Malignant Comments Classification Project

Submitted by:

DOLYPONA DAS

ACKNOWLEDGMENT

I would like to thank our SME (Shubham Yadav) for his expert advice and encouragement throughout this project.

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

1. Firstly we will check whether the problem is supervised or not, mostly we will have a supervised one where there will be target variable.
2. After that we will check whether the project is a regression type or a classification type.
3. We will also check whether our dataset is balanced or imbalanced. If it is an imbalanced one, we will apply sampling techniques to balance the dataset.
4. Then we will do model building and check its accuracy.
5. Our main motto is to build a model with good accuracy and for that we will also go for hyperparameter tuning.

• Review of Literature

I am summarizing my research done on the topic.

- I have imported important libraries for my project.
- I have created the dataframe for both train and test dataset.
- I have analysed my data by checking the shape of the train and test dataset, information of the train and test dataset, presence of null values in train dataset if any and also the statistical summary of train and test dataset.

▪ Then I have done some data cleaning steps, e.g checking correlation between the dependant and independent variables using heatmap, visualizing data using count plots, removing stopwords, converting text into vectors using TF-IDF and splitting the data into independent and dependant variables.

▪ I have used Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Ada Boost Classifier and K-Neighbours Classifier for model building and Random Forest Classifier model is showing the best accuracy where Training accuracy is 0.9988898736783678, Test accuracy is 0.954879679144385 and cross validation score is 95.66211845827077.

- Motivation for the Problem Undertaken

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.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

If you look at data science, we are actually using mathematical models to model (and hopefully through the model to explain some of the things that we have seen) business circumstances, environment etc and through these model, we can get more insights such as the outcomes of our decision undertaken, what should we do next or how shall we do it to improve the odds. So mathematical models are important, selecting the right one to answer the business question can bring tremendous value to the organization.

Here I am using 5 algorithms for model building ,out of which Random Forest Classifier is giving us the best accuracy.

- Data Sources and their formats

Data Source: The read_csv function of the pandas library is used to read the content of a CSV file into the python environment as a pandas DataFrame. The function can read the files from the OS by using proper path to the file.

Data description: Pandas describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. When this method is applied to a series of string, it returns a different output which is shown below.

```
In [29]: #Statistical summary of the train dataset
         train.describe()
```

Out[29]:

	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

```
In [30]: #Statistical summary of the test dataset
         test.describe()
```

Out[30]:

	id	comment_text
count	153164	153164
unique	153164	153164
top	950b8e090f2cd295	== Thought you quit == \n\n I thought you quit...
freq	1	1

- Data Preprocessing Done
  - I have checked the correlation between the target variables.
  - I have studied the statistical summary of the train and test dataset.
  - I have done some visualization using count plots.
  - Then I have converted the comments in train data into lowercase.
  - Then I have removed punctuations , stopwords to get a clean length.
  - Then I have converted text into vectors using TF-IDF so that the data gets ready for model building.
  - I have splitted the independent and dependant variables into x and y.
- Hardware and Software Requirements and Tools Used

Hardware requirements:
Processor: Intel(R) Celeron(R) CPU N3050 @1.60 GHz 1.60

GHz
RAM: 3.92 GB
System type: 64-bit operating system, x64-based processor
Software requirements:
Python: One of the most used programming languages
Tools used:
Jupyter notebook: Jupyter is a free, open-source,interactive web tool known as a computational notebook where I have written my python codes.
NumPy: NumPy is an open-source numerical Python library. NumPy contains a multi-dimensional array and matrix data structures.
Pandas: Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.
Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python.
Seaborn: It is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy.
Scikit-learn: It is probably the most useful library for machine learning in Python.The sklearn library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction.
Scipy.stats: This module contains a large number of probability distributions as well as a growing library of statistical functions.
Natural Language Toolkit: NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
  1. To check the correlation among the data, I have used **heatmap** to visualize it.
  2. To get a clear view of the columns visually, I have used **countplots**.
  3. For training and testing the data, I have imported **train_test_split library** from scikit-learn.
  4. For model building, I have used 5 algorithms, out of which **Random Forest Classifier** gives the best accuracy.

- Testing of Identified Approaches (Algorithms)

Logistic Regression- Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

Decision Tree Classifier- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

Random Forest Classifier- The term "Random Forest Classifier" refers to the classification algorithm made up of several decision trees. The algorithm uses randomness to build each individual tree to promote uncorrelated forests, which then uses the forest's predictive powers to make accurate decisions.

Ada Boost Classifier- AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances.

KNeighbors Classifier-It is a supervised machine learning algorithm. The algorithm can be used to solve both classification and regression problem statements. The number of nearest neighbours to a new unknown variable that has to be predicted or classified is denoted by the symbol 'K'.

- Run and Evaluate selected models
  - Logistic Regression

In [54]:
```
#LogisticRegression
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 0.9595520103134316
Test accuracy is 0.9553183489304813
[[42729   221]
 [ 1918  3004]]
              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
```

  - Decision Tree Classifier

In [55]:
```
#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 0.9988898736783678
Test accuracy is 0.938983121657754
[[41574  1376]
 [ 1545  3377]]
              precision    recall  f1-score   support

           0       0.96      0.97      0.97     42950
           1       0.71      0.69      0.70      4922

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

  - Random Forest Classifier

In [56]:
```
#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 0.9988540631518635
Test accuracy is 0.9550885695187166
[[42393   557]
 [ 1593  3329]]
              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
```

  - AdaBoost Classifier

In [58]:
```
#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 0.951118631321677
Test accuracy is 0.9490307486631016
[[42553   397]
 [ 2043  2879]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     42950
           1       0.88      0.58      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
```

- KNeighbors Classifier

In [59]:
```
#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))
```

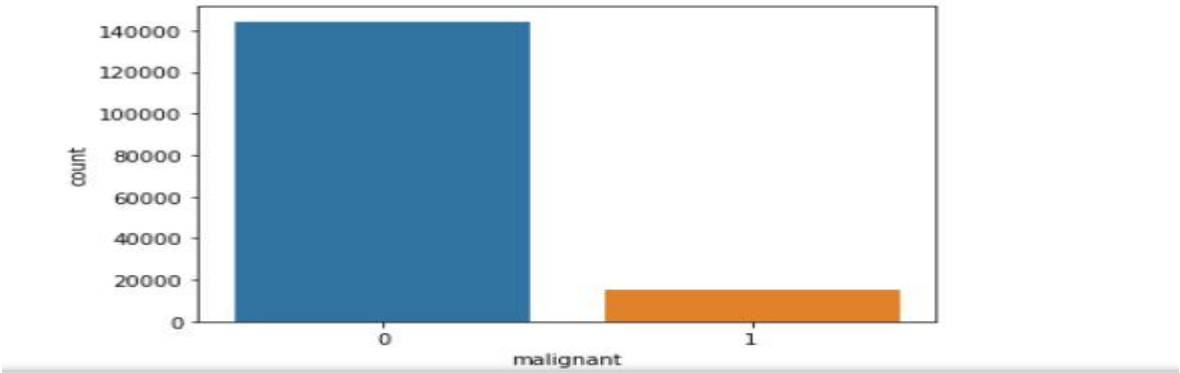
```
Training accuracy is 0.922300110117369
Test accuracy is 0.9173629679144385
[[42809   141]
 [ 3815  1107]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42950
           1       0.89      0.22      0.36      4922

    accuracy                           0.92     47872
   macro avg       0.90      0.61      0.66     47872
weighted avg       0.91      0.92      0.89     47872
```

- Key Metrics for success in solving problem under consideration

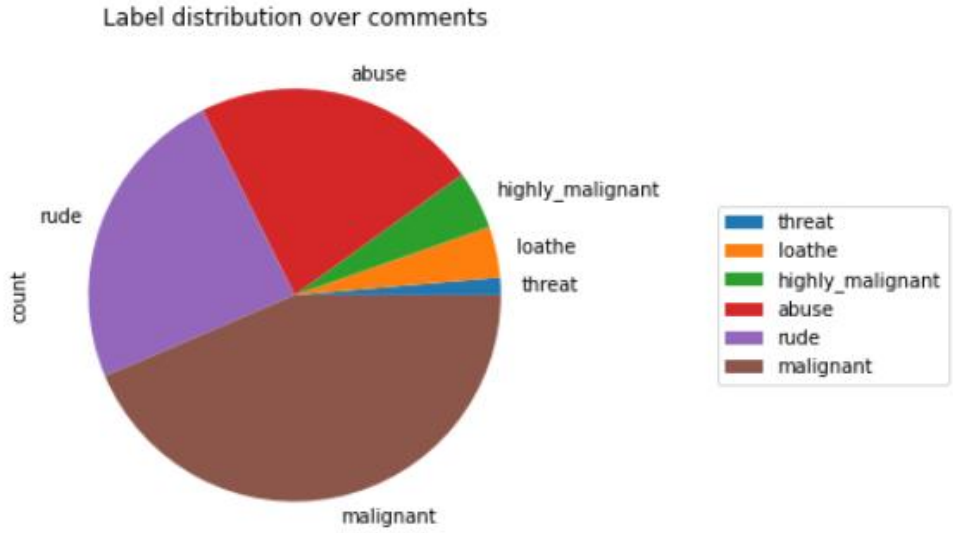
  The key metrics used are accuracy_score, confusion_matrix and classification_report .

- Visualizations
  - Countplot used to check the target variables

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

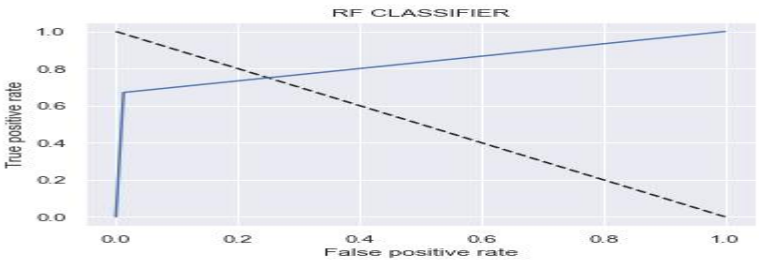


- Pie plot



- AUC- RUC curve

```
fpr,tpr,thresholds=roc_curve(y_test,y_pred_test)
roc_auc=auc(fpr,tpr)
plt.plot([0,1],[1,0],'k--')
plt.plot(fpr,tpr,label = 'RF Classifier')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('RF CLASSIFIER')
plt.show()
```



- ## Interpretation of the Results

In the visualization part, I have seen how my data looks like using heatmap,count- plots, pie- plot etc.

In the pre-processing part, I have cleaned my data using some NLP methods and other methods too.

In the modelling part, I have designed our model using algorithms like Logistic Regression, Decision Tree Classifier,Random Forest Classifier,Ada Boost Classifier and KNeighbors Classifier.
The accuracy ,f1 score, confusion_matrix, classification_report are achieved for each model.

CONCLUSION

- ## Key Findings and Conclusions of the Study

  The key findings are we have to study the data very clearly so that we are able to decide which data are relevant for our findings.
  The conclusion of our study is we have to achieve a model with good accuracy and f1-score.

- ## Learning Outcomes of the Study in respect of Data Science

  We will develop relevant programming abilities. We will demonstrate proficiency with statistical analysis of data. We will develop the ability to build and assess data-based models. We will execute statistical analyses with professional statistical software. The best algorithm for this project according to my work is Random Forest Classifier because the accuracy and f1 score that I have achieved is quite satisfactory than the other model.