PROJECT REPORT ON:

"Emails Spam 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 SMS Spam Collection is a set of SMSs tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according being ham (legitimate) or spam.

What is a Spam Filtering? Spam Detector is used to detect unwanted, malicious and virus infected texts and helps to separate them from the no spam texts. It uses a binary type of classification containing the labels such as 'ham' (no spam) and spam. Application of this can be seen in Google Mail (GMAIL) where it segregates the spam emails in order to prevent them from getting into the user's inbox.

The files contain one message per line. Each line is composed by two columns: v1 contains the label (ham or spam) and v2 contains the raw text.

1.2 Conceptual Background of the Domain Problem

Predictive modelling, Classification algorithms are some of the machine learning techniques used along with the various libraries of the NLTK suite for Classification of comments. Using NLTK tools, the frequencies of malignant words occurring in textual data were estimated and given appropriate weightage, whilst filtering out words, and other noise which do not have any impact on the semantics of the comments and reducing the words to their base lemmas for efficient processing and accurate classification of the comments.

1.3 Review of Literature

Nowadays, email messaging is a way with which people communicate important messages to each other using Internet. It's a very common way through which clients communicate among themselves formally. Now a days the extent to which these emails are sent has been increasing rapidly. Along with these emails, Spam emails are also sent in bulk through different platforms. These spam emails are usually difficult to recognize and it is the major problem that is being faced by the users. Spam consumes almost 98% of billions of emails sent every day. Due to the presence of different email filtering systems already present in the market, Spammers have become aware of these systems. Therefore, Spammers are trying different ways to send spam or junk mails to a number of users. One of them is by sending spam images and pdfs. For this kind of spam emails, presently there are not very effective systems present in the market. This paper illustrates a survey of different existing email classification system which can classify the email as ham or spam.

1.4 Motivation for the Problem Undertaken

Email has become one of the most important forms of communication. In 2014, there are estimated to be 4.1 billion email accounts worldwide, and about 196 billion emails are sent each day worldwide. Spam is one of the major threats posed to email users. In 2013, 69.6% of all email flows were spam. Links in spam emails may lead to users to websites with malware or phishing schemes, which can access and disrupt the receiver's computer system. These sites can also gather sensitive information from. Additionally, spam costs businesses around $2000 per employee per year due to decreased productivity. Therefore, an effective spam filtering technology is a significant contribution to the sustainability of the cyberspace and to our society.

So that we need to do spam filtering so user more user friendly. From above model building we got the Complement Naive Bayes Classifier is a best model deciding whether the emails have spam or not.

2.Analytical Problem Framing

2.1 Mathematical/ Analytical Modelling of the Problem

Various Classification analysis techniques were used to build predictive models to understand the relationships that exist between user emails and the corresponding user label.

The user emails are collected, processed and normalized. Based on the context of the reviews on various items, with similar label, prediction of the label for a given email can be made based on similar email which already have corresponding label.

In order to predict label for user emails, models such as Logistic regression, Random Forest Classifier Boost Classifier, Extreme Gradient Boost Classifier, and Complement Naïve Bayes Classifier.

2.2 Data Sources and their formats

The dataset is provided by Flip Robo which is in the format csv. This dataset give use for exercise. The SMS Spam Collection is a set of SMSs tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according being ham (legitimate) or spam.

Data Descriptions

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN
...	...	...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

5572 rows × 5 columns

2.3 Data Pre-processing Done

- Rows with null values were removed.
- Columns: Unnamed: 0(just a series of numbers) was dropped since it doesn't contribute to building a good model for predicting the target variable values.
- The train and test dataset contents were then converted into lowercase.
- Punctuations, unnecessary characters etc were removed, currency symbols, phone numbers, web urls, email addresses etc were replaced with single words
- Tokens that contributed nothing to semantics of the

messages were removed as Stop words. Finally retained tokens were lemmatized using WordNetLemmatizer().
- The string lengths of original comments and the cleaned comments were then compared.

2.4 Data Inputs - Logic - Output Relationships

The comment tokens so vectorized using TfidVectorizer are input and the corresponding rating is predicted based on their context as output by classification models. State the set of assumptions related to the problem under consideration

The emails content made available in Dataset is assumed to be written in English Language in the standard Greco-Roman script. This is so that the Stopword package and WordNetLemmatizer can be effectively used.

2.5 Hardware, Software and Tool Used

Hardware Used:

Processor – Intel core i3

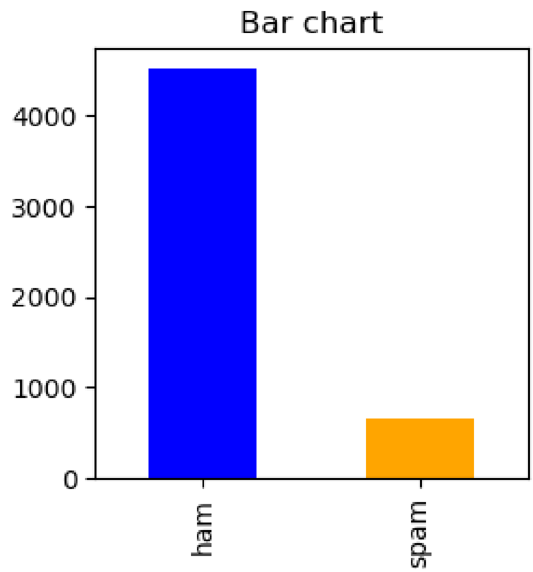
Physical Memory – 8 GB

Software Used:

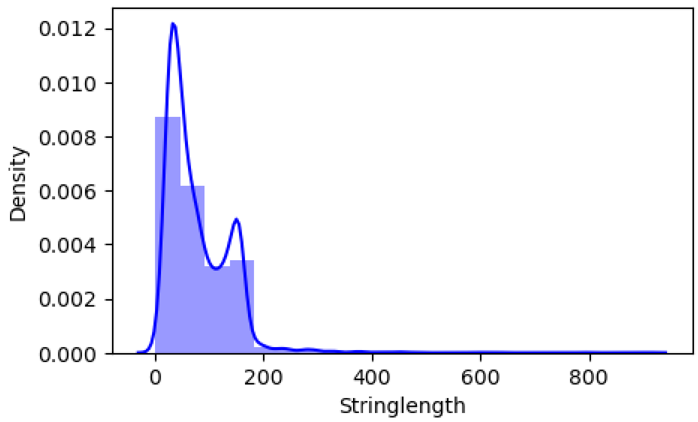
- Windows 10 Operating System
- Anaconda Package and Environment Manager
- Jupyter Notebook
- Python Libraries used: In Which Pandas, Seaborn, Matplotlib, Numpy and Scipy

- sklearn for Modelling Machine learning algorithms, Data Encoding, Evaluation metrics, Data Transformation, Data Scaling, Component analysis, Feature selection etc.

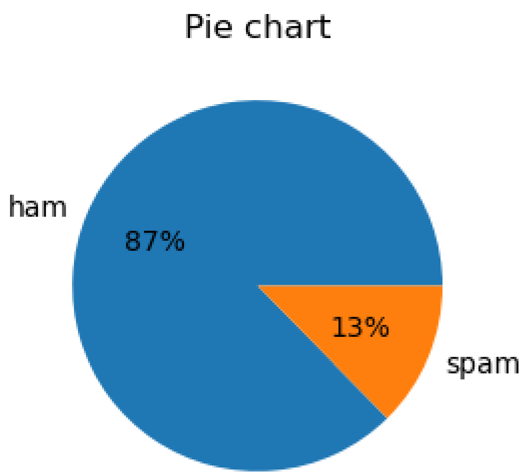
3.Data Analysis and Visualization



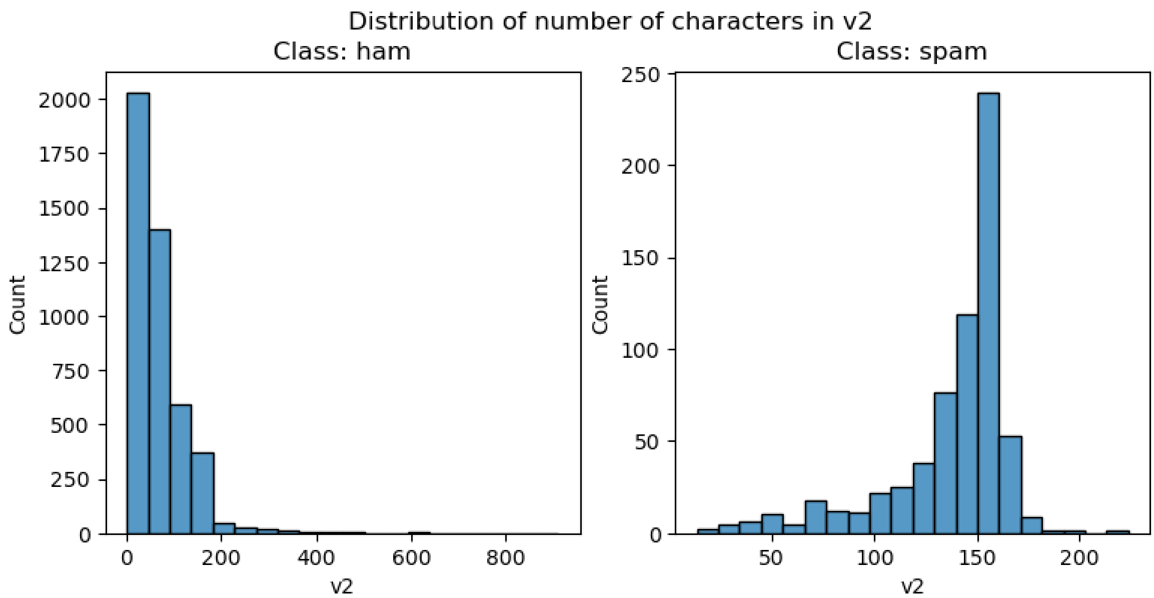
- We can see the label is not balanced wo we need balanced it.



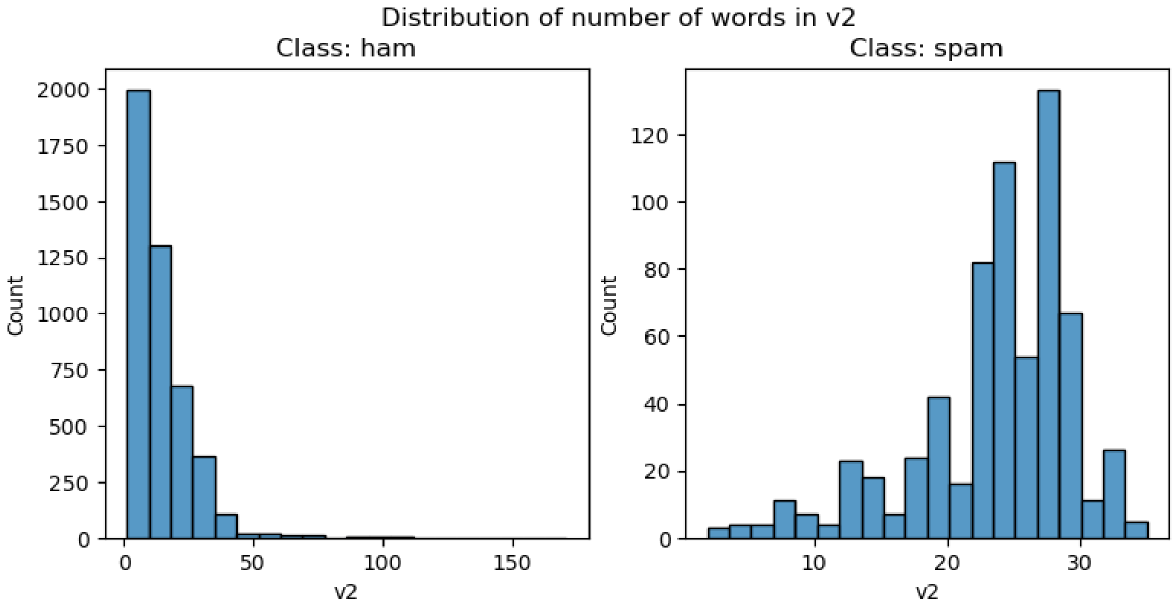
- we can see most of the emails are lies between 0 to 200 words.



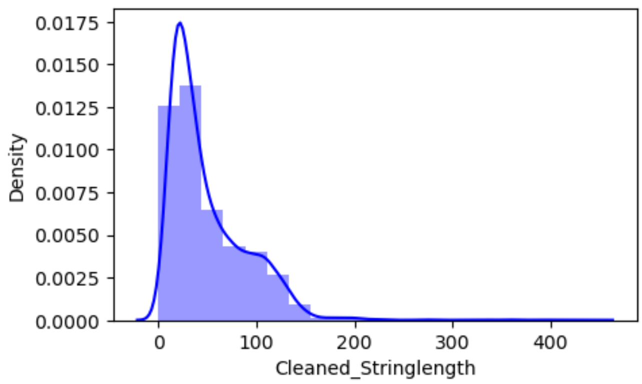
- We can see, the 13.4% of emails is spam but 86.6.
- 86.6% emails are non-spam(ham)



- We can see, that non-spam emails are lies between the 0 to 200 characters mostly.
- Spam emails characters mostly lies between the 110 to 160.

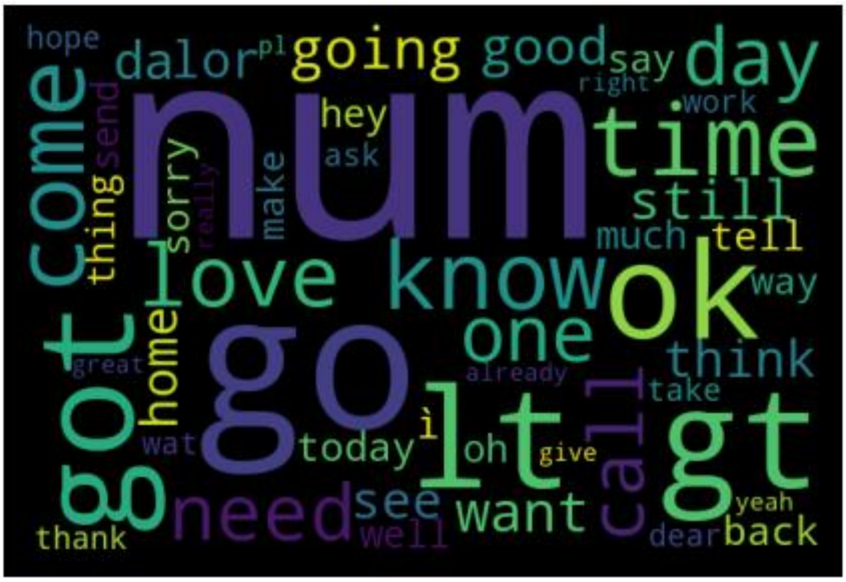


- The non-spam words in emails mostly lies between 0 to 30.
- But in spam mostly number of words lies in between 20 to 30.

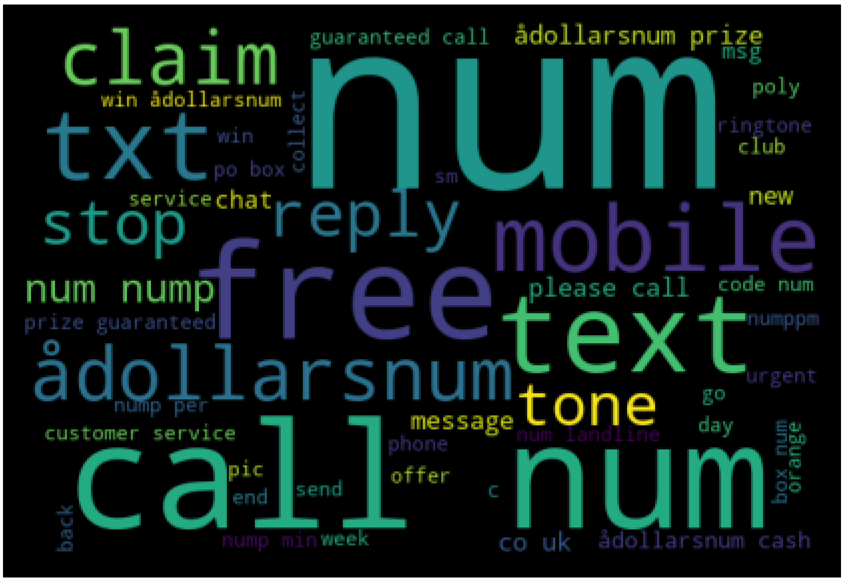


- We can see, word density has been reduced.

World clouds of the most frequent words in emails.



- We can see, so many emails are not spam.
- num, ok, love, need and come these are the word are mostly used.



- We can see, spam emails words that are mostly used as show above.
- txt, call, claim, adollarsnum and mobile these are the words are mostly used.

4. Models Development and Evaluation

4.1 The model algorithms used

The model algorithms used were as follows:

- Logistic Regression: It is a classification algorithm used to find the probability of event success and event failure. It is used when the dependent variable is binary (0/1, True/False, Yes/No) in nature. It supports categorizing data into discrete classes by studying the relationship from a given set of labelled data. It learns a linear relationship from the given dataset and then introduces a nonlinearity in the form of the Sigmoid function. It not only provides a measure of how appropriate a predictor (coefficient size) is, but also its direction of association (positive or negative).
- XGBClassifier: XGBoost uses decision trees as base learners; combining many weak learners to make a strong learner. As a result, it is referred to as an ensemble learning method since it uses the output of many models in the final prediction. It uses the power of parallel processing and supports regularization.
- RandomForestClassifier: A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the

predictive accuracy and control over-fitting. A random forest produces good predictions that can be understood easily. It reduces overfitting and can handle large datasets efficiently. The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm

- Complement Naïve Bayes Classifier: Complement Naive Bayes is somewhat an adaptation of the standard Multinomial Naive Bayes algorithm. Complement Naive Bayes is particularly suited to work with imbalanced datasets. In complement Naive Bayes, instead of calculating the probability of an item belonging to a certain class, we calculate the probability of the item belonging to all the classes.

Balancing out the classes by using SMOTE technique

```
from imblearn.over_sampling import SMOTE as sm

smt_x,smt_y = sm().fit_resample(X,y)
```

Train Test Split

Best random state was found to be 11.

```python
from sklearn.ensemble import RandomForestClassifier
maxAcc = 0
maxRS=0
for i in range(0,100):
    x_train,x_test,y_train,y_test = train_test_split(smt_x,smt_y,test_size = .30, random_state = i)
    modRF =  RandomForestClassifier()
    modRF.fit(x_train,y_train)
    pred = modRF.predict(x_test)
    acc  = accuracy_score(y_test,pred)
    if acc>maxAcc:
        maxAcc=acc
        maxRS=i
print(f"Best Accuracy is: {maxAcc} on random_state: {maxRS}")
```

```
Best Accuracy is: 0.9974169741697417 on random_state: 11
```

```python
x_train,x_test,y_train,y_test = train_test_split(smt_x,smt_y,test_size = 0.3,random_state = 11)
```

Model Building

Random Forest Classifier

```python
def metric_score(clf, x_train,x_test,y_train,y_test, train=True):
    if train:

        y_pred = clf.predict(x_train)

        print("\n========================== Train Result============================")

        print(f"Accuracy Score: {accuracy_score(y_train, y_pred) * 100:.2f}%")


    elif train==False:
        pred = clf.predict(x_test)

        print("\n==========================Test Result======================")
        print(f"Accuracy Score: {accuracy_score(y_test,pred) * 100:.2f}%")

        print("\n \n Test Classification Report \n", classification_report(y_test, pred, digits=2))

        print('\n Confusion Matrix: \n',confusion_matrix(y_test,pred))
```

```python
rf = RandomForestClassifier()
rf.fit(x_train,y_train)

metric_score(rf,x_train,x_test,y_train, y_test, train=True)

metric_score(rf,x_train,x_test,y_train, y_test, train=False)
```

```
========================= Train Result=========================
Accuracy Score: 99.98%

=========================Test Result======================
Accuracy Score: 99.67%


 Test Classification Report
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      1344
           1       1.00      0.99      1.00      1366

    accuracy                           1.00      2710
   macro avg       1.00      1.00      1.00      2710
weighted avg       1.00      1.00      1.00      2710


 Confusion Matrix:
 [[1343    1]
 [   8 1358]]
```

Cross Validation for random forest classifier

```
from sklearn.model_selection import cross_val_score
```

```
pred_rf = rf.predict(x_test)
accu = accuracy_score(y_test,pred_rf)
```

```
for j in range(4,10):
    cross = cross_val_score(rf,X,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 97.7944917739962
accuracy_score is:- 99.6678966789668


At cv:- 5
Cross validation score is:- 97.85256740334904
accuracy_score is:- 99.6678966789668


At cv:- 6
Cross validation score is:- 97.8911731444129
accuracy_score is:- 99.6678966789668


At cv:- 7
Cross validation score is:- 97.96847504527626
accuracy_score is:- 99.6678966789668


At cv:- 8
Cross validation score is:- 97.96853541709534
accuracy_score is:- 99.6678966789668


At cv:- 9
Cross validation score is:- 97.98791428908076
accuracy_score is:- 99.6678966789668
```

```
lsscore_selected = cross_val_score(rf,X,y,cv=9).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9800723796057836
The accuracy score is:  0.9966789667896679
```

Logistic Regression

```
lr = lr=LogisticRegression()
lr.fit(x_train,y_train)

metric_score(lr,x_train,x_test,y_train, y_test, train=True)

metric_score(lr,x_train,x_test,y_train, y_test, train=False)
```

```
========================= Train Result=======================
Accuracy Score: 98.70%

==========================Test Result======================
Accuracy Score: 98.41%


 Test Classification Report
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      1344
           1       0.98      0.98      0.98      1366

    accuracy                           0.98      2710
   macro avg       0.98      0.98      0.98      2710
weighted avg       0.98      0.98      0.98      2710


 Confusion Matrix:
 [[1322   22]
 [  21 1345]]
```

Cross Validation for Logistic Regression

```
pred_lr = lr.predict(x_test)
accu = accuracy_score(y_test,pred_lr)
```

```
for j in range(4,10):
    cross = cross_val_score(lr,X,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 96.45960686142817
accuracy_score is:- 98.41328413284133

At cv:- 5
Cross validation score is:- 96.49837752616276
accuracy_score is:- 98.41328413284133

At cv:- 6
Cross validation score is:- 96.61417280397532
accuracy_score is:- 98.41328413284133

At cv:- 7
Cross validation score is:- 96.67224074135193
accuracy_score is:- 98.41328413284133

At cv:- 8
Cross validation score is:- 96.73038458041641
accuracy_score is:- 98.41328413284133

At cv:- 9
Cross validation score is:- 96.69167971182817
accuracy_score is:- 98.41328413284133
```

```
lsscore_selected = cross_val_score(lr,X,y,cv=8).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9673038458041641
The accuracy score is:  0.9841328413284133
```

Complement Naive Bayes

```
CNB = ComplementNB()
CNB.fit(x_train,y_train)

metric_score(CNB,x_train,x_test,y_train, y_test, train=True)

metric_score(CNB,x_train,x_test,y_train, y_test, train=False)
```

```
========================== Train Result====================
Accuracy Score: 98.69%

==========================Test Result=====================
Accuracy Score: 98.19%

 Test Classification Report
              precision    recall  f1-score   support

           0       1.00      0.97      0.98      1344
           1       0.97      1.00      0.98      1366

    accuracy                           0.98      2710
   macro avg       0.98      0.98      0.98      2710
weighted avg       0.98      0.98      0.98      2710


 Confusion Matrix:
 [[1301   43]
 [   6 1360]]
```

Cross validation for Complement Naïve Bayes

```
pred_cnb = CNB.predict(x_test)
accu = accuracy_score(y_test,pred_cnb)
```

```
for j in range(4,10):
    cross = cross_val_score(CNB,X,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 91.37158526861717
accuracy_score is:- 98.19188191881919


At cv:- 5
Cross validation score is:- 91.23637562001343
accuracy_score is:- 98.19188191881919


At cv:- 6
Cross validation score is:- 91.23635891645625
accuracy_score is:- 98.19188191881919


At cv:- 7
Cross validation score is:- 91.25569653144822
accuracy_score is:- 98.19188191881919


At cv:- 8
Cross validation score is:- 91.21702451419029
accuracy_score is:- 98.19188191881919


At cv:- 9
Cross validation score is:- 91.27519399417596
accuracy_score is:- 98.19188191881919
```

```
lsscore_selected = cross_val_score(lr,X,y,cv=4).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9645960686142817
The accuracy score is:  0.9819188191881919
```

XGBClassifier

```python
xgb = XGBClassifier()

xgb.fit(x_train,y_train)

metric_score(xgb,x_train,x_test,y_train, y_test, train=True)

metric_score(xgb,x_train,x_test,y_train, y_test, train=False)
```

```
========================= Train Result==================
Accuracy Score: 99.59%

=========================Test Result====================
Accuracy Score: 99.04%


 Test Classification Report
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1344
           1       0.99      0.99      0.99      1366

    accuracy                           0.99      2710
   macro avg       0.99      0.99      0.99      2710
weighted avg       0.99      0.99      0.99      2710


 Confusion Matrix:
 [[1330   14]
 [  12 1354]]
```

Cross Validation for XGBClassifier

```python
pred_xgb = xgb.predict(x_test)
accu = accuracy_score(y_test,pred_xgb)
```

```python
for j in range(4,10):
    cross = cross_val_score(xgb,X,y,cv=j)
    lsc = cross.mean()
    print("At cv:-",j)
    print('Cross validation score is:-',lsc*100)
    print('accuracy_score is:-',accu*100)
    print('\n')
```

```
At cv:- 4
Cross validation score is:- 97.71706246303626
accuracy_score is:- 99.04059040590406


At cv:- 5
Cross validation score is:- 97.94933537554698
accuracy_score is:- 99.04059040590406


At cv:- 6
Cross validation score is:- 97.79427418072656
accuracy_score is:- 99.04059040590406


At cv:- 7
Cross validation score is:- 97.89099331163426
accuracy_score is:- 99.04059040590406


At cv:- 8
Cross validation score is:- 97.79435690325914
accuracy_score is:- 99.04059040590406


At cv:- 9
Cross validation score is:- 97.81376559107208
accuracy_score is:- 99.04059040590406
```

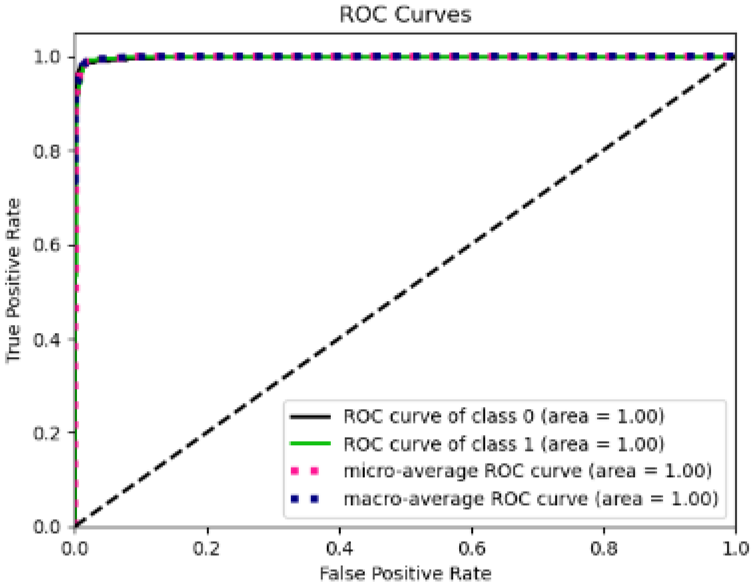
```python
lsscore_selected = cross_val_score(xgb,X,y,cv=5).mean()
print("The cv score is: ",lsscore_selected,"\nThe accuracy score is: ",accu)
```

```
The cv score is:  0.9794933537554698
The accuracy score is:  0.9904059040590406
```

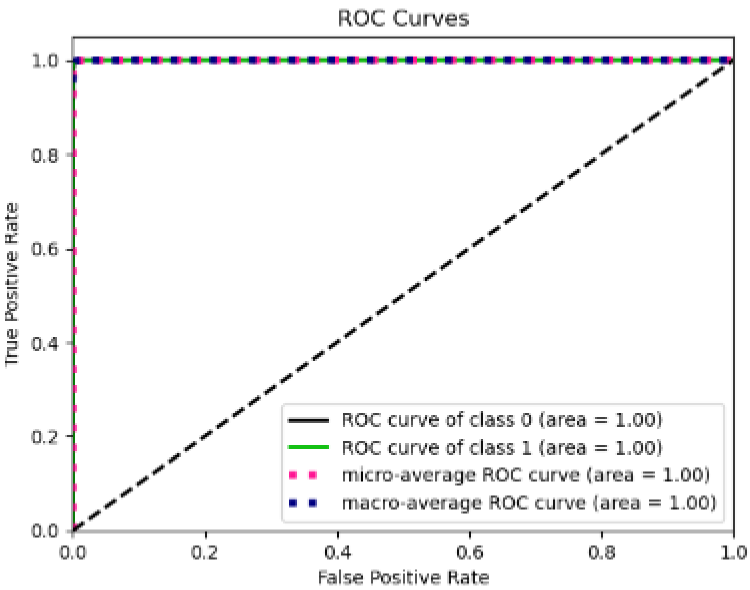
4.2 ROC AUC Curve

The AUC-ROC curve helps us visualize how well our machine learning classifier is performing. ROC curves are appropriate when the observations are balanced between each class.

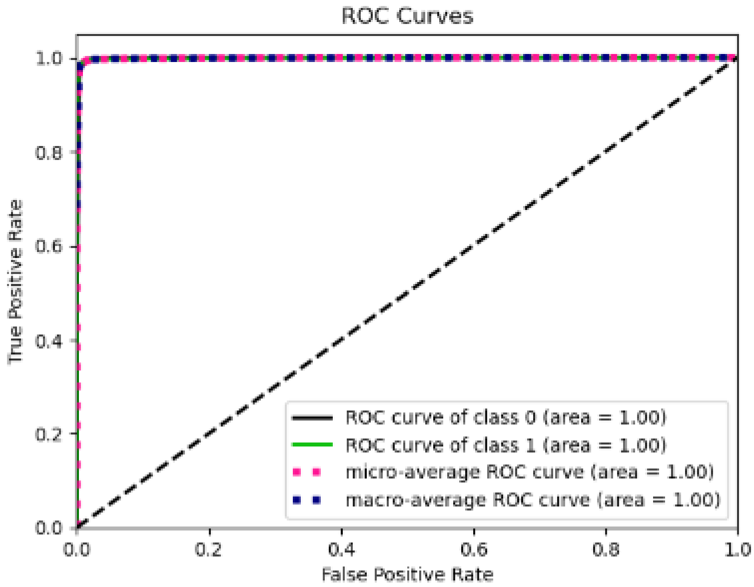
Random Forest Classifier



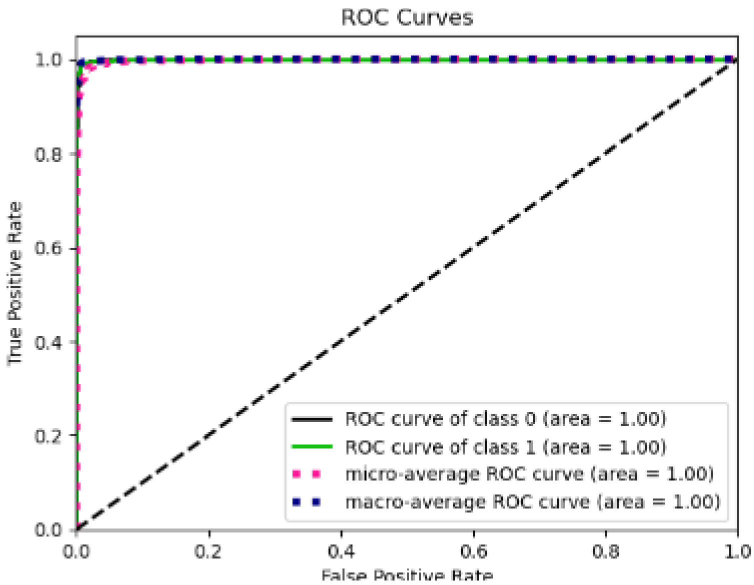
Logistic Regression



XGBClassifier



Complement Naïve Bayes



4.3 Interpretation of the results

Based on comparing the above graphs, Precision, Recall, Accuracy Scores with Cross validation scores, it is determined that Complement Naive Bayes Classifier is the best model for the dataset.

4.4 Hyperparameter Tuning

```
from sklearn.model_selection import GridSearchCV
```

```
params = {'alpha': [0.01, 0.1, 0.5, 1.0, 10.0, ],
          'fit_prior': [True, False],
          'norm': [True, False],
         # 'class_prior': [None, [0.1,]* len(n_classes), ]
         }

complement_nb_grid = GridSearchCV(ComplementNB(), param_grid=params, n_jobs=-1, cv=5, verbose=5)
complement_nb_grid.fit(x_train,y_train)
```

```
complement_nb_grid.best_score_
```

```
0.984973858007705
```

```
complement_nb_grid.best_params_
```

```
{'alpha': 0.01, 'fit_prior': True, 'norm': False}
```

```
cnb = ComplementNB(alpha = 0.01, fit_prior = True, norm = False)
cnb.fit(x_train,y_train)

metric_score(cnb,x_train,x_test,y_train, y_test, train=True)

metric_score(cnb,x_train,x_test,y_train, y_test, train=False)
```

```
========================== Train Result=======================
Accuracy Score: 99.24%

==========================Test Result=======================
Accuracy Score: 98.89%


 Test Classification Report
              precision    recall  f1-score   support

           0       1.00      0.98      0.99      1344
           1       0.98      1.00      0.99      1366

    accuracy                           0.99      2710
   macro avg       0.99      0.99      0.99      2710
weighted avg       0.99      0.99      0.99      2710


 Confusion Matrix:
 [[1316   28]
 [   2 1364]]
```

5. Conclusions

5.1 Key Finding and Conclusions

- The final model performed with 98.89% accuracy, Recall score of 0.98. It means that the model is optimized better to predict label whether it is spam email or not.
- Spam emails have become a major concern for the internet community as it poses a threat to integrity and productivity of the users. Filtering of email is very much necessary for email communication. The accurate detection of spam emails is a big issue and many filtering methods have been proposed by various research
- Not only does spam filtering help keep garbage out of email inboxes, it helps with the quality of life of business emails because they run smoothly and are only used for their desired purpose.
- So that we need to do spam filtering so user more user friendly. From above model building we got the Complement Naive Bayes Classifier is a best model deciding whether the emails have spam or not.

5.2 Limitation of this works and scope for future works

A small dataset to work with posed a challenge in building highly accurate models. By training the models on more diverse data sets, longer comments, and a more balanced dataset, more accurate and efficient classification models can be built.