NEW YORK INSTITUTE OF TECHNOLOGY

PERSONAL PROJECT - 2

College of Engineering & Computing Science

DTSC 620 M01 - Statistics For Data Science

PROJECT ASSIGNMENT - 2

Name: AVIVARTTA KRISHNA

ID: 1261351

CLASS: DTSC 620 MO1

DATE: 12/10/2022

Professor: Dr. Kiran Balgani

Reporting Tasks:

- ➔ **Compare the accuracies of the fused model with AdaBoost Ensemble with Decision Tree as the base learner. Train the classifiers using the first 1000 instances and use the remaining 3601 for testing. Feel free to create separate training and testing data files. Report your observations/conclusions and provide evidence to support your conclusions. [25 points]**

- ➔ **Compare the accuracies of the fused model with Random Forest (with 1000 base learners). Train the classifiers using the first 1000 instances and use the remaining 3601 for testing. Feel free to create separate training and testing data files. Report your observations/conclusions and provide evidence to support your conclusions. [25 points]**

- ➔ **Study the impact of training sample size on the accuracies of the fused classifier and the AdaBoost Ensemble with Decision Tree as the base learner. Compare their accuracies with the following training-test splits: 50%-50%, 60%-40%, 70%-30%, and 80%-20%. Report your observations/conclusions and provide evidence to support your conclusions. [50 points]**

DATA PRE-PROCESSING

Using spam.info() on the dataset given, we are able to see that there are 58 attributes present with 55 in Float64 data type, 2 in int64 data type and one in Object Data type.

We are also able to see from the figure present below that all attributes are non-null, therefore there is no need for data cleaning.

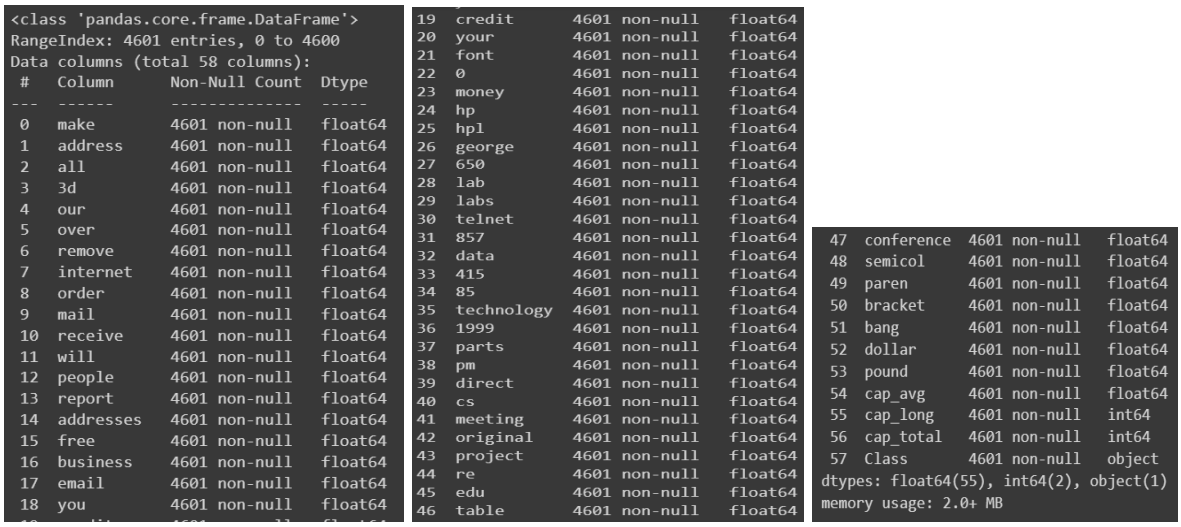


Figure 1 - Information on each attribute present in dataset

Using print(spam.columns), we are able to see all the column names in an Array index. Visualization of the same is present below in Figure 2.

```
Index(['make', 'address', 'all', '3d', 'our', 'over', 'remove', 'internet',
       'order', 'mail', 'receive', 'will', 'people', 'report', 'addresses',
       'free', 'business', 'email', 'you', 'credit', 'your', 'font', '0',
       'money', 'hp', 'hpl', 'george', '650', 'lab', 'labs', 'telnet', '857',
       'data', '415', '85', 'technology', '1999', 'parts', 'pm', 'direct',
       'cs', 'meeting', 'original', 'project', 're', 'edu', 'table',
       'conference', 'semicol', 'paren', 'bracket', 'bang', 'dollar', 'pound',
       'cap_avg', 'cap_long', 'cap_total', 'Class'],
      dtype='object')
```

Figure 2 - Index array on all column names present in dataset.

Using spam.head(), we are able to see the first five rows of the dataset with all the information present.

	make	address	all	3d	our	over	remove	internet	order	mail	...	semicol	paren	bracket	bang	dollar	pound	cap_avg	cap_long	cap_total	Class
0	0.00	0.00	0.29	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.178	0.0	0.044	0.000	0.00	1.666	10	180	ham
1	0.46	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.125	0.0	0.000	0.000	0.00	1.510	10	74	ham
2	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.000	0.0	0.000	0.000	0.00	1.718	11	55	ham
3	0.33	0.44	0.37	0.0	0.14	0.11	0.00	0.07	0.97	1.16	...	0.006	0.159	0.0	0.069	0.221	0.11	3.426	72	819	spam
4	0.00	2.08	0.00	0.0	3.12	0.00	1.04	0.00	0.00	0.00	...	0.000	0.000	0.0	0.263	0.000	0.00	1.428	4	20	spam

5 rows × 58 columns

Figure 3 - first five columns of the dataset

Since one of our classifiers is Adaboost, substantial data processing is required to provide results that are more accurate than those of random forest.

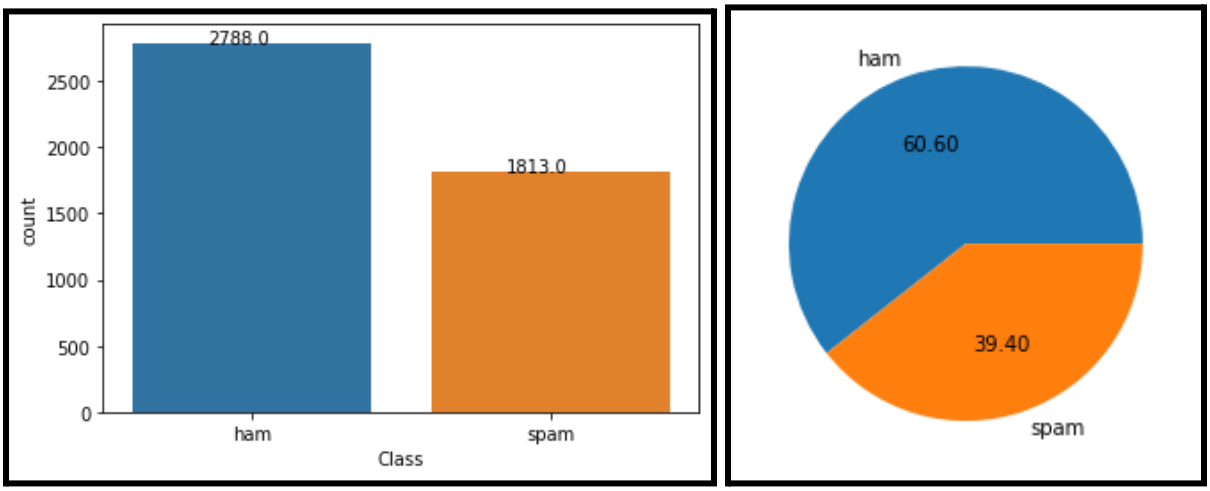


Figure 4 - "CLASS" count

```
data['Class'].describe()

count     4601
unique       2
top        ham
freq      2788
Name: Class, dtype: object
```

Figure 5 - Class Describe

Below I have also put the correlation matrix as the correlation matrix helps to predict the evolution of the relationship between the variables. The correlation matrix allows you to have a global view of the more or less strong relationship between several variables.

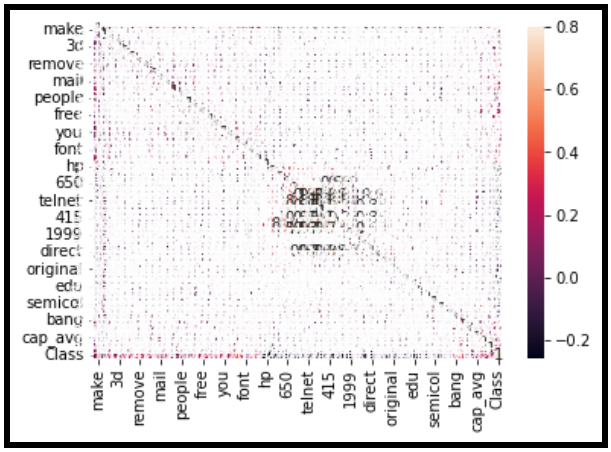
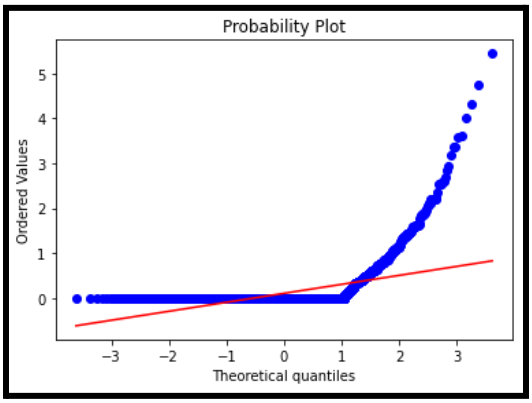


Figure 6 - Correlation Matrix



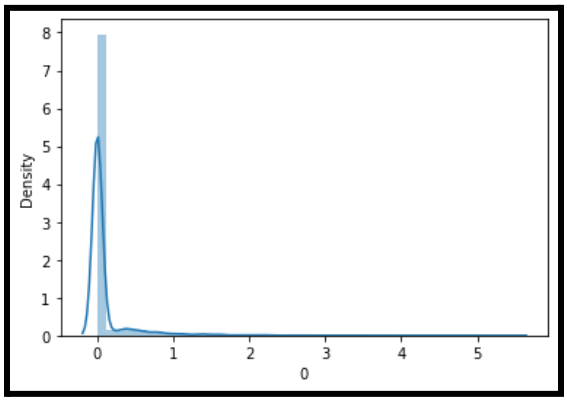


Figure 7 & 8 - probability plots of observed values and density of variables

Datasets that contain duplicates may contaminate the training data with the test data or vice versa. Entries with missing values will lead models to misunderstand features, and outliers will undermine the training process – leading your model to "learn" patterns that do not exist in reality. Given the knowledge gathered previously, we are aware that Random forest would have the maximum model accuracy barring modifications to the data frame itself.

```
data.duplicated().sum()

391
```

Figure 9 - Data Duplicates

REPORTING TASKS

to have 1000 train cases

```
[64] #@title to have 1000 train cases
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.7826, random_state=10, shuffle=False)
     #Compare their accuracies with the following training-test splits: 50%-50%, 60%-40%, 70%-30%, and 80%-20%
```

voting classifier

```
[65] from sklearn.linear_model import LogisticRegression
     from sklearn.naive_bayes import GaussianNB
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.ensemble import RandomForestClassifier, VotingClassifier, AdaBoostClassifier

     #@title voting classifier
     vc = VotingClassifier(estimators=[('lr', lr), ('dt', dt), ('gnb', gnb)], voting='hard')
     vc = vc.fit(X_train, y_train)
     y_pred_vc = vc.predict(X_test)
```

```
[66] lr = LogisticRegression()
     dt = DecisionTreeClassifier()
     gnb = GaussianNB()
```

Random Classifier model

```
[67] #@title Random Classifier model
     rfc_model = RandomForestClassifier()
     rfc_model.fit(X_train,y_train)
     y_pred_rfc = rfc_model.predict(X_test)
```

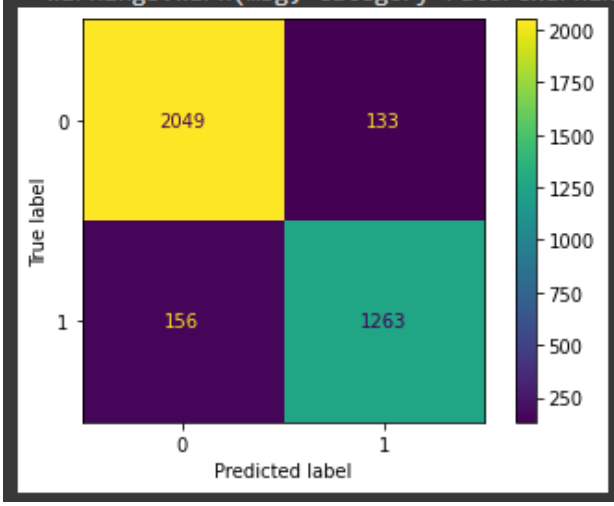
AdaBoost model

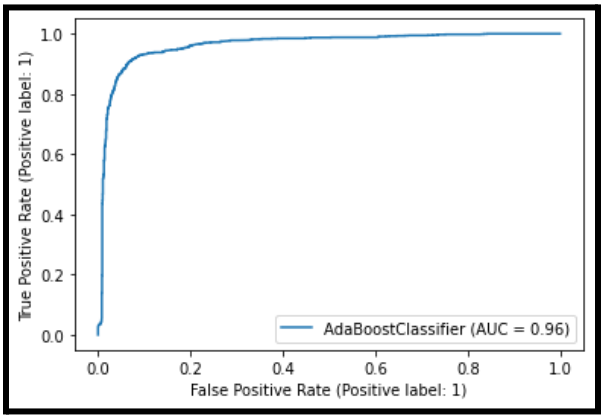
```
[68] #@title AdaBoost model
     ada = AdaBoostClassifier()
     ada.fit(X_train, y_train)
     y_pred_abc = ada.predict(X_test)
```

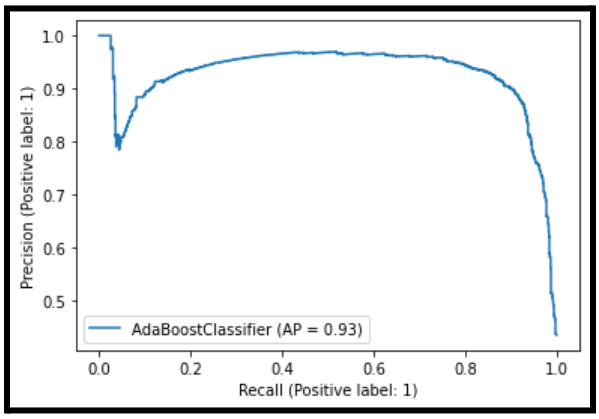
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.7826, random_state=10, shuffle=False)

print("With a test-train split of 78-22%")
print("Adaboost                          accuracy:",accuracy_score(y_test, y_pred_abc))
print("Random Forest Classifier          accuracy:",accuracy_score(y_test, y_pred_rfc))
print("Majority Voting                   accuracy:",accuracy_score(y_test, y_pred_vc))
```

```
With a test-train split of 78-22%
Adaboost                          accuracy: 0.9197445154123854
Random Forest Classifier          accuracy: 0.9316856428769786
Majority Voting                   accuracy: 0.9186337128575396
```

```
report(ada)

0.9197445154123854
              precision    recall  f1-score   support

           0       0.94      0.93      0.93      2205
           1       0.89      0.90      0.90      1396

    accuracy                           0.92      3601
   macro avg       0.91      0.92      0.92      3601
weighted avg       0.92      0.92      0.92      3601

/usr/local/lib/python3.8/dist-packages/sklearn/utils/de
  warnings.warn(msg, category=FutureWarning)
```



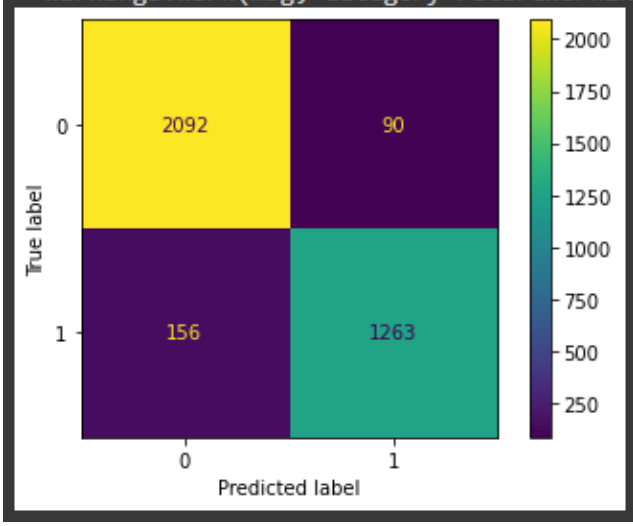


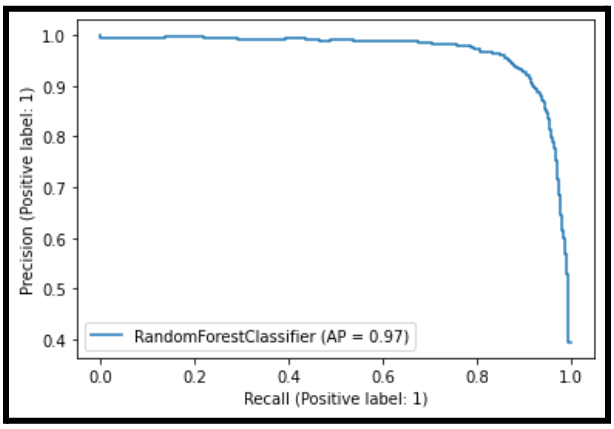


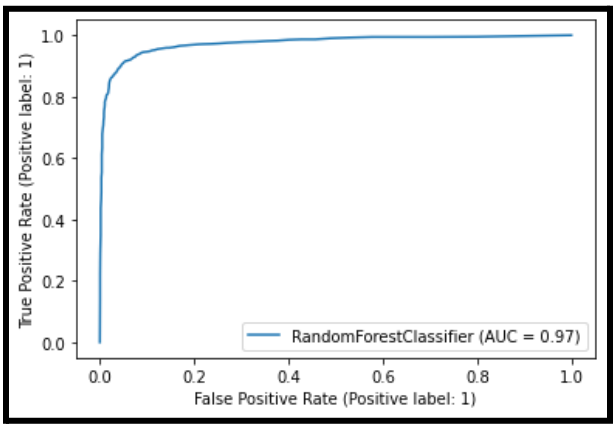
ADABOOST - Reporting
ACCURACY - 91.97%

```
report(rfc_model)
```

```
0.9316856428769786
              precision    recall  f1-score   support

           0       0.96      0.93      0.94      2248
           1       0.89      0.93      0.91      1353

    accuracy                           0.93      3601
   macro avg       0.92      0.93      0.93      3601
weighted avg       0.93      0.93      0.93      3601

/usr/local/lib/python3.8/dist-packages/sklearn/utils/de
  warnings.warn(msg, category=FutureWarning)
```



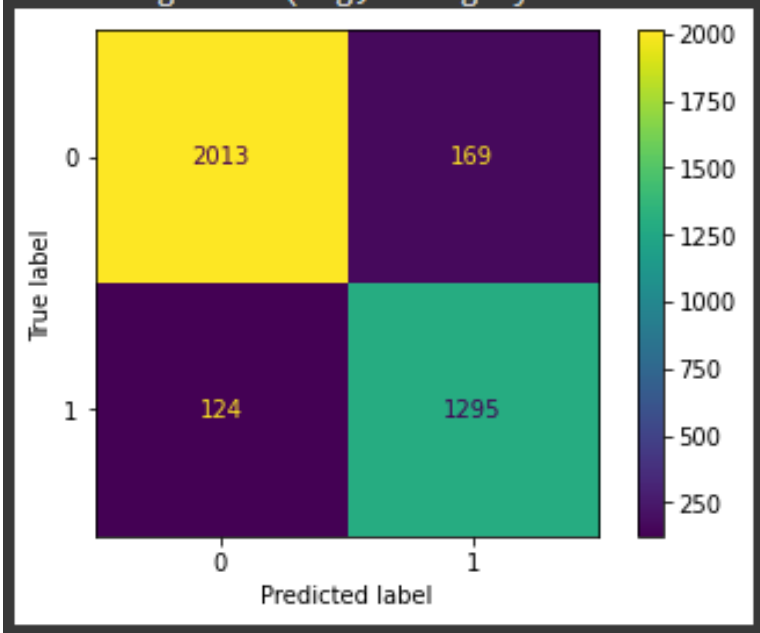




RANDOM FOREST MODEL
ACCURACY - 93.16%

```
report(vc)
```

```
0.9186337128575396
              precision    recall  f1-score   support

           0       0.92      0.94      0.93      2137
           1       0.91      0.88      0.90      1464

    accuracy                           0.92      3601
   macro avg       0.92      0.91      0.92      3601
weighted avg       0.92      0.92      0.92      3601

/usr/local/lib/python3.8/dist-packages/sklearn/utils/de
  warnings.warn(msg, category=FutureWarning)
```



MAJORITY VOTING CLASSIFIER

Fusion of decision tree, gaussian naives bayes & logistic regression

ACCURACY - 91.86%

Accuracy Comparisons Across Different Test-Train Splits

```
With a test-train split of 50.0 - 50.0
Adaboost                          : 0.93133420252064 33
Random Forest Classifier          : 0.9448066058235 55
Majority Voting                   : 0.93220338983050 84
```

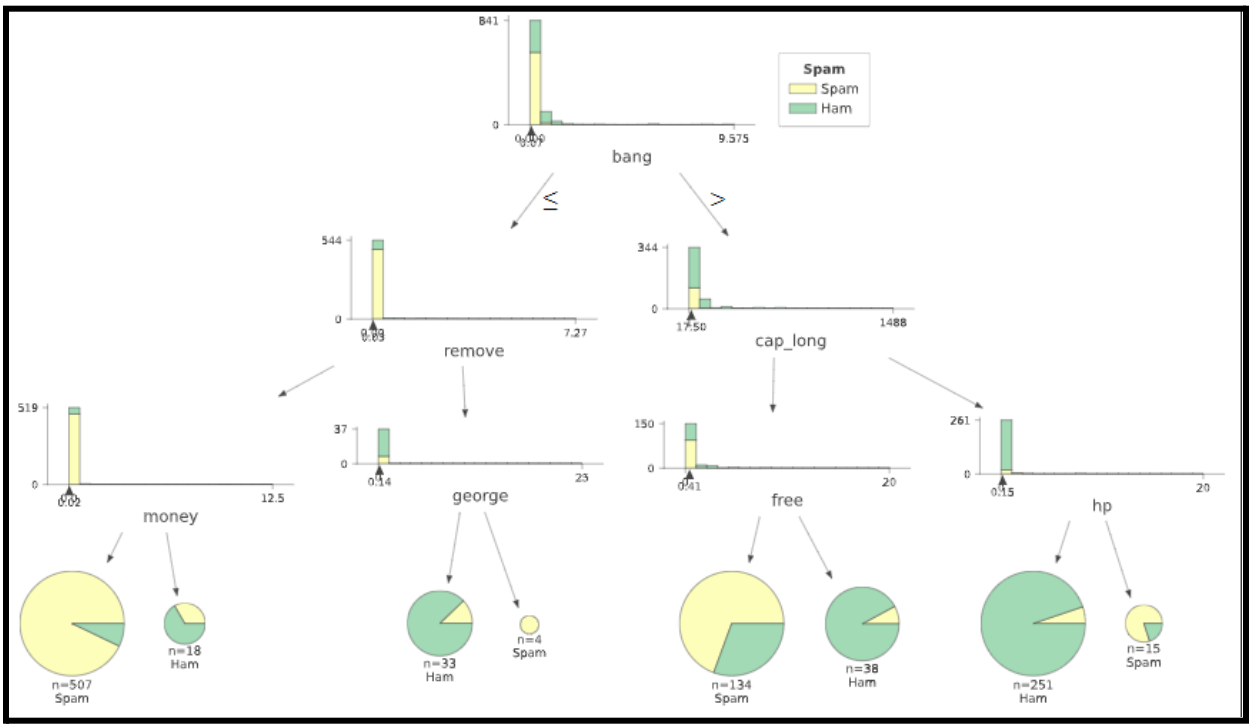
```
With a test-train split of 40.0 - 60.0
Adaboost                          : 0.93644758283541 55
Random Forest Classifier          : 0.94676806083650 19
Majority Voting                   : 0.933188484519283
```

```
With a test-train split of 30.0 - 70.0
Adaboost                          : 0.93772628530050 68
Random Forest Classifier          : 0.95148443157132 51
Majority Voting                   : 0.92831281679942 07
```

```
With a test-train split of 20.0 - 80.0
Adaboost                          : 0.94679695982627 58
Random Forest Classifier          : 0.95439739413680 78
Majority Voting                   : 0.93159609120521 17
```

```
With a test-train split of 10.0 - 90.0
Adaboost                          : 0.95227765726681 12
Random Forest Classifier          : 0.95444685466377 44
Majority Voting                   : 0.93275488069414 31
```

100 Training Instances: with changes to the data frame to suit Adaboost



FEATURE IMPORTANCE

- Doing feature importance we get - bang, remove, cap_long, money, free, george, and hp.
- So we remove these and run all three classifiers again and see the changes.
- Also we will drop all the duplicates in this instance to get a more precise accuracy.

```
data2 = data.drop(columns = ['bang', 'remove', 'cap_long', 'money', 'george', 'free', 'hp'])
data2.drop_duplicates()
Xnew = data2.iloc[:,:-1]
Ynew = data2.iloc[:,:-1]
#X_train, X_test, y_train, y_test = train_test_split(Xnew, Ynew, test_size = 0.7826, random_state=10, shuffle=False)

#voting classifier
vc = VotingClassifier(estimators=[('lr', lr), ('dt', dt), ('gnb', gnb)], voting='hard')
vc = vc.fit(X_train, y_train)
y_pred_vc = vc.predict(X_test)

#Random Classifier Model
rfc_model = RandomForestClassifier()
rfc_model.fit(X_train,y_train)
y_pred_rfc = rfc_model.predict(X_test)

#Adaboost Model
ada = AdaBoostClassifier()
ada.fit(X_train, y_train)
y_pred_abc = ada.predict(X_test)

print("With a test-train split of 78-22%")
print("Adaboost                          accuracy:",accuracy_score(y_test, y_pred_abc))
print("Random Forest Classifier          accuracy:",accuracy_score(y_test, y_pred_rfc))
print("Majority Voting                   accuracy:",accuracy_score(y_test, y_pred_vc))
```

```
With a test-train split of 78-22%
Adaboost                          accuracy: 0.9197445154123854
Random Forest Classifier          accuracy: 0.9297417384059984
Majority Voting                   accuracy: 0.9202999166898084
```

INFERENCE

As per our prediction, the accuracies of the different classifiers was as follows:
Majority Voting Classifier < Adaboost Classifier < Random Forest

Majority Voting Classifier : as the name suggests the majority votes among each individual classifier wins. Two different voting schemes are common among voting classifiers: In hard voting (also known as majority voting), every individual classifier votes for a class, and the majority wins. In statistical terms, the predicted target label of the ensemble is the mode of the distribution of individually predicted labels.

Adaboost Classifier: is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Random Forest: is a meta estimator that fits a number of decision trees on various sub-samples of the dataset, and uses averaging to improve the predictive accuracy and control over-fitting.

In most cases, adaboost tends to outperform random forest in terms of accuracy. However it shows a lower accuracy in this case due to inconsistencies such as duplicates, and outliers within the data.

LINK TO CODE:
https://colab.research.google.com/drive/1Pmfu00mdJvB9yno_lQHqlYiZ0mOXzPqH#scrollTo=sj6UnpP4NqIl

---