Project report on

Malignant Comment Classifier

Submitted By

Mr. Abhinandan Kadam

ACKNOWLEDGMENT

It is my sensual gratification to present this report on MALIGNANT COMMET CLASSIFIER project. Working on this project was an good experience that has given me a basic knowledge about machine learning model with NLP.

I would like to express my sincere thanks to MR. KESHAV BANSAL for a regular follow up and valuable guidance provided throughout.

And I am also thankful to FlipRobo Technologies Bangalore for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the 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, cyber bullying, 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 cyber bullying 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.

Conceptual background of domain problem

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 cyber bullying.

Review of Literature

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’.

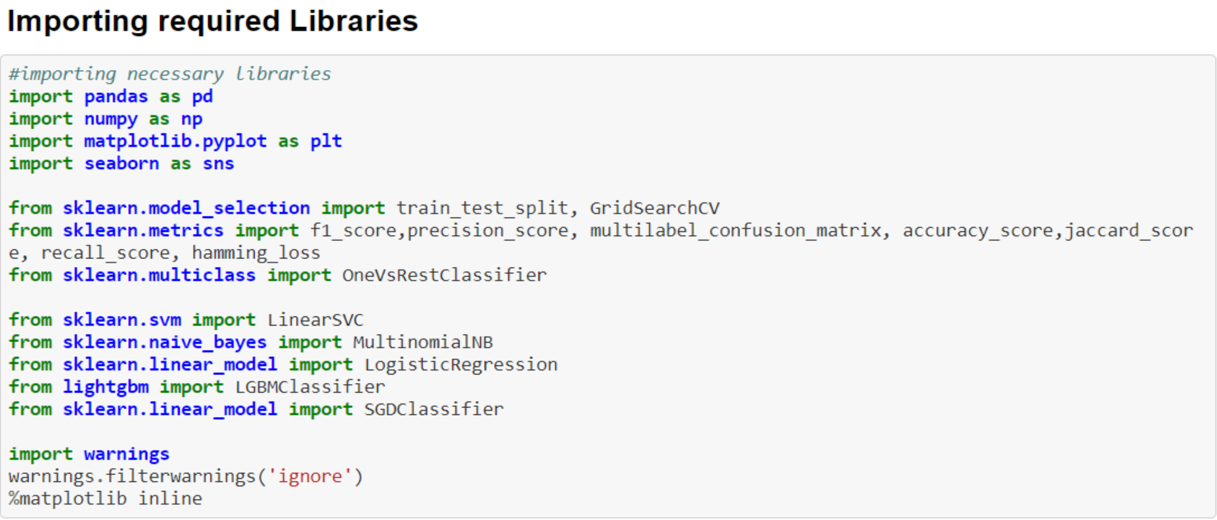
The label can be either 0 or 1, where 0 denotes a NO while 1 denotes a YES. There are various comments which have multiple labels. The first attribute is a unique ID associated with each comment.

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.

This project is more about exploration, feature engineering and classification that can be done on this data. Since the data set is huge and includes many categories of comments, we can do good amount of data exploration and derive some interesting features using the comments text column available.

We need to build a model that can differentiate between comments and its categories.



Importing required Libraries

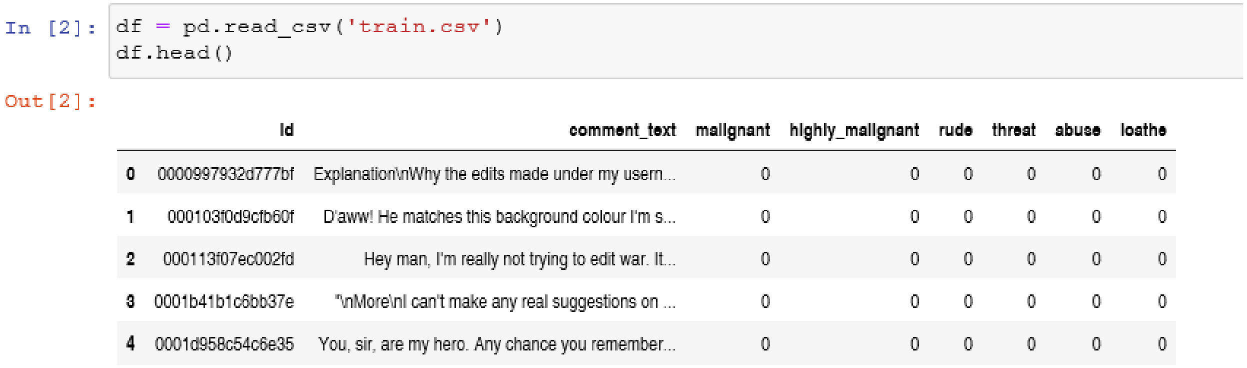
```
#importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import f1_score,precision_score, multilabel_confusion_matrix, accuracy_score,jaccard_scor
e, recall_score, hamming_loss
from sklearn.multiclass import OneVsRestClassifier

from sklearn.svm import LinearSVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from lightgbm import LGBMClassifier
from sklearn.linear_model import SGDClassifier

import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

Analytical Problem Framing

– Mathematical/Analytical modeling of the problem

Here in this project we have provided two datasets as train and test. I will build a machine learning model by using NLP using train dataset. And using this model we will make predictions for our test dataset.

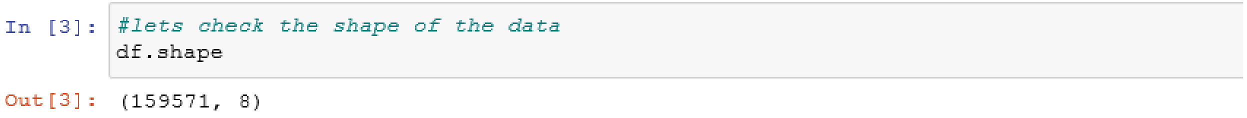
Loading the Data:



```
In [2]: df = pd.read_csv('train.csv')
        df.head()
```

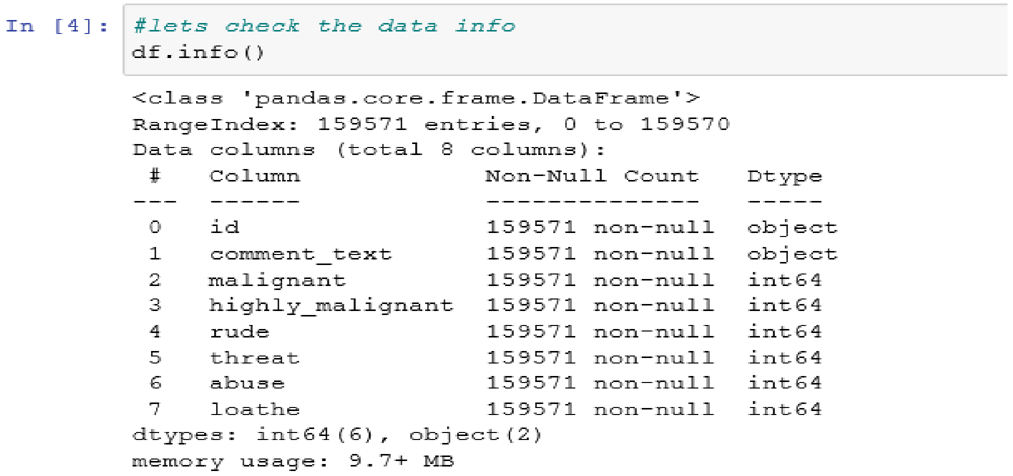
Out[2]:

	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



```
In [3]: #lets check the shape of the data
        df.shape

Out[3]: (159571, 8)
```

By looking at the shape of our training data set we came to know that this data set is having 159571 rows and 8 columns. Where malignant, highly_malignant, rude,threat, abuse and loathe are our target variabls which are with binary values.



```
In [4]: #lets check the data info
        df.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
```

By looking at the info of the train data set we came to know that the columns id and comment_text are of object type; rest all columns are having int data type.

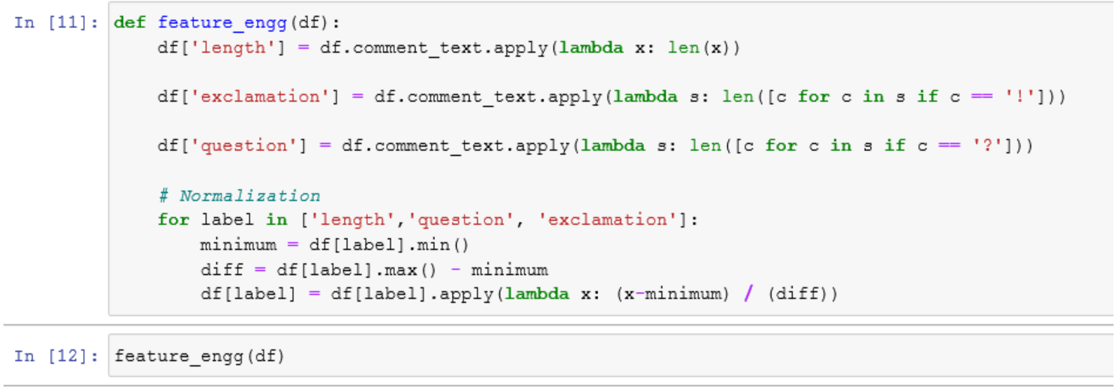
Feature Engineering

I am creating a function for feature engineering and making three different columns using comment_text column

Length: indicating the length of the text.

Exclamation: indicates whether ‘!’ is present in the text or not.

Question: indicates whether ‘?’ is present in the text or not.



```
In [11]: def feature_engg(df):
             df['length'] = df.comment_text.apply(lambda x: len(x))

             df['exclamation'] = df.comment_text.apply(lambda s: len([c for c in s if c == '!']))

             df['question'] = df.comment_text.apply(lambda s: len([c for c in s if c == '?']))

             # Normalization
             for label in ['length','question', 'exclamation']:
                 minimum = df[label].min()
                 diff = df[label].max() - minimum
                 df[label] = df[label].apply(lambda x: (x-minimum) / (diff))

In [12]: feature_engg(df)
```

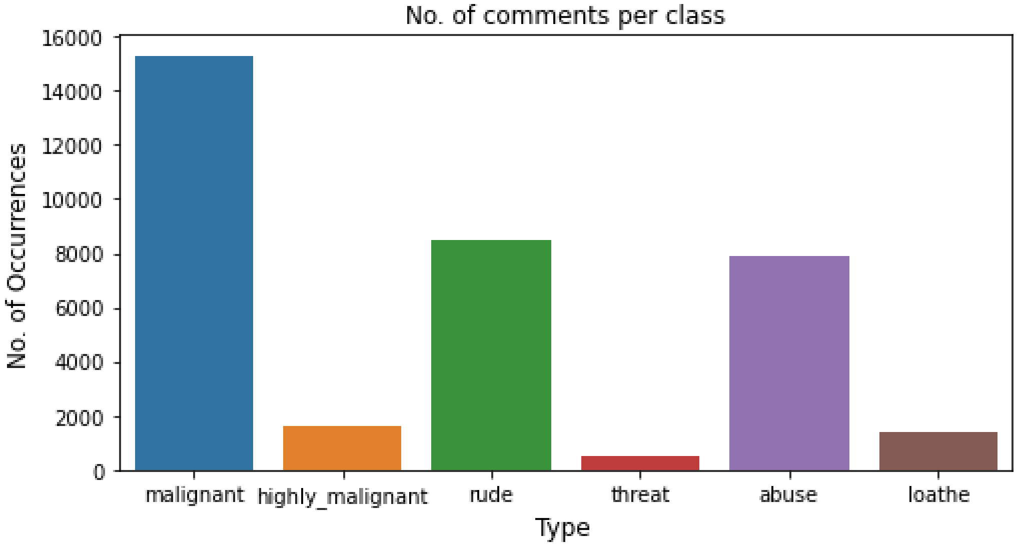
Text processing

By observing these comments we can say that we need to do lot of text processing as there are many words which are not important for prediction, as well as numbers and other stuff.

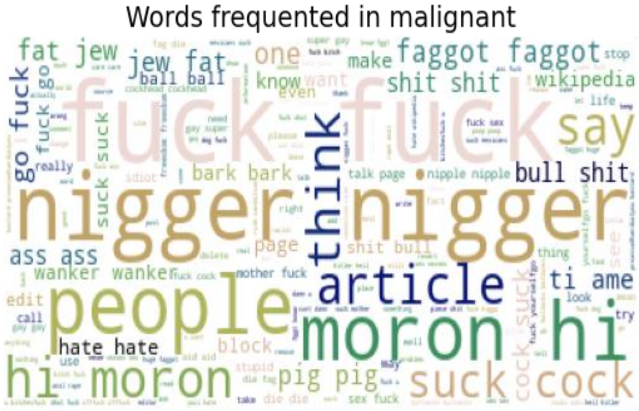
For text processing I am using nltk library and RegEx. And did following processing steps

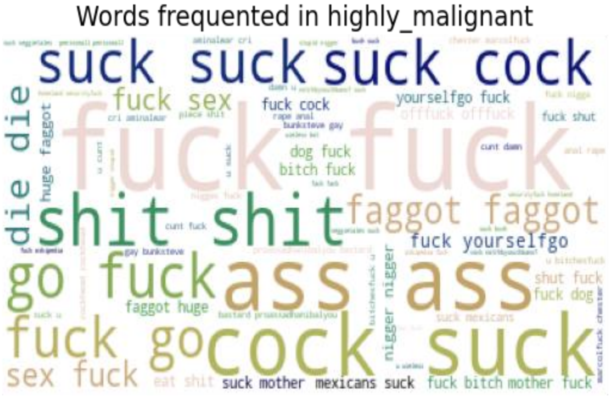
- Lowercasing the text.
- Removing numbers.
- Removing \n
- Some word corrections.
- Lemmatization
- Removing punctuations.
- Removing stopwords.

EDA

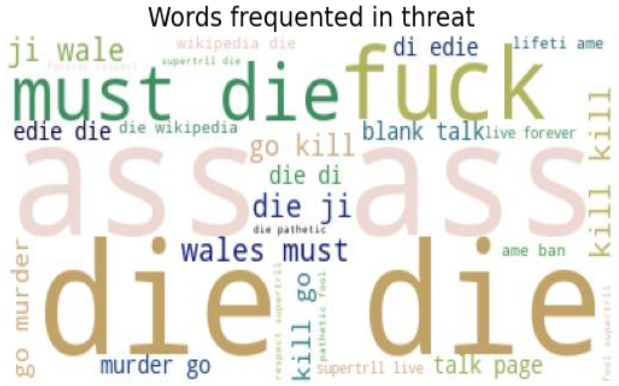


The above figure represents count plot for all our labels. Looking at this plot we can conclude that more number of comments has been labelled as malignant compared to others. Very less number of comments has been labelled as threat.

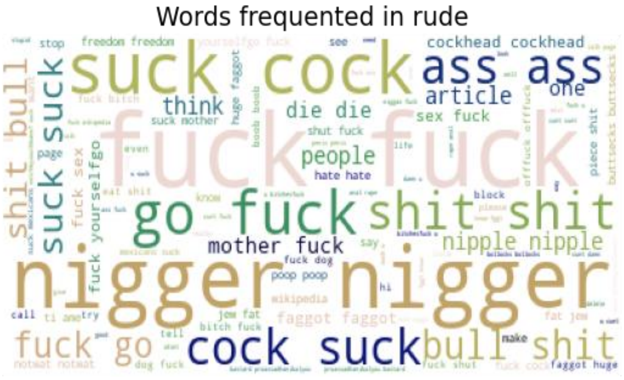




The above both figures are representing the word occurrence in case of malignant and highly malignant comments respectively.



Words frequented in threat



Words frequented in rude

The above both figures are representing the word occurrence in case of threat and highly rude comments respectively.

Hardware and Software Requirements and Tools Used

- For hardware I have used my laptop that have i5 processor and 8gb ram
- For software I have used Jupyter notebook
- For Tools I have use this following library-
  Numpy
  Pandas
  Seaborn
  Matplotlib
  Sklearn, wordcloud
  Nltk and RegEX for text processing

Model/s Development and Evaluation

In this nlp based project we need to predict multiple targets which are binary. I have converted the text into vectors using Tfidf vectorizer and separated our feature and labels then build the model using OneVsRestClassifier. Among all the algorithms which I have used for this purpose I have chosen LinearSVC as best suitable algorithm for our final model as it is performing well compared to other algorithms while evaluating with different metrics
I have used following algorithms and evaluated them

- LinearSVC
- LogisticRegression
- MultinomialNB
- LightGBMClassifier
- SGDClassifier

From all of these above models LinearSVC was giving me good performance.

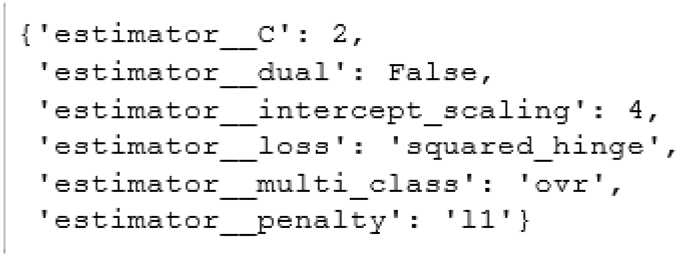
Key Metrics for success in solving problem under consideration

I have used the following metrics for evaluation:

- As this is classification problem I am using accuracy score here. But this is a case of multi-target classification so we can’t consider accuracy_score as a major factor.
- I have used jaccard score. Jaccard similarity coefficient score. The Jaccard index [1], or Jaccard similarity coefficient, defined as the size of the intersection divided by the size of the union of two label sets, is used to compare set of predicted labels for a sample to the corresponding set of labels in y_true .
- I have also used f1_score, precision_score, recall_score, multilabel_confusion_matrix and hamming loss all these evaluation metrics to select best suitable algorithm for our final model.

Hyperparameter Tuning

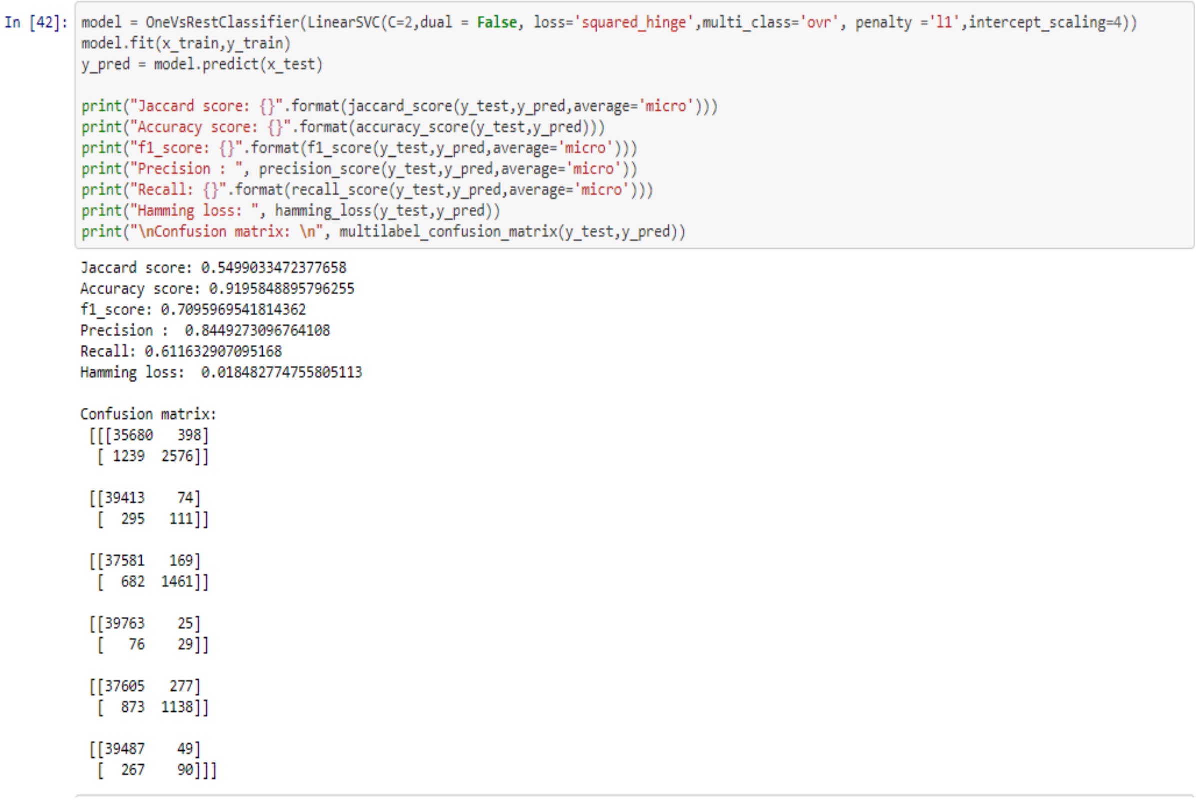
I have did hyperparameter tuning for LinearSVC for the parameters like ‘estimator__penalty, ‘estimator__loss’, ‘estimator__multi_class’, ’ estimator__dual’, ‘estimator__intercept_scaling’, ‘estimator__C’.



```
{'estimator__C': 2,
 'estimator__dual': False,
 'estimator__intercept_scaling': 4,
 'estimator__loss': 'squared_hinge',
 'estimator__multi_class': 'ovr',
 'estimator__penalty': 'l1'}
```

And after doing hyperparameter tuning I got above parameters as best suitable parameters for our final model.

I have tested my final model using these parameters and got better results compared to earlier results for my final model.



```
In [42]: model = OneVsRestClassifier(LinearSVC(C=2,dual = False, loss='squared_hinge',multi_class='ovr', penalty ='l1',intercept_scaling=4))
         model.fit(x_train,y_train)
         y_pred = model.predict(x_test)

         print("Jaccard score: {}".format(jaccard_score(y_test,y_pred,average='micro')))
         print("Accuracy score: {}".format(accuracy_score(y_test,y_pred)))
         print("f1_score: {}".format(f1_score(y_test,y_pred,average='micro')))
         print("Precision : ", precision_score(y_test,y_pred,average='micro'))
         print("Recall: {}".format(recall_score(y_test,y_pred,average='micro')))
         print("Hamming loss: ", hamming_loss(y_test,y_pred))
         print("\nConfusion matrix: \n", multilabel_confusion_matrix(y_test,y_pred))
```

```
Jaccard score: 0.5499033472377658
Accuracy score: 0.9195848895796255
f1_score: 0.7095969541814362
Precision :  0.8449273096764108
Recall: 0.611632907095168
Hamming loss:  0.018482774755805113

Confusion matrix:
 [[[35680   398]
  [ 1239  2576]]

 [[39413    74]
  [  295   111]]

 [[37581   169]
  [  682  1461]]

 [[39763    25]
  [   76    29]]

 [[37605   277]
  [  873  1138]]

 [[39487    49]
  [  267    90]]]
```

After training and building our final model I used this model to make predictions for test dataset. Before doing predictions the test dataset has been cleaned and processed with the same functions which are used for train dataset. And then doing vectorization I have predicted the output labels with our final model.

Conclusion

Key findings of the study

- For this project we have provided with huge amount of comments with multiple targets which are binary in nature. I observe that there are many words with incorrect spellings. At first I have created three columns one is with the length of the text, another as 'question' whether the comment contains '?' mark or not and third as 'exclamation' whether the comment contains '!' mark.
- To clean the column comment_text I have gone through different text processing steps like lowercasing the text, removing unwanted elements like stopwords, '\n', Urls, numbers, punctuations etc.
- As the text column is with many miss-spelled words and the problem is multi-labelled so we are getting slightly lower accuracy for this task. However we have selected best model among all the algorithms. There are some comments which are from different language other than English we can try the same approach by removing those comments with other languages.