# PROJECT DEVELOPMENT PHASE

## SPRINT 1

| | |
|---|---|
| **Date** | 16 November 2022 |
| **Team ID** | PNT2022TMID01346 |
| **Project Name** | Developing a Flight Delay Prediction Model using Machine Learning |
| **Marks** | 8 marks |

## DATA PREPROCESSING AND MODEL BUILDING:

In this Sprint development phase, we have create an model with the help of Pre-processed dataset. We have used Decision Tree Classifier Algorithm for model development. Also we have implement method to check the accuracy of our model and convert the model into pkl file by importing Pickle python library. With the help of pickle model file the prediction is performed by Flask App.

## Jupyter Screenshots:

## Importing Libraries:

```
import sys
import numpy as np #Linear Algebra
import pandas as pd #Data Processing
import seaborn as sns #Data Visualizaton.
import pickle
%matplotlib inline
from matplotlib import pyplot as plt
from sklearn.preprocessing import LabelEncoder #LabelEncoding From Sklearm
from sklearn.preprocessing import OneHotEncoder #One-Hot Encoding From Sklearn
from sklearn.model_selection import train_test_split #split Data in Train & Test Array
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier#ml Algorithm
from sklearn.metrics import accuracy_score #Calculate Accuracy Score
import sklearn.metrics as metrics #Confusion Matrix
```

## Dataset:

```
dataset= pd.read_csv("/content/flightdata.csv")
```

## Analyze the data:

```
dataset.info()
```

```
RangeIndex: 11231 entries, 0 to 11230
Data columns (total 26 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   YEAR                 11231 non-null  int64
 1   QUARTER              11231 non-null  int64
 2   MONTH                11231 non-null  int64
 3   DAY_OF_MONTH         11231 non-null  int64
 4   DAY_OF_WEEK          11231 non-null  int64
 5   UNIQUE_CARRIER       11231 non-null  object
 6   TAIL_NUM             11231 non-null  object
 7   FL_NUM               11231 non-null  int64
 8   ORIGIN_AIRPORT_ID    11231 non-null  int64
 9   ORIGIN               11231 non-null  object
 10  DEST_AIRPORT_ID      11231 non-null  int64
 11  DEST                 11231 non-null  object
 12  CRS_DEP_TIME         11231 non-null  int64
 13  DEP_TIME             11124 non-null  float64
 14  DEP_DELAY            11124 non-null  float64
 15  DEP_DEL15            11124 non-null  float64
 16  CRS_ARR_TIME         11231 non-null  int64
 17  ARR_TIME             11116 non-null  float64
 18  ARR_DELAY            11043 non-null  float64
 19  ARR_DEL15            11043 non-null  float64
 20  CANCELLED            11231 non-null  float64
 21  DIVERTED             11231 non-null  float64
 22  CRS_ELAPSED_TIME     11231 non-null  float64
```

```
 14  DEP_DELAY            11124 non-null  float64
 15  DEP_DEL15            11124 non-null  float64
 16  CRS_ARR_TIME         11231 non-null  int64
 17  ARR_TIME             11116 non-null  float64
 18  ARR_DELAY            11043 non-null  float64
 19  ARR_DEL15            11043 non-null  float64
 20  CANCELLED            11231 non-null  float64
 21  DIVERTED             11231 non-null  float64
 22  CRS_ELAPSED_TIME     11231 non-null  float64
 23  ACTUAL_ELAPSED_TIME  11043 non-null  float64
 24  DISTANCE             11231 non-null  float64
 25  Unnamed: 25          0 non-null      float64
dtypes: float64(12), int64(10), object(4)
memory usage: 2.2+ MB
```

```
dataset.describe()
```

| | YEAR | QUARTER | MONTH | DAY_OF_MONTH | DAY_OF_WEEK | FL_NUM | ORIGIN_AIRPORT_ID | DEST_AIRPORT_ID | CRS_DEP_TIME | DEP_T |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 11231.0 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 | 11124.000 |
| **mean** | 2016.0 | 2.544475 | 6.628973 | 15.790758 | 3.960199 | 1334.325617 | 12334.516695 | 12302.274508 | 1320.798326 | 1327.189 |
| **std** | 0.0 | 1.090701 | 3.354678 | 8.782056 | 1.995257 | 811.875227 | 1595.026510 | 1601.988550 | 490.737845 | 500.306 |
| **min** | 2016.0 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 7.000000 | 10397.000000 | 10397.000000 | 10.000000 | 1.000 |
| **25%** | 2016.0 | 2.000000 | 4.000000 | 8.000000 | 2.000000 | 624.000000 | 10397.000000 | 10397.000000 | 905.000000 | 905.000 |
| **50%** | 2016.0 | 3.000000 | 7.000000 | 16.000000 | 4.000000 | 1267.000000 | 12478.000000 | 12478.000000 | 1320.000000 | 1324.000 |
| **75%** | 2016.0 | 3.000000 | 9.000000 | 23.000000 | 6.000000 | 2032.000000 | 13487.000000 | 13487.000000 | 1735.000000 | 1739.000 |
| **max** | 2016.0 | 4.000000 | 12.000000 | 31.000000 | 7.000000 | 2853.000000 | 14747.000000 | 14747.000000 | 2359.000000 | 2400.000 |

8 rows × 22 columns

## Handling Missing Values:

```
dataset.isnull().sum()
```

```
YEAR                      0
QUARTER                   0
MONTH                     0
DAY_OF_MONTH              0
DAY_OF_WEEK               0
UNIQUE_CARRIER            0
TAIL_NUM                  0
FL_NUM                    0
ORIGIN_AIRPORT_ID         0
ORIGIN                    0
DEST_AIRPORT_ID           0
DEST                      0
CRS_DEP_TIME              0
DEP_TIME                107
DEP_DELAY               107
DEP_DEL15               107
CRS_ARR_TIME              0
ARR_TIME                115
ARR_DELAY               188
ARR_DEL15               188
CANCELLED                 0
DIVERTED                  0
CRS_ELAPSED_TIME          0
ACTUAL_ELAPSED_TIME     188
DISTANCE                  0
Unnamed: 25           11231
dtype: int64
```

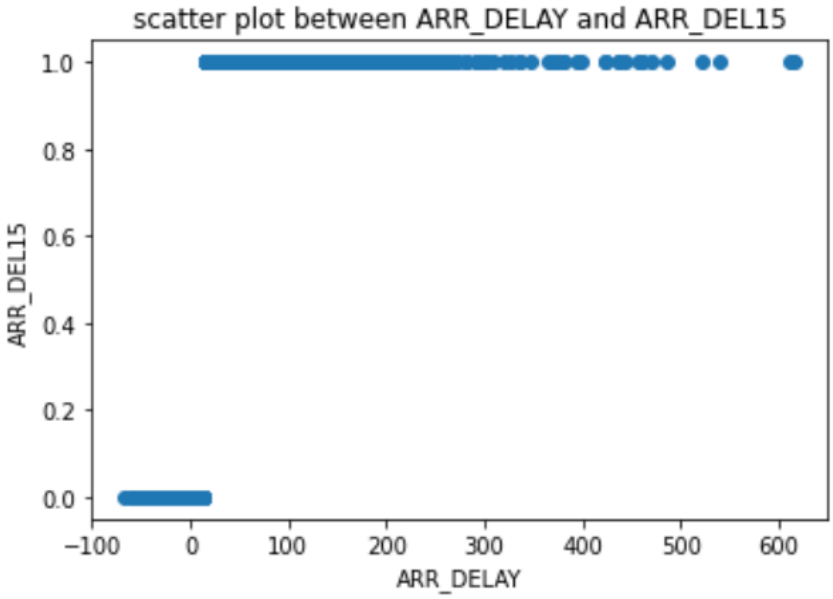
```
dataset['DEST'].unique()
```

```
array(['SEA', 'MSP', 'DTW', 'ATL', 'JFK'], dtype=object)
```

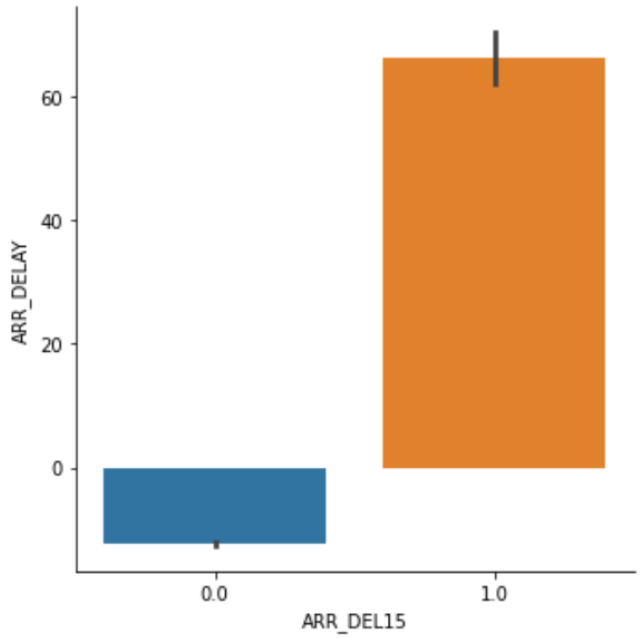
## Data Visualization:

```
plt.scatter(dataset['ARR_DELAY'],dataset['ARR_DEL15'])
plt.xlabel('ARR_DELAY')
plt.ylabel('ARR_DEL15')
plt.title('scatter plot between ARR_DELAY and ARR_DEL15')
```

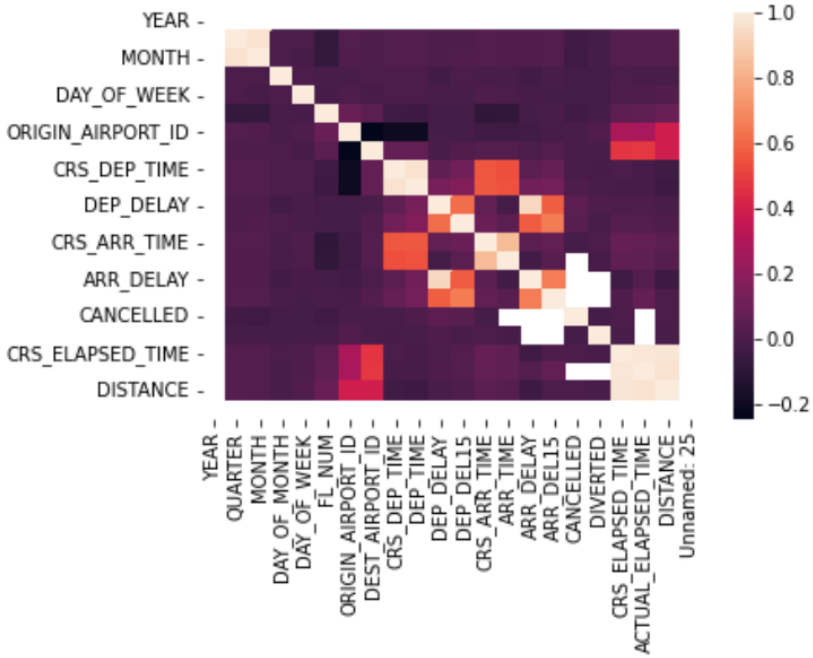
```
Text(0.5, 1.0, 'scatter plot between ARR_DELAY and ARR_DEL15')
```



```
sns.catplot(x="ARR_DEL15",y="ARR_DELAY",kind='bar',data=dataset)
```




```
sns.heatmap(dataset.corr())
```



## Dropping Un Necessary Columns:

```
dataset=dataset.drop('Unnamed: 25',axis=1)
dataset.isnull().sum()
```

```
YEAR                     0
QUARTER                  0
MONTH                    0
DAY_OF_MONTH             0
DAY_OF_WEEK              0
UNIQUE_CARRIER           0
TAIL_NUM                 0
FL_NUM                   0
ORIGIN_AIRPORT_ID        0
ORIGIN                   0
DEST_AIRPORT_ID          0
DEST                     0
CRS_DEP_TIME             0
DEP_TIME               107
DEP_DELAY              107
DEP_DEL15              107
CRS_ARR_TIME             0
ARR_TIME               115
ARR_DELAY              188
ARR_DEL15              188
CANCELLED                0
DIVERTED                 0
CRS_ELAPSED_TIME         0
ACTUAL_ELAPSED_TIME    188
DISTANCE                 0
dtype: int64
```

```
dataset=dataset[["FL_NUM","MONTH","DAY_OF_MONTH","DAY_OF_WEEK","ORIGIN","DEST","CRS_ARR_TIME","DEP_DEL15","ARR_DEL15"]]
dataset.isnull().sum()
```

```
FL_NUM           0
MONTH            0
DAY_OF_MONTH     0
DAY_OF_WEEK      0
ORIGIN           0
DEST             0
CRS_ARR_TIME     0
DEP_DEL15      107
ARR_DEL15      188
dtype: int64
```

```
dataset=dataset.fillna({'ARR_DEL15':1})
dataset=dataset.fillna({'DEP_DEL15':0})
dataset.iloc[177:185]
```

| | FL_NUM | MONTH | DAY_OF_MONTH | DAY_OF_WEEK | ORIGIN | DEST | CRS_ARR_TIME | DEP_DEL15 | ARR_DEL15 |
|---|---|---|---|---|---|---|---|---|---|
| **177** | 2834 | 1 | 9 | 6 | MSP | SEA | 852 | 0.0 | 1.0 |
| **178** | 2839 | 1 | 9 | 6 | DTW | JFK | 1724 | 0.0 | 0.0 |
| **179** | 86 | 1 | 10 | 7 | MSP | DTW | 1632 | 0.0 | 1.0 |
| **180** | 87 | 1 | 10 | 7 | DTW | MSP | 1649 | 1.0 | 0.0 |
| **181** | 423 | 1 | 10 | 7 | JFK | ATL | 1600 | 0.0 | 0.0 |
| **182** | 440 | 1 | 10 | 7 | JFK | ATL | 849 | 0.0 | 0.0 |
| **183** | 485 | 1 | 10 | 7 | JFK | SEA | 1945 | 1.0 | 0.0 |
| **184** | 557 | 1 | 10 | 7 | MSP | DTW | 912 | 0.0 | 1.0 |

```
import math
for index, row in dataset.iterrows():
    dataset.loc[index,'CRS_ARR_TIME']=math.floor(row['CRS_ARR_TIME']/100)
dataset.head()
```

| | FL_NUM | MONTH | DAY_OF_MONTH | DAY_OF_WEEK | ORIGIN | DEST | CRS_ARR_TIME | DEP_DEL15 | ARR_DEL15 |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1399 | 1 | 1 | 5 | ATL | SEA | 21 | 0.0 | 0.0 |
| **1** | 1476 | 1 | 1 | 5 | DTW | MSP | 14 | 0.0 | 0.0 |
| **2** | 1597 | 1 | 1 | 5 | ATL | SEA | 12 | 0.0 | 0.0 |
| **3** | 1768 | 1 | 1 | 5 | SEA | MSP | 13 | 0.0 | 0.0 |
| **4** | 1823 | 1 | 1 | 5 | SEA | DTW | 6 | 0.0 | 0.0 |

## Label Encoding and One Hot Encoding:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
dataset['DEST']=le.fit_transform(dataset['DEST'])
dataset['ORIGIN']=le.fit_transform(dataset['ORIGIN'])
```

```
dataset.head(5)
```

| | FL_NUM | MONTH | DAY_OF_MONTH | DAY_OF_WEEK | ORIGIN | DEST | CRS_ARR_TIME | DEP_DEL15 | ARR_DEL15 |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1399 | 1 | 1 | 5 | 0 | 4 | 21 | 0.0 | 0.0 |
| **1** | 1476 | 1 | 1 | 5 | 1 | 3 | 14 | 0.0 | 0.0 |
| **2** | 1597 | 1 | 1 | 5 | 0 | 4 | 12 | 0.0 | 0.0 |
| **3** | 1768 | 1 | 1 | 5 | 4 | 3 | 13 | 0.0 | 0.0 |
| **4** | 1823 | 1 | 1 | 5 | 4 | 1 | 6 | 0.0 | 0.0 |

## Splitting The Dataset Into Dependent And Independent Variables:

```
dataset=pd.get_dummies(dataset,columns=['ORIGIN','DEST'])
dataset.head()
```

| | FL_NUM | MONTH | DAY_OF_MONTH | DAY_OF_WEEK | CRS_ARR_TIME | DEP_DEL15 | ARR_DEL15 | ORIGIN_0 | ORIGIN_1 | ORIGIN_2 | ORIGIN_3 | ORIGIN_4 | DEST |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1399 | 1 | 1 | 5 | 21 | 0.0 | 0.0 | 1 | 0 | 0 | 0 | 0 | |
| **1** | 1476 | 1 | 1 | 5 | 14 | 0.0 | 0.0 | 0 | 1 | 0 | 0 | 0 | |
| **2** | 1597 | 1 | 1 | 5 | 12 | 0.0 | 0.0 | 1 | 0 | 0 | 0 | 0 | |
| **3** | 1768 | 1 | 1 | 5 | 13 | 0.0 | 0.0 | 0 | 0 | 0 | 0 | 1 | |
| **4** | 1823 | 1 | 1 | 5 | 6 | 0.0 | 0.0 | 0 | 0 | 0 | 0 | 1 | |

```
x=dataset.iloc[:,0:8].values
y=dataset.iloc[:,8:9].values
```

```
from sklearn.preprocessing import OneHotEncoder
oh=  OneHotEncoder()
z=oh.fit_transform(x[:,4:5]).toarray()
t=oh.fit_transform(x[:,5:6]).toarray()
```

```
z
```

```
array([[0., 0., 0., ..., 1., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

```
t
```

```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]])
```

## Split The Dataset Into Train Set And Test Set:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_test.shape
```

```
(2247, 8)
```

```
x_train.shape
```

```
(8984, 8)
```

```
y_test.shape
```

```
(2247, 1)
```

```
y_train.shape
```

```
(8984, 1)
```

```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
x_train= sc.fit_transform(x_train)
x_test=sc.transform(x_test)
```

```
!pip install imblearn
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: imblearn in /usr/local/lib/python3.7/dist-packages (0.0)
Requirement already satisfied: imbalanced-learn in /usr/local/lib/python3.7/dist-packages (from imblearn) (0.8.1)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from imbalanced-learn->imblearn) (1.2.0)
Requirement already satisfied: scipy>=0.19.1 in /usr/local/lib/python3.7/dist-packages (from imbalanced-learn->imblearn) (1.7.3)
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.7/dist-packages (from imbalanced-learn->imblearn) (1.21.6)
Requirement already satisfied: scikit-learn>=0.24 in /usr/local/lib/python3.7/dist-packages (from imbalanced-learn->imblearn) (1.0.
2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn>=0.24->imbalanced-l
earn->imblearn) (3.1.0)
```

```
import imblearn
```

```
from imblearn.over_sampling import SMOTE
smote=SMOTE()
```

```
x_train_smote,y_train_smote =smote.fit_resample(x_train,y_train)
```

```
from sklearn.tree import DecisionTreeClassifier
classifier =DecisionTreeClassifier(random_state=0)
classifier.fit(x_train_smote,y_train_smote)
```

```
DecisionTreeClassifier(random_state=0)
```

```
decisiontree = classifier.predict(x_test)
```

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

```
0.9706275033377837
```

```python
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,decisiontree)
```

```python
cm
```

```
array([[1758,   44],
       [  22,  423]])
```

```python
import pickle
pickle.dump(classifier,open('flight.pkl','wb'))
```