

4.1.2 Decision tree

In the Decision tree model, we tuned our model on 2 hyperparameters max depth and min samples. For these hyperparameters, we used three different values 3, 5, and 7 for Max depth and 10, 100, and 1000 form min samples. We looped through these different values, and MSE value was calculated at every loop.

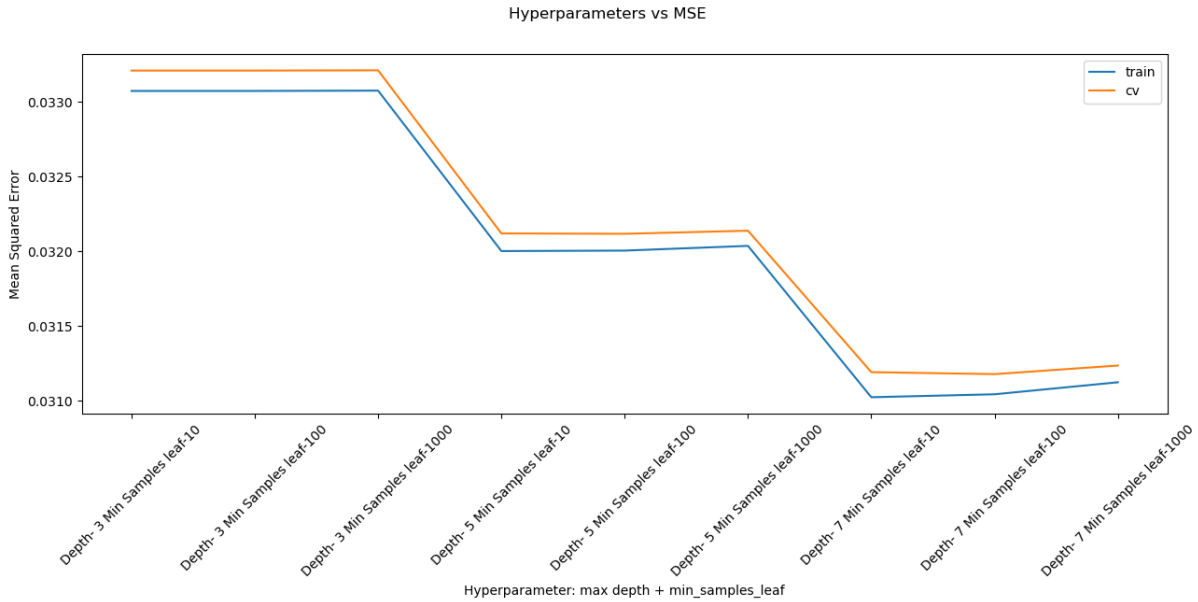


Figure14: Hyperparameters vs MSE

After analysing the output, we found the best model with a Max depth of 7 and min sample of 100 had the lowest MSE value, which is 0.031174175 0.031177265480264175. MSE value is calculated on a cross-validation set.

	weights
stupid	0.399523
idiot	0.264382
pathet	0.068405
fool	0.067381
moron	0.063471

Figure15: MSE values

Words like ‘Stupid’, ‘Idiot’, and ‘Fool’ are often associated with the toxic comment. Higher the weight, the more important the feature is for target variable.

4.2 Term Frequency-Inverse Document Frequency (TFIDF)

TFIDF stands for Term frequency-inverse document frequency is a very famous feature extraction technique used in natural language processing. It calculates the frequency of each word in a document, and the number of documents that contains that word. This gives more importance to rare words that occur in very few documents. This approach is best for large datasets.

4.2.1 SDR regressor with Hyper-parameter

We used the values of the same hyperparameters for a penalty and for an alpha, we used values of 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, and 100.

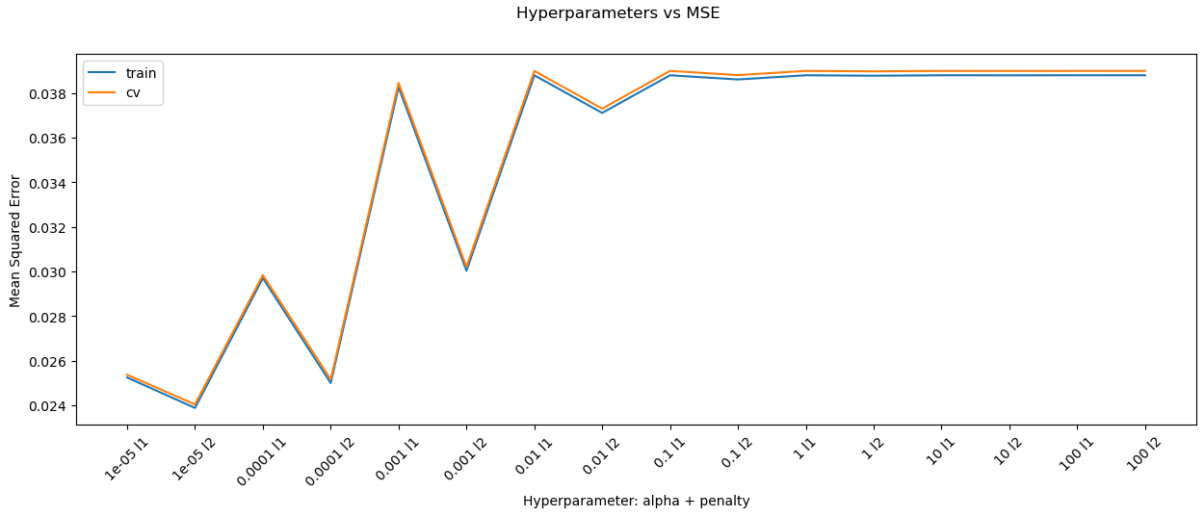


Figure16: Hyperparameter vs MSE

We got the best MSE value with alpha equal to 0.00001 and penalty equal to L2, which was 0.02404554702921834.

	weights		weights
stupid	1.569629	**thank**	-0.092360
idiot	1.266662	**interest**	-0.087282
fool	0.657220	**agre**	-0.077450
ignor	0.605213	**stori**	-0.077260
dumb	0.592775	**great**	-0.071112

Figure17: MSE values

Words with positive weights are more likely to be associated with Toxic comments like ‘stupid’, ‘idiot’, and ‘fool’. Words with negative weights are associated with non-toxic comments like ‘thank’, ‘interest’, and ‘great’.

4.2.2 Decision tree with Hyper-parameter

We used the same hyperparameters values for decision trees as we used in the Bag of Words earlier.

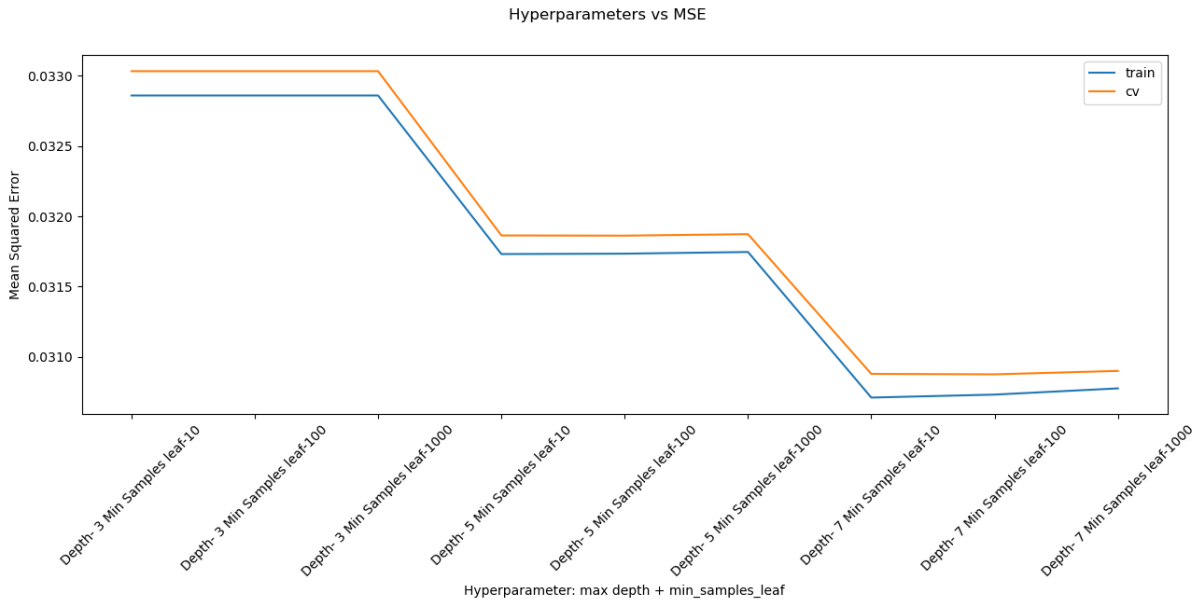


Figure18: Hyperparameter vs MSE

After tuning the hyperparameters, we found that the best MSE value was 0.030876105245497155 for a max depth of 7 and a min sample of 100.

	weights
stupid	0.407776
idiot	0.267245
fool	0.071524
pathet	0.069263
moron	0.064115

Figure19: MSE values

Our model accurately classifies toxic and non-toxic comments. Comments containing ‘stupid’, ‘idiot’, and ‘fool’ are more likely to be a toxic comment.

5. Output

The model is trained for 5 epochs, and the history object is used to store the loss and mean squared error values for both the training and validation data. These values are then used to plot the loss curves for both the training and validation data using matplotlib.

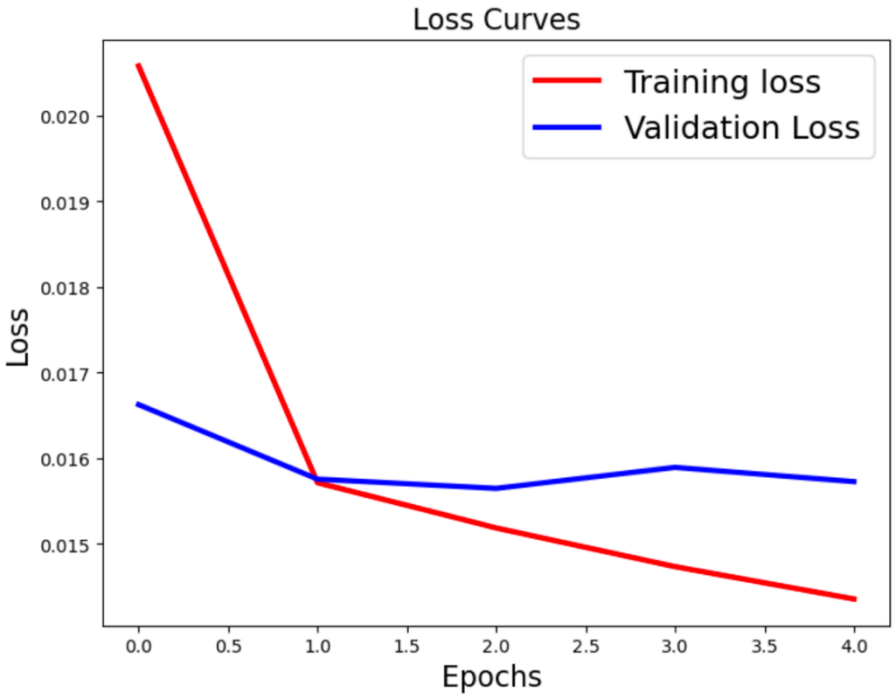


Figure20: Loss Curves

6. Challenges

- Data bias: One of the biggest challenges in building a Toxicity Classification Model is ensuring that the model is not biased towards any particular group or community. This can be particularly difficult when dealing with sensitive topics or controversial issues. To mitigate data bias, it is crucial to use diverse and inclusive datasets when training a Toxicity Classification Model. This can help ensure that the model is not biased towards any particular group or community.
- Contextual understanding: Toxic language can be expressed in various forms, including sarcasm, irony, or even humor. It can be challenging for a model to understand the context and tone of a message, which can lead to misclassification. Incorporating contextual analysis techniques can help the model better understand the tone and context of a message, and avoid misclassification.

- During the building process, we updated our existing installed packages resulting in changes in either function call or function/library location. Eg: “from keras.utils import Sequence changed to “from tensorflow.keras.preprocessing import sequence”. This function was essential for our model execution.

7. Limitations and Future Scope

1. False Positives: Toxicity Classification Models can produce false positives, flagging messages as toxic when they are not. This can lead to censorship and limit free speech. Future research should focus on reducing false positives while maintaining a high level of accuracy.

2. Limited Understanding of Nuances: Toxicity Classification Models are limited in their ability to understand the nuances of language. Future research should explore ways to incorporate more sophisticated language models that can better understand the context and meaning of messages.

3. Continuous Learning: Toxicity Classification Models require constant updating and training to keep up with evolving language and social norms. Future research should focus on developing models that can continuously learn from new data to improve their accuracy and reduce bias.

In conclusion, while Toxicity Classification Models have the potential to make a significant impact on improving online discourse, there are still many challenges to be addressed. Overcoming these challenges will require ongoing research and development efforts to ensure that these models are effective, fair, and promote a safer online environment.

8. References

- https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDRegressor.html
- https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/
- https://www.geeksforgeeks.org/decision-tree/
- https://www.geeksforgeeks.org/decision-tree-implementation-python/
- https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data
- https://github.com/hemangsharma/Toxicity-Classsification-Model
- https://drive.google.com/drive/folders/1WMY6VMZ81LD2oblBI1MVtAup4M2I8xX7?usp=share_link

Individual Contribution Report

My Contributions

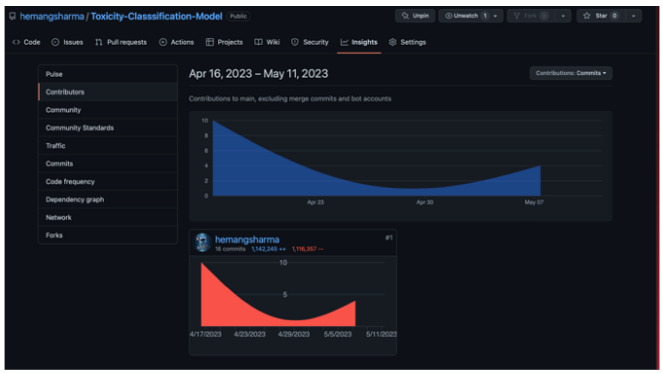
- I came up with the idea for the project, to build a model to detect toxicity in comments.
- I helped Nusrat in finding the dataset for our project.
- I helped Rajveer in EDA tasks.
- I was responsible for the machine learning models.
- I also contributed to report writing.

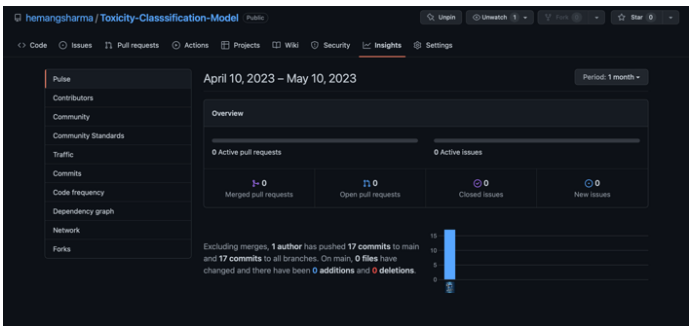
Challenges that I Faced (on individual level)

- TensorFlow library does not have native GPU support for “M-series” powered MacBooks, we had to find a work around to use it. we had to install “tensorflow-metal” as a plugin.
- Jupyter Notebook was unable to load and import TensorFlow library, so we switched to VS Code for programming.
- Data set exceed the GitHub file upload limit to had to rely on Google drive for data set sharing.

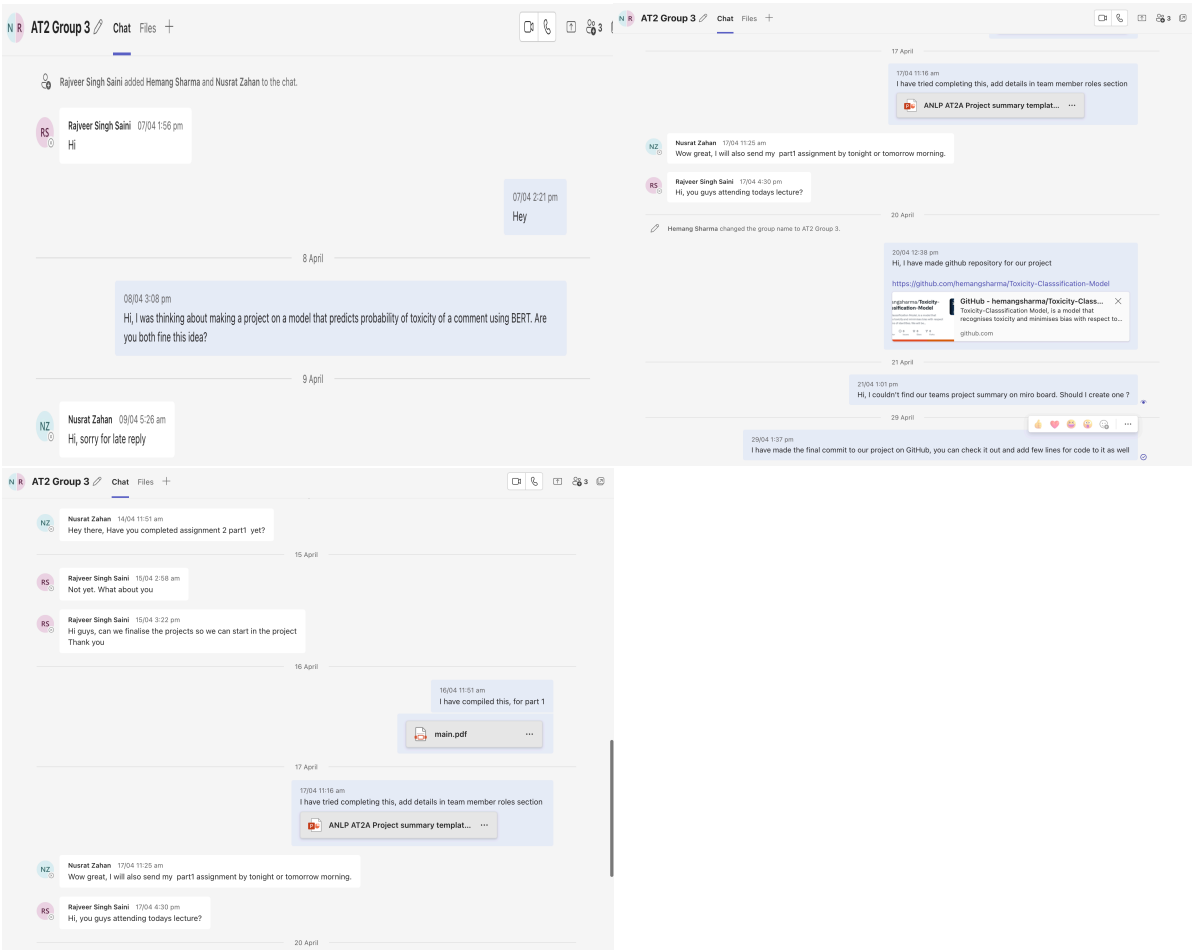
Contributions Proofs

1. GitHub:





2. Microsoft Teams:



3. WhatsApp: