PROJECT REPORT ON:

“Malignant Comment Classifier”

SUBMITTED BY

Ajit Madame

ACKNOWLEDGMENT

I would like to express my special gratitude to "Flip Robo" team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills. And I want to express my huge gratitude to Ms.Gulshana Chaudhary (SME Flip Robo), she is the person who has helped me to get out of all the difficulties I faced while doing the project.

A huge thanks to my academic team "Data trained" who are the reason behind what I am today. Last but not least my parents who have been my backbone in every step of my life. And also thank you for many other persons who has helped me directly or indirectly to complete the project.

Contents:

1.INTRODUCTION

1.1Business 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 unoffensive, 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.

1.2 Conceptual Background of the Domain Problem

Online forum and social media platforms have provided individuals with the means to put forward their thoughts and freely express their opinion on various issues and incident. In some cases, these online comments contain explicit language which may hurt the readers. Comments containing explicit language can be classified into myriad categories such as Malignant, Highly Malignant, Rude, Threat, Abuse and Loathe. The threat of abuse and harassment means that people stop expressing themselves and give up on seeking different opinions.

To protect users from being exposed to offensive language on online forums or social media sites, companies have started flagging comments and blocking users who are found guilty of using unpleasant language. Several Machine Learning models have been developed and deployed to filter out the unruly language and protect internet users from becoming victims of online harassment and cyberbullying.

1.3 Review of Literature

The purpose of the literature review is to:

1. Identify the foul words or foul statements that are being used.

2. Stop the people from using these foul languages in online public forum.

To solve this problem, we are now building a model using our machine learning that identifies all the foul language and foul words, using which the online platforms like social media principally stops these mob using the foul language in an online community or even block them or block them from using this foul language.

I have used different Classification algorithms and shortlisted the best on basis of the metrics of performance and I have chosen one algorithm and build a model in that algorithm.

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.

1.4 Motivation for the Problem Undertaken

Now a days social media users are increasing continuously so regularization on social media platform is very necessary. Many users are abuse or harass from social media, and teenager are having bad impact of this because they may face threat call or massages social

media due to this, they take unwanted or unnecessary step. For Avoiding this, if we install a machine whose filter out the comment. So, we will be decreasing the thus threat or unwanted activities from social media platforms. One of the first lessons we learn as children is that the louder you scream and the bigger of a tantrum you throw, you more you get your way. Part of growing up and maturing into an adult and functioning member of society is learning how to use language and reasoning skills to communicate our beliefs and respectfully disagree with others, using evidence and persuasiveness to try and bring them over to our way of thinking.

2.Analytical Problem Framing

2.1 Mathematical/ Analytical Modelling of the Problem

The libraries/dependencies imported for this project are shown below:

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

```
import nltk
nltk.download('stopwords', quiet=True)
nltk.download('punkt', quiet=True)
from wordcloud import WordCloud
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
from nltk.tokenize import word_tokenize, regexp_tokenize

from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV,
from scipy.sparse import csr_matrix
```

```
from sklearn import metrics
from sklearn.svm import SVC, LinearSVC
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.metrics import roc_curve, auc, roc_auc_score, multilabel_confusion_matrix
```

Here in this project, we have been provided with two datasets namely train and test CSV files. I will build a machine learning model using train dataset. And using this model we will make predictions for our test dataset.

2.2 Data Sources and their formats

We have been provided with two datasets namely train and test CSV filers. Train datasets contains 159571 rows and 8 columns.

	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

2.3 Data Pre-processing Done

- First step I have imported required libraries and I have imported the dataset which was in csv format.
- Then I did all the statistical analysis like checking shape, nunique, value counts, info etc.
- I found that, only one features are used to make prediction and here are 6 categories to predict.
- Apply Stemming using SnowballStemmer.
- Here I convert text data into vector form by using TfidfVectorizer.

- Then doing some EDA and Building Models.

2.4 Data Inputs - Logic - Output Relationships

I have analysed the input output logic with word cloud and I have word clouded the sentenced that as classified as foul language in every category. A tag/word cloud is a novelty visual representation of text data, typically used to depict keyword metadata on websites, or to visualize free form text. It's an image composed of words used in a particular text or subject, in which the size of each word indicates its frequency or importance.

```
: #Getting sense of loud words which are offensive
from wordcloud import WordCloud
hams = data['comment_text'][data['malignant']==1]
spam_cloud = WordCloud(width=600,height=400,background_color='white',max_words=50).generate(' '.join(hams))
plt.figure(figsize=(4,4),facecolor='k')
plt.imshow(spam_cloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```

2.5 Hardware, Software and Tool Used

Hardware Used:

Processor – Intel core i3

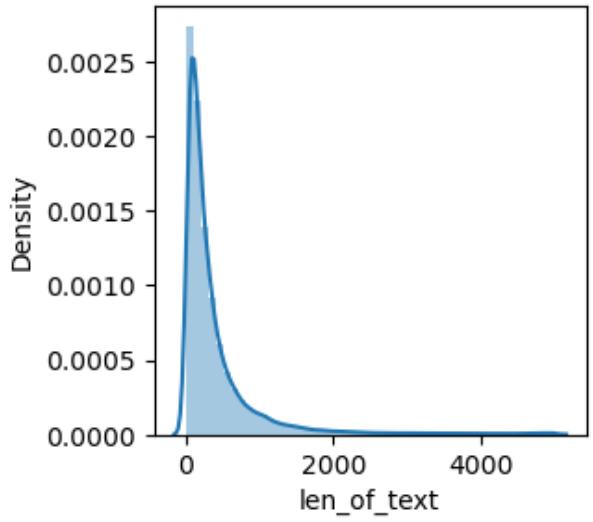
Physical Memory – 8 GB

Software Used:

- Windows 10 Operating System
- Anaconda Package and Environment Manager
- Jupyter Notebook
- Python Libraries used: In Which Pandas, Seaborn, Matplotlib, Numpy and Scipy
- sklearn for Modelling Machine learning algorithms, Data Encoding, Evaluation metrics, Data Transformation, Data Scaling, Component analysis, Feature selection etc.

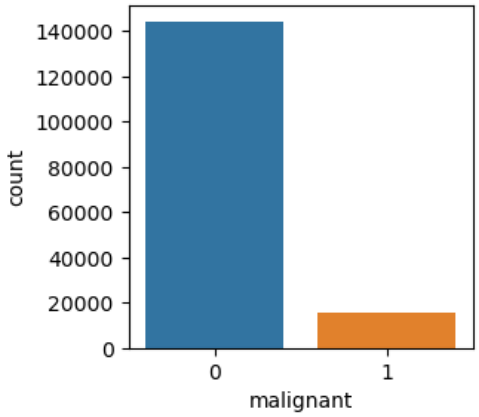
3.Data Analysis and Visualization

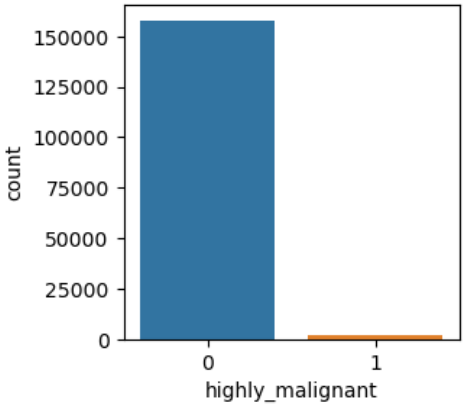
3.1 Univariate Visualization

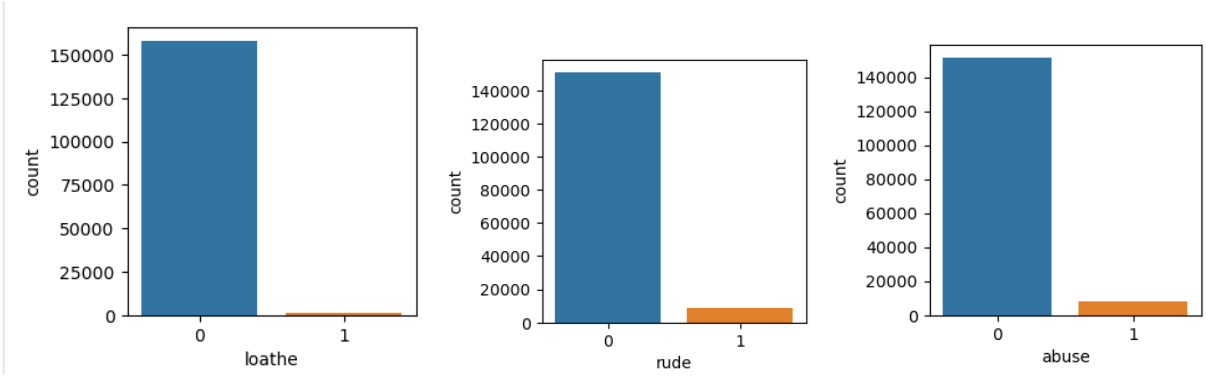


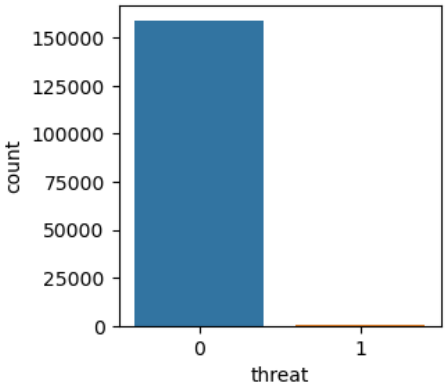
- We can see, up to 2000 words comment test are used. It seems to be people are used so many malignant words while commenting.

- But there is also more 4000 words but in less numbers are also used malignant words to commenting.









- We can see, in malignant categories of comment there are having less malignant comment. 1 has low count than 0 it means that0 may be normal comment 1 may be malignant.

- In Highly malignant 1 has less than 0 it meant that there is less highly malignant word using by users. But their magnitude is high because it may cause the depression to others users.
- In loathe, there is also low percentage of 1 category.
- In rude categories, rude comment is high than others categories so it may cause another user.
- Rude and Abuse is having almost same quantity of comment but their magnitude or impact will be different. Rude has low impact than abuse comment.
- We can see, there is also threat comment but their ration very small but it may have very large impact.

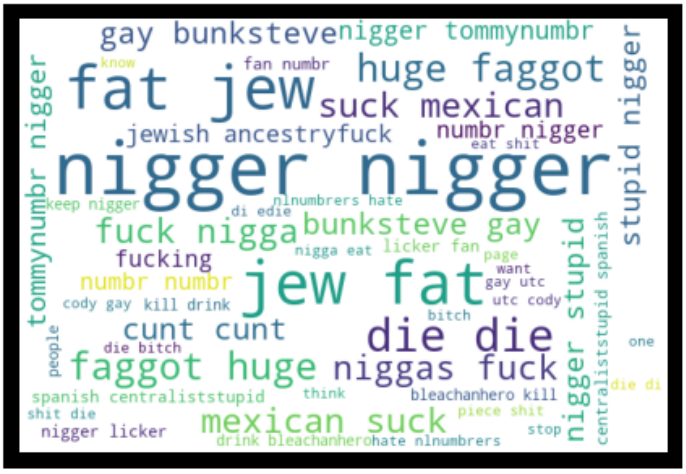
3.2 Word Cloud Visualization

- We can see, so many malignant comments are used by users which show in above.
- Pig, wanker, hate, die and shit these are the word are mostly used.

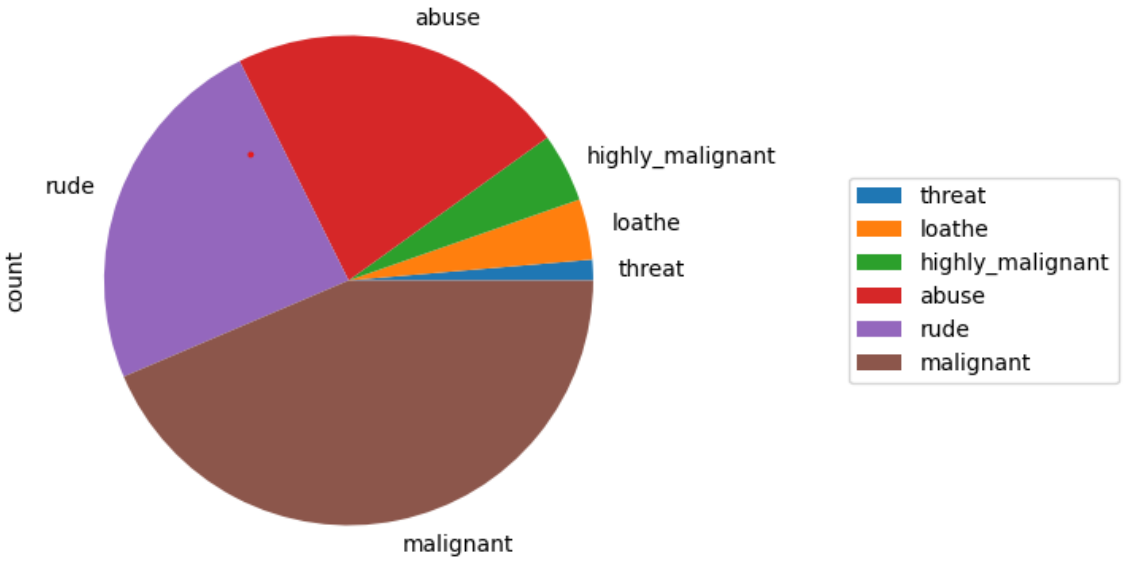
- The vocabulary used in all categories is quite similar (expect for 'none' of course). Frequencies are varying a bit across (for example 'fuck' and 'suck'.

- We can see, ass, Jim, wale, fucking, die and going kill these are the word are mostly used by users for threatening the another's users.
- Due to this impact of threat massages will causes several risks to users so we need to improve the social media to make user friendly.

- There are so many words are used by user for abusing someone. like, faggot, fuck, Morongo, suck, and jew.
- Here also have highly malignant word also.

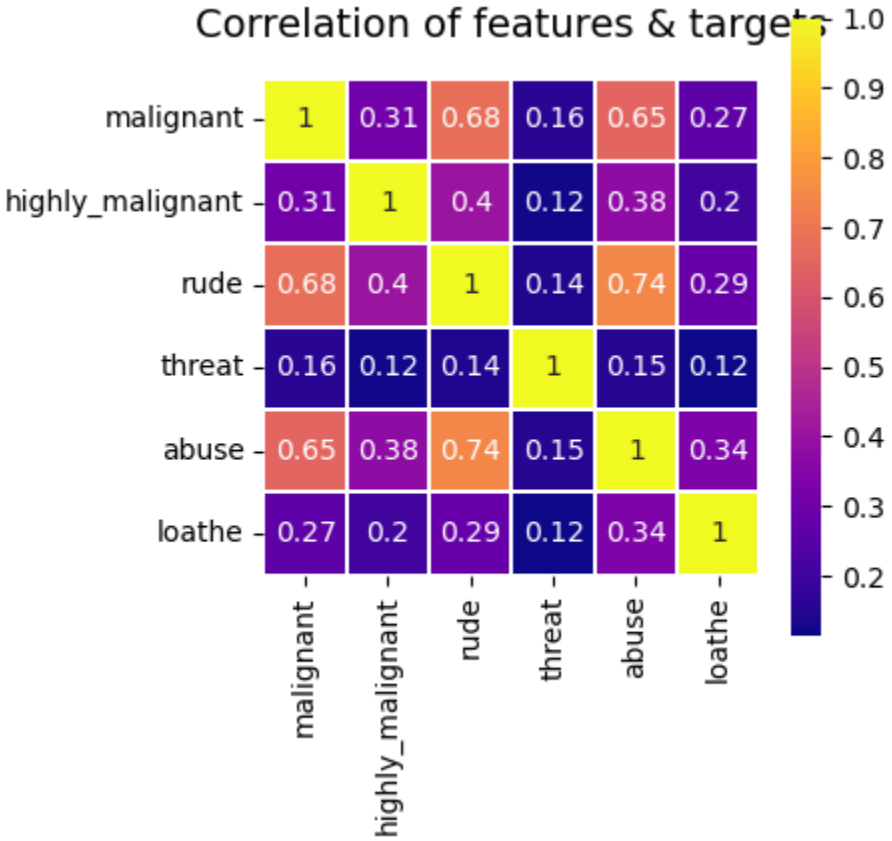


- We can see loathe comment where so many word are used for commenting on others some of is nigger, die, gay, Mexican, stupid these words are used to loathe the people or users.



- We can see, malignant is having maximum comment than others categories followed by rude.
- Threat categories comment is having low count but it has high impact than others.

- Highly malignant comments are having high impact on user whose face such problem.



- Indeed, it looks like some of the labels are higher correlated, e.g., abuse-rude has the highest at 0.74,
- followed by malignant-rude and malignant-abuse.

4. Models Development and Evaluation

```
# Convert all messages to lower case
data['comment_text'] = data['comment_text'].str.lower()

# Replace email addresses with 'email'
data['comment_text'] = data['comment_text'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',
                                 'emailaddress')

# Replace URLs with 'webaddress'
data['comment_text'] = data['comment_text'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                  'webaddress')

# Replace money symbols with 'moneysymb' (£ can by typed with ALT key + 156)
data['comment_text'] = data['comment_text'].str.replace(r'£|\$', 'dollers')

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


# Replace numbers with 'numbr'
data['comment_text'] = data['comment_text'].str.replace(r'\d+(\.\d+)?', 'numbr')


data['comment_text'] = data['comment_text'].apply(lambda x: ' '.join(
    term for term in x.split() if term not in string.punctuation))

stop_words = set(stopwords.words('english') + ['u', 'ü', 'ur', '4', '2', 'im', 'dont', 'doin', 'ure'])
data['comment_text'] = data['comment_text'].apply(lambda x: ' '.join(
    term for term in x.split() if term not in stop_words))

lem=WordNetLemmatizer()
data['comment_text'] = data['comment_text'].apply(lambda x: ' '.join(
 lem.lemmatize(t) for t in x.split()))
```

```
#  Convert text into vectors using TF-IDF
from sklearn.feature_extraction.text import TfidfVectorizer
tf_vec = TfidfVectorizer(max_features = 10000, stop_words='english')
features = tf_vec.fit_transform(df1.comment_text).toarray()
x = features
```

```
# Convert text into Vectors by using TfidVectorizer

tf = TfidfVectorizer(max_features=4000)
features = tf.fit_transform(df1.comment_text).toarray()
```

```
# Input Variables
X = features

# Output Variable
Y = csr_matrix(df1[output_labels]).toarray()
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import hamming_loss, accuracy_score

model = OneVsRestClassifier(estimator=LogisticRegression())
model.fit(x_train, y_train)

prediction = model.predict(x_test)
print('Accuracy Score: ', accuracy_score(y_test, prediction))
print('hamming loss : ', hamming_loss(y_test, prediction))
```

```
Accuracy Score:  0.88
hamming loss :  0.04055555555555555
```

Random Forest Classifier

```
model = OneVsRestClassifier(estimator=RandomForestClassifier())
model.fit(x_train, y_train)

prediction = model.predict(x_test)
print('Accuracy Score: ', accuracy_score(y_test, prediction))
print('hamming loss : ', hamming_loss(y_test, prediction))
```

```
Accuracy Score:  0.89
hamming loss :  0.03138888888888889
```

Support Vector Machine

```
model = OneVsRestClassifier(estimator=SVC())
model.fit(x_train, y_train)

prediction = model.predict(x_test)
print('Accuracy Score: ', accuracy_score(y_test, prediction))
print('hamming loss : ', hamming_loss(y_test, prediction))
```

```
Accuracy Score:  0.8866666666666667
hamming loss :  0.03611111111111111
```

4.2 Interpretation of the results

Based on comparing Accuracy Score results with hamming loss results, it is determined Random Forest Classifier is the best model. It has least difference between accuracy score and hamming loss.

4.3 Hyperparameter Tuning

```
# RandomForestClassifier
params = {'n_estimators':[13,15],
          'criterion':['entropy','gini'],
          'max_depth':[10,15],
          'min_samples_split':[10,11],
          'min_samples_leaf':[5,6]}
```

```
rf = RandomForestClassifier()
```

```
rf = OneVsRestClassifier(estimator=RandomForestClassifier())
```

```
grd = GridSearchCV(rf,param_grid = params)
grd.fit(x_train,y_train)

print('Best_params  = > ',grd.best_params_)
```

```
Best_params  = >  {'criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 13}
```

```
VsRestClassifier(estimator=RandomForestClassifier(criterion='gini',min_samples_leaf=1, max_depth=20,min_samples_split=2, n_estima
```

```
rf.fit(x_train, y_train)

prediction = model.predict(x_test)
print('Accuracy Score: ', accuracy_score(y_test, prediction))
print('hamming loss : ', hamming_loss(y_test, prediction))
```

```
Accuracy Score:  0.8866666666666667
hamming loss :  0.03611111111111111
```

5. Conclusions

5.1 Key Finding and Conclusions

The finding of the study is that only few users over online use unparliamentary language. And most of these sentences have more stop words and are being quite long. As discussed before few motivated

disrespectful crowds use these foul languages in the online forum to bully the people around and to stop them from doing these things that they are not supposed to do. Our study helps the online forums and social media to induce a ban to profanity or usage of profanity over these forums.

5.2 Limitation of this works and scope for future works

Problems faced while working in this project:

- More computational power was required as it took more than 2 hours
- Imbalanced dataset and bad comment texts
- Good parameters could not be obtained using hyperparameter tuning as time was consumed more

Areas of improvement:

- Could be provided with a good dataset which does not take more time.
- Less time complexity
- Providing a proper balanced dataset with less errors.