Syracuse University

CIS 600 - Wine Quality Prediction using Machine Learning Algorithms

Yuexin Yu,
Jingwen Ling,
Vaidy Krish

College of Engineering & Computer Science

Syracuse University

Project Introduction and Objective

Wine

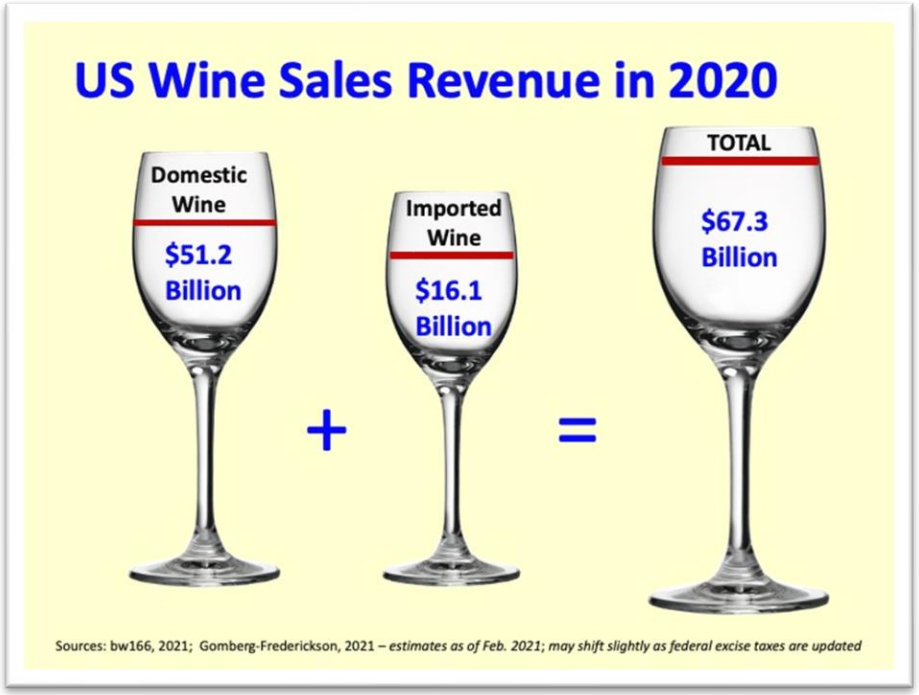
Wine is an alcoholic beverage made with the fermented juice of grapes

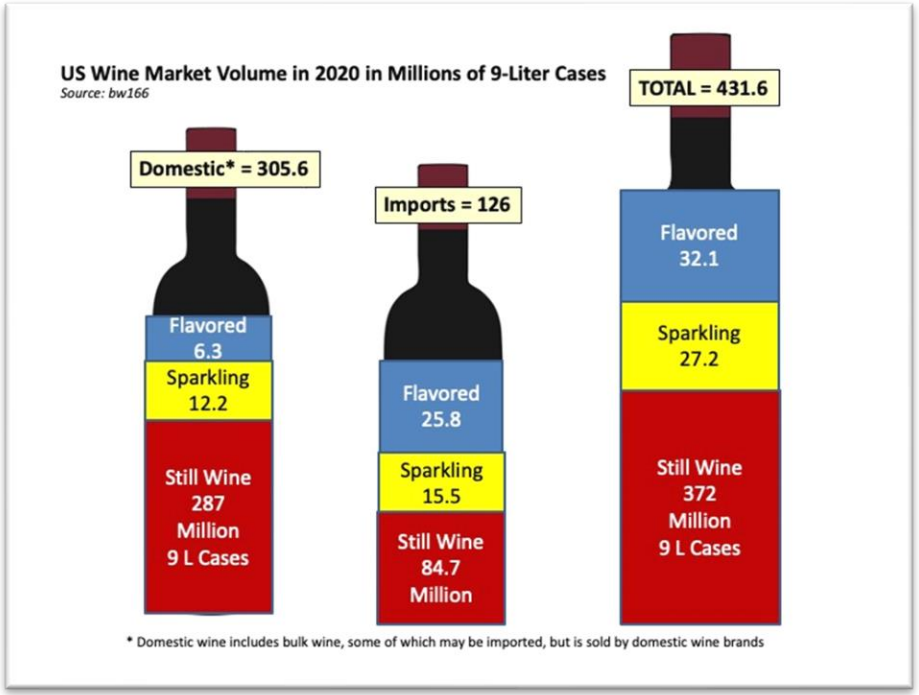
Four Types of Wine

- Red wine
- White wine
- Rose wine
- Sparkling wines/champagne.

Wine Market in the U.S.

- The USA is the world's largest wine market, with the largest population of regular wine drinkers.
- In 2020, the USA market consumed 126 million cases of imported grape wine valued at nearly $US 16 billion.





Objective

- Explore and visualize the dataset :
- To predict the quality of wine on a scale of 0 to 10.
- Which variables are most significant.
- Build using various machine learning models predict …..
- Generate a set of insights and recommendations that will help the business to understand which wine has the highest quality.

Data Information

Attributes' information about the datasets (winequality-red.csv and winequality-white.csv)

Input variables (based on physicochemical tests):

- 1 - fixed acidity
- 2 - volatile acidity
- 3 - citric acid
- 4 - residual sugar
- 5 - chlorides
- 6 - free sulfur dioxide
- 7 - total sulfur dioxide
- 8 - density
- 9 - pH
- 10 - sulphates
- 11 – alcohol

Output variable (based on sensory data):

- 12 - quality (score between 0 and 10)

Syracuse University

Data Preparation Steps Involved

Data Preparation & Exploratory Data Analysis

- The maximum rating of wine is 9 and the third quartile value is 6 i.e. at least 75% of wines have rating 6 or below.
- There might be outliers in data where the wines have high sulfur dioxide (free sulfur dioxide and total sulfur dioxide) content as there is a big difference in 3rd quartile and maximum values.
- Most wines are on a pH scale between 3 and 4.

```
red_train_data = pd.read_csv('winequality-red.csv', low_memory=False, sep=';')
print("Size of the red-wine training dataset: ", red_train_data.shape)
red_train_data.head()
```

```
Size of the red-wine training dataset:  (1599, 12)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

```
white_train_data = pd.read_csv('winequality-white.csv', low_memory=False, sep=';')
print("Size of the white-wine training dataset: ", white_train_data.shape)
white_train_data.head()
```

```
Size of the white-wine training dataset:  (4898, 12)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.0010	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.9940	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.9951	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.9956	3.19	0.40	9.9	6

Data Preprocessing and Cleaning
- 1, Check and Remove Duplicates

- We found 240 duplicated rows in the red wine dataset and 973 duplicated rows in the white wine dataset.
- We will drop them out of the training dataset.

Check Duplicates:

```
red_train_data.duplicated().sum()
```

```
240
```

```
print(red_train_data.shape)
```

```
(1599, 12)
```

There are 240 duplicated rows in the training dataset. Let's drop them.

```
white_train_data.duplicated().sum()
```

```
937
```

```
print(white_train_data.shape)
```

```
(4898, 12)
```

There are 937 duplicated rows in the training dataset. Let's drop them.

Data Preprocessing and Cleaning
- 2, Check Missing Data

- There are no missing data rows in both data set.

```
display(red_train_data.isnull().any())
```

```
fixed acidity           False
volatile acidity        False
citric acid             False
residual sugar          False
chlorides               False
free sulfur dioxide     False
total sulfur dioxide    False
density                 False
pH                      False
sulphates               False
alcohol                 False
quality                 False
dtype: bool
```

```
display(white_train_data.isnull().any())
```

```
fixed acidity           False
volatile acidity        False
citric acid             False
residual sugar          False
chlorides               False
free sulfur dioxide     False
total sulfur dioxide    False
density                 False
pH                      False
sulphates               False
alcohol                 False
quality                 False
dtype: bool
```

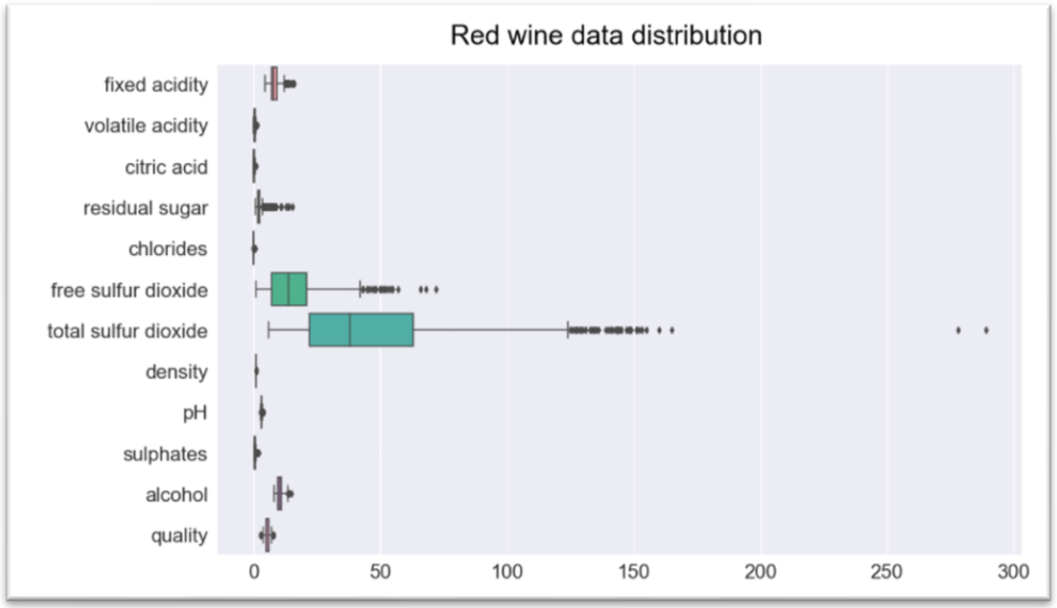
Data Preprocessing and Cleaning
- 3, Check and Remove Outliers

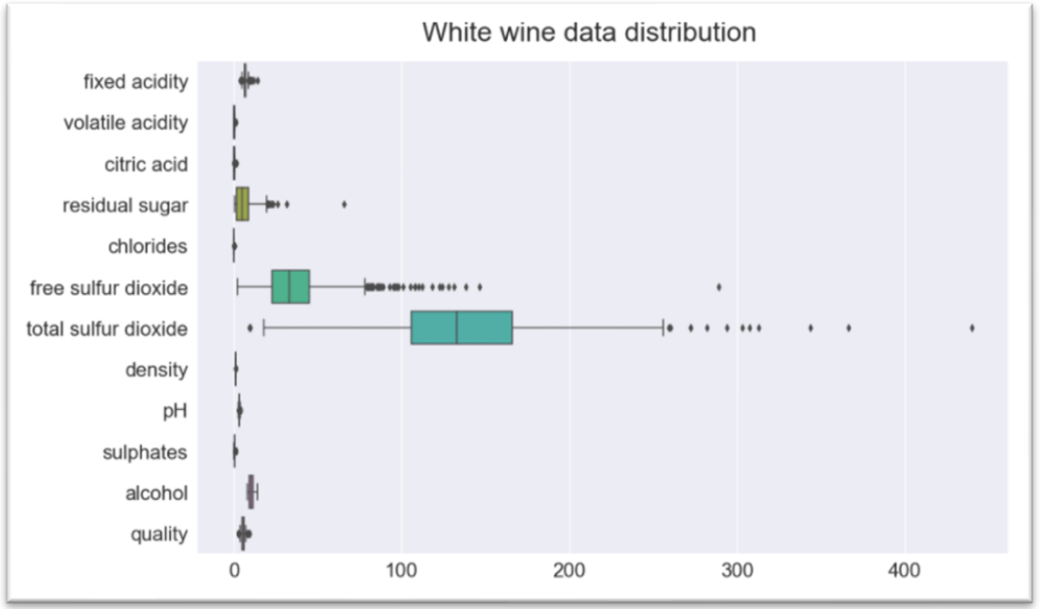
Red wine data set:

- Remove outliers on 'free sulfur dioxide' and 'total sulfur dioxide' columns.

White wine data set:

- Remove outliers on 'residual sugar', 'free sulfur dioxide' and 'total sulfur dioxide' columns.





Data transformation and preparation
- Explore Different Data Types

Discrete Type of Data

```
quality : [5 6 7 4 8 3]
```

Continuous Type of Data

```
fixed acidity : Minimum: 4.6, Maximum: 15.9
volatile acidity : Minimum: 0.12, Maximum: 1.58
citric acid : Minimum: 0.0, Maximum: 1.0
residual sugar : Minimum: 0.9, Maximum: 15.5
chlorides : Minimum: 0.012, Maximum: 0.611
free sulfur dioxide : Minimum: 1.0, Maximum: 72.0
total sulfur dioxide : Minimum: 6.0, Maximum: 175.0
density : Minimum: 0.9900700000000001, Maximum: 1.00369
pH : Minimum: 2.74, Maximum: 4.01
sulphates : Minimum: 0.33, Maximum: 2.0
alcohol : Minimum: 8.4, Maximum: 14.9
```

Categorical Type of Data

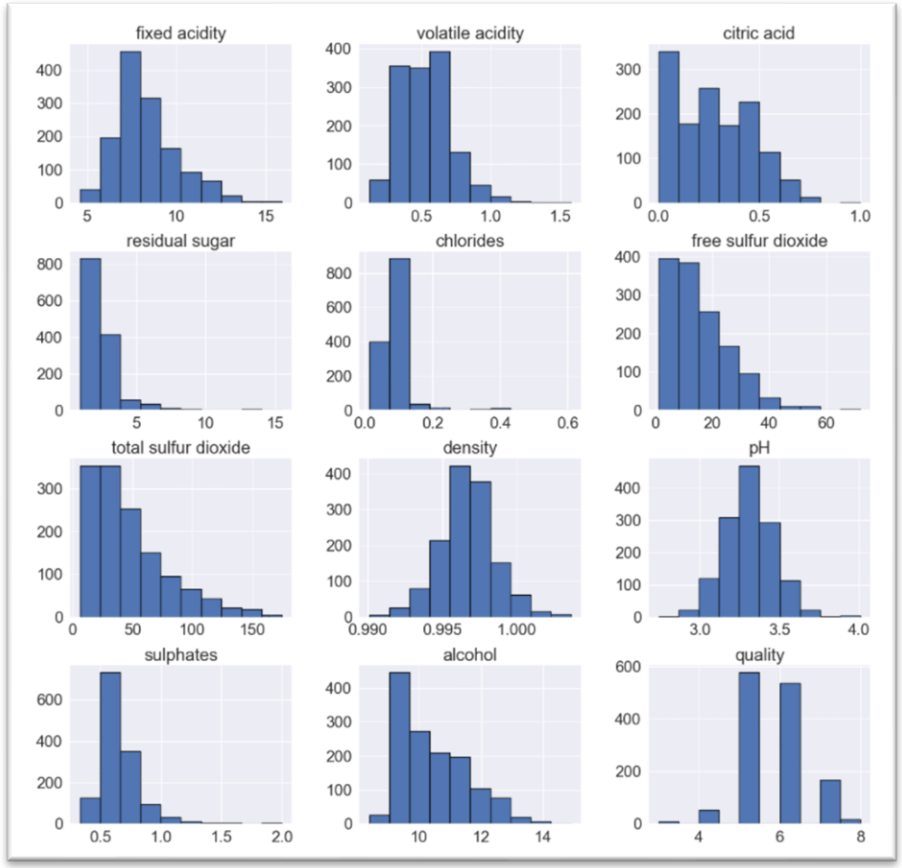
There is no categorical type of data in these two datasets.

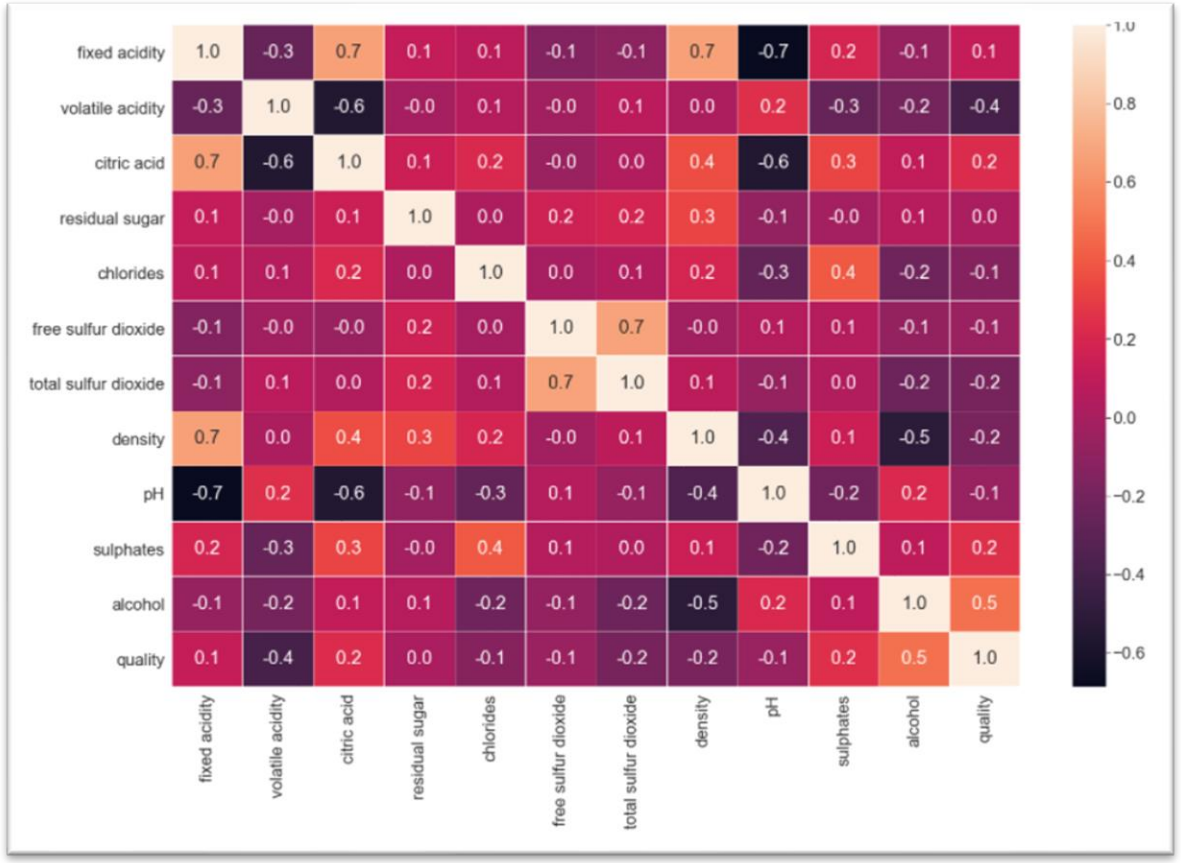
Syracuse University

Exploratory Data Analysis and Key Insights

Exploratory Data Analysis (EDA):
- Explore Numerical Features – Red wine dataset

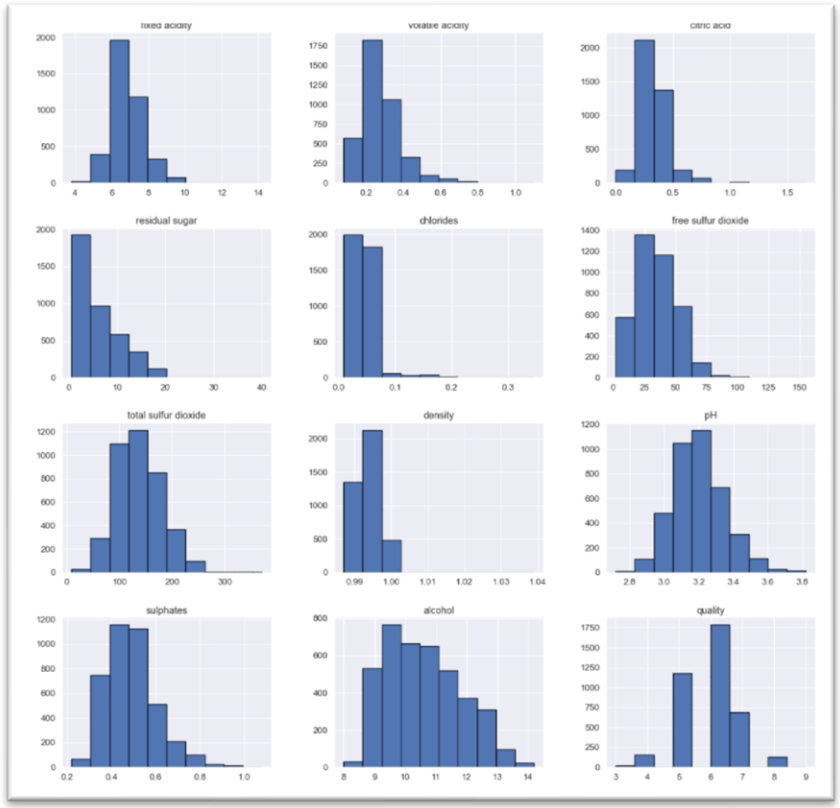
- From the heatmap, we can see the high negative correlation on volatile acidity, total sulfur dioxide, density with red wine quality, and high positive correlation on citric acid, sulphates, alcohol with red wine quality.

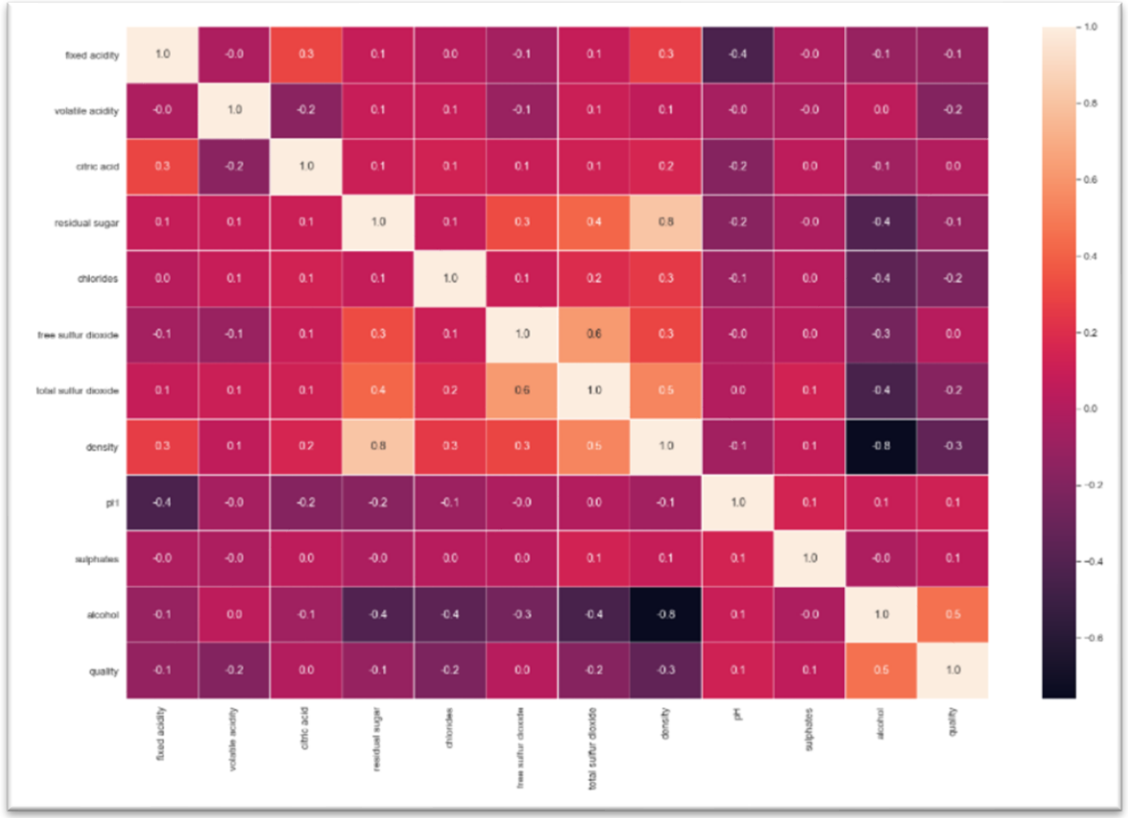




Exploratory Data Analysis (EDA):
- Explore Numerical Features – White wine dataset

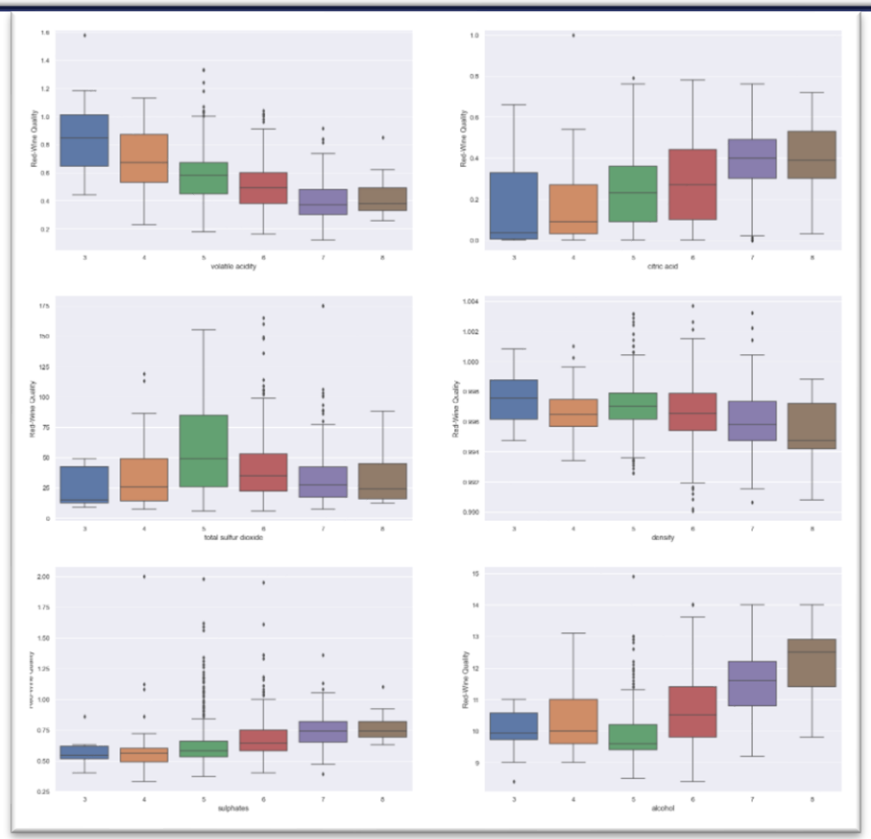
- From the heatmap, we can see the high negative correlation on volatile acidity, chlorides, total sulfur dioxide, density with white wine quality, and high positive correlation on alcohol with white wine quality.
- Fixed acidity has a strong positive correlation with citric acid and density.
- The total sulfur dioxide and free sulfur dioxide have a strong correlation.
- The quality of wine shows a moderate correlation with alcohol.

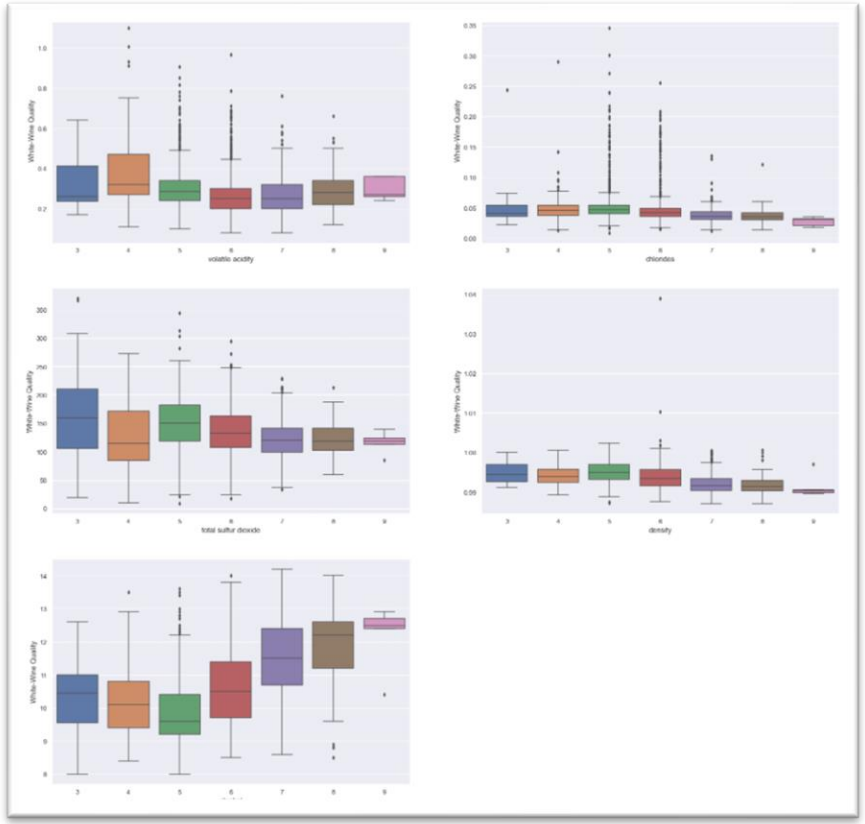




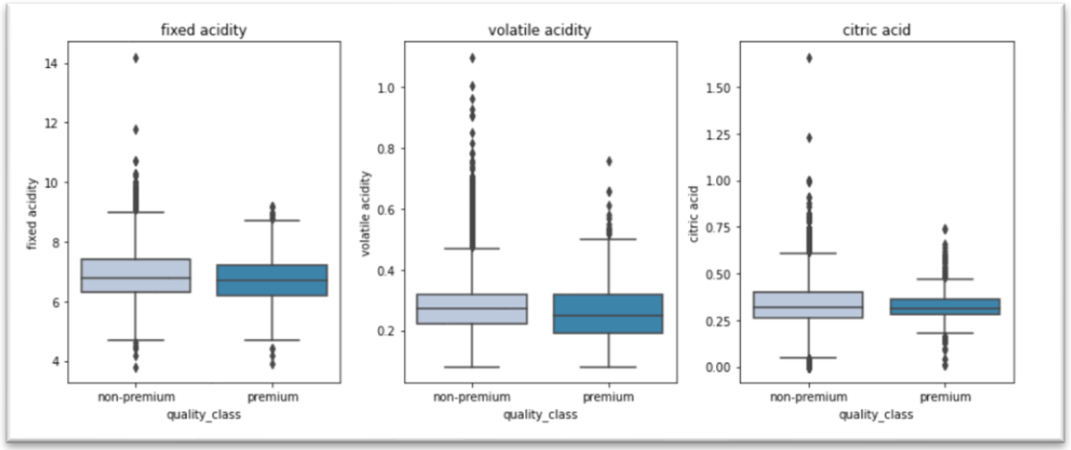
Exploratory Data Analysis (EDA):
- Data Insight – Suggestion for manufacturer

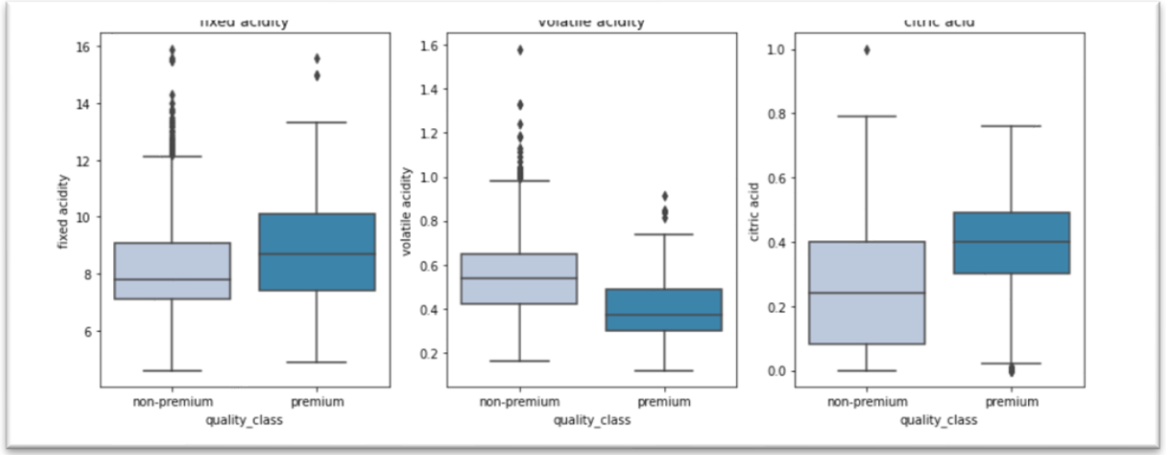
- **Red wine:** we can see increasing on volatile acidity, total sulfur dioxide, density will reduce the red wine quality, increasing on citric acid, sulphates, alcohol will improve the red wine quality.
- **White wine:** we can see increasing on volatile acidity, chlorides, total sulfur dioxide, density will reduce the white wine quality, but increasing on alcohol will improve the white wine quality.

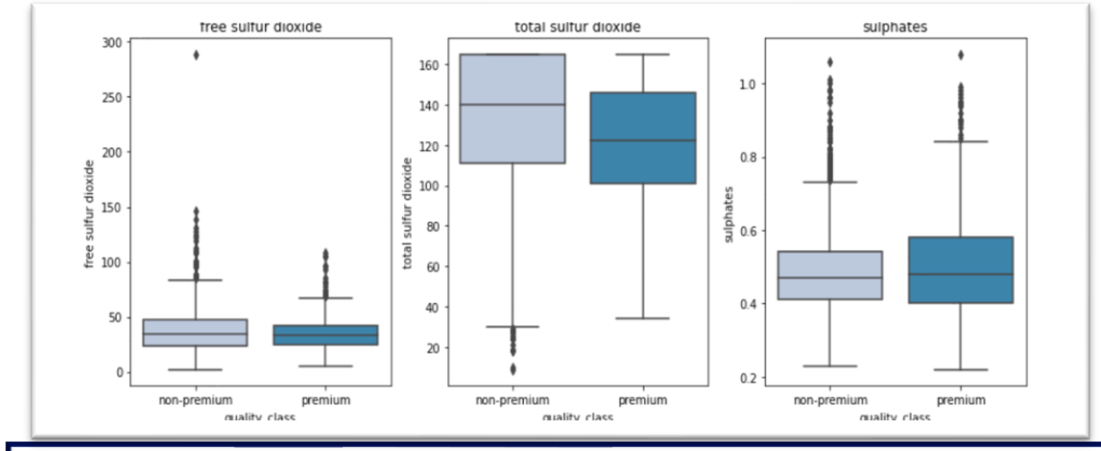


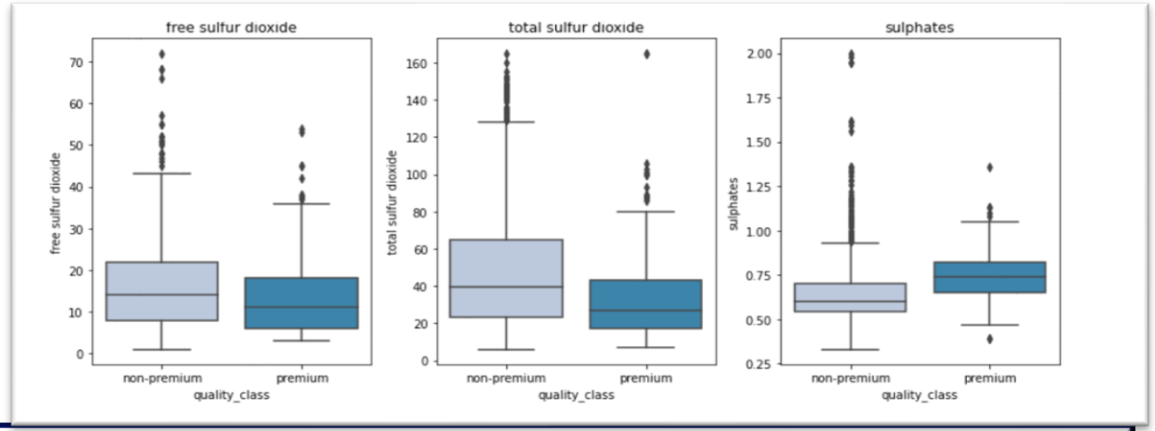


Exploratory Data Analysis (EDA):
- Data Insight – Quality vs Acidity / Sulphur









- Premium wines have higher fixed acidity and citric acid and lower volatile acidity.
- Premium wines have less concentration of free sulfur dioxide and total sulfur dioxide but higher sulphates that helps in preserving them for a longer period.

Syracuse University

Model Building Steps and Criteria's

Model Building Steps

- Data preparation
- Partition the data into train and test set.
- Build model on the train data.
- Tune the model if required.
- Test the data on test set.
- Model evaluation criterion
- Model can make wrong predictions as:
- Which case is more important?

Model Evaluation Criteria

Model can make wrong predictions as:

- Predicting a wine is of premium quality when it is of non-premium quality.
- Predicting a wine is of non-premium quality when it is of premium quality.

Which case is more important?

- 1. If the model predicts a wine is of non-premium quality but it is of premium quality then the company would incur loss of good wine and resources used.
- 2. If the model predicts a wine is of premium quality but it is not then the company would roll out low-quality wine which would affect their customer base and their reputation.

Which metric to optimize?

- We would want F1-Score to be maximized, the greater the F1-Score higher the chances of predicting both the classes correctly.

Syracuse University

Machine Learning Models

Model Learning Models Researched and Evaluated

- K-Nearest Neighbor
- Naïve Bayes
- Support Vector Machines
- Decision Tree
- Random Forest
- Boosting Classifier
- Ada Boost
- Gradient Boost
- XG Boost
- Ensemble Learning / Stacking classifier
- Hyperparameter tuning (Grid Search and Random Search)

Syracuse University

K-Nearest Neighbor

K Nearest Neighbor Algorithm
- Model baseline performance

- Categorical data encoding: quality column
- Split train and test data into 7:3 for model evaluation
- Apply standard scaling
- Fit default model to train data
- Baseline performance:

Accuracy: 66%

	precision	recall	f1-score	support
0	0.66	0.60	0.62	191
1	0.67	0.72	0.70	217
accuracy			0.66	408
macro avg	0.66	0.66	0.66	408
weighted avg	0.66	0.66	0.66	408

Test Accuracy : 0.664
Training Accuracy : 0.810

		Predict Stay	Predict Leave
Real	Stay	114	77
	Leave	60	157

K Nearest Neighbor Algorithm
- Hyperparameter tuning with GridSearchCV

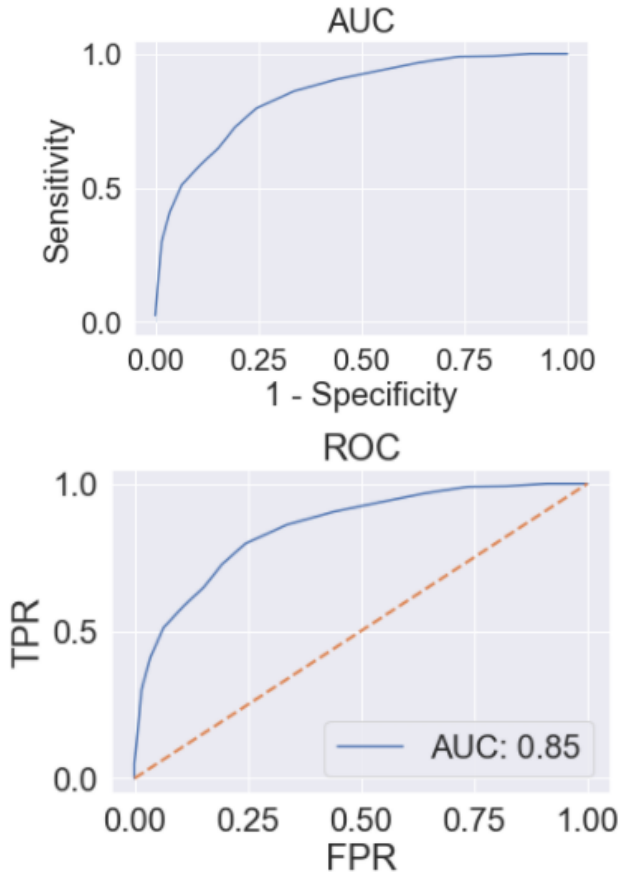
The best accuracy after grid search is 75.8%
with metric='manhattan', n_neighbors=17, and weights='uniform'.

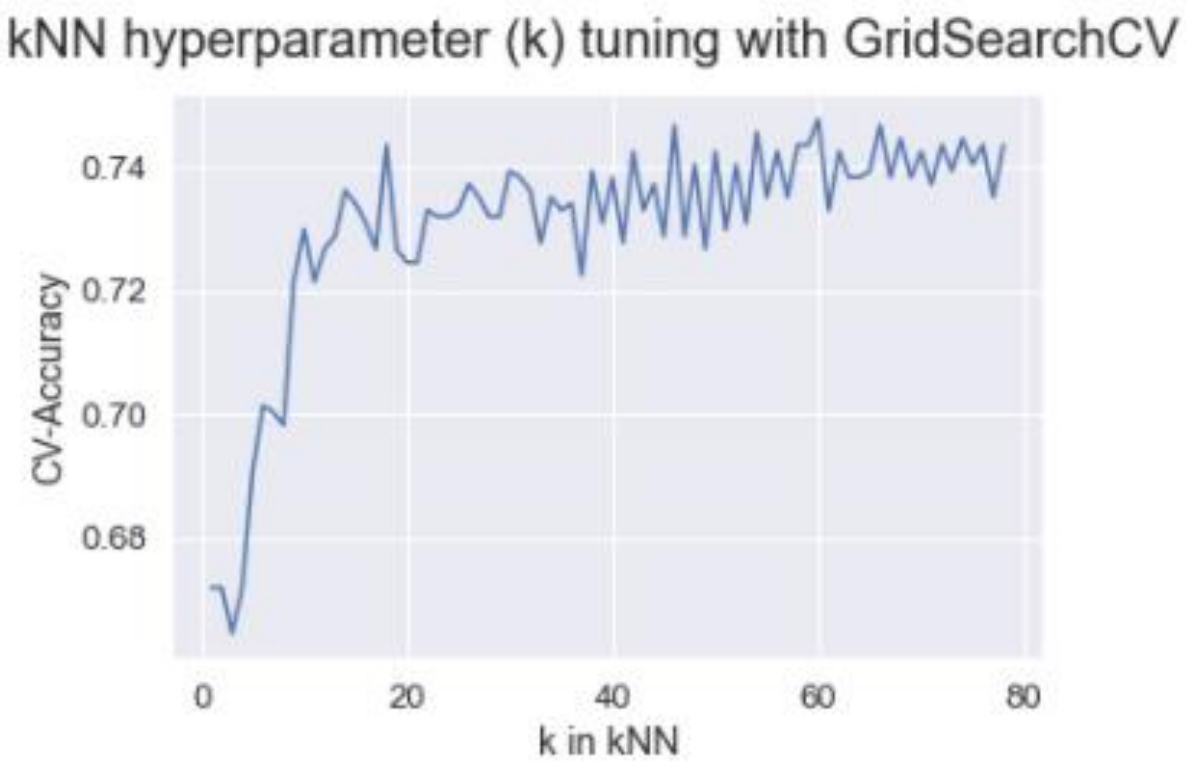
```
params = {
    'n_neighbors': np.arange(1, 40),
    'weights': ['uniform', 'distance'],
    'metric': ['euclidean', 'manhattan']
}

knn_grid = GridSearchCV(knn, params, n_jobs=-1, cv=5, verbose=5)
knn_grid.fit(X_train, y_train)

print('Train Accuracy : %.3f'%knn_grid.best_estimator_.score(X_train, y_train))
print('Test Accuracy : %.3f'%knn_grid.best_estimator_.score(X_test, y_test))
print('Best Accuracy Through Grid Search : %.3f'%knn_grid.best_score_)
print('Best Parameters : ',knn_grid.best_params_)
```

```
Fitting 5 folds for each of 156 candidates, totalling 780 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  48 tasks      | elapsed:   10.6s
[Parallel(n_jobs=-1)]: Done 276 tasks      | elapsed:   12.1s

Train Accuracy : 0.777
Test Accuracy : 0.703
Best Accuracy Through Grid Search : 0.758
Best Parameters :  {'metric': 'manhattan', 'n_neighbors': 17, 'weights': 'uniform'}

[Parallel(n_jobs=-1)]: Done 780 out of 780 | elapsed:   14.7s finished
```

K Nearest Neighbor Algorithm
- Model Performance Evaluation

AUC (Area Under Curve)
AUC = 0.8530621389719518





Syracuse University

Naïve Bayes

Naive Bayes Classifier Algorithm

- With estimator ***BernoulliNB***

- Model baseline performance

- Categorical data encoding: quality column
- Split train and test data into 7:3 for model evaluation
- Apply standard scaling
- Fit default model to train data
- Baseline performance:

Accuracy: 69%

	precision	recall	f1-score	support
0	0.65	0.71	0.68	191
1	0.72	0.67	0.70	217
accuracy			0.69	408
macro avg	0.69	0.69	0.69	408
weighted avg	0.69	0.69	0.69	408

Test Accuracy : 0.689
Training Accuracy : 0.744

		Predict Stay	Predict Leave
Real	Stay	136	55
	Leave	72	145

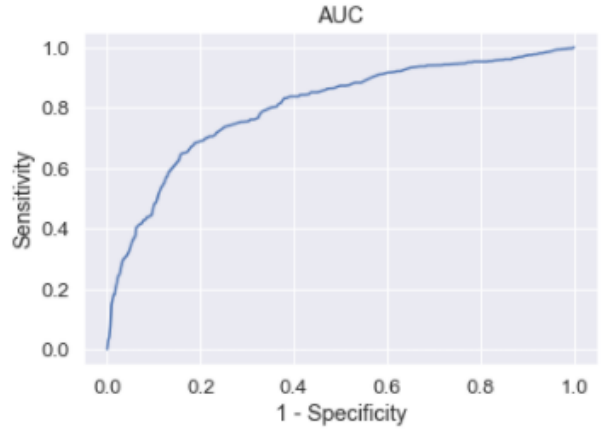
Naive Bayes Classifier Algorithm

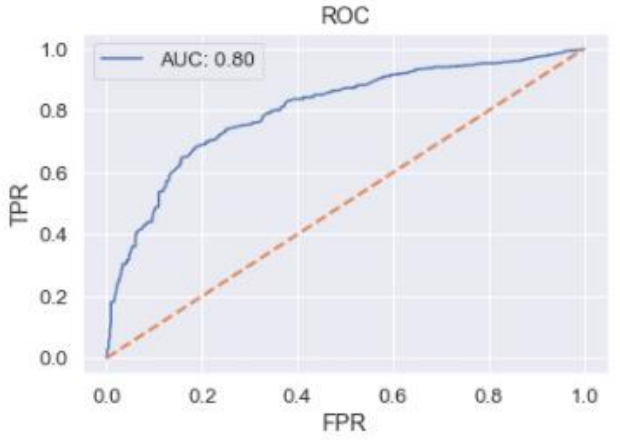
- With estimator ***BernoulliNB***
- Hyperparameter tuning and Evaluation

We can get the best accuracy 72.8% with the best parameter "alpha" 0.01.

```
Fitting 5 folds for each of 6 candidates, totalling 30 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
Train Accuracy : 0.744
Test Accuracy : 0.689
Best Accuracy Through Grid Search : 0.728
Best Parameters :  {'alpha': 0.01}
[Parallel(n_jobs=-1)]: Done  14 out of  30 | elapsed:   10.6s remaining:   12.2s
[Parallel(n_jobs=-1)]: Done  21 out of  30 | elapsed:   10.7s remaining:    4.5s
[Parallel(n_jobs=-1)]: Done  28 out of  30 | elapsed:   10.7s remaining:    0.7s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:   10.7s finished
```

AUC = 0.8021078270437181



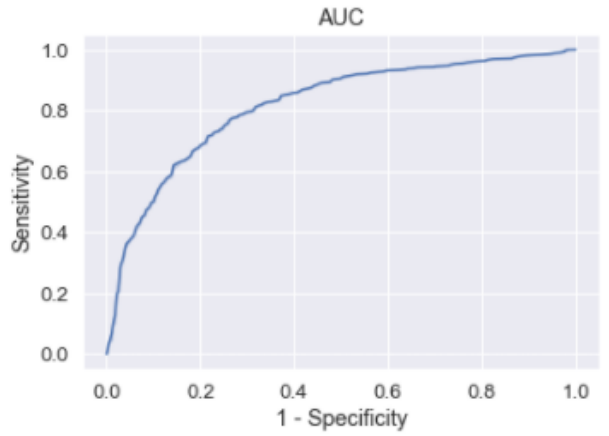


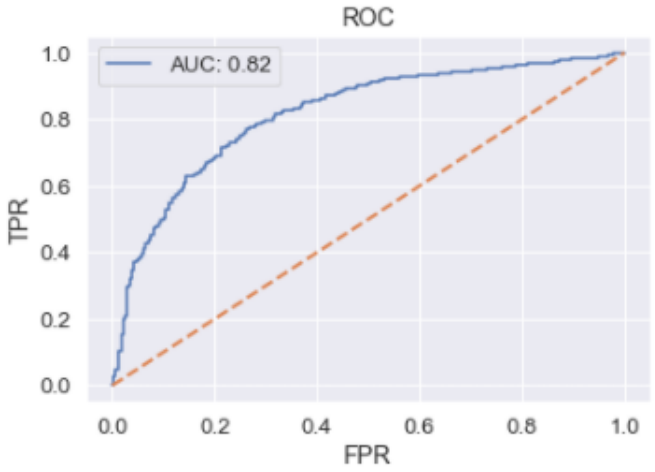
Naive Bayes Classifier Algorithm

- With estimator ***GaussianNB***
- Hyperparameter tuning and Evaluation

```
Accuracy: 70%
              precision    recall  f1-score   support

           0       0.68      0.67      0.68       191
           1       0.71      0.72      0.72       217

    accuracy                           0.70       408
   macro avg       0.70      0.70      0.70       408
weighted avg       0.70      0.70      0.70       408

Test Accuracy : 0.699
Training Accuracy : 0.754

Fitting 5 folds for each of 100 candidates, totalling 500 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  62 tasks       | elapsed:    2.0s
[Parallel(n_jobs=-1)]: Done 242 tasks       | elapsed:    6.0s
[Parallel(n_jobs=-1)]: Done 477 out of 500 | elapsed:   10.4s remaining:    0.4s

Train Accuracy : 0.751
Test Accuracy : 0.691
Best Accuracy Through Grid Search : 0.746
Best Parameters :  {'var_smoothing': 0.23101297000831597}

[Parallel(n_jobs=-1)]: Done 500 out of 500 | elapsed:   11.1s finished
```

		Predict	
		Stay	Leave
Real	Stay	128	63
	Leave	60	157





We can get the best accuracy 74.6% with the best parameter "var_smoothing"=0.23101297000831597.

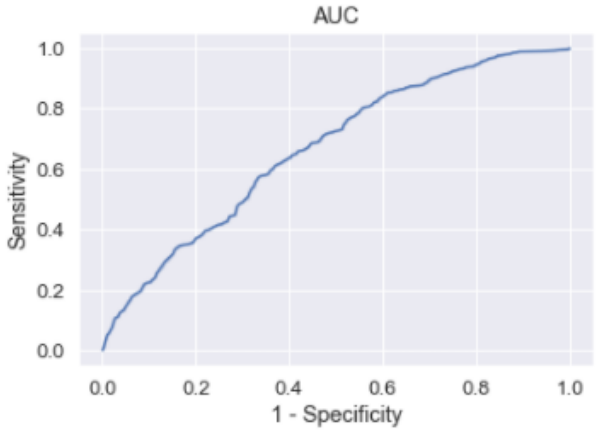
Naive Bayes Classifier Algorithm

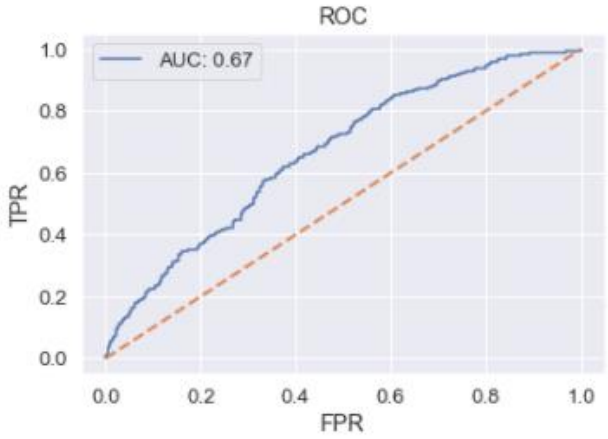
- With estimator ***ComplementNB***

- Hyperparameter tuning and Evaluation

```
Accuracy: 63%
              precision    recall  f1-score   support

           0       0.64      0.47      0.54       191
           1       0.62      0.77      0.69       217

    accuracy                           0.63       408
   macro avg       0.63      0.62      0.61       408
weighted avg       0.63      0.63      0.62       408

Test Accuracy : 0.627
Training Accuracy : 0.616

Fitting 5 folds for each of 6 candidates, totalling 30 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.

Train Accuracy : 0.616
Test Accuracy : 0.627
Best Accuracy Through Grid Search : 0.616
Best Parameters :  {'alpha': 0.01}

[Parallel(n_jobs=-1)]: Done  14 out of  30 | elapsed:   10.5s remaining:   12.1s
[Parallel(n_jobs=-1)]: Done  21 out of  30 | elapsed:   10.6s remaining:    4.5s
[Parallel(n_jobs=-1)]: Done  28 out of  30 | elapsed:   10.6s remaining:    0.7s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:   10.7s finished
```

		Predict	
		Stay	Leave
Real	Stay	136	55
	Leave	72	145





We can get the best accuracy 61.6% with the best parameter "alpha" 0.01.

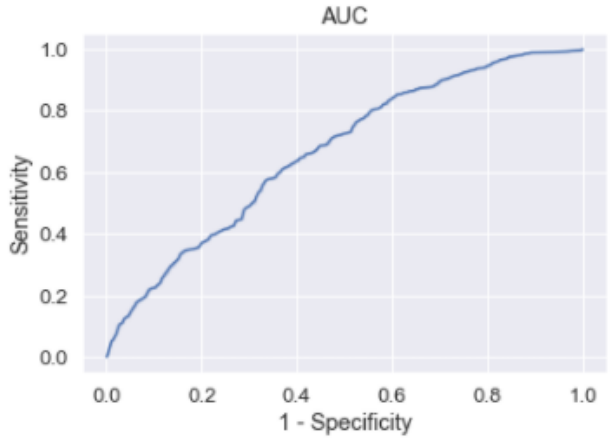
Naive Bayes Classifier Algorithm

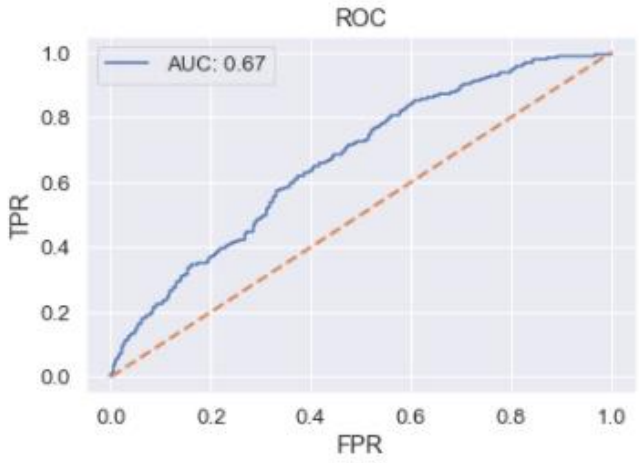
- With estimator ***MultinomialNB***

- Hyperparameter tuning and Evaluation

```
Accuracy: 63%
              precision    recall  f1-score   support

           0       0.65      0.46      0.54       191
           1       0.62      0.78      0.69       217

    accuracy                           0.63       408
   macro avg       0.63      0.62      0.61       408
weighted avg       0.63      0.63      0.62       408

Test Accuracy : 0.630
Training Accuracy : 0.623

Fitting 5 folds for each of 19 candidates, totalling 95 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.

Train Accuracy : 0.623
Test Accuracy : 0.630
Best Accuracy Through Grid Search : 0.621
Best Parameters :  {'alpha': 0.4}

[Parallel(n_jobs=-1)]: Done  71 tasks      | elapsed:    0.3s
[Parallel(n_jobs=-1)]: Done  72 out of  95 | elapsed:    0.3s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done  92 out of  95 | elapsed:    0.3s remaining:    0.0s
[Parallel(n_jobs=-1)]: Done  95 out of  95 | elapsed:    0.3s finished
```

		Predict	
		Stay	Leave
Real	Stay	136	55
	Leave	72	145





We can get the best accuracy 62.1% with the best parameter "alpha" 0.4.

Syracuse University

Support Vector Machines

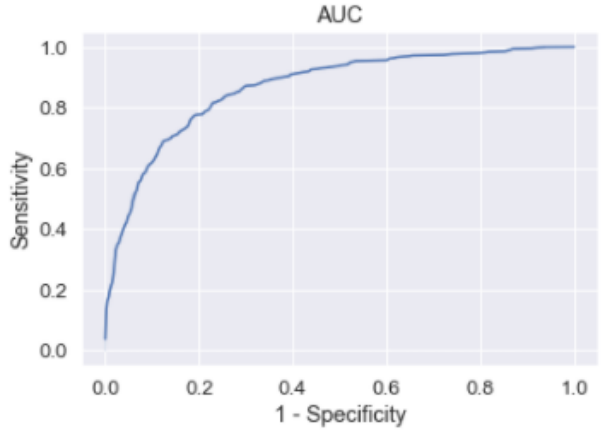
Support Vector Machines Algorithm
- Hyperparameter tuning and Evaluation

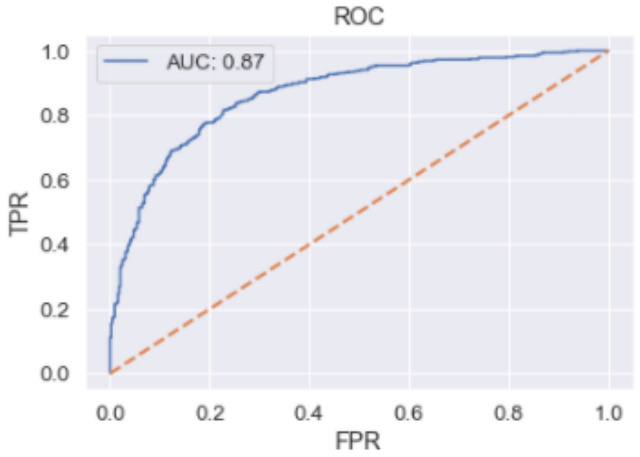
```
Accuracy: 72%
              precision    recall  f1-score   support

           0       0.70      0.71      0.70       191
           1       0.74      0.74      0.74       217

    accuracy                           0.72       408
   macro avg       0.72      0.72      0.72       408
weighted avg       0.72      0.72      0.72       408

Test Accuracy : 0.723
Training Accuracy : 0.804
```

		Predict Stay	Predict Leave
Real	Stay	135	56
	Leave	57	160

```
Fitting 5 folds for each of 25 candidates, totalling 125 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 concurrent workers.
[Parallel(n_jobs=-1)]: Done  48 tasks      | elapsed:   12.7s
Train Accuracy : 0.787
Test Accuracy : 0.718
Best Accuracy Through Grid Search : 0.759
Best Parameters :  {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
[Parallel(n_jobs=-1)]: Done 125 out of 125 | elapsed:   18.3s finished
```

The best accuracy after grid search is 75.9% with parameters 'C': 10, 'gamma': 0.01, 'kernel': 'rbf'.



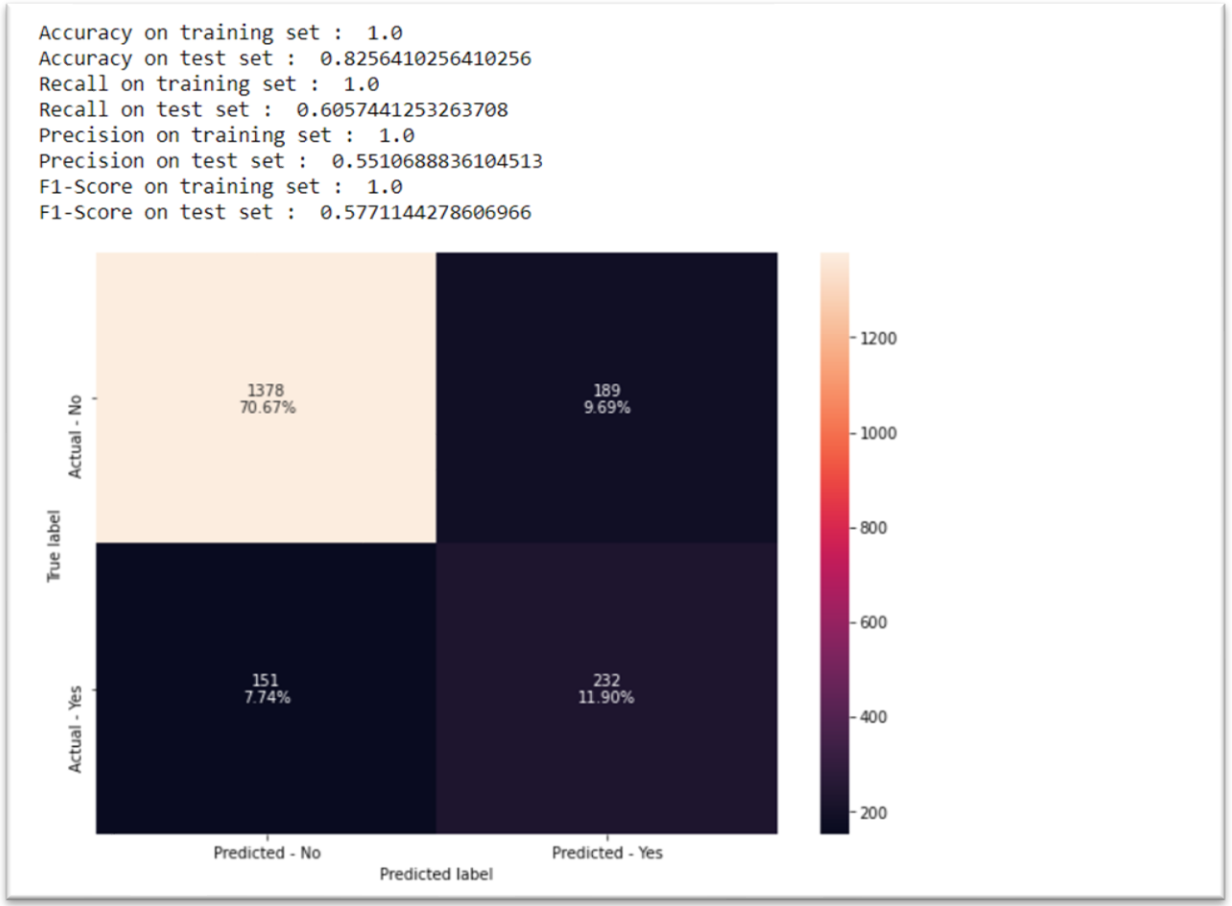


Syracuse University

Decision Tree

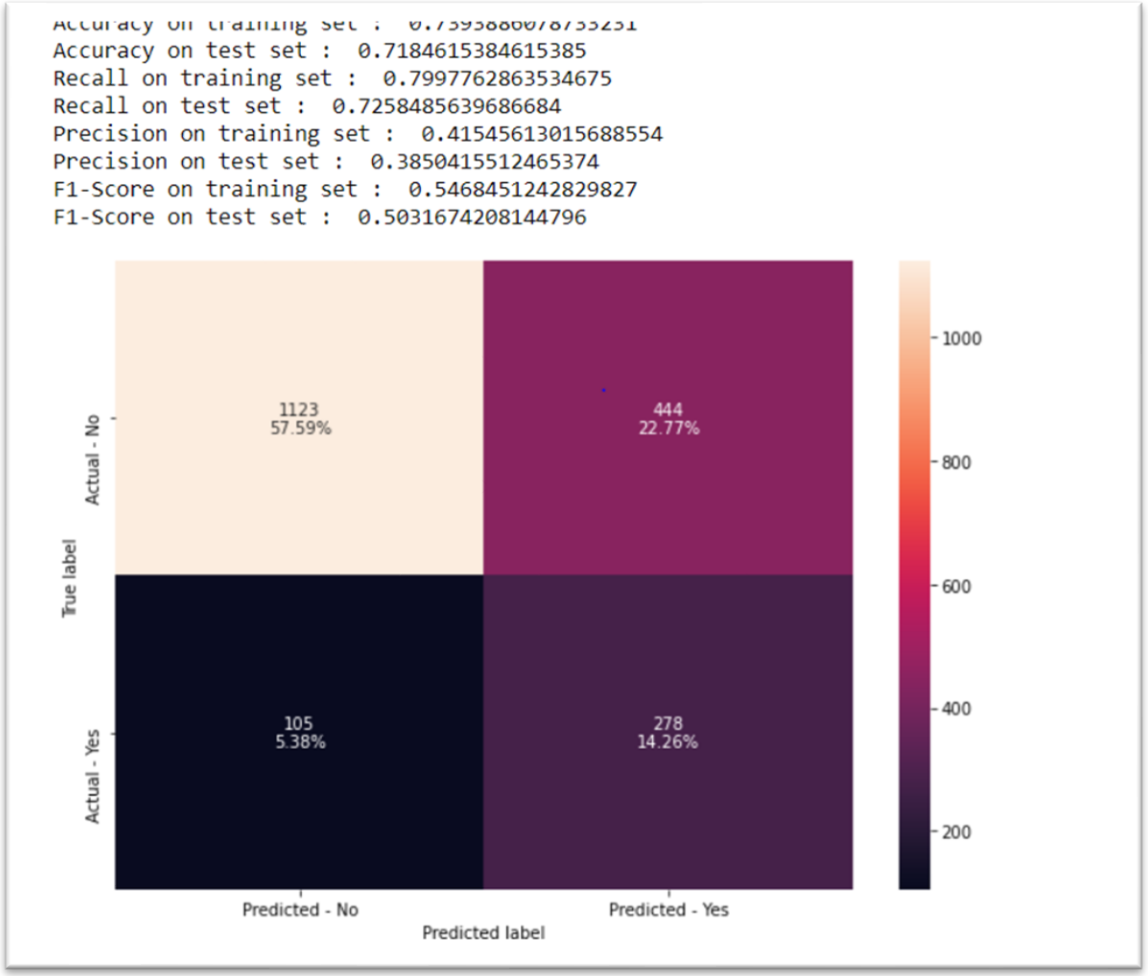
Decision Tree Classifier Algorithm
- Model baseline performance

- Decision tree is overfitting the training data.
- Let's try hyperparameter tuning and see if the model performance improves.



Decision Tree Classifier Algorithm
- Hyperparameter tuning and Evaluation

- The overfitting has reduced but the test f1-score has also decreased.
- Let's try some other models.

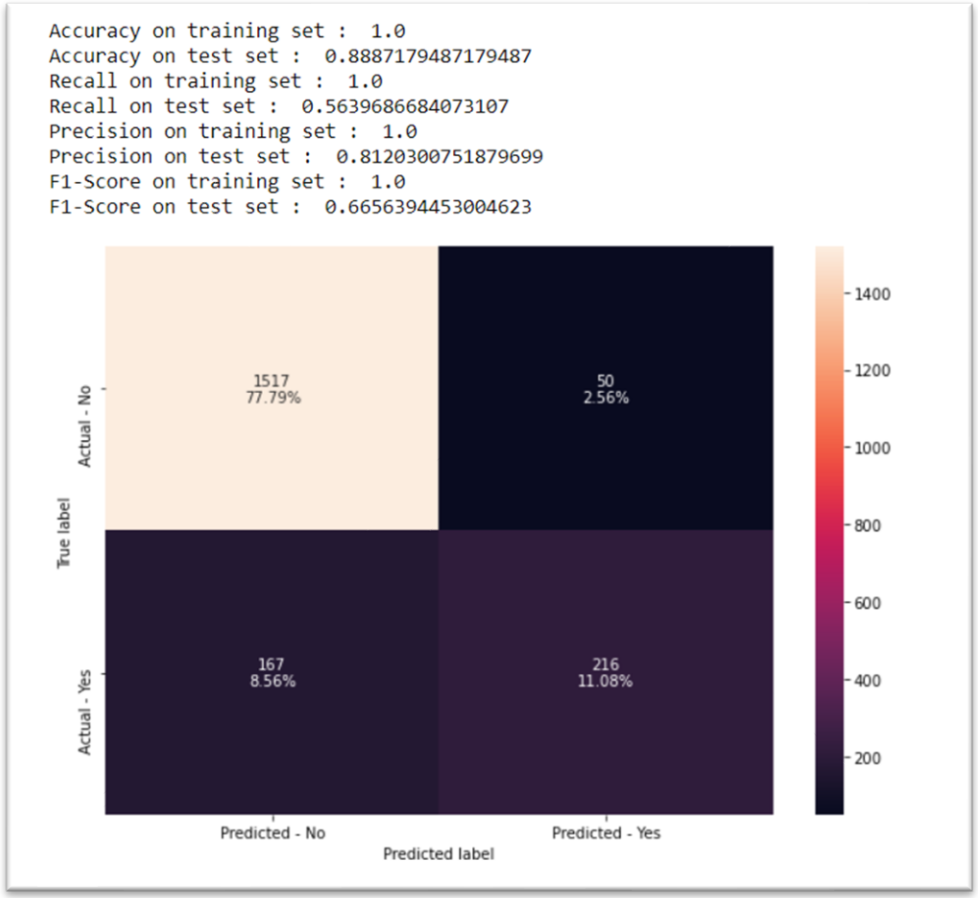


Syracuse University

Random Forest

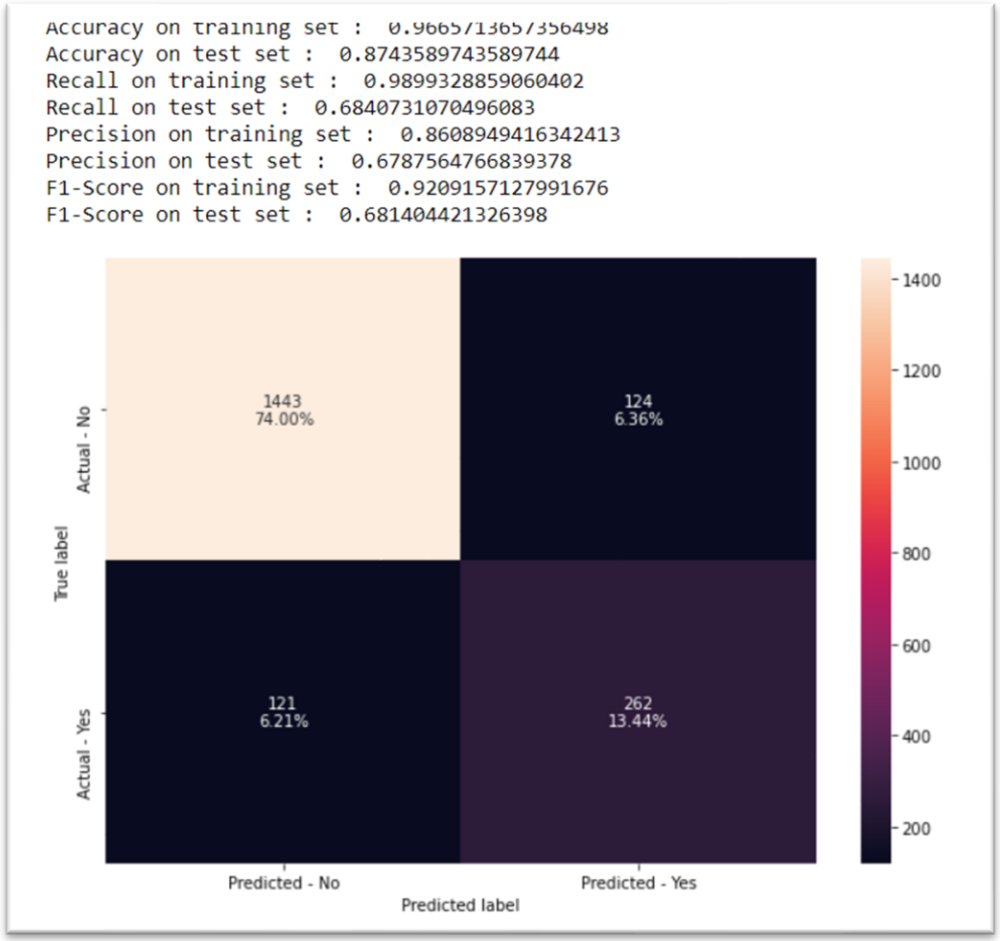
Random Forest Classifier Algorithm
- Model baseline performance

- Random forest is giving slightly higher test f1-score than decision trees but it is overfitting the training data.
- Let's try hyperparameter tuning and see if the model performance improves.



Random Forest Classifier Algorithm
- Hyperparameter tuning and Evaluation

- The overfitting has reduced significantly and the model performance has improved.
- The test recall and test f1-score has increased.



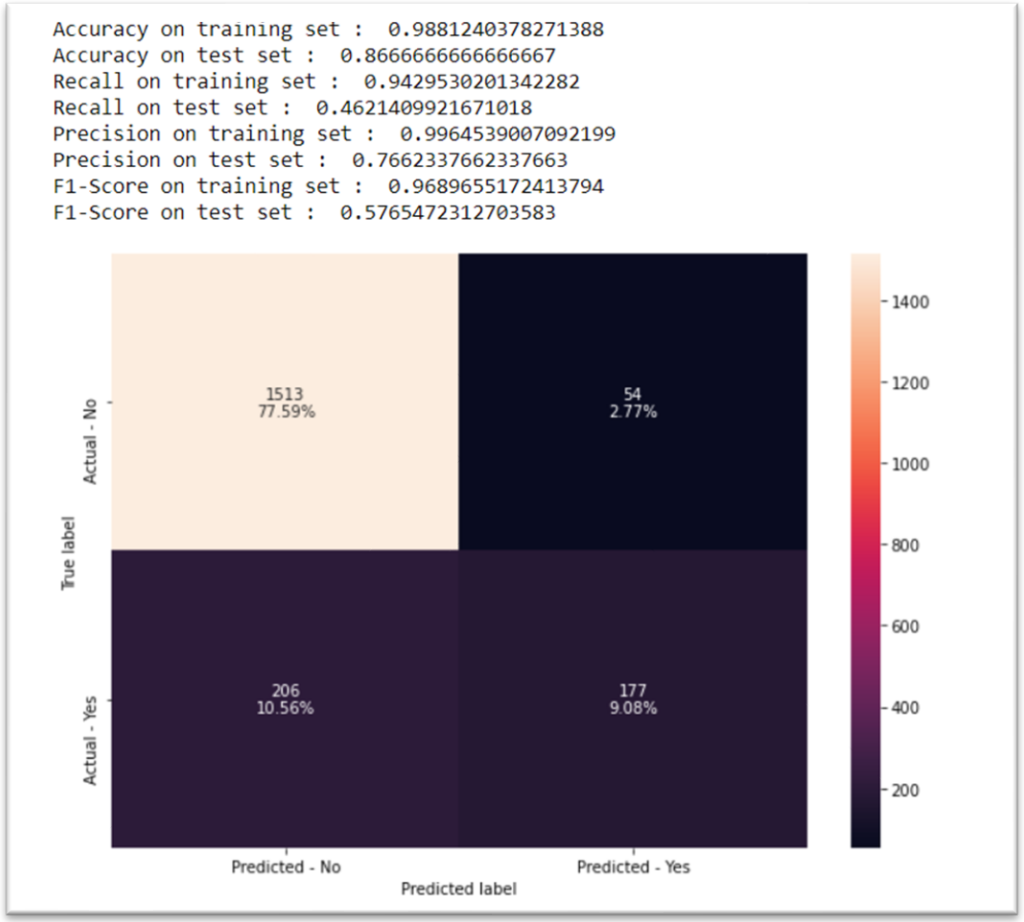
Syracuse University

Bagging Classifier

Bagging Classifier Algorithm
- Model baseline performance

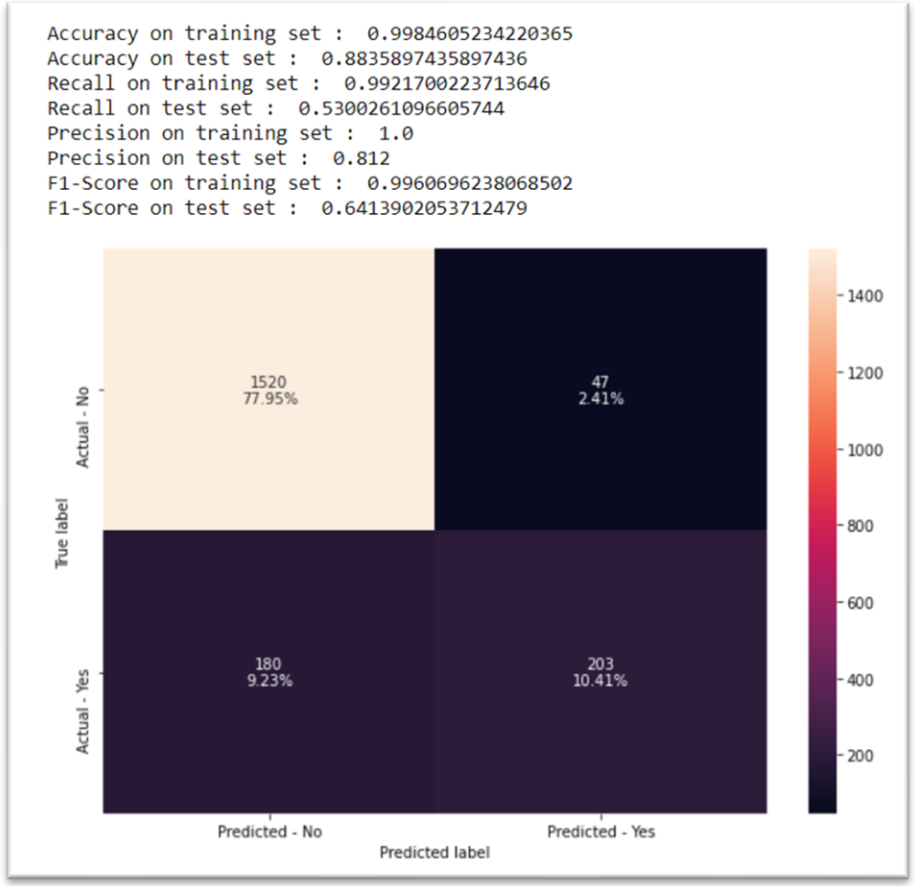
- Bagging classifier is overfitting the training data.
- Let's try hyperparameter tuning and see if the model performance improves.

```
Accuracy on training set :  0.9881240378271388
Accuracy on test set :  0.8666666666666667
Recall on training set :  0.9429530201342282
Recall on test set :  0.4621409921671018
Precision on training set :  0.9964539007092199
Precision on test set :  0.7662337662337663
F1-Score on training set :  0.9689655172413794
F1-Score on test set :  0.5765472312703583
```



Bagging Classifier Algorithm
- Hyperparameter tuning and Evaluation

- Surprisingly,the model performance has decreased after hyperparameter tuning.
- Let's try now try boosting models.

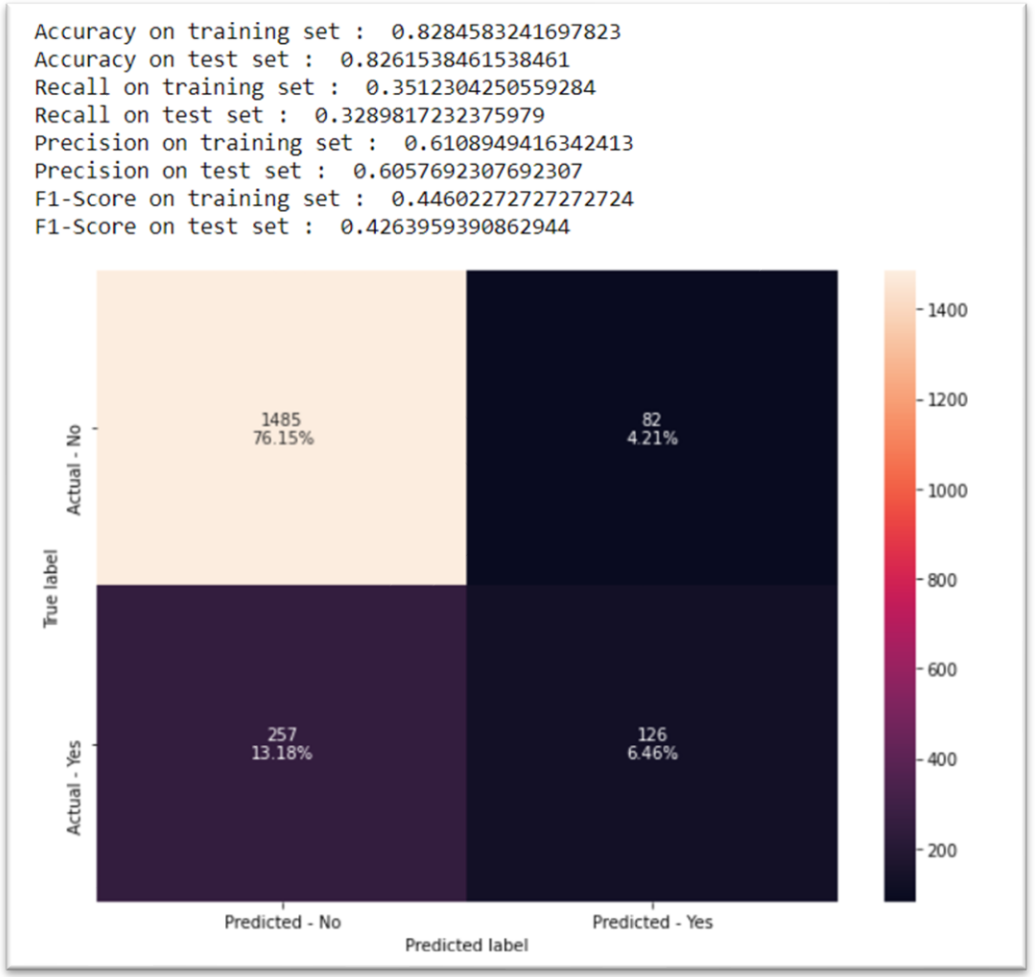


Syracuse University

Boosting
-AdaBoost
-Gradient Boost
-XG Boost

AdaBoost Classifier Algorithm
- Model baseline performance

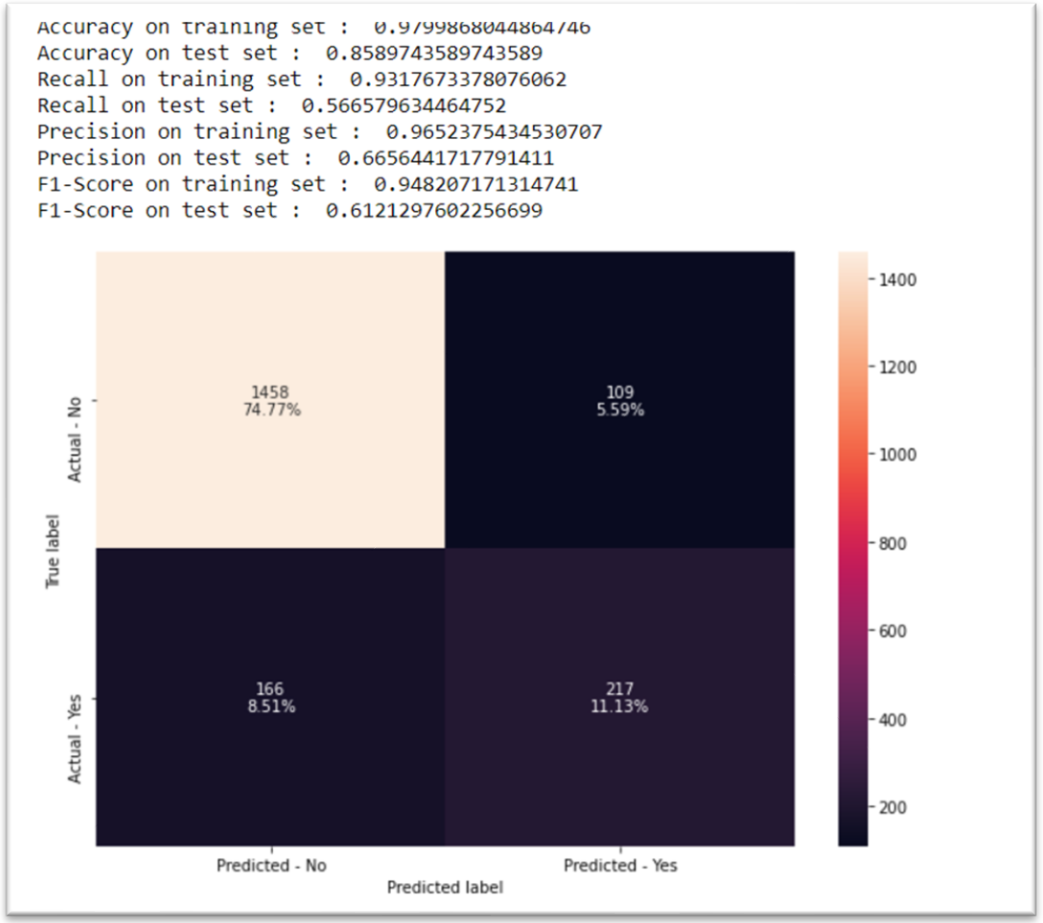
- Adaboost is giving more generalized performance than previous models but the test f1-score is too low.



AdaBoost Classifier Algorithm
- Hyperparameter tuning and Evaluation

- The model performance has increased slightly but the model has started to overfit the training data.

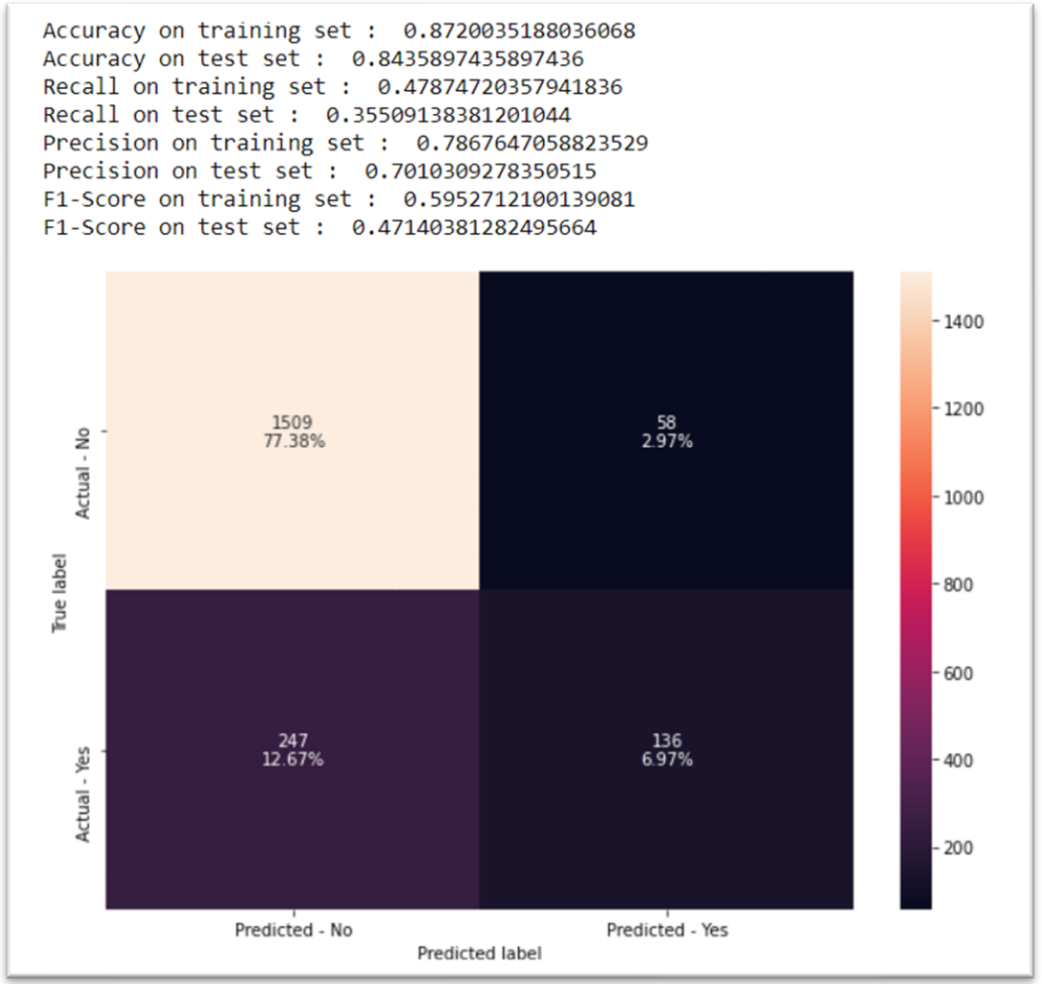
```
Accuracy on training set :  0.9799868044864746
Accuracy on test set :  0.8589743589743589
Recall on training set :  0.9317673378076062
Recall on test set :  0.566579634464752
Precision on training set :  0.9652375434530707
Precision on test set :  0.6656441717791411
F1-Score on training set :  0.948207171314741
F1-Score on test set :  0.6121297602256699
```



Gradient Boosting Algorithm
- Model baseline performance

- The gradient boosting classifier is overfitting the training data.

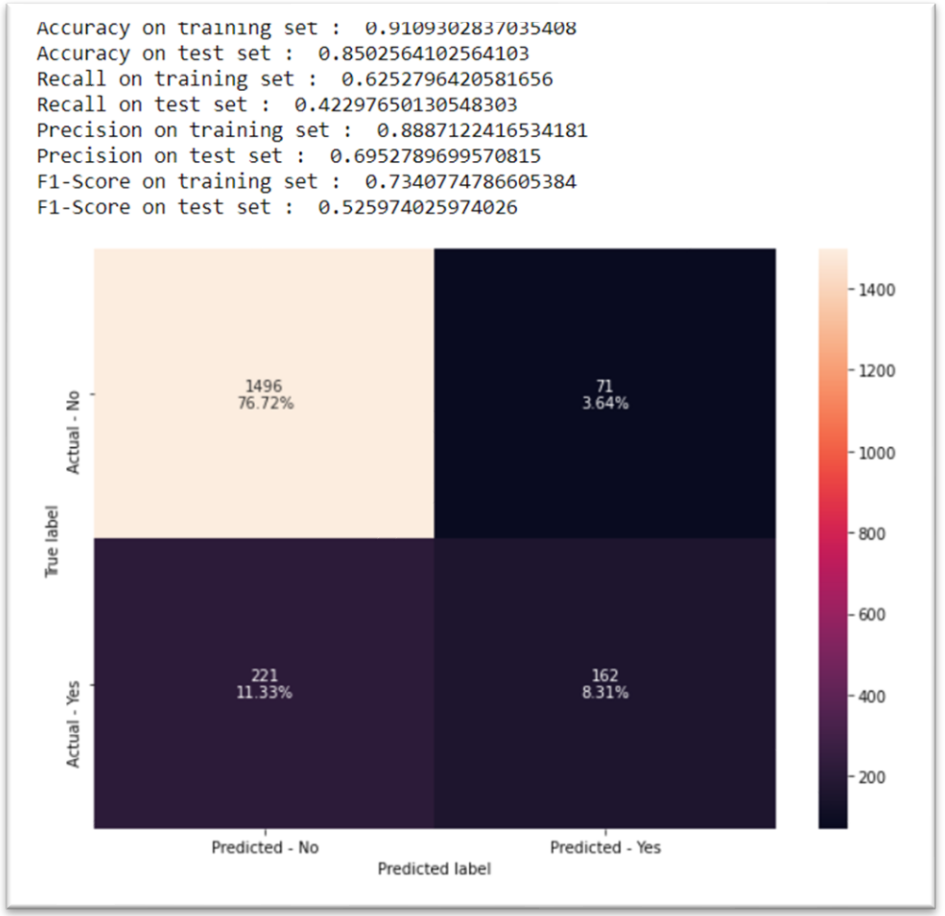
```
Accuracy on training set :  0.8720035188036068
Accuracy on test set :  0.8435897435897436
Recall on training set :  0.47874720357941836
Recall on test set :  0.35509138381201044
Precision on training set :  0.7867647058823529
Precision on test set :  0.7010309278350515
F1-Score on training set :  0.5952712100139081
F1-Score on test set :  0.47140381282495664
```



Gradient Boosting Algorithm
- Hyperparameter tuning and Evaluation

- There is not much difference in the model performance after hyperparameter tuning.

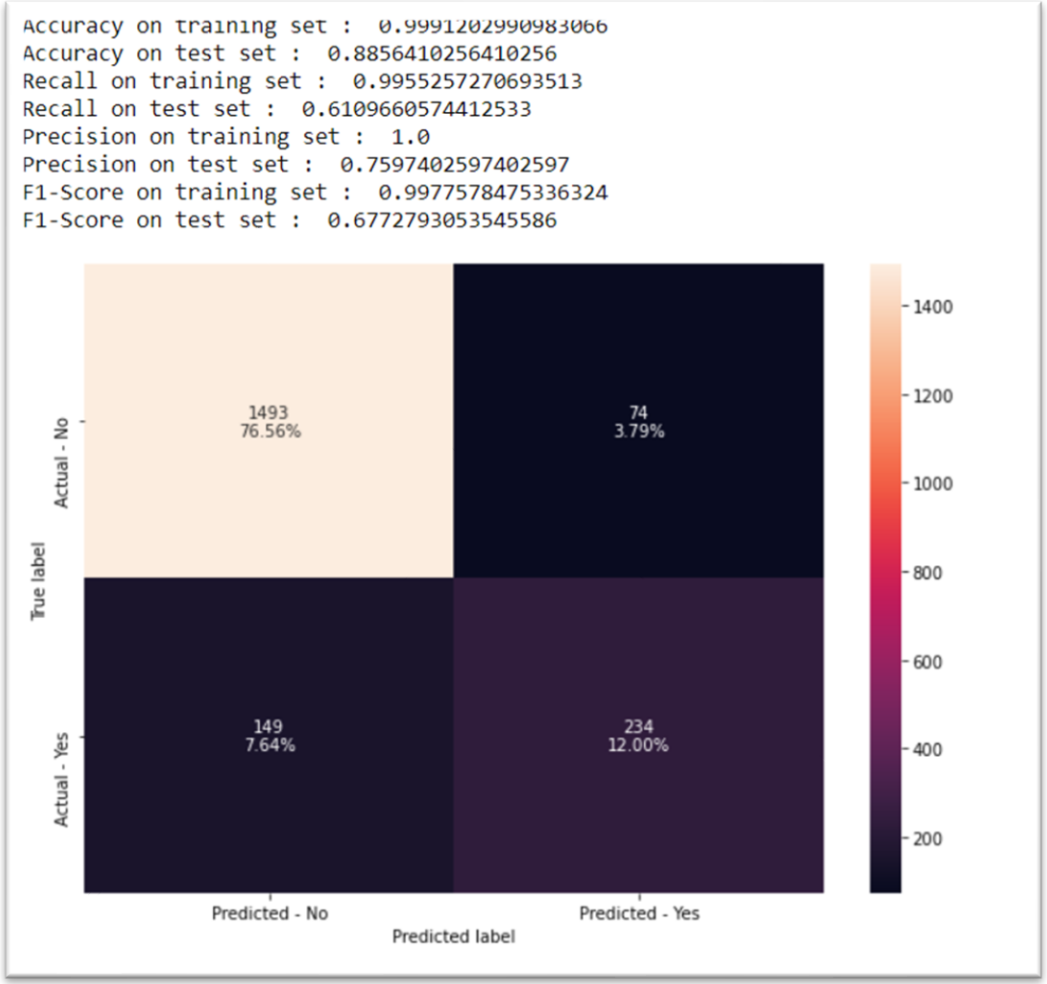
```
Accuracy on training set :  0.9109302837035408
Accuracy on test set :  0.8502564102564103
Recall on training set :  0.6252796420581656
Recall on test set :  0.42297650130548303
Precision on training set :  0.8887122416534181
Precision on test set :  0.6952789699570815
F1-Score on training set :  0.7340774786605384
F1-Score on test set :  0.525974025974026
```



XG Boost Algorithm
- Model baseline performance

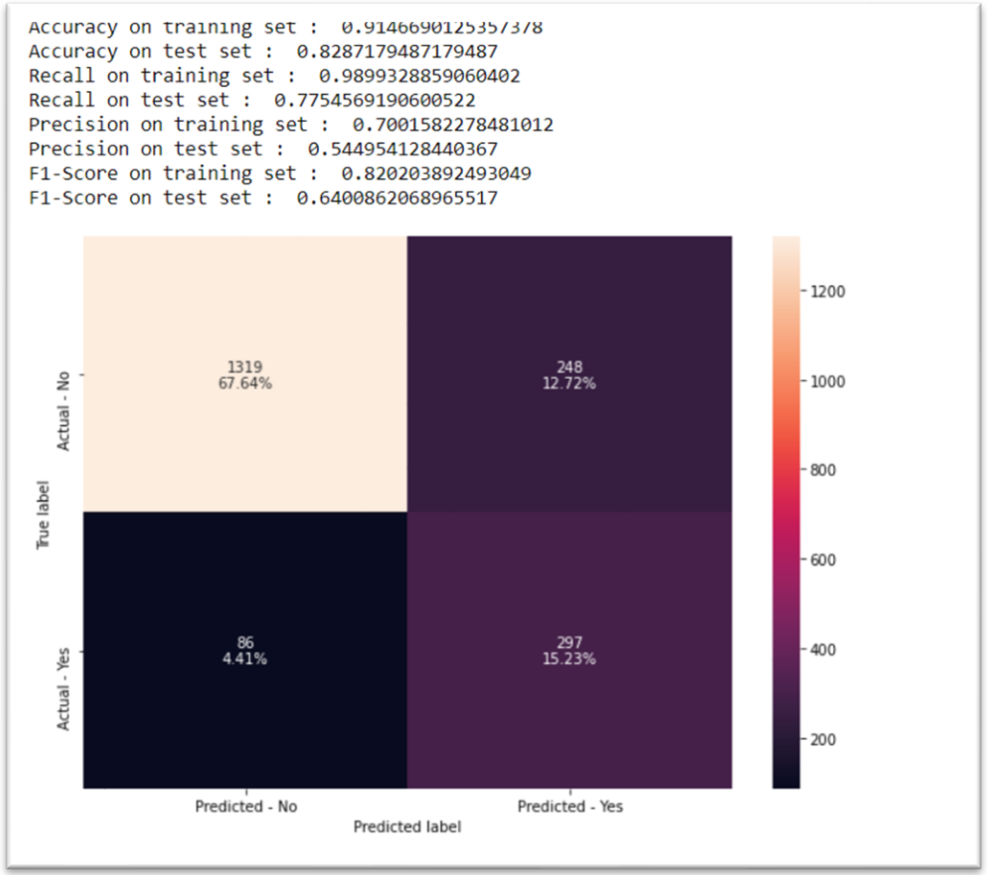
- Xgboost classifier is overfitting the training data.
- Let's try hyperparameter tuning and see if the model performance improves.

```
Accuracy on training set :  0.9991202990983066
Accuracy on test set :  0.8856410256410256
Recall on training set :  0.9955257270693513
Recall on test set :  0.6109660574412533
Precision on training set :  1.0
Precision on test set :  0.7597402597402597
F1-Score on training set :  0.9977578475336324
F1-Score on test set :  0.6772793053545586
```



XG Boost Algorithm
- Hyperparameter tuning and Evaluation

- The overfitting has reduced slightly but there is not much difference in the model performance.



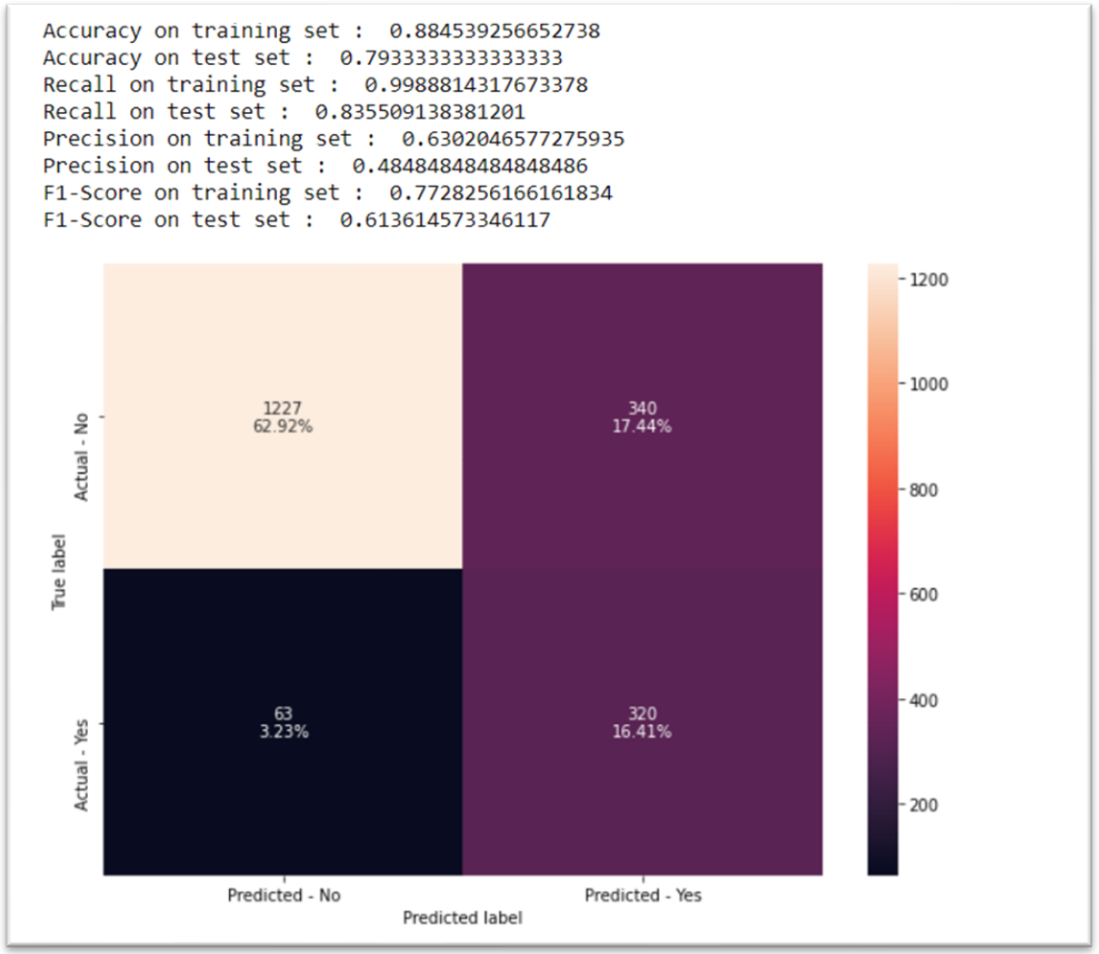
Syracuse University

Stacking Classifier (Ensemble learning)

Ensemble Learning (Stacking)
- Model baseline performance

- The stacking classifier is giving similar performance as compared to XGBoost with slightly less overfitting.
- The confusion matrix shows that the model can identify most premium quality wines, but it is better at identifying non-premium quality wines.

```
Accuracy on training set :  0.884539256652738
Accuracy on test set :  0.7933333333333333
Recall on training set :  0.9988814317673378
Recall on test set :  0.835509138381201
Precision on training set :  0.6302046577275935
Precision on test set :  0.48484848484848486
F1-Score on training set :  0.7728256166161834
F1-Score on test set :  0.613614573346117
```



Syracuse University

Model Comparisons

Comparison of all models

	Model	Train_Accuracy	Test_Accuracy	Train_Recall	Test_Recall	Train_Precision	Test_Precision	Train_F1-Score	Test_F1-Score
3	Tuned Random Forest	0.966571	0.874359	0.989933	0.684073	0.860895	0.678756	0.920916	0.681404
10	XGBoost Classifier	0.999120	0.885641	0.995526	0.610966	1.000000	0.759740	0.997758	0.677279
2	Random Forest	1.000000	0.888718	1.000000	0.563969	1.000000	0.812030	1.000000	0.665639
5	Bagging Classifier Tuned	0.998461	0.883590	0.992170	0.530026	1.000000	0.812000	0.996070	0.641390
11	Tuned XGBoost Classifier	0.914669	0.828718	0.989933	0.775457	0.700158	0.544954	0.820204	0.640086
12	Stacking Classifier	0.884539	0.793333	0.998881	0.835509	0.630205	0.484848	0.772826	0.613615
7	Tuned AdaBoost Classifier	0.979987	0.858974	0.931767	0.566580	0.965238	0.665644	0.948207	0.612130
0	Decision Tree	1.000000	0.825641	1.000000	0.605744	1.000000	0.551069	1.000000	0.577114
4	Bagging Classifier	0.988124	0.866667	0.942953	0.462141	0.996454	0.766234	0.968966	0.576547
9	Tuned Gradient Boosting Classifier	0.910930	0.850256	0.625280	0.422977	0.888712	0.695279	0.734077	0.525974
1	Tuned Decision Tree	0.739389	0.718462	0.799776	0.725849	0.415456	0.385042	0.546845	0.503167
8	Gradient Boosting Classifier	0.872004	0.843590	0.478747	0.355091	0.786765	0.701031	0.595271	0.471404
6	AdaBoost Classifier	0.828458	0.826154	0.351230	0.328982	0.610895	0.605769	0.446023	0.426396

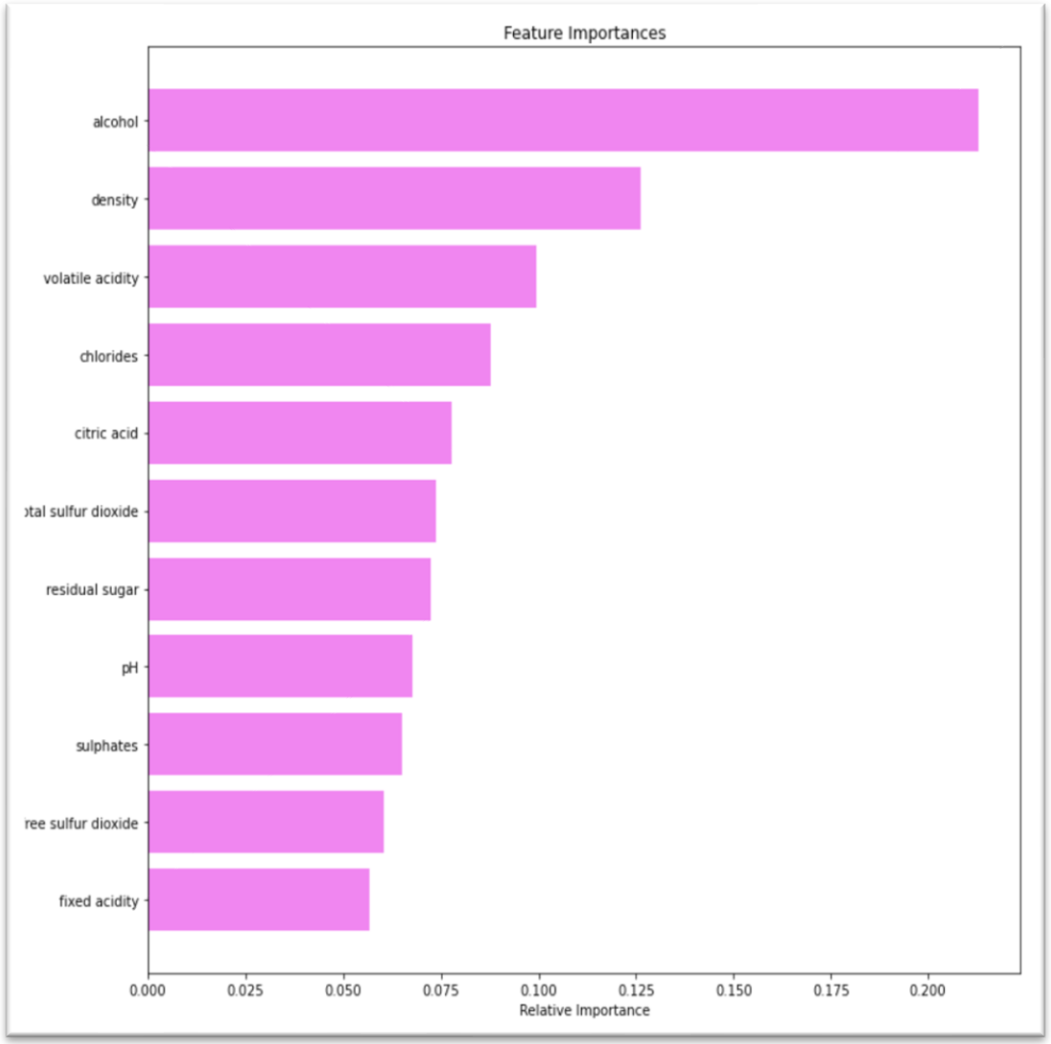
- Majority of the models are overfitting the training data in terms of F1-score.
- Bagging classifier is giving the highest f1-score on the test data but is overfitting the training data.
- The tuned random forest has given the second highest test f1-score and is giving more generalized performance as compared to the bagging classifier.

Syracuse University

Feature Importance

Feature Importance of Tuned Random Forest

Alcohol is the most important feature in Identifying premium quality wine followed by sulphates' and volatile acidity.



Syracuse University

Business Recommendations and Insights

Business Recommendations and Insights

- ✓ Based on our analysis, we can say that the premium quality wine has following features in comparison to the non-premium quality wine:
  - Higher concentration of alcohol.
  - A fair and higher concentration of sulphates. Fair indicates smaller range of values or less extreme values.
  - Less volatile acidity.
  - Higher fixed acidity.
  - Higher citric acid concentration.
  - A fair and lower concentration of total sulfur dioxide and free sulfur dioxide.
  - Lower concentration of chlorides.
  - Lesser density.
  - A fair pH level i.e., neither very acidic or very less acidic.
- ✓ The company should be more precise with the concentration and level of alcohol, volatile acidity, and sulphates as these are the most important factors in determining the quality of wines. For example, the common alcohol level for the premium quality wine should be between 11-13%.
- ✓ Once the desired performance is achieved from the model, the company can use it to identify the premium quality wines for new production. This would help to reduce the cost and increase the efficiency of the process.