Toxic vs. Non-Toxic Comments Classification Report

AUTHOR: AIKATERINI KOUTRIS

DATE: 19 SEPTEMBER 2024

Introduction

Toxic comments can harm online communities and discourage constructive discourse, making this classification essential for moderation. Identifying abusive behaviours, such as hate speech, offensive language, sexism and racism, in utterances from social media platforms is known as abuse detection (Founta et al., 2018). The purpose of this document is to report on the descriptive and exploratory data analysis of the project where abuse detection is investigated by identifying and classifying comments as either toxic or non-toxic using natural language processing techniques (NLP) and machine learning.

Data Overview

DATASET DESCRIPTION

The comments dataset has three dataset attributes namely "Unnamed", "toxic" and "text". The "Unnamed" column refers to the individual comments with a unique identification number and is of type integer. The "toxic" column refers to the comment's class label whereby 0 indicates that the comment's class is non-toxic and 1 indicates that the comment's class is toxic. The data type in the "toxic" column is of type integer. The "text" column refers to the comment's text and is of type object.

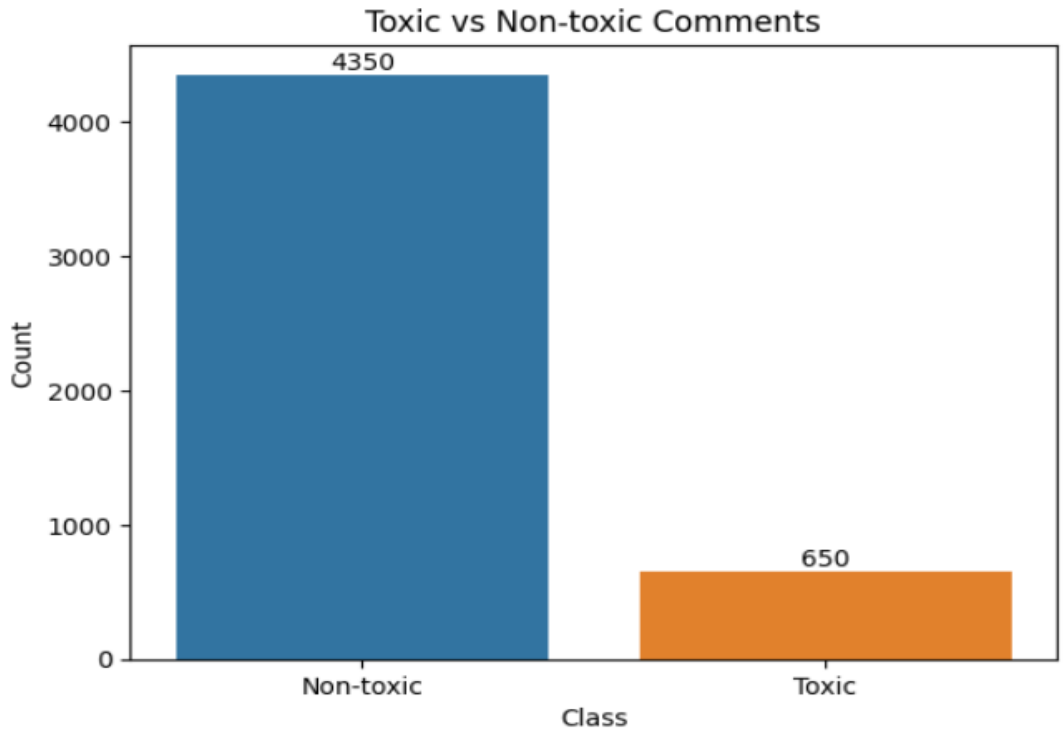
MISSING VALUES

The comments dataframe is checked for missing values and their distribution across variables, as missing information can notably affect the quality and reliability of the evaluation. In this case there are no missing values, therefore we do not need to remove any missing values.

Descriptive Statistics

The comments dataset has 5 000 entries. According to the visual bar chart showing the counts for toxicity, the comments dataframe is highly imbalanced as there are more non-toxic comments (4 350 comments) than there are of toxic comments (650 comments) in the dataset as shown in Figure 1.

Figure 1



Note. A bar chart showing the distribution of toxic vs. non-toxic comments.

Data Preprocessing

The "Unnamed: 0" column is unnecessary to include in the comments dataframe as the data in the "Unnamed: 0" column matches that of the observation column values. Therefore, the "Unnamed" column can be dropped from the dataframe. Additionally, any duplicated data is removed from the dataframe. Operations specific to strings, like concatenation, substring extraction, or regular expression matching were performed, thus the "text" column needed to be converted from object type to string type.

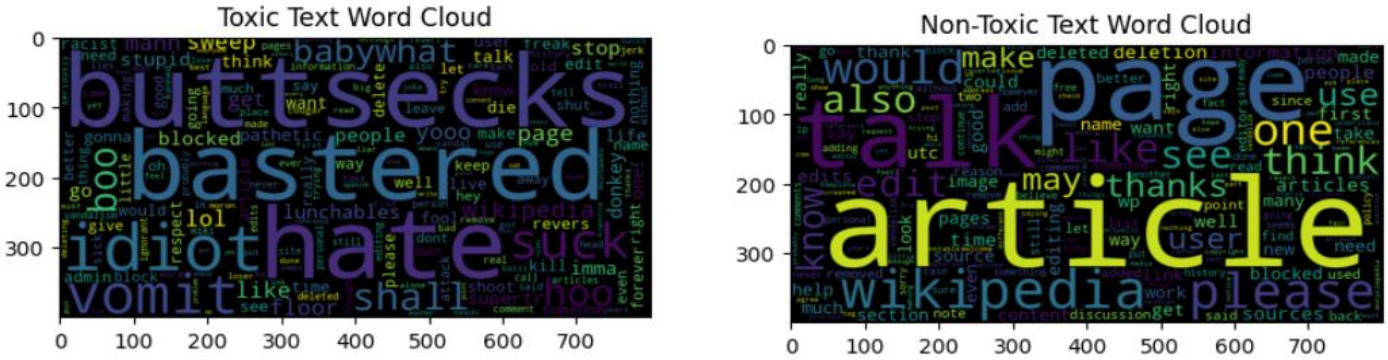
Handling categorical variables such as the 'Toxic' column is better represented in the machine if changed to numerical data, therefore reverted the "toxic" values back into numerical representation. Will use One-hot encoding where "toxic" is assigned [1, 0] and [0, 1] is for "non-toxic".

Text preprocessing is important for performing sentiment analysis, as it helps to clean and normalize the text data, making it easier to analyse (De Silva, 2023). Some techniques included converting to lowercase, removing HTML tags, special characters, digits, and stop words. Removing common and irrelevant words that are unlikely to convey much sentiment. Stop words are words that are very common in a language and do not carry much meaning, such as "and" "the," "of," and "it." These words can cause noise and skew the analysis if they are not removed.

Exploratory Data Analysis (EDA)

Considering text analysis using a visual representation of the most common words in toxic vs. non-toxic comments using word clouds as illustrated in Figure 2. The distribution of comment lengths for both classes, toxic and non-toxic and the top 5 common toxic and non-toxic words were calculated as shown in Table 1 and Table 2.

Figure 2



Note**.** Word clouds visualizing the most common words in toxic comments vs. non-toxic comments.

Table 1

Top 5 Common Toxic Words

Word	Count
buttsecks	497
hate	468
bastered	435
idiot	308
vomit	217

Table 2

Top 5 Common Non-toxic Words

Word	Count
article	1252
page	974
talk	895
wikipedia	863
please	736

Model Selection and Training

Variables Identification, Data Splitting and Pipelines Creation

The independent and dependent variables were identified where the text column was the dependent variable, and the toxic column was the independent variable. The comment dataset was divided into training (80%) and testing (20%) sets.

Pipelines were created as they combine multiple steps of data processing and model training into a single workflow, ensuring each step of the process is executed in a specific order and maintain consistency throughout the workflow (Whorton, 2021). The three pipelines that were used are as follows: LogisticRegression, RandomForestClassifier, and MultinomialNB.

Performance evaluation was conducted through model evaluation metrics such as accuracy, precision, recall, and F1-score.

Hyperparameter Turning

To find the best set of hyperparameters that results in the highest performance of the model, hyperparameter turning using GridSearchCV was conducted and showed different combinations of hyperparameters, and their corresponding model performance (score) as shown in Table 3, overall, the model with clf__C = 100 gives the best result.

Table 3

Summary of Hyperparameter turning using GridSearchCV

Results	Explanation
Best parameters found: {'clf__C': 100} Best score: 0.90725	Setting the C parameter to 100 produced the highest model performance (score) of 0.90725. The C parameter is utilised in Logistic Regression, and a greater value indicates less regularisation.
Best parameters found: {'clf__criterion': 'gini'} Best score: 0.9065	The best result was obtained with the decision tree criterion set to gini. The score of 0.9065 is close to the first result, showing that using the Gini index to separate nodes in a decision tree classifier produced similar results.
Best parameters found: {'clf__alpha': 0.5, 'clf__fit_prior': False} Best score: 0.8915	This means that for a Naive Bayes classifier, the best performance was obtained with alpha set to 0.5 and fit_prior set to False. The lowest score is 0.8915, indicating that this combination of hyperparameters resulted in a suboptimal model.

Retraining Model

Retraining a model with 5-fold cross-validation involves training it 5 times on 5 subsets of data to accurately predict performance more reliably. The findings of the 5-fold cross-validation demonstrate that Logistic Regression outperforms the other two models, with an accuracy of 0.9073, recall of 0.9073, and F1 score of 0.8948. It provides a lower precision (0.8976) yet exceeds the other models with regard to of overall balance. Random Forest is close, with an accuracy of 0.9065, precision of 0.9089, and recall of 0.9065, but its F1 score is lower, at 0.8844, indicating that it is less balanced than Logistic Regression. In contrast, Multinomial Naive Bayes performs poorly, with the lowest accuracy (0.8915), precision (0.8767), and F1 score (0.8791).

Overall, Logistic Regression is the most dependable model.

Test and Evaluate All Three Models on Held-Out Dataset

The model was assessed on a held-out dataset to simulate how the model will perform with unseen data.

Table 4

Performance Metrics for three models:

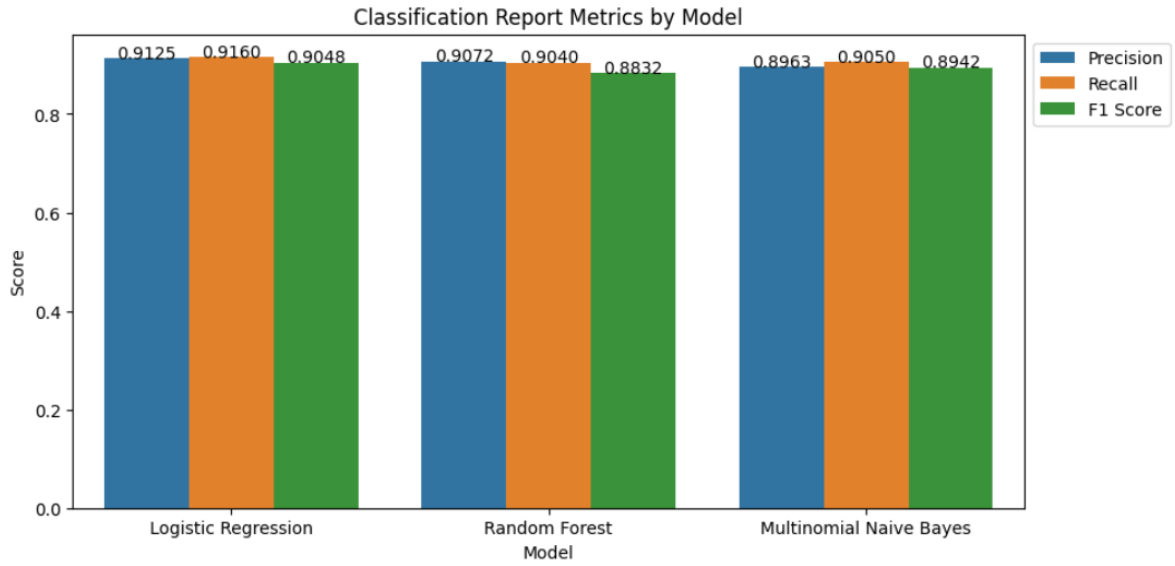
	Logistic Regression	Random Forest	Multinomial Naive Bayes
Accuracy	0.916000	0.904000	0.905000
Recall	0.916000	0.904000	0.905000
Precision	0.91246	0.907167	0.896343
F1 Score	0.904824	0.883156	0.894157

The table above gives specific metrics for each model and overall performance, whereas the metrics for the evaluation of each model provided a summary across multiple evaluations. Comparing the table's metrics to the evaluation model's metrics for recall and F1-score values might be challenging as the values vary depending on the various folds. However, the average accuracy value should align closely with the accuracy values reported in the table.

The Logistic Regression accuracy was 0.9073 and the classification report accuracy was 0.9160. The Random Forest accuracy was 0.9065 and the classification report accuracy was 0.904. The Multinomial Naive Bayes accuracy was 0.8915 and the classification report accuracy was 0.9050.

Visualization

Figure 3



Note. Bar chart of classification Report Metrices by Models.

Visualising the classification report of the three models in Figure 3, the dataset is highly imbalanced as there are more non-toxic comments than there are of toxic comments with reference to the bar chart showing the class distribution. Therefore, for imbalanced datasets, precision, recall and F1-scores are more important than accuracy. The classifiers that optimize these metrics are preferred.

A high precision value indicates that a classifier can predict a positive outcome and is more likely to be correct. A higher recall value indicates that the classifier is successfully identifying a larger proportion of actual positive instances. A F1-score reflects a good balance between precision and recall. Therefore, the Logistic Regression model is the best classifier with precision value of 0.9125, recall value of 0.9160 and F1-score value of 0.9048. The Logistic Regression model values scored the highest between the three models.

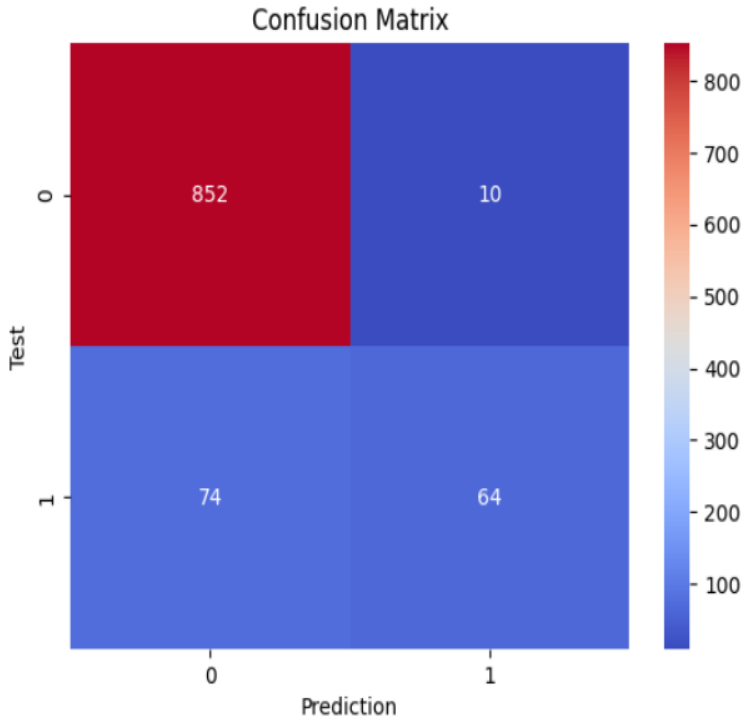
Classification Report for Logistic Regression Model

The classification report gives specific metrics for each model and overall performance, whereas the metrics for the evaluation of each model provided a summary across multiple evaluations. Comparing the classification report to the evaluation model metrics for recall and F1-score values might be challenging as the values vary depending on the various folds. However, the average accuracy value should align closely with the accuracy values reported in the classification report. The Logistic Regression accuracy was 0.9160 and the classification report accuracy was 0.9160.

Considering the whole dataset, the model performed well enough to be able to recognise a text as being either toxic or non-toxic correctly. Regarding the precision for the classes, where the model predicted would be a toxic text, 92.01% were classified as non-toxic (class 0) and 86.49% were classified as toxic (class 1). For the recall of the classes, out of all the classes that the model predicted would be said text, 98.84% were classified as class 0 and 46.38% were classified as class 1. This means that model is successfully identifying a larger proportion of actual positive instances as class 0. Class 0 has a f1-score equal to 0.9530 which means that the model determined the text as non-toxic very accurately. The model struggled to determine class 1 as the toxic text with 60.38% accurately, which is still very high but less than class 0's f1-score. Among the classes in the test dataset, the highest samples/ instances presented in the dataset (137) were from class 0 and the least samples/ instances presented in the dataset are from class 1.

Visualization

Figure 4



Note. Confusion matrix visual for the best-performing model.

The confusion matrix (Figure 4) is a representation of the what the model predicted compared to what is actually correct. The confusion matrix is evaluates the performance of a classification model. The model recognised 852 instances where the given non-toxic text was actually non-toxic (0), however the model misclassified the given text as toxic (1) in 10 instances. The model recognised 74 instances where the given toxic text was actually toxic (1), however the model misclassified the given text as non-toxic (0) in 64 instances.

The accuracy value is calculated as follows (TP+TN)/(TP+TN+FP+FN) which gives a value of 0.9160, indicating that the ratio of correctly predicted instances to the total instances is high. The precision value is calculated as follows TP/(TP+FP) which gives a value of 0.9125, indicating that a high value of the predicted positive cases were actually positive. The recall value is calculated as follows TP/(TP+FN) which gives a value of 0.9160, indicating a high value where positive cases were correctly identified. The F1-score value is calculated as follows Formula: 2×(Precision×Recall)/(Precision+Recall) which gives a value of 0.9048.

Prediction

Using the Logistic Regression Model to predict the label and probability for sentences and comparing it to the prediction using predict_proba as in Table 5. (Toxic = 1, Non-toxic = 0).

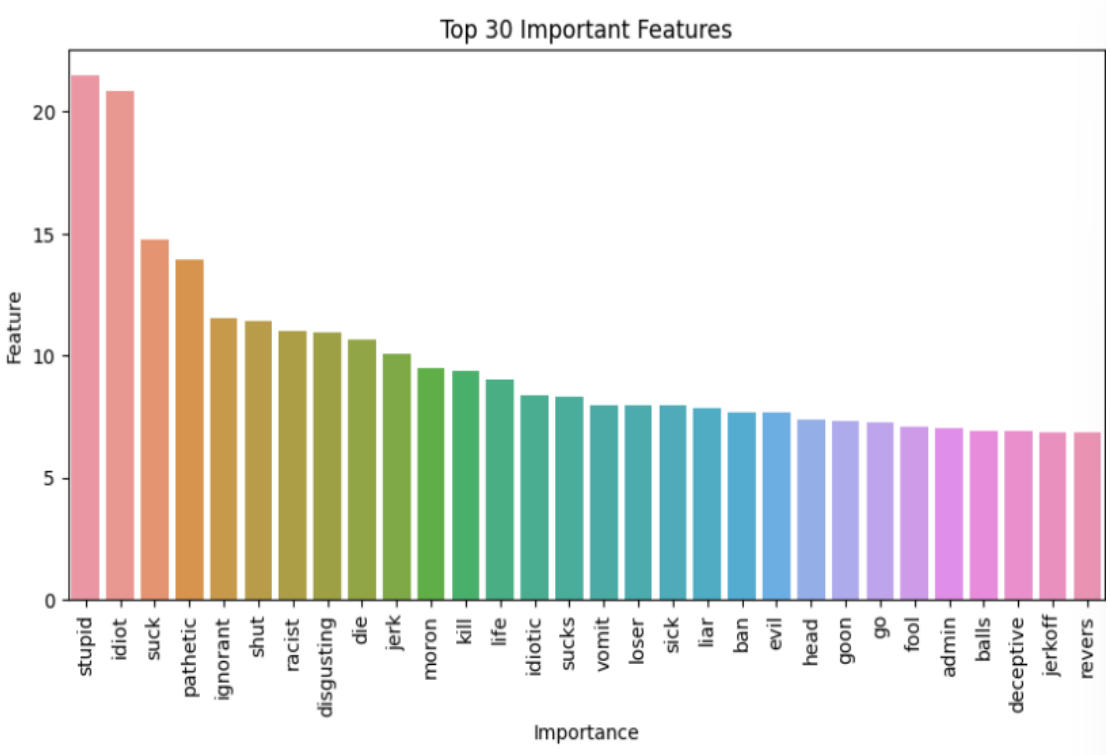
Table 5

Sentence	Using Logistic Regression		Using predict_proba	
	Label	Probabilities	Label	Probabilities
'the king is an awful human and must shut up'	1	Non-toxic: 0.0005, Toxic: 0.9995	1	[0.46923456 0.53076544]
'I really enjoyed the flight, but the food was bad'	0	Non-toxic: 0.9905, Toxic: 0.0095	0	0.85020719 0.14979281]

Feature Importance

There are certain words (e.g., "stupid", “idiot”, “suck”, “pathetic”) that were strong indicators of toxicity as illustrated in Figure 5. Online communities could use this list of words as reference to flag any comment made in the future for endorsing online toxicity.

Figure 5



Note. Top 30 toxic words.

Balancing the Dataset

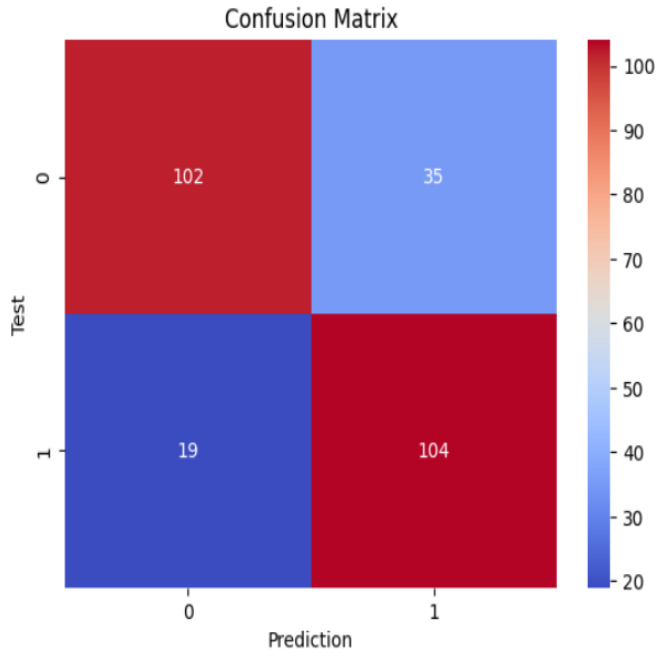
An unbalanced dataset can skew machine learning performance, making metrics like accuracy less useful. For example, in toxic comment classification, a model might predict "non-toxic" for most comments, achieving high accuracy but failing to identify toxic ones. Therefore, metrics like precision, recall, and F1-score, which are better suited for imbalanced data, should be prioritized.

Comparison for Classification Reports

Looking at the classification report for the best_logit model, the model performed well enough to be able to recognise a text as being either toxic or non-toxic correctly. The precision score for the classes, where the model predicted would be a toxic text, 84.30% were classified as non-toxic (class 0) and 74.82% were classified as toxic (class 1). The recall score indicated that the model predicted would be said text, 74.45% were classified as class 0 and 84.55% were classified as class 1. Therefore, the model is successfully identifying a larger proportion of actual positive instances as class 0. Class 0 has a f1-score equal to 0.9412 which means that the model determined the text as non-toxic very accurately. The model struggled to determine class 1 as the toxic text with 79.23% accurately. Among the classes in the test dataset, the highest samples/ instances presented in the dataset (137) were from class 0 and the least samples/ instances presented in the dataset are from class 1. Considering the classification reports, the Logistic Regression model performed better than the best_logit model.

Visualization

Figure 6



Note. Confusion matrix visual.

The model as shown in Figure 6, recognised 102 instances where the given non-toxic text was actually non-toxic (0), however the model misclassified the given text as toxic (1) in 35 instances. The model recognised 104 instances where the given toxic text was actually toxic (1), however the model misclassified the given text as non-toxic (0) in 19 instances.

The accuracy value is calculated as follows (TP+TN)/(TP+TN+FP+FN) which gives a value of 0.7923, indicating that the ratio of correctly predicted instances to the total instances is high. The precision value is calculated as follows TP/(TP+FP) which gives a value of 0.7981, indicating that a high value of the predicted positive cases were actually positive. The recall value is calculated as follows TP/(TP+FN) which gives a value of 0.7923, indicating a high value where positive cases were correctly identified. The F1-score value is calculated as follows Formula: 2×(Precision×Recall)/(Precision+Recall) which gives a value of 0.7922.

Data Preprocessing and Feature Extraction

TF-IDF Vectorization

The text data was converted to numerical characteristics using TF-IDF. This method converts each document into a vector of numbers representing the relevance of a given word (or phrase) in the document in relation to the full corpus (set of documents). TF-IDF highlights words which are distinctive or relevant to a single text while reducing the weight of common words (such as "the," "is," or "and") that are found in multiple documents but have little value in identifying a document's content. The shape of the resulting feature matrix is (5000, 16676), where 5 000 is the number of documents (rows) and 16 676 is the number of unique terms (features) across all documents.

Dimensionality Reduction with PCA

The TF-IDF feature matrix has a large dimensionality (16 676 features), so it was necessary to compress it for better visualisation and analysis. PCA (Principal Component Analysis) was used to reduce the dataset to two dimensions. This transformation produced a matrix of shape (5 000, 2), with each document represented by its two main elements. PCA minimises dataset complexity by finding the most relevant characteristics that explain variance, allowing for 2D plotting while keeping critical information.

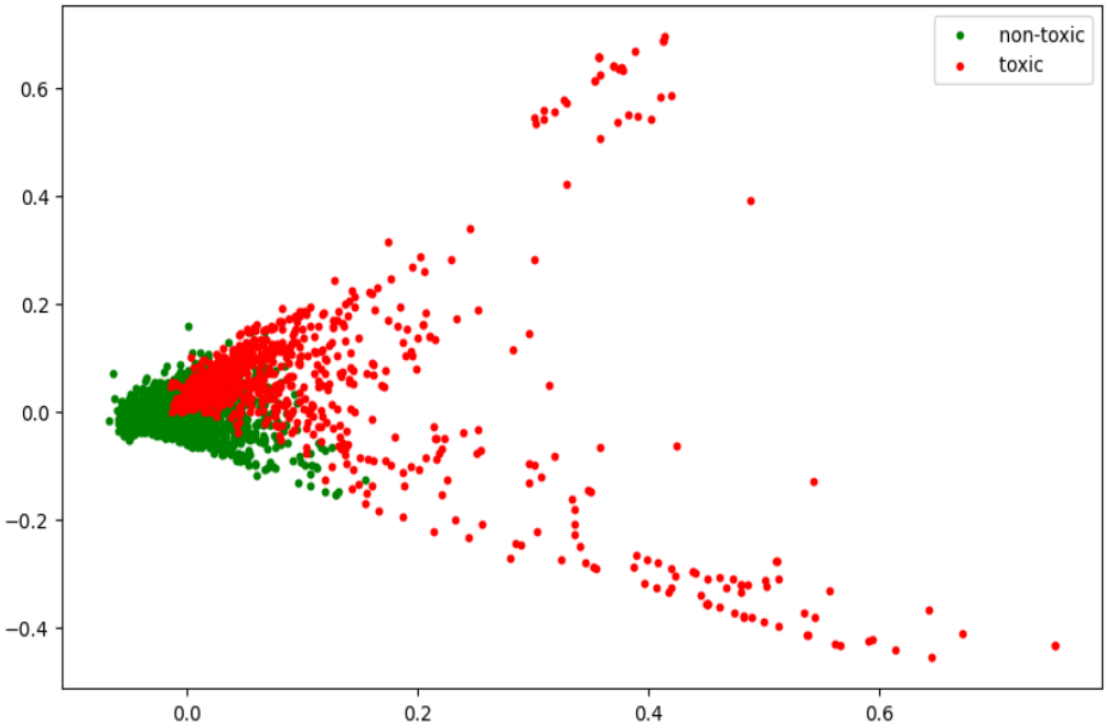
Clustering with K-Means

To cluster the documents, the K-Means Clustering technique was used, which divides the data into a predetermined number of clusters (K = 2). K-Means iteratively allocates each data point to the nearest cluster centre and changes the cluster centres until convergence is achieved.

Visualization

Using the minimised dimensions from PCA as x and y coordinates for visualisation, with each document coloured according to its allocated cluster. Figure 7 depicts how documents are categorised based on content similarity. The graph depicts the overlap between clusters, demonstrating locations where the algorithm struggled to distinguish between papers.

Figure 7



Note. Clustering model of non-toxic and toxic comments.

Conclusion

This analysis demonstrates the feasibility of classifying comments into toxic and non-toxic categories with good accuracy. Future work could involve refining the model with more advanced techniques (e.g., deep learning) and expanding the dataset to include more diverse examples. Limitations include potential biases in the dataset and the challenges of context understanding in comments

References

De Silva, M. (2023, August 29). Preprocessing Steps for Natural Language Processing (NLP): A Beginner's Guide. *Medium*. https://medium.com/@maleeshadesilva21/preprocessing-steps-for-natural-language-processing-nlp-a-beginners-guide-d6d9bf7689c9

Founta, A., Chatzakou, D., Kourtellis, N., Blackburn, J., Vakali, A., & Leontiadis, I. (2018, February 1). *A Unified Deep Learning Architecture for Abuse Detection*. arXiv.org. https://arxiv.org/abs/1802.00385

Stackoverflow. (2021, April 30). Pandas Replace NaN with blank/empty string. https://stackoverflow.com/questions/26837998/pandas-replace-nan-with-blank-empty-string

Stackoverflow. (2015, November 30). How can I add a column from one dataframe to another dataframe? https://stackoverflow.com/questions/33497896/how-can-i-add-a-column-from-one-dataframe-to-another-dataframe

Spark by examples. (2024, May 22). Pandas GroupBy Multiple Columns Explained. https://sparkbyexamples.com/pandas/pandas-groupby-multiple-columns/#google_vignette

Whorton, C. (2021, December 16). Pipeline For Text Data Pre-processing - Towards Data Science. *Medium*. https://towardsdatascience.com/pipeline-for-text-data-pre-processing-a9887b4e2db3