FLIP ROBO

MALIGNANT COMMENTS CLASSIFIER

Submitted by:

ASHUTOSH MISHRA

ACKNOWLEDGMENT

I'd like to extend my gratitude to my mentor Mr. Md. Kashif for giving me this opportunity to work upon this project. Below are all the details of the project which I consumed while preparing and drafting the project –

The references, research papers, data sources, professionals, etc are majorly referred from "Flip Robo Technologies" study collaterals and data repository. Also, some of the other resources like "Research Gate" were explored to gain a deep understanding on the assigned subject.

Above stated details stands correct to the best of my knowledge and I hereby acknowledge the same.

INTRODUCTION

See, this is the era of digitalization and almost each of the individuals across the world are connected by some common social media platforms e.g. Twitter, Facebook, Instagram, You-tube and etcetera. As per the some of the websites 'More than 300 million photos get uploaded per day and every minute there are 510,000 comments posted and 293,000 statuses updated' and this is literally a huge number in terms of data and posts.

If we talk about the type of comments that people use to post in large numbers are generally malignant comments, as we're aware that good comments are rare now a days whereas these negative comments are very often as they are abusive, loathe, threatening and rude in nature, and these type of comments usually spread intentionally by the end-users in-order to effect the mental health, fame, prestige of the person and even such malignant comments can lead to a person to commit suicide.

We can term the word 'Malignant Comments' as the online disease as this can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts. So in my opinion 'Machine Learning' could be the instant resolution to identify such comments and can let the user to see what they actually want to read or see.

Business Problem Framing

According to the some researchers, online hate is a big problem across multiple social media platforms but still there is a lack of models for online hate detection.

Having said that malignant comments can consists of different type of comments like- rude, abusive, threatening and loathe; so it’s really important to identify the comments type so that user can let themselves aware about the same.

See, the demand to build such models is increasing rapidly as now a day’s most of the celebrities are facing a lot of offensive, abusive and hateful comments on their social media platforms and it is impacting their large fan base, reputation and glory in the respective industry and to tackle the same they want to be proactive by identifying such highly malignant utterances and in-order to do so machine learning could be that uprising thing to predict and classify these comments.

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.

Conceptual Background of the Domain Problem

Data science and Machine Learning comes as a vital tool to solve almost any type of business problems to help companies optimize the standards resulting in overall revenue and profits growth. Moreover, it also improves their marketing strategies and demands focus on changing trends.

Now, if we talk about the Malignant Comment Classifier project what I feel is that Classification models like Naive-Bayes, Decision Tree Classifier, Random Forest Classifier and Gradient Boosting Classifier and etcetera would be the perfect algorithms in identify these malignant comment types.

Also, implementation of Hyper Parameter Tuning would be really interesting to see the best parameters while predicting the target variable Malignant.

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. 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.

Being a Data Scientist, I have used the Classification Model using multiple algorithms to design and optimize the results as the target variable is in the form of categorical label.

Some of the classes which I explored from Scikit-learn Libraries were – Statistics, Analytical Modelling, Hyper Tuning Method, CV Ratings, Predictive Modelling and Classification Model, etc.

In totality, my project comprises of six different test cases of Classification models where the objective was to train the model with the help of train dataset and later test for accuracy score and different classification reports using ROC_AUC score, F1 score and ROC_AUC curve to understand the accuracy of the different algorithms.

Initially two dataset were provided to me; one was train dataset for model building and other was test dataset for predicting the target variable ‘malignant’. Train dataset consists of 159571 rows and 8 different attributes name as- id, comment_text, malignant, highly_malignant, rude, abuse, loathe and threat whereas the test dataset consist of 153164 rows and only 2 columns name as- id and comment_text.

Also, I have observed that there are 6 numeric columns whereas 2 categorical columns are there in the train dataset; there are no null values are present in both dataset.

As a result of the above analytical modelling, I have managed to achieve below top model, which are given below as following–

Classification Models	Testing Accuracy (in %)
Gaussian NB	80
Gradient Boosting Classifier	79
K Neighbors Classifier	73
Random Forest Classifier	62
Decision Tree Classifier	61
Ada Boost Classifier	52

Motivation for the Problem Undertaken

See, we can’t let someone stop to share their thoughts or opinion over any media platform or in general but we can easily block or restrict those toxic comments, with the help of machine learning models, which are hurtful or offensive in nature.

Genuinely speaking, abusive or offensive comments not only defame an individual characteristic but can also affect the prestige of the Country too.

In year 2021, most of the formers from different states of India came in large number to protest against the government for the revocation of Farm Laws and most of the online haters had used this protest as an opportunity to defame the dignity of our country and more importantly they had showcase the false image of it to the rest of world and that is what we need to tackle.

Hence by building a blueprint of online hate and abuse comment classifier such toxic comments Can be controlled and restricted from spreading hatred and cyberbullying.

Also, this project guided me to baseline each aspect carefully and be concrete on the decision making process regardless it’s an individual or an entity like a company.

In order to cater to the above project, my current knowledge and skill set has aided me a lot which I’d explore on this exponentially further.

Analytical Problem Framing

Mathematical Modeling- I have used the below statistical models to find-out the corresponding mean, median, mode and relationship among the variables.

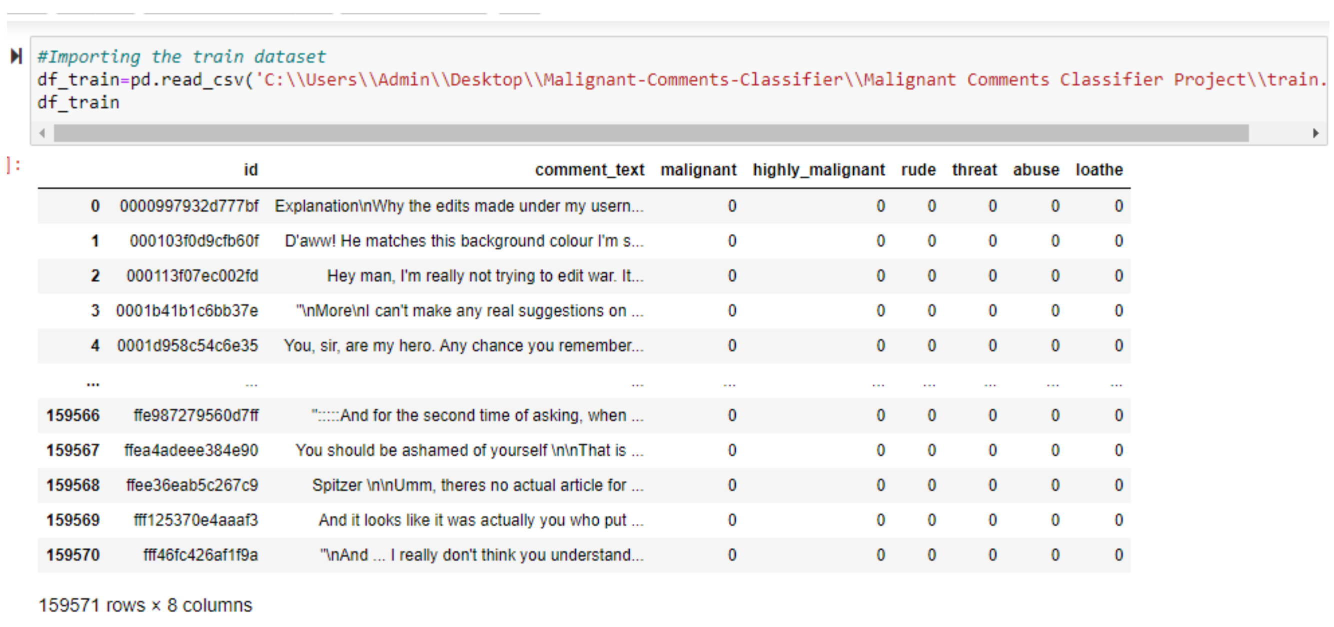
- Descriptive Statistics- To find out the mean, median, mode, percentiles and IQR (**Interquartile Range**)
- Statistical Modeling, Correlation- To find out the relationship among the variables i.e. finding out the positive, negative and zero correlated attributes.
- Multicollinearity- To find out all those variables who are giving similar information to the target variable (‘**malignant**’)
- Skewness- To check whether the attributes are the skewing left hand side or right hand side (Threshold value is=0.5).
- Outliers- To find out variables those are having the value greater than 3.

Analytical Modeling-

- Label Encoder- To convert all the categorical columns into numeric categories
- Simple Imputer-To replace all the null values present in the columns with mean or mode.
- Principle Component Analysis- To remove the curse of dimensionality
- Hyper Parameter Tuning- To find out the best parameter

Data Sources and their formats

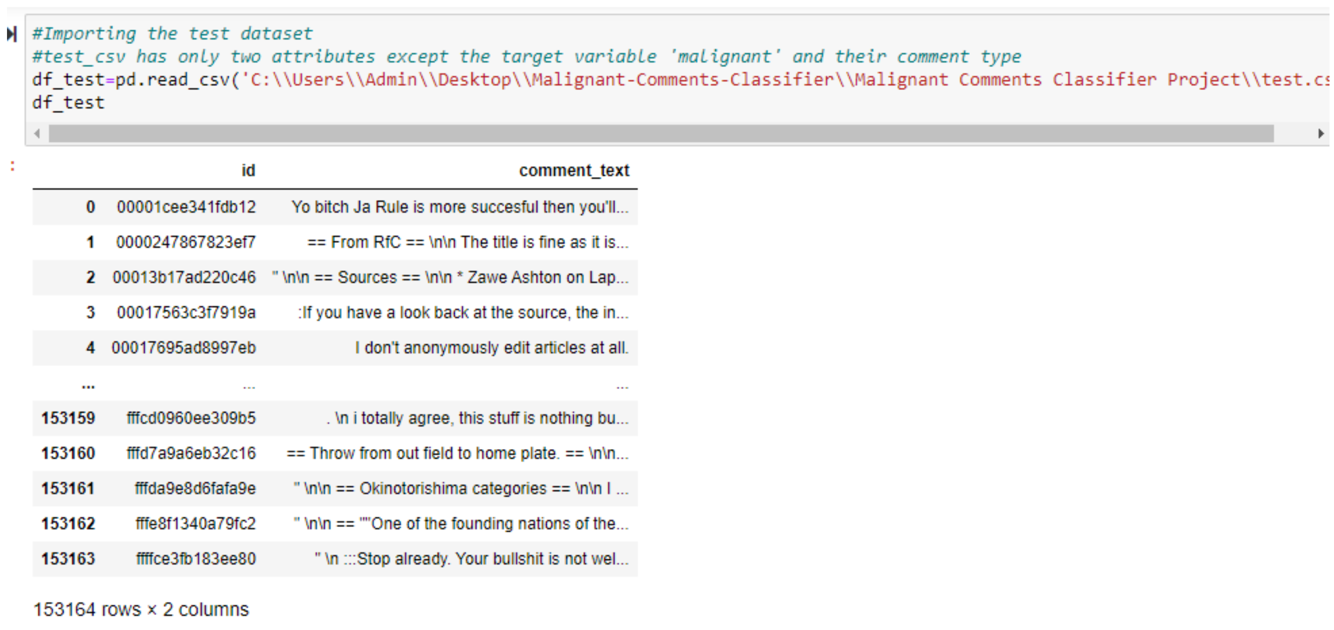
I’ve been provided with two CSV datasets one is train dataset and other is test dataset. Train dataset is given to build the model whereas the test dataset is given to predict the target variable. Let’s have a look on these data sources and their attributes.



```
#Importing the train dataset
df_train=pd.read_csv('C:\\Users\\Admin\\Desktop\\Malignant-Comments-Classifier\\Malignant Comments Classifier Project\\train.
df_train
```

	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
...	...	...	...	...	...	...	...	...
159566	ffe987279560d7ff	":::::And for the second time of asking, when ...	0	0	0	0	0	0
159567	ffea4adeee384e90	You should be ashamed of yourself \n\nThat is ...	0	0	0	0	0	0
159568	ffee36eab5c267c9	Spitzer \n\nUmm, theres no actual article for ...	0	0	0	0	0	0
159569	fff125370e4aaaf3	And it looks like it was actually you who put ...	0	0	0	0	0	0
159570	fff46fc426af1f9a	"\nAnd ... I really don't think you understand...	0	0	0	0	0	0

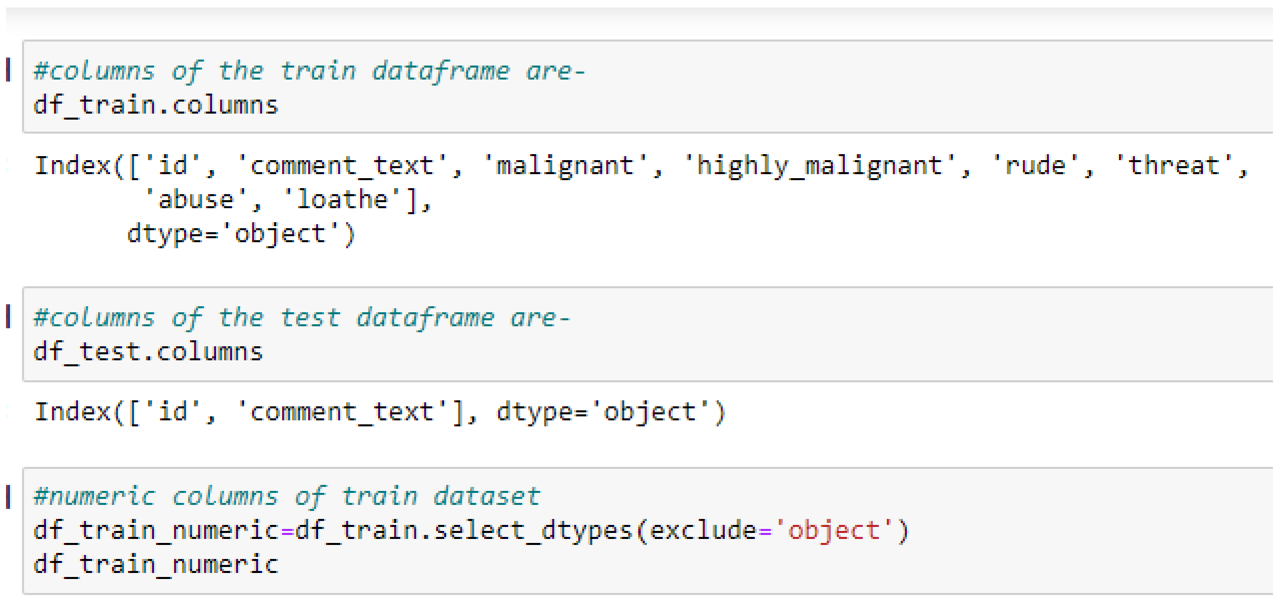
159571 rows × 8 columns



```
#Importing the test dataset
#test_csv has only two attributes except the target variable 'malignant' and their comment type
df_test=pd.read_csv('C:\\Users\\Admin\\Desktop\\Malignant-Comments-Classifier\\Malignant Comments Classifier Project\\test.cs
df_test
```

	id	comment_text
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...
3	00017563c3f7919a	:If you have a look back at the source, the in...
4	00017695ad8997eb	I don't anonymously edit articles at all.
...	...	...
153159	fffcd0960ee309b5	. \n i totally agree, this stuff is nothing bu...
153160	fffd7a9a6eb32c16	== Throw from out field to home plate. == \n\n...
153161	fffda9e8d6fafa9e	" \n\n == Okinotorishima categories == \n\n I ...
153162	fffe8f1340a79fc2	" \n\n == ""One of the founding nations of the...
153163	ffffce3fb183ee80	" \n :::Stop already. Your bullshit is not wel...

153164 rows × 2 columns



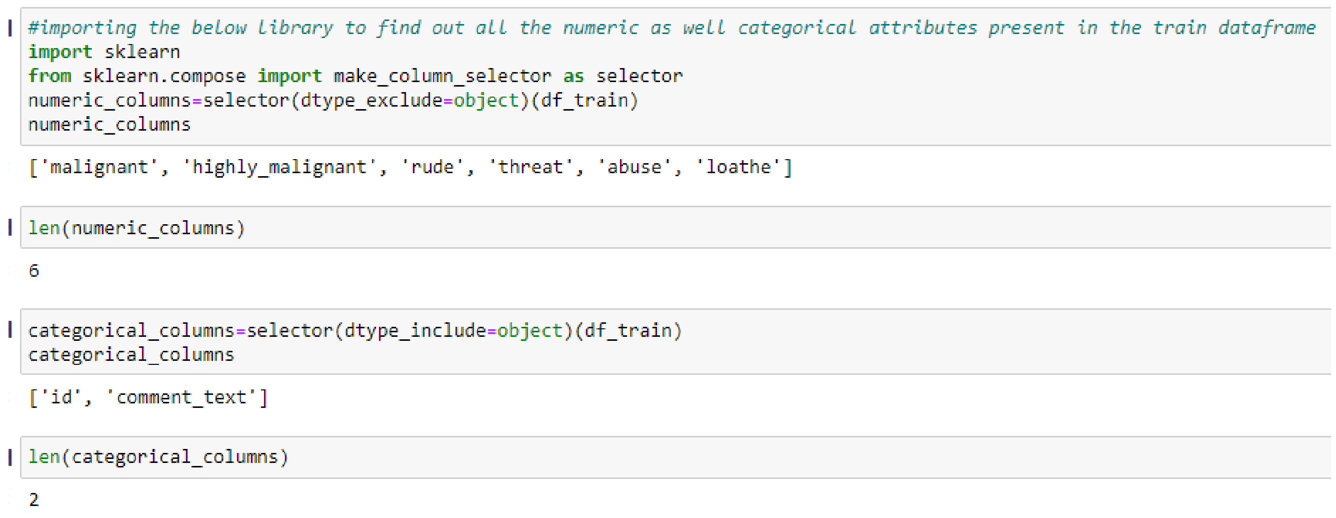
```
#columns of the train dataframe are-
df_train.columns
```

```
Index(['id', 'comment_text', 'malignant', 'highly_malignant', 'rude', 'threat',
       'abuse', 'loathe'],
      dtype='object')
```

```
#columns of the test dataframe are-
df_test.columns
```

```
Index(['id', 'comment_text'], dtype='object')
```

```
#numeric columns of train dataset
df_train_numeric=df_train.select_dtypes(exclude='object')
df_train_numeric
```



```
#importing the below library to find out all the numeric as well categorical attributes present in the train dataframe
import sklearn
from sklearn.compose import make_column_selector as selector
numeric_columns=selector(dtype_exclude=object)(df_train)
numeric_columns
```

```
['malignant', 'highly_malignant', 'rude', 'threat', 'abuse', 'loathe']
```

```
len(numeric_columns)
```

```
6
```

```
categorical_columns=selector(dtype_include=object)(df_train)
categorical_columns
```

```
['id', 'comment_text']
```

```
len(categorical_columns)
```

```
2
```

Exploratory Data Analysis

```
print(f'For Train dataset-')
print('\tTotal Row"s are',df_train.shape[0])
print('\tTotal Columns are',df_train.shape[1])
print('\tShape is',df_train.shape)
```

```
For Train dataset-
        Total Row"s are 159571
        Total Columns are 8
        Shape is (159571, 8)
```

```
print(f'For Test dataset-')
print('\tTotal Row"s are',df_test.shape[0])
print('\tTotal Columns are',df_test.shape[1])
print('\tShape is',df_test.shape)
```

```
For Test dataset-
        Total Row"s are 153164
        Total Columns are 2
        Shape is (153164, 2)
```

```
#two dimensional of train dataframe
df_train.ndim
```

```
]: 2
```

```
#numeric columns of train dataset
df_train_numeric=df_train.select_dtypes(exclude='object')
df_train_numeric
```

	malignant	highly_malignant	rude	threat	abuse	loathe
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
...	...	...	...	...	...	...
159566	0	0	0	0	0	0
159567	0	0	0	0	0	0
159568	0	0	0	0	0	0
159569	0	0	0	0	0	0
159570	0	0	0	0	0	0

159571 rows × 6 columns

```
#Categorical columns of train dataset
df_train_categorical=df_train.select_dtypes(include='object')
df_train_categorical
```

	id	comment_text
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...
...	...	...
159566	ffe987279560d7ff	":::::And for the second time of asking, when ...
159567	ffea4adeee384e90	You should be ashamed of yourself \n\nThat is ...
159568	ffee36eab5c267c9	Spitzer \n\nUmm, theres no actual article for ...
159569	fff125370e4aaaf3	And it looks like it was actually you who put ...
159570	fff46fc426af1f9a	"\nAnd ... I really don't think you understand...

159571 rows × 2 columns

```
df_train.nunique()
```

```
5]: id                  159571
    comment_text        159571
    malignant                2
    highly_malignant         2
    rude                     2
    threat                   2
    abuse                    2
    loathe                   2
    dtype: int64
```

As we can see that the comments type has only two categories

```
df_test.nunique()
```

```
6]: id              153164
    comment_text    153164
    dtype: int64
```

```
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
```

```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 153164 entries, 0 to 153163
Data columns (total 2 columns):
 #   Column        Non-Null Count   Dtype 
---  ------        --------------   ----- 
 0   id            153164 non-null  object
 1   comment_text  153164 non-null  object
dtypes: object(2)
memory usage: 2.3+ MB
```

```
#Checking for null values in train dataset
df_train.isnull().sum()
```

```
id                  0
comment_text        0
malignant           0
highly_malignant    0
rude                0
threat              0
abuse               0
loathe              0
dtype: int64
```

```
#Checking for null values in test dataset
df_test.isnull().sum()
```

```
id              0
comment_text    0
dtype: int64
```

Data Preprocessing Done

There were no null value present in the train and test dataset and I've used Label Encoder method to convert all the categorical variables into the numeric form. Also, I have created separated data frame for numeric as well as categorical variables. The threshold value of Skewness is +/=0.5. I've used power transformer method to remove the skewness and later on I performed standard scaler to normalize the dataset.

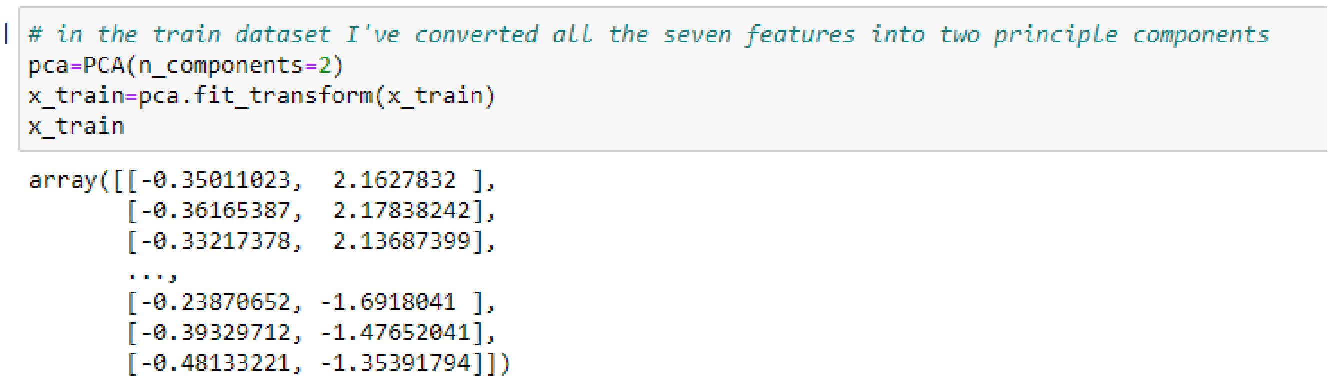
After applying the correlation matrix I found that attribute 'id' is almost zero correlated w.r.t. our target variable hence we could delete this attribute from our train and test dataset as well but as there were less variables given so I didn't drop that column but later I have converted these seven attributes into two variables w.r.t. PCA technique as these information could be useful.

Variables 'Rude and Abuse' are highly correlated with each other about 74%. Also Variables Rude and abuse are highly correlated w.r.t. our target variable 'Malignant' as 68% and 65% respectively. Variable Id is zero percent correlated w.r.t. target variable

Since two of the feature variables don't have any outliers presence and rest of the feature variables have only 0 and 1 as the label, hence we didn't remove the outliers because if we try to remove it then our rows containing label as 1 would be deleted and our dataset will be consisting of label 0 only. Also, we can't perform outlier removal in our target variable.

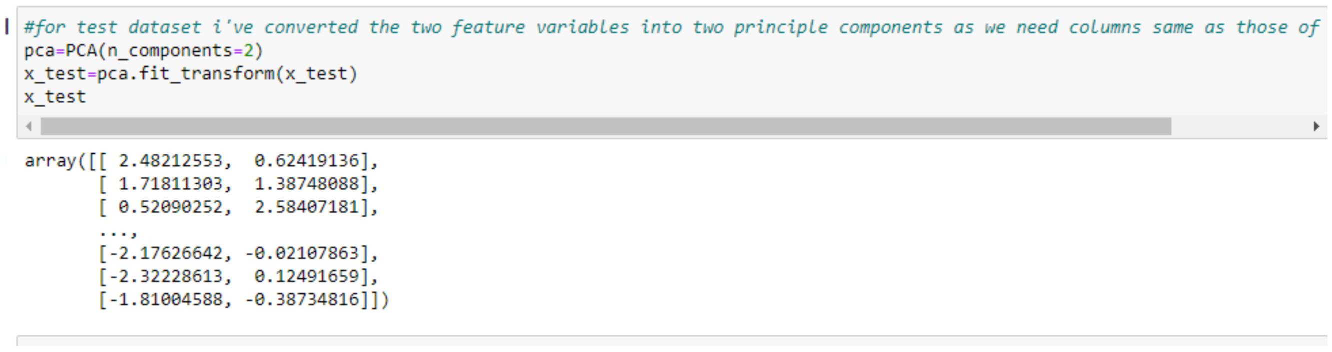
I’ve used the Principle Component Analysis (PCA) method in the train and test dataset in-order to reduce the dimensionality into two principle components. Initially there were seven feature variables presented in the train data frame and with the help of PCA I’ve converted it into 2 Principle Components and the same number of principle components were obtained in test dataset as well.

Kindly see the below screenshots for your reference-



```
# in the train dataset I've converted all the seven features into two principle components
pca=PCA(n_components=2)
x_train=pca.fit_transform(x_train)
x_train
```

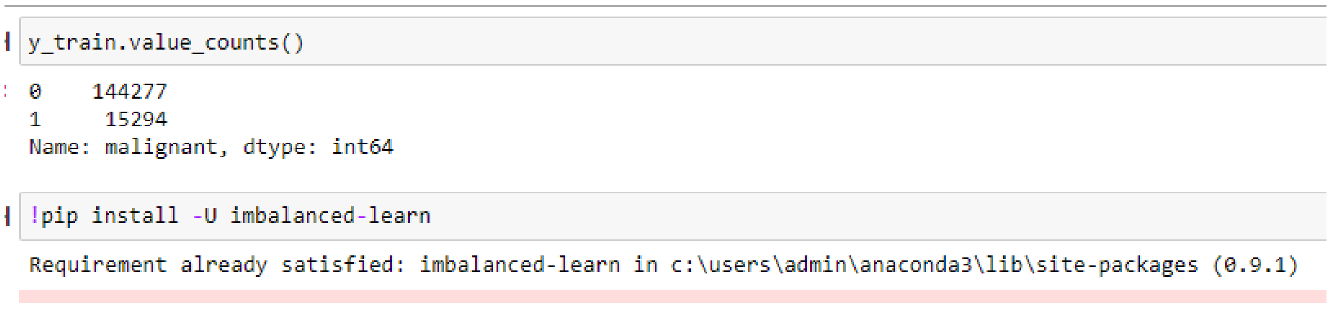
```
array([[-0.35011023,  2.1627832 ],
       [-0.36165387,  2.17838242],
       [-0.33217378,  2.13687399],
       ...,
       [-0.23870652, -1.6918041 ],
       [-0.39329712, -1.47652041],
       [-0.48133221, -1.35391794]])
```



```
#for test dataset i've converted the two feature variables into two principle components as we need columns same as those of
pca=PCA(n_components=2)
x_test=pca.fit_transform(x_test)
x_test
```

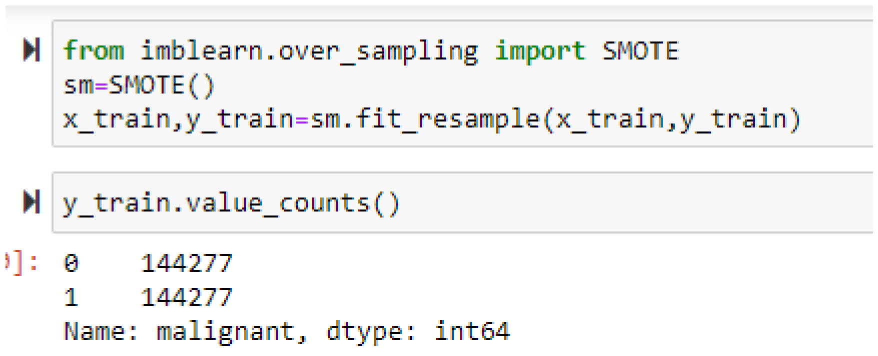
```
array([[ 2.48212553,  0.62419136],
       [ 1.71811303,  1.38748088],
       [ 0.52090252,  2.58407181],
       ...,
       [-2.17626642, -0.02107863],
       [-2.32228613,  0.12491659],
       [-1.81004588, -0.38734816]])
```

At the end I’ve used Imblearn Balancing technique to convert the unbalanced classes of our target variable ‘malignant’ into equal ratio of classes. As you can see in the below attached screenshot the classes are imbalanced in nature.



```
y_train.value_counts()

0    144277
1     15294
Name: malignant, dtype: int64

!pip install -U imbalanced-learn

Requirement already satisfied: imbalanced-learn in c:\users\admin\anaconda3\lib\site-packages (0.9.1)
```

After performing the Synthetic Minority Oversampling Technique (**SMOTE**) we've increased the number of cases in our label in a balanced way as you can see in the below attached screenshot.



```
from imblearn.over_sampling import SMOTE
sm=SMOTE()
x_train,y_train=sm.fit_resample(x_train,y_train)

y_train.value_counts()

]:  0    144277
    1    144277
    Name: malignant, dtype: int64
```

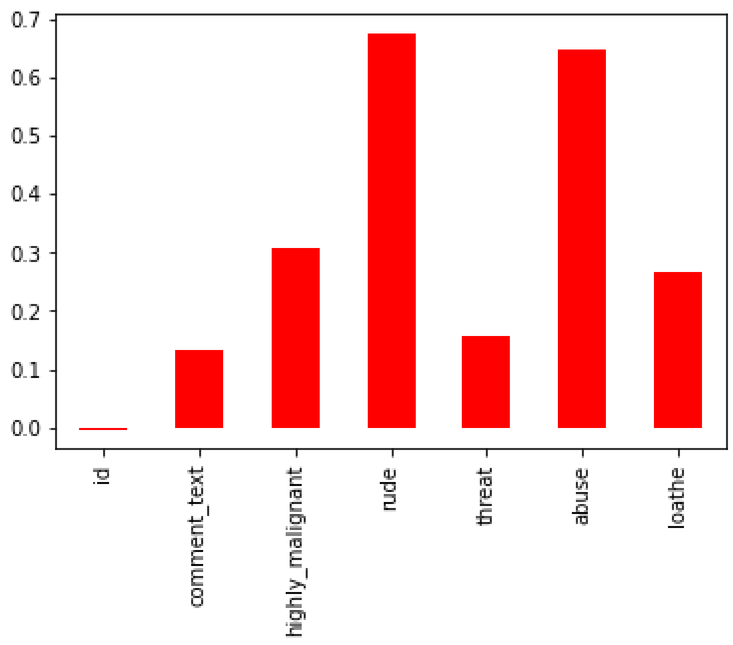
Now the dataset of our target variable is balanced

Data Inputs- Logic- Output Relationships

Since, as mentioned in our problem statement we've to build malignant comment classifier model. So on the basis of output I have checked the correlational input data with my output, as shown in below figure.

Multicollinearity

```
df_train.corr()['malignant'].drop(['malignant']).plot(kind='bar',color='r')
plt.show()
```



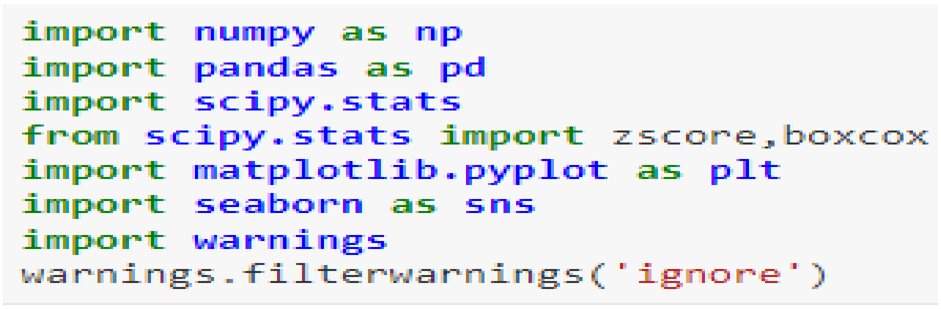
As I checked the input and found that almost all the data is quite positively correlated with my output data, except one column data. Variable 'id' is almost zero correlated with our target variable.

State the set of assumptions (if any) related to the problem under consideration

No.

Hardware and Software Requirements and Tools Used

I have used **Python IDE** (Integrated Development environment) as a dedicated software throughout solving this project.



```
import numpy as np
import pandas as pd
import scipy.stats
from scipy.stats import zscore,boxcox
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Python Libraries that I've used throughout the process are-

- Numpy- It is use for linear algebra
- Pandas- data analysis/manipulation library
- Scipy- Utility function for optimization
- Matplotlib-Data visualization and plotting library
- Seaborn- Data visualization and statistical plotting library
- Sklearn- Machine Learning Tool
- Imblearn- Deal with classification problems of Imbalanced classes
- Statsmodels-Deal with advanced statistics

Classes-

- Label Encoder- Encoding the categorical variables into number category
- Simple Imputer- Replacing the null values with mean, median or mode
- Variance_Inflation_Factor- Calculate Multicollinearity
- Power Transformer- Remove skewness
- Standard Scaler- Normalize the feature variables
- Principle Component Analysis- Reduce the dimension of the data frame
- Synthetic Minority Oversampling Technique (SMOTE)
- Cross_Val_Score- CV score
- Grid Search CV- Find out the best parameters for the model

Model/s Development and Evaluation

Identification of possible problem-solving approaches (methods)

Statistical Method-

When I used the describe function then I find out that some of the attributes has more median than its mean and so it indicates that there is the possibility of left skewed data in the dataset and the interquartile range for the variables **id** and **comment text** are varying slightly hence it shows that some variables can skew left hand side and it indicates that some of the variables might not be normally distributed.

Also, I have used correlation method to check what are the variables that are giving strong correlation w.r.t Target variable **Malignant** and they are- 'Abuse and Rude'.

Analytical Method-

I've uses Boxplot, Scatter Plots and Distribution Plots to check the outliers and skewness of the variables respectively through the plotting's.

Testing of Identified Approaches (Algorithms)

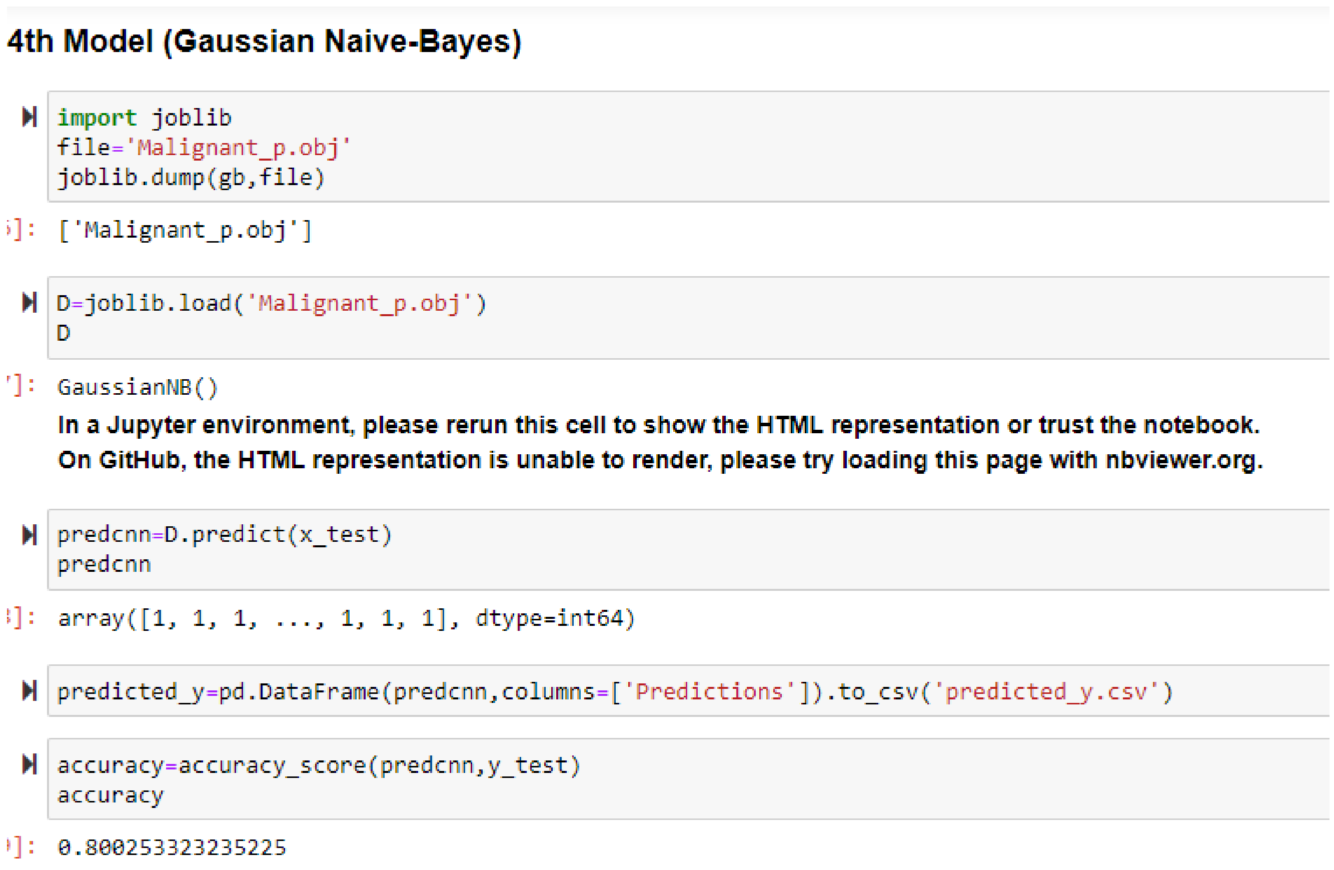
Listing down all the algorithms used for the training and testing.

- from sklearn.linear_model import LogisticRegression
- from sklearn.model_selection import train_test_split
- from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,roc_auc_score,roc_curve
- from sklearn.model_selection import cross_val_score
- from sklearn.model_selection import GridSearchCV
- from sklearn.naive_bayes import GaussianNB
- from sklearn.neighbors import KNeighborsClassifier
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier,GradientBoostingClassifier
- from sklearn.metrics import roc_curve,roc_auc_score
- lr=LogisticRegression()
- gb=GaussianNB()
- neighbor=KNeighborsClassifier()
- dtc=DecisionTreeClassifier()
- rfc=RandomForestClassifier()
- ad=AdaBoostClassifier()
- grd=GradientBoostingClassifier()

Run and Evaluate selected models

Best Model (Gaussian Naive - Bayes)

I've tested 6 different model and the best model that I got is 'Gaussian Naive- Bayes' having 80% of testing accuracy, F1-score and roc_auc score each.



4th Model (Gaussian Naive-Bayes)

```
import joblib
file='Malignant_p.obj'
joblib.dump(gb,file)
```

```
['Malignant_p.obj']
```

```
D=joblib.load('Malignant_p.obj')
D
```

```
GaussianNB()
```

**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.**

```
predcnn=D.predict(x_test)
predcnn
```

```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

```
predicted_y=pd.DataFrame(predcnn,columns=['Predictions']).to_csv('predicted_y.csv')
```

```
accuracy=accuracy_score(predcnn,y_test)
accuracy
```

```
0.800253323235225
```

```python
print('\n Classification Report-\n',classification_report(y_test,predcnn))
print('\n Confusion Metrix-\n',confusion_matrix(y_test,predcnn))
```

```
 Classification Report-
               precision    recall  f1-score   support

           0       0.88      0.77      0.82     91392
           1       0.71      0.85      0.77     61772

    accuracy                           0.80    153164
   macro avg       0.80      0.81      0.80    153164
weighted avg       0.81      0.80      0.80    153164


 Confusion Metrix-
 [[70283 21109]
 [ 9485 52287]]
```

```python
conclusion=pd.DataFrame(data=[predcnn,y_test],index=['Predicted Malignant','Original Malignant'])
conclusion
```

2]:

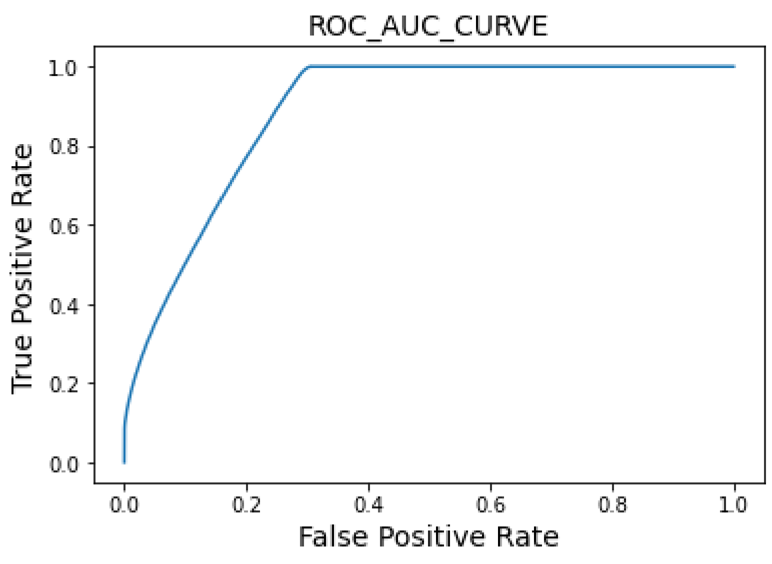
	0	1	2	3	4	5	6	7	8	9	...	153154	153155	153156	153157	153158	153159	153160	153161	153162	153163
Predicted Malignant	1	1	1	1	1	1	1	1	1	1	...	0	0	1	0	0	1	0	1	1	1
Original Malignant	1	1	1	1	1	1	1	1	1	1	...	0	0	0	0	0	0	0	0	0	0

2 rows × 153164 columns

Plotting ROC_AUC_CURVE

```python
prob=D.predict_proba(x_test)[:,1]
FPR,TPR,THRESHOLD=roc_curve(y_test,prob)
plt.plot(FPR,TPR)
plt.xlabel('False Positive Rate',fontsize=14)
plt.ylabel('True Positive Rate',fontsize=14)
plt.title('ROC_AUC_CURVE',fontsize=14)
print(f'\nThe ROC AUC Score is= {roc_auc_score(y_test,predcnn)}')
```

```
The ROC AUC Score is= 0.8077396951767012
```



Key Metrics for success in solving problem under consideration

Below are the some **Classification Models** where I used below metrics -

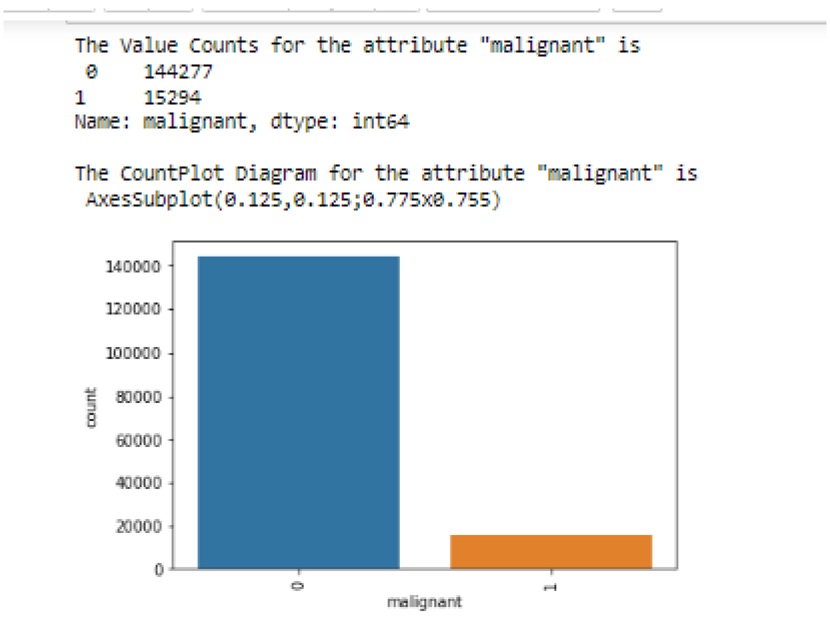
Classification Models
Gaussian NB
Gradient Boosting Classifier
K Neighbors Classifier
Random Forest Classifier
Decision Tree Classifier
Ada Boost Classifier

- CV Score – Model testing Accuracy
- F1-Score – It combines precision and recall into a single number
- ROC AUC Score – ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes.
- ROC AUC Curve- The Reciever operating characteristic curve plots the true positive (TP) rate versus the false positive (FP) rate at different classification thresholds

Visualizations

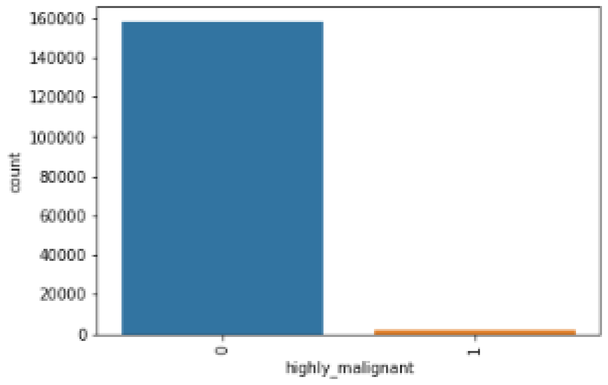
The plots that I've visualized during my projects are –

- Histograms
- Count Plot
- Pearson Correlation Heatmap
- Scatter Plot
- Distribution plot, box plot, Violin Plot and etcetera.

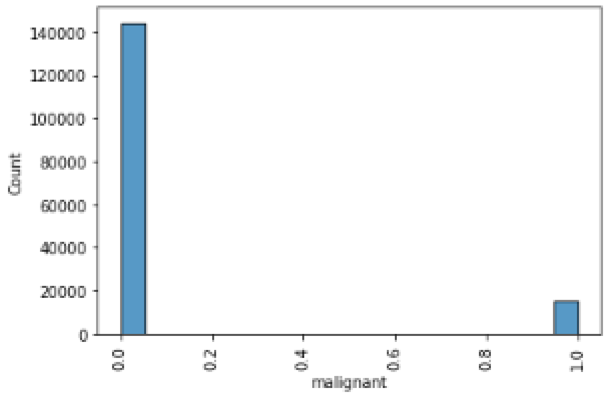


```
The Value Counts for the attribute "malignant" is
 0    144277
1     15294
Name: malignant, dtype: int64

The CountPlot Diagram for the attribute "malignant" is
 AxesSubplot(0.125,0.125;0.775x0.755)
```

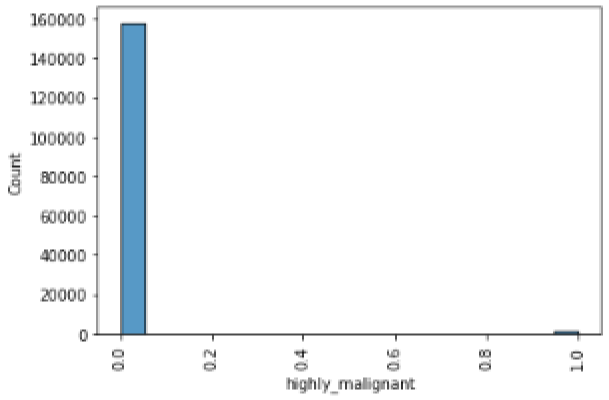
```
The Value Counts for the attribute "highly_malignant" is
 0    157976
1      1595
Name: highly_malignant, dtype: int64

The CountPlot Diagram for the attribute "highly_malignant" is
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
The Histogram Diagram for the attribute "malignant" is
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
The Histogram Diagram for the attribute "highly_malignant" is
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
df_train[(df_train['threat']==1) & (df_train['loathe']==1)]
```

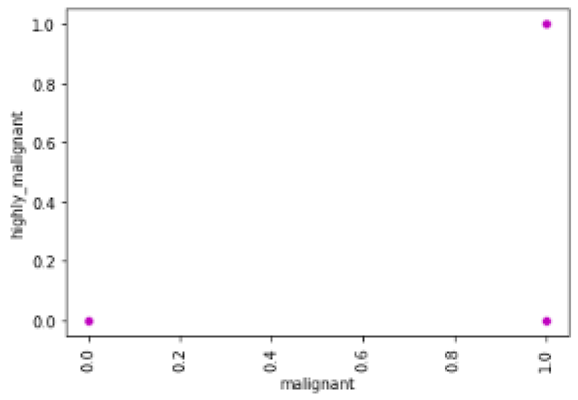
7]:

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
176	006b94add72ed61c	I think that your a Fagget get a oife and burn...	1	0	1	1	1	1
1017	02c6e41e4b317ac3	WOULDN'T BE THE FIRST TIME BITCH. FUCK YOU I'L...	1	1	1	1	1	1
1312	039296aa294ee10b	SHUT UP, YOU FAT POOP, OR I WILL KICK YOUR ASS!!!	1	1	1	1	1	1
1535	0420f5f4e950566b	Demonte Morton \n\nU bastard stop deletin' my ...	1	0	1	1	1	1
1878	0512f33cf8807fa2	Aaron Swartz \n\nStop fucking reverting my god...	1	0	1	1	1	1
...	...	...	...	...	...	...	...	...
157010	d70546ae47ad8b67	of killing all the jews	1	0	0	1	0	1
157428	ddf8dc5cf6931f48	Ok.... \n\nBitch i swear to God i will fuckin ...	1	1	1	1	1	1
157718	e26b106943e02cbf	bitch \nyou are a fucking hore. you suck dick ...	1	0	1	1	1	1
159029	f780e4f42aa5a344	Death to Musulmans!	1	0	0	1	0	1
159400	fd052883fa6a8697	Shalom \n\nSemite, get the fuck out of here. I...	1	1	1	1	1	1

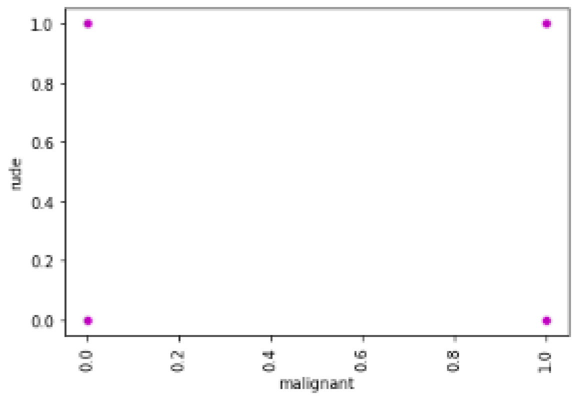
98 rows × 8 columns

98 threat comments are loathe comments

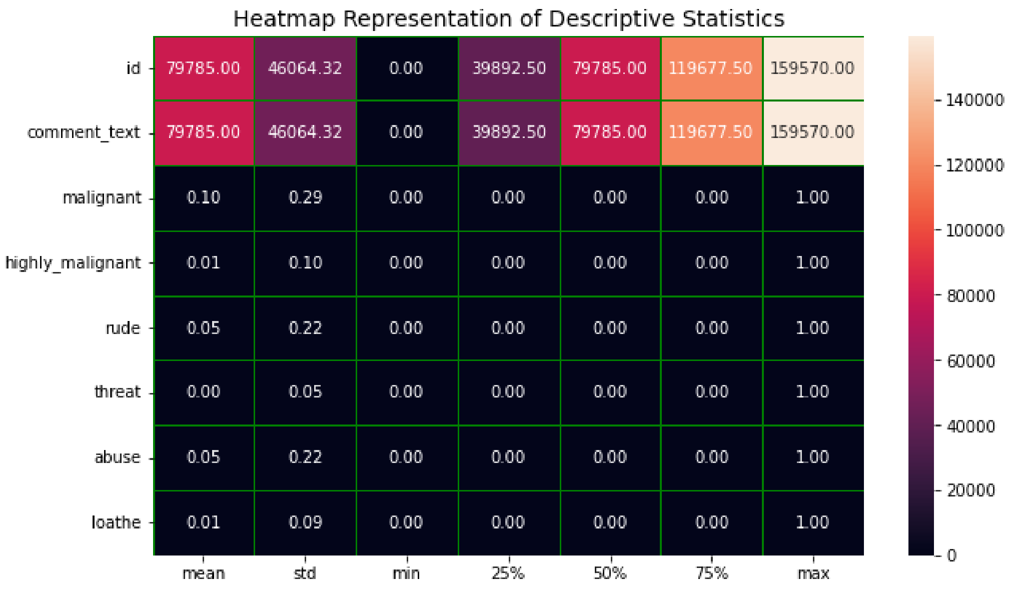
```
The Scatter Plot for the attribute "malignant" & "highly_malignant" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
The Scatter Plot for the attribute "malignant" & "rude" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```

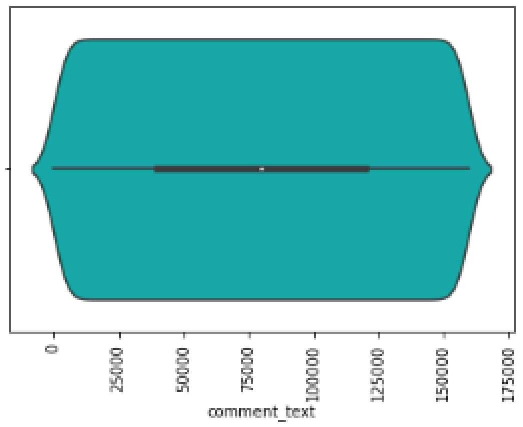


```
plt.figure(figsize=(10,6))
sns.heatmap(df_train.describe()[1:].transpose(),annot=True,linecolor='Green',linewidth='0.5',fmt='0.2f')
plt.title('Heatmap Representation of Descriptive Statistics',fontsize=14)
plt.show()
```

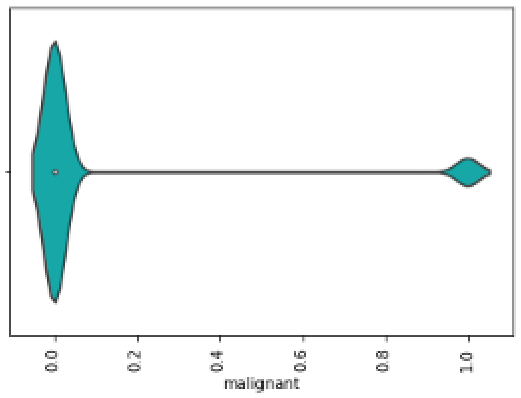


	mean	std	min	25%	50%	75%	max
id	79785.00	46064.32	0.00	39892.50	79785.00	119677.50	159570.00
comment_text	79785.00	46064.32	0.00	39892.50	79785.00	119677.50	159570.00
malignant	0.10	0.29	0.00	0.00	0.00	0.00	1.00
highly_malignant	0.01	0.10	0.00	0.00	0.00	0.00	1.00
rude	0.05	0.22	0.00	0.00	0.00	0.00	1.00
threat	0.00	0.05	0.00	0.00	0.00	0.00	1.00
abuse	0.05	0.22	0.00	0.00	0.00	0.00	1.00
loathe	0.01	0.09	0.00	0.00	0.00	0.00	1.00

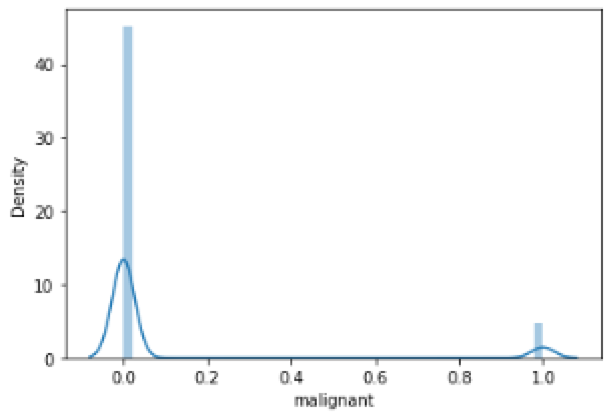
```
The Violin-Plot for the attribute "comment_text" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



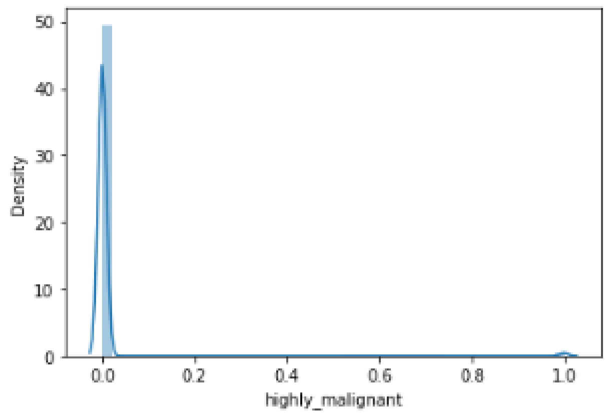
```
The Violin-Plot for the attribute "malignant" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



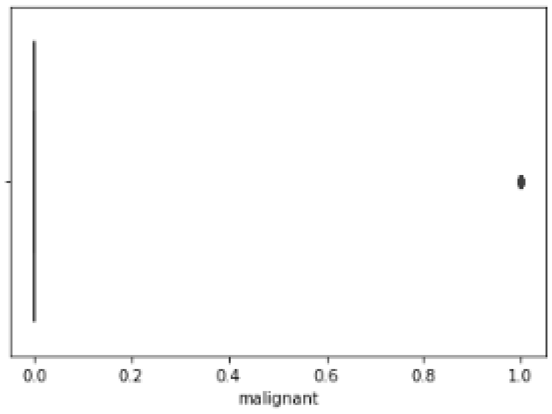
```
The Distribution Plot for attribute "malignant" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



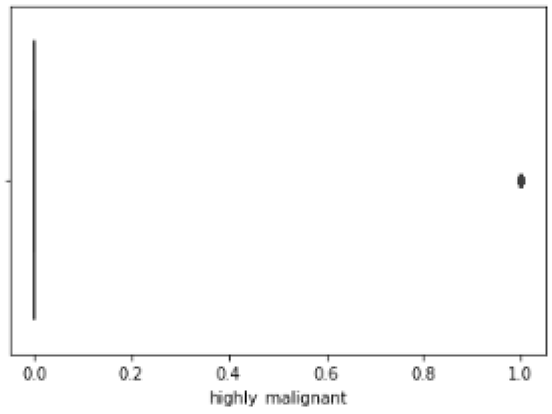
```
The Distribution Plot for attribute "highly_malignant
 AxesSubplot(0.125,0.125;0.775x0.755)
```



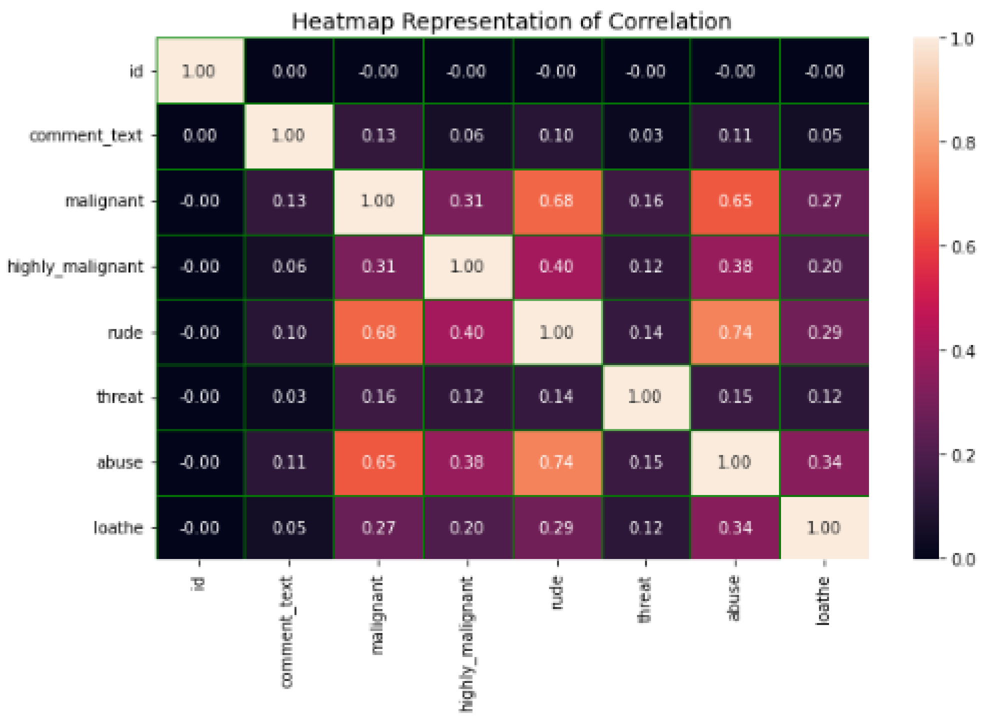
```
The Box-Plot for attribute "malignant" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
The Box-Plot for attribute "highly_malignant" is-
 AxesSubplot(0.125,0.125;0.775x0.755)
```



```
plt.figure(figsize=(10,6))
sns.heatmap(df_train.corr(),annot=True,linecolor='green',linewidth='1',fmt='0.2f')
plt.title('Heatmap Representation of Correlation',fontsize=14)
plt.show()
```



	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
id	1.00	0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00
comment_text	0.00	1.00	0.13	0.06	0.10	0.03	0.11	0.05
malignant	-0.00	0.13	1.00	0.31	0.68	0.16	0.65	0.27
highly_malignant	-0.00	0.06	0.31	1.00	0.40	0.12	0.38	0.20
rude	-0.00	0.10	0.68	0.40	1.00	0.14	0.74	0.29
threat	-0.00	0.03	0.16	0.12	0.14	1.00	0.15	0.12
abuse	-0.00	0.11	0.65	0.38	0.74	0.15	1.00	0.34
loathe	-0.00	0.05	0.27	0.20	0.29	0.12	0.34	1.00

Interpretation of the Results

Below are some of the point's w.r.t. visualization, pre-processing presented above -

- Out of 159571, 5666 comments are classified as malignant but these comments are neither highly_malignant, neither rude, neither threat, neither abusive & nor loathe.
- 3800 comments are rude and abusive both.
- 1758 comments are neither abusive nor loathe
- 1215 comments are only abusive comments
- 989 comments are highly malignant as they are abusive and rude both.
- 618 comments are not highly malignant but they are abusive, rude and loathe all together.
- 317 comments those are not malignant are of rude type.
- 134 comments are abusive and loathe both.
- 56 comments are not highly malignant but they are rude, abusive, loathe and threat all together.
- 41 comments are not rude, abusive, loathe and threat all together but they are malignant and highly malignant comments.
- There are total 143346 comments which are not malignant, not highly_malignant, not rude, not threat, not abusive and not loathe all together.

- There are total 31 comments which are malignant, highly_malignant, rude, threat, abusive and loathe all together.
- 98 threat comments are loathe comments
- 1160 abusive comments are loathe
- 1032 rude comments are loathe
- 313 highly malignant comments are loathe
- 1302 malignant comments are loathe.
- 7344 malignant comments are abusive.
- 449 malignant comments are threat.
- 7926 malignant comments are rude.
- 1595 malignant comments are highly malignant.
- Total 15294 comments are malignant
- Out of 159571 comments only 15294 comments are malignant.
- Out of 159571 comments only 1595 comments are highly_malignant.
- Out of 159571 comments only 8449 comments are rude.
- Out of 159571 comments only 478 comments are threat.
- Out of 159571 comments only 7877 comments are abuse.
- Out of 159571 comments only 1405 comments are loathe.
- Also there are 159571 unique id's and comment texts are recorded in this datasets.

CONCLUSION

Classification Models	Testing Accuracy (in %)	ROC AUC Score (in %)	F1-Score (in %)	Ranking
Gaussian NB	80	80	80	1
Gradient Boosting Classifier	79	74	79	2
K Neighbors Classifier	73	69	73	3
Random Forest Classifier	62	60	62	4
Decision Tree Classifier	61	64	61	5
Ada Boost Classifier	52	52	52	6

As we can see above all the models, Gaussian Naive Bayes model seems perfect one as compare to all as the testing accuracy score is 80%, F1 score is 80% and auc-roc score is of 80% which is best as compare to other models, also the ROC AUC Score is 0.80 which is greater than 0.6 of threshold value and it indicates that out of 100 times, 80 times model is predicting the right classes and it is still a great accuracy. As we can see in the conclusion portion in the project itself we have got almost same value in prediction column as compare to original column. So we can say that this model has great accuracy in predicting the malignant comments.

Learning Outcomes of the Study in respect of Data Science

I have used the Classification Model using multiple algorithms to design and optimize the results. Some of the classes which I explored from Scikit-learn libraries were – Statistics, Analytical Modelling, Hyper Tuning Method, CV Score, Predictive Modelling and different parameter, etc.

I condensed to 2 columns keeping in mind the Principle Component Analysis. This resulted in condensed data being trained in a much shorter span of time with utmost accuracy. The visualizations helped in better and quick understanding of the outliers and skewness present in the data sets.

With Mathematical Modeling helped me to find-out the corresponding mean, median, mode and relationship among the variables. Whereas Statistical Modeling helped in Correlation for understanding the relationship among the variables.

One of the key challenges was like initially there were two dataset given; one is for training and other is for testing and more importantly the test dataset had only two columns and the train dataset had 7 columns except the target variable 'Malignant', so that's why I condensed the train dataset into two principle components because we need same array while making prediction of the target variable.

Limitations of this work and Scope for Future Work

The same analysis which is done for this project can be used for other malignant classifier projects because we've applied different classifier algorithms on large number of dataset and that's why probability could be same while predicting other test datasets.