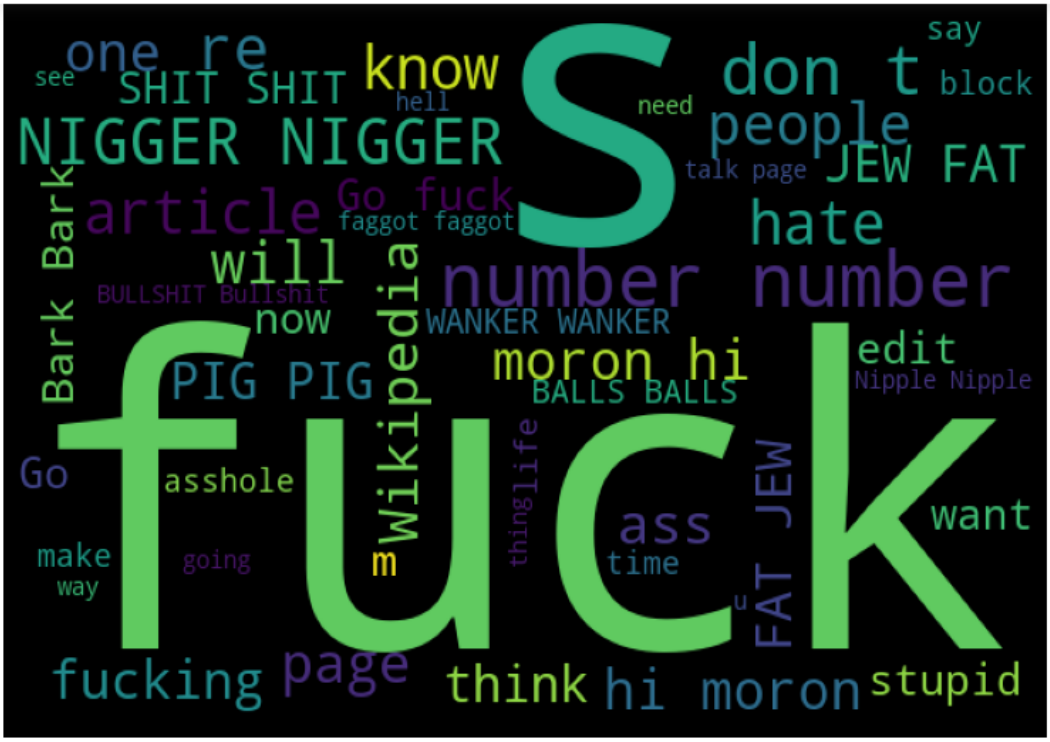
FLIP ROBO

# Malignant Comments Classification Project




Submitted by:

Deepam Purkayastha

# ACKNOWLEDGMENT

All the required information & the dataset are provided by Flip Robo. Also, I have used a few below external resources that helped me to complete the project.

External Resources:

1) Google & Youtube
2) https://scikit-learn.org
3) kaggle & github

# INTRODUCTION

- ## Business Problem Framing

  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. 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 cyber bullying.

- ## Conceptual Background of the Domain 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. This can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts.

- ## Review of Literature

  Based on the sample data provided where we have understood that 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. The data set explains it is a classification problem as we need to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not. Also, we have other independent features that would help to decide which all variables are important to predict the price of the variable and how do these variables describe the price of the house.

### Motivation for the Problem Undertaken

Based on the problem statement & the data provided, I have understood that the objective behind to make this project 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 cyber bullying. Based on the analysis on the comments in the dataset, I would be able to classify the negative comments, which would be help to stop spreading hatred.

# Analytical Problem Framing

- ## Mathematical/ Analytical Modeling of the Problem

  The statistical analysis by using data. Describe().

| | malignant | highly_malignant | rude | threat | abuse | loathe |
|---|---|---|---|---|---|---|
| count | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 | 159571.000000 |
| mean | 0.095844 | 0.009996 | 0.052948 | 0.002996 | 0.049364 | 0.008805 |
| std | 0.294379 | 0.099477 | 0.223931 | 0.054650 | 0.216627 | 0.093420 |
| min | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 50% | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 75% | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| max | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 |

Here, I got to know statistics of the data set where it explains the mean, standard deviation, minimum & maximum value, and how the % of the data is distributed. Then by the help of .info(), I get to know the data type & if there are any missing values. By using correlation function & heat map, I have understood if there is any multi-collinearity issue or which feature is negative/positive correlated with the target variable. I have used few visualization techniques to understand more about the data and which helped me to decide the importance of independent features.

- Data Sources and their formats

The data set contains the training set, which has approximately 1,59,000 samples and the test set which contains nearly 1,53,000 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.

The data set includes:

- **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.

- Data Pre-processing Done

1) Checking for null values: No Null values found & removed the ID column as it is having unique ID.
2) Checked for the correlation to visualize the feature importance & accordingly dropped a few features if required, which are highly correlated to each other.
3) Converted the comments in train data into lowercase.
4) Have removed punctuations, stop words to get a clean length.

5) Checked the length of top 10 comments for test & train data.
6) Replaced all the comments text with required data format for both train & test data set.
7) Checked for all the offensive loud words using word cloud.
8) Checked the distribution of label over comments by using pie plot.
9) Have converted text into vectors using TF-IDF for model building.
10) Original Length of training set before cleaning: 62893130 & Original Length of Test set before cleaning: 55885733.
11) Created train test split: We have split the train & test data in 0.30 test size with random state 56.

- ## Data Inputs- Logic- Output Relationships

I have all the data format as Integer & float in the dataset. To visualize the inputs-output relationship, I have used pie plot & count plot, which helped me to understand the distribution of the data. Also, used World cloud to check the loud words, which are offensive.

- ## Hardware and Software Requirements and Tools Used

**Hardware**: Laptop (OS: Windows, RAM: 8GB)

**Software**: Anaconda jupyter notebook

**Libraries**:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import sklearn
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve, roc_auc_score, classification_report, plot_roc_curve
import statsmodels.api as sm
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier,GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
import joblib
import warnings
warnings.filterwarnings('ignore')
```

```
import nltk
import string
from nltk.corpus import stopwords
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

**1)** I have used numpy pandas to load the dataset, perform the data cleaning part & perform EDA.
**2)** To visualize the dataset & to check correlation/multi-collinearity, I have used seaborn & matplotlib.

3) I have used NLTK & Stop words to remove punctuations & stop words.
4) To convert the text into vectors, I have used TF-IDF.
5) Sklearn has been used for pre- processing the data and for model building.
6) Joblib has used to save the model.

## Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  I have used both statistical and analytical approaches to solve the problem which mainly includes the pre-processing of the data and EDA to check the correlation of independent & dependent features. Also, before building the model, I make sure that the data is cleaned & scaled.

- Testing of Identified Approaches (Algorithms)
  1) Logistic Regression
  2) DecisionTreeClassifier
  3) RandomForestClassifier
  4) AdaBoostClassifier
  5) KNeighborsClassifier

- Run and Evaluate selected models
  1) Logistic Regression

```
LG = LogisticRegression(C=1, max_iter = 3000)
LG.fit(x_train, y_train)
y_pred_train = LG.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = LG.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9596773471561966
Test accuracy is 0.9554227941176471
[[42730   220]
 [ 1914  3008]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.93      0.61      0.74      4922

    accuracy                           0.96     47872
   macro avg       0.94      0.80      0.86     47872
weighted avg       0.95      0.96      0.95     47872
```

  2) DecisionTreeClassifier

```
DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
y_pred_train = DT.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = DT.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9990241631527588
Test accuracy is 0.9420538101604278
[[41684  1266]
 [ 1508  3414]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97     42950
           1       0.73      0.69      0.71      4922

    accuracy                           0.94     47872
   macro avg       0.85      0.83      0.84     47872
weighted avg       0.94      0.94      0.94     47872
```

## 3) RandomForestClassifier

```
RF = RandomForestClassifier()
RF.fit(x_train, y_train)
y_pred_train = RF.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = RF.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9990152105211327
Test accuracy is 0.9559450200534759
[[42414   536]
 [ 1573  3349]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.86      0.68      0.76      4922

    accuracy                           0.96     47872
   macro avg       0.91      0.83      0.87     47872
weighted avg       0.95      0.96      0.95     47872
```

## 4) AdaBoostClassifier

```
ada=AdaBoostClassifier(n_estimators=100)
ada.fit(x_train, y_train)
y_pred_train = ada.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = ada.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9509664365840339
Test accuracy is 0.9490307486631016
[[42541   409]
 [ 2031  2891]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     42950
           1       0.88      0.59      0.70      4922

    accuracy                           0.95     47872
   macro avg       0.92      0.79      0.84     47872
weighted avg       0.95      0.95      0.94     47872
```

5) KNeighborsClassifier

```
knn=KNeighborsClassifier(n_neighbors=9)
knn.fit(x_train, y_train)
y_pred_train = knn.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = knn.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9213153206385016
Test accuracy is 0.9163602941176471
[[42809   141]
 [ 3863  1059]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42950
           1       0.88      0.22      0.35      4922

    accuracy                           0.92     47872
   macro avg       0.90      0.61      0.65     47872
weighted avg       0.91      0.92      0.89     47872
```

**Cross Validation**

```
from sklearn.model_selection import cross_val_score

scr = cross_val_score(LG, x_train, y_train, cv=5)
print("Cross validation score for Logistic Regression model:" , scr.mean())
```

```
Cross validation score for Logistic Regression model: 0.9542072964316475
```

```
scr = cross_val_score(DT, x_train, y_train, cv=5)
print("Cross validation score for Decision Tree model:" , scr.mean())
```

```
Cross validation score for Decision Tree model: 0.9397577432917681
```

```
scr = cross_val_score(ada, x_train, y_train, cv=5)
print("Cross validation score for ada Boost model:" , scr.mean())
```

```
Cross validation score for ada Boost model: 0.949525067524857
```

```
scr = cross_val_score(knn, x_train, y_train, cv=5)
print("Cross validation score for Knn model:" , scr.mean())
```

```
Cross validation score for Knn model: 0.9163645202918822
```

```
scr = cross_val_score(RF, x_train, y_train, cv=5)
print("Cross validation score for Random forest model:" , scr.mean())
```

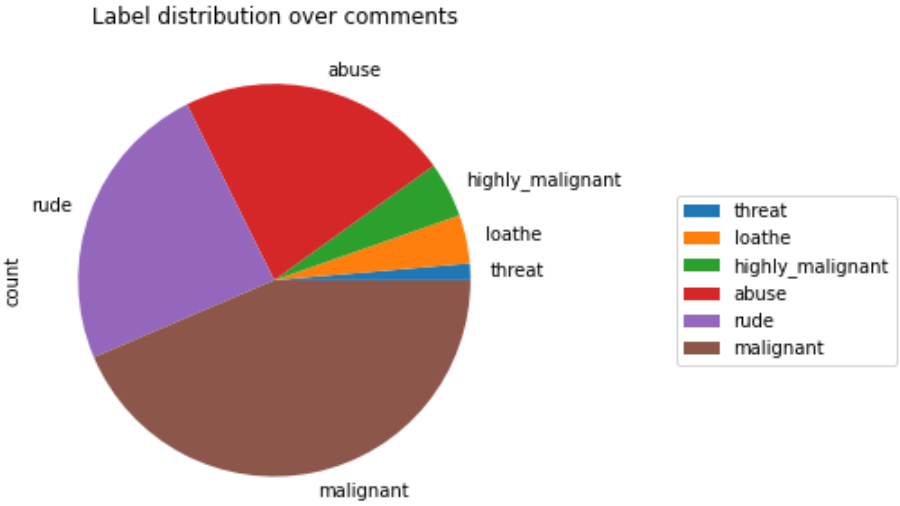
```
Cross validation score for Random forest model: 0.9564275484344096
```

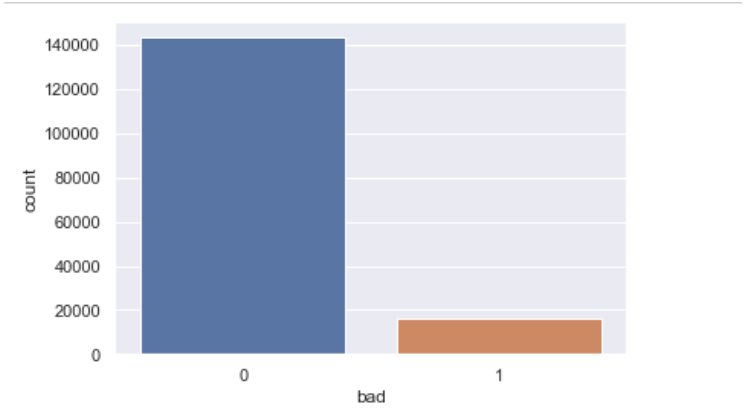
- Key Metrics for success in solving problem under consideration

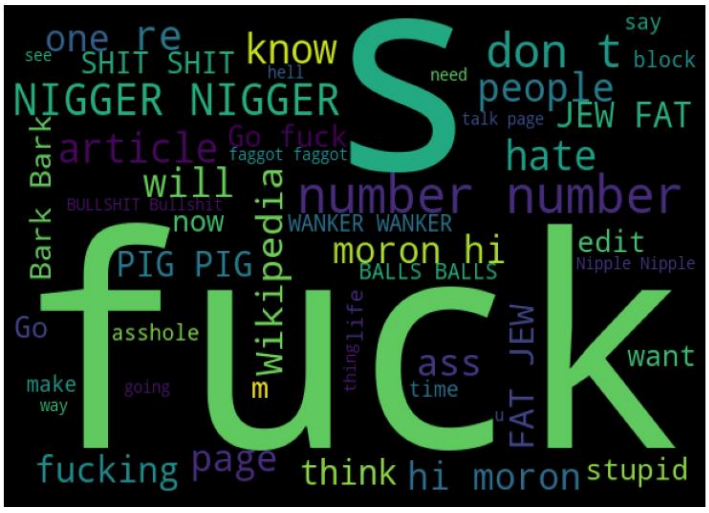
  I have used above shown screenshot of key metrics for model building & it helped me to understand why out of 5 models, I choose the best model based on the metrics such as accuracy score, confusion matrix, the classification report (Precision, recall & f1-score), Cross validation Score, & AUC-ROC.

- Visualizations:

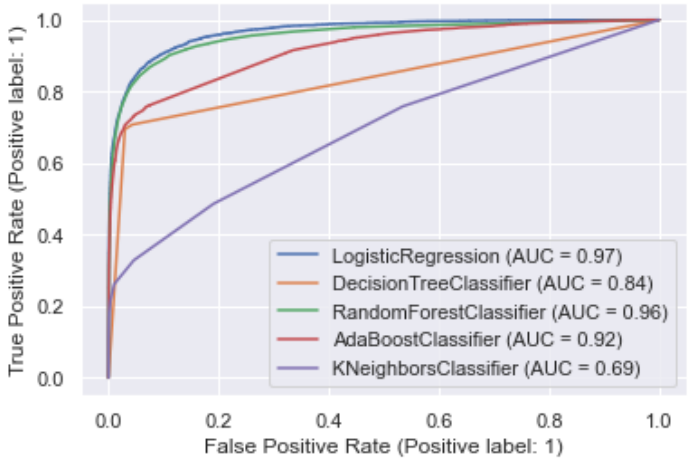
  Count plot, pie plot, & word cloud plot helped me to understand the distribution of comments, loud words & the count of bad words.



- ROC-AUC: Helped me to understand & select the best model out of 5 models based on roc-auc score.




- Interpretation of the Results

**Visualizations**: It helped me to understand the correlation between independent & dependent features. I got to know the count of particular category for each features by using count plot & most importantly AUC-ROC helped me to select the best model.

**Pre-processing:** Basically before building the model the dataset should be cleaned & scaled by performing NLP, which I mentioned above in the Pre-processing steps where all the important features are present in the data set and ready for model building.

**Model Creation:** Now, after performing the train test split, I have x_train, x_test, y_train & y_test, which are required to build Machine learning models. I have built multiple classification models to get the best accuracy score, confusion matrix, the classification report (Precision, recall & f1-score), Cross validation Score, and AUC-ROC.

# CONCLUSION

- Key Findings and Conclusions of the Study

  After performing the model building, I have got the highest score for RandomForestClassifier with 96% accuracy score having less Type-I &II error as compared to other models but it could be due to overfitting, so I have checked for the cross validation scores & found 95% score, which gives very less difference between accuracy score & CV score. Hence, based on accuracy score, confusion matrix, the classification report (Precision, recall & f1-score), Cross validation Score, and AUC-ROC, I have got the best fit model is RandomForestClassifier.

  Let's predict & compare the results:

```
test_df['Prediction'] = prediction
test_df.head()
```

| | comment_text | Prediction |
|---|---|---|
| 0 | Yo bitch Ja Rule is more succesful then you ll... | 0 |
| 1 | From RfC The title is fine as it is IMO | 0 |
| 2 | Sources Zawe Ashton on Lapland | 0 |
| 3 | If you have a look back at the source the info... | 0 |
| 4 | I don t anonymously edit articles at all | 0 |

## Concluding Remarks

1) Saving the model: The model is ready & we have saved the model in 'pkl' format by using "joblib".

## Final conclusion:

As we have seen, the prediction with the test data is showing almost similar relationship with the actual value from the train data set, which means the model predicted correctly & this could help to classify hate and offensive comments & to control and restrict from spreading hatred comments.

- Learning Outcomes of the Study in respect of Data Science

1) Visualization helped me to understand the data as it provides graphical representation of huge data, which helped me to understand the feature importance, with the help of pie plot, count pot & word cloud, I am able to see the distribution of threat comments.

2) Data cleaning is the most important part of model building as before model building, I make sure the data is cleaned & scaled.
3) I have performed multiple algorithms to get the best model, and found Random Forest classifier is the best fit model out of all the algorithms based on the metrics I have performed.

4) The challenges I faced while working on this project is to find punctuations & stop words, which took time to run using NLP. Also, as the data set is huge it took time to run few algorithms & to check the cross validation score.