Project Development Phase
Model Performance Test

Date	19 November 2022
Team ID	PNT2022TMID30209
Project Name	DEVELOPING A FLIGHT DELAY PREDICTION MODEL USING MACHINE LEARNING
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values
1.	Metrics	**Classification Model:** Confusion Matrix – [1840,0,0,407] Accuracy Score- 100% Classification Report – 100%
2.	Tune the Model	Hyperparameter Tuning – 100% Validation Method – RandomizedSearchCV

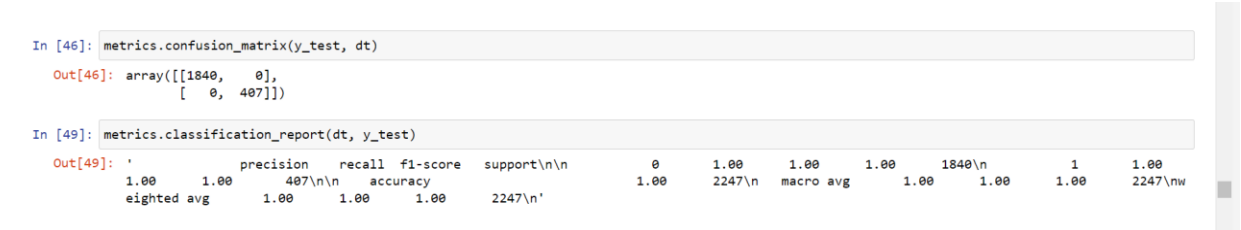
SCREENSHOTS

METRICS

```
from sklearn.metrics import confusion_matrix
```

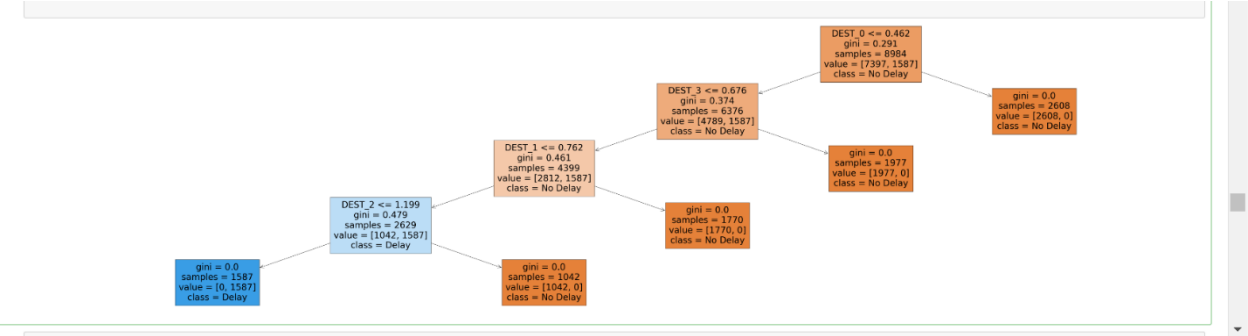
```
from sklearn import tree
fig = plt.figure(figsize =(80,20))
_ = tree.plot_tree(dtc,
                   feature_names=data.columns,
                   class_names=['No Delay', "Delay"],
                   filled=True)

# plot_tree(dtc, feature_names=x_train.columns, max_depth=2, filled=True);
```



```
In [46]: metrics.confusion_matrix(y_test, dt)
Out[46]: array([[1840,    0],
                [   0,  407]])

In [49]: metrics.classification_report(dt, y_test)
Out[49]: '              precision    recall  f1-score   support\n\n           0       1.00      1.00      1.00      1840\n           1       1.00
         1.00      1.00       407\n\n    accuracy                           1.00      2247\n   macro avg       1.00      1.00      1.00      2247\nw
         eighted avg       1.00      1.00      1.00      2247\n'
```




```
import matplotlib.pyplot as plt
import numpy
from sklearn import metrics

confusion_matrix = metrics.confusion_matrix(y_test, dt)

cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False, True])

cm_display.plot()
plt.show()
```

```
from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test, dt)
acc
```

```
2]:  1.0
```

TUNING THE MODEL

```
▸ RandomizedSearchCV
▾ estimator: DecisionTreeClassifier
DecisionTreeClassifier()
    ▾ DecisionTreeClassifier
    DecisionTreeClassifier()
```

```
DT_grid.best_estimator_
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=15, max_features='log2',
                       min_samples_split=6)
```

```
from sklearn.model_selection import RandomizedSearchCV
```

```
parameters = {'max_depth' : (3,5,7,9,10,15,20,25)
              , 'criterion' : ('gini', 'entropy')
              , 'max_features' : ('auto', 'sqrt', 'log2')
              , 'min_samples_split' : (2,4,6)
             }
```

```
DT_grid  = RandomizedSearchCV(DecisionTreeClassifier(), param_distributions = parameters, cv = 5, verbose = True)
```

```
DT_grid.fit(x_train,y_train)
```

```
DT_grid.fit(x_train,y_train)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
```