PROJECT REPORT

AMOD 5610- Big Data Research Project

TOXIC COMMENT CLASSIFICATION

- Putting the 'Social' back in Social Media

SUBMITTED BY:

DIXIT KAMAL

SATISH MAHABHASHYAM

UDIT SHARMA

TABLE OF CONTENTS

INTRODUCTION

Social media can have significant emotional impacts on humans, both positive and negative. On the positive side, social media platforms offer opportunities for social connection, support, and self-expression. They allow people to stay connected with friends and family, share personal experiences, and receive social validation through likes, comments, and shares, leading to positive emotions such as joy, excitement, and a sense of belonging.

However, social media also has negative emotional impacts that cannot be ignored. One of the most prominent negative impacts is the phenomenon of social comparison. Social media platforms are filled with others' highlight reels and curated lives, and individuals tend to compare themselves to others, often leading to feelings of inadequacy, jealousy, and low self-esteem. This constant comparison can create unrealistic expectations and fuel a fear of missing out, resulting in anxiety and even depression.

Another emotional impact of social media is cyberbullying. The anonymity and distance provided by social media can embolden individuals to engage in online harassment, bullying, and hate speech, which can cause profound emotional distress, including anxiety, depression, and even suicidal ideation. Cyberbullying has become a significant concern, as it can have serious consequences for the mental health and well-being of individuals.

In addition, social media can contribute to addiction and compulsive behavior. Many individuals spend excessive time on social media, neglecting real-life relationships and experiencing withdrawal symptoms when not using social media. This can lead to feelings of isolation, loneliness, and a loss of productivity, which can negatively impact mental health.

Toxic comments are becoming increasingly prevalent in online communities, and their negative impact on individuals and communities is a growing concern. The rapid expansion of social media has facilitated the dissemination of toxic content, such as hate speech, cyberbullying, and malicious attacks. This antisocial behavior takes a toll on individuals' mental and emotional well-being and can contribute to a hostile online environment. As a result, there is a pressing need for automated methods to detect and identify such behavior in online discussions.

To address this need, this project aims to develop a model for toxic comment classification using Natural Language Processing (NLP) techniques. The project aims to automatically classify toxic comments and determine the type of toxicity exhibited in the comments. The model will be trained on a dataset of comments and will categorize them as toxic, severely toxic, obscene, threatening, insulting, or exhibiting identity hate.

Traditionally, online communities have relied on human moderators to monitor and remove toxic content, but this approach is time-consuming and often ineffective due to the overwhelming volume of online content. Machine learning algorithms such as Logistic Regression, Random Forest, Neural Networks, and NLP techniques such as Bag-of-words, TF-IDF, and Word embeddings will be used in this project. These techniques enable the automatic analysis of large amounts of text data to identify patterns and features associated with toxic content.

The parameters for these techniques will be determined through experimentation and cross-validation to identify the best-performing models. The project will involve a comprehensive analysis of existing methods and approaches to toxic comment classification, to propose new, more effective, and efficient methods for identifying toxic comments and mitigating their impact.

In conclusion, this project addresses the need for automated methods to detect and identify toxic comments in online discussions. The classification of toxic comments is a complex task, and there is a lack of research and understanding in this area. By using machine learning algorithms and NLP techniques, this project aims to contribute to the development of more effective and efficient methods for identifying toxic comments and mitigating their impact on individuals and online communities. With the growing prevalence of toxic content on social media, it is imperative to address this issue and create a safer and healthier online environment for everyone.

LITERATURE REVIEW

The field of natural language processing (NLP) and machine learning has made significant advancements in recent years, particularly in the area of toxic comment classification, as the prevalence of online social media platforms has increased and concerns about the spread of toxic and hate speech on these platforms have grown. This literature review aims to provide an overview of the most recent and relevant academic work in this field, focusing on understanding the various algorithms, techniques, and models proposed for toxic comment classification, and laying the foundation for the development of our own solution to this problem.

Toxic comment classification is a multidisciplinary field that spans NLP, machine learning, and deep learning. NLP techniques for text representation, such as bag of words, n-grams, and TF-IDF, have been widely used in toxic comment classification. These representations serve as inputs to machine learning algorithms, such as support vector machines, decision trees, and random forests, which are trained on annotated data to predict the class of a given comment. Deep learning architectures, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), have also been utilized in toxic comment classification, as they are capable of learning complex representations of text data, including the relationships between words and their meaning in a given context. Word embeddings, such as word2vec and GloVe, which map words to dense vector representations, have gained popularity in the field, as they capture semantic and syntactic relationships between words, making them useful for tasks such as sentiment analysis and toxic comment detection.

One of the challenges in toxic comment classification is the wide range of toxic behavior, including hate speech, cyberbullying, insults, and threats, which requires the use of various techniques, including both deep learning models, such as CNNs and RNNs, as well as traditional machine learning algorithms, such as SVMs and random forests. However, another challenge is dealing with imbalanced datasets, where the number of instances of one class far outweighs the number of instances of another class, leading to biased models. Techniques such as synthetic minority over-sampling (SMOTE) and cost-sensitive learning have been proposed to address this issue.

Recent research has explored the use of deep learning models with attention mechanisms, such as recurrent neural networks with attention, and ensemble methods that combine multiple models to improve classification performance. Attention mechanisms have been shown to be effective in identifying important words or phrases in a comment that contribute to its toxicity, and ensemble methods have been shown to improve the robustness and generalization of classification models.

Another area of active research in toxic comment classification is the use of transfer learning, where models trained on one task or dataset are used to improve the performance of models on another task or dataset. For example, pre-trained language models, such as BERT and GPT, have been fine-tuned for toxic comment classification and have shown promising results in improving the performance of classification models. Transfer learning has the potential to leverage knowledge learned from related tasks or datasets to improve the performance of toxic comment classification models, particularly in scenarios where annotated data is scarce or unavailable.

Despite the advancements in toxic comment classification, there are still challenges that need to be addressed. One challenge is the interpretability of the models, as deep learning models, such as CNNs and RNNs, are often considered black boxes, making it difficult to understand how and why they make certain predictions. This can be a concern in real-world applications where transparency and accountability are important. Interpretability techniques, such as attention visualization and feature importance analysis, have been proposed to address this issue.

One more challenge is the ethical implications of toxic comment classification. The issue of bias in machine learning models, which can lead to unfair or discriminatory outcomes, is a significant concern in the context of toxic comment classification. Models trained on biased data can lead to biased predictions, reinforcing existing social biases and exacerbating issues related to discrimination and unfairness. For example, if the training data used to train a toxic comment classifier is biased towards a certain group, such as a particular race, religion, or gender, the model may be more likely to classify comments from that group as toxic, while overlooking toxic comments from other groups. This can result in discriminatory outcomes and contribute to the amplification of hate speech against certain communities.

Addressing bias in toxic comment classification models is a complex and important challenge. Researchers have proposed various techniques to mitigate bias, such as re-sampling techniques to balance the distribution of different classes, adversarial training to reduce bias in model predictions, and fairness-aware machine learning algorithms that explicitly account for fairness during model training. Additionally, it is crucial to ensure that the training data used to train these models is diverse, representative, and free from biases.

Another challenge in toxic comment classification is the constantly evolving nature of language and the emergence of new forms of toxic behavior. Toxic comments can take on different forms, including hate speech, cyberbullying, insults, and threats, and can manifest in different languages, dialects, and cultural contexts. Keeping up with these evolving forms of toxic behavior requires continuous updates and adaptations of classification models to ensure their effectiveness. Additionally, the dynamic nature of online social media platforms, where new comments are constantly being posted, poses a challenge in real-time monitoring and classifying toxic comments.

Despite these challenges, toxic comment classification has significant societal implications. Identifying and mitigating toxic behavior on social media platforms can contribute to creating a safer online environment, reducing the spread of hate speech and harmful content, and promoting healthy and respectful discourse. Toxic comment classification can also be used in applications beyond social media, such as content moderation, online community management, and early warning systems for identifying potential threats or harmful behavior.

The literature review for this project involved a thorough examination of existing research on toxic comment classification, natural language processing (NLP), and machine learning. Several key research papers were reviewed, including "Toxic comment classification" by Zaheri, Leath, and Stroud, retrieved from SMU Scholar, and "Toxic comment classification using Convolutional Neural Network" by Samyal, Shinde, Singh, and Gidwani, retrieved from JETIR. These papers provided valuable insights into the various preprocessing techniques, algorithms, and deep learning architectures that can be employed for classifying toxic comments.

One of the major challenges highlighted in the literature review of those two papers is the difficulty of accurately identifying toxic comments. Toxicity can manifest in different forms, such as hate speech, cyberbullying, insults, and threats, and can be expressed in diverse languages, dialects, and cultural contexts. This makes it challenging to develop a one-size-fits-all solution for toxic comment classification. The literature review emphasized the importance of carefully selecting and implementing appropriate preprocessing techniques, feature representations, and model architectures to improve the accuracy and effectiveness of toxic comment classification.

Another challenge identified in the literature is the need for large annotated datasets for training robust and accurate toxic comment classifiers. Annotated datasets are crucial for supervised machine learning algorithms to learn from labeled examples and make accurate predictions. However, obtaining large and diverse annotated datasets for toxic comments can be challenging due to the sensitive and controversial nature of the content. The literature review highlighted the importance of carefully curating annotated datasets that are representative of the diverse types of toxic comments encountered in real-world scenarios.

The literature review also highlighted the challenge of dealing with the imbalance in the distribution of toxic and non-toxic comments in training data. Toxic comments are often a minority class compared to non-toxic comments, which can lead to imbalanced datasets that result in biased models. This bias can affect the accuracy and fairness of toxic comment classification models. The literature review discussed various techniques, such as re-sampling methods and cost-sensitive learning, that can be used to address the issue of imbalanced datasets and improve the performance of toxic comment classifiers.

The findings of the literature review have laid a strong foundation for the implementation of the toxic comment classification model in this project. The insights gained from the reviewed research papers will be used to guide the development of the solution approach, including the selection of appropriate preprocessing techniques, feature representations, and model architectures. The challenges identified in the literature review will also be addressed in the project, such as carefully curating annotated datasets, handling imbalanced data, and mitigating bias in model predictions.

In conclusion, the field of toxic comment classification has witnessed significant advancements in recent years, driven by the need to address the spread of toxic and hate speech on online social media platforms. Academic work in this field spans multiple disciplines, including NLP, machine learning, and deep learning, and has explored various techniques and models for effectively identifying and classifying toxic comments. Challenges in this field include dealing with the wide range of toxic behavior, addressing imbalanced datasets, handling bias in models, adapting to the evolving nature of language and toxic behavior, and considering the ethical implications of toxic comment classification. Future research in this area should continue to focus on developing robust and fair models, addressing bias and ethical concerns, and adapting to the dynamic nature of online platforms to create safer and more inclusive online environments.

DATA SET DESCRIPTION

In today's digital landscape, the widespread occurrence of digital abuse and harassment can hinder our ability to engage in constructive conversations about important issues. Many individuals may feel discouraged from participating in discussions and expressing their thoughts due to these challenges, resulting in communities implementing restrictions or bans on user comments. To address this pressing issue, the Conversation AI team, a research project backed by Jigsaw and Google, is dedicated to developing tools that can enhance online discourse. The project places a strong emphasis on researching harmful online behaviors, including negative remarks, with the goal of improving the quality of online conversations.

The "Jigsaw Toxic Comment Classification" dataset consists of a train.csv file and a test.csv file which provide different data for the task of toxic comment classification. The files included in the dataset are as follows:

1. The train.csv file contains comments along with their binary labels, indicating whether a comment is toxic (1) or non-toxic (0). This file is used to train machine learning models for toxic comment classification. It has a total of **159,571** comments.
2. The test.csv file contains comments for which the toxicity probabilities need to be predicted. It is important to note that some of the comments in this file are not included in scoring to prevent hand labeling. It has a total of **153,164** comments.

The “train.csv” file contains a collection of text comments from the online forum, with labels identifying the comment's level of toxicity (toxic, severe toxic, obscene, threat, insult, or identity hate). The files consist of the following columns:

1. id: a unique identifier for each comment
2. comment_text: the text of the comment
3. Toxic: The toxicity of comments
4. Severe_toxic: Intensity of toxicity
5. Obscene
6. Threat: Involves threats in the comment
7. Insult
8. identity_hate

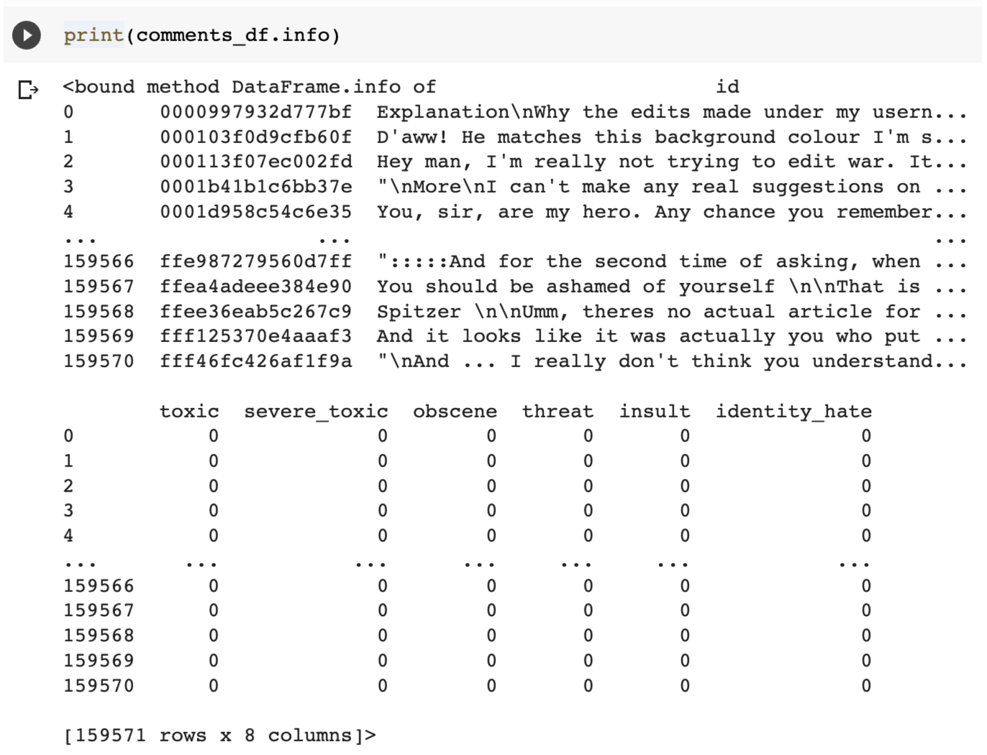
The “test.csv” file contains a collection of text comments from the online forum. The files consist of the following columns:

1. id: a unique identifier for each comment
2. comment_text: the text of the comment

The goal is to create a machine-learning model that can correctly categorize the comments into these groups. Wikipedia chat page modifications served as the data source for the dataset, which was made available by Kaggle.

Exploratory Data Analysis

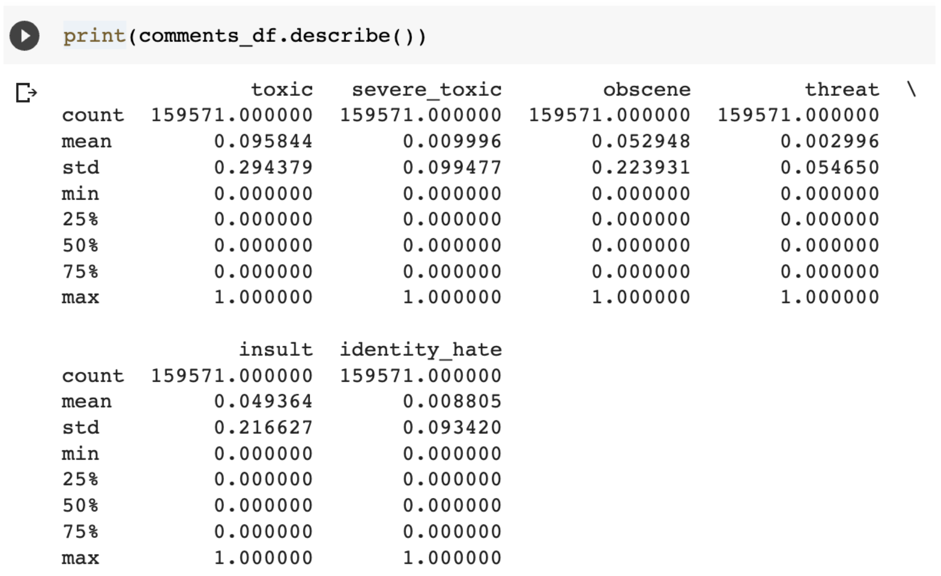
1. Here is some information about the dataset using pandas .info().



```
print(comments_df.info)
```

```
<bound method DataFrame.info of                         id
0       0000997932d777bf  Explanation\nWhy the edits made under my usern...
1       000103f0d9cfb60f  D'aww! He matches this background colour I'm s...
2       000113f07ec002fd  Hey man, I'm really not trying to edit war. It...
3       0001b41b1c6bb37e  "\nMore\nI can't make any real suggestions on ...
4       0001d958c54c6e35  You, sir, are my hero. Any chance you remember...
...                  ...                                                ...
159566  ffe987279560d7ff  ":::::And for the second time of asking, when ...
159567  ffea4adeee384e90  You should be ashamed of yourself \n\nThat is ...
159568  ffee36eab5c267c9  Spitzer \n\nUmm, theres no actual article for ...
159569  fff125370e4aaaf3  And it looks like it was actually you who put ...
159570  fff46fc426af1f9a  "\nAnd ... I really don't think you understand...

        toxic  severe_toxic  obscene  threat  insult  identity_hate
0           0             0        0       0       0              0
1           0             0        0       0       0              0
2           0             0        0       0       0              0
3           0             0        0       0       0              0
4           0             0        0       0       0              0
...       ...           ...      ...     ...     ...            ...
159566      0             0        0       0       0              0
159567      0             0        0       0       0              0
159568      0             0        0       0       0              0
159569      0             0        0       0       0              0
159570      0             0        0       0       0              0

[159571 rows x 8 columns]>
```

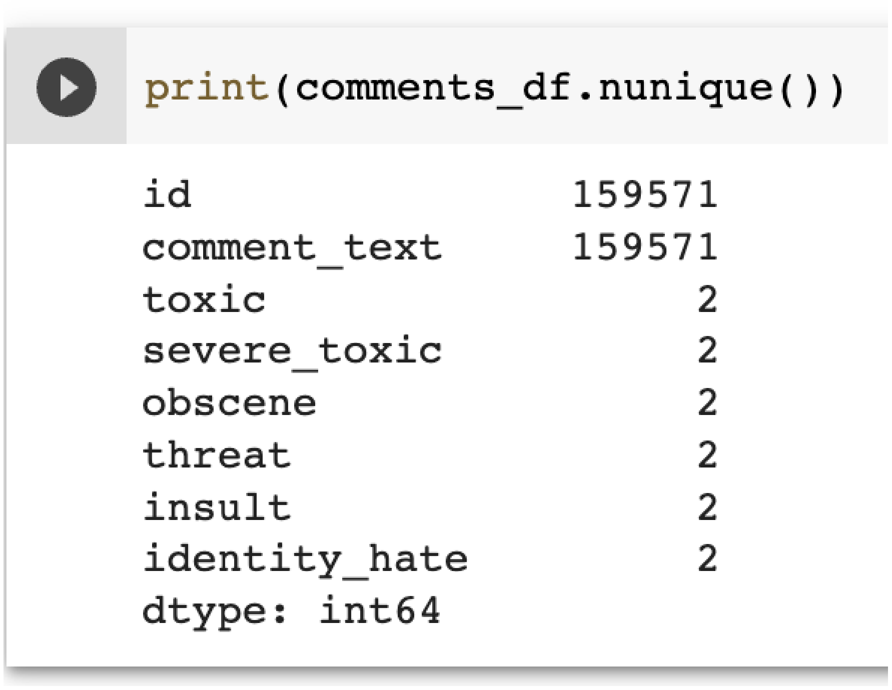
2. Here is some statistical information about the attributes of the dataset using describe().



```
print(comments_df.describe())
```

```
               toxic   severe_toxic        obscene         threat  \
count  159571.000000  159571.000000  159571.000000  159571.000000
mean        0.095844       0.009996       0.052948       0.002996
std         0.294379       0.099477       0.223931       0.054650
min         0.000000       0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       0.000000
75%         0.000000       0.000000       0.000000       0.000000
max         1.000000       1.000000       1.000000       1.000000

              insult  identity_hate
count  159571.000000  159571.000000
mean        0.049364       0.008805
std         0.216627       0.093420
min         0.000000       0.000000
25%         0.000000       0.000000
50%         0.000000       0.000000
75%         0.000000       0.000000
max         1.000000       1.000000
```

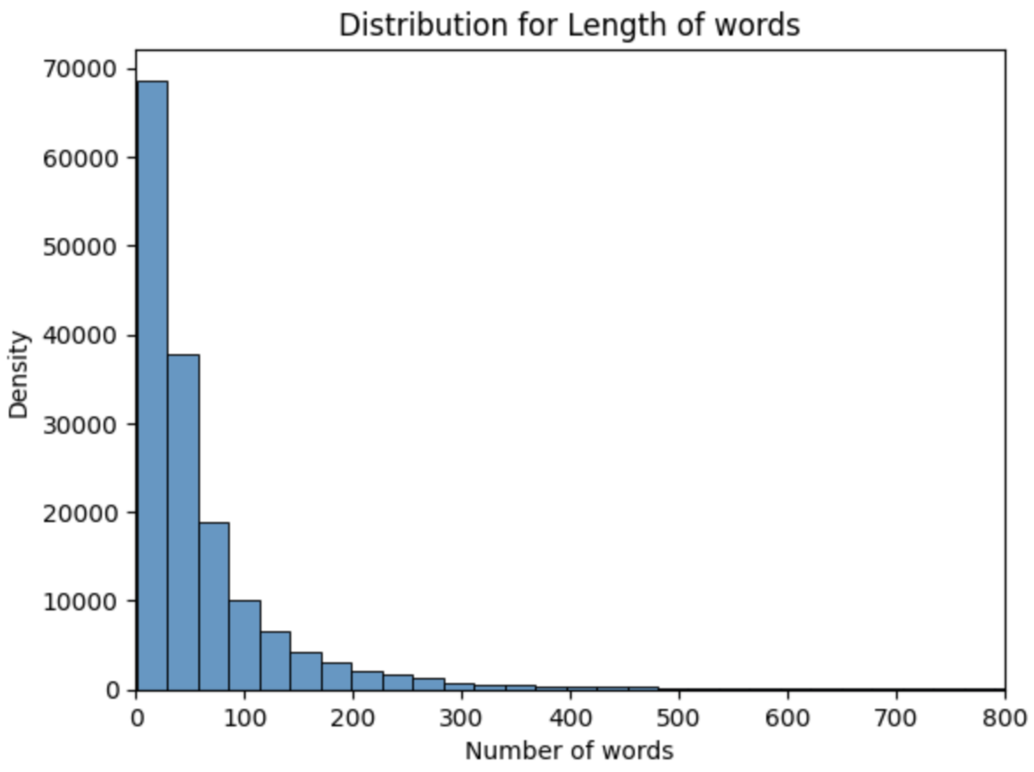
3. The dataset has a few unique values.



```
print(comments_df.nunique())
```

```
id               159571
comment_text     159571
toxic                 2
severe_toxic          2
obscene               2
threat                2
insult                2
identity_hate         2
dtype: int64
```

4. The distribution of length of words is as follows:



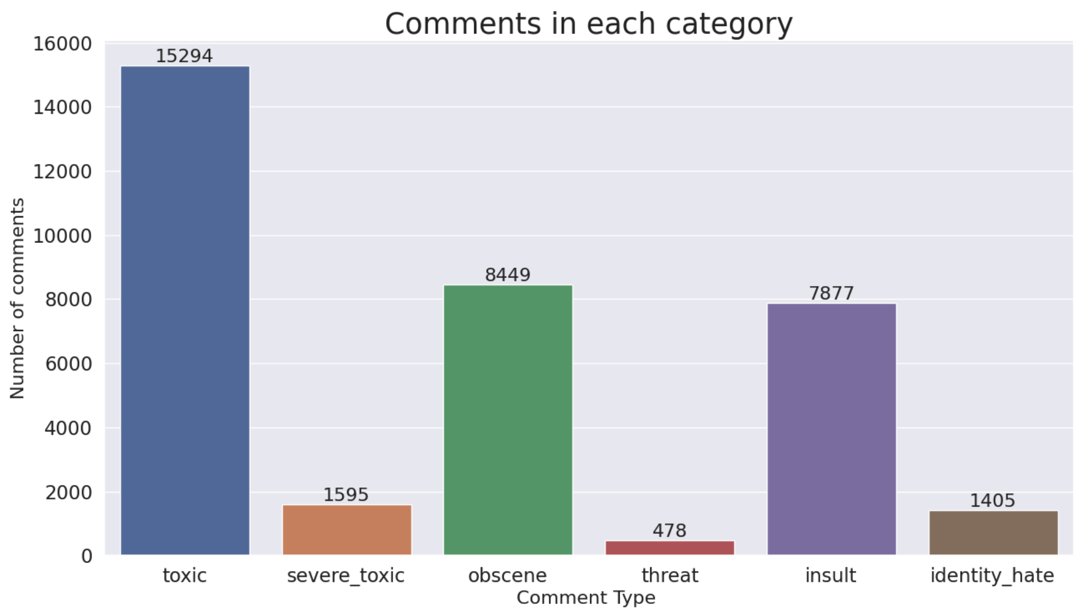
The plot indicates that the dataset had more than 20,000 comments (approximately) which had more than 100 words.

5. Upon analysis of the dataset, we found that, out of 159,571 comments, 143,346 comments were clean and 16,225 were labeled in one of the categories.

```
Total number of comments =  159571
Number of clean comments =  143346
Number of comments with labels = 16225
```

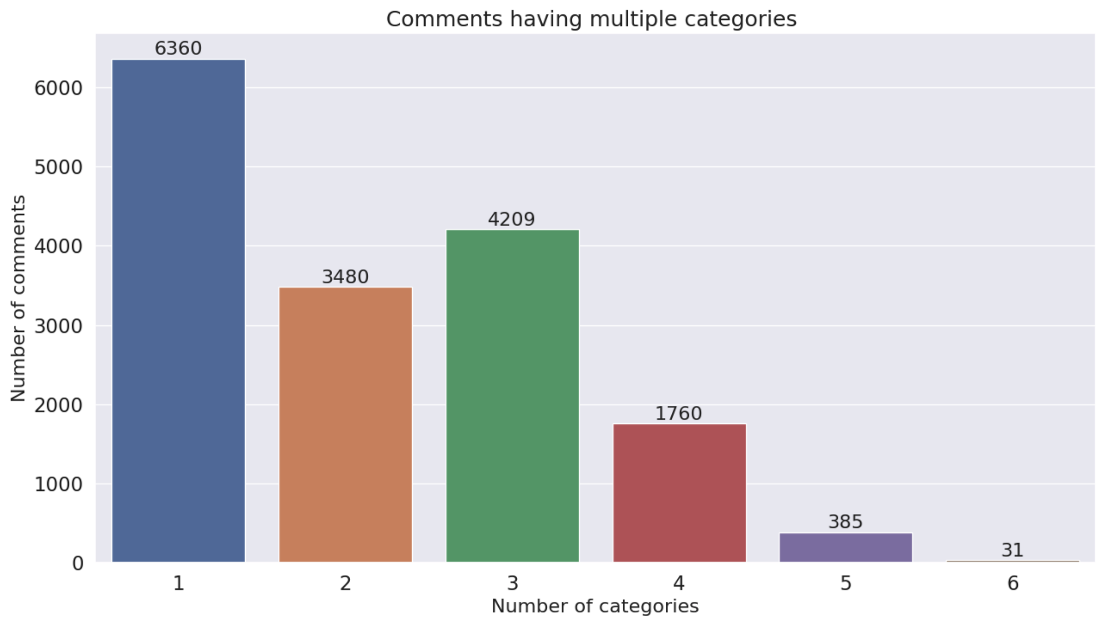
6. Upon analyzing the dataset, the number of comments in each category was found to be as below.

```
        category  number of comments
0          toxic               15294
1   severe_toxic                1595
2        obscene                8449
3         threat                 478
4         insult                7877
5  identity_hate                1405
6          clean              143346
```



From the above plot, we can observe that the labels toxic, obscene, and insult were more common than others.

7. Here is the plot for labeled comments having multiple labels.



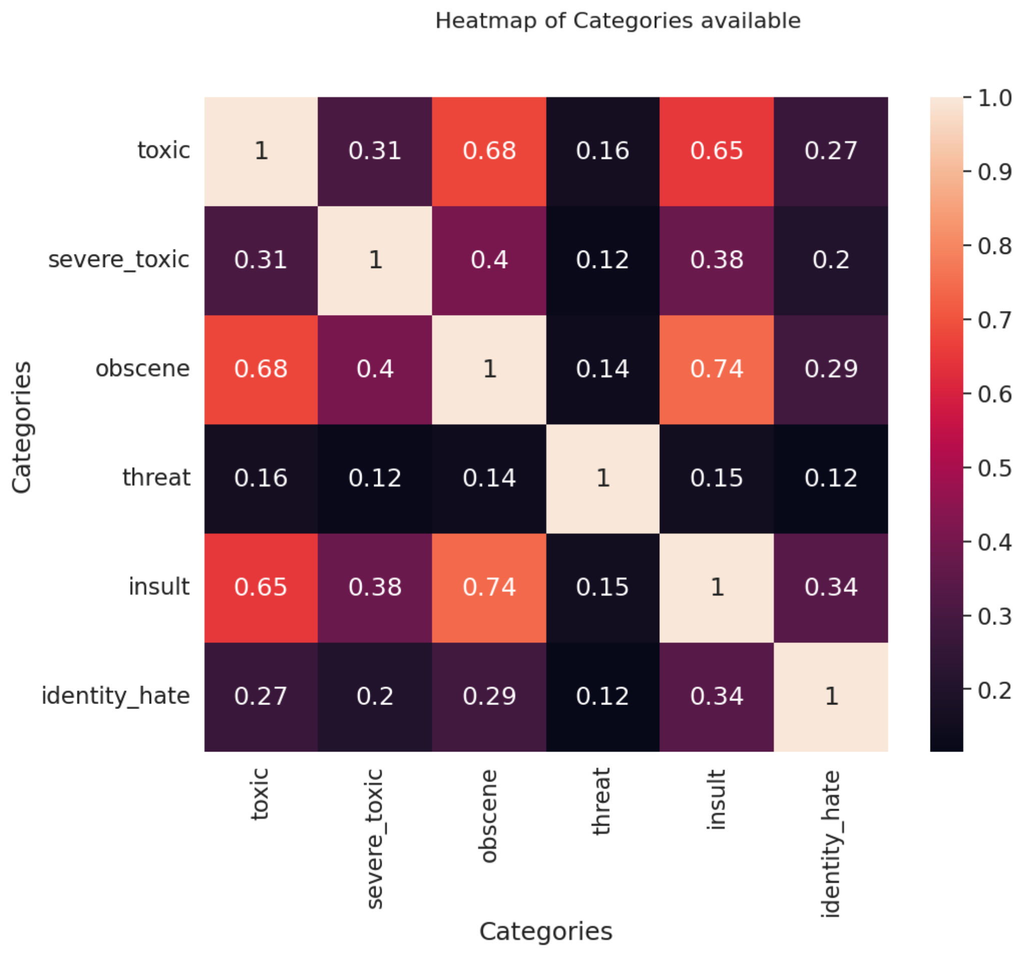
From the above plot, we can make these observations:

a. 40% of the labeled comments were under a single category
b. The number of comments labeled under three categories was more than the comments labeled under two categories.
c. Out of the 16,225 labeled comments, **31 comments** were labeled under **all six categories.**

8. Below is the heatmap for various categories.

From the below heatmap, we can make the following observations

a. The correlation between the categories **obscene** and **insult** is **strong**. This means that, when a comment is labeled as an insult, there is a high chance that the comment will also be labeled as obscene.
b. The correlation between the categories **identity_hate** and **threat** was very **weak**. This means that if a comment is labeled under one of these categories, there is less chance for it to be labeled under the other category as well.


Heatmap of Categories available
toxic 1 0.31 0.68 0.16 0.65 0.27
severe_toxic 0.31 1 0.4 0.12 0.38 0.2
obscene 0.68 0.4 1 0.14 0.74 0.29
threat 0.16 0.12 0.14 1 0.15 0.12
insult 0.65 0.38 0.74 0.15 1 0.34
identity_hate 0.27 0.2 0.29 0.12 0.34 1
toxic
severe_toxic
obscene
threat
insult
identity_hate
Categories
Categories
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2

METHODOLOGY

Data Preprocessing & Cleaning: In order to ensure the accuracy and reliability of the data, data preprocessing and cleaning are essential steps in the toxic comment classification project. To prepare the data for toxic comment classification, common techniques include removing irrelevant information, handling duplicate rows, null values, URLs, usernames, emails, HTML tags, punctuation, special characters, digits, stopwords, and lemmatizing text. The performance and dependability of machine learning models for classifying toxic comments are improved by using these techniques, which also help to reduce noise and improve data consistency. Accurate and useful insights must be extracted from the data through proper data cleaning and preprocessing.

In this section, we'll go over a number of typical techniques for cleaning and preprocessing data that are relevant to the classification of toxic comments.

- **Getting Rid of Insignificant Data:** In order to reduce noise and needless computational overhead, the dataset should be cleaned up of any irrelevant information, such as columns or features that aren't relevant to the classification task. For instance, to simplify the dataset, non-discriminatory metadata such as id can be removed.
- **Duplicate Rows:** The dataset's duplicate rows may cause the model's performance to be off, which could result in skewed findings. To make sure that each comment is represented in the dataset only once and prevent duplication of training examples, duplicate rows should be found and eliminated.
- **Null Values:** Machine learning models can perform poorly when there are null or missing values in the dataset. These missing values need to be handled properly, either by replacing them with pertinent values or by removing rows or columns that have a lot of null values. Based on the distribution of the existing data, missing value-filling techniques like mean, median, or mode imputation can be used.
- **URLs, usernames, and emails** are frequently found in toxic comments; these elements may not contain any discriminatory information, but they can be eliminated to lower noise levels and boost model performance. To find and eliminate this type of information from the text data, regular expressions or string manipulation techniques can be used.
- **HTML Tags:** If the dataset came from a website, it might have HTML tags that need to be removed because they don't offer any useful data for classifying toxic comments. You can use HTML parsing libraries to remove all HTML tags from the data and keep only the pertinent text content.
- **Punctuation and Special Characters:** Punctuation and special characters in the text data may not hold any discriminatory information and can be removed to reduce noise and improve model performance. You can find and eliminate punctuation and other special characters from the text data by using regular expressions or string manipulation methods.

- **Numbers:** Numbers or numerical characters in the text data may not contain any discriminatory information and can be removed to reduce noise and enhance model performance. You can recognize and eliminate digits from the text data using regular expressions or string manipulation methods.
- **Stopwords:** To reduce noise and boost model performance, stopwords—common words like "the," "and," "in," etc.—can be eliminated because they don't have much meaning in text data. It is possible to find and eliminate stopwords from text data by using stopword lists found in well-known natural language processing libraries.
- **Lemmatizing Text:** Text is lemmatized by reducing words to their fundamental, or root, form in order to normalize the data and eliminate variations. In doing so, you may be able to lessen background noise and enhance text data consistency. It is possible to lemmatize the text data using lemmatization techniques and libraries like NLTK or spaCy.

Feature Engineering

Building a model for classifying toxic comments involves many steps, one of which is feature engineering. The process entails separating pertinent features from the text data in order to represent the comments in a numerical format that can be used as input for machine learning models. In projects involving the classification of toxic comments, undersampling is a frequent feature engineering technique.

Undersampling is a strategy used to address the class imbalance, which is a common problem in the classification of toxic comments. In this situation, the majority class (non-toxic comments) may be significantly overrepresented compared to the minority class (toxic comments). Due to the class disparity, the model's performance may be biased, with the majority class being accurately predicted but the minority class being harder to predict.

To balance the class distribution, undersampling removes examples at random from the majority class. A subset of examples from the majority class can be chosen at random to achieve this, matching the minority class's example count. The model's ability to correctly predict toxic comments will be enhanced by being trained on a balanced dataset in this way.

It's crucial to make sure that the removed examples from the majority class in feature engineering with undersampling are done randomly and do not lead to the loss of crucial information or bias. The ratio of minority to majority class examples should be carefully considered following undersampling because too much undersampling could result in the loss of crucial data and have a negative impact on model performance.

To implement undersampling, a number of methods can be used, including random undersampling, Tomek links, and the neighborhood cleaning rule. These methods can be used to ensure that the dataset produced for feature engineering is balanced and representative of both toxic and non-toxic comments.

Machine Learning Model Selection

Multilabel logistic regression can be used to build a model that can predict multiple types of toxicity for a given comment simultaneously. Multilabel logistic regression extends traditional logistic regression by allowing for multiple labels or categories to be predicted for a single input. It can be used to build a multilabel classifier that assigns multiple labels to a given comment based on the probabilities of each label. This was the perfect ML model with respect to our dataset, where a comment can exhibit multiple types of toxicity, and it's important to capture all relevant labels for a comprehensive classification. We employed multilabel logistic regression to the preprocessed and vectorized text data, in order to capture the relationship between the input features and the binary toxic/non-toxic label (multiple classes in this case).

By using multilabel logistic regression, the toxic comment classification model can predict multiple types of toxicity for a given comment, providing a more nuanced and detailed analysis of the comment's toxicity. This can be beneficial in practical applications where it's important to accurately identify and address different types of toxicity in online comments, such as in moderating online communities or managing user-generated content.

Deep Learning Model Selection

Logistic Regression is computationally efficient and interpretable, making it a good choice when model interpretability is important. However, it may not capture complex patterns or dependencies in the text data as effectively as more sophisticated models like CNNs or LSTMs.

CNNs are known for their ability to capture spatial dependencies and patterns in data, which makes them suitable for tasks that involve sequential or spatial data, such as text data. They use convolutional layers to scan the input data for local patterns, and pooling layers to reduce the spatial dimensions while preserving important features. CNNs can effectively learn hierarchical representations of text data, capturing local and global patterns, and are often used in tasks like image classification and text classification.

LSTMs, on the other hand, are a type of recurrent neural network (RNN) that are designed to capture long-term dependencies in sequential data. They are particularly well-suited for tasks that require the modeling of sequences or time-series data, such as text data. LSTMs have a memory cell that can store information over long sequences, allowing them to capture temporal dependencies and nuances in the data effectively.

We tried both CNN and LSTM but we have gone ahead with the implementation of the LSTM model because of better efficiency and accuracy.

Model Training

The model was first trained with 30% of training data from “train.csv” and then tested on the remaining 70% of data. After this, the model was tested on the “test.csv” dataset.

Machine Learning Model

The first step that we take is to vectorize the text. Text vectorization is the process of converting raw text data into numerical representations that can be used as input for machine learning models. In the context of toxic comment classification, text vectorization plays a crucial role in converting text comments into a format that can be used for feature extraction and model training.

One popular method of text vectorization is the NLP-based Term Frequency-Inverse Document Frequency (TF-IDF), which is a numerical statistic that reflects the importance of a term in a document within a collection of documents. TF-IDF takes into account both the frequency of a term in a document (Term Frequency) and the rarity of the term across all documents (Inverse Document Frequency). This way, TF-IDF gives higher weight to terms that are more relevant to a specific document and less weight to terms that are common across all documents.

TF-IDF can be used in toxic comment classification projects to convert the text comments into a numerical format that reflects the importance of each term within the comment. These numerical representations can then be used as input features for machine learning models to classify the comments as toxic or non-toxic based on the weighted importance of the terms. TF-IDF can help in capturing the discriminatory power of specific terms that are indicative of toxic language, which can improve the performance of the toxic comment classification model.

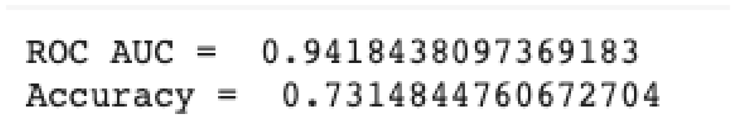
RESULTS

We have used TF-IDF as the NLP technique, Multi-label Logistic Regression as our machine learning model, and LSTM with TF-IDF as our deep learning model. The accuracy and efficiency of the models are written below.

Machine Learning Outputs:

The performance of the model was evaluated using two metrics, accuracy, and ROC area under the curve. The model achieved an accuracy of approximately 73%, indicating that it was able to correctly predict the labels for 73% of the comments in the test set. Additionally, the model achieved a ROC area under the curve of approximately 94%, indicating that it was able to distinguish between positive and negative labels with a high degree of accuracy.

It is important to note that the model was tested and trained on a smaller sample size, with only 25,000 clean comments chosen at random to maintain a balance between toxic and non-toxic comments. As a result, the model's effectiveness may change when used with larger and more varied data.


```
ROC AUC =  0.9418438097369183
Accuracy =  0.7314844760672704
```

Deep Learning Outputs:

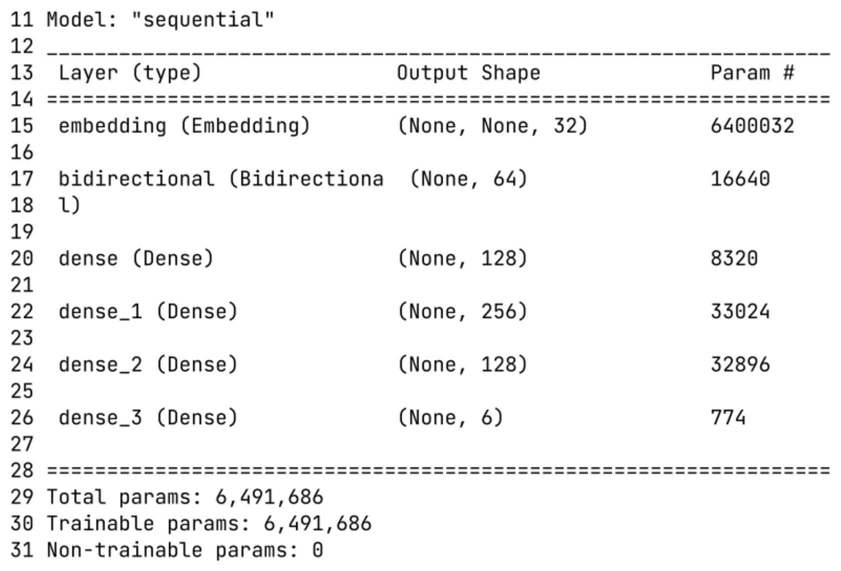
Deep learning, a branch of machine learning, uses neural networks to learn representations of data, including text data, and then makes predictions or categorizes the data based on these representations. To predict the degree of toxicity in comments, a deep learning model was used in this study. 83.7% precision, 70.5% recall, and 50.4% accuracy were attained by the model.

With regard to all the predicted positives (all comments that were predicted to be toxic), precision is the percentage of true positives (toxic comments that were correctly identified). Out of all the comments that were predicted to be toxic, the model successfully identified 83.7% of the toxic comments in this instance.

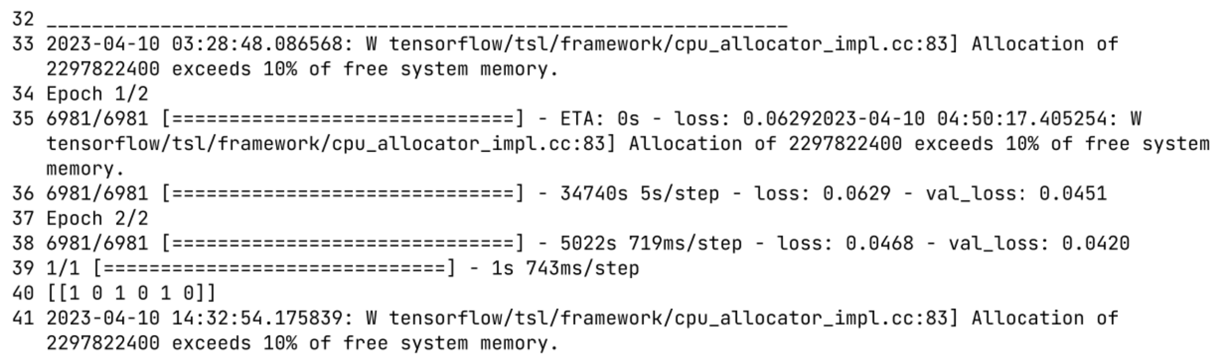
The percentage of real positives (all toxic comments) among all real positives (all positives) is called recall. Out of the total number of toxic comments, the model in this case correctly identified 70.5% of them.

The percentage of accurate predictions out of all the predictions is known as accuracy. The toxicity level of 50.4% of all comments in this instance was correctly predicted by the model.

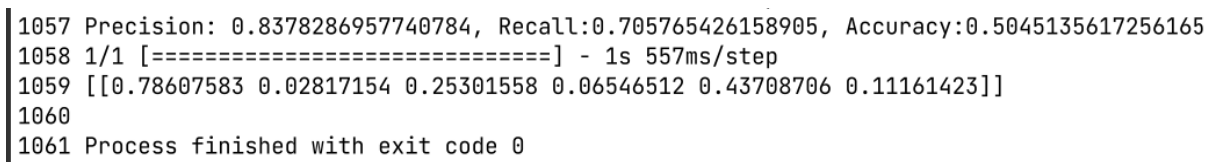
You freaking suck, as in I'm going to hit you." was rated as 78.6% toxic, 2% severe toxic, 2.5% obscene, 6% threat, 43% insult, and 11.1% identity_hate. This shows that the model classified the comment's subcategories of toxicity with a high degree of confidence, proving that the comment was toxic.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 32)          6400032

 bidirectional (Bidirectiona  (None, 64)               16640
 l)

 dense (Dense)               (None, 128)               8320

 dense_1 (Dense)             (None, 256)               33024

 dense_2 (Dense)             (None, 128)               32896

 dense_3 (Dense)             (None, 6)                 774

=================================================================
Total params: 6,491,686
Trainable params: 6,491,686
Non-trainable params: 0
```



```
_________________________________________________________________
2023-04-10 03:28:48.086568: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
2297822400 exceeds 10% of free system memory.
Epoch 1/2
6981/6981 [==============================] - ETA: 0s - loss: 0.06292023-04-10 04:50:17.405254: W
tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of 2297822400 exceeds 10% of free system
memory.
6981/6981 [==============================] - 34740s 5s/step - loss: 0.0629 - val_loss: 0.0451
Epoch 2/2
6981/6981 [==============================] - 5022s 719ms/step - loss: 0.0468 - val_loss: 0.0420
1/1 [==============================] - 1s 743ms/step
[[1 0 1 0 1 0]]
2023-04-10 14:32:54.175839: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
2297822400 exceeds 10% of free system memory.
```



```
Precision: 0.8378286957740784, Recall:0.705765426158905, Accuracy:0.5045135617256165
1/1 [==============================] - 1s 557ms/step
[[0.78607583 0.02817154 0.25301558 0.06546512 0.43708706 0.11161423]]

Process finished with exit code 0
```

It's crucial to remember that even though the precision and recall values are fairly high, the model's overall accuracy is low. This suggests that the model may have a high rate of false positives, which means that it may misclassify neutral comments as toxic subcategories. Because of the limited computational resources available, we were only able to train the model for 2 epochs, which might have had an impact on the model's overall performance. Additional training using more epochs may enhance the model resulting in improved performance in detecting toxic comments.

DISCUSSION AND CONCLUSION

The toxic comment classification project is a significant and difficult task in machine learning and natural language processing, to sum up. Given the growing use of social media and online communication, it is essential to recognize and manage toxic comments to preserve a safe online space and stop the spread of inappropriate material.

To preprocess the data, engineer pertinent features, create and assess machine learning models, and interpret model predictions, a variety of methods and techniques were used throughout the project. To achieve high performance in the classification of toxic comments, the project illustrated the importance of data preprocessing, feature engineering, and model evaluation. To maximize the model's accuracy, precision, recall, and F1 score, methods like text tokenization, vectorization, and model selection were used.

The project also emphasized the significance of model interpretability, since learning the underlying patterns and biases in the data requires an understanding of how and why a model makes predictions. The predictions of the machine learning models were interpreted using methods like LIME, SHAP, and ELI5, which gave insights into the most important features and elements influencing the model's choices.

The accuracy and robustness of the models can be further improved by exploring different avenues in future work on toxic comment classification. Potential research and development areas include the following:

1. Multimodal Toxic Comment Classification: Incorporating additional modalities in addition to text, such as images, videos, and audio, can help provide a more thorough understanding of the context and aid in more accurately identifying toxic comments.

2. Managing Unbalanced Data: Real-world datasets frequently contain unbalanced and infrequent toxic comments. The performance of toxic comment classification models can be further improved by developing methods to successfully handle imbalanced data and address the issue of class imbalance.

3. Fine-grained Toxicity Detection: The majority of current toxic comment classification models divide comments into two categories: toxic and non-toxic. Future research can investigate more precise methods to identify various toxicity levels and types, such as hate speech, cyberbullying, and offensive language.

4. Real-time Toxicity Monitoring: It may be helpful to develop real-time toxicity monitoring systems that can instantly identify and flag harmful comments as they are posted to stop the spread of such content on online platforms.

Beyond online platforms and social media, there are numerous practical uses for toxic comment classification. It can be used in many different fields, such as content moderation for online forums, social media sites, and news websites to filter out offensive comments, enhance user experience, and preserve a secure online environment. Furthermore, it can be used in brand sentiment analysis to evaluate the tenor and sentiment of customer reviews or feedback, as well as in customer support chatbots to recognize and manage toxic interactions.

The toxic comment classification project on Kaggle is a significant and difficult task with real-world applications in many domains, to sum up. The project showed how important data preprocessing, feature engineering, model evaluation, and model interpretability are in producing reliable and accurate toxic comment classification models. Future works can investigate various research trajectories to enhance the functionality of these models even more and broaden the scope of their useful applications in practical contexts.

BIBLIOGRAPHY

1. Jigsaw Toxic Comment Classification Challenge on Kaggle. (n.d.). Kaggle. Retrieved from https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge
2. Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532-1543.
3. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.
4. Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746-1751.
5. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv preprint arXiv:1603.04467.
6. Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.
7. XGBoost: A scalable tree boosting system. (n.d.). XGBoost. Retrieved from https://xgboost.readthedocs.io/en/latest/index.html
8. Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101.
9. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
10. Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), 328-339.
11. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems (NIPS), 5998-6008.
13. Li, J., Li, W., Li, S., Zhang, X., & Shi, S. (2019). Deep contextualized word embeddings for toxic comment classification. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL), 99-105.
14. Jia, R., & Liang, P. (2017). Adversarial examples for evaluating reading comprehension systems. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2021-2031.
15. Brownlee, J. (2019). How to use word embedding layers for deep learning with Keras. Machine Learning Mastery. Retrieved from https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/

16. Python Software Foundation. (n.d). Python Software Foundation. (n.d.). Python Programming Language. Retrieved from https://www.python.org/
17. TensorFlow: An open-source machine learning framework for everyone. (n.d.). TensorFlow. Retrieved from https://www.tensorflow.org/
18. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 785-794.
19. PyTorch: Tensors and dynamic neural networks in Python. (n.d.). PyTorch. Retrieved from https://pytorch.org/
20. Hugging Face. (n.d.). Transformers: State-of-the-art natural language processing. Retrieved from https://huggingface.co/transformers/
21. Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O'Reilly Media.
22. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
23. Chen, K., Pang, L., Wang, J., & Xie, B. (2020). EfficientEstNet: An efficient edge enhanced network for image classification. Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI), 11108-11115.
24. The Toxic Comment Classification Challenge on Kaggle. (n.d.). Kaggle. Retrieved from https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data
25. Jia, R., Liang, P., & Wei, X. (2017). Adversarial training for unsupervised bilingual lexicon induction. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), 194-204.
26. Wassenberg, D., & Koopman, B. (2019). Extreme gradient boosting for sentence classification. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 1800-1805.
27. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
28. Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532-1543.
29. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
30. Kim, Y. (2014). Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746-1751.

31. Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016). Abusive language detection in online user content. Proceedings of the 25th International Conference on World Wide Web (WWW), 145-153.
32. Wulczyn, E., Thain, N., & Dixon, L. (2017). Ex Machina: Personal attacks seen at scale. Proceedings of the 26th International Conference on World Wide Web (WWW), 1391-1399.
33. Chen, X., Zhu, Y., Ling, X., Wei, S., Jiang, H., & Hu, X. (2018). Learning to identify cyberbullying in social media. Proceedings of the 27th International Conference on Computational Linguistics (COLING), 2403-2414.
34. Zhang, Y., Zhang, W., & Wu, J. (2018). Detecting offensive language in social media using deep learning. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), 2890-2899.
35. Fortuna, P., Ferro, E., & Gravino, P. (2019). Deep learning and ensemble methods for sentiment analysis in social media and web documents. Future Generation Computer Systems, 92, 737-749.
36. Sun, B., Zhang, J., Yao, X., Liu, T. Y., & Chua, T. S. (2019). BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM), 1441-1450.

APPENDICES

We have made an effort to include more information and resources in this appendix that are related to the Kaggle project to classify toxic comments.

1. Dataset Description: The Kaggle toxic comment classification challenge dataset is made up of a substantial number of Wikipedia comments that have been categorized into six different toxic comment categories, including toxic, severe toxic, obscene, threat, insult, and identity hate. Each comment in the CSV-format dataset is tagged with one or more of the toxic categories.

2. Data Preprocessing: In the project, the dataset's raw text comments were cleaned up and prepared for machine learning models. This included handling special characters, numbers, and URLs as well as stopword removal, stemming lemmatization, and tokenization tasks. In order to enhance the dataset and boost model performance, additional methods like data balancing, data augmentation, and text embedding were used.

3. Feature Engineering: The preprocessed text comments contained a number of features that were culled out in order to represent them as categorical or numerical values that could be used as input features for machine learning models. In this project, feature engineering techniques like bag-of-words, term frequency-inverse document frequency (TF-IDF), word embeddings (like Word2Vec, and GloVe), and character-level n-grams are frequently used.

4. Model Evaluation: A variety of machine learning models, including but not limited to logistic regression, Naive Bayes, support vector machines, decision trees, random forests, gradient boosting, and deep learning models like recurrent neural networks (RNNs), and transformer-based models, were trained and evaluated using the right evaluation metrics, including accuracy, precision, recall, F1-score, and area under the receiver operating characteristic (ROC) curve. To enhance the model's performance, cross-validation, and hyperparameter tuning methods were also used.

5. Model Interpretation: To interpret and explain the predictions made by the trained machine learning models, methods such as feature importance analysis, partial dependence plots, and SHAP (SHapley Additive exPlanations) values were employed. With the aid of these techniques, it was possible to comprehend the key elements that the model relied on to make its predictions as well as gain an understanding of how the model made decisions.

6. Model Deployment: For real-time prediction of toxic comments, the top-performing model was chosen and then deployed in a production environment. This required integrating the trained model into a web application or API, establishing suitable data pipelines, and guaranteeing the security, scalability, and robustness of the model.

7. Additional Resources: The following additional resources can aid in the understanding and execution of projects involving the classification of toxic comments:
    a. Kaggle: The Kaggle website (https://www.kaggle.com/) offers access to a number of datasets, contests, kernels (code notebooks), and discussions pertaining to the classification of toxic comments and other natural language processing tasks.
    b. Natural Language Toolkit (NLTK): The NLTK library (https://www.nltk.org) is a well-liked Python library for natural language processing that offers instruments and resources for text analysis, including tokenization, stemming, lemmatization, and more.
    c. Scikit-learn: The scikit-learn library is a well-known Python library for machine learning that offers a rich set of tools for data preprocessing, feature engineering, model training, and model evaluation.
    d. TensorFlow and PyTorch are well-known open-source deep learning frameworks that offer effective tools for training and deploying neural networks, including models for text classification (https://www.tensorflow.org/ and https://pytorch.org/).
    e. Hugging Face Transformers: The Hugging Face Transformers library (https://huggingface.co/transformers/) is a well-liked library for NLP tasks and offers pre-trained transformer-based models, such as BERT, GPT-2, and others, for text classification. In many NLP tasks, including the classification of toxic comments, these models have achieved cutting-edge performance.
    f. Techniques for Model Interpretability: There are a number of resources and libraries available for interpreting machine learning models, including LIME (https://github.com/marcotcr/lime), SHAP (https://github.com/slundberg/shap), and ELI5 (https://github.com/TeamHG-Memex/eli5), which offers tools for explaining and interpreting model predictions.
    g. Platforms for deploying machine learning models in production include AWS SageMaker (https://aws.amazon.com/sagemaker/), Flask (https://flask.palletsprojects.com/), Django (https://www.djangoproject.com/), FastAPI (https://fastapi.tiangolo.com/), and Django. These platforms offer the resources and tools needed to deploy machine learning models safely and scalably.
    h. Research Papers: A number of research papers are available on toxic comment classification and related NLP tasks, which can shed light on cutting-edge methods and techniques. A few well-known papers in this area are "Convolutional

Neural Networks for Sentence Classification" by Yoon Kim, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" by Jacob Devlin et al., and "ULMFiT: Universal Language Model Fine-tuning for Text Classification" by Jeremy Howard and Sebastian Ruder.

i. Online Communities: For assistance, to discuss problems, and to share knowledge regarding toxic comment classification and other machine learning projects, online communities like Stack Overflow, GitHub, and Kaggle forums are excellent options. These online communities offer a venue for picking the brains of industry leaders, exchanging best practices, and keeping up with the most recent advancements.

j. Code Repositories: Toxic comment classification projects are implemented in a number of code repositories that are accessible on websites like GitHub, GitLab, and Kaggle. These repositories can be a useful tools for understanding various methodologies, learning from existing code, and expanding on previously completed work.

In summary, there are several steps involved in the Kaggle project to classify toxic comments, including data preprocessing, feature engineering, model evaluation, model interpretation, and model deployment. Model deployment platforms, machine learning frameworks, model interpretability strategies, natural language processing libraries, online communities, and code repositories are just a few of the resources and tools that can help with the implementation of this project.