Predicting Airline Passenger Satisfaction

Airin Konno

Aim

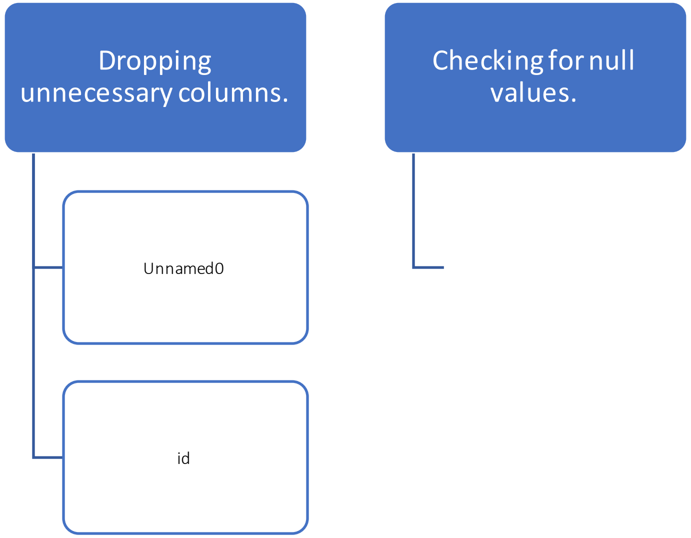
- Given survey data from an airline, predict customer satisfaction (satisfied or dissatisfied) using ML algorithms.
- Conduct EDA and data visualization to better understand the data.
- Potential business value:
  - Understand which features have the biggest impact on customer satisfaction and work towards improving those features.

Data Collection

Dataset from Kaggle.

- Already split into train (80% of data) and test (20% of data) files.

Data Preprocessing



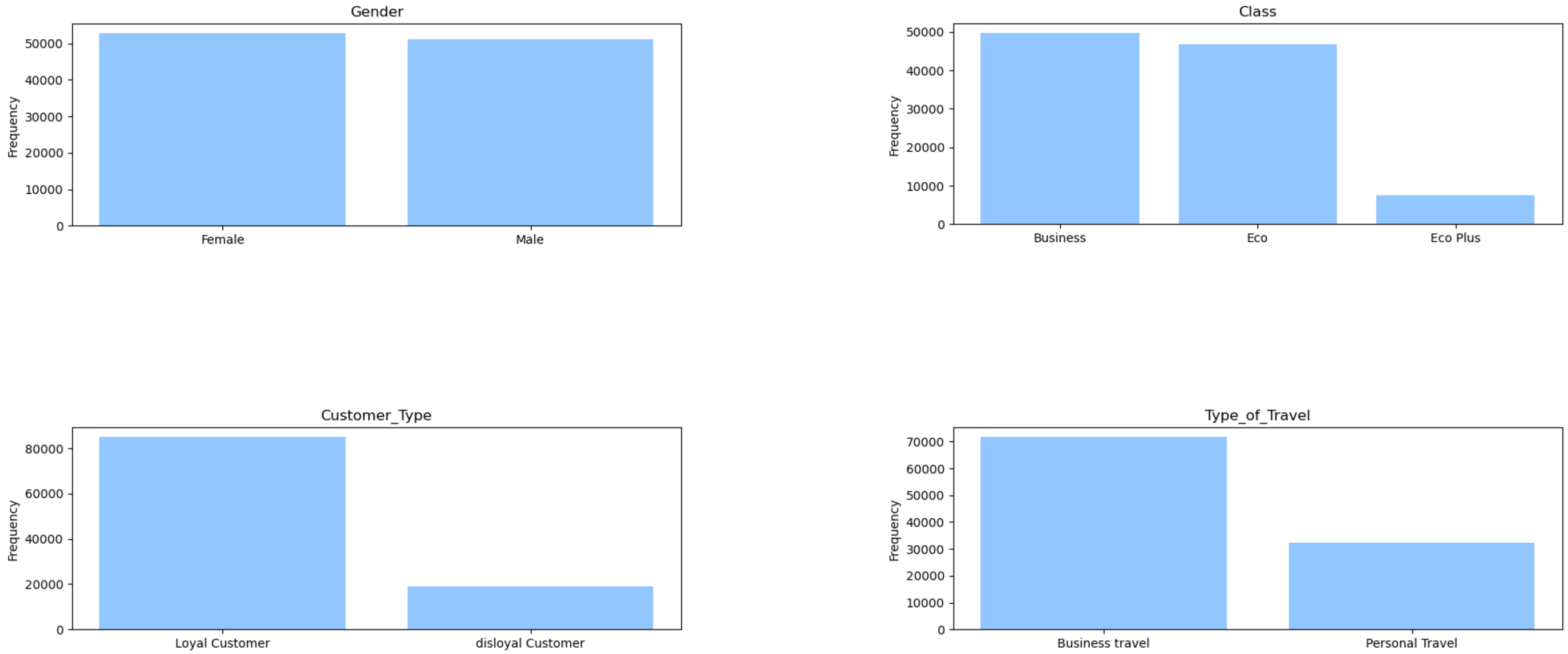
Dropping unnecessary columns.

Unnamed0

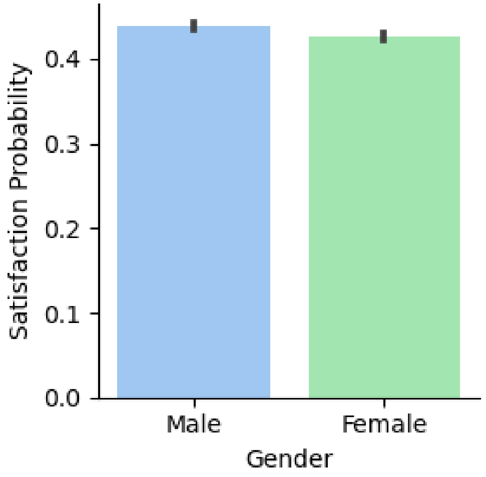
id

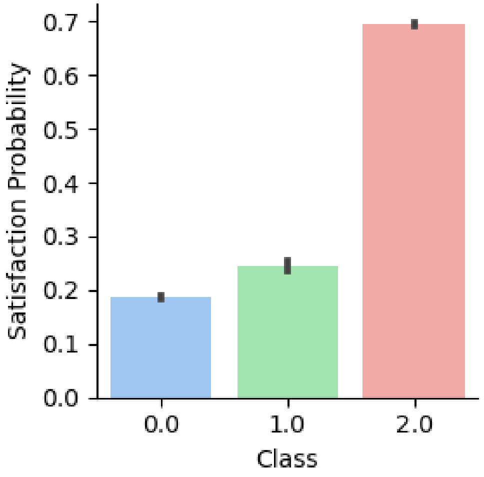
Checking for null values.

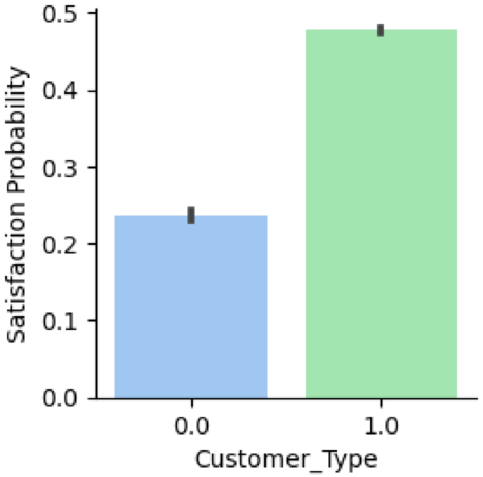
EDA – Categorical Variables Distribution

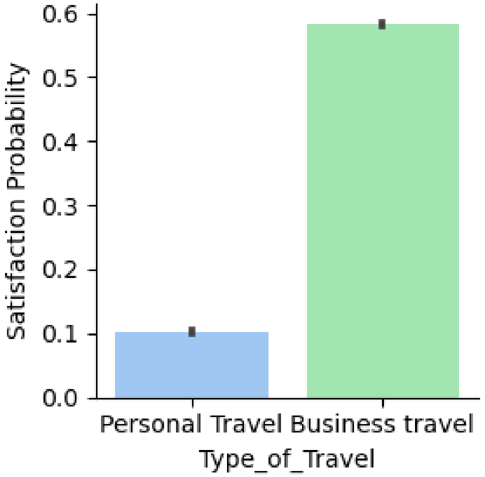


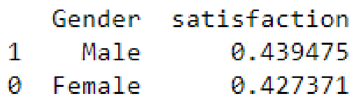
EDA – Relationship with Satisfaction



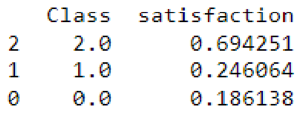




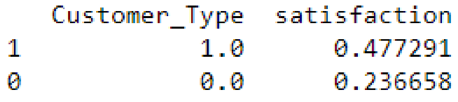




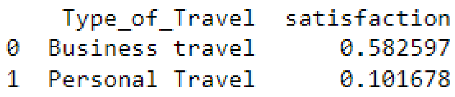
```
   Gender  satisfaction
1    Male      0.439475
0  Female      0.427371
```



```
   Class  satisfaction
2    2.0      0.694251
1    1.0      0.246064
0    0.0      0.186138
```

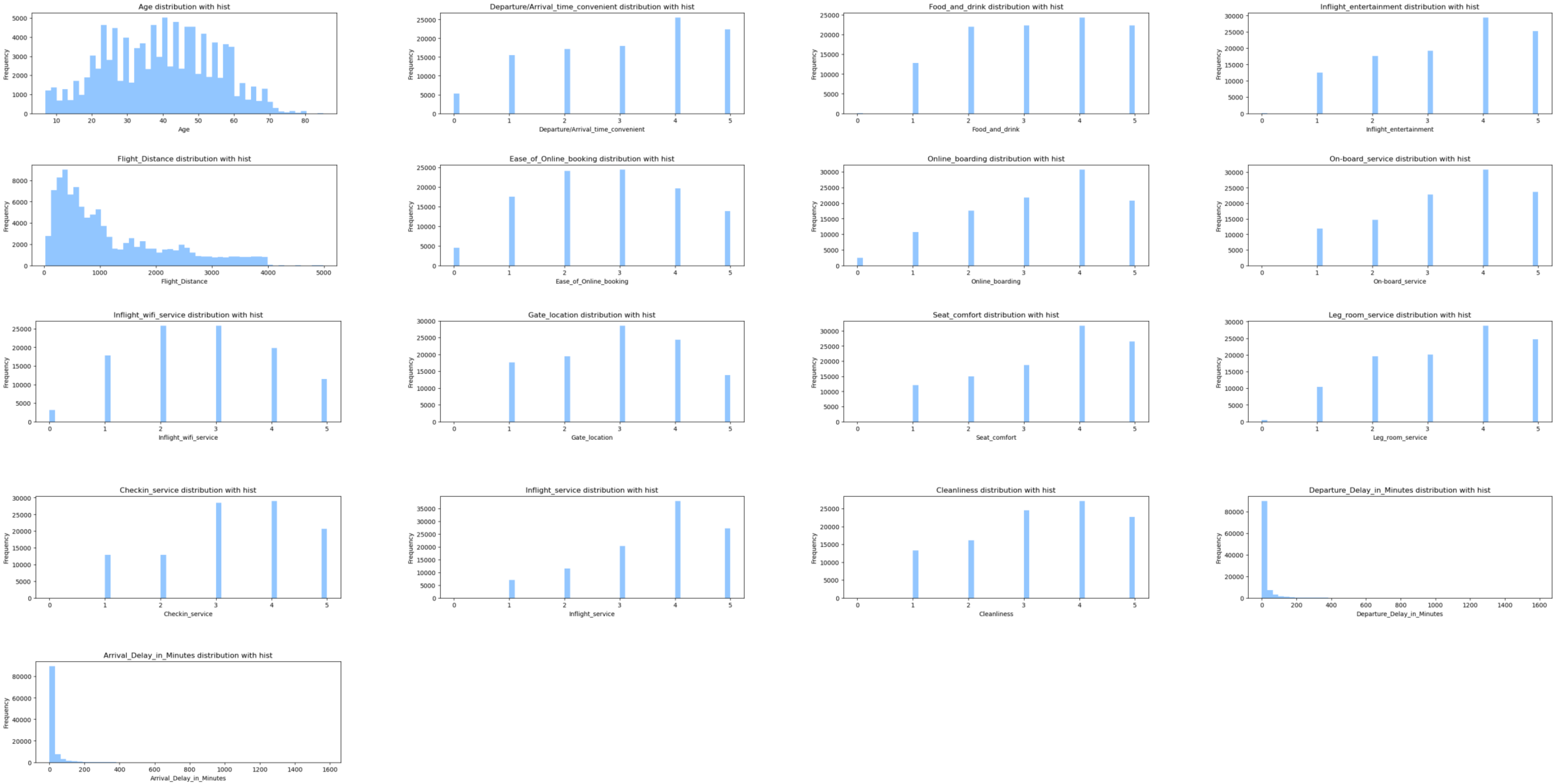


```
   Customer_Type  satisfaction
1            1.0      0.477291
0            0.0      0.236658
```



```
    Type_of_Travel  satisfaction
0  Business travel      0.582597
1  Personal Travel      0.101678
```

EDA – Numerical Variables Distribution



Age distribution with hist

Departure/Arrival_time_convenient distribution with hist

Food_and_drink distribution with hist

Inflight_entertainment distribution with hist

Flight_Distance distribution with hist

Ease_of_Online_booking distribution with hist

Online_boarding distribution with hist

On-board_service distribution with hist

Inflight_wifi_service distribution with hist

Gate_location distribution with hist

Seat_comfort distribution with hist

Leg_room_service distribution with hist

Checkin_service distribution with hist

Inflight_service distribution with hist

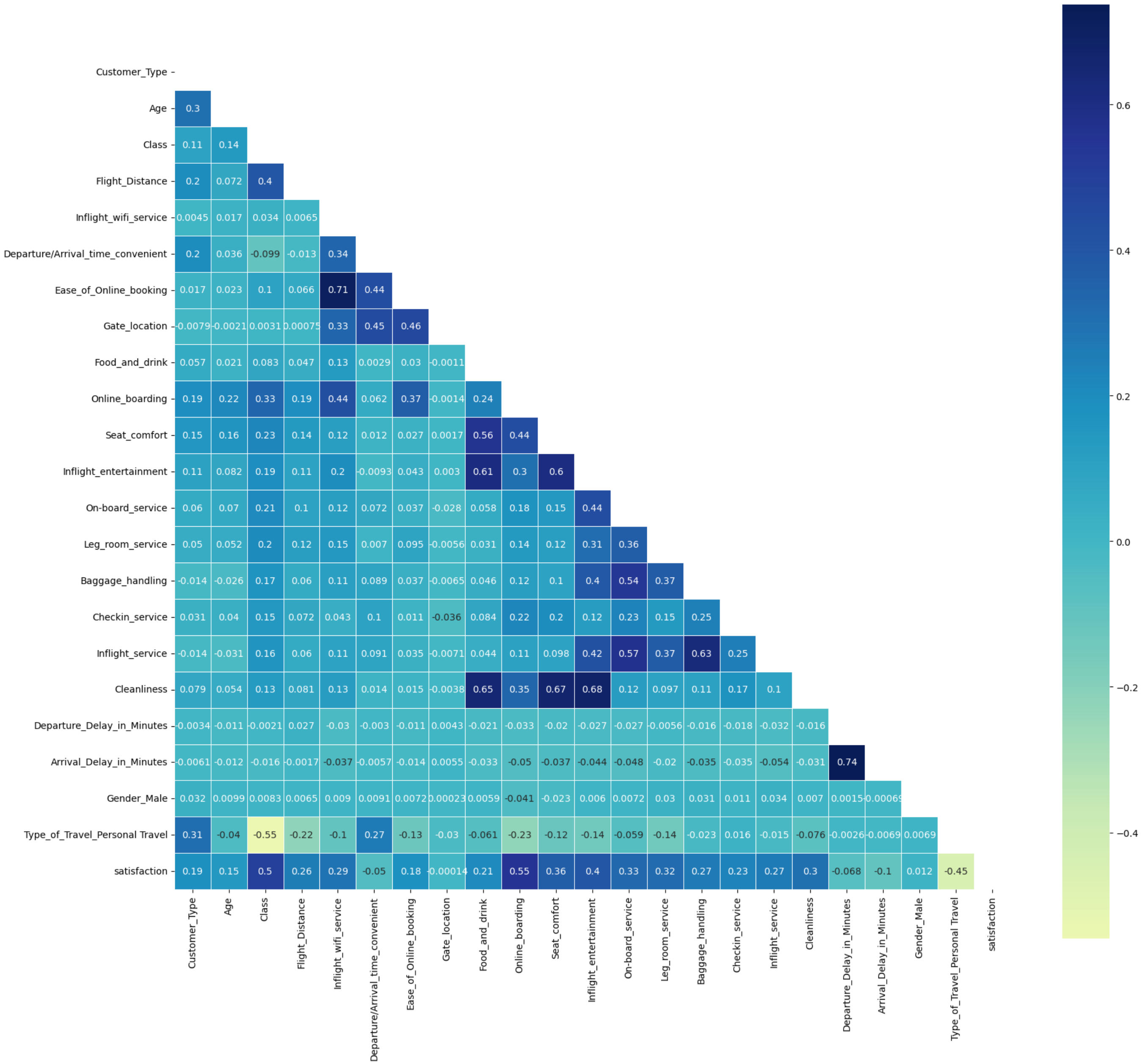
Cleanliness distribution with hist

Departure_Delay_in_Minutes distribution with hist

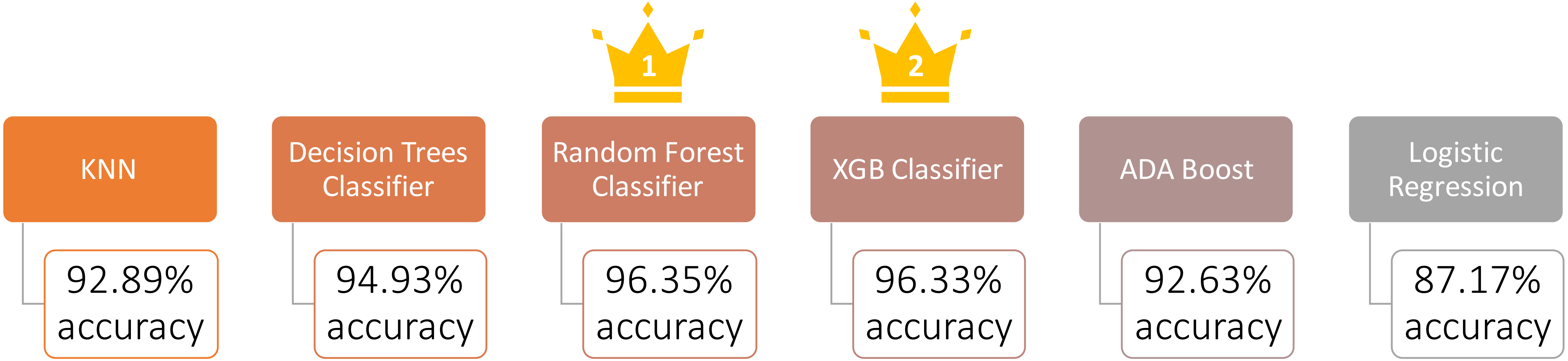
Arrival_Delay_in_Minutes distribution with hist

EDA - Correlation Heat Map

- Shows which features correlate well with customer satisfaction.
- Best features
  - Online Booking, Class, and Inflight Entertainment
- Worst features
  - Type of Travel, Arrival Delay in Minutes, Departure/Arrival time convenient



Machine Learning Algorithms



KNN
92.89% accuracy

Decision Trees Classifier
94.93% accuracy

Random Forest Classifier
96.35% accuracy

XGB Classifier
96.33% accuracy

ADA Boost
92.63% accuracy

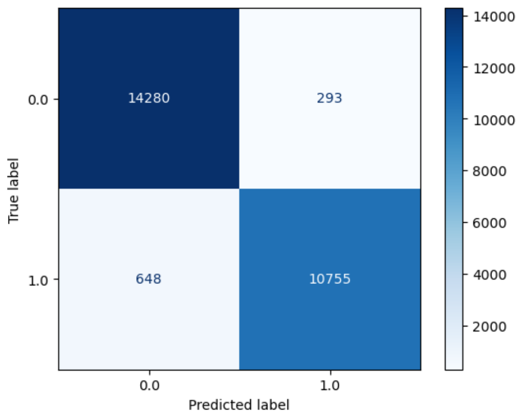
Logistic Regression
87.17% accuracy

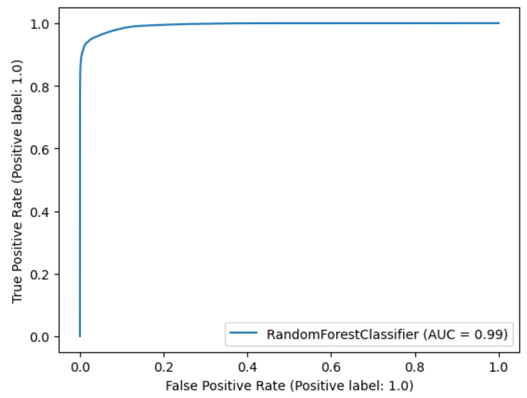
Model 1

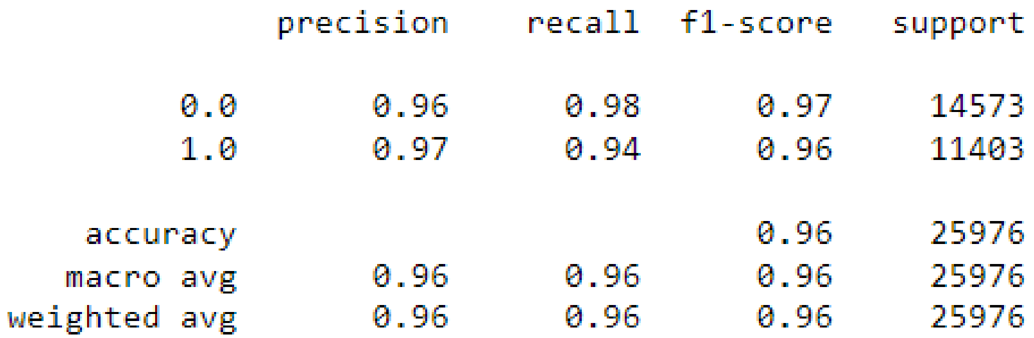
Random Forest Classifier

Accuracy: 96.35%

Next steps: PCA, Grid Search





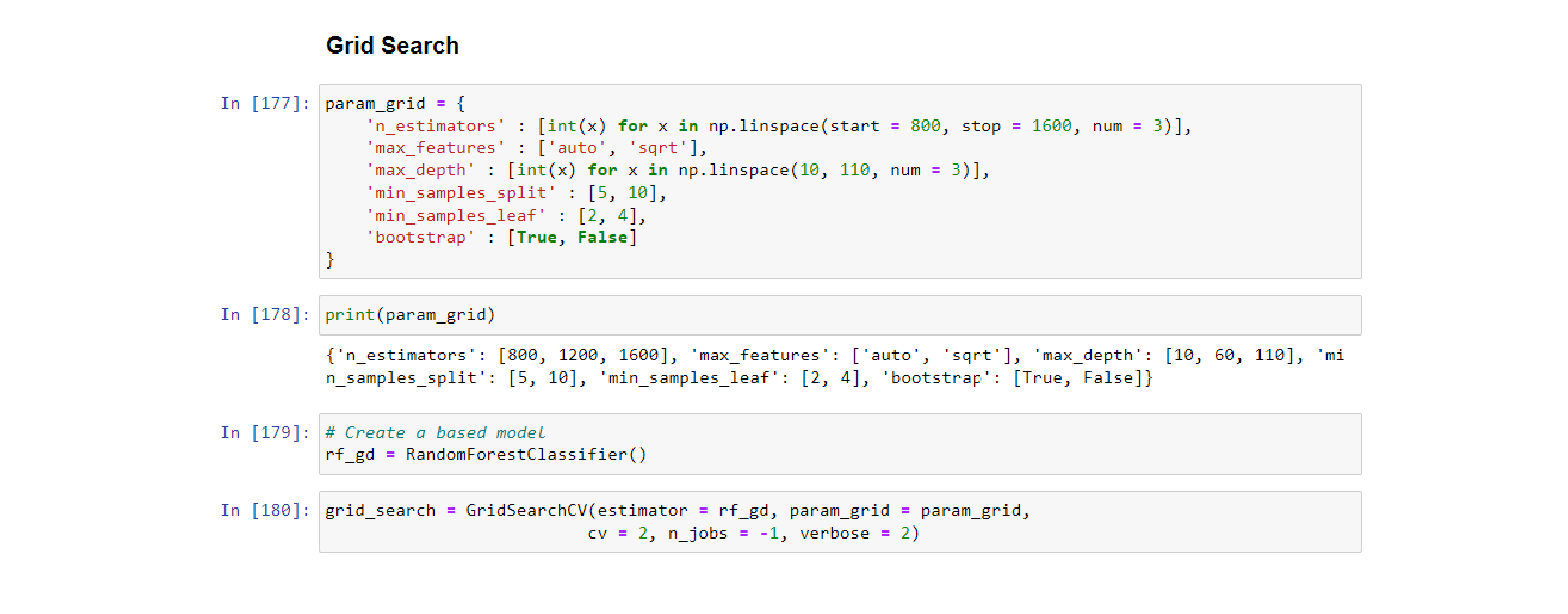


```
              precision    recall  f1-score   support

         0.0       0.96      0.98      0.97     14573
         1.0       0.97      0.94      0.96     11403

    accuracy                           0.96     25976
   macro avg       0.96      0.96      0.96     25976
weighted avg       0.96      0.96      0.96     25976
```

Model 1: Dimensionality Reduction PCA



Grid Search

```
In [177]: param_grid = {
              'n_estimators' : [int(x) for x in np.linspace(start = 800, stop = 1600, num = 3)],
              'max_features' : ['auto', 'sqrt'],
              'max_depth' : [int(x) for x in np.linspace(10, 110, num = 3)],
              'min_samples_split' : [5, 10],
              'min_samples_leaf' : [2, 4],
              'bootstrap' : [True, False]
          }
```

```
In [178]: print(param_grid)

          {'n_estimators': [800, 1200, 1600], 'max_features': ['auto', 'sqrt'], 'max_depth': [10, 60, 110], 'mi
          n_samples_split': [5, 10], 'min_samples_leaf': [2, 4], 'bootstrap': [True, False]}
```

```
In [179]: # Create a based model
          rf_gd = RandomForestClassifier()
```

```
In [180]: grid_search = GridSearchCV(estimator = rf_gd, param_grid = param_grid,
                                     cv = 2, n_jobs = -1, verbose = 2)
```

Model 1: Grid Search

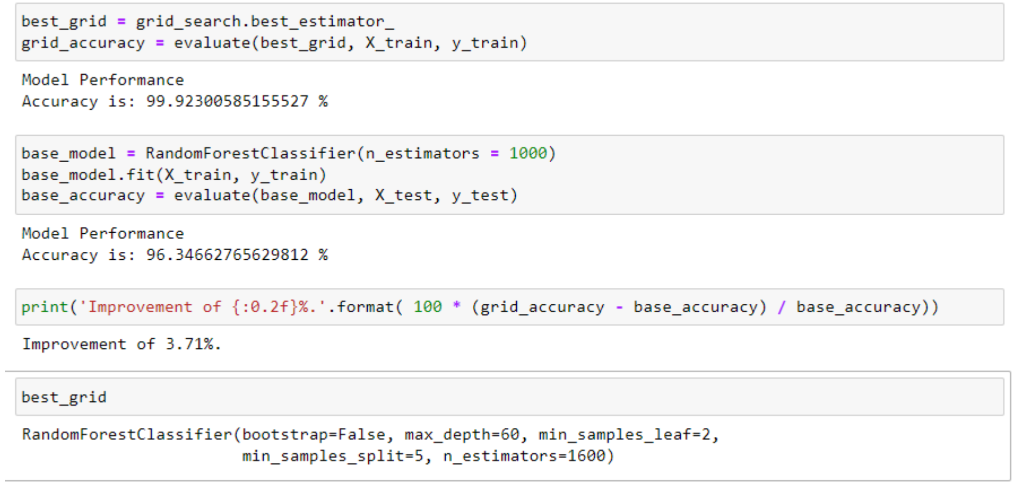
6 parameters were tuned.

There were a total of 144 combinations.

Fitted 2 folds, creating 288 fits in total.

Model 1:

Grid Search Results



```
best_grid = grid_search.best_estimator_
grid_accuracy = evaluate(best_grid, X_train, y_train)
```

```
Model Performance
Accuracy is: 99.92300585155527 %
```

```
base_model = RandomForestClassifier(n_estimators = 1000)
base_model.fit(X_train, y_train)
base_accuracy = evaluate(base_model, X_test, y_test)
```

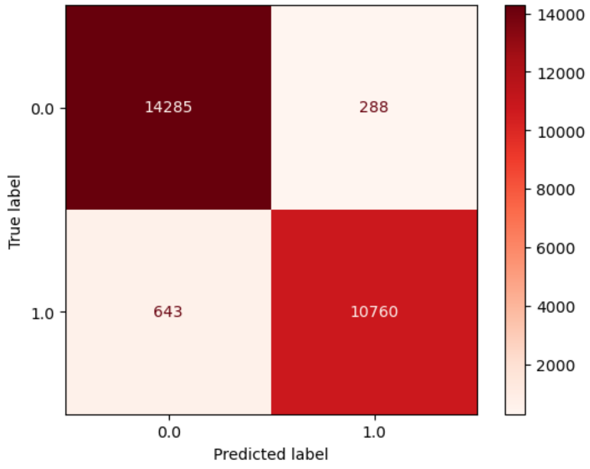
```
Model Performance
Accuracy is: 96.34662765629812 %
```

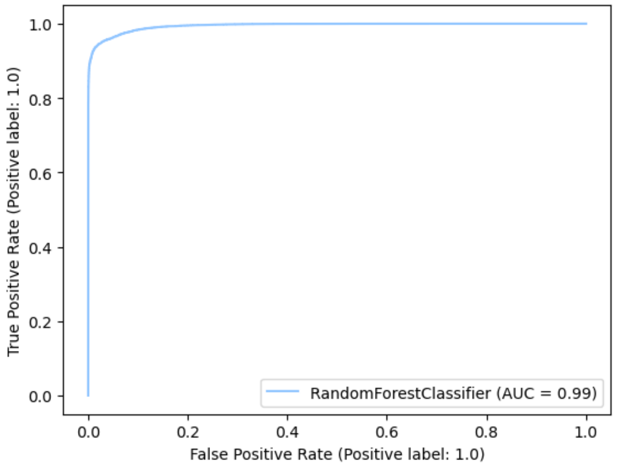
```
print('Improvement of {:0.2f}%.'.format( 100 * (grid_accuracy - base_accuracy) / base_accuracy))
```

```
Improvement of 3.71%.
```

```
best_grid
```

```
RandomForestClassifier(bootstrap=False, max_depth=60, min_samples_leaf=2,
                       min_samples_split=5, n_estimators=1600)
```



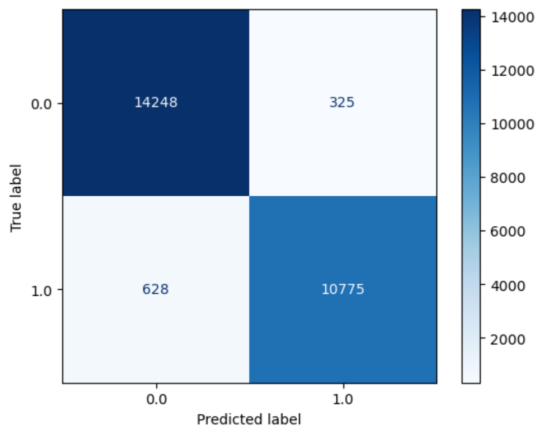


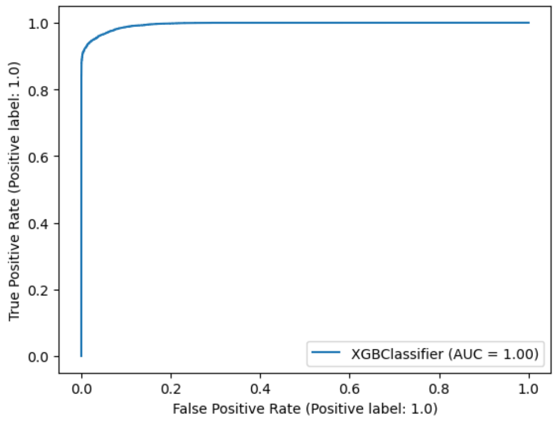
Model 2

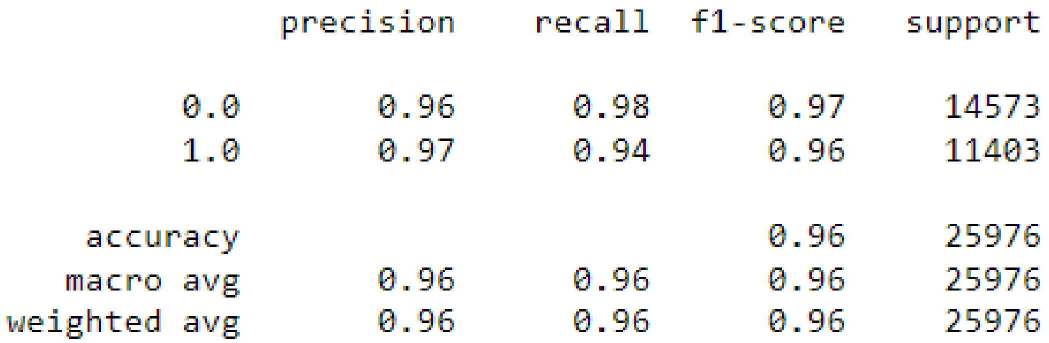
XGB Classifier

Accuracy: 96.33%

Next steps: PCA, Randomized Search





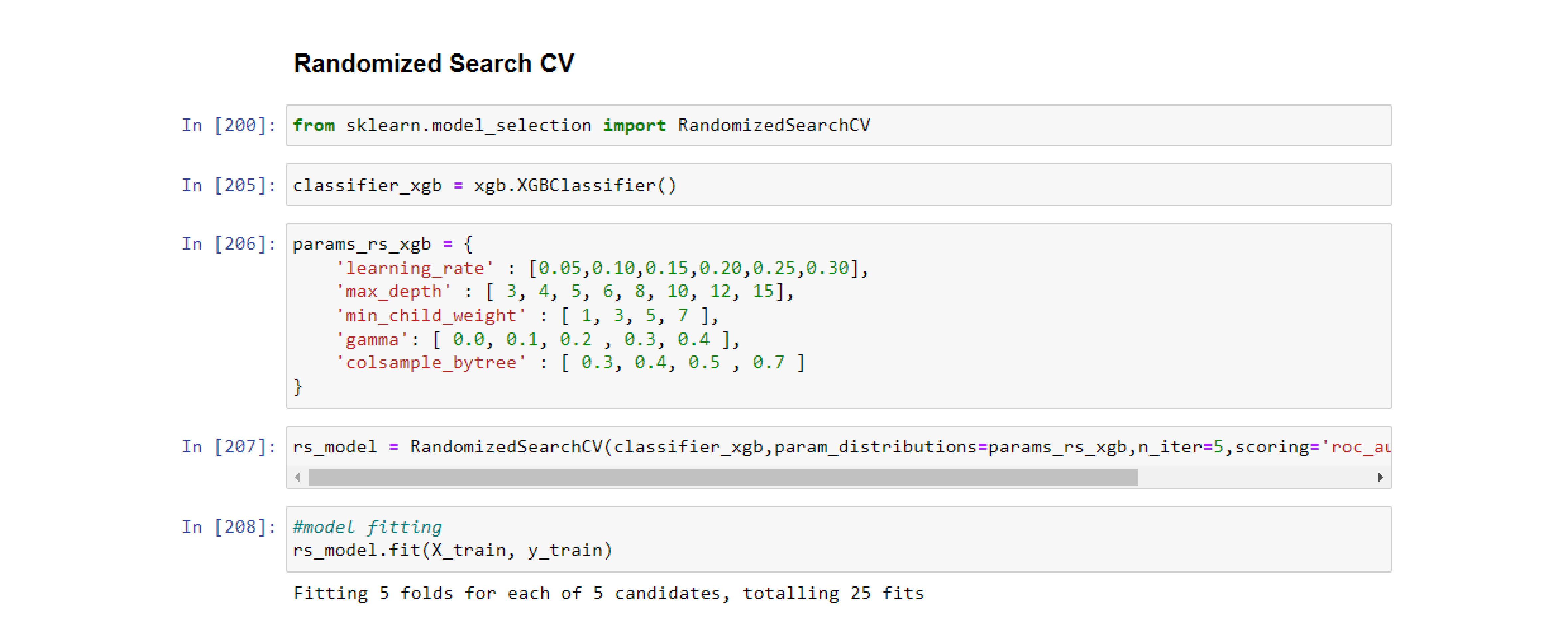


```
              precision    recall  f1-score   support

         0.0       0.96      0.98      0.97     14573
         1.0       0.97      0.94      0.96     11403

    accuracy                           0.96     25976
   macro avg       0.96      0.96      0.96     25976
weighted avg       0.96      0.96      0.96     25976
```

Model 2: Dimensionality Reduction PCA



Randomized Search CV

```
In [200]: from sklearn.model_selection import RandomizedSearchCV

In [205]: classifier_xgb = xgb.XGBClassifier()

In [206]: params_rs_xgb = {
              'learning_rate' : [0.05,0.10,0.15,0.20,0.25,0.30],
              'max_depth' : [ 3, 4, 5, 6, 8, 10, 12, 15],
              'min_child_weight' : [ 1, 3, 5, 7 ],
              'gamma': [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
              'colsample_bytree' : [ 0.3, 0.4, 0.5 , 0.7 ]
          }

In [207]: rs_model = RandomizedSearchCV(classifier_xgb,param_distributions=params_rs_xgb,n_iter=5,scoring='roc_au

In [208]: #model fitting
          rs_model.fit(X_train, y_train)

          Fitting 5 folds for each of 5 candidates, totalling 25 fits
```

Model 1: Randomized Search

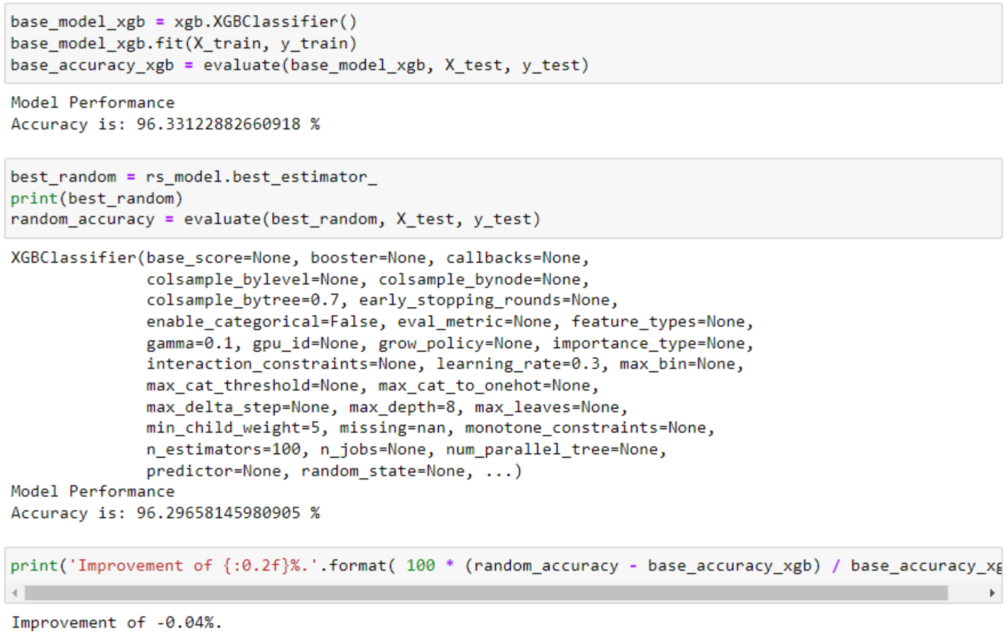
5 parameters were tuned.

Fitted 5 folds for each of the 5 candidates

Total = 25 fits

Model 2:

Randomized Search Results



```
base_model_xgb = xgb.XGBClassifier()
base_model_xgb.fit(X_train, y_train)
base_accuracy_xgb = evaluate(base_model_xgb, X_test, y_test)
```

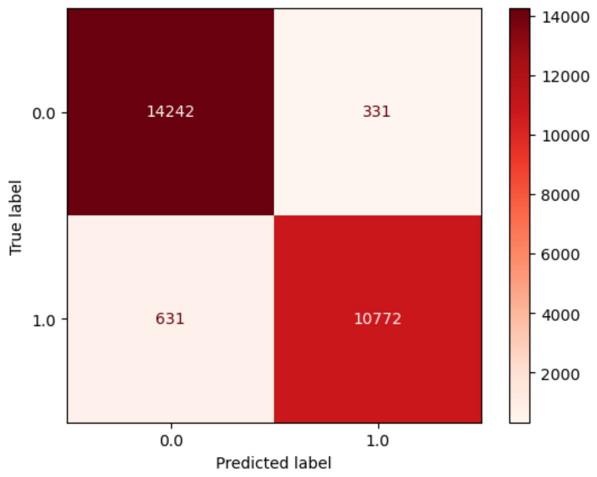
```
Model Performance
Accuracy is: 96.33122882660918 %
```

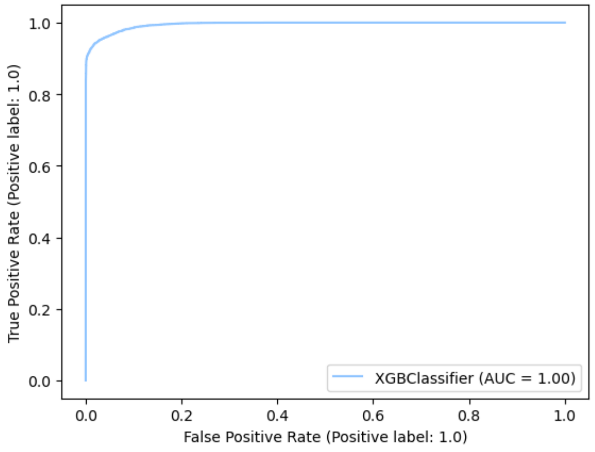
```
best_random = rs_model.best_estimator_
print(best_random)
random_accuracy = evaluate(best_random, X_test, y_test)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=0.7, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=0.1, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.3, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=8, max_leaves=None,
              min_child_weight=5, missing=nan, monotone_constraints=None,
              n_estimators=100, n_jobs=None, num_parallel_tree=None,
              predictor=None, random_state=None, ...)
Model Performance
Accuracy is: 96.29658145980905 %
```

```
print('Improvement of {:0.2f}%.'.format( 100 * (random_accuracy - base_accuracy_xgb) / base_accuracy_xg
```

```
Improvement of -0.04%.
```





Modelling Conclusions

- Model 1 – Random Forest Classifier with Grid Search worked best (99.92% accuracy achieved).
- Even though PCA was conducted, I still proceeded with using the original X_train and X_test value.
- Randomized search also did not help Model 2.
    - Perhaps there were too few iterations.
    - Can try to increase the iterations next time to have more fits, which could help with the accuracy.