MAGLINANT CLASSIFIER PROJECT

Submitted by:

Amrutha Dhondale

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 Ms.Swati Mahaseth, 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.

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.

  • 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 inoffensive, 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.

- Conceptual Background of the Domain Problem

  • In the past few years its seen that the cases related to social media hatred have increased exponentially. The social media is turning into a dark venomous pit for people now a days. Online hate is the result of difference in opinion, race, religion, occupation, nationality etc.

- In social media the people spreading or involved in such kind of activities uses filthy languages, aggression, images etc. to offend and gravely hurt the person on the other side. This is one of the major concerns now.
- The result of such activities can be dangerous. It gives mental trauma to the victims making their lives miserable. People who are not well aware of mental health online hate or cyber bullying become life threatening for them. Such cases are also at rise. It is also taking its toll on religions. Each and every day we can see an incident of fighting between people of different communities or religions due to offensive social media posts.
- Online hate, described as abusive language, aggression, cyberbullying, hatefulness, insults, personal attacks, provocation, racism, sexism, threats, or toxicity has been identified as a major threat on online social media platforms. These kinds of activities must be checked for a better future

Review of Literature

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.

Motivation for the Problem Undertaken

The project was the first provided to me by FlipRobo as a part of the internship programme. The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary objective. However, the motivation for taking this project was that it is relatively a new field of research. Here we have many options but less concrete solutions. The main motivation 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

  Here we are dealing with one main text columns which held some importance of the data and others shows the multiple types of behaviour inferred from the text. I prefer to select on focus more on the words which has great value of importance in the context. Countvector is the NLP terms I am going to apply on text columns. This converts the important words proper vectors with some weights.

- Data Sources and their formats

  The data was provided by FlipRobo in CSV format. After loading the training dataset into Jupyter Notebook using Pandas and it can be seen that there are eight columns named as:
  "Id, comment text, "malignant, highly malignant, rude, threat, abuse, loathe". There are 8 columns in the dataset provided:
  The description of each of the column is given below:
  *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.

```
# Information of the train dataframe.
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159571 entries, 0 to 159570
Data columns (total 8 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   id                159571 non-null  object
 1   comment_text      159571 non-null  object
 2   malignant         159571 non-null  int64 
 3   highly_malignant  159571 non-null  int64 
 4   rude              159571 non-null  int64 
 5   threat            159571 non-null  int64 
 6   abuse             159571 non-null  int64 
 7   loathe            159571 non-null  int64 
dtypes: int64(6), object(2)
memory usage: 9.7+ MB
```

```
# Check the features, duplicate values and nan values in the Datasets

print("\nFeatures Present in the Dataset: \n", df_train.columns)
shape=df_train.shape
print("\nTotal Number of Rows : ",shape[0])
print("Total Number of Features : ", shape[1])
print("\n\nData Types of Features :\n", df_train.dtypes)
print("\nDataset contains any NaN/Empty cells : ", df_train.isnull().values.any())
print("\nTotal number of empty rows in each feature:\n", df_train.isnull().sum(),"\n\n")
print("Total number of unique values in each feature:")
for col in df_train.columns.values:
    print("Number of unique values of {} : {}".format(col, df_train[col].nunique()))
```

```
: # Check value counts for each feature

cols=['malignant', 'highly_malignant', 'rude', 'threat','abuse', 'loathe',]
for col in cols:
    print("Number of value_counts of {} : {}".format(col, df_train[col].nunique()))
    print(df_train[f'{col}'].value_counts())
```

- Data Pre-processing Done

  After loading all the required libraries, we loaded the data into our jupyter notebook.

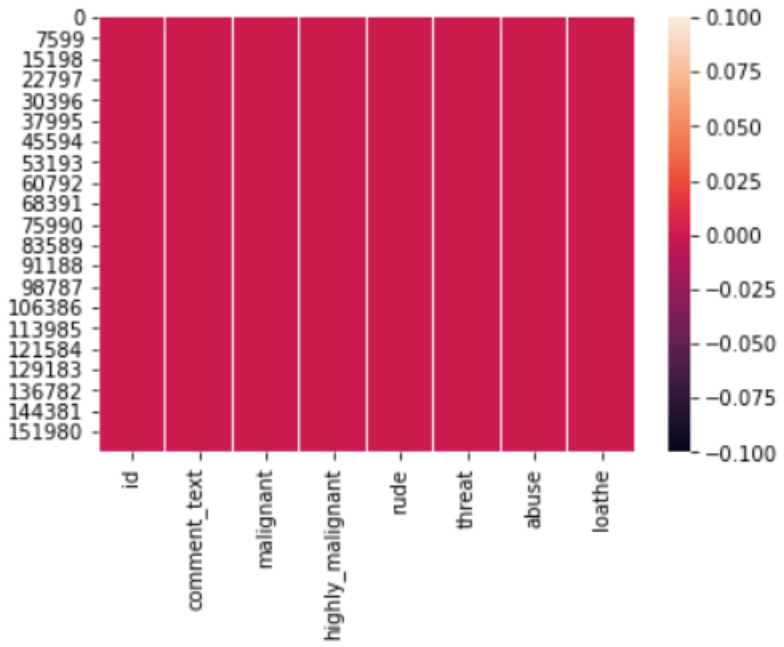
```
# Importing all the required libraries.

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from collections import Counter
import string
import re

# packages from gensim
!pip install gensim
from gensim import corpora
from gensim.parsing.preprocessing import STOPWORDS
from gensim.utils import simple_preprocess

# packages from sklearn
from sklearn.feature_extraction.text import TfidfVectorizer

# packages from nltk
import nltk
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer, SnowballStemmer
from nltk import pos_tag

import warnings
warnings.filterwarnings('ignore')
```

Feature Engineering has been used for cleaning of the data. We first did data cleaning. We first looked percentage of values missing in columns.

```
# Reading train dataset.
df_train=pd.read_csv('malignant train.csv')
```

```
# Reading test dataset.
df_test=pd.read_csv('malignant test.csv')
```

```
#checking null values using heatmap
sns.heatmap(df_train.isnull());
```



For Data pre-processing we did some data cleaning, where we used wordNetlemmatizerto clean the words and removed special characters using Regexp Tokenizer and filter the words by removing stop words and then used lemmatizers and joined and return the filtered words.

Used TFIDF vectorizer to convert those text into vectors, and split the data and into test and train and trained various Machine learning algorithms.

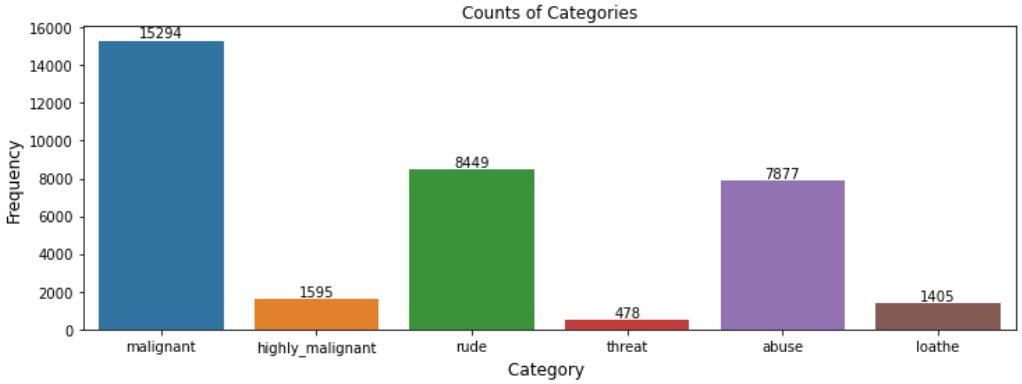
```
#Creating a function to filter using POS tagging.

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
```

```python
# Function for data cleaning...
def Processed_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()

    # iterating through each words and checking if they are stopwords or not,
    comments=[word for word in comments if not word in set(STOPWORDS)]

    # 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
```

- Data Inputs- Logic- Output Relationships

  EDA was performed by creating valuable insights using various visualization libraries

```
# Let's plot the counts of each category

plt.figure(figsize=(12,4))
ax = sns.barplot(counts.index, counts.values)
plt.title("Counts of Categories")
plt.ylabel('Frequency', fontsize=12)
plt.xlabel('Category ', fontsize=12)
rects = ax.patches
labels = counts.values
for rect, label in zip(rects, labels):
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2, height + 5, label, ha='center', va='bottom')
plt.show()
```



- Hardware and Software Requirements and Tools Used

Jupyter Notebook (Anaconda 3) – Python

Microsoft Excel 2010

LIBRARIES: The tools, libraries and packages we used for accomplishing this project are pandas, numpy, matplotlib, seaborn, scipy stats, etc.

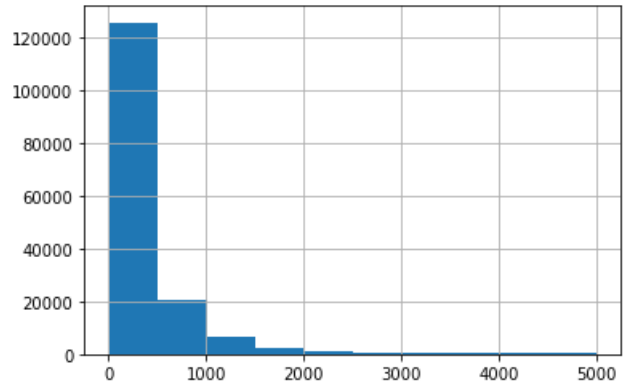
Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  The dataset is loaded and stored in a data frame. We need to perform some text processing to remove unwanted words and characters from our text. I used the nltk library and the string library. Then the data was analysed and visualized to extract insights about the comments. The sentence in the cleaned data, were broken down into vectors using Tokenizer from Keras and each word was converted into sequence of integers. Comments are variable in length, some are one-word replies while others are vastly elaborated thoughts. To overcome this issue, we use Padding. With the help of padding, we can make the shorter sentences as long as the others by filling the shortfall by zeros, and on the other hand, we can trim the longer ones to the same length as the short ones [3]. I used the “pad_sequences” function from the “Keras” library and, I fixed the sentence length at 200 words and applied pre padding (i.e., for shorter sentences, 0’s will be added at the beginning of the sequence vector) A model was built using Keras and Tensorflow. For our classification task, I used both CNN and LSTM neural networks. The model consisted of Embedding layer, which is responsible for embedding. MaxPool layer used to focus on the important features. Bi-directional LSTM was used for one forward and one backward network. Last layer consisted of Sigmoid layer, which will predict probabilities for each kind of features in our dataset. The training dataset was split into training and validation set. 20% of the training data was kept aside for validation. The model was compiled with various optimizers, amongst which adam performed better and metrics like loss and AUC were used to evaluate the model. The dataset was then fit on training data and validated on validation dataset. It gave a quite good AUC of about 98.3% with 2 epochs. The loss was also decreasing significantly with increase in epoch, and finally the model was used to predict on the testing dataset.

VISUALIZATIONS

```
# Let's Plot the Length in a histogram

lens.hist();
```

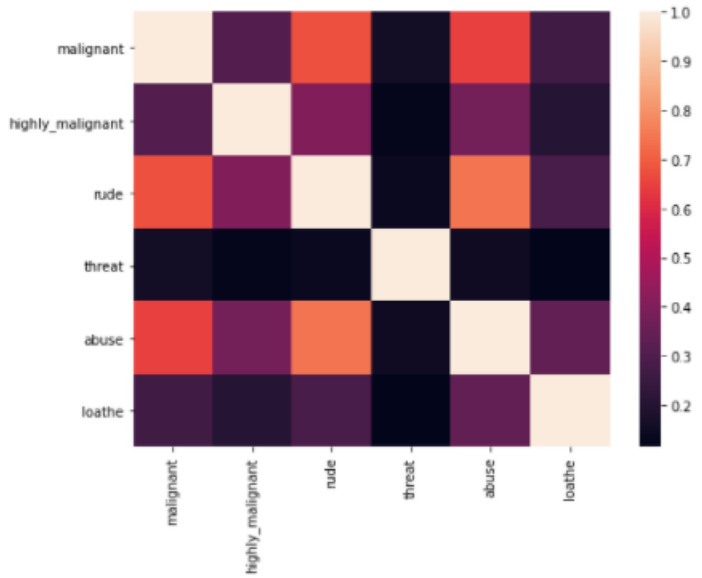



```
# Let's plot the correlation chart

df_train.corr().style.background_gradient(cmap='Blues_r')
```

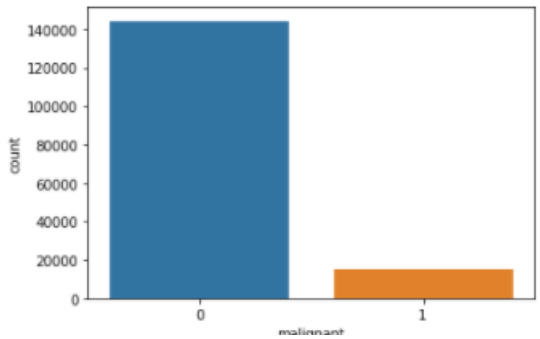
	malignant	highly_malignant	rude	threat	abuse	loathe
malignant	1.000000	0.308619	0.676515	0.157058	0.647518	0.266009
highly_malignant	0.308619	1.000000	0.403014	0.123601	0.375807	0.201600
rude	0.676515	0.403014	1.000000	0.141179	0.741272	0.286867
threat	0.157058	0.123601	0.141179	1.000000	0.150022	0.115128
abuse	0.647518	0.375807	0.741272	0.150022	1.000000	0.337736
loathe	0.266009	0.201600	0.286867	0.115128	0.337736	1.000000

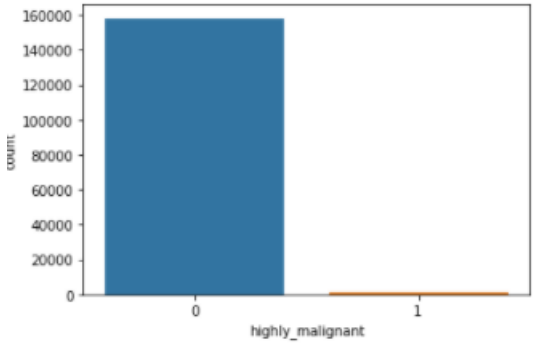
```
# Let's view the Correlation heatmap among variables
plt.figure(figsize=(8,6))
sns.heatmap(df_train.corr())
```

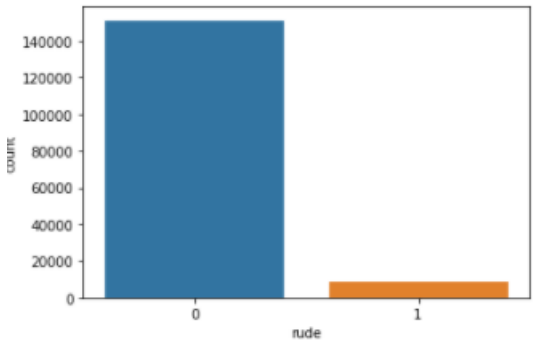
```
<AxesSubplot:>
```

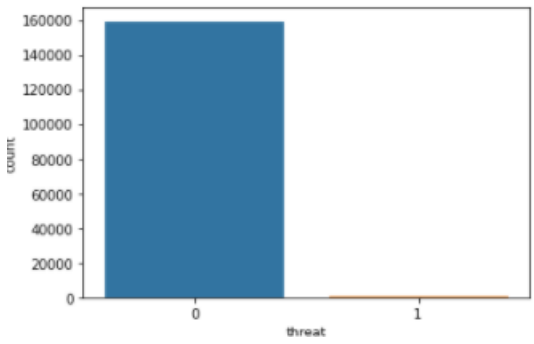


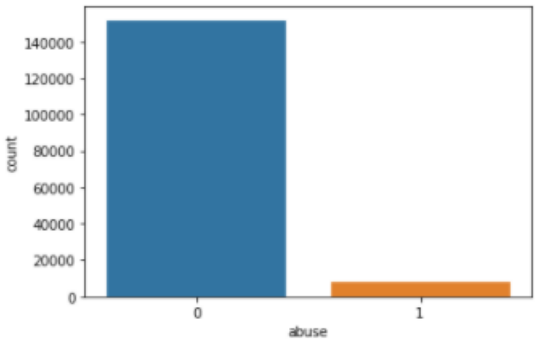

```
for i in features:
    sns.countplot(df_train[i])
    plt.show()
```

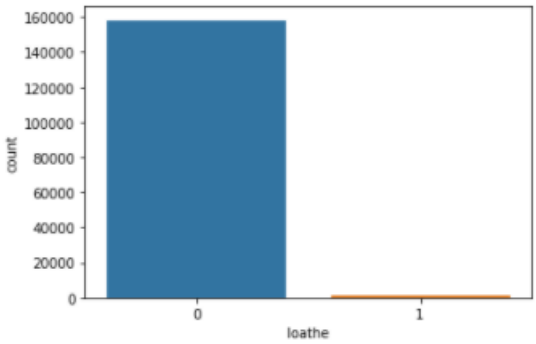



160000
140000
120000
100000
80000
60000
40000
20000
0
count
0
1
highly_malignant

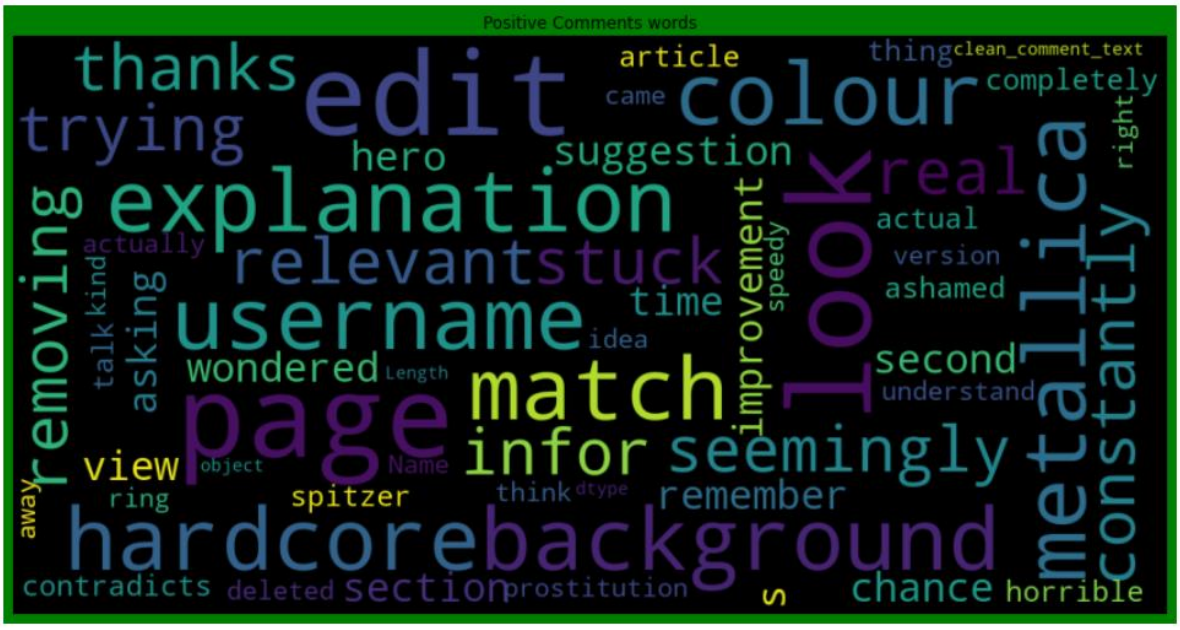

140000
120000
100000
80000
60000
40000
20000
0
count
0
1
rude


160000
140000
120000
100000
80000
60000
40000
20000
0
count
0
1
threat


140000
120000
100000
80000
60000
40000
20000
0
count
0
1
abuse


160000
140000
120000
100000
80000
60000
40000
20000
0
count
0
1
loathe

```
# Non-Negative/Good Comments - in training data
Display_wordcloud(df_train['clean_comment_text'][df_train['label']==0],"Positive Comments")
```




```
# Negative Comments - in training data
Display_wordcloud(df_train['clean_comment_text'][df_train['label']==1],"Negative Comments")
```

- Run and evaluate selected models

```
# Creating instances for different Classifiers

LR=LogisticRegression()
MNB=MultinomialNB()
DT=DecisionTreeClassifier()
KNN=KNeighborsClassifier()
RFC=RandomForestClassifier()
GBC=GradientBoostingClassifier()
```

```
# Creating a list model where all the models will be appended for further evaluation in loop.
models=[]
models.append(('LogisticRegression',LR))
models.append(('MultinomialNB',MNB))
models.append(('DecisionTreeClassifier',DT))
models.append(('KNeighborsClassifier',KNN))
models.append(('RandomForestClassifier',RFC))
models.append(('GradientBoostingClassifier',GBC))
```

```
# Displaying scores :
results=pd.DataFrame({'Model': Model,'Learning Score': Score,'Accuracy Score': Acc_score,'Cross Val Score':cvs,
                      'Auc_score':rocscore,'Log_Loss':lg_loss})
results
```

	Model	Learning Score	Accuracy Score	Cross Val Score	Auc_score	Log_Loss
0	LogisticRegression	95.777939	95.316678	96.406347	79.240073	1.617566
1	MultinomialNB	93.974879	93.547376	92.649067	68.846225	2.228658
2	DecisionTreeClassifier	99.826319	93.885779	83.471845	82.660860	2.111801
3	KNeighborsClassifier	92.353557	89.701705	69.054857	62.233465	3.556929
4	RandomForestClassifier	99.824528	95.433656	95.536326	81.126878	1.577166
5	GradientBoostingClassifier	94.247039	94.006935	88.922018	71.889209	2.069934

Observing all the Scores, I have selected Random Forest Classifier

Hyperparameter Tuning - Random Forest

```
from sklearn.model_selection import RandomizedSearchCV
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=42,test_size=.30,stratify=y)
parameters={'bootstrap': [True, False],
 'max_depth': [10, 50, 100, None],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [100, 300, 500, 800, 1200]}


LG=LogisticRegression()

# Applying Randomized Search CV for hyperparameter tuning with scoring= "accuracy"
rand = RandomizedSearchCV(estimator = RFC, param_distributions = parameters,
                          n_iter = 10, cv = 3, verbose=2, random_state=42, n_jobs = -1,scoring='accuracy')
rand.fit(x_train,y_train)
rand.best_params_
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

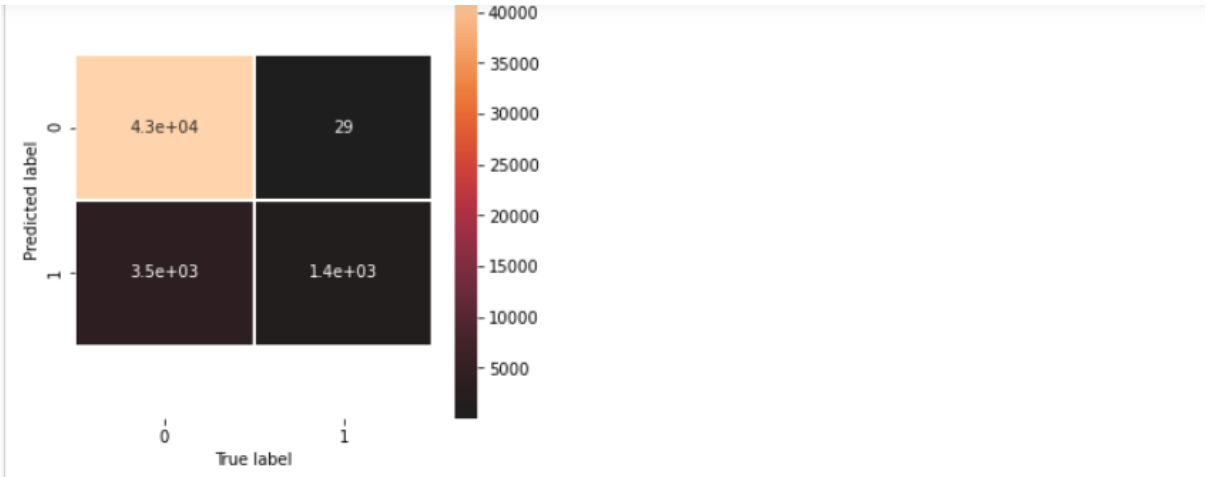
```
{'n_estimators': 500,
 'min_samples_split': 2,
 'min_samples_leaf': 1,
 'max_depth': 100,
 'bootstrap': False}
```

```
RFC=RandomForestClassifier(n_estimators= 500,
                           min_samples_split= 2,
                           min_samples_leaf=1,
                           max_depth= 100,
                           bootstrap= False)
```

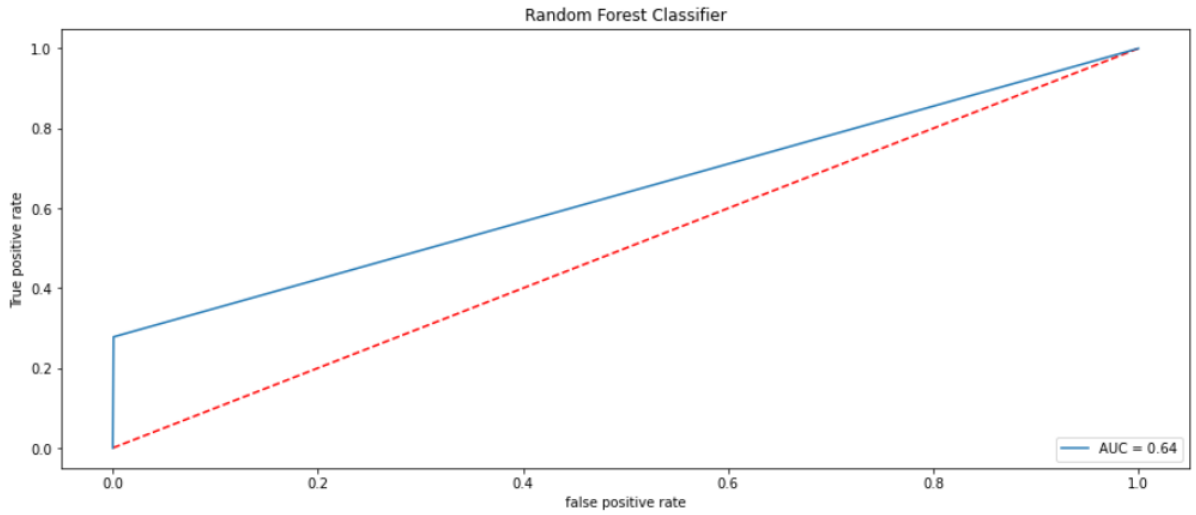
```
RFC.fit(x_train,y_train)
RFC.score(x_train,y_train)
pred=RFC.predict(x_test)
print('Accuracy Score:',accuracy_score(y_test,pred))
print('Log loss : ', log_loss(y_test,pred))
print('Confusion Matrix:',confusion_matrix(y_test,pred))
print('Classification Report:','\n',classification_report(y_test,pred))
```

```
Accuracy Score: 0.9259692513368984
Log loss :  2.5569319588044985
Confusion Matrix: [[42975    29]
 [ 3515  1353]]
Classification Report:
               precision    recall  f1-score   support

           0       0.92      1.00      0.96     43004
           1       0.98      0.28      0.43      4868

    accuracy                           0.93     47872
   macro avg       0.95      0.64      0.70     47872
weighted avg       0.93      0.93      0.91     47872
```

```
# Confusion matrix Visualization
fig, ax =plt.subplots(figsize=(5,5))
sns.heatmap(confusion_matrix(y_test, pred),annot=True,linewidths=1,center=0)
plt.xlabel("True label")
plt.ylabel("Predicted label")
bottom, top = ax.get_ylim()
ax.set_ylim(bottom + 0.5, top - 0.5)
```




```
# Roc-Auc score
f,ax = plt.subplots(figsize = (15,6))
# Calculate fpr, tpr and thresholds
fpr, tpr, thresholds = roc_curve(y_test, pred)
ax.plot([0,1],[0,1],'r--')
ax.plot(fpr,tpr,label='AUC = %0.2f'% roc_auc_score(y_test, pred))
ax.legend(loc='lower right')
ax.set_xlabel('false positive rate')
ax.set_ylabel('True positive rate')
ax.set_title('Random Forest Classifier')
```




```
def Tf_idf_test(text):
    tfid = TfidfVectorizer(max_features=43194,smooth_idf=False)
    return tfid.fit_transform(text)
```

PREDICTION

```
x_testing_data=Tf_idf_test(df_test['clean_comment_text'])
```

```
x_testing_data.shape
```

```
(153164, 43194)
```

```
Prediction=RFC.predict(x_testing_data)
df_test['Predicted values']=Prediction
df_test
```

	id	comment_text	comment_length	clean_comment_text	clean_comment_length	Predicted values
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...	367	bitch rule succesful whats hating mofuckas bit...	184	0
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...	50	title fine	10	0
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...	54	source zawe ashton lapland	26	0
3	00017563c3f7919a	:If you have a look back at the source, the in...	205	look source information updated correct form g...	109	0
4	00017695ad8997eb	I don't anonymously edit articles at all.	41	anonymously edit article	24	0
...	...	...	...	...	...	...
153159	fffcd0960ee309b5	. \n i totally agree, this stuff is nothing bu...	60	totally agree stuff long crap	29	0
153160	fffd7a9a6eb32c16	== Throw from out field to home plate. == \n\n...	198	throw field home plate faster throwing direct ...	85	0
153161	fffda9e8d6fafa9e	" \n\n == Okinotorishima categories == \n\n I ...	423	okinotorishima category change agree correct g...	212	0
153162	fffe8f1340a79fc2	" \n\n == ""One of the founding nations of the...	502	founding nation germany return similar israel ...	275	0
153163	ffffce3fb183ee80	" \n :::Stop already. Your bullshit is not wel...	141	stop bullshit welcome fool think kind explinat...	54	0

153164 rows × 6 columns

CONCLUSION

- Key Findings and Conclusions of the Study
  - 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.
  - From the above analysis the below mentioned results were achieved which depicts the chances and conditions of a comment being a hateful comment or a normal comment.
  - With the increasing popularity of social media, more and more people consume feeds from social media and due differences they spread hate comments to instead of love and harmony. It has strong negative impacts on individual users and broader society.
- Learning Outcomes of the Study in respect of Data Science

  It is possible to classify the comments content into the required categories of Malignant and Non-Malignant. However, using this kind of project an awareness can be created to know what is good and bad. It will help to stop spreading hatred among people.
- Limitations of this work and Scope for Future Work
  - Machine Learning Algorithms like Decision Tree Classifier took enormous amount of time to build the model and Ensemble techniques were taking a lot more time thus I have not included Ensemble models.
  - Using Hyper-parameter tuning would have resulted in some more accuracy.
  - Every effort has been put on it for perfection but nothing is perfect and this project is of no exception. There are certain areas which can be enhanced. Comment detection is an emerging research area with few public datasets. So, a lot of works need to be done on this field.

THANK YOU