FLIP ROBO

MALIGNANT COMMENTS CLASSIFIER

SUBMITTED BY:-

RISHABH JOHRI

ACKNOWLEDGEMENT:-

It is a great pleasure to express my gratitude to Team Flip Robo, for giving me the opportunity to work on a interesting project, which helped me in improving my knowledge, coding skills and my analysation skills.

Team Flip Robo also gave me opportunity to build PowerPoint Presentation and Project Report, which will help me to share steps taken while building the entire model. It has helped me in deciding about the future prospects of various Data Science fields. Now, I will explain the understanding of the project through this report.

INTRODUCTION:-

Online forums and social media platforms have provided individuals with the means to put forward their thoughts and freely express their opinion on various issues and incidents. In some cases, these online comments contain explicit language which may hurt the readers. Comments containing explicit language can be classified into some categories such as Toxic, Severe Toxic, Obscene, Threat, Insult, and Identity Hate. The threat of abuse and harassment means that many 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 also.

The main purpose of building this model is to prevent the abusive comment which in turn will Detroit the mindset of an individual or people, now-a-days a lot of abusive and lethargic comment can be seen on various social media platforms which create a negative environment among the people and community, so to stop this type of activity a machine learning model is built to identify the malignant text and filter it out as soon as it encounters it.

Review of Literature:-

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 cyberbullying using various Machine Learning Techniques. In this, we are investigating the application of machine learning techniques and vectorisation concept to predict the comments. The predictions are based on historical data collected from various websites. Here, Different techniques will be used to build a model for predicting the malignant comments.

Motivation for the Problem Undertaken :-

Online platforms when used by normal people can only be comfortably used by them only when they feel that they can express themselves freely and without any reluctance. If they come across any kind of a malignant or toxic type of a reply which can also be a threat or an insult or any kind of harassment which makes them uncomfortable, they might defer to use the social media platform in future.

Thus, it becomes extremely essential for any organization or community to have an automated system which can efficiently identify and keep a track of all such comments and thus take any respective action for it, such as reporting or blocking the same to prevent any such kind of issues in the future.

Analytical Problem Framing:-

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 focus more on the words which has greater value of importance in the context. CommentText is the NLP term used here on text columns for fetching comment data.

DATA SOURCES AND 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**".**

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.

DATA PREPROCESSING:-

Data processing and feature engineering are a crucial part in machine learning to build a prediction model. Furthermore, a model cannot be made without some data processing. For instance, the model could not be trained before handling the missing values and converting the text in the dataset into numerical values. Hence, from the experiment, we saw that pre-processing the data does improve the prediction accuracy.

- **Info( )**: it is used to give info about not null value and datatype of features. here datatype of comments is (object) which is been taken care where as others have int datatype.

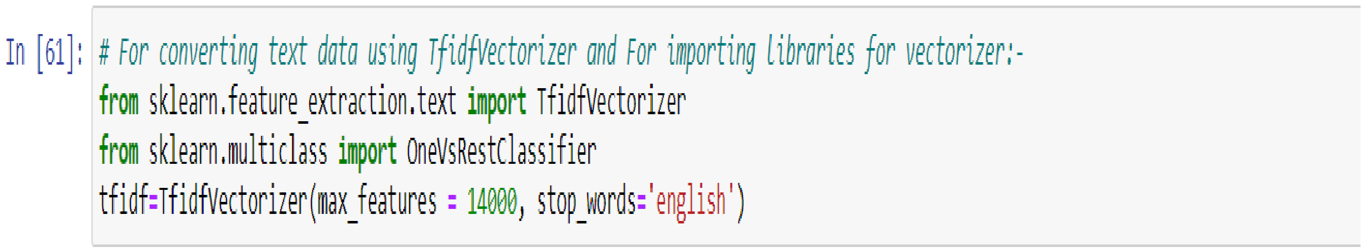
- **Describe( )**: The describe method is used for calculating some statistical data like **percentile, mean** and **std** of the numerical values of the Series or Data Frame. It analyses both numeric and object series and also the Data Frame column sets of mixed data types.it also give info about distribution of data.

- **Lemmatization:** Lemmatization on the surface is very similar to stemming, where the goal is to remove inflections and map a word to its root form. The only difference is that, lemmatization tries to do it the proper way. It doesn’t just chop things off, it actually transforms words to the actual root. For example, the word “better” would map too “good”. It may use a dictionary such as word net for mapping or some special rule-based approaches.

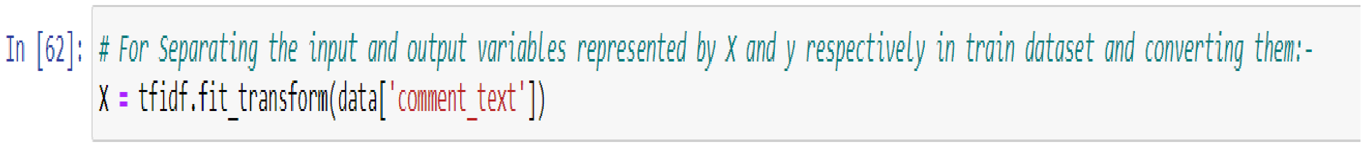
- **Tf-Idf vectorization:** In NLP, tf-idf is an important measure and is used by algorithms like cosine similarity to find documents that are similar to a given search query.

- What is Term Frequency (tf): tf is the number of times a term appears in a particular document. So, it’s specific to a document. A few of the ways to calculate tf is given below: -

tf(t) = No. of times term ‘t’ occurs in a document



```
In [61]: # For converting text data using TfidfVectorizer and For importing libraries for vectorizer:-
         from sklearn.feature_extraction.text import TfidfVectorizer
         from sklearn.multiclass import OneVsRestClassifier
         tfidf=TfidfVectorizer(max_features = 14000, stop_words='english')
```



```
In [62]: # For Separating the input and output variables represented by X and y respectively in train dataset and converting them:-
         X = tfidf.fit_transform(data['comment_text'])
```

Now, comes model building, I have used 4 classifier algorithm () which fits best in this dataset and predict good accuracy. After finding my best model with the help of accuracy and cross validation score, there comes hyper parameter tuning for best fit model.

- **Hardware requirement**: -

Minimum i5 or more.

Python is widely used in scientific and numeric computing.
SciPy is a collection of packages for mathematics, science, and engineering.

Pandas is a data analysis and modelling library.

Modules or the libraries required for project data analysis and visualization:

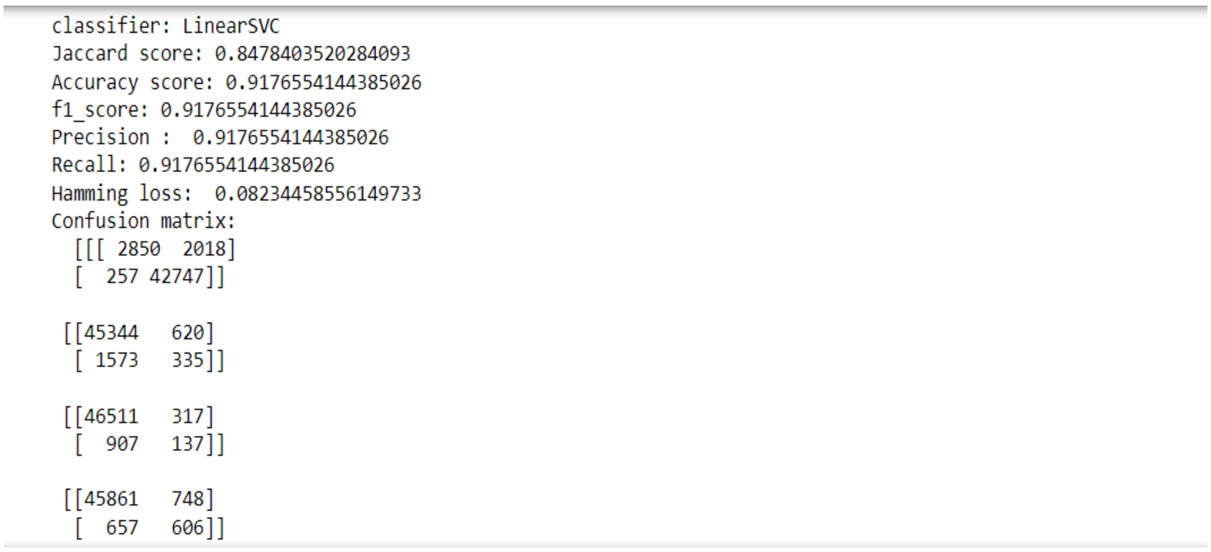
- **Pandas** for data analysis and import
- **NumPy** to perform Mathematical operation.
- **Seaborn** and **matplotlib** for data visualization.
- **Scikit-learn:** All the models, metrics and feature selection etc. are present inside of that module. We have imported all from this library according to our need.

- **Train-Test split**: There are two primary phases in the system:

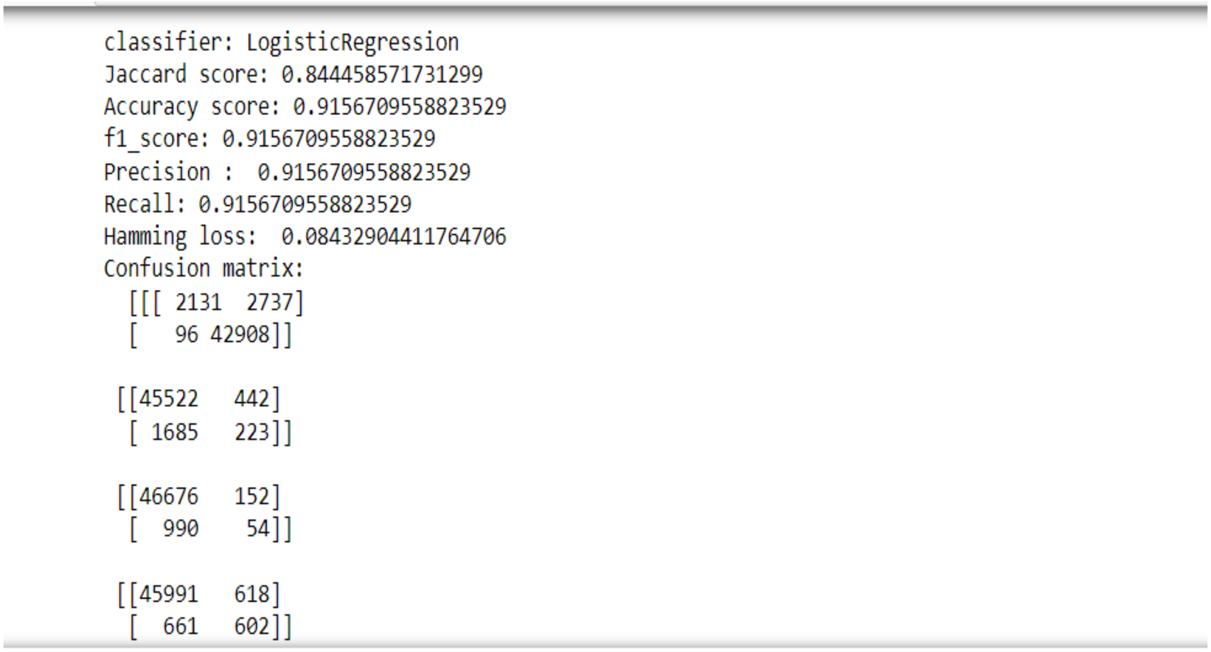
1. **Training phase:** The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly.

2. **Testing phase:** The system is provided with the testing data, and it is tested for its working. The accuracy is checked. And therefore, the data that is used to train the model or test it, must be appropriate.

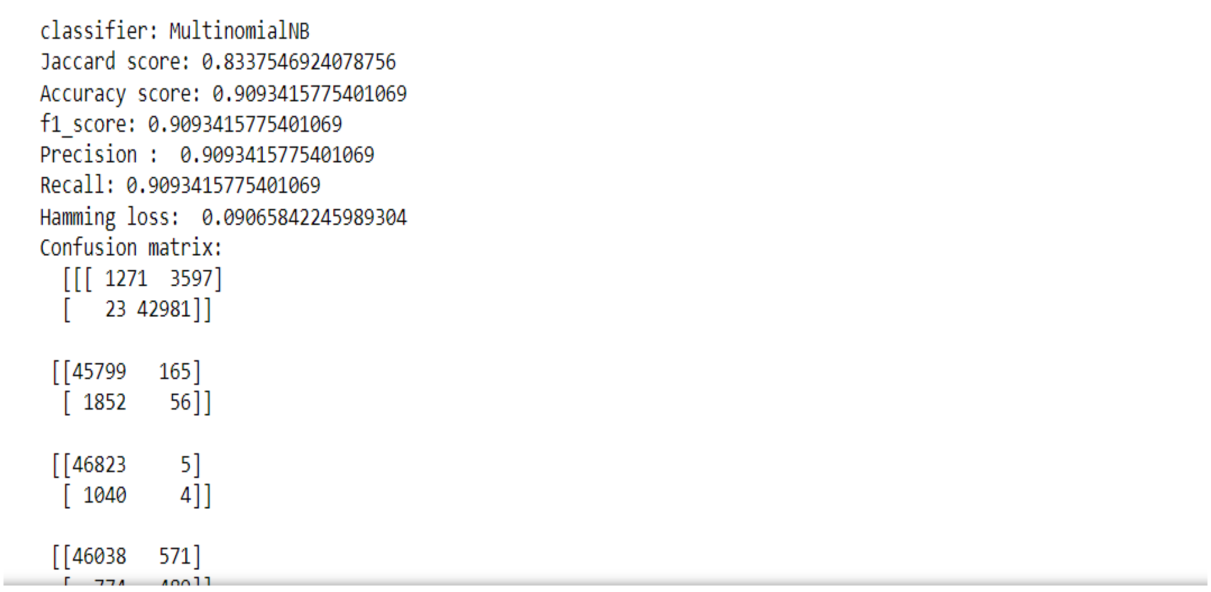
MODEL BUILDING:-



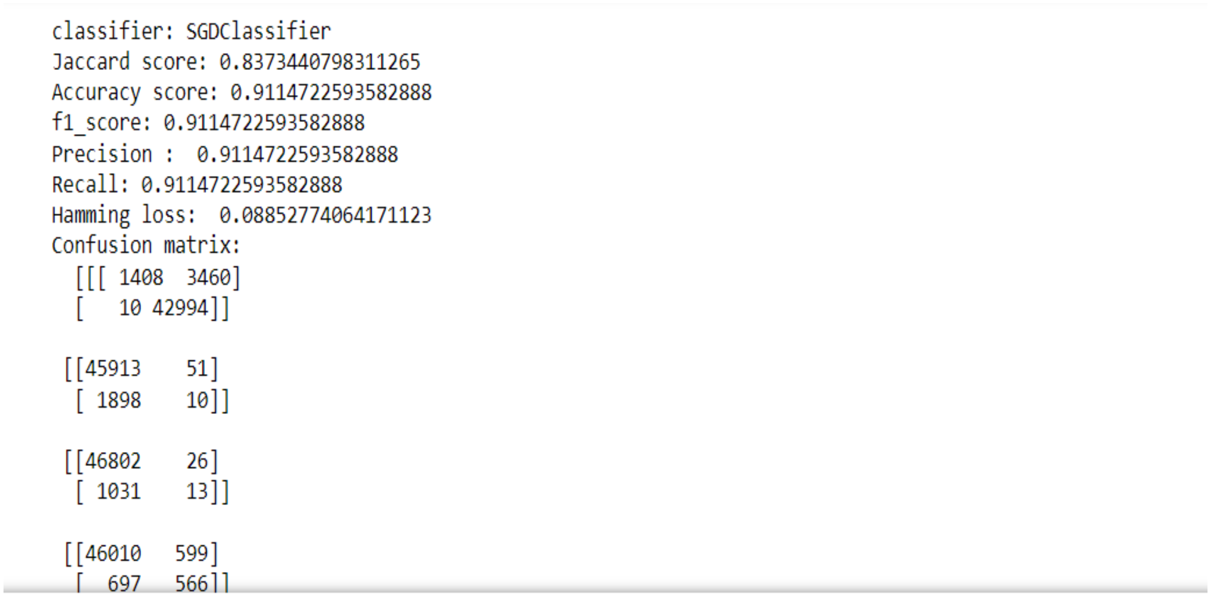
```
classifier: LinearSVC
Jaccard score: 0.8478403520284093
Accuracy score: 0.9176554144385026
f1_score: 0.9176554144385026
Precision :  0.9176554144385026
Recall: 0.9176554144385026
Hamming loss:  0.08234458556149733
Confusion matrix:
  [[[ 2850  2018]
  [  257 42747]]

 [[45344   620]
  [ 1573   335]]

 [[46511   317]
  [  907   137]]

 [[45861   748]
  [  657   606]]
```



```
classifier: LogisticRegression
Jaccard score: 0.844458571731299
Accuracy score: 0.9156709558823529
f1_score: 0.9156709558823529
Precision :  0.9156709558823529
Recall: 0.9156709558823529
Hamming loss:  0.08432904411764706
Confusion matrix:
  [[[ 2131  2737]
  [   96 42908]]

 [[45522   442]
  [ 1685   223]]

 [[46676   152]
  [  990    54]]

 [[45991   618]
  [  661   602]]
```

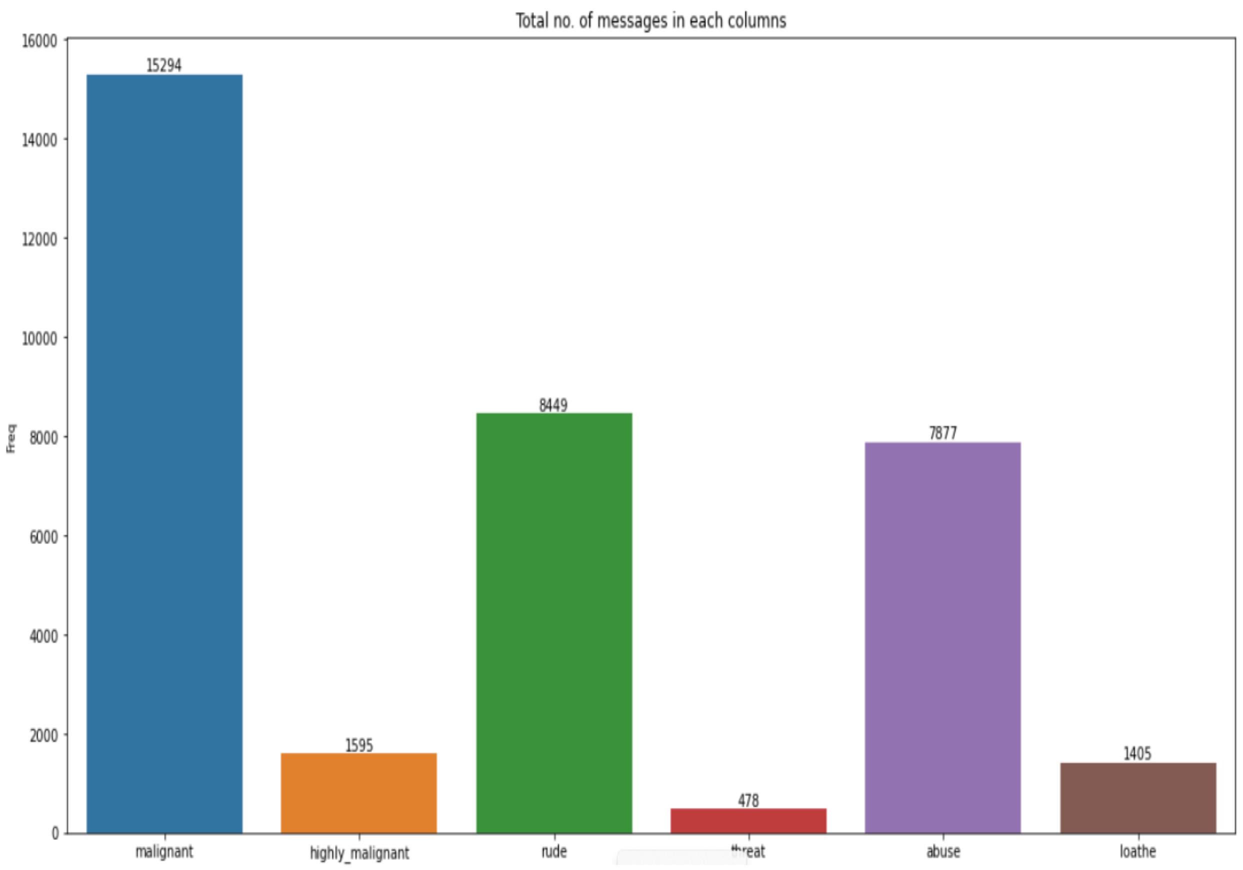


```
classifier: MultinomialNB
Jaccard score: 0.8337546924078756
Accuracy score: 0.9093415775401069
f1_score: 0.9093415775401069
Precision :  0.9093415775401069
Recall: 0.9093415775401069
Hamming loss:  0.09065842245989304
Confusion matrix:
  [[[ 1271  3597]
  [   23 42981]]

 [[45799   165]
  [ 1852    56]]

 [[46823     5]
  [ 1040     4]]

 [[46038   571]
  [  774   489]]
```



```
classifier: SGDClassifier
Jaccard score: 0.8373440798311265
Accuracy score: 0.9114722593582888
f1_score: 0.9114722593582888
Precision :  0.9114722593582888
Recall: 0.9114722593582888
Hamming loss:  0.08852774064171123
Confusion matrix:
  [[[ 1408  3460]
  [   10 42994]]

 [[45913    51]
  [ 1898    10]]

 [[46802    26]
  [ 1031    13]]

 [[46010   599]
  [  697   566]]
```

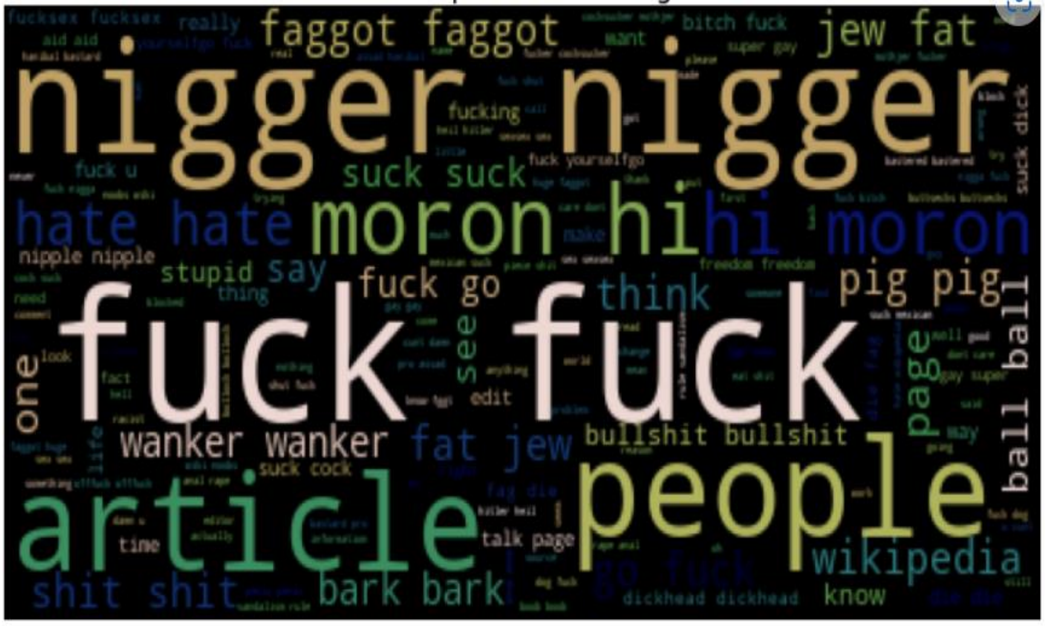
DATA VISUALISATIONS:-



WordCloud:-

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

Words frequented in malignant



Hyper Parameter Tuning:-

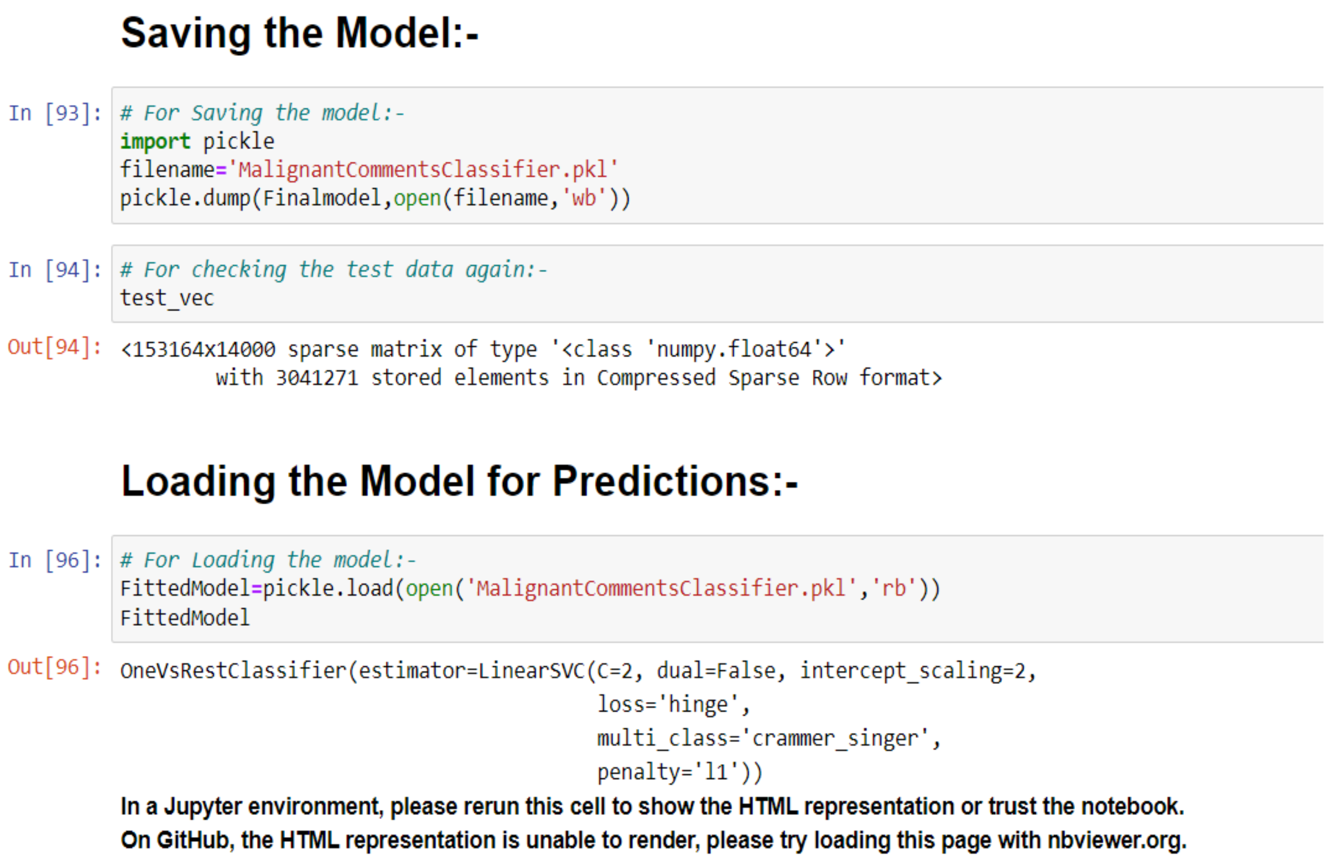
```
In [88]: Finalmodel = OneVsRestClassifier(LinearSVC(C=2,dual = False, loss='hinge',multi_class='crammer_singer', penalty ='l1',intercept_s
         Finalmodel.fit(x_train,y_train)
         y_pred = Finalmodel.predict(x_test)

         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: 0.8479830148619958
Accuracy score: 0.9177389705882353
f1_score: 0.9177389705882353
Precision :  0.9177389705882353
Recall: 0.9177389705882353
Hamming loss:  0.0822610294117647

Confusion matrix:
 [[[ 2888  1980]
  [  279 42725]]

 [[45465   499]
  [ 1633   275]]

 [[46575   253]
  [  904   140]]

 [[45765   844]
  [  598   665]]

 [[47065   279]
  [  411   117]]

 [[47684    72]
```

Many machine learning algorithms are used to predict. However, the prediction accuracy of these algorithms depends heavily on the given data when training the model. If the data is in bad shape, the model will be overfitted,which means that data pre-processing is an important part of this experiment and will affect the final results.

Thus, multiple combinations of pre-processing methods which needs to be tested before getting the data ready to be used in training. After analyzing every model, Linear SVC shows the best accuracy and Cross validation score with least difference between the two and While doing Hyper parameter tuning its accuracy reaches to 91.77%.

SAVING THE MODEL AND LOADING PREDICTIONS:-



Saving the Model:-

```
In [93]: # For Saving the model:-
         import pickle
         filename='MalignantCommentsClassifier.pkl'
         pickle.dump(Finalmodel,open(filename,'wb'))
```

```
In [94]: # For checking the test data again:-
         test_vec

Out[94]: <153164x14000 sparse matrix of type '<class 'numpy.float64'>'
                 with 3041271 stored elements in Compressed Sparse Row format>
```

Loading the Model for Predictions:-

```
In [96]: # For Loading the model:-
         FittedModel=pickle.load(open('MalignantCommentsClassifier.pkl','rb'))
         FittedModel

Out[96]: OneVsRestClassifier(estimator=LinearSVC(C=2, dual=False, intercept_scaling=2,
                                                 loss='hinge',
                                                 multi_class='crammer_singer',
                                                 penalty='l1'))
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

CONCLUSION:-

Communication is one of the basic necessities of everyone’s life. People need to talk and interact with one another to express what they think. Over the years, social media and social networking have been increasing exponentially due to a rise in the use of the internet. Flood of information arises from online conversation on a daily basis, as people are able to discuss, express themselves and express their opinion via these platforms. While this situation is highly productive and could contribute significantly to the quality of human life, it could also be destructive and enormously dangerous. The responsibility lies on the social media administration, or the host of organization to control and monitor these comments.

This research work focuses on developing a model that would automatically classify a comment as either malignant or non-malignant using Linear SVC.

Therefore, this study aims to develop a multi-headed model to detect different types of malignant comment like threats, rude, abusive, and loathe. By collecting and preprocessing malignant comments for training and testing using term frequency- inverse document frequency (TF-IDF) algorithm, developing a multi-headed model will detect different types of malignant comment using machine learning algorithms to train the dataset, and evaluate the model using confusion matrix.

SCOPE FOR FUTURE WORK:-

In future, this machine learning model may bind with various website which can provide real time data for price prediction. We can build an android app as user interface for interacting with user. For better performance, we plan to judiciously design Deep Learning Network Structures, use adaptive learning rates and train on clusters of data rather than the whole dataset.