



Malignant Comment Classification Project

Submitted by:

Ambika Saraf

ACKNOWLEDGMENT

I have taken efforts in this project however it would not have been completed without guidance from other people. I warmly acknowledge the invaluable supervision and an inspired guidance by our SME Mr. Keshav Bansal, FlipRobo Technology.

I would also like to express my sincere thanks to Data trained Education and FlipRobo Technology for giving me an opportunity to work on this project.

I also want to express my gratitude towards my friends and family who have patiently extended all sorts of help for accomplishing this.

I am grateful to one and all who are directly or indirectly involved in successful completion of this project.

Other reference websites used to complete this project are:

1. Data source provided by the client.
2. Wikipedia
3. Towardsdatascience.com
4. Datacamp.com

INTRODUCTION

In the past few years, the cases related to social media hatred have increased exponentially. Social media is turning into a venomous pit for people. In Social media people involved in such acts use filthy and aggressive language to offend and hurt the person on other side.

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

There has been increase in cases of cyber bullying and trolls on various social media platforms. This can affect anyone 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.

Our goal is to build a prototype of online hate and abuse comment classifier which can be used to classify hate and offensive comments so that it can be controlled and restricted from spreading hatred and cyber bullying. We have made use of Natural Language Processing techniques to build this classification model.

ANALYTICAL FRAMING

We have build a machine learning model here and before that we have done necessary pre-processing steps involved in NLP. We have build various models and selected the best one. Steps followed are:

1. Data Cleaning
2. Exploratory Data Analysis
3. Data Pre-Processing
4. Model Building
5. Model Evaluation

Data Set Description

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.

We have test and train dataset given. Test dataset consists of "**Comment text**" and "**id**". We will use train dataset to build the model and predict values in test dataset.

Train data Sample:

| | 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'aww! 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 | "\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 |

Test data Sample:

| | id | comment_text |
|---|------------------|---------------------------------------------------|
| 0 | 00001cee341fdb12 | Yo bitch Ja Rule is more succesful then you'll... |
| 1 | 0000247867823ef7 | == From RfC == \n\n The title is fine as it is... |
| 2 | 00013b17ad220c46 | " \n\n == Sources == \n\n * Zawe Ashton on Lap... |
| 3 | 00017563c3f7919a | :If you have a look back at the source, the in... |
| 4 | 00017695ad8997eb | I don't anonymously edit articles at all. |

Hardware and Software Requirements and Tools Used

- Laptop with stable internet connection (Project done in jupyter notebook)
- scikit-learn
- TfidfVectorizer
- nltk
- matplotlib
- pandas
- numpy

Exploratory Data Analysis

This project involves extensive EDA to gain conclusion from dataset provided. We have first checked the count of label in each target variable

Malignant: No: 144277
Yes: 15294

Highly-Malignant: No: 157976
Yes: 1595

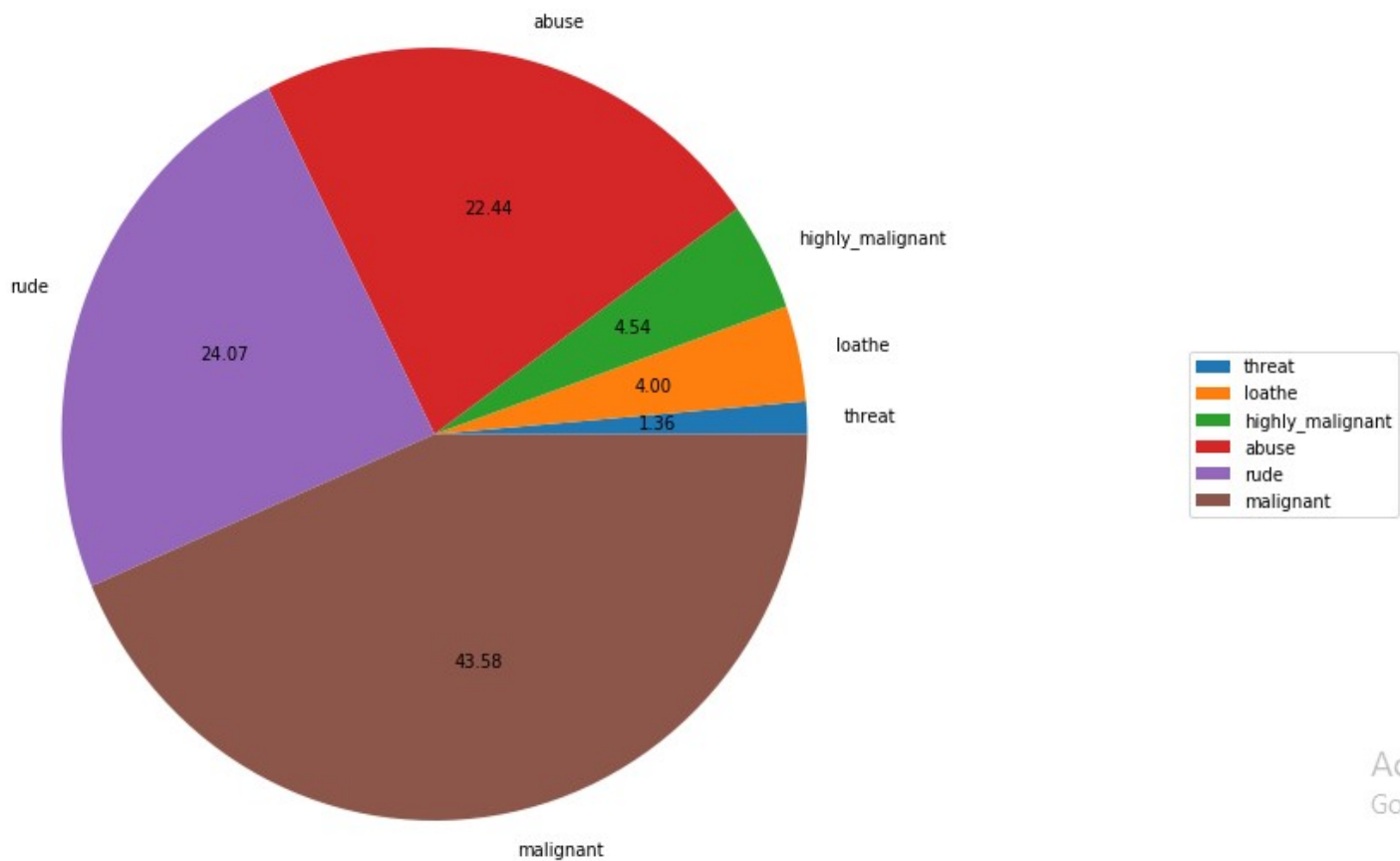
Threat: No: 159093
Yes: 478

Rude: No: 151122
Yes: 8449

Abuse: No: 151694
Yes: 7877

Loathe: No: 158166
Yes: 1405

Label distribution over comments:



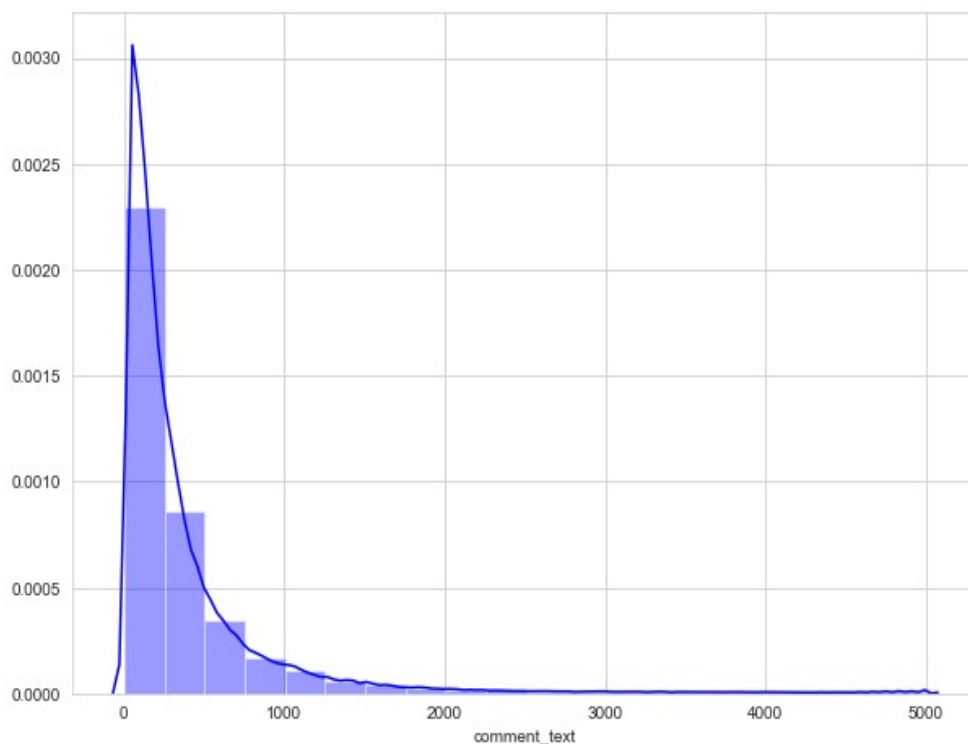
From the label counts and label distribution over comments we observe imbalanced data and among the negative comments least number of comments are classified as threat whereas, more number of comments are classified as malignant. We can further classify data as good/neutral and negative comments and below is the percent we obtain:

Percentage of good/neutral comments = 89.83211235124176

Percentage of negative comments = 10.167887648758239

Since there are 6 independent features whose value is to be predicted so we will introduce a new column in dataset as '**label**' which indicates '**1**' for good/neutral comments and '**0**' for negative comments.

Distribution of comments length



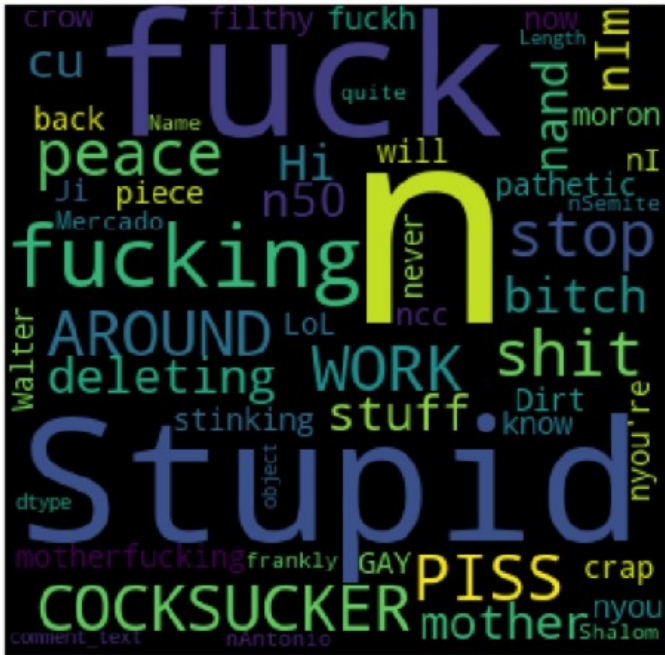
Next we have observed most frequent words in positive and negative comments through word-cloud

From the wordclouds, we can see that the large texts have more weightage in their respective type of comments whereas small texts have the lesser weightage.

WORDS TAGGED AS MALIGNANT



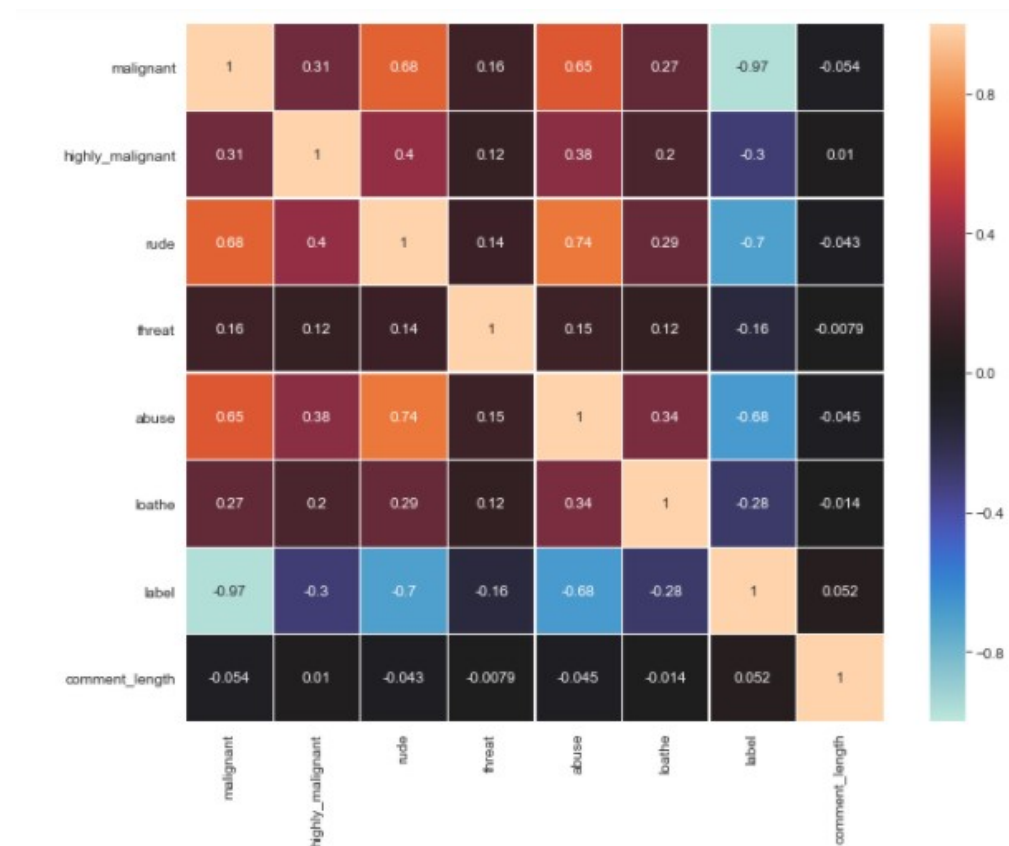
WORDS TAGGED AS HIGHLY-MALIGNANT







Finding relations with help of correlation



Above is the heatmap showing correlation between features. We make following conclusions:

- Highest correlation is observed between 'rude' and 'abuse'.
- Correlation observed here is both positive and negative

Data Pre-Processing

We have used data cleaning steps involved in NLP and modified data which is not in proper format.

- First step is to remove unnecessary features like 'id'.
- Next we have made a function to filter using POS tagging.
- Created another function named "Clean_data" which involves :
 1. Removing '\n' or extra lines.
 2. Removing numbers, HTML tags, websites, unwanted white spaces, etc.
 3. Transforming text into lower case.
 4. Removing remaining small and irrelevant tokens.

5. Lemmatizing words using WordsNetLemmatizer().

The code for given steps is mentioned below:

```
: #Creating a function to filter using POS tagging.

def get_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN

# Function for data cleaning...
def clean_data(comments):
    # Replace email addresses with 'email'
    comments=re.sub(r'^.+@[^\.\.]*\.[a-z]{2,}$',' ', comments)

    # Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
    comments=re.sub(r'^\((?[\d]{3}\)\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',' ',comments)

    # getting only words(i.e removing all the special characters)
    comments = re.sub(r'[^\\w]', ' ', comments)

    # getting only words(i.e removing all the" _ ")
    comments = re.sub(r'[_ ]', ' ', comments)

    # getting rid of unwanted characters(i.e remove all the single characters Left)
    comments=re.sub(r'\\s+[a-zA-Z]\\s+', ' ', comments)

    # Removing extra whitespaces
    comments=re.sub(r'\\s+', ' ', comments, flags=re.I)

    #converting all the Letters of the review into lowercase
    comments = comments.lower()

    # splitting every words from the sentences
    comments = comments.split()

    # remove empty tokens
    comments = [text for text in comments if len(text) > 0]

    # getting pos tag text
    pos_tags = pos_tag(comments)

    # considering words having length more than 3only
    comments = [text for text in comments if len(text) > 3]

    # performing Lemmatization operation and passing the word in get_pos function to get filtered using POS ...
    comments = [(WordNetLemmatizer().lemmatize(text[0], get_pos(text[1]))))for text in pos_tags]

    # considering words having length more than 3 only
    comments = [text for text in comments if len(text) > 3]
    comments = ' '.join(comments)
    return comments
```

After performing these steps we have added a new_comment_length variable.

Preparing Data for Model

We have first separated data and then converted it to number features using TfidfVectorizer. Further split our data into train and test part.

```
def Tf_idf_train(text):  
    tfidf = TfidfVectorizer(min_df=3,smooth_idf=False)  
    return tfidf.fit_transform(text)  
x=Tf_idf_train(train['clean_comment_text'])  
  
print("Shape of x: ",x.shape)  
  
y = train['label'].values  
print("Shape of y: ",y.shape)
```

```
Shape of x: (159571, 43141)  
Shape of y: (159571,)
```

```
#Splitting the training and testing data  
from sklearn.model_selection import train_test_split  
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.30, random_state=42, stratify=y)
```

MODEL BUILDING AND EVALUATION

Algorithms used are:

- Logistic Regression
- Decision Tree Classifier
- KNeighbors Classifier
- Random Forest Classifier
- Gradient Boosting Classifier
- AdaBoostClassifier

Logistic Regression

*** Logistic Regression ***

accuracy_score: 0.9529578877005348

cross_val_score: 0.9537823308256245

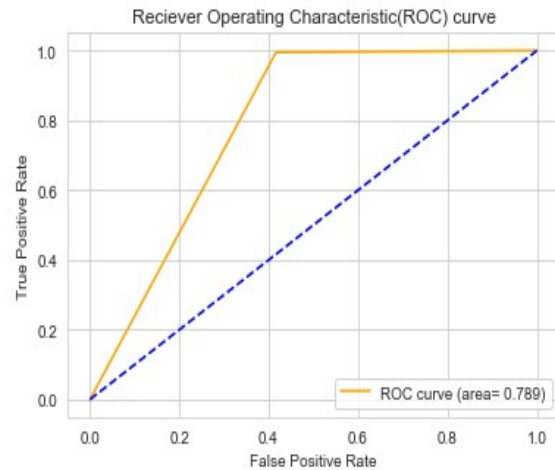
roc_auc_score: 0.7892786644906702

Classification report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.93 | 0.58 | 0.72 | 4868 |
| 1 | 0.95 | 0.99 | 0.97 | 43004 |
| accuracy | | | 0.95 | 47872 |
| macro avg | 0.94 | 0.79 | 0.85 | 47872 |
| weighted avg | 0.95 | 0.95 | 0.95 | 47872 |

Confusion matrix:

```
[[ 2842  2026]
 [  226 42778]]
```



DecisionTree

*** DecisionTreeClassifier ***

accuracy_score: 0.937917780748663

cross_val_score: 0.9380213133700785

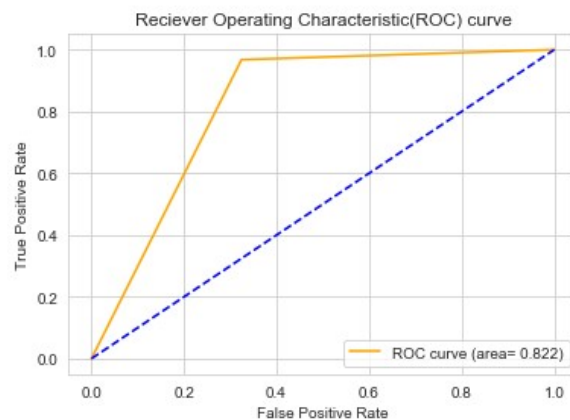
roc_auc_score: 0.8217133228782982

Classification report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.70 | 0.68 | 0.69 | 4868 |
| 1 | 0.96 | 0.97 | 0.97 | 43004 |
| accuracy | | | 0.94 | 47872 |
| macro avg | 0.83 | 0.82 | 0.83 | 47872 |
| weighted avg | 0.94 | 0.94 | 0.94 | 47872 |

Confusion matrix:

```
[[ 3290  1578]
 [ 1394 41610]]
```



KNeighbors

*** KNeighborsClassifier ***

accuracy_score: 0.8386112967914439

cross_val_score: 0.8471464524877395

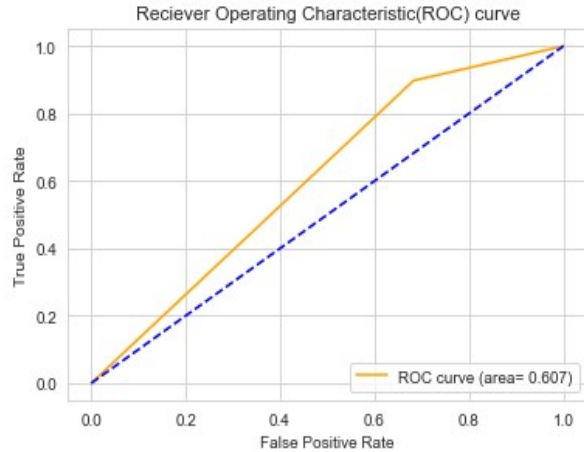
roc_auc_score: 0.6074964878771094

Classification report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.26 | 0.32 | 0.29 | 4868 |
| 1 | 0.92 | 0.90 | 0.91 | 43004 |
| accuracy | | | 0.84 | 47872 |
| macro avg | 0.59 | 0.61 | 0.60 | 47872 |
| weighted avg | 0.85 | 0.84 | 0.85 | 47872 |

Confusion matrix:

```
[[ 1545  3323]
 [ 4403 38601]]
```



Random Forest

*** RandomForestClassifier ***

accuracy_score: 0.9510987633689839

cross_val_score: 0.9515137516440152

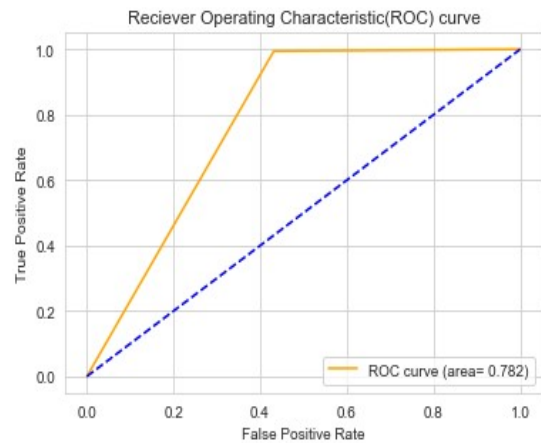
roc_auc_score: 0.7815036047553467

Classification report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.92 | 0.57 | 0.70 | 4868 |
| 1 | 0.95 | 0.99 | 0.97 | 43004 |
| accuracy | | | 0.95 | 47872 |
| macro avg | 0.94 | 0.78 | 0.84 | 47872 |
| weighted avg | 0.95 | 0.95 | 0.95 | 47872 |

Confusion matrix:

```
[[ 2768  2100]
 [  241 42763]]
```



AdaBoost

```
*** AdaBoostClassifier ***
```

```
accuracy_score: 0.9444351604278075
```

```
cross_val_score: 0.9457796177925338
```

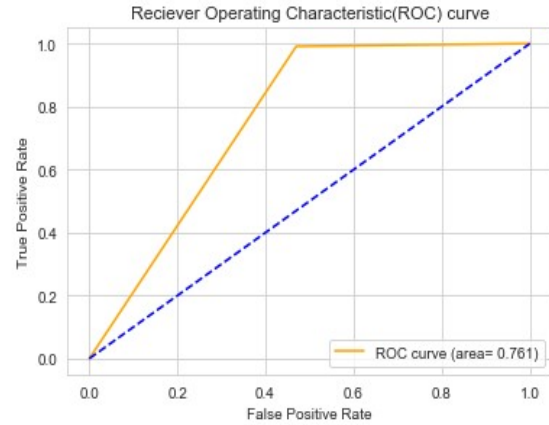
```
roc_auc_score: 0.7607617972439092
```

```
Classification report:
```

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.87 | 0.53 | 0.66 | 4868 |
| 1 | 0.95 | 0.99 | 0.97 | 43004 |
| accuracy | | | 0.94 | 47872 |
| macro avg | 0.91 | 0.76 | 0.81 | 47872 |
| weighted avg | 0.94 | 0.94 | 0.94 | 47872 |

```
Confusion matrix:
```

```
[[ 2581 2287]
 [ 373 42631]]
```



Gradient Boosting

```
*** GradientBoostingClassifier ***
```

```
accuracy_score: 0.9396933489304813
```

```
cross_val_score: 0.9407536437789279
```

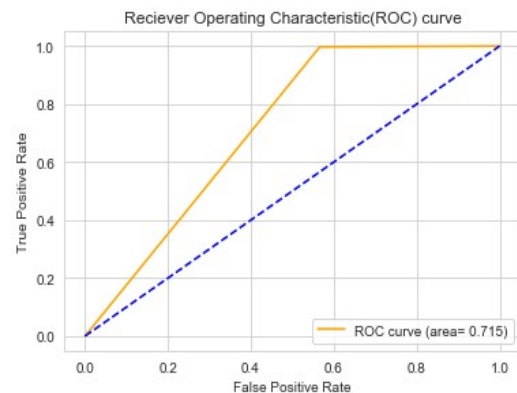
```
roc_auc_score: 0.7154037552219445
```

```
Classification report:
```

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.94 | 0.43 | 0.59 | 4868 |
| 1 | 0.94 | 1.00 | 0.97 | 43004 |
| accuracy | | | 0.94 | 47872 |
| macro avg | 0.94 | 0.72 | 0.78 | 47872 |
| weighted avg | 0.94 | 0.94 | 0.93 | 47872 |

```
Confusion matrix:
```

```
[[ 2112 2756]
 [ 131 42873]]
```



Choosing Best Model

After running the loop we get a dataframe showing each model and scores obtained from it.

| | Model | Accuracy_score | Cross_val_score | ROC_Score |
|---|----------------------------|----------------|-----------------|-----------|
| 0 | Logistic Regression | 95.295789 | 95.378233 | 78.927866 |
| 1 | DecisionTreeClassifier | 93.791778 | 93.802131 | 82.171332 |
| 2 | KNeighborsClassifier | 83.861130 | 84.714645 | 60.749649 |
| 3 | RandomForestClassifier | 95.109876 | 95.151375 | 78.150360 |
| 4 | AdaBoostClassifier | 94.443516 | 94.577962 | 76.076180 |
| 5 | GradientBoostingClassifier | 93.969335 | 94.075364 | 71.540376 |

Looking the various metrics we conclude “**Decision Tree Model**” as our best model.

We will now finalize it by saving our model using pickle.

To predict values in test dataset we will first apply “Clean_data” function to clean text and then use TfidfVectorizer to convert it into numeric form.

```
] def Tf_idf_test(text):  
    tfidf=TfidfVectorizer(max_features=43141,smooth_idf=False)  
    return tfidf.fit_transform(text)  
x_test=Tf_idf_test(test['clean_comment_text'])  
x_test.shape  
]  
(153164, 43141)
```

Next we will predict value using the model build.

```
In [59]: Predict=lr.predict(x_test)
test['Predicted Label']=Predict
test
```

Out[59]:

| | comment_text | comment_length | clean_comment_text | clean_comment_length | Predicted Label |
|----|----------------------------------------------------|----------------|----------------------------------------------------|----------------------|-----------------|
| 0 | Yo bitch Ja Rule is more succesful then you'll... | 387 | bitch rule more succesful then ever whats with... | 227 | 1 |
| 1 | == From RFC == \n\n The title is fine as it is... | 50 | from title fine | 15 | 1 |
| 2 | " \n\n == Sources == \n\n * Zawe Ashton on Lap... | 54 | source zawe ashton lapland | 28 | 1 |
| 3 | :If you have a look back at the source, the in... | 205 | have look back source information updated corr... | 140 | 1 |
| 4 | I don't anonymously edit articles at all. | 41 | anonymously edit article | 24 | 1 |
| 5 | Thank you for understanding. I think very high... | 98 | thank understanding think very highly would re... | 69 | 1 |
| 6 | Please do not add nonsense to Wikipedia. Such ... | 176 | please nonsense wikipedia such edits considere... | 123 | 1 |
| 7 | :Dear god this site is horrible. | 32 | dear this site horrible | 23 | 1 |
| 8 | " \n Only a fool can believe in such numbers. ... | 556 | only fool believe such number correct number b... | 356 | 1 |
| 9 | == Double Redirects == \n\n When fixing double... | 224 | double redirects when fixing double redirects ... | 144 | 1 |
| 10 | I think its crap that the link to roggienbier i... | 119 | think crap that link roggienbier this article s... | 83 | 1 |
| 11 | "::: Somebody will invariably try to add Relig... | 448 | somebody will invariably religion really mean ... | 293 | 1 |
| 12 | , 26 February 2010 (UTC) \n\n :::Looking it ov... | 293 | february 2010 looking over clear that banned s... | 198 | 1 |
| 13 | " \n\n It says it right there that it IS a typ... | 501 | right there that type type institution needed ... | 349 | 1 |
| 14 | " \n\n == Before adding a new product to the l... | 334 | before adding product list make sure relevant ... | 229 | 1 |
| 15 | ==Current Position== \n Anyone have confirmati... | 113 | current position anyone have confirmation that... | 81 | 1 |
| 16 | this other one from 1897 | 24 | this other from 1897 | 20 | 1 |
| 17 | == Reason for banning throwing == \n\n This ar... | 159 | reason banning throwing this article need sect... | 101 | 1 |
| 18 | :: Wallamoose was changing the cited material ... | 336 | wallamoose changing cited material thing origi... | 199 | 1 |
| 19 | [blocked]] from editing Wikipedia. I | 38 | blocked from editing wikipedia | 30 | 1 |
| 20 | ==Indefinitely blocked== \n I have indefinitel... | 68 | indefinitely blocked have indefinitely blocked... | 59 | 0 |
| 21 | == Arabs are committing genocide in Iraq, but ... | 104 | arab committing genocide iraq protest europe e... | 66 | 1 |

After predicting we will save this file in csv format.

CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

We have first loaded both test and train dataset and looked into it to check for null values, data types, dropping unnecessary columns, etc.

Exploratory Data Analysis is performed using visualizations to gain insights. We observe more of comments classified as malignant comments.

We have used NLP to pre-process and clean data. Since we had 6 labels to be predicted we introduced a new column 'label' which shows '1' for good/neutral comments and '0' for negative comments.

We have used Vectorizer to convert comment tokens to numeric form.

After considering the mentioned algorithms we concluded Decision Tree Classifier to be best model which gave an accuracy of 95.2%.

We further used the model selected to predict 'label' for test dataset and saved it in csv file.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

After the completion of this project, we got an insight of how to preprocess the data and analysing the data using Natural Language Processing techniques. We have used various NLP techniques to clean our data like Lemmatization, POS tagging, etc.

We were also able to learn to convert strings into vectors through vectorizer.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

Since the data keeps changing we cannot fully rely on this project in the distant future we need to update it with updation in data.

Due to lack of high computational power it took a lot of time to predict scores for each algorithm.

The Dataset here is imbalanced which may have caused a problem in scores obtained.

This project is done with limited resources and can be made more efficient in future.