COMMENTS CLASSIFIER PROJECT

DISCLAIMER: The report contains text that may be considered profane, vulgar, or offensive.

Submitted by:

ASHIN DILEEP T P

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my Mentors for their efforts to make me understand the concepts of Data Science, because of that I could able to complete this project.

I also thank the following websites, https://towardsdatascience.com/, https://machinelearningmastery.com/, https://www.analyticsvidhya.com/ and Wikipedia for their information on their webpages that helped me in completing this project.

INTRODUCTION

- Business Problem

The proliferation of social media enables people to express their opinions widely online. However, at the same time, this has resulted in the emergence of conflict and hate, making online environments uninviting for users. Although researchers have found that hate is a problem across multiple platforms, there is a lack of models for online hate detection.

Online hate, described as abusive language, aggression, cyberbullying, hatefulness and many others has been identified as a major threat on online social media platforms. Social media platforms are the most prominent grounds for such toxic behaviour.

There has been a remarkable increase in the cases of cyberbullying and trolls on various social media platforms. Many celebrities and influences are facing backlashes from people and have to come across hateful and offensive comments. This can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts.

Internet comments are bastions of hatred and vitriol. While online anonymity has provided a new outlet for aggression and hate speech, machine learning can be used to fight it. The problem we sought to solve was the tagging of internet comments that are aggressive towards other users. This means that insults to third parties such as celebrities will be tagged as unoffensive, but “u are an idiot” is clearly offensive.

Our goal is to build a prototype of online hate and abuse comment classifier which can used to classify hate and offensive

comments so that it can be controlled and restricted from spreading hatred and cyberbullying.

- Review of Literature

Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding.

Machine learning is a form of artificial intelligence which compose available computers with the efficiency to be trained without being veraciously programmed. Machine learning algorithms are broadly classified into three divisions, namely; Supervised learning, Unsupervised learning and Reinforcement learning. This project is of Supervised learning. Supervised learning is a learning in which we teach or train the machine using data which is well labelled that means some data is already tagged with correct answer. After that, machine is provided with new set of examples so that supervised learning algorithm analyses the training data and produces a correct outcome.

Text Classification is an example of supervised machine learning task since a labelled dataset containing text documents and their labels is used for training a classifier. An end-to-end text classification pipeline is composed of three main components:

1.Dataset Preparation: The first step is the Dataset Preparation which includes the process of loading a dataset and performing basic pre-processing. The dataset is then splitted into train and validation sets.

2. Feature Engineering: The next step is the Feature

Engineering in which the raw dataset is transformed into flat features which can be used in a machine learning model. This step also includes the process of creating new features from the existing data.

3. Model Training: The final step is the Model Building step in which a machine learning model is trained on a labelled dataset.

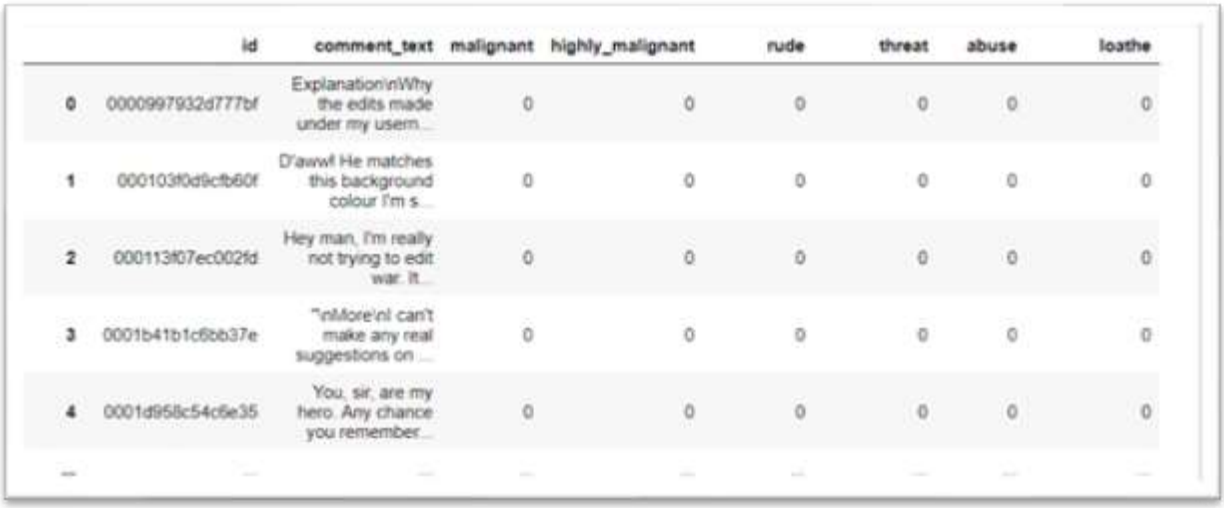
- The Objective for the Problem Undertaken

Our goal is to build a prototype of online hate and abuse comment classifier which can used to classify hate and offensive comments so that it can be controlled and restricted from spreading hatred and cyberbullying.

Analytical Solving Of Problem

- Data Description and it's Source

The data set contains the training set, which has approximately 80814 samples and the test set which contains 1,53,164 samples. All the data samples contain 8 fields which includes 'Id', 'Comments', 'Malignant', 'Highly malignant', 'Rude', 'Threat', 'Abuse' and 'Loathe'.
The label can be either 0 or 1, where 0 denotes a NO while 1 denotes a YES. There are various comments which have multiple labels. The first attribute is a unique ID associated with each comment.



	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'awww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\n\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...

The following are the features/attributes present in our dataset:

Malignant: It is the Label column, which includes values 0 and 1, denoting if the comment is malignant or not.
Highly Malignant: It denotes comments that are highly malignant and hurtful.

Rude: It denotes comments that are very rude and offensive.
Threat: It contains indication of the comments that are giving any threat to someone.
Abuse: It is for comments that are abusive in nature.
Loathe: It describes the comments which are hateful and loathing in nature.
ID: It includes unique Ids associated with each comment text given.
Comment text: This column contains the comments extracted from various social media platforms.

- Mathematical/ Analytical Modelling of the Problem

In this project we have used 'shape' function to check how many rows and columns are there in the dataset.
Then we have used 'isnull' function to check whether there are any null values in the dataset.
Then we have used 'bar plot' to check what is the number of comments under each label and the number of comments having multiple labels.
Then we have used 'Word Cloud'. Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance.

- Data Pre-processing

Certain label columns had text data other than 0 and 1 hence those data were replaced with NaN values. And some values i.e., 0 and 1 were in string format i.e., '0' and '1' hence those data were replaced with 0 and 1 values.

The dataset has missing values in it. Since those NaN values were not in huge numbers, those values were dropped from the dataset.

Then ‘emojis’ present in the comments were eliminated, all the comments were converted to lower case, replaced 'numbers' present in the comments with 'space', removed all the punctuations present in the comments, replaced 'wide-space' between terms/words with a ‘single space’, removed underscores, removed leading and trailing ‘wide-space’.Then converted label columns into ‘int’ Dtype.

After that removed 'stop words' and then stemmed words using Porter Stemmer.

After that the text (the comments) were converted into features using Tf-idf Vectorizer Model.

Same methods are used to clean the test dataset. After that ‘set functions’ has been used to retain only those features which have been used to train the model i.e., dropped features which are not used to train and test the models from test dataset and added features which were used in training and testing the models.

- Hardware and Software Requirements and Tools Used

Some of the essential hardware requirements are as follows Windows OS, CPU with i3/i5 processor and 4GB/8Gb RAM.

Some of the essential software that we need to do this project is ‘Jupyter notebook’ which is present on ‘Anaconda framework’.

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment.

Some of the packages and libraries that are required for this project are:

Pandas: Pandas are used to read our file from an excel/csv file, to load the read file into a pandas dataframe and manipulate it for further use.

Numpy: Numpy is used to convert our data into a format suitable to feed our models.

Matplotlib and Seaborn: Matplotlib and Seaborn are libraries which will be helpful for data visualizations.

NLTK: This library is used to import stopwords and Porter Stemmer so that we can remove stopwords from our text and perform stemming on the words present in the text.

Wordcloud: WordCloud is imported from wordcloud. Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance.

Sklearn: Feature_Extraction module, various Algorithms/Models and its Metrics are imported from Scikit-learn (sklearn) library.

Joblib: Joblib library is used to save the best model for production and for future prediction.

Model/s Development and Evaluation

- Methodology of problem solving

After data cleaning the comments are passed to Tf-idf Vectorizer model to convert the comments into features. Then the features are stored in 'x' variable and label columns are stored in 'y' variable. After that the x and y is split into train set and test set. After that the data is passed to models for training and testing.

After that pass the test dataset to best performing model i.e., Logistic Regression model, to predict the comments classes.

- Selected Algorithms

Since the problem is to build a machine learning model which can predict comments classes i.e., classification problem. Hence let's import algorithms/models such as Logistic Regression, SGDClassifier, K Neighbors Classifier, Decision Tree Classifier, Random Forest Classifier from sklearn library.

```
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

Before using machine learning algorithms/models we should always split our data into a training set and testing set.

Let's split our data into train set and test set using 'train_test_split' function. We can import train_test_split function from sklearn.model_selection. Let the test size be 25% and remaining 75% be train set.

- Run and Evaluating selected Algorithms/Models

We can now train our model. Since it is a multi-label classification problem, we can use One Vs Rest Classifier model. One-vs-rest (OvR for short, also referred to as One-vs-All or OvA) is a heuristic method for using algorithms for multi-class classification.

It involves splitting the multi-class dataset into multiple binary classification problems. A binary classifier is then trained on each binary classification problem and predictions are made using the model that is the most confident.

Let's run the code shown below and see which model gives us high 'accuracy score' because higher the 'accuracy score' better is the model performance.

```
for mod in mod_list:
    print(mod)
    ovr=OneVsRestClassifier(mod, n_jobs=-1) # OneVsRestClassifier object
    ovr.fit(x_train,y_train)
    y_pred=ovr.predict(x_test)
    mod_acc=accuracy_score(y_test,y_pred)
    print('Accuracy score:',mod_acc*100)
    print(classification_report(y_test,y_pred))
    print('**********************************************************************
```

After running the code shown above, we came to know that the accuracy score of models such Logistic Regression, SGDClassifier, K Neighbors Classifier, Decision Tree Classifier, Random Forest Classifier are 92, 91, 89, 88, 91, respectively. Hence, we can conclude that 'Logistic Regression' model is performing better with an accuracy score of 92%.

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
Accuracy score: 92.03504776991238
              precision    recall  f1-score   support

           0       0.92      0.62      0.74      1896
           1       0.54      0.25      0.34       203
           2       0.91      0.64      0.75      1044
           3       0.88      0.13      0.23        52
           4       0.79      0.51      0.62       979
           5       0.76      0.16      0.26       180

   micro avg       0.87      0.56      0.68      4354
   macro avg       0.80      0.38      0.49      4354
weighted avg       0.86      0.56      0.67      4354
 samples avg       0.05      0.05      0.05      4354
```

After that let's consider Logistic Regression model for hyper-parameter tuning to find out which parameters of Logistic Regression model can be used so that model's performance can be enhanced. Then after performing hyper-parameter tuning, we got these parameters which can be used on Logistic Regression model.

```
print(gscv1.best_params_,
      gscv2.best_params_)
```

```
{'estimator__dual': False, 'estimator__penalty': 'l2'} {'estimator__fit_intercept': True, 'estimator_
_solver': 'liblinear'}
```

Finally, after using these parameters on Logistic Regression model we got an accuracy score of 92.03%.

```
lr = LogisticRegression(penalty='l2', dual=False, fit_intercept=True, solver='saga')
ovr=OneVsRestClassifier(lr, n_jobs=-1) # OneVsRestClassifier object
ovr.fit(x_train,y_train)
y_pred=ovr.predict(x_test)
mod_acc=accuracy_score(y_test,y_pred)
print('Accuracy score:',mod_acc*100)
print(classification_report(y_test,y_pred))
```

```
Accuracy score: 92.03504776991238
```

- Key Metrics for Evaluation of Algorithms/Models

There are various metrics available to evaluate selected Algorithms. Some of the metrics used in this project are as follows:

'Accuracy score': Accuracy score is one of the best metrics to evaluate Algorithms, higher the 'accuracy score' better is the model/algorithm performance. It is simply a ratio of correctly predicted observation to the total observations.

'Cross val score': Cross-validation is a statistical method used to estimate the skill of machine learning models. This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k − 1 folds. Cross-validation score (Cross val score) is a metrics similar

to accuracy score where higher the 'cross val score' better is the model/algorithm performance.

```
# cross_val_score
cv_score=cross_val_score(ovr,x,y,cv=8).mean()
print('Cross_val_score:',cv_score*100)
```

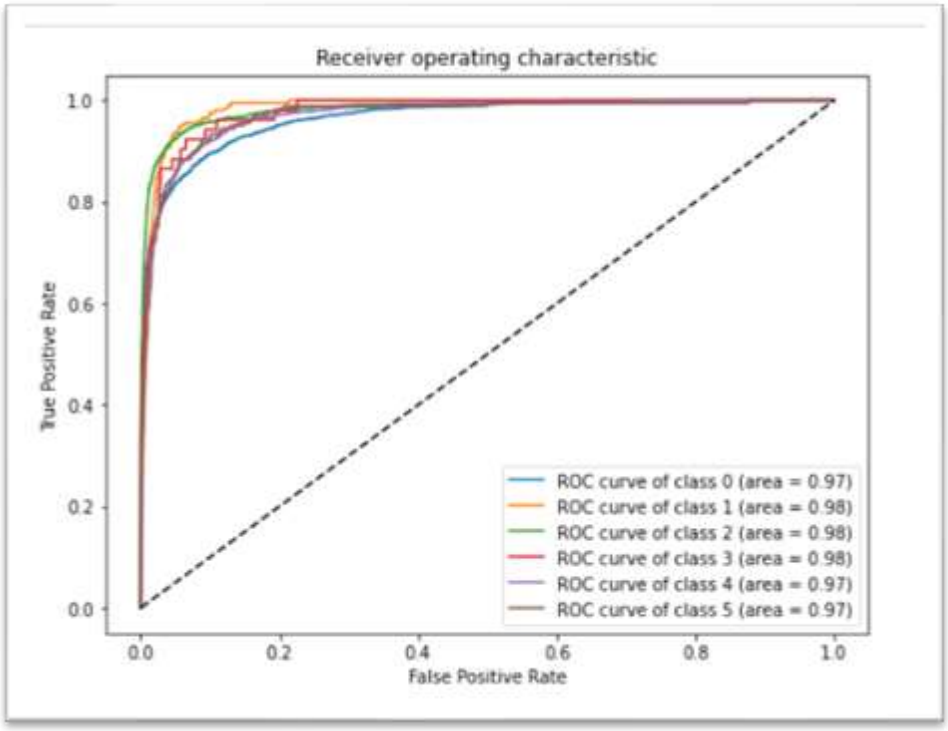
```
Cross_val_score: 91.75660421328739
```

'Classification Report': A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False? More specifically, 'Precision' and 'Recall' are calculated using True Positives, False Positives, True negatives and False Negatives.

0	0.92	0.62	0.74	1896
1	0.55	0.25	0.34	203
2	0.91	0.64	0.75	1044
3	0.88	0.13	0.23	52
4	0.79	0.51	0.62	979
5	0.75	0.15	0.25	180
micro avg	0.87	0.56	0.68	4354
macro avg	0.80	0.38	0.49	4354
weighted avg	0.86	0.56	0.67	4354
samples avg	0.05	0.05	0.05	4354

'Roc curve': Receiver operating characteristic curve (Roc curve) is a metrics which is nothing but graphical representation of confusion matrix. By plotting True Positive Rate (Recall) v/s False

Positive Rate (1 – Specificity) we get Roc curve. If the area under the curve is maximum, better is the model performance.



'Roc-Auc score': Roc-Auc score is a metrics similar to Accuracy score where higher the 'Roc-Auc score' better is the model/algorithm performance. In this project the Roc-Auc score is 0.97.

```
# roc_auc_score
ra_score = roc_auc_score(y_test,y_pred_proba,multi_class='ovr')
ra_score
```

```
0.9754174489160278
```

'Log Loss': Log Loss quantifies the accuracy of a classifier by penalising false classifications. Minimising the Log Loss is basically equivalent to maximising the accuracy of the classifier.

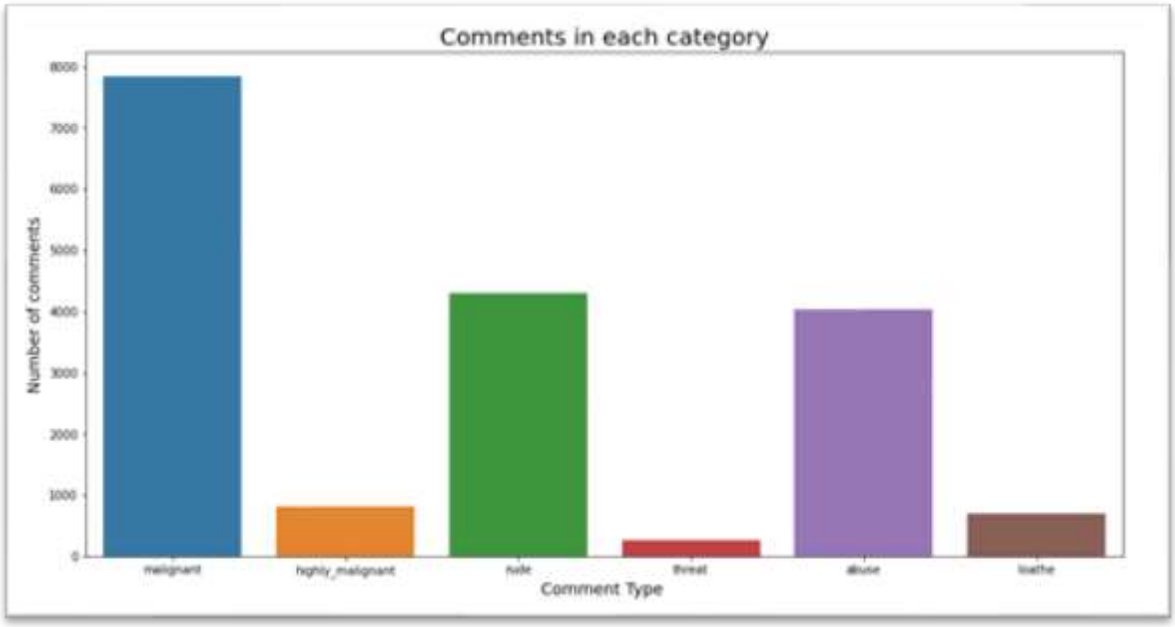
```
# importing log_loss
from sklearn.metrics import log_loss

# predicted probabilities
y_pred_proba = ovr.predict_proba(x_test)
# log_loss
ll=log_loss(y_test,y_pred_proba)
ll
```

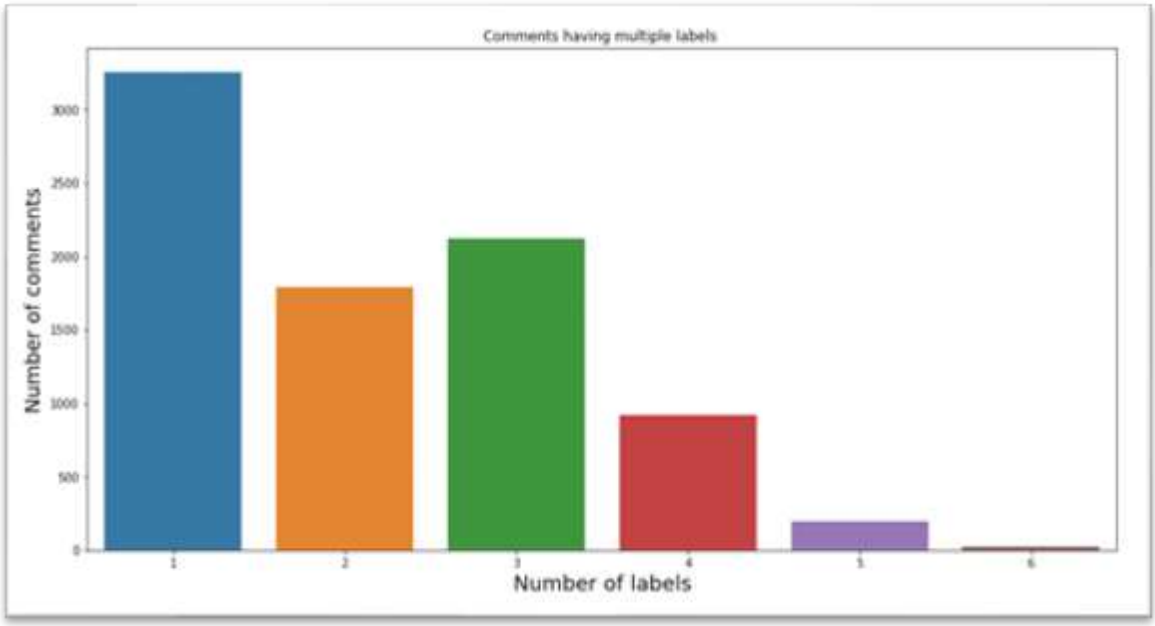
```
0.2764847721476803
```

- Visualizations/EDA

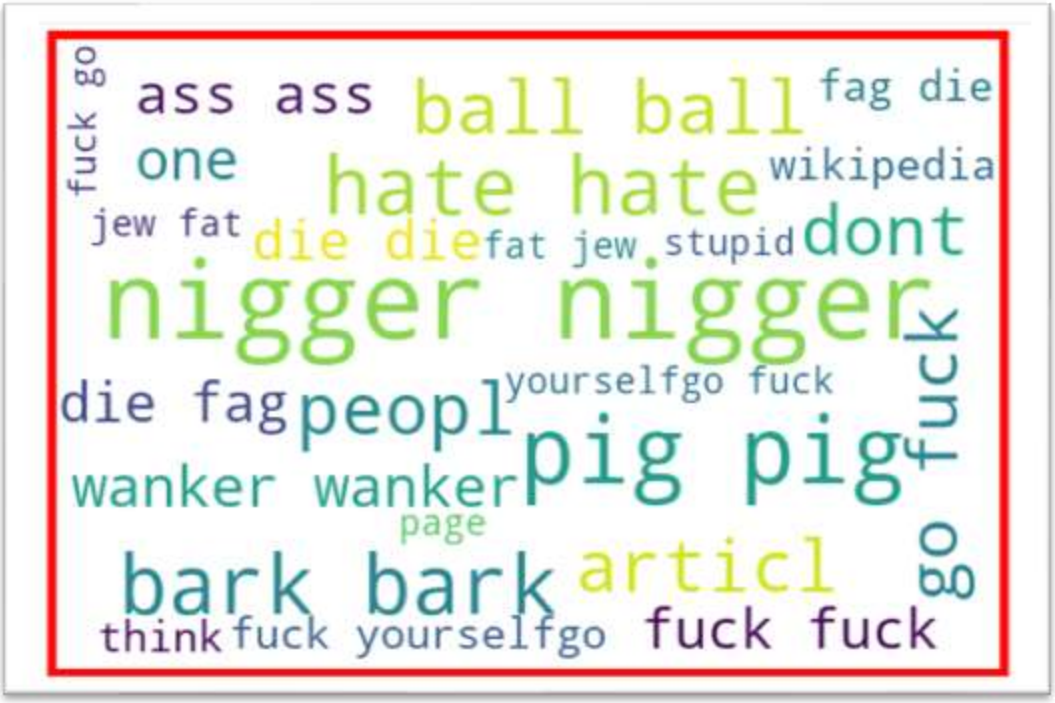
  Number of comments under each label.



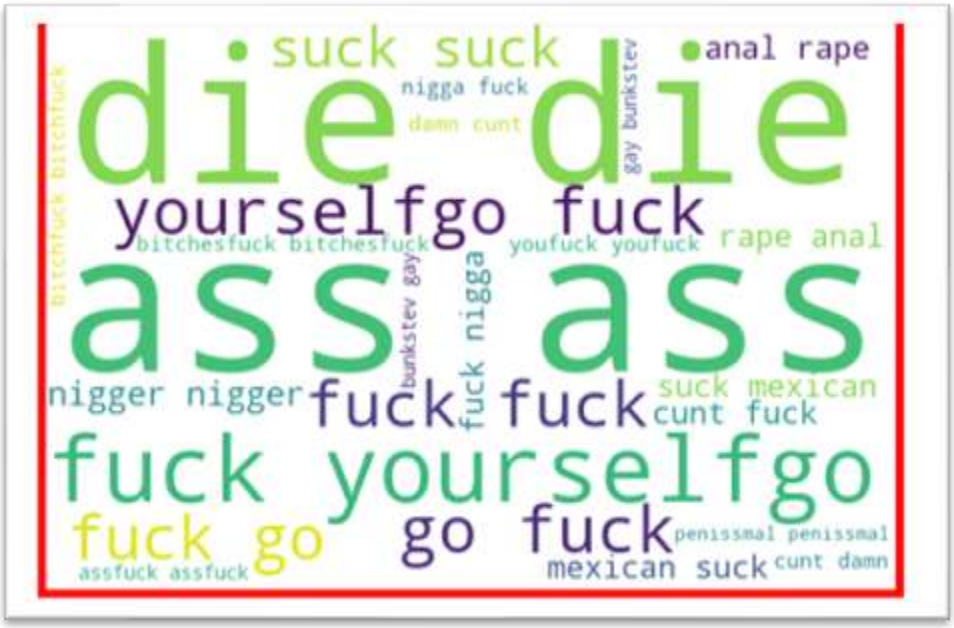
Number of comments having multiple labels.



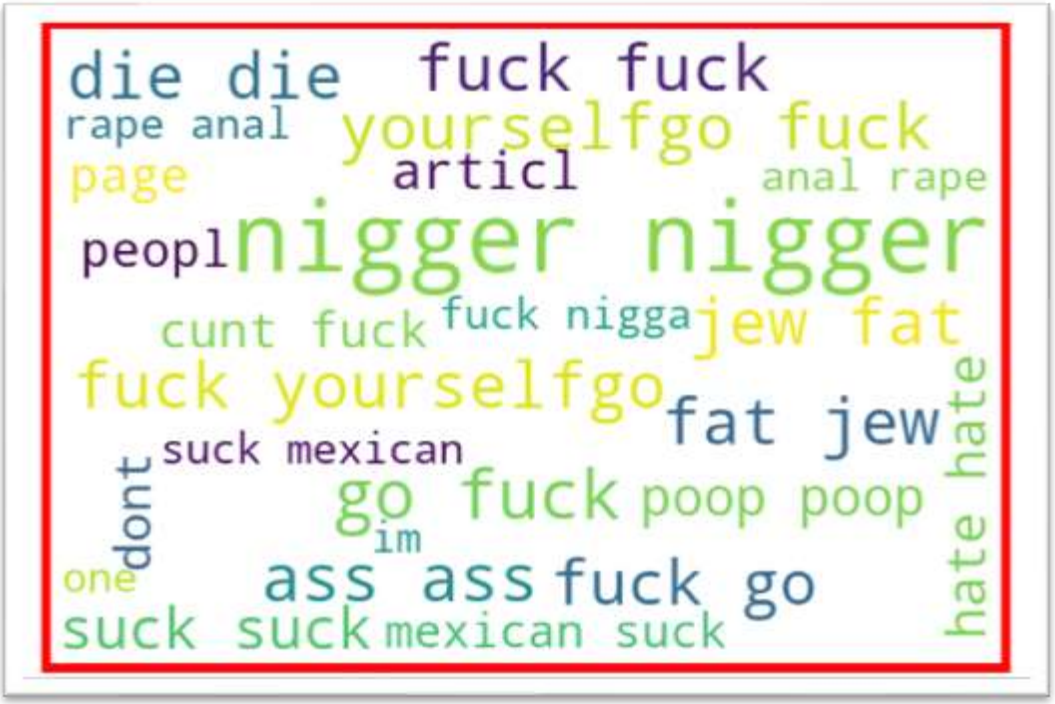
Cloud words in malignant comments.



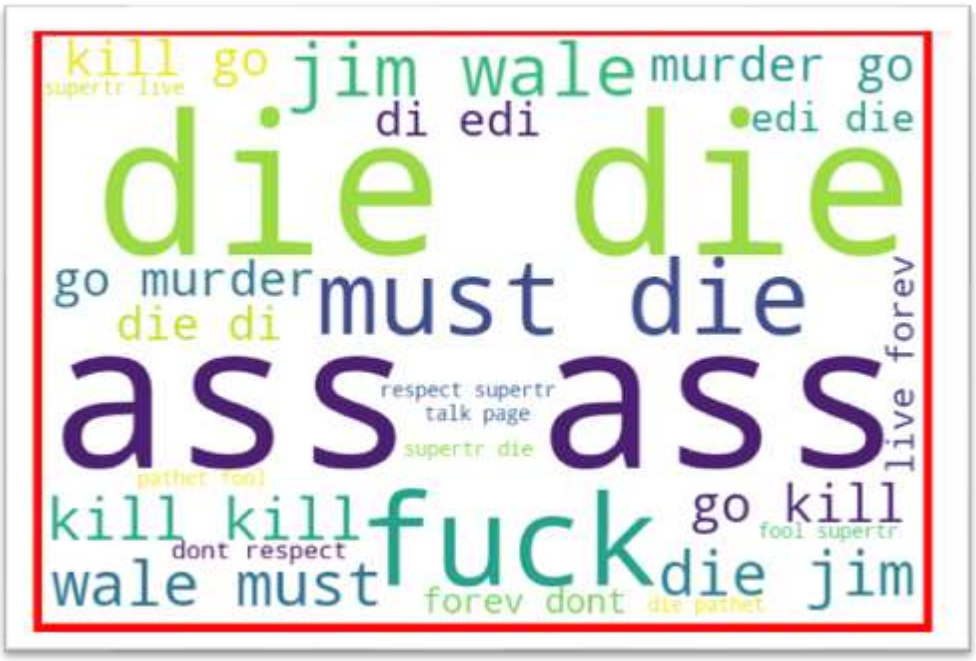
Cloud words in highly_malignant comments.



Cloud words in rude comments.



Cloud words in threat comments.



Cloud words in abuse comments.

Cloud words in loathe comments.

- Interpretation of the Results

After doing data cleaning we could able to reduce the sum of co mments Length from 30047792 to 17926693.
After training and testing multiple algorithms, we came to know that Logistic Regression model is performing better with an acc uracy score of 92% to 94%. Then after performing hyper-param eter tuning the model's final accuracy score was 92.03%. And fi nally, the Roc-Auc score is 0.97.

Conclusion

- Key Findings and Conclusions of the Study

  By looking at the cloud words generated for different classes we can conclude that some comments are offensive. Hence it should be controlled and restricted so that we can avoid spreading hatred and cyberbullying.
  Among selected algorithms/models Logistic Regression model is performing better with an accuracy score of 92% to 94%. And finally, the Roc-Auc score is 0.97.

- Learning Outcomes of the Study in respect of Data Science

  With the help of this project i.e., Comments Classifier Project we came to know about the strengths/effectiveness of Data Science packages/libraries. With the help of Pandas library, we could able to read our data and manipulate it for further use. With the help of Matplotlib and Seaborn we could able to visualize things. With the help of WordCloud we could able to visualize words frequency or importance in the text data based on the size of each word. With the help of Scikit-Learn library we could able to import algorithms/models and then build algorithms/models which can be used to predict comments classes.
  In this project the model which worked best is Logistic regression. Logistic regression is a supervised learning algorithm which is mostly used to solve binary "**classification"** tasks although it contains the word "**regression**". "Regression" contradicts with "classification" but the focus of logistic regression is on the word "logistic" referring to **logistic function** which actually does the classification task in the

algorithm. Logistic regression is a simple yet very effective classification algorithm so it is commonly used for many binary classification tasks. Customer churn, spam email, website or ad click predictions are some examples of the areas where logistic regression offers a powerful solution. It is even used as an activation function for neural network layers.

The basis of logistic regression is the logistic function, also called the **sigmoid function**, which takes in any real valued number and maps it to a value between 0 and 1. Logistic regression model takes a linear equation as input and use logistic function and log odds to perform a binary classification task.

- Limitations of this work and Scope for Future Work
    - Limitations of this work

      The dataset we got here is an imbalanced one.

    - Scope for Future Work

      Collect some more comments and consider few more para meters while hyper-parameter tuning the Logistic Regressi on model.