PROJECT DEVELOPMENT PHASE

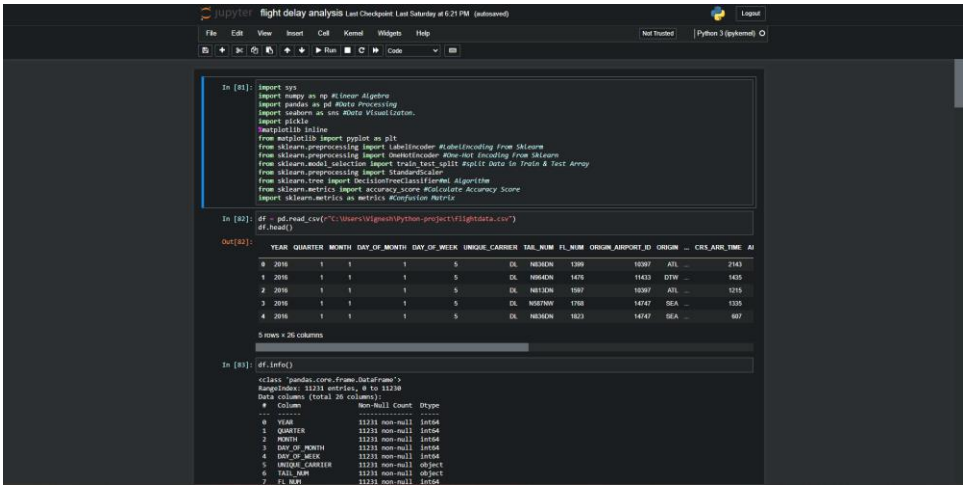
SPRINT 2 – CODE AND TESTCASE

Date	10 November 2022
Team ID	PNT2022TMID02840
Project	Flight delay prediction using Machine learning
Marks	8 Marks

We have performed the uploading the Dataset and performed the Data Pre-processing and also we have split the dataset into train data and Test dataset in this Sprint development phase.

Jupyter notebook :

Screenshots :



jupyter flight delay analysis Last Checkpoint: Last Saturday at 6:21 PM (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

```
In [84]: df.describe()
```

Out[84]:

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	FL_NUM	ORIGIN_AIRPORT_ID	DEST_AIRPORT_ID	CRS_DEP_TIME	DEP_
count	11231.0	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11124.00
mean	2016.0	2.544475	6.628873	15.790758	3.960199	1334.325917	12334.516995	12302.274508	1320.798326	1327.18
std	0.0	1.096701	3.354678	8.782056	1.995257	811.875227	1595.026510	1601.968550	498.737845	500.38
min	2016.0	1.000000	1.000000	1.000000	1.000000	7.000000	10397.000000	10397.000000	10.000000	1.00
25%	2016.0	2.000000	4.000000	8.000000	2.000000	624.000000	10397.000000	10397.000000	905.000000	905.00
50%	2016.0	3.000000	7.000000	16.000000	4.000000	1267.000000	12478.000000	12478.000000	1320.000000	1324.00
75%	2016.0	3.000000	9.000000	23.000000	6.000000	2032.000000	13487.000000	13487.000000	1735.000000	1739.00
max	2016.0	4.000000	12.000000	31.000000	7.000000	2853.000000	14747.000000	14747.000000	2359.000000	2400.00

8 rows × 22 columns

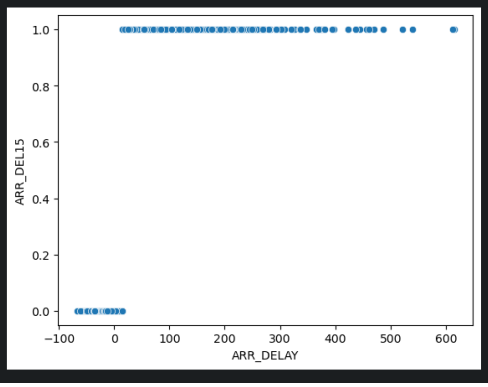
```
In [85]: df.isnull().sum()
```

Out[85]:

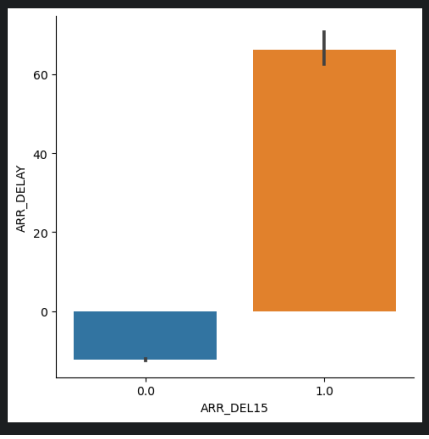
```
YEAR                    0
QUARTER                 0
MONTH                   0
DAY_OF_MONTH            0
DAY_OF_WEEK             0
UNIQUE_CARRIER          0
TAIL_NUM                0
FL_NUM                  0
ORIGIN_AIRPORT_ID       0
ORIGIN                  0
DEST_AIRPORT_ID         0
DEST                    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
```

```
In [86]: sns.scatterplot(x='ARR_DELAY',y='ARR_DEL15',data=df)
```

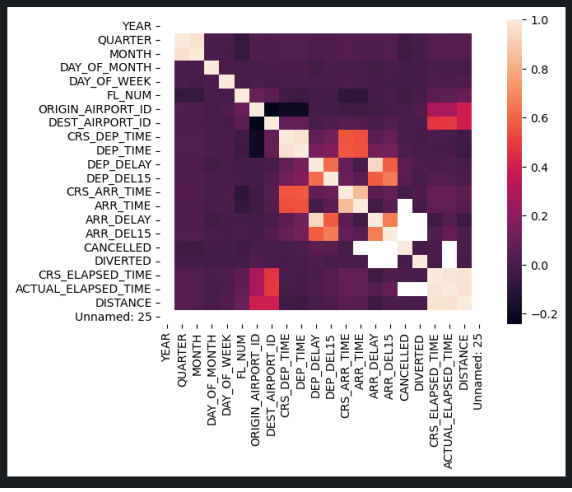
Out[86]: <AxesSubplot:xlabel='ARR_DELAY', ylabel='ARR_DEL15'>



```
In [87]: sns.catplot(x='ARR_DEL15',y='ARR_DELAY',kind='bar',data=df)
Out[87]: <seaborn.axisgrid.FacetGrid at 0x20cec7120a0>
```



```
In [88]: sns.heatmap(df.corr())
Out[88]: <AxesSubplot:>
```




```
In [89]: df = df.drop('Unnamed: 25',axis=1)
         df.isnull().sum()

Out[89]: 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

In [90]: df = df[["FL_NUM", "MONTH", "DAY_OF_MONTH", "DAY_OF_WEEK", "ORIGIN", "DEST", "CRS_ARR_TIME","DEP_DEL15", "ARR_DEL15"]]
         df.isnull().sum()

Out[90]: 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
```

In [91]:
```
df = df.fillna ({'ARR_DEL15': 1})
df = df.fillna ({'DEP_DEL15': 0})
df.iloc[177:185]
```

Out[91]:

	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

In [92]:
```
import math
for index, row in df.iterrows():
    df.loc[index, 'CRS_ARR_TIME'] = math.floor(row['CRS_ARR_TIME'] / 100)
df.head()
```

Out[92]:

	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

In [93]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['DEST'] = le.fit_transform(df ['DEST'])
df['ORIGIN'] =le.fit_transform(df ['ORIGIN'])
df.head(5)
```

Out[93]:

	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

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

In [ ]:
```
z
```

In [ ]:
```
t
```

In [97]:
```
x=df.iloc[:, 4:5].values
y=df.iloc[:, 5:6].values
x.shape
```

Out[97]:
```
(11231, 1)
```

In [98]:
```
y
```

Out[98]:
```
array([[4],
       [3],
       [4],
       ...,
       [4],
       [4],
       [1]])
```

In [99]:
```
x=df.iloc[:, 0:8].values
y=df.iloc[:, 8:9].values
df.head()
```

Out[99]:

	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

In [100]:
```
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
```

Out[100]: (2247, 8)

In [101]:
```
x_test.shape
```

Out[101]: (2247, 8)

In [102]:
```
x_train.shape
```

Out[102]: (8984, 8)

In [103]:
```
y_test.shape
```

Out[103]: (2247, 1)

In [104]:
```
y_train.shape
```

Out[104]: (8984, 1)