FLIP ROBO

MALIGNANT COMMENT CLASSIFICATION

Submitted by:

Rijul Kumar

ACKNOWLEDGMENT

Reference that i have used are:

- Data Trained Education online video
- Materials provided by Flip Robo
- Geeks for Geeks
- Stackoverflow

INTRODUCTION

- ## Business Problem Framing

  1. 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.
  2. 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.
  3. 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

  The classification of the hate and abusive comments can be done by many methods. Such as:

  - Number of characters in a comment
  - By using hateful words which are repeating in past comments

- And so on…

- ## Motivation for the Problem Undertaken

  - We have to build a model which can be used to identify as well as classify hateful comments. This will be a multi label binary classifier model.

  - Then with help of this model people can filter out the hate comments leading to a healthier mental state (i.e lack of depression, mental illness, self-hatred and suicidal thoughts).

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem
    - First of all i imported data from train and test csv files to DataFrames using pandas.
    - After that i used .describe() to know the statistical information (such as max, min value,etc) of train and test dataframe.
    - Then i used .shape to know shape of train and test dataframe.
    - Afterwards i used .dtypes to know data type of each column of train and test dataframe.
    - Next i moved on to data preprocessing part.

- Data Sources and their formats
    - Problem Statement.docx provided by FlipRobo
    - test.csv and train.csv provided by FlipRobo
    - Data_Description.xlsx provided by FlipRobo

- <u>Data Preprocessing Done</u>

  - First of all for data preprocessing i checked whether there is a NULL value or not in train and test dataframe using heatmap as well as .isnull()
  - After that i used count plots from seaborn library to plot all categorical columns for visualisation.
  - After that i plotted many graphs related to number of characters in comments and different hate comment type.
  - Then i added a ‘comment_size’ column to train data as well as test data.
  - Then i did encoded the dataframe using Ordinal Encoder.
  - After that i checked for correlations using heatmaps, correlation matrix and BAR plot.
  - Finally i confirmed that there were no high correlations using VIF.
  - After that i checked skewness for continuous data in both train and test dataframe.
  - Afterwards i checked for outliers for continuous data in both train and test dataframe using boxplot as well as z-score method.

- ## Data Inputs- Logic- Output Relationships

  - Data Input :

    These are basically the comments which are to be checked and classified under different categories of hated comments.

  - Data Output :

    As this is a Multi label binary classification problem so output are the multiple target variables which are in this problem different categories of hated comments.

- ## Hardware and Software Requirements and Tools Used

  - Hardware used:

    i. Laptop with intel core i5 7th gen

  - Software used:

    i. Jupyter notebook

    ii. Required python libraries such as numpy, pandas, seaborn, matplotlib, etc

    iii. Required libraries for model such as sklearn, etc

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
  - After preprocessing data (checking NULL, encoding, removing high correlations, removing skewness and outliers) i separated columns into features and different target.
  - As this is a multi label binary classification problem so we tried 4 models - LogisticRegression, DecisionTreeClassifier, RandomForestClassifier and Gaussian Naive Bayes.
  - We checked for each model analytically (classification report and confusion matrix) as well as graphically (AUC-ROC curves).
  - Finally i used Ensemble Technique.

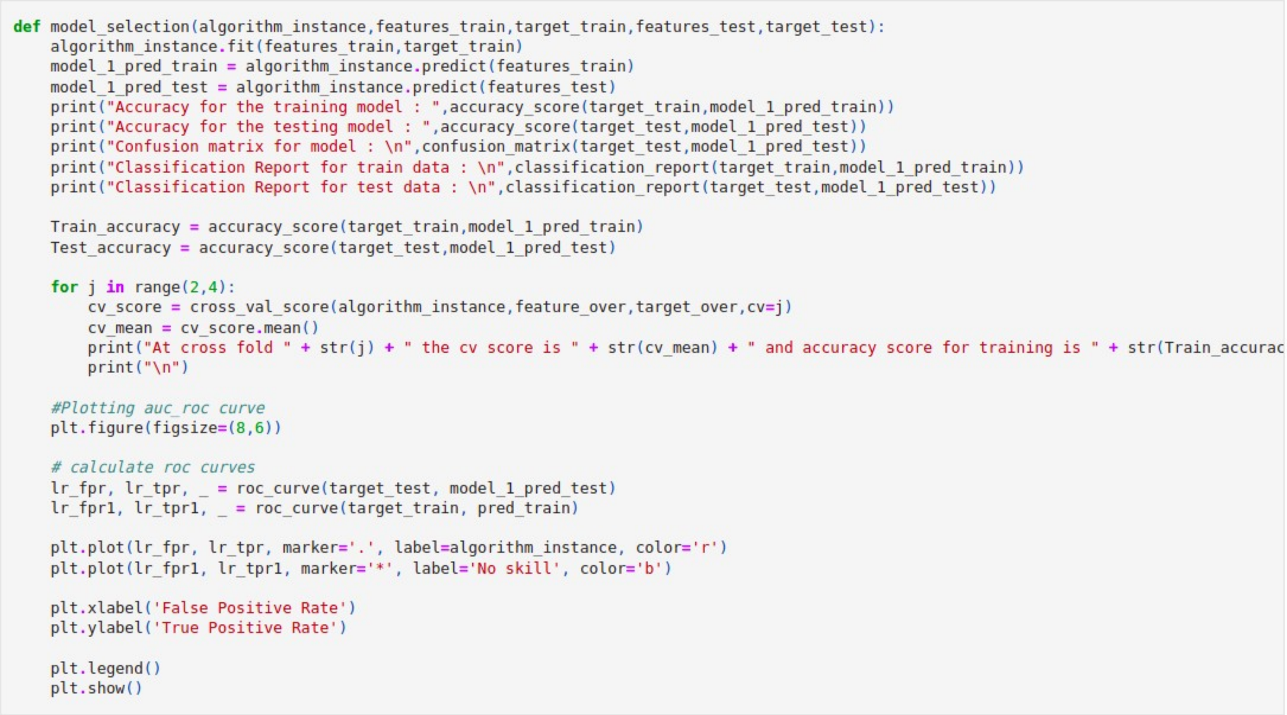
- Testing of Identified Approaches (Algorithms)

  As this is a regression problem so we tried following 4 models -

  - LogisticRegression
  - DecisionTreeClassifier
  - RandomForestClassifier
  - Gaussian Naive Bayes

- ## Run and Evaluate selected models

  I defined a function model and then tried 4 different models using it



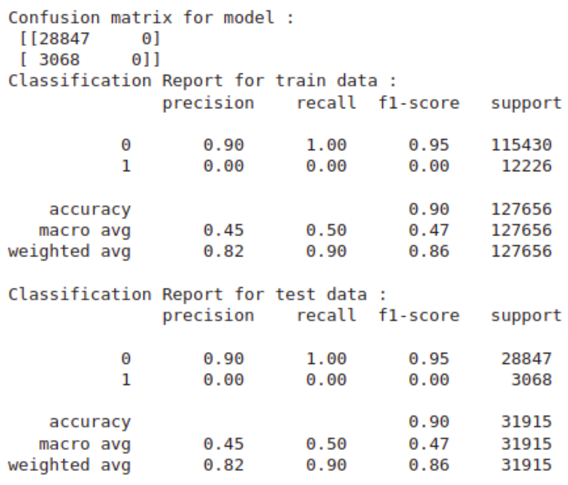
```
def model_selection(algorithm_instance,features_train,target_train,features_test,target_test):
    algorithm_instance.fit(features_train,target_train)
    model_1_pred_train = algorithm_instance.predict(features_train)
    model_1_pred_test = algorithm_instance.predict(features_test)
    print("Accuracy for the training model : ",accuracy_score(target_train,model_1_pred_train))
    print("Accuracy for the testing model : ",accuracy_score(target_test,model_1_pred_test))
    print("Confusion matrix for model : \n",confusion_matrix(target_test,model_1_pred_test))
    print("Classification Report for train data : \n",classification_report(target_train,model_1_pred_train))
    print("Classification Report for test data : \n",classification_report(target_test,model_1_pred_test))

    Train_accuracy = accuracy_score(target_train,model_1_pred_train)
    Test_accuracy = accuracy_score(target_test,model_1_pred_test)

    for j in range(2,4):
        cv_score = cross_val_score(algorithm_instance,feature_over,target_over,cv=j)
        cv_mean = cv_score.mean()
        print("At cross fold " + str(j) + " the cv score is " + str(cv_mean) + " and accuracy score for training is " + str(Train_accurac
        print("\n")

    #Plotting auc_roc curve
    plt.figure(figsize=(8,6))

    # calculate roc curves
    lr_fpr, lr_tpr, _ = roc_curve(target_test, model_1_pred_test)
    lr_fpr1, lr_tpr1, _ = roc_curve(target_train, pred_train)

    plt.plot(lr_fpr, lr_tpr, marker='.', label=algorithm_instance, color='r')
    plt.plot(lr_fpr1, lr_tpr1, marker='*', label='No skill', color='b')

    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')

    plt.legend()
    plt.show()
```

For target variable ‘malignant’

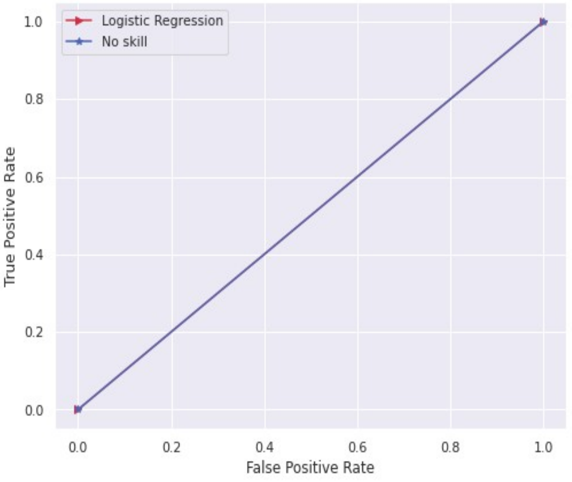
Result that i got for each model :

- LogisticRegression :
  At random state 4 the training accuracy is : 0.9042269850222473
  At random state 4 the testing accuracy is : 0.9038696537678208

  At cross fold 2 the cv score is 0.9041555169761383 and accuracy score for training is 0.9042269850222473 and accuracy score for testing is 0.9038696537678208

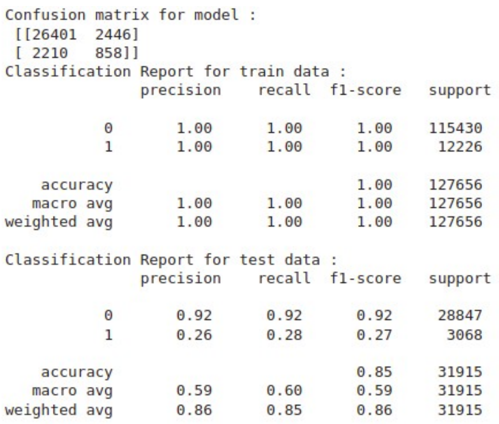


```
Confusion matrix for model :
 [[28847     0]
 [ 3068     0]]
Classification Report for train data :
              precision    recall  f1-score   support

           0       0.90      1.00      0.95    115430
           1       0.00      0.00      0.00     12226

    accuracy                           0.90    127656
   macro avg       0.45      0.50      0.47    127656
weighted avg       0.82      0.90      0.86    127656

Classification Report for test data :
              precision    recall  f1-score   support

           0       0.90      1.00      0.95     28847
           1       0.00      0.00      0.00      3068

    accuracy                           0.90     31915
   macro avg       0.45      0.50      0.47     31915
weighted avg       0.82      0.90      0.86     31915
```

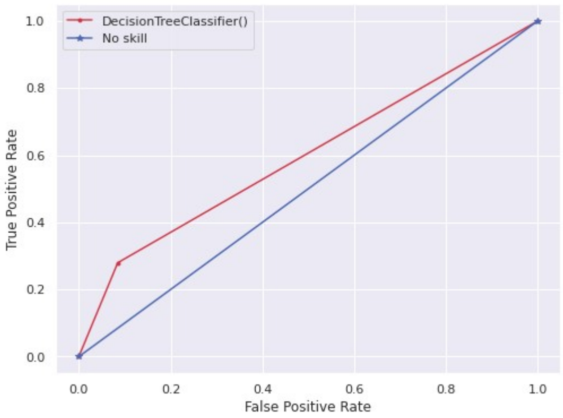


- DecisionTreeClassifier :
  Accuracy for the training model : 1.0
  Accuracy for the testing model :
  0.8541124862917123

  At cross fold 2 the cv score is 0.852492000123015 and accuracy score for training is 1.0 and accuracy score for testing is 0.8541124862917123

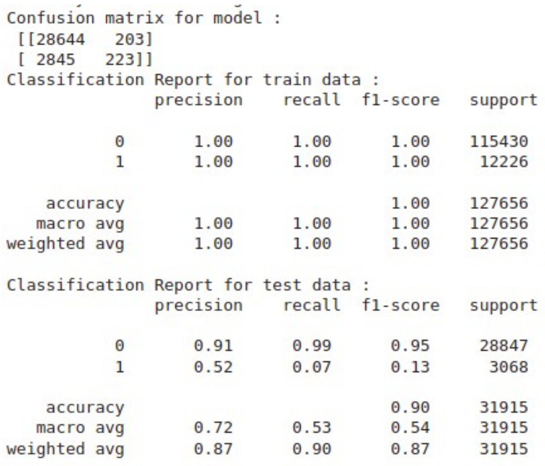


```
Confusion matrix for model :
 [[26401  2446]
 [ 2210   858]]
Classification Report for train data :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    115430
           1       1.00      1.00      1.00     12226

    accuracy                           1.00    127656
   macro avg       1.00      1.00      1.00    127656
weighted avg       1.00      1.00      1.00    127656

Classification Report for test data :
              precision    recall  f1-score   support

           0       0.92      0.92      0.92     28847
           1       0.26      0.28      0.27      3068

    accuracy                           0.85     31915
   macro avg       0.59      0.60      0.59     31915
weighted avg       0.86      0.85      0.86     31915
```

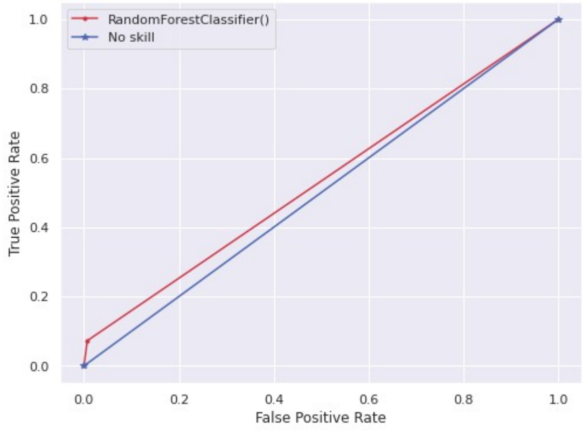


- RandomForestClassifier :
  Accuracy for the training model :
  0.9998668296045623
  Accuracy for the testing model :
  0.9044963183456055

At cross fold 2 the cv score is 0.9044876576458245 and accuracy score for training is 0.9998668296045623 and accuracy score for testing is 0.9044963183456055

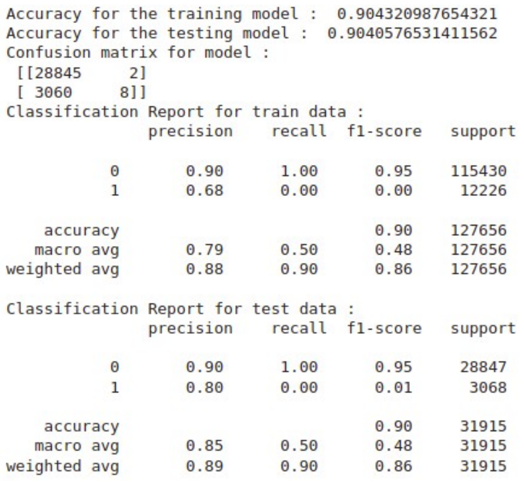


```
Confusion matrix for model :
 [[28644   203]
 [ 2845   223]]
Classification Report for train data :
               precision    recall  f1-score   support

           0       1.00      1.00      1.00    115430
           1       1.00      1.00      1.00     12226

    accuracy                           1.00    127656
   macro avg       1.00      1.00      1.00    127656
weighted avg       1.00      1.00      1.00    127656

Classification Report for test data :
               precision    recall  f1-score   support

           0       0.91      0.99      0.95     28847
           1       0.52      0.07      0.13      3068

    accuracy                           0.90     31915
   macro avg       0.72      0.53      0.54     31915
weighted avg       0.87      0.90      0.87     31915
```

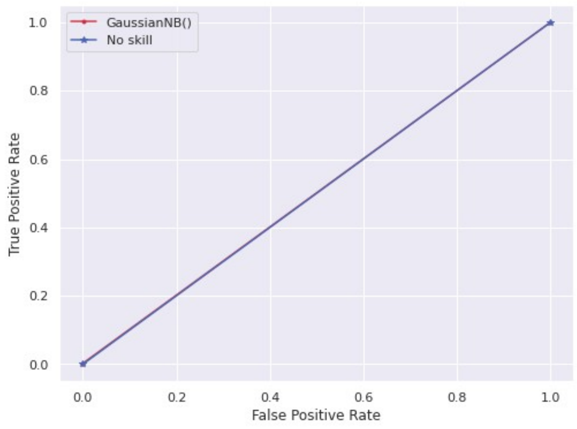


- <u>GaussianNB</u> :
  Accuracy for the training model : 0.904320987654321
  Accuracy for the testing model : 0.9040576531411562

  At cross fold 2 the cv score is 0.9026013401258531 and accuracy score for training is 0.904320987654321 and accuracy score for testing is 0.9040576531411562



```
Accuracy for the training model :  0.904320987654321
Accuracy for the testing model :  0.9040576531411562
Confusion matrix for model :
 [[28845     2]
 [ 3060     8]]
Classification Report for train data :
               precision    recall  f1-score   support

           0       0.90      1.00      0.95    115430
           1       0.68      0.00      0.00     12226

    accuracy                           0.90    127656
   macro avg       0.79      0.50      0.48    127656
weighted avg       0.88      0.90      0.86    127656

Classification Report for test data :
               precision    recall  f1-score   support

           0       0.90      1.00      0.95     28847
           1       0.80      0.00      0.01      3068

    accuracy                           0.90     31915
   macro avg       0.85      0.50      0.48     31915
weighted avg       0.89      0.90      0.86     31915
```



Finally i concluded that DecisionTreeClassifier() gives best accuracy which can be seen from auc-roc curve as well as from classification report.

Hence i took DecisionTreeClassifier() as main model

For target variable ‘high_malignant’

- DecisionTreeClassifier :

  Accuracy for the training model : 1.0

  Accuracy for the testing model : 0.9827667241109196

  At cross fold 2 the cv score is 0.98312976770736 and accuracy score for training is 1.0 and accuracy score for testing is 0.9827667241109196

For target variable ‘rude’

- DecisionTreeClassifier :

  Accuracy for the training model : 1.0

  Accuracy for the testing model : 0.9130816230612565

  At cross fold 2 the cv score is 0.9092316120592818 and accuracy score for training is 1.0 and accuracy score for testing is 0.9130816230612565

For target variable ‘threat’

- DecisionTreeClassifier :

  Accuracy for the training model : 1.0

  Accuracy for the testing model : 0.9948613504621652

At cross fold 2 the cv score is 0.9945604156626277 and accuracy score for training is 1.0 and accuracy score for testing is 0.9948613504621652

For target variable ‘abuse’

- DecisionTreeClassifier :

  Accuracy for the training model : 1.0

  Accuracy for the testing model : 0.9189409368635438

  At cross fold 2 the cv score is 0.9165324437989752 and accuracy score for training is 1.0 and accuracy score for testing is 0.9189409368635438

For target variable ‘loathe’

- DecisionTreeClassifier :

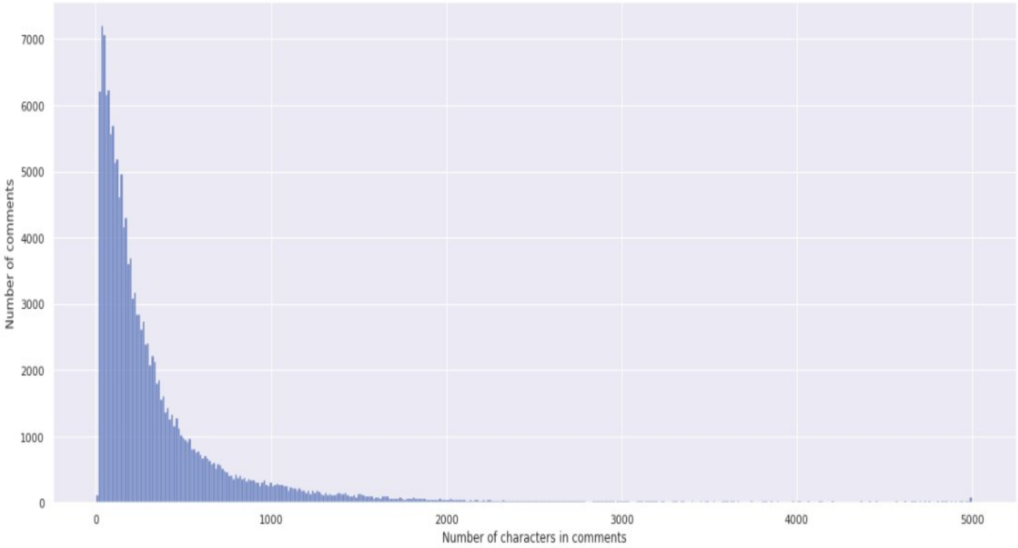
  Accuracy for the training model : 1.0

  Accuracy for the testing model : 0.9827040576531412

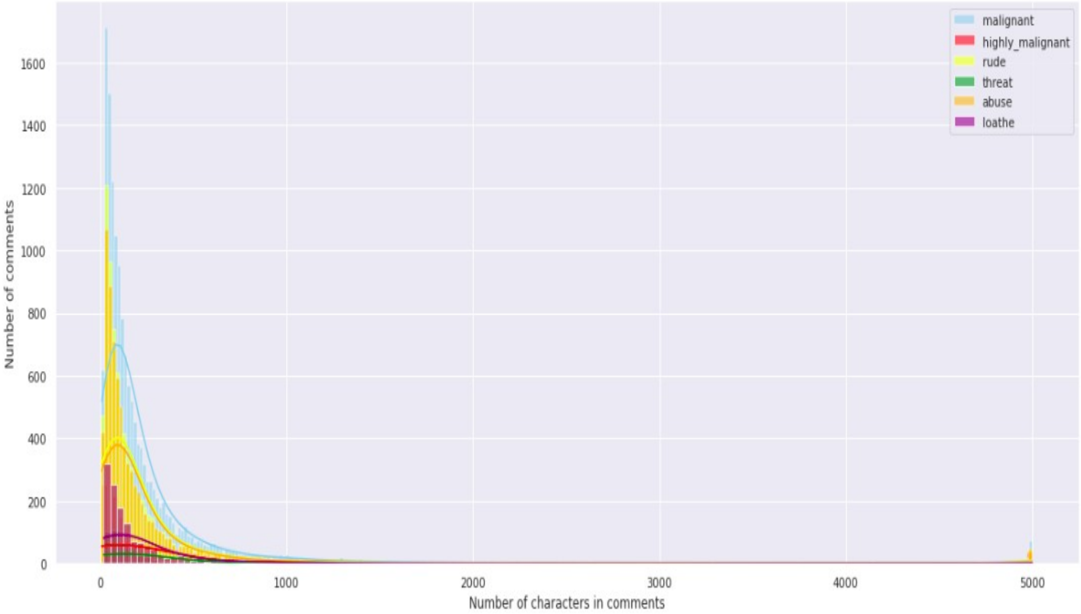
  At cross fold 2 the cv score is 0.8214264433131013 and accuracy score for training is 1.0 and accuracy score for testing is 0.9827040576531412

- ## Visualizations

This graph represents number of comments vs comment size



This graph represents comparison for all target variables of number of comments vs comment size



- ## Interpretation of the Results

  So the results which i got were:

  - DecisionTreeClassifier() gives best accuracy which can be seen from auc-roc curve as well as from classification report.
  - We got our final accuracy as 90.41% for target variable (‘malignant’) after hypertuning.

CONCLUSION

- ## Key Findings and Conclusions of the Study
    - All the target variables are mutually exclusive i.e. we can ignore any correlation between them.
    - We can identify as well as categorise the type of hated comments based on some factors such as number of character in a comment.

- ## Limitations of this work and Scope for Future Work

  Some limitations are :

    - There are many other factors (such as topic on which the person has commented,etc) which are not in the data which may play major role in predicting and classifying the type of hated comments.
    - Unrelated factors (such as person who is commenting might be under stress in daily life,etc) also plays minor role in affecting our target variable.

  Scope for future work :

    - This can be made further accurate by taking more and more factors into account.