FLIP ROBO

Malignant Comments Classifier Project

Submitted by:

Jaideep Pitale

ACKNOWLEDGMENT

Following are the websites which we used during this project for reference and study purpose:

1)https://www.kaggle.com/surekharamireddy/malignant-comment-classification-starter-notebook

2)https://towardsdatascience.com/toxic-comment-classification-using-lstm-and-lstm-cnn-db945d6b7986

3)https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a ---- for log loss information.

4)https://towardsdatascience.com/hyperparameter-tuning-for-machine-learning-models-1b80d783b946 --- for hyper parameter tuning

INTRODUCTION

- ## Business Problem Framing

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.

- ## Conceptual Background of the Domain Problem

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.

- Review of Literature

Following are the websites which we used during this project for reference and study purpose:

1)https://www.kaggle.com/surekharamireddy/malignant-comment-classification-starter-notebook

2)https://towardsdatascience.com/toxic-comment-classification-using-lstm-and-lstm-cnn-db945d6b7986

3)https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a ---- for log loss information.

4)https://towardsdatascience.com/hyperparameter-tuning-for-machine-learning-models-1b80d783b946 --- for hyper parameter tuning

- Motivation for the Problem Undertaken

This project is done as part of internship program between Data Trained and Flip Robo Technologies.
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 Problem Framing

- Mathematical/ Analytical Modeling of the Problem

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.

- Data Sources and their formats

  Training and Testing dataset have been provided to us as part of this project.

  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’.

  Comments and ID column are of object data type whereas rest of the columns are of integer datatype.

- Data Preprocessing Done

  Since id column has unique id for each comment, we are dropping this column.

  Following is the preprocesing done on ‘Comments’ column:

  1. Convert all messages to lower case
  2. Replace email addresses with 'email'
  3. Replace URLs with 'web address'

4. Replace money symbols with 'money symbol' (£ can by typed with ALT key + 156)
5. Replace 10-digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
6. Replace numbers with 'numbr'
7. Convert text into vectors using TF-IDF

- ## Data Inputs- Logic- Output Relationships

  For training dataset, all the data samples contain 8 fields which includes ‘Id’, ‘Comments’, ‘Malignant’, ‘Highly malignant’, ‘Rude’, ‘Threat’, ‘Abuse’ and ‘Loathe’.

  In testing dataset, only ‘comments’ column is present which is of Object data type and output of this dataset is whether comment is malignant or not (0 or 1) of integer data type.

- ## State the set of assumptions (if any) related to the problem under consideration

  None

- ## Hardware and Software Requirements and Tools Used

  We are using Jupyter Notebook for coding purpose along with Python 3.7.9 version for our model building process.

  We are using below libraries for our model building process:

  import numpy as np

  import pandas as pd

  import sklearn

```
import seaborn as sns
import matplotlib.pyplot as plt
from nltk.stem import WordNetLemmatizer
import nltk
from nltk.corpus import  stopwords
import string
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, f1_score, roc_curve ,roc_auc_score, auc
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score,GridSearchCV
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier,
AdaBoostClassifier, GradientBoostingClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import AdaBoostClassifier,
GradientBoostingClassifier
import pickle
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  ‘Comments’ column is the one on which we have focused more as based on content of this column output was to be decided.

  There is lot of data pre-processing that is being done on this column. Refer section ‘Data Preprocessing Done‘ for all details.

  Then NLTK tool is used to process the comments. Stop word is used to remove unnecessary and more frequently used words so as to reduce the population of words for better model building.

  WordNetLemmatizer is used to break entire comment into seperate words.

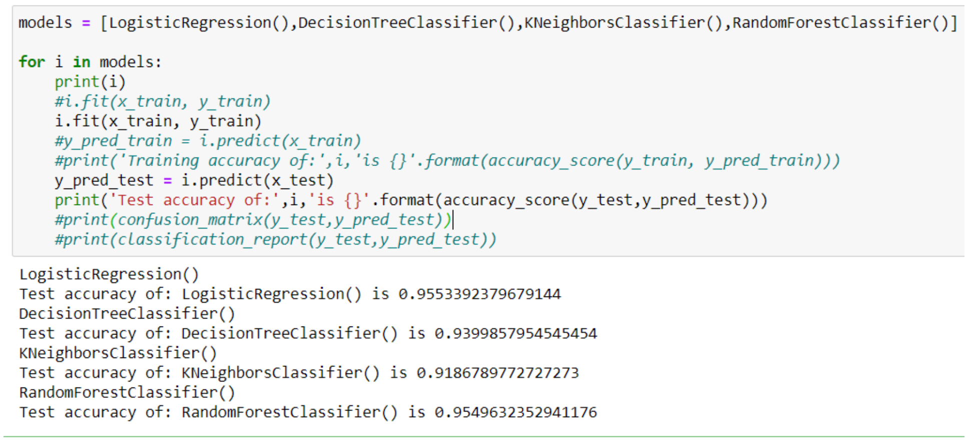
  TfidfVectorizer is used to Convert text into vectors using TF-IDF.

- Testing of Identified Approaches (Algorithms)

  Following is the list of algorithms on which this program is tested:

  a. LogisticRegression
  b. DecisionTreeClassifier
  c. KNeighborsClassifier
  d. RandomForestClassifier
  e. GradientBoostingClassifier

- Run and Evaluate selected models



```
models = [LogisticRegression(),DecisionTreeClassifier(),KNeighborsClassifier(),RandomForestClassifier()]

for i in models:
    print(i)
    #i.fit(x_train, y_train)
    i.fit(x_train, y_train)
    #y_pred_train = i.predict(x_train)
    #print('Training accuracy of:',i,'is {}'.format(accuracy_score(y_train, y_pred_train)))
    y_pred_test = i.predict(x_test)
    print('Test accuracy of:',i,'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))
```

```
LogisticRegression()
Test accuracy of: LogisticRegression() is 0.9553392379679144
DecisionTreeClassifier()
Test accuracy of: DecisionTreeClassifier() is 0.9399857954545454
KNeighborsClassifier()
Test accuracy of: KNeighborsClassifier() is 0.9186789772727273
RandomForestClassifier()
Test accuracy of: RandomForestClassifier() is 0.9549632352941176
```

- Key Metrics for success in solving problem under consideration

  Accuracy score and AUC_ROC curve is used measure the performance of the model.

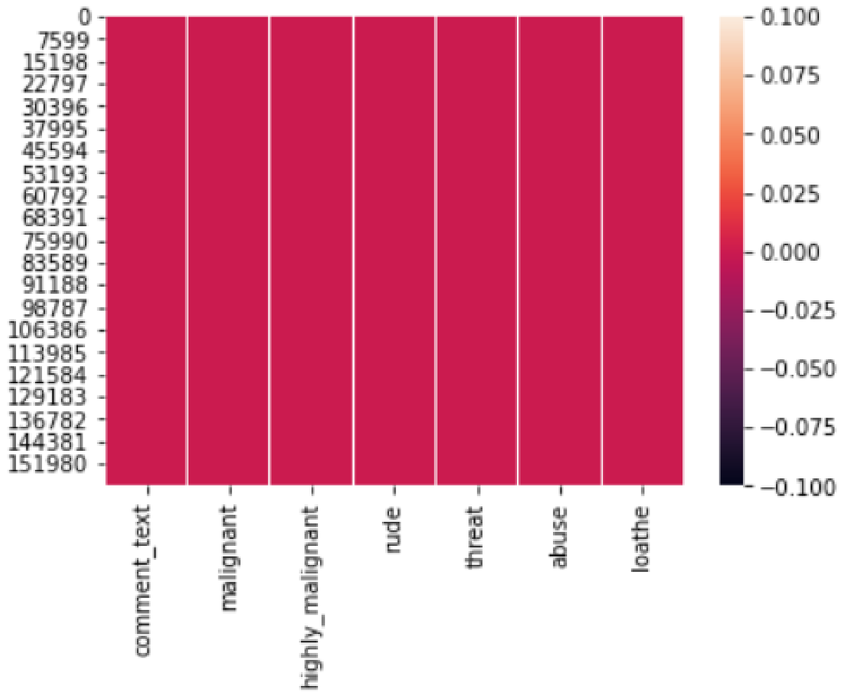
  GradientBoostingClassifier is also used to enhance the performance of the model.

- Visualizations

  Heatmap is used to check for any null values in dataset.

```
: sns.heatmap(raw_train_data.isnull()) # heatmap also ind
```

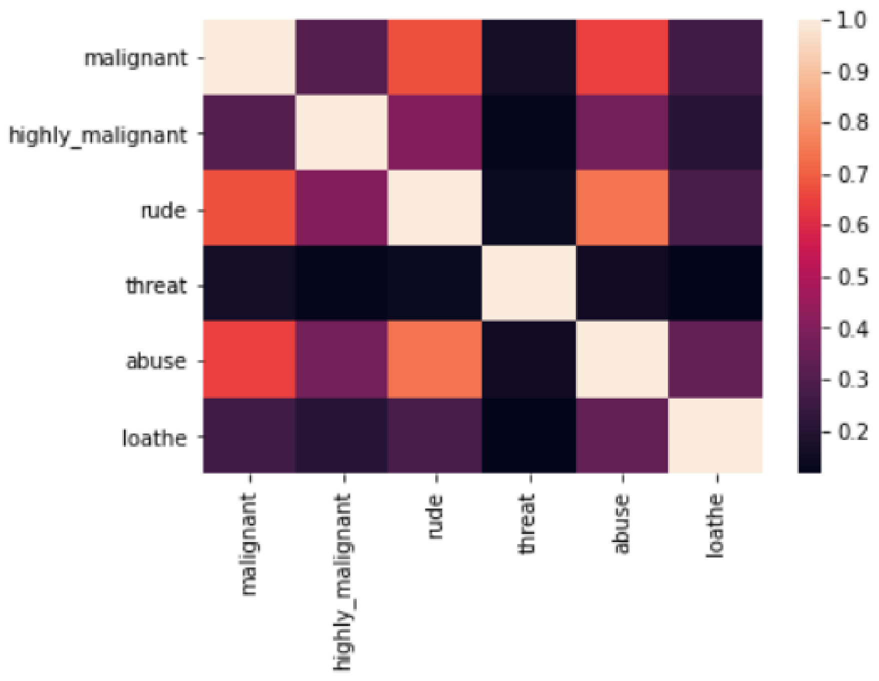
```
: <AxesSubplot:>
```



Heatmap is also used to check the correlation between columns in dataset.

```
sns.heatmap(raw_train_data.corr())
```

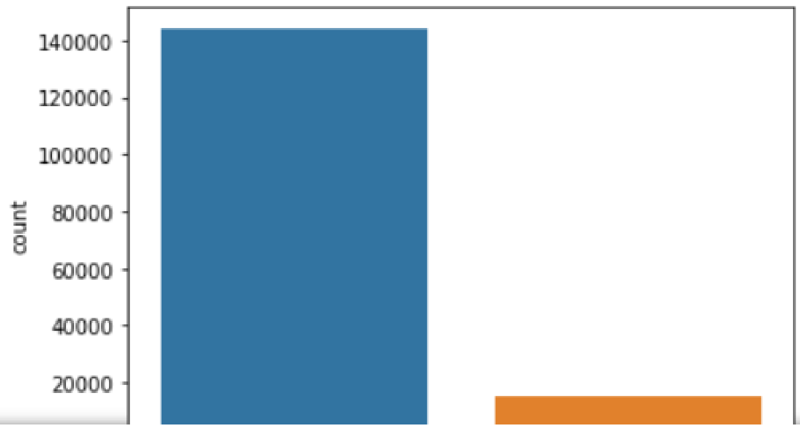
```
<AxesSubplot:>
```



Count plot is used to check the count of comment belonging to below groups: 'malignant', 'highly_malignant', 'loathe', 'rude', 'abuse', 'threat'

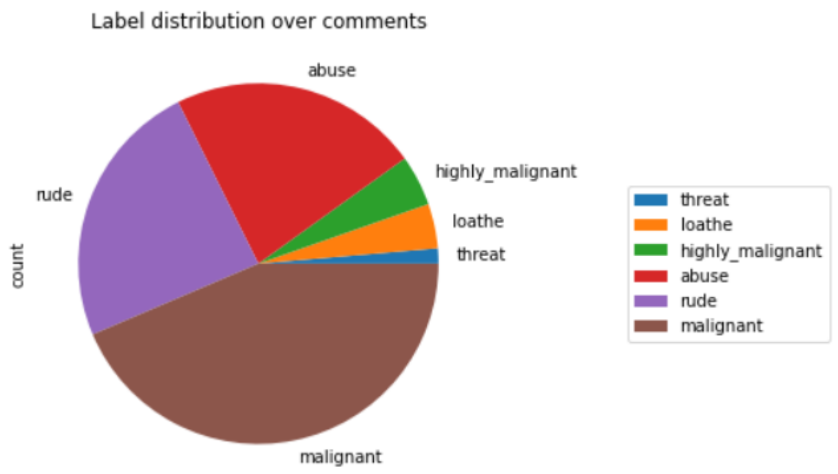
```
malignant


0    144277
1     15294
Name: malignant, dtype: int64
```

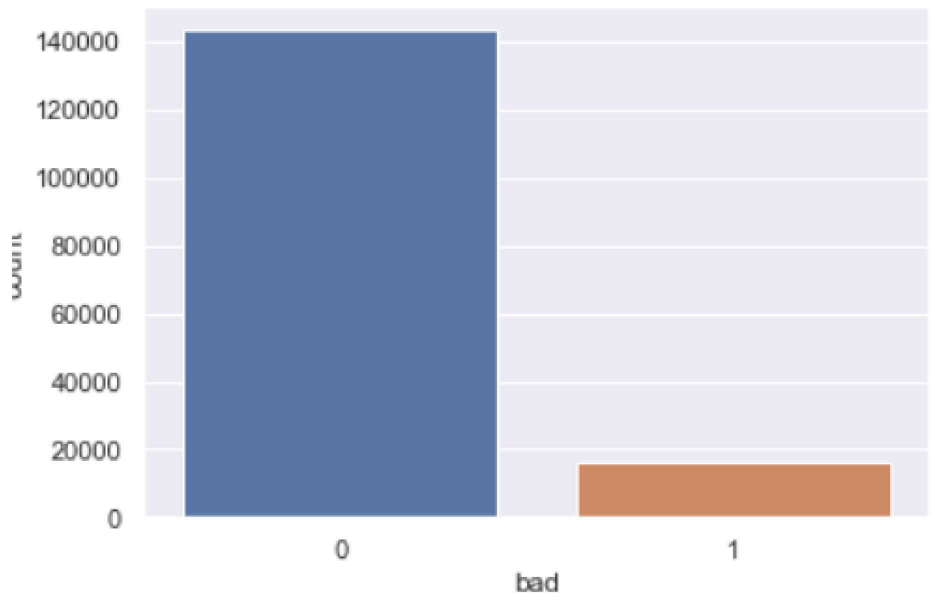


Pie chart is used to check the proportion of each type of comment as below:

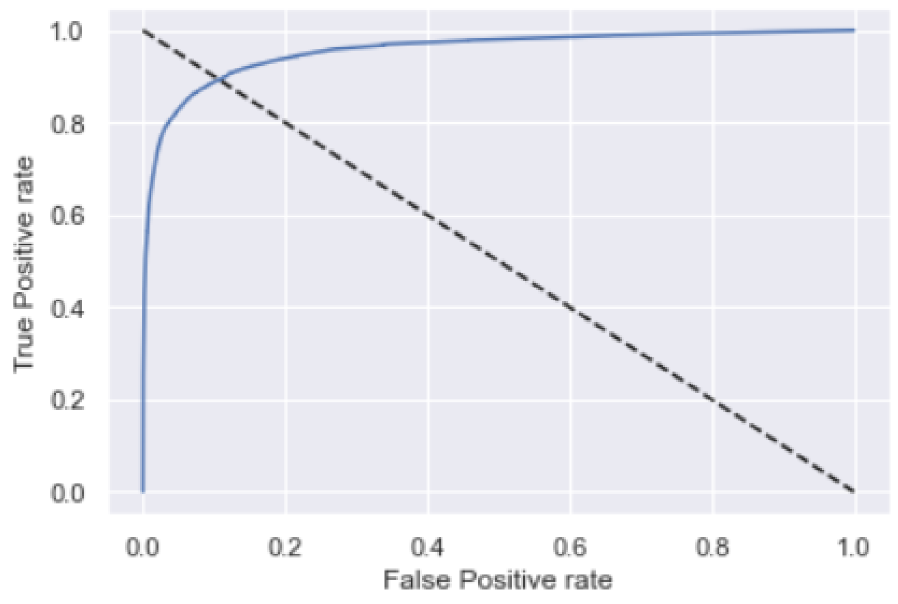
```
: <matplotlib.legend.Legend at 0x17b83438850>
```



Count plot is also used to check the count.



Finally, AUC_ROC curve used to check the performance of the model.



- Interpretation of the Results

Heatmap is used to check for any null values in dataset and also to check the correlation between columns in dataset.

Count plot is used to check the count of comment belonging to below groups: 'malignant', 'highly_malignant', 'loathe', 'rude', 'abuse', 'threat'

Pie chart is used to check the proportion of each type of comment.

AUC_ROC curve used to check the performance of the model.

Accuracy score and AUC_ROC curve score indicates that RandomForestClassifier model gives the best performance of model at 95.49%.

CONCLUSION

- Key Findings and Conclusions of the Study

  From given test dataset, 89.97 % of comments are non-malignant.

- Learning Outcomes of the Study in respect of Data Science

1. Heatmaps are very useful for data visualization as it depicts the data in a very simple and user-friendly manner.
2. RandomForestClassifier algorithm gives better performance than DecisionTree algorithm.
3. RandomForestClassifier model gives the best performance of model at 95.49%.
4. AUC ROC curve is very useful for checking the performance of model.

- Limitations of this work and Scope for Future Work

  In future scope we can work on proper cleaning of data. We can optimize the solution/ performance by making sure that data with outliers and mostly positive data is provided to algorithms. More data visualization methods can be used to describe and explain the data