DATA PREPROCESSING AND MODEL BUILDING

Date	12 November 2022
Team ID	PNT2022TMID13084
Project Name	Flight Delay Prediction Using Machine Learning

DEVELOPMENT PHASE:

Outline:

1. Data Pre-processing
2. Data Analysis
3. Model Building
4. Saving Best Model

Required Libraries:

- Pandas - Data Pre-processing
- Numpy - Data Pre-processing, Analysis
- Matplotlib - Visualization
- Seaborn - Visualization
- Sklearn - Model Building
- Pickle - Model saving

Software/Tool:

- Google colab
- Used Language Python

Data Pre-processing:

Data Collection:

Dataset is collected from the IBM career smartinternz portal in Guided Project.

Dataset description:

The dataset contains 26 variables with various data types such as string, object, time, integer, float.

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
23	ACTUAL_ELAPSED_TIME	11043 non-null	float64
24	DISTANCE	11231 non-null	float64
25	Unnamed: 25	0 non-null	float64

Columns Description:

Dest means Destination Airport.

Crs_dep_time and crs_arr_time is planned departure and arrival time.

Crs_elapsed _time is estimated travel time as per plan.

Arr_time and dep_time are actual arrival and departure time.

Actual_elapsed_time is actual travelled time

To pre-process our dataset, we need to import above mentioned required libraries, then import data using pandas.

This data does not contain any duplicated values and null values except in arrival , departure time columns, because these left empty when flights are cancelled.

Descriptive Analytics:

```
dataset.describe()
```

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	FL_NUM	ORIGIN_AIRPORT_ID	DEST_AIRPORT_ID	CRS_DEP_TIME	DEP_TIME	...	CRS_ARR_TIME	ARR_TIME	1
count	11231.0	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11124.000000	...	11231.000000	11116.000000	
mean	2016.0	2.544475	6.628973	15.790758	3.960199	1334.325617	12334.516695	12302.274508	1320.798326	1327.189410	...	1537.312795	1523.978499	
std	0.0	1.090701	3.354678	8.782056	1.995257	811.875227	1595.026510	1601.988550	490.737845	500.306462	...	502.512494	512.536041	
min	2016.0	1.000000	1.000000	1.000000	1.000000	7.000000	10397.000000	10397.000000	10.000000	1.000000	...	2.000000	1.000000	
25%	2016.0	2.000000	4.000000	8.000000	2.000000	624.000000	10397.000000	10397.000000	905.000000	905.000000	...	1130.000000	1135.000000	
50%	2016.0	3.000000	7.000000	16.000000	4.000000	1267.000000	12478.000000	12478.000000	1320.000000	1324.000000	...	1559.000000	1547.000000	
75%	2016.0	3.000000	9.000000	23.000000	6.000000	2032.000000	13487.000000	13487.000000	1735.000000	1739.000000	...	1952.000000	1945.000000	
max	2016.0	4.000000	12.000000	31.000000	7.000000	2853.000000	14747.000000	14747.000000	2359.000000	2400.000000	...	2359.000000	2400.000000	

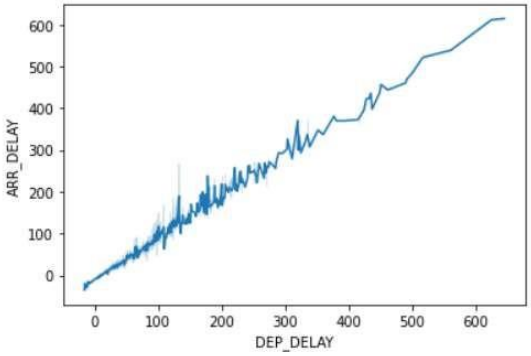
8 rows × 22 columns

AIRPORT_ID	CRS_DEP_TIME	DEP_TIME	...	CRS_ARR_TIME	ARR_TIME	ARR_DELAY	ARR_DEL15	CANCELLED	DIVERTED	CRS_ELAPSED_TIME	ACTUAL_ELAPSED_TIME	DISTANCE	Unnamed: 25
231.000000	11231.000000	11124.000000	...	11231.000000	11116.000000	11043.000000	11043.000000	11231.000000	11231.000000	11231.000000	11043.000000	11231.000000	0.0
302.274508	1320.798326	1327.189410	...	1537.312795	1523.978499	-2.573123	0.124513	0.010150	0.006589	190.652124	179.661233	1161.031965	NaN
601.988550	490.737845	500.306462	...	502.512494	512.536041	39.232521	0.330181	0.100241	0.080908	78.386317	77.940399	643.683379	NaN
397.000000	10.000000	1.000000	...	2.000000	1.000000	-67.000000	0.000000	0.000000	0.000000	93.000000	75.000000	509.000000	NaN
397.000000	905.000000	905.000000	...	1130.000000	1135.000000	-19.000000	0.000000	0.000000	0.000000	127.000000	117.000000	594.000000	NaN
478.000000	1320.000000	1324.000000	...	1559.000000	1547.000000	-10.000000	0.000000	0.000000	0.000000	159.000000	149.000000	907.000000	NaN
487.000000	1735.000000	1739.000000	...	1952.000000	1945.000000	1.000000	0.000000	0.000000	0.000000	255.000000	236.000000	1927.000000	NaN
747.000000	2359.000000	2400.000000	...	2359.000000	2400.000000	615.000000	1.000000	1.000000	1.000000	397.000000	428.000000	2422.000000	NaN

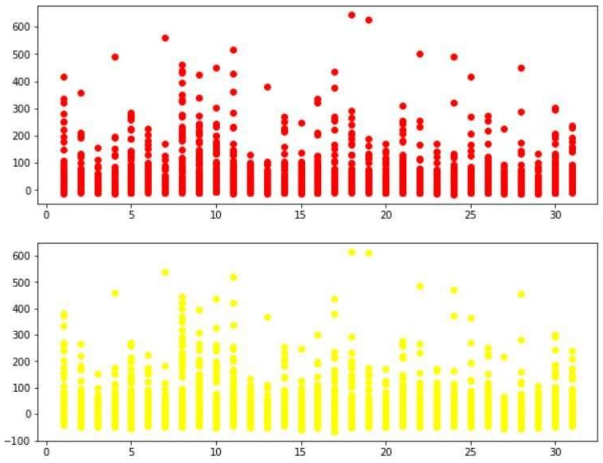
Data Analysis And Visualization:

```
sns.lineplot(x="DEP_DELAY",y="ARR_DELAY",data=data1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd957145710>
```

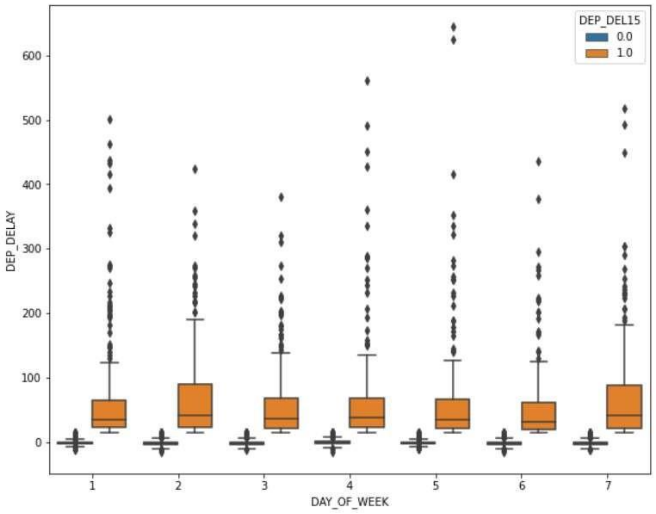


```
[31] plt.figure(figsize=(10,8))
     plt.subplot(2,1,1)
     plt.scatter(data1["DAY_OF_MONTH"],data1["DEP_DELAY"],color="red")
     plt.subplot(2,1,2)
     plt.scatter(data1["DAY_OF_MONTH"],data1["ARR_DELAY"],color="yellow")
     plt.show()
```

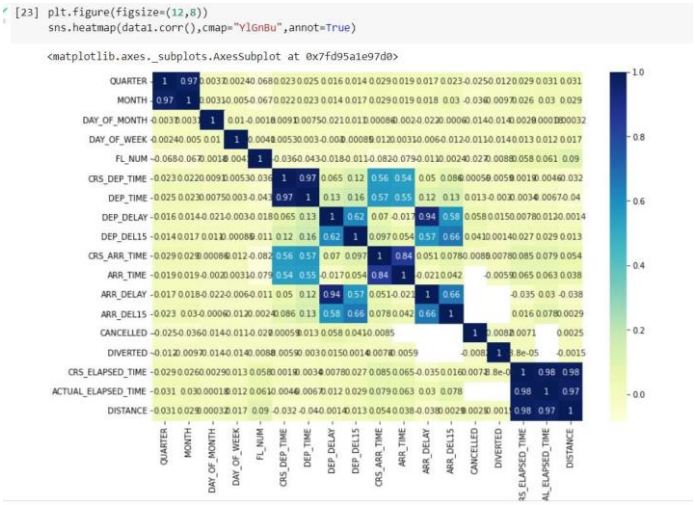



```
[32] plt.figure(figsize=(10,8))
     sns.boxplot(x="DAY_OF_WEEK",y="DEP_DELAY",data=data1,hue="DEP_DEL15")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd955856790>
```



Correlation between columns:



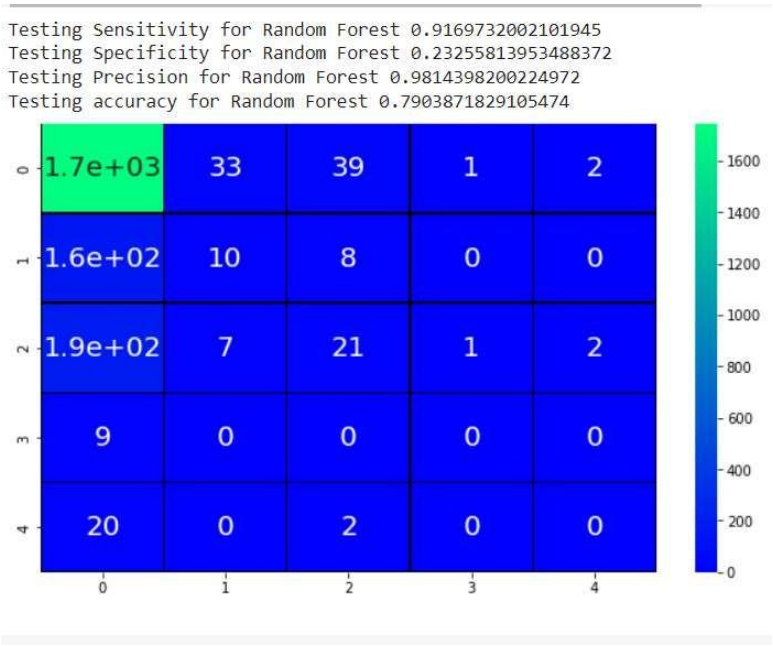
Model Buliding:

We builded

Decision Tree with 0.6875834445927904
Random Forest with 0.7903871829105474
SVM with 0.7601246105919003
KNN with 0.7900474855362706

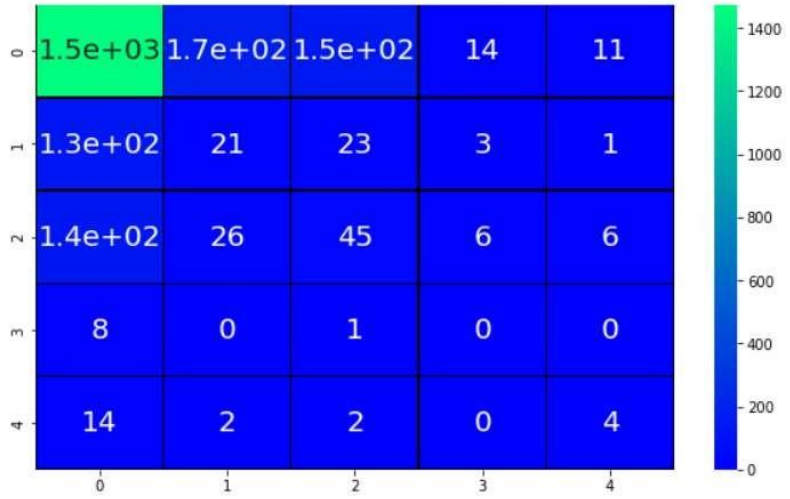
We will explore Random Forest and Decision Tree

Random Forest:



Decision Tree:

```
Testing Accuracy for Decision Tree 0.8320355951056729
Testing Sensitivity for Decision Tree 0.9201497192763568
Testing Specificity for Decision Tree 0.1076923076923077
Testing Precision for Decision Tree 0.8944815039417829
Testing accuracy for Decision Tree 0.6875834445927904
```



Model Saving:Random Forest gives the best accuracy then others , so we save random forest model using pickle.

```
[101] import pickle

pickle.dump(dc,open("dcmodel.pkl",'wb'))
```

Conclusion: In this sprint , we builded our model , evaluated and saved. In next sprint, we deploy our model IBM cloud using IBM Watson and building Dashboard.