Springboard–DSC
Capstone Project 2

Toxic Comment Classification

By Zohreh Asaee

June 2022

DISCLAIMER AND WARNING: *Because of the nature of this project, this report contains words that are very offensive. Needless to say, it is not the intention of the author of this report, and this project, to offend anybody.*

1. Executive Summary

Social media has been growing fast and profound. The percentage of US adults who use social media increased from 5% in 2005 to 79% in 2019[1]. The rapid growth of social media has raised serious concerns about the threat of abuse and harassment on these platforms. Social media platforms struggle to facilitate conversations effectively and may lead to limiting or completely blocking user comments.

As mentioned in the Kaggle competition from where the dataset used in the project was obtained: "The conversation AI team, a research initiative founded by Jigsaw and Google (both a part of Alphabet), is working on tools to help improve the online conversation."[2] One area of focus for this team is the study of negative online behaviors, like toxic comments (*i.e.,* comments that are rude, disrespectful, or otherwise likely to make someone leave a discussion).

This capstone project aims to classify comments into non-toxic and toxic groups. In this regard, the main goals of this project are:

- To create models to classify comments into non-toxic and toxic.
- To perform an interpretability analysis and specify features impacting each class of non-toxic and toxic.

The dataset used in this project is obtained from the Kaggle website and includes labeled Wikipedia comments. We trained 20 models to classify the comments as non-toxic and toxic. Moreover, we used BayesianSearchCV to perform hyperparameter tuning of the trained models.

[1]https://ourworldindata.org/rise-of-social-media#:~:text=The%20percentage%20of%20US%20adults,to%20around%2030%25%20in%202018.
[2] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

The current project assumes that a social media platform is willing to detect toxic comments and block the corresponding accounts. Therefore, it is essential to decrease false positives since we don't want to block an account by mistake (high Precision). On the other hand, it is crucial to detect toxic comments and block them (high Recall). In practice, we can't achieve both high Precision and high Recall. An increase in Precision usually reduces Recall, and vice versa. This is called the precision/recall tradeoff. Therefore, the *f1*-score might be a better metric of performance when we are seeking a balance between precision and recall. In particular, the compromise is expressed in the form of the harmonic mean of precision and recall.

In terms of toxic class, our best model for f1-score was the LGBM classifier with an f1-score of 0.68. Also, our best model for recall parameter was Logistic Regression in conjunction with Random Undersampling with a score of 0.82.

We used the SHAP package[3] to analyze the impact of features on the classification output. We identified the features impacting the toxic output with a larger SHAP value compared to those impacting the non-toxic class. Therefore, an appearance of these features in a comment shifts the classification towards the toxic class.

The intended stakeholders of this project are social media platforms. It enables them to block the toxic comments and the associated profiles and decreases the toxicity and harassment on their platforms.

The implementation details can be found in the notebooks, along with all the project deliverables, in this GitHub repository[4].

[3] https://shap.readthedocs.io/en/latest/index.html
[4] https://github.com/ZAsaee/Toxic_Commnets_Classification

<u>DISCLAIMER AND WARNING</u>: Because of the nature of this project, this report contains words that are very offensive. Needless to say, it is not the intention of the author of this report, and this project, to offend anybody.

2. Approach

2.1. Data Acquisition and Wrangling

The main source of data for this project is Jigsaw toxic comment classification challenge[5] on Kaggle. The dataset includes Wikipedia comments labeled by human raters for toxic behavior. Also, there are two separate training and test datasets.

The datasets are labeled binary for toxic behavior, and the level of toxicity is defined according to six different categories. The attributes of these datasets include

- id
- Comment text
- Toxic
- Severe toxic
- Obscene
- Threat
- Insult
- Identity hate.

The first few rows of the training dataset (already stored in a Pandas DataFrame) dataframe are shown in Figure 1.

[5] https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my username Hardcore Metallica Fan were reverted? They weren't vandalisms, just closure on some GAs after I ...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm seemingly stuck with. Thanks. (talk) 21:51, January 11, 2016 (UTC)	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It's just that this guy is constantly removing relevant information and talking to me through edits in...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on improvement - I wondered if the section statistics should be later on, or a subsection of ""types of...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember what page that's on?	0	0	0	0	0	0

Figure 1. Few rows of the train dataframe

The initial inspection reveals that there are no null cells in the dataset. The data type of the id and comment text features is "object". As mentioned earlier, the comments are labeled binary; therefore, the data type of corresponding columns in the train dataset is "int64", including 0 and 1. However, the labels of these columns in the test dataset include 0, 1, and -1. Therefore, the rows with the "-1" label are dropped. The first few rows of the test dataframe are shown in Figure 2.

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my username Hardcore Metallica Fan were reverted? They weren't vandalisms, just closure on some GAs after I ...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm seemingly stuck with. Thanks. (talk) 21:51, January 11, 2016 (UTC)	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It's just that this guy is constantly removing relevant information and talking to me through edits in...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on improvement - I wondered if the section statistics should be later on, or a subsection of ""types of...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember what page that's on?	0	0	0	0	0	0

Figure 2. Few rows of the test dataframe

The training and testing datasets consist of 159,571 and 63,978 rows, respectively. The details of data wrangling are available in the Data Wrangling notebook[6] for this project.

[6]https://github.com/ZAsaee/Toxic_Commnets_Classification/blob/main/Notebooks/01_Data_Wrangling.ipynb

2.2. Exploratory Data Analysis and Initial Findings

We performed the exploratory data analysis by investigating questions about the data, including the number of comments under each category, word and character count, and the most common words under each category.

The details of the EDA for the training[7] and testing[8] datasets are available in the linked notebooks..

2.2.1. Number of Comments under Each Category

In the training dataset, there are 159,571 rows consisting of 143,346 non-toxic and 16,225 toxic comments, respectively. Therefore, the ratio of toxic class to non-toxics is 10:90, and therefore the dataset is imbalanced. Please note that the toxic class includes six categories of toxic, severe toxic, obscene, threat, insult, and identity hate. Moreover, an individual comment may belong to more than one toxic category. Figure 3 shows the number of comments under each toxic category. The "toxic" category has the most significant number of comments. The number of comments in the "obscene and insult" categories is considerable, too.

[7]https://github.com/ZAsaee/Toxic_Commnets_Classification/blob/main/Notebooks/02-1-EDA_Train_Dataset.ipynb

[8]https://github.com/ZAsaee/Toxic_Commnets_Classification/blob/main/Notebooks/02-2-EDA_Test_Dataset.ipynb

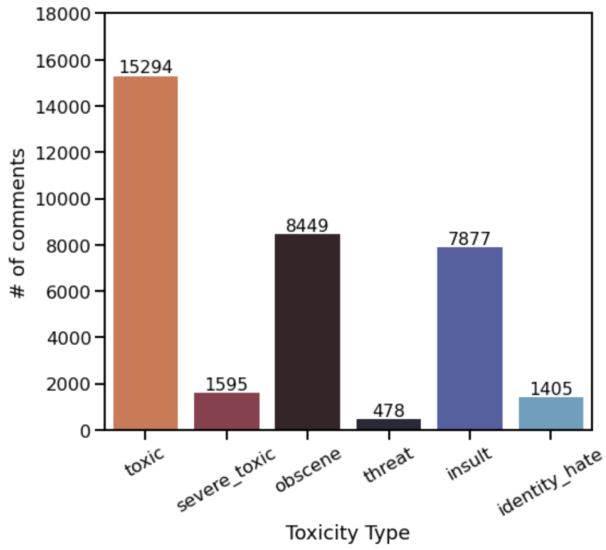


Figure 3. Number of comments for each toxicity type

The correlation analysis of the six toxic categories reveals the highest correlation between toxic, obscene, and insult classes. The threat category has the least correlation with the other categories.

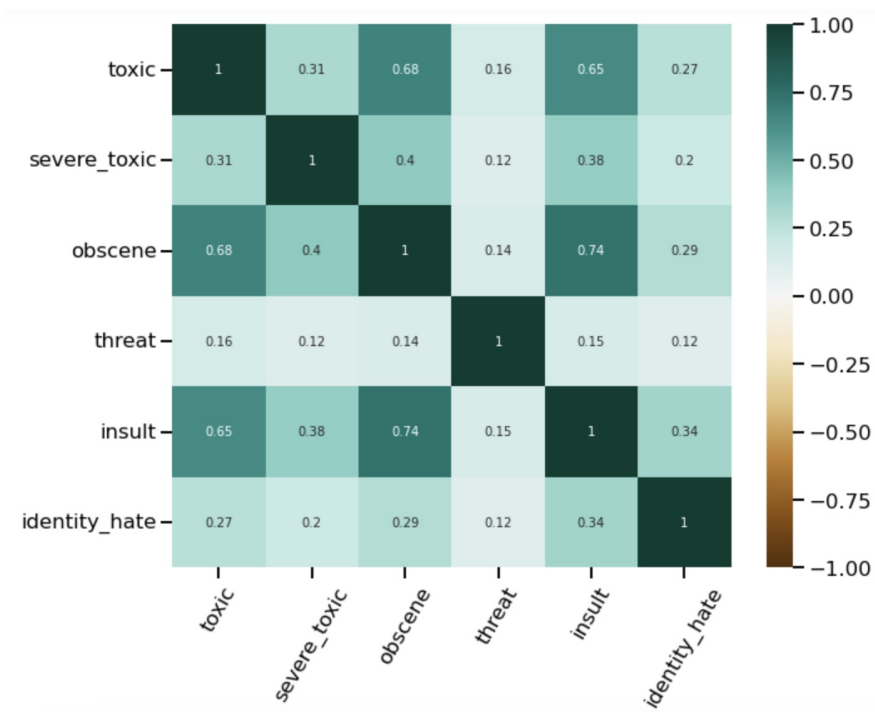
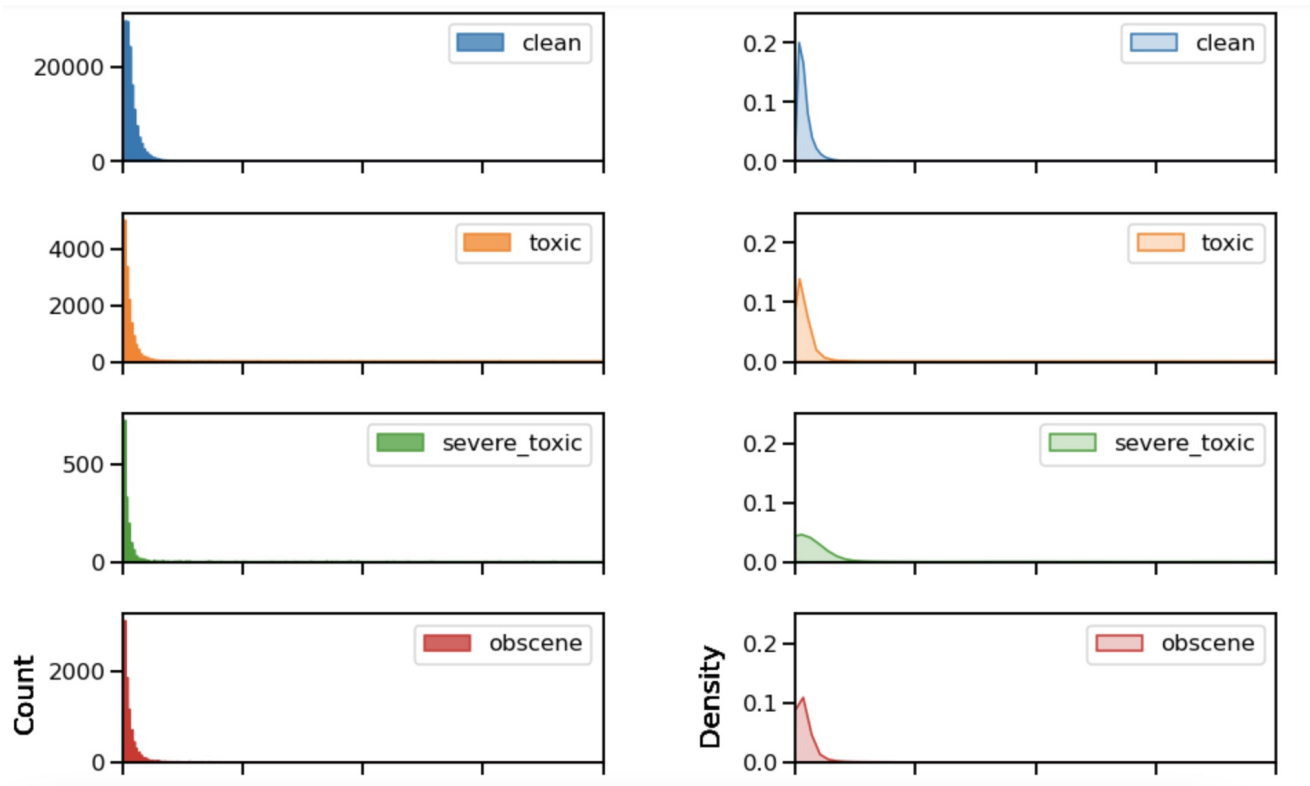


Figure 4. Correlation of six toxic categories

2.2.2. Sentence and Word Counts

Estimating the length of the text associated with a comment can give a great insight into the dataset. Figure 5 depicts the histogram of the sentence counts by toxicity type. All categories are positively skewed. In general, the toxic comments tend to be longer than the non-toxic comments.



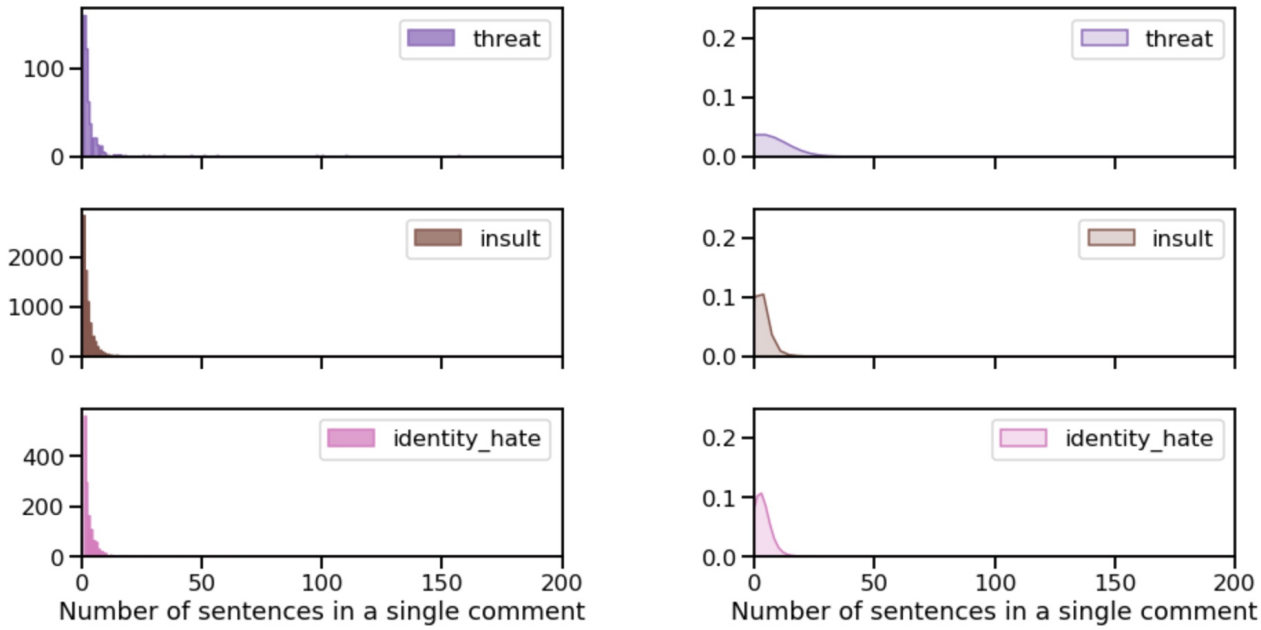


Figure 5. Count and density of number of sentences in a single comment based on the comment type

Figure 6 represents the histogram of the word counts by toxicity type. All categories are positively skewed, and no differences are observed from the analysis. However, comments that belong to the "severe toxic" and "threat" categories tend to include more words compared to the others (see Figure 6).

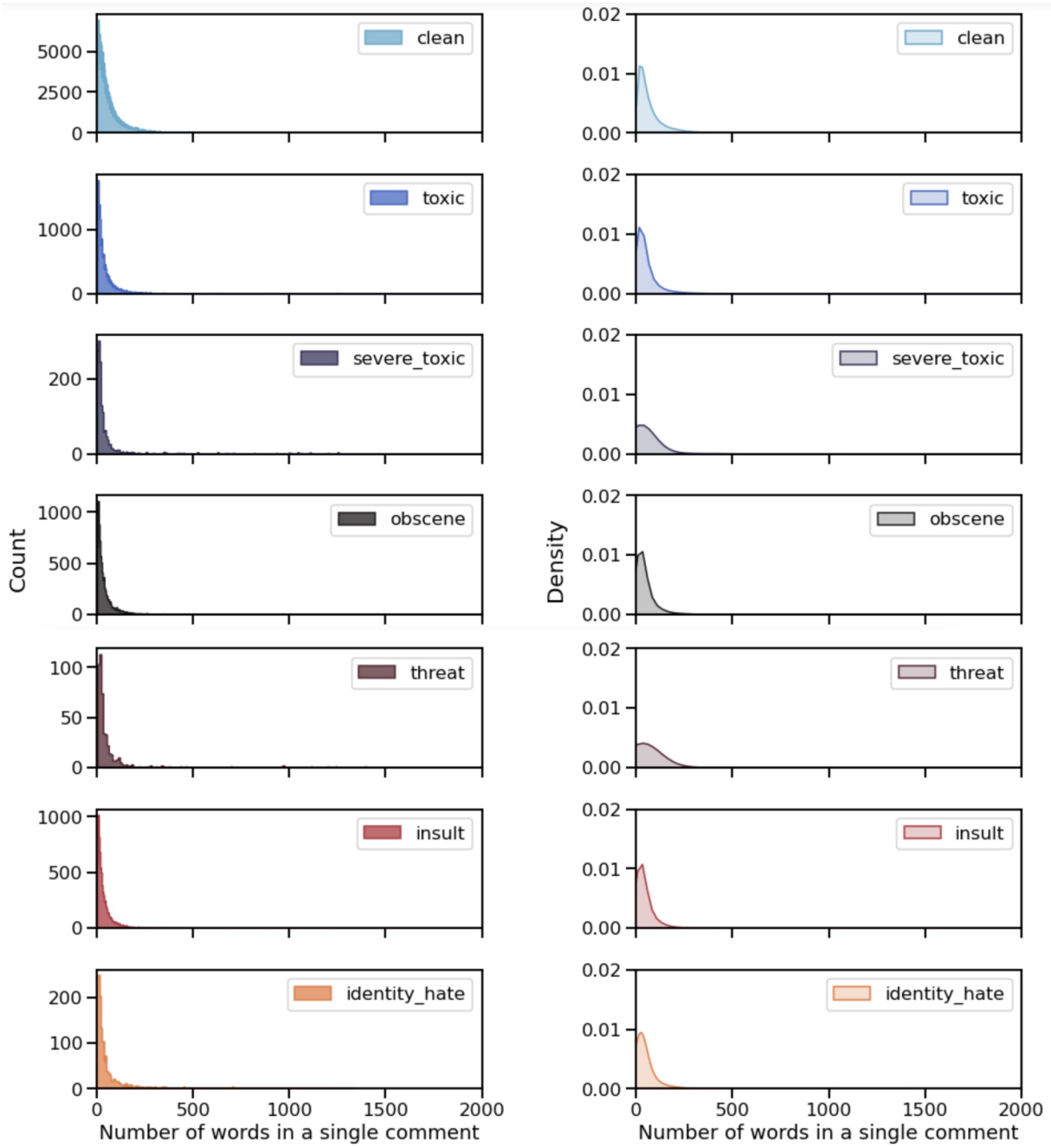


Figure 6. Count and density of a number of words in a single comment based on the comment type

2.2.3. Word Clouds

Word Cloud is a visualization technique for text information where each word is shown with font size and color that are proportional/associated with their

frequencies. Figure 7 depicts the word clouds consisting of the most common words in each class (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

Figure 7. Word cloud consists of the most common words in each class

(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

The word cloud results reveal that the text is noisy, containing abbreviations, repetitions, and pause-filling words. Therefore, we perform the following steps to

clean up the text: remove special characters and HTML tags, lemmatization, remove stop words, and case conversion. Figure 9 represents the word cloud after performing the text cleaning.

Figure 8. Word cloud consists of the most common words in each class after text cleaning

(**<u>Please see the Disclaimer and Warning in the Executive Summary of this project</u>**)

It is mentioned earlier that there is an overlap between the toxic categories. Therefore, all toxic categories are merged and considered a toxic class in this project.

As a result, the project is a binary classification: non-toxic and toxic.

2.2.4. Text Vectorization,

The input to supervised machine learning algorithms is a two-dimensional numerical matrix where rows and columns are data instances and features, respectively. Therefore, one common first step before applying any machine learning algorithm on text data is transforming text into a numerical vector. This process is known as text vectorization. There are several methods for text vectorization, such as absolute frequency (a.k.a. Bag of Words–or BOW), one-hot encoding, term frequency-inverse document frequency (TF-IDF), and distributed representation. In this project, I use BOW and TF-IDF methods.

The BOW method is to fill in the vector a given document with the frequency of each word in the document as it appears in the so-called dictionary associated with the aggregation of all the documents (a.k.a. corpus) . In this method, each document (*i.e.*, an individual comment in this project) is represented as a multiset of tokens with a value in the vector equal to its count. The frequency-based method considers each document in a standalone fashion and does not take into account the context of the corpus. On the other hand, the TF-IDF method considers the relative frequency of tokens in the document against their frequency in other documents. The main idea of the TF-IDF method is to accentuate tokens that are very relevant to a specific topic or instance.

2.3. Baseline Modeling

Classification predictive modeling is a category of problems where a class label is assigned to each data point in a given dataset. Common types of classification problems are binary classification, multi-class classification, multi-label

classification, and imbalanced classification. The current project is an instance of a binary imbalanced classification problem, where the labels are "toxic" and "non-toxic". The distribution of comments in these two labels is not balanced, as shown in Table 1.

Table 1. Number of comments in each class

	Label	Number of Comments
1	Non-toxic	143346
2	Toxic	16225

This is a class of problems that is challenging because the bias in the dataset is usually inherited by models when they are naively built. It usually requires the use of a high-level framework for systematically handling the skewed class distribution. In this project, I use the Logistic Regression algorithm, in conjunction with two text vectorization methods, in order to build baseline models. As part of this process, a proper performance metric must be picked before applying the models–in other words, a performance metric that captures the underlying business goals.

2.3.1. Performance Metrics

Selecting the right metric is one of the main steps for evaluating a model. There are mainly two factors for the choice of classification: imbalanced or balanced dataset and the business use-case to solve. In the case of imbalanced classification, selecting the appropriate metric is more challenging since most of the widely used standard metrics assume a balanced class distribution.

For example, classification accuracy is one of the most common standard metrics used to evaluate the performance of classification models. However, using this

metric for imbalanced classification problems is generally misleading, because it is artificially inflated by the performance of the model on data points that belong to the larger class (a.k.a. "majority class"), while the actual performance on data points that belong to the smaller class (a.k.a. "minority class") is poor.[9] That is why one would need to use more nuanced performance metrics, that can measure the performance of models with respect to both the minority and majority class–in the case of binary classification problems.[10]

There are four types of outcomes for the classification prediction: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Four performance metrics based on these outcomes are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

For binary imbalanced classification tasks, the majority class represents "the normal" (called the "negative class"), and the minority class represents "the exception" (called the "positive class"). As mentioned earlier, the accuracy metric is not an appropriate choice for imbalanced classification problems. The right

[9] Consider the classical example of a binary classification problem with a minority class representing 10% of the dataset. If a "lazy classifier" blindly classifies any given input as belonging to the majority class, then such a classifier would have a **global** accuracy of 90%! However, the same classifier would have an accuracy of 0% on the minority class. Since in many problems, the class of interest is usually the minority class, this lazy classifier would be utterly useless–despite the fact that its global accuracy is 90%!

[10] These performance metrics naturally generalize to more than two classes–for instance, by considering multiple "one-vs-all" problems, each of which is binary.

performance metric can be selected based on the client's identified tolerance level with respect to false positives and false negatives.

The current project assumes that a social media platform is willing to detect toxic comments and block the corresponding accounts. Therefore, it is essential to decrease false positives since we don't want to block an account by mistake (high Precision). On the other hand, it is crucial to detect toxic comments and block them (high Recall).

In practice, we can't achieve both high Precision and high Recall. An increase in Precision usually reduces Recall, and vice versa. This is called the precision/recall tradeoff. Therefore, the *f1*-score might be a better metric of performance when we are seeking a balance between precision and recall. In particular, the compromise is expressed in the form of the harmonic mean of precision and recall.

Moreover, the receiver operator characteristic curve, and the associated area under this curve (a.k.a. ROC/AUC) are two useful metrics to evaluate the performance of classification algorithms. For classification problems with probability outputs, a threshold can be used for the classification. ROC plots the sensitivity and specificity for the different thresholds and possible outputs.

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specifity = \frac{FP}{FP + TN}$$

In the current study, the performance metric to evaluate models are *f1*-score and ROC/AUC.

2.3.2. Logistic Regression - Baseline Model

We use Logistic Regression (without hyperparameter tuning) as the baseline model. Table 2 represents the summary of results for logistic regression in conjunction with BOW and TF-IDF techniques. The test score in this analysis is *f1*-score.

Table 2. Summary of the baseline model (Logistic Regression in conjunction with BOW and TF-IDF for the test dataset)

	BOW Term Frequency	***TF-IDF***
Mean fit time	*10.43*	*3.05*
Mean test score	*0.64*	*0.68*

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.95     57298
           1       0.57      0.73      0.64      6243

    accuracy                           0.92     63541
   macro avg       0.77      0.84      0.80     63541
weighted avg       0.93      0.92      0.92     63541
```

Figure 9. Classification Report for BOW and logistic regression model for the test dataset

```
              precision    recall  f1-score   support

           0       0.97      0.95      0.96     57298
           1       0.62      0.76      0.68      6243

    accuracy                           0.93     63541
   macro avg       0.80      0.85      0.82     63541
weighted avg       0.94      0.93      0.93     63541
```

Figure 10. Classification Report for TF-IDF and logistic regression model for the test dataset

The TF-IDF model is three times faster than the BOW model and has a higher performance metric (for both classes) compared to BOW; therefore, TF-IDF is used for the next models. Moreover, the low precision of the baseline model shows a large number of false positives (see Figures 9 and 10). Therefore, the goal in the next step is to improve it by testing alternative algorithms.

2.4. Extended Modeling

The goal of extended modeling is to increase the *f1*-score (decrease the false positives). As mentioned earlier, this dataset is imbalanced, requiring a high-level framework for systematically handling the skewed class distribution. In this project, we follow the systematic framework below to evaluate various classification methods. This framework consists of several classification algorithms as listed below.

Also, the ensemble stacking model, which is a meta-learning algorithm, and a combination of the predictions from several models are employed. Moreover, we use the resampling technique, including oversampling and undersampling, to balance the dataset. The XGBOOST and LGBM algorithms have a parameter to balance the dataset. We perform the resampling for these two algorithms using this parameter. Finally, we use BayesSearchCV to perform hyper-parameter optimization.

- Evaluated classification algorithms including:
    - Logistic Regression[11]
    - Random Forest Classifier[12]
    - XGBOOST Classifier[13]

[11] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
[12]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html?highlight=random%20forest#sklearn.ensemble.RandomForestClassifier
[13] https://xgboost.readthedocs.io/en/stable/python/python_api.html

    - LGBM Classifier[14]
    - Naive Bayes[15]
    - Ensemble Stacking Models[16]

- Evaluated imbalance algorithms including:
    - Random Oversampling[17]
    - Synthetic Minority Oversampling (SMOTE)[18]
    - Random Undersampling[19]
    - Near Miss[20]

- Hyperparameter tuning used:
    - BayesSearchCV[21]

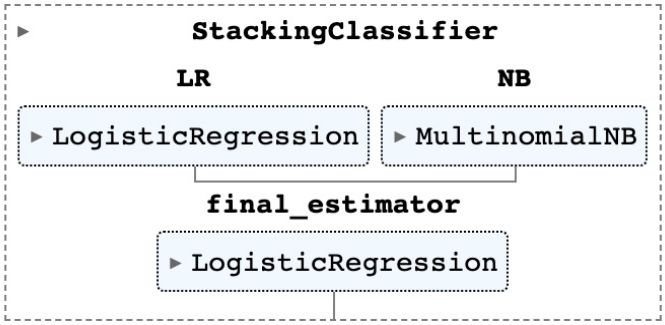
The metrics results after applying these models are shown in Tables 3 and 4.

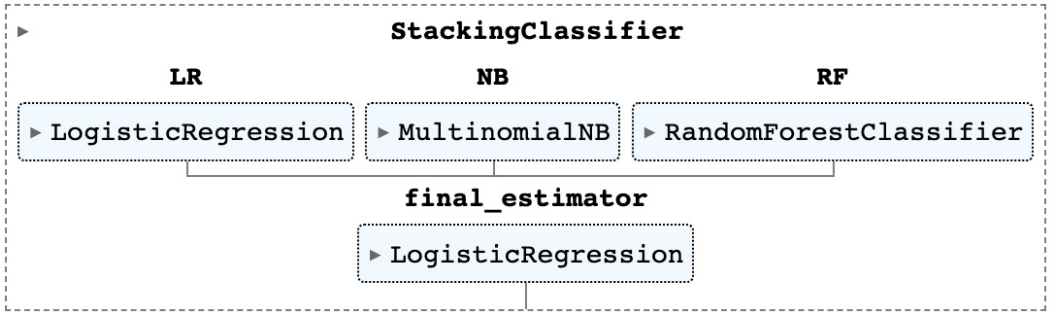
[14]https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html
[15] https://scikit-learn.org/stable/modules/naive_bayes.html
[16] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingClassifier.html
[17]https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html#imblearn.over_sampling.RandomOverSampler
[18]https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html
[19]https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html
[20]https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.NearMiss.html
[21]https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html

Table 3. Summary of results from extended models for the test dataset

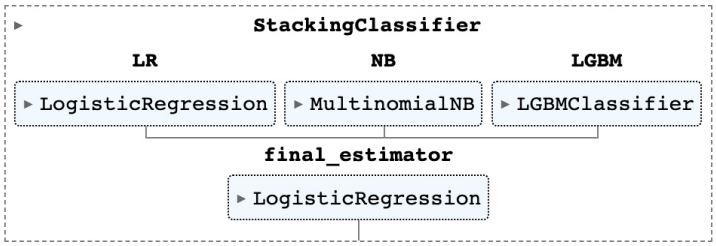
	Classification Algorithm	Resampling Technique	Non-Toxic Class			Toxic Class		
			Precision	Recall	f1	Precision	Recall	f1
1	Logistic Regression	None	0.98	0.93	0.96	0.57	0.81	0.66
2	Logistic Regression	Random Oversample	0.98	0.91	0.95	0.51	0.86	0.64
3	Logistic Regression	SMOTE	0.98	0.92	0.95	0.53	0.81	0.64
4	Logistic Regression	Random Undersample	0.98	0.91	0.95	0.52	0.86	0.64
5	Logistic Regression	NearMiss	0.98	0.91	0.95	0.52	0.86	0.64
6	Naive Bayes	None	0.96	0.97	0.96	0.70	0.60	0.65
7	Naive Bayes	Random Oversample	0.98	0.92	0.95	0.52	0.79	0.63
8	Naive Bayes	SMOTE	0.98	0.92	0.95	0.52	0.80	0.63
9	Naive Bayes	Random Undersample	0.98	0.92	0.95	0.53	0.79	0.63
10	Naive Bayes	NearMiss	0.98	0.92	0.95	0.53	0.80	0.64
11	Random Forests	None	0.98	0.93	0.96	0.57	0.81	0.66
12	Random Forests	Random Oversample	0.98	0.93	0.95	0.55	0.81	0.66
13	Random Forests	Random Undersample	0.96	0.96	0.96	0.62	0.64	0.63
14	XGBOOST	None	0.97	0.95	0.96	0.62	0.73	0.67
15	XGBOOST	Scale weight	0.97	0.94	0.96	0.59	0.75	0.65
16	LGBM	None	0.98	0.94	0.96	0.59	0.81	0.68
17	LGBM	Scale Weight	0.97	0.94	0.96	0.60	0.76	0.67

Table 4. Summary of results from ensemble stacking models

18	Logistic Regression and Naive Bayes				
					
Non-Toxic Class			**Toxic Class**		
Precision	**Recall**	**f1**	**Precision**	**Recall**	**f1**
0.98	0.94	0.96	0.58	0.80	0.67

19	Logistic Regression, Naive Bayes and Random Forests				
					
Non-Toxic Class			**Toxic Class**		
Precision	**Recall**	**f1**	**Precision**	**Recall**	**f1**
0.98	0.93	0.96	0.57	0.84	0.68

20	Logistic Regression, Naive Bayes and LGBM



Non-Toxic Class			Toxic Class		
Precision	**Recall**	**f1**	**Precision**	**Recall**	**f1**
0.98	0.94	0.96	0.58	0.81	0.68

3. Findings

3.1. Best Model

The LGBM model has the highest *f1*-score of 68% (see Table 3, row 16) among all the trained models. Figure 11 depicts the confusion matrix and precision-recall curve of this model. In this model, the number of false positives is greater than false negatives.

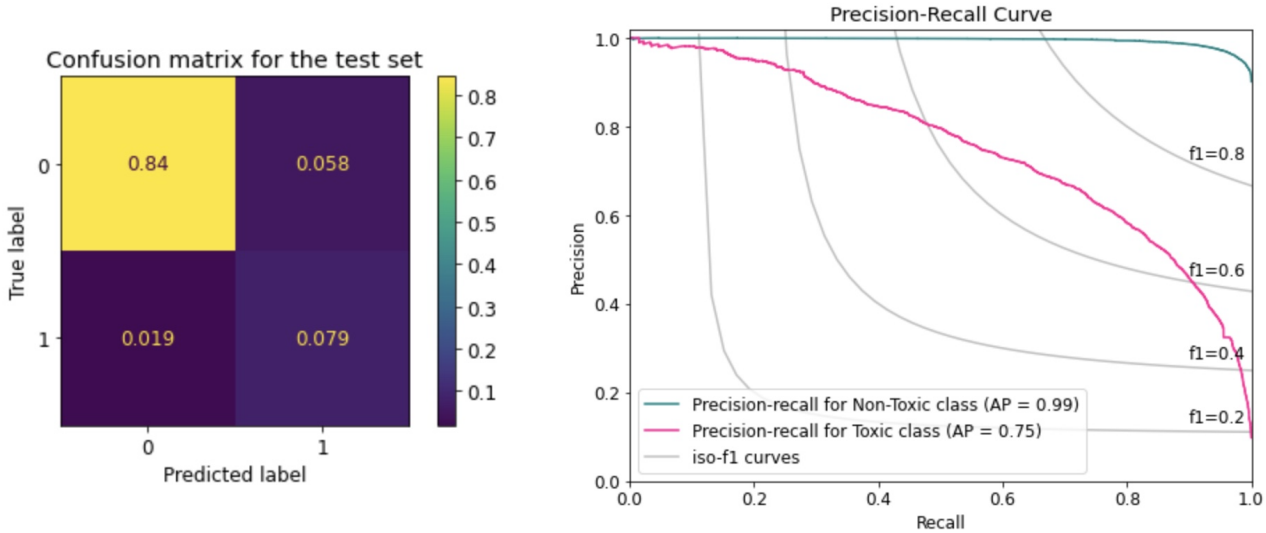


Figure 11. Confusion matrix and precision-recall curve of LGBM model with 68% f1-score

Across all the trained models, the Logistic Regression models with resampled dataset have the largest recalls. The logistic regression with the random undersampling technique has 86% recall. Figure 12 depicts the confusion matrix and precision-recall curve of this model. In general, resampling techniques (regardless of whether it is under- or over-sampling) for all trained models increase the recall and decrease the precision.

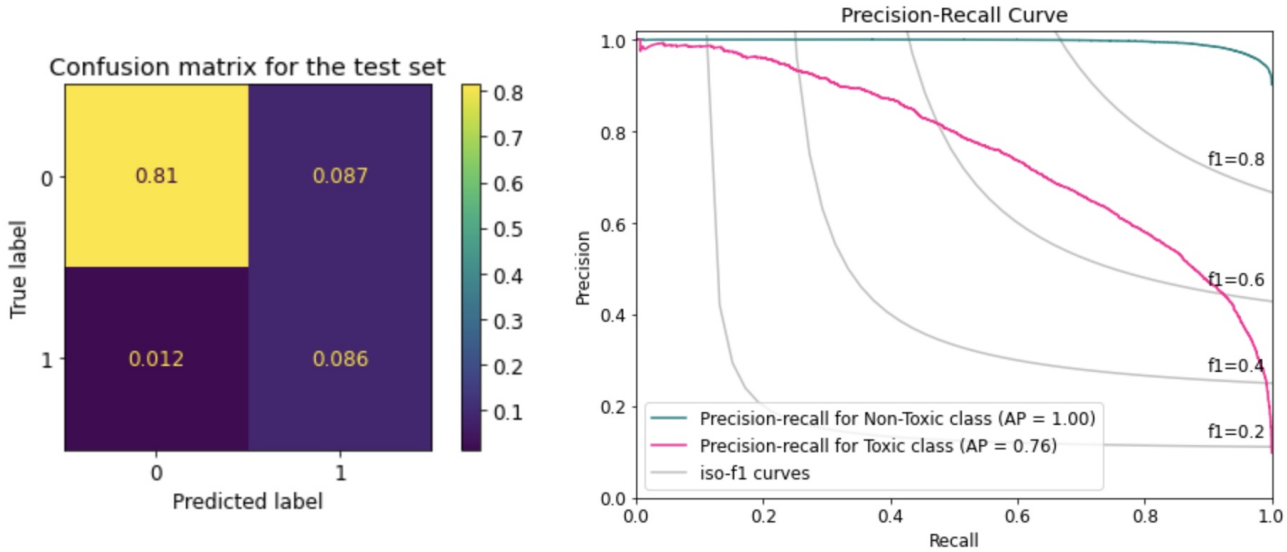


Figure 12. Confusion matrix and precision-recall curve of logistic regression and random undersampling model with 86% recall

The Logistic Regression with Random Undersampling model has higher false positives and lower false negatives compared to the LGBM model. The Naive Bayes model has the highest precision of 70% amongst all trained models. Figure 13 depicts the confusion matrix and precision-recall curve of this model.

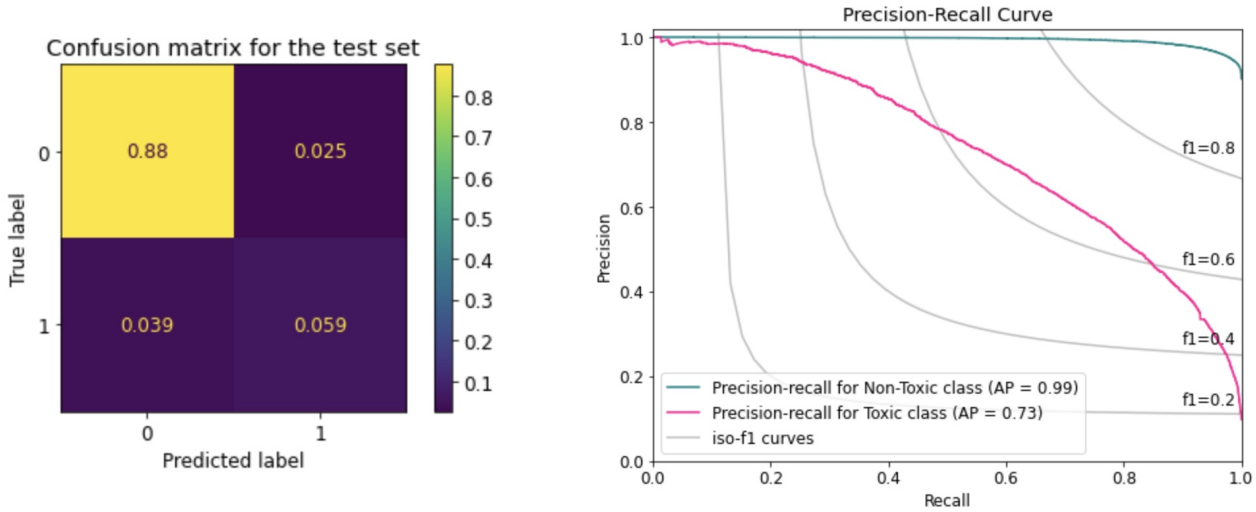


Figure 13. Confusion matrix and precision-recall curve of Naive-Bayes model with 70% precision

Based on the above plots, we can conclude that those three models, including LGBM, Logistic Regression, and Naive Bayes have the best performance in terms of *f1*-score, recall, and precision, respectively. As mentioned earlier, the target performance metric in this study is the *f1*-score since we are interested in increasing recall but keeping precision high, too. Therefore, we concluded that the LGBM model is the best model among all trained models.

3.2. Feature Importance/Impact Analysis

In this project, we use two methods to perform the feature importance/impact analysis:

- Directly interpretable models (Logistic Regression and Random Forest)
- Interpretable machine learning methods (SHAP).

Inherent interpretable models are based on how the model exactly functions (such as Logistic Regression and Random Forest). However, the interpretable machine learning methods use model-agnostic techniques that can be used to interpret the results of a model. These techniques assume the model as a black box and try to explain the relationship of features with the model predictions.

The interpretable machine learning method used in this study is SHapley Additive exPlanations (SHAP[22]). SHAP is an approach based on Game Theory that can be used to interpret the output of machine learning models. SHAP interprets the predictions based on the calculation of Shapley values, for some types of models–like XGBOOST, for instance.

3.2.1. Standardized Feature Importance of Logistic Regression

The first 60 important features for non-toxic and toxic classes based on the Logistic Regression without resampling model are shown in Figure 14. The negative and positive coefficients belong to the non-toxic and toxic classes, respectively. The coefficients of features impacting the toxic class are considerably larger–three to five times–than those for the non-toxic class. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

[22] https://shap.readthedocs.io/en/latest/index.html

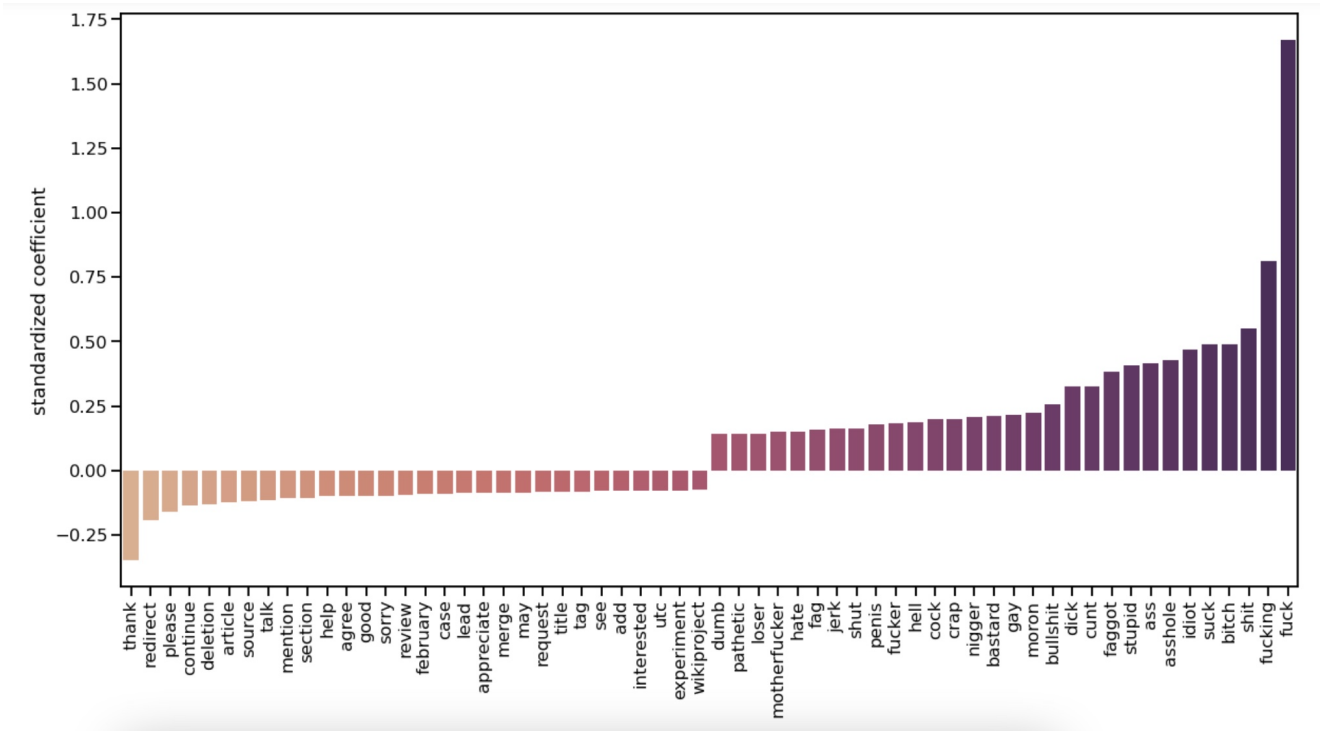


Figure 14. Feature importance analysis of Logistic Regression
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

3.2.2. Feature Importance of Random Forests

The first 60 important features for both classes based on the Random Forests model without resampling are shown in Figure 15. The features from the Random Forests model are not separated based on their impact on the class. However, the first few features with a considerable value impact detection of the toxic class. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

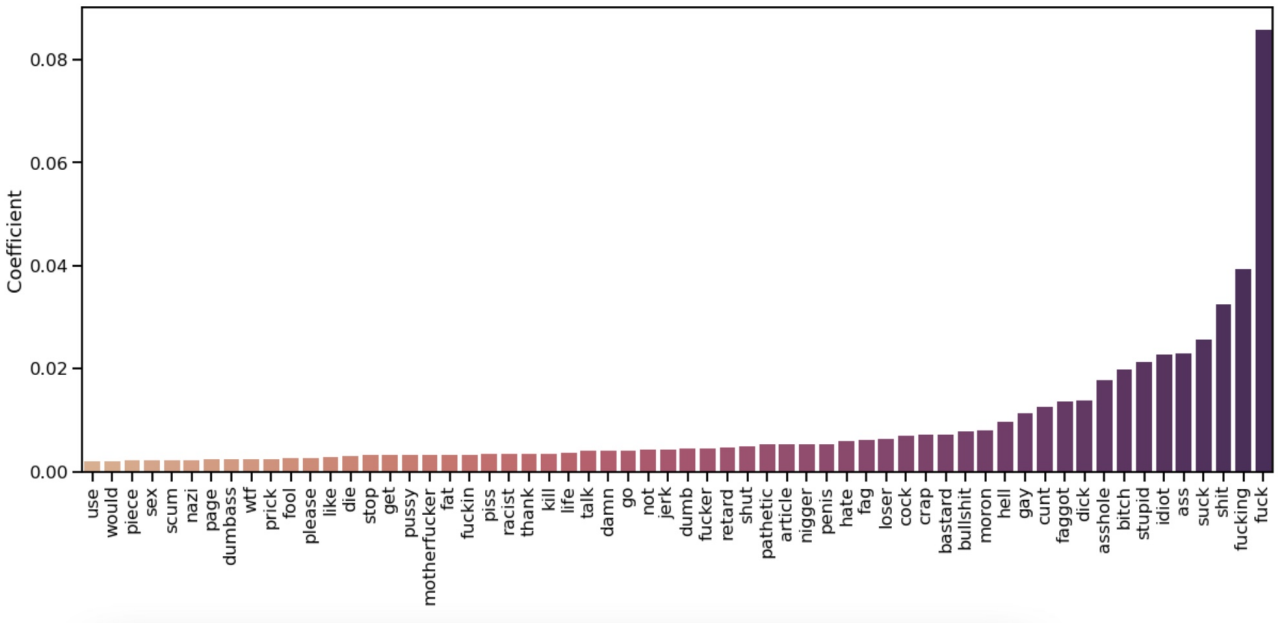


Figure 15. Feature importance analysis of Random Forest model
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

3.2.3. Feature Importance Using SHAP Technique

3.2.3.1. Random Forests Model

Figure 16 represents the SHAP summary Beeswarm plot for the Random Forest model. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

The positive SHAP values increase the likelihood of positive classification–meaning: toxic, while negative values diminish this likelihood, and are thus associated with the non-toxic class.

The SHAP value of features detecting the toxic class is considerably larger than those impacting the non-toxic class. In terms of the features of the non-toxic class, the larger feature value usually results in a higher SHAP value.

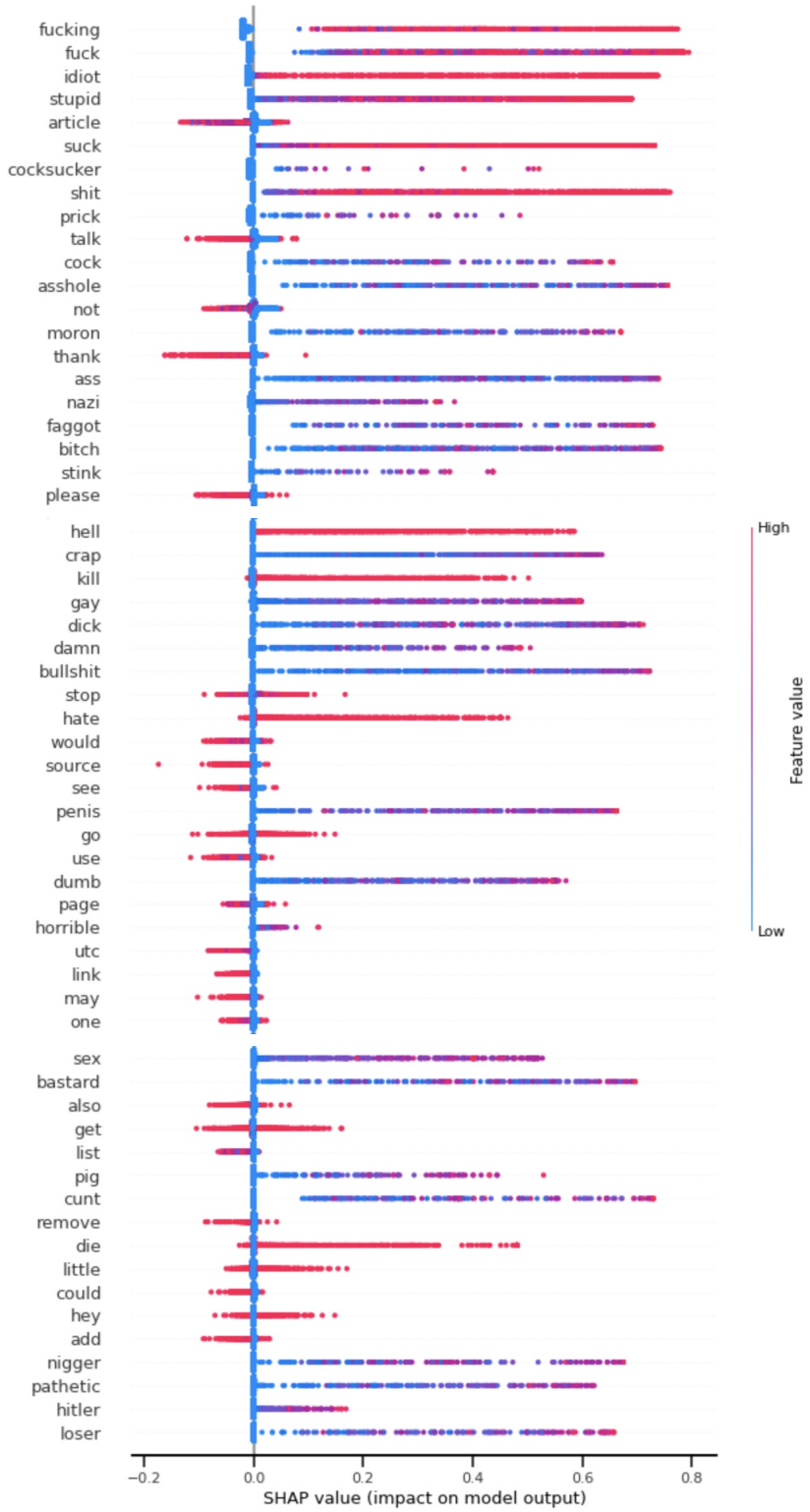


Figure 16. SHAP Swarm plot random Forest model

(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

Figure 17 depicts the scatter plots for two features impacting the non-toxic and toxic classes. (**Please see the Disclaimer and Warning in the Executive Summary of this project**). The TF-IDF score of the "f-ing" feature generally increases the SHAP value; however, the pattern of the plot is sparse. The zero TF-IDF score of this feature results in a shift towards the negative SHAP value.

Regarding the "thank" feature, the zero feature value shifts the SHAP value towards the positive SHAP value.

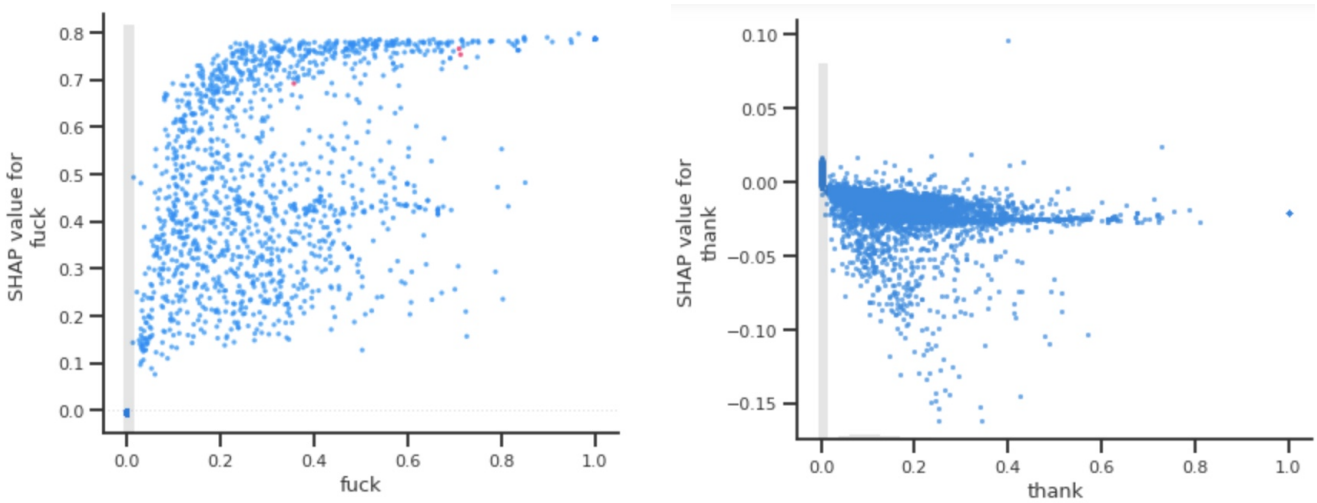


Figure 17. SHAP value vs. the feature value of two features impacting the toxic and non-toxic class detection
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

SHAP waterfall plot displays how the SHAP values of each feature shift the model output from the prior expectation to the final model prediction. Waterfall plots for some of the non-toxic and toxic comments are as follows. Figure 18 represents the SHAP waterfall plot of a non-toxic comment. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

The zero value of some features such as “f-ing”, “i-t”, and “f-k" move the output towards the negative value. The “f-ing” feature with zero value has the largest SHAP value for this comment.

```
energy edit introduction previously say passive transport not use kind energy not true passive transport rely kinetic
energy substance transport kinetic energy cause move around random chance cross membrane difference active transport
actually use cell energy atp electrochemical gradient pump substance across membrane
Class: 0
```

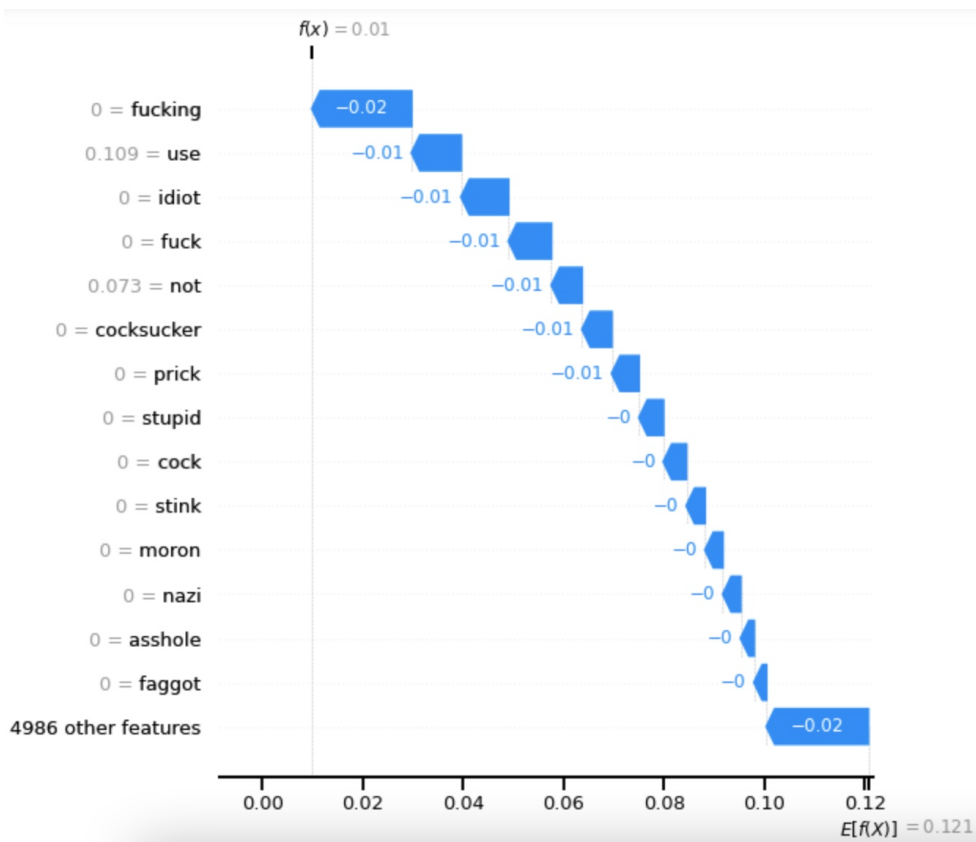


Figure 18. SHAP waterfall plot of a non-toxic comment
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

Figure 19 represents the SHAP waterfall plot for a non-toxic comment interpreted as toxic. The plot reveals that the “c-p” feature considerably impacts the output, shifts it towards the positive SHAP value, and misclassified the comment.

```
somebody invariably try add religion really mean way people invariably keep add religion samuel beckett infobox bothe
r bring long dead completely non existent influence issue flail make crap fly comparison explicit acknowledgement ent
ire amos oz article personally jewish category
Class: 0
```

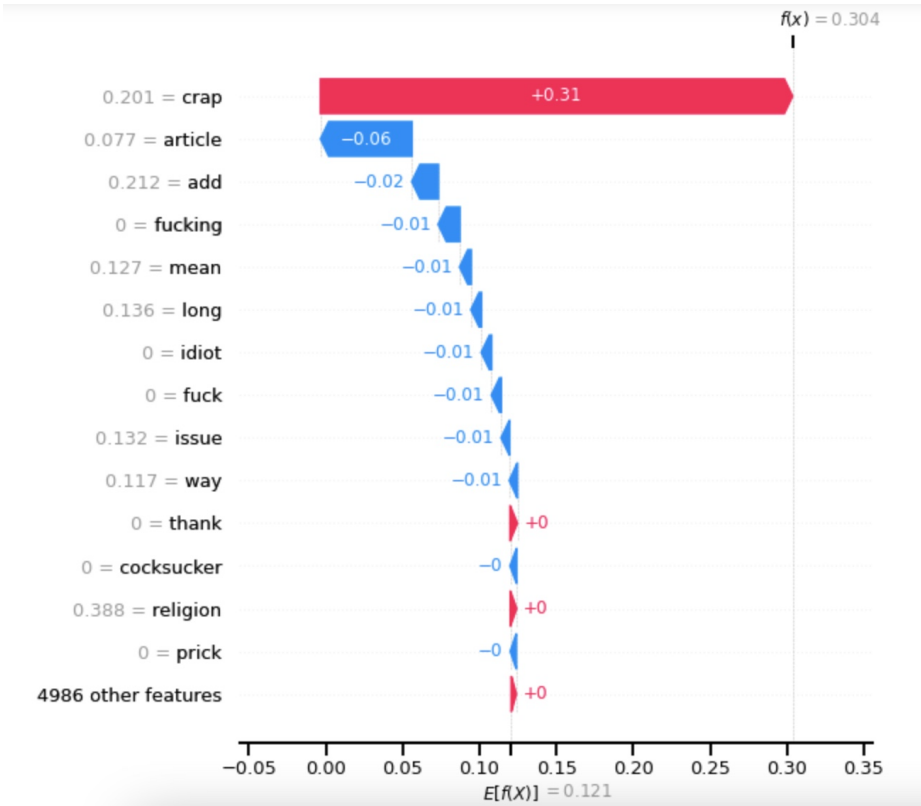


Figure 19. SHAP waterfall plot of a non-toxic comment misinterpreted as toxic
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

Figure 20 depicts the SHAP waterfall plot of a toxic comment interpreted as toxic. The "h-ll" feature drastically moves the output towards the positive value. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

```
arabs commit genocide iraq no protest europe may europe also burn hell
Class: 1
```

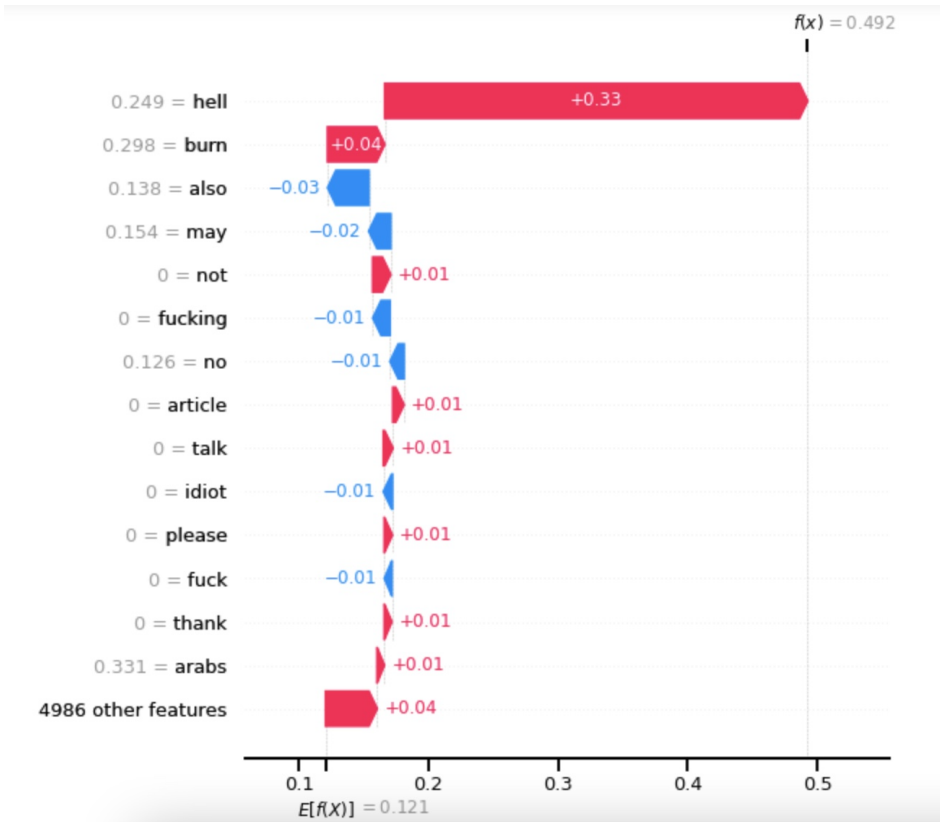
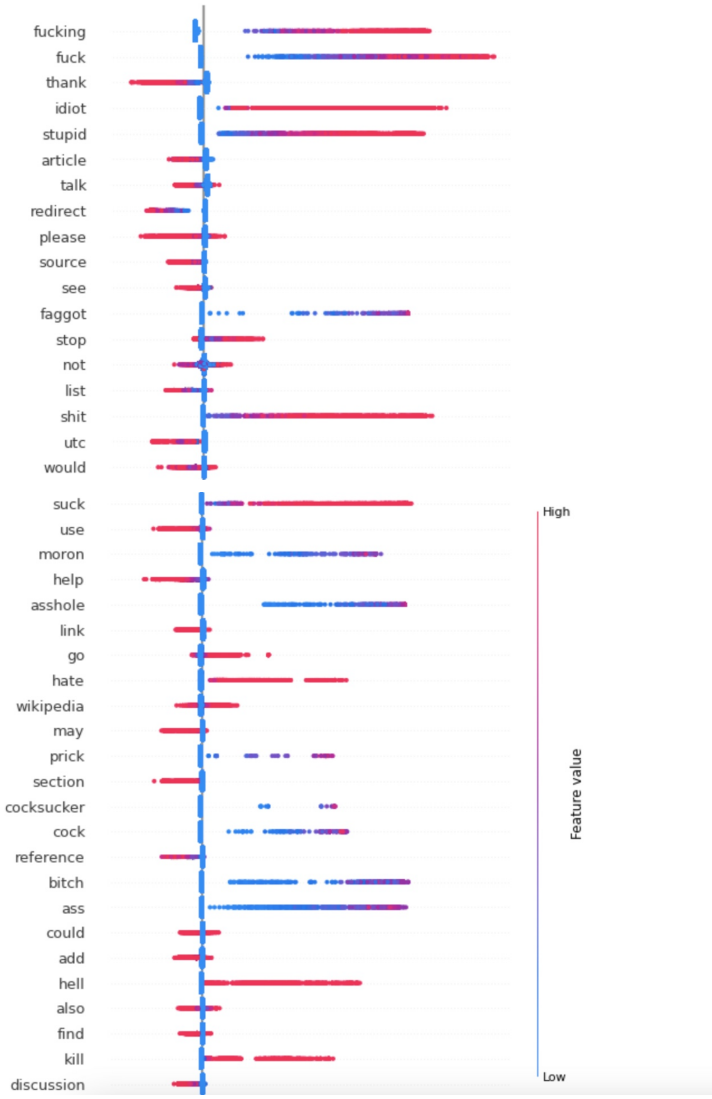


Figure 20. SHAP waterfall plot of a toxic comment interpreted as toxic
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

3.2.3.2. XGBOOST Model

Figure 21 represents the SHAP summary Beeswarm plot for the XGBOOST model.(**Please see the Disclaimer and Warning in the Executive Summary of this project**).

The positive SHAP values interpret the toxic class, whereas the positive values detect the non-toxic class. Similar to the random forests model, the SHAP value of features detecting the toxic class is considerably larger than those impacting the non-toxic class. The most important features of the XGBOOST model are similar to those of the random forests model (see Figure 16).


fucking
fuck
thank
idiot
stupid
article
talk
redirect
please
source
see
faggot
stop
not
list
shit
utc
would
suck
use
moron
help
asshole
link
go
hate
wikipedia
may
prick
section
cocksucker
cock
reference
bitch
ass
could
add
hell
also
find
kill
discussion
High
Feature value
Low

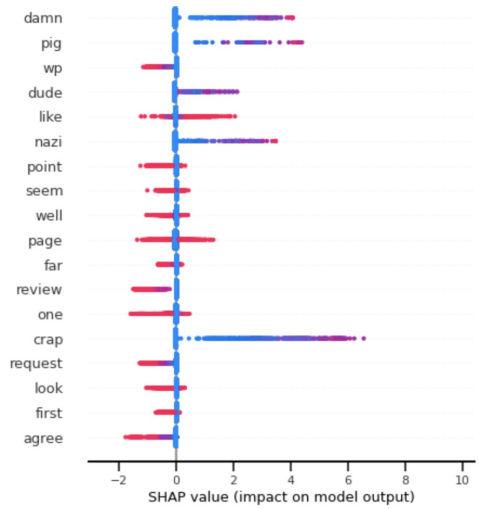


Figure 21. SHAP summary plot of XGBOOST model
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

The scatter plots of SHAP values vs. feature values for two features impacting the toxic and non-toxic classes are shown in Figure 22. (**Please see the Disclaimer and Warning in the Executive Summary of this project**).

For both features, an increase in the feature value results in an increase in the SHAP value. Also, the pattern is not as sparse as in the previous analysis of the random forest model. Moreover, the zero feature value shifts the output towards the opposite side for both features.

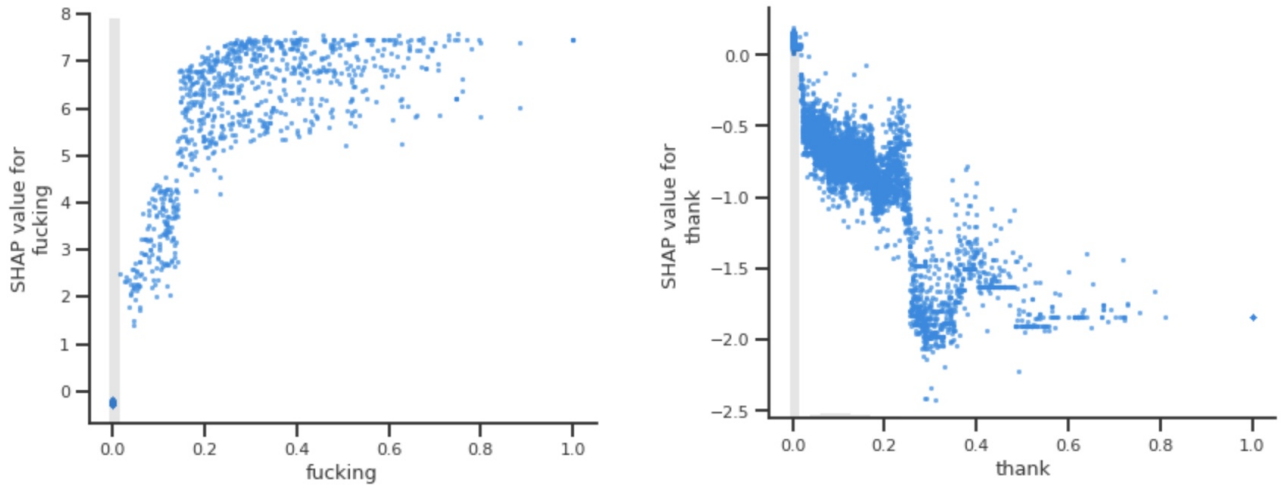


Figure 22. SHAP value vs. the feature value of two features impacting the toxic and non-toxic class detection
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

Figure 23 represents the SHAP waterfall plot for a non-toxic comment interpreted as non-toxic. This comment is misinterpreted by the previous model as toxic (see Figure 19). The plot reveals that the "c-p" feature moves the output to the positive value; however, the other features shift it towards the negative SHAP value and interpret it as non-toxic.

```
somebody invariably try add religion really mean way people invariably keep add religion samuel beckett infobox bothe
r bring long dead completely non existent influence issue flail make crap fly comparison explicit acknowledgement ent
ire amos oz article personally jewish category
Class: 0
```

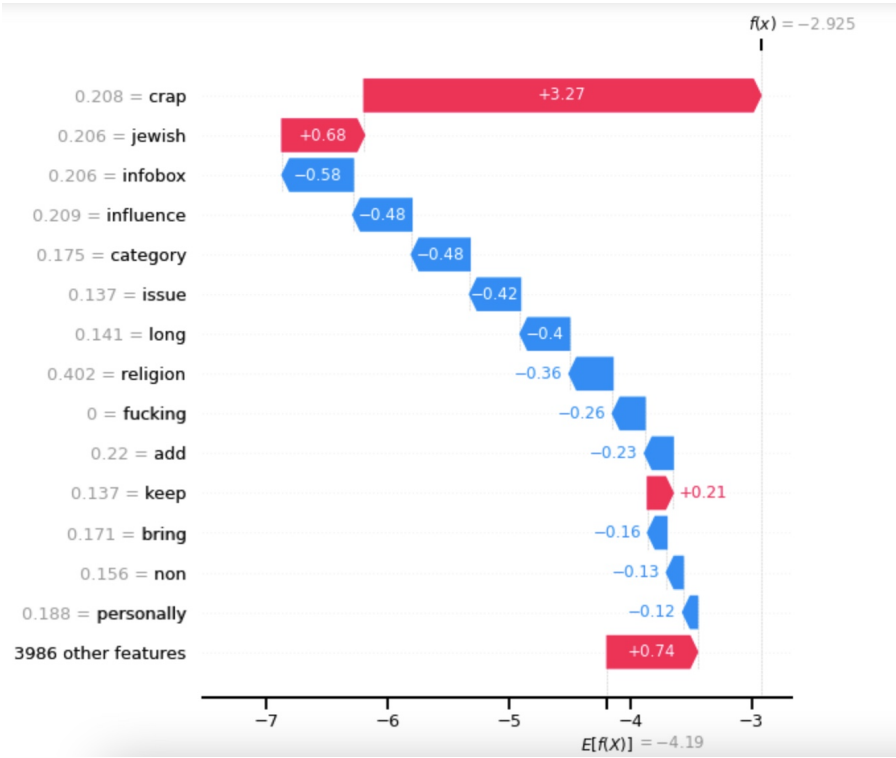


Figure 23. SHAP waterfall plot of a non-toxic comment interpreted as non-toxic
(**Please see the Disclaimer and Warning in the Executive Summary of this project**)

This analysis shows that the features (or words) impacting the model output as toxic have a larger SHAP value and help to interpret the toxicity of a comment.

4. Conclusions and Future Work

Through the exploratory data analysis, we identified that the toxic comments tend to be longer and contain more words compared to the non-toxic comments.

Several ML models (Logistic Regression, Random Forest, Naive Bayes, XGBOOST, LGBM Classifier, and Ensemble Stacking models) have been tested. Resampling techniques (Random Oversample, SMOTE, Random Undersample, and Near Miss)

have been used in combination with the mentioned classification algorithms to address the bias in the dataset.

Among all tested models, the LGBM classifier without resampling showed the best performance in terms of *f1*-score (0.68). However, the Logistic Regression in conjunction with Random Undersampling had 0.86 recall. Therefore, based on the importance of toxicity detection and application, either of these models is recommended.

Finally, the interpretability analysis showed that the features (in this case, proxies of words in the documents) impacting the model output as toxic have a considerably larger SHAP value and will help to interpret the toxicity of a comment. It was shown that the SHAP model could detect the toxicity of comments even with a small feature value.

As for future work, It is recommended to develop models using anomaly detection algorithms. We observed that the number of false positives was considerable, and an anomaly detection algorithm will help to identify the outliers. The other suggestion is to develop Recurrent Neural Network (RNN) models since RNN-based models perform well for text processing problems.

5. Recommendations for Clients

In creating recommendations for our results, it is essential to understand the application of this specific platform and the age range of users. In platforms with youth age range users, our goal is to block toxic comments as many as possible. Our recommended model can block 86% percent of the toxic comments. However, only 58% of the blocked comments are toxic. It means that 42% of the blocked comments

are clean and non-toxic. Moreover, our results prove a prevalence of some words with the toxicity of comments. The other recommendation can be blocking comments consisting of these words.

If we are willing to detect toxic comments and block the corresponding accounts, it is essential to decrease the false detections since we don't want to block an account by mistake. On the other hand, it is crucial to detect toxic comments and block them. In this case, our recommended model can detect 81% of the toxic comments, while only 41% of the blocked comments are clean and non-toxic.