MINI PROJECT

Problem Statement: To determine which model best fits for the given data frame

LINEAR REGRESSION

In [98]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
```

In [99]:
```
train_df=pd.read_csv(r"C:\Users\rubin\Documents\Data_Train1.csv")
train_df
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Dura
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? NAG ? BLR	18:05	23:30	5h
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR ? NAG	16:50	21:35	4h

In [100]:
```
test_df=pd.read_csv(r"C:\Users\rubin\Documents\Test_set26.csv")
test_df
```

Out[100]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL ? BOM ? COK	17:30	04:25 07 Jun	10h 55m
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? MAA ? BLR	06:20	10:20	4h
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	19:15	19:00 22 May	23h 45m
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	08:00	21:00	13h
4	Air Asia	24/06/2019	Banglore	Delhi	BLR ? DEL	23:55	02:45 25 Jun	2h 50m
...	...	...	...	...	...	...	...	...
2666	Air India	6/06/2019	Kolkata	Banglore	CCU ? DEL ? BLR	20:30	20:25 07 Jun	23h 55m
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU ? BLR	14:20	16:55	2h 35m
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	21:50	04:25 07 Mar	6h 35m
2669	Air India	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	04:00	19:15	15h 15m
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL ? BOM ? COK	04:55	19:15	14h 20m

2671 rows × 10 columns

In [101]:
```
train_df.head()
```

Out[101]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	To
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h 50m	
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m	
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h	
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? NAG ? BLR	18:05	23:30	5h 25m	
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR ? NAG ? DEL	16:50	21:35	4h 45m	

In [102]:
```
test_df.head()
```

Out[102]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	To
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL ? BOM ? COK	17:30	04:25 07 Jun	10h 55m	
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? MAA ? BLR	06:20	10:20	4h	
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	19:15	19:00 22 May	23h 45m	
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	08:00	21:00	13h	
4	Air Asia	24/06/2019	Banglore	Delhi	BLR ? DEL	23:55	02:45 25 Jun	2h 50m	

In [103]:
```
train_df.tail()
```

Out[103]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU ? BLR	19:55	22:25	2h 30m
10679	Air India	27/04/2019	Kolkata	Banglore	CCU ? BLR	20:45	23:20	2h 35m
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR ? DEL	08:20	11:20	3h
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR ? DEL	11:30	14:10	2h 40m
10682	Air India	9/05/2019	Delhi	Cochin	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 20m

In [104]:
```
test_df.tail()
```

Out[104]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
2666	Air India	6/06/2019	Kolkata	Banglore	CCU ? DEL ? BLR	20:30	20:25 07 Jun	23h 55m
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU ? BLR	14:20	16:55	2h 35m
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	21:50	04:25 07 Mar	6h 35m
2669	Air India	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	04:00	19:15	15h 15m
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL ? BOM ? COK	04:55	19:15	14h 20m

In [105]:
```
train_df.shape
```

Out[105]: (10683, 11)

In [106]:
```
test_df.shape
```

Out[106]: (2671, 10)

In [107]:
```
train_df.describe()
```

Out[107]:

	Price
count	10683.000000
mean	9087.064121
std	4611.359167
min	1759.000000
25%	5277.000000
50%	8372.000000
75%	12373.000000
max	79512.000000

In [108]:
```
test_df.describe()
```

Out[108]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
count	2671	2671	2671	2671	2671	2671	2671	2671
unique	11	44	5	6	100	199	704	320
top	Jet Airways	9/05/2019	Delhi	Cochin	DEL ? BOM ? COK	10:00	19:00	2h 50m
freq	897	144	1145	1145	624	62	113	122

In [109]:
```
train_df.duplicated().sum()
```

Out[109]:
```
220
```

In [110]:
```
test_df.duplicated().sum()
```

Out[110]:
```
26
```

In [111]:
```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Airline          10683 non-null  object
 1   Date_of_Journey  10683 non-null  object
 2   Source           10683 non-null  object
 3   Destination      10683 non-null  object
 4   Route            10682 non-null  object
 5   Dep_Time         10683 non-null  object
 6   Arrival_Time     10683 non-null  object
 7   Duration         10683 non-null  object
 8   Total_Stops      10682 non-null  object
 9   Additional_Info  10683 non-null  object
 10  Price            10683 non-null  int64 
dtypes: int64(1), object(10)
memory usage: 918.2+ KB
```

In [112]:
```
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2671 entries, 0 to 2670
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Airline          2671 non-null   object
 1   Date_of_Journey  2671 non-null   object
 2   Source           2671 non-null   object
 3   Destination      2671 non-null   object
 4   Route            2671 non-null   object
 5   Dep_Time         2671 non-null   object
 6   Arrival_Time     2671 non-null   object
 7   Duration         2671 non-null   object
 8   Total_Stops      2671 non-null   object
 9   Additional_Info  2671 non-null   object
dtypes: object(10)
memory usage: 208.8+ KB
```

In [113]:
```
train_df.fillna(method='ffill',inplace=True)
```

In [114]:
```
test_df.fillna(method='ffill',inplace=True)
```

In [115]:
```
x=np.array(train_df['Total_Stops']).reshape(-1,1)
y=np.array(train_df['Price']).reshape(-1,1)
```

In [116]:
```
x=np.array(test_df['Dep_Time']).reshape(-1,1)
y=np.array(test_df['Total_Stops']).reshape(-1,1)
```

In [117]:
```
train_df.dropna(inplace=True)
```

In [118]:
```
test_df.dropna(inplace=True)
```

In [119]:
```
train_df.isnull().sum()
```

Out[119]:
```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Additional_Info    0
Price              0
dtype: int64
```

In [120]:
```
test_df.isnull().sum()
```

Out[120]:
```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Additional_Info    0
dtype: int64
```

In [121]:
```
train_df.columns
```

Out[121]:
```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info', 'Price'],
      dtype='object')
```

In [122]:
```
test_df.columns
```

Out[122]:
```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info'],
      dtype='object')
```

In [123]:
```
convert={"Total_Stops":{"non-stop":1,"1 stop":2,"2 stops":3,"3 stops":4,"4 stc
train_df=train_df.replace(convert)
train_df
```

Out[123]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h 50m
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? NAG ? BLR	18:05	23:30	5h 25m
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR ? NAG ? DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	..
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU ? BLR	19:55	22:25	2h 30m
10679	Air India	27/04/2019	Kolkata	Banglore	CCU ? BLR	20:45	23:20	2h 35m
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR ? DEL	08:20	11:20	3h
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR ? DEL	11:30	14:10	2h 40m
10682	Air India	9/05/2019	Delhi	Cochin	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 20m

10683 rows × 11 columns

In [124]:
```
airline={"Airline":{"Jet Airways":0,"IndiGo":1,"Air India":2,"Multiple carrier
"SpiceJet":4,"Vistara":5,"Air Asia":6,"GoAir":7,
"Multiple carriers Premium economy":8,
"Jet Airways Business":9,"Vistara Premium economy":10,"Trujet":11}}
train_df=train_df.replace(airline)
train_df
```

Out[124]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	1	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h 50m
1	2	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m
2	0	9/06/2019	Delhi	Cochin	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h
3	1	12/05/2019	Kolkata	Banglore	CCU ? NAG ? BLR	18:05	23:30	5h 25m
4	1	01/03/2019	Banglore	New Delhi	BLR ? NAG ? DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	6	9/04/2019	Kolkata	Banglore	CCU ? BLR	19:55	22:25	2h 30m
10679	2	27/04/2019	Kolkata	Banglore	CCU ? BLR	20:45	23:20	2h 35m
10680	0	27/04/2019	Banglore	Delhi	BLR ? DEL	08:20	11:20	3h
10681	5	01/03/2019	Banglore	New Delhi	BLR ? DEL	11:30	14:10	2h 40m
10682	2	9/05/2019	Delhi	Cochin	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 20m

10683 rows × 11 columns

In [125]:
```
destination={"Destination":{"Cochin":0,"Banglore":1,"Delhi":2,"New Delhi":3,"H
train_df=train_df.replace(destination)
train_df
```

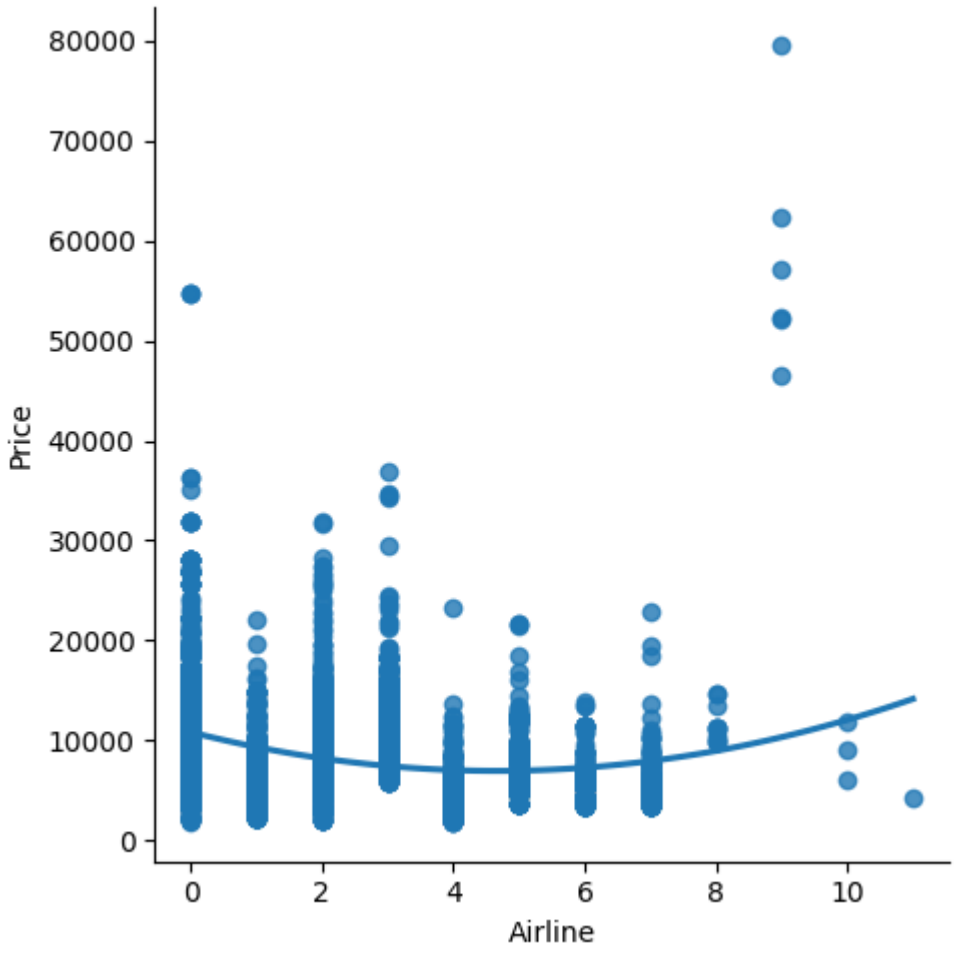
Out[125]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	1	24/03/2019	Banglore	3	BLR ? DEL	22:20	01:10 22 Mar	2h 50m
1	2	1/05/2019	Kolkata	1	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m
2	0	9/06/2019	Delhi	0	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h
3	1	12/05/2019	Kolkata	1	CCU ? NAG ? BLR	18:05	23:30	5h 25m
4	1	01/03/2019	Banglore	3	BLR ? NAG ? DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	6	9/04/2019	Kolkata	1	CCU ? BLR	19:55	22:25	2h 30m
10679	2	27/04/2019	Kolkata	1	CCU ? BLR	20:45	23:20	2h 35m
10680	0	27/04/2019	Banglore	2	BLR ? DEL	08:20	11:20	3h
10681	5	01/03/2019	Banglore	3	BLR ? DEL	11:30	14:10	2h 40m
10682	2	9/05/2019	Delhi	0	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 20m

10683 rows × 11 columns

In [126]:
```
sns.lmplot(x="Airline",y="Price",data=train_df,order=2,ci=None)
```

Out[126]:
```
<seaborn.axisgrid.FacetGrid at 0x2c96e320be0>
```

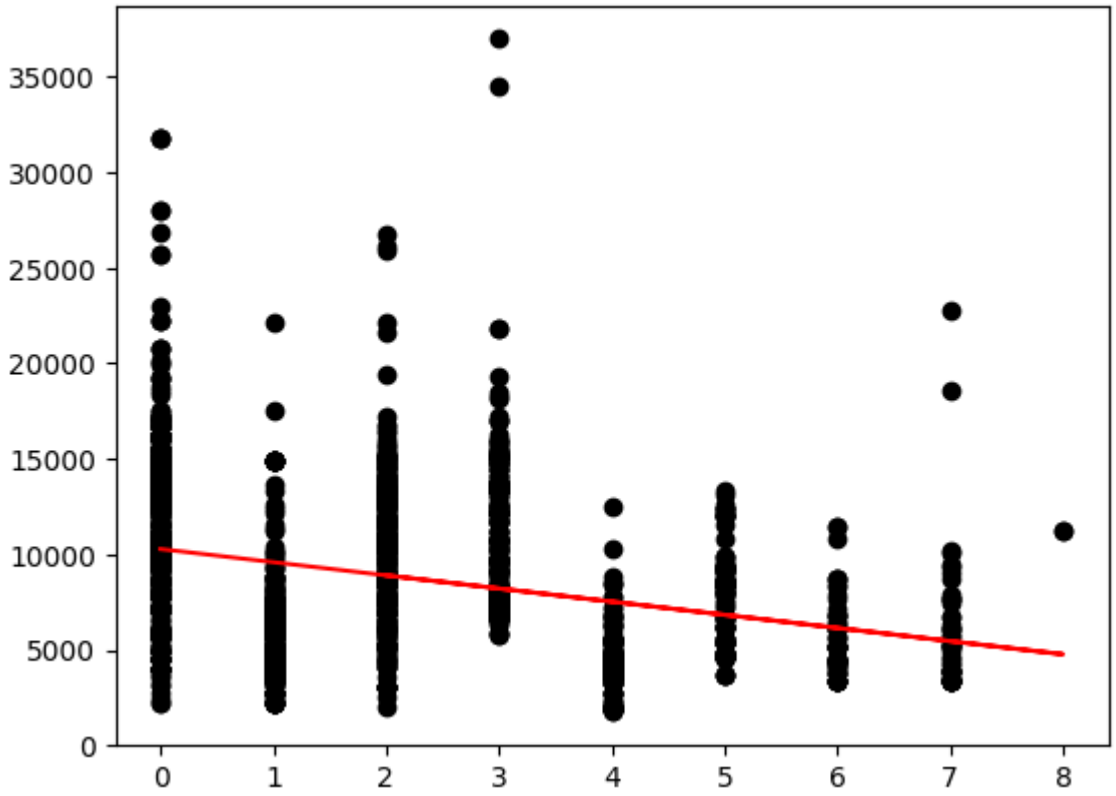


In [127]:
```
x=np.array(train_df['Airline']).reshape(-1,1)
y=np.array(train_df['Price']).reshape(-1,1)
```

In [128]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
0.09673876851846563
```

In [129]:
```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='black')
plt.plot(x_test,y_pred,color='r')
plt.show()
```



In [130]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("R2 Score:",r2)
```

```
R2 Score: 0.09673876851846563
```

Visualization

In [131]:
```
x=train_df[['Airline','Source','Destination','Total_Stops']]
y=train_df['Price']
```

In [132]:
```
airline={"Airline":{"Jet Airways":0,"IndiGo":1,"Air India":2,"Multiple carrier
"SpiceJet":4,"Vistara":5,"Air Asia":6,"GoAir":7,
"Multiple carriers Premium economy":8,
"Jet Airways Business":9,"Vistara Premium economy":10,"Trujet":11}}
test_df=test_df.replace(airline)
test_df
```

Out[132]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	0	6/06/2019	Delhi	Cochin	DEL ? BOM ? COK	17:30	04:25 07 Jun	10h 55m
1	1	12/05/2019	Kolkata	Banglore	CCU ? MAA ? BLR	06:20	10:20	4h
2	0	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	19:15	19:00 22 May	23h 45m
3	3	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	08:00	21:00	13h
4	6	24/06/2019	Banglore	Delhi	BLR ? DEL	23:55	02:45 25 Jun	2h 50m
...	...	...	...	...	...	...	...	...
2666	2	6/06/2019	Kolkata	Banglore	CCU ? DEL ? BLR	20:30	20:25 07 Jun	23h 55m
2667	1	27/03/2019	Kolkata	Banglore	CCU ? BLR	14:20	16:55	2h 35m
2668	0	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	21:50	04:25 07 Mar	6h 35m
2669	2	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	04:00	19:15	15h 15m
2670	3	15/06/2019	Delhi	Cochin	DEL ? BOM ? COK	04:55	19:15	14h 20m

2671 rows × 10 columns

In [133]:
```
convert={"Additional_Info":{"No info":1,"In-flight meal not included":2}}
train_df=train_df.replace(convert)
train_df
```

Out[133]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
0	1	24/03/2019	Banglore	3	BLR ? DEL	22:20	01:10 22 Mar	2h 50m
1	2	1/05/2019	Kolkata	1	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m
2	0	9/06/2019	Delhi	0	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h
3	1	12/05/2019	Kolkata	1	CCU ? NAG ? BLR	18:05	23:30	5h 25m
4	1	01/03/2019	Banglore	3	BLR ? NAG ? DEL	16:50	21:35	4h 45m
...	...	...	...	...	...	...	...	...
10678	6	9/04/2019	Kolkata	1	CCU ? BLR	19:55	22:25	2h 30m
10679	2	27/04/2019	Kolkata	1	CCU ? BLR	20:45	23:20	2h 35m
10680	0	27/04/2019	Banglore	2	BLR ? DEL	08:20	11:20	3h
10681	5	01/03/2019	Banglore	3	BLR ? DEL	11:30	14:10	2h 40m
10682	2	9/05/2019	Delhi	0	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 20m

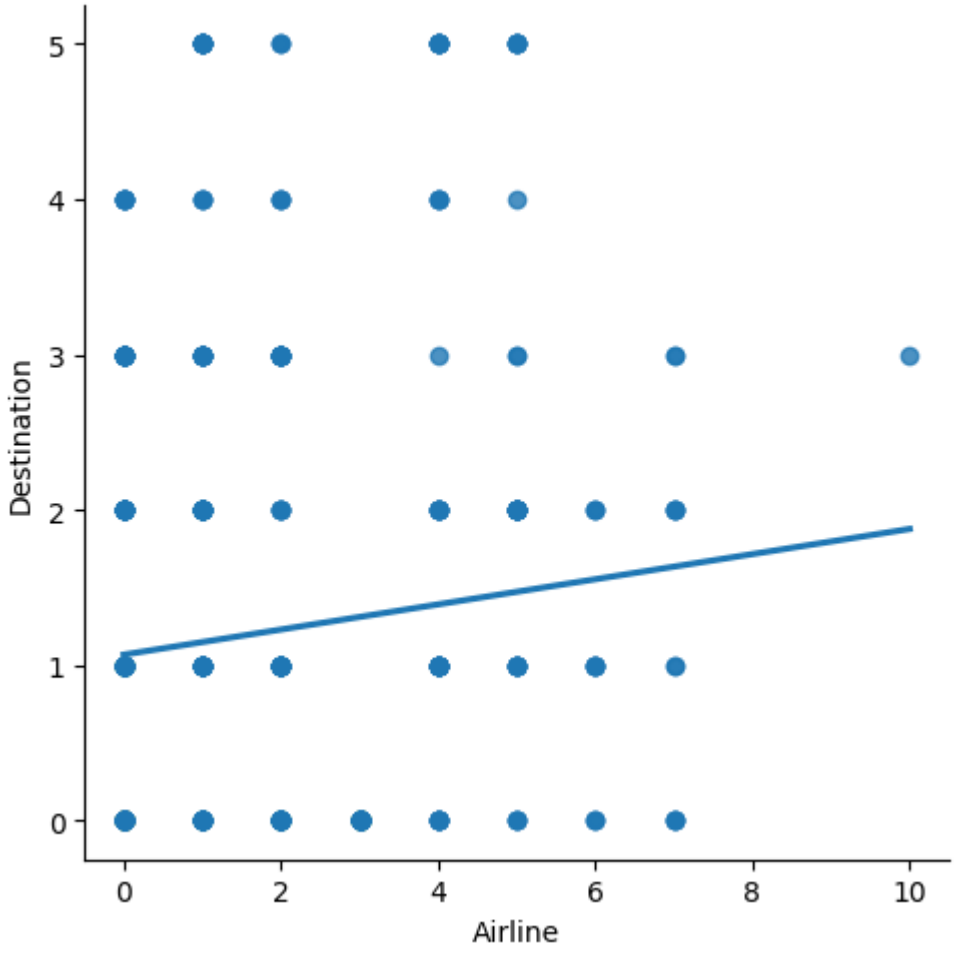
10683 rows × 11 columns

In [134]:
```
train_df=train_df[['Airline','Destination','Total_Stops','Price']]
sns.heatmap(train_df.corr(),annot=True)
```

Out[134]: <Axes: >

In [135]:
```
train_df500=train_df[:][:500]
sns.lmplot(x="Airline",y="Destination",data=train_df500,order=1,ci=None)
```

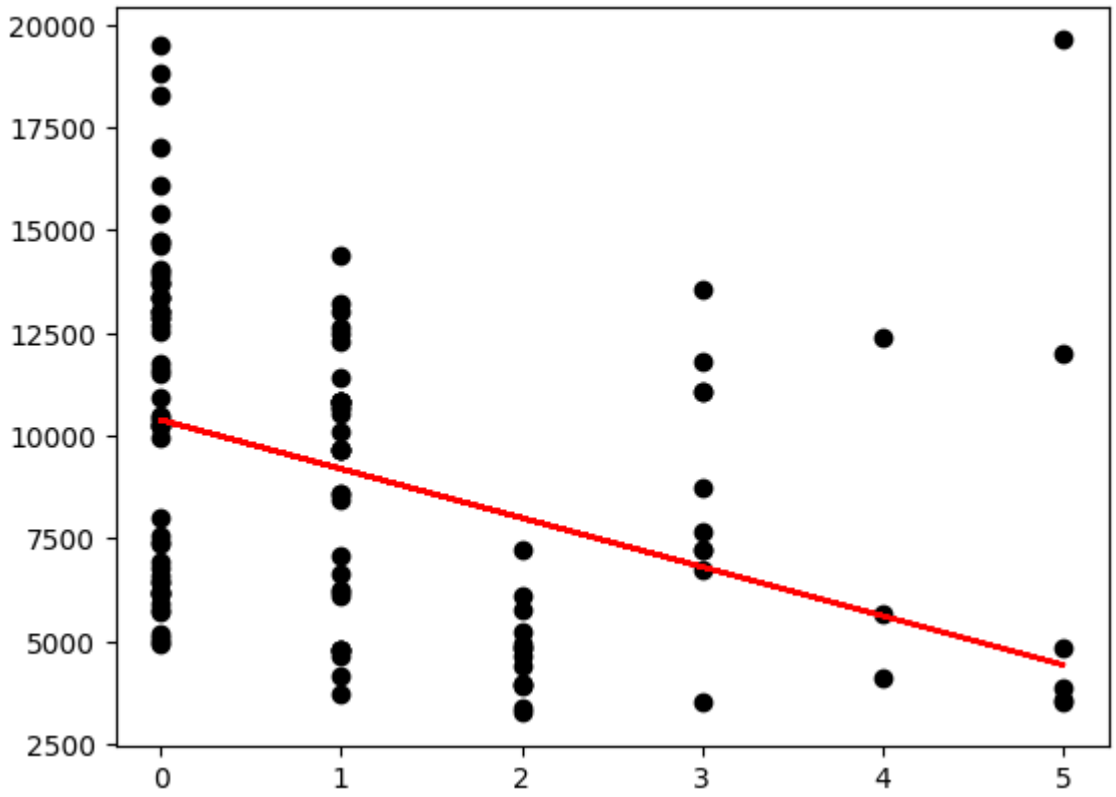
Out[135]: <seaborn.axisgrid.FacetGrid at 0x2c96487bc40>



In [136]:
```
train_df500=train_df[:][:500]
```

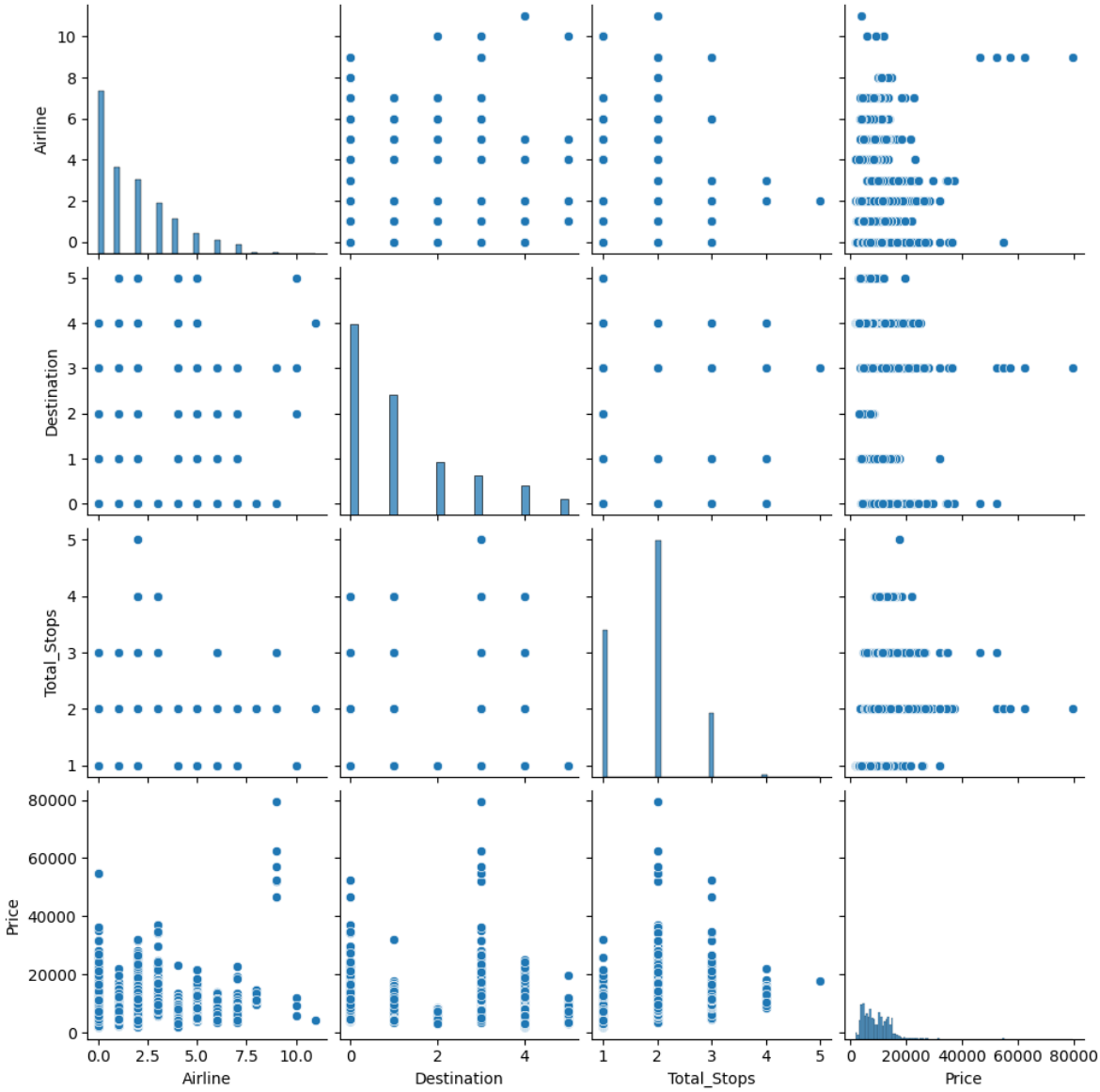
In [137]:
```
train_df500.fillna(method='ffill',inplace=True)
x=np.array(train_df500['Destination']).reshape(-1,1)
y=np.array(train_df500['Price']).reshape(-1,1)
train_df500.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression()
regr.fit(x_train,y_train)
print("Regression:",regr.score(x_test,y_test))
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='black')
plt.plot(x_test,y_pred,color='r')
plt.show()
```

```
Regression: 0.08256667152030595
```



In [138]:
```
sns.pairplot(train_df)
```

Out[138]: <seaborn.axisgrid.PairGrid at 0x2c96f6b7b50>



In [139]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("R2 Score:",r2)
```

```
R2 Score: 0.08256667152030595
```

In [140]:
```
train_df.isnull().sum()
```

Out[140]:
```
Airline        0
Destination    0
Total_Stops    0
Price          0
dtype: int64
```

ELASTIC NET REGRESSION

In [141]:
```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
regr.score(x,y)
```

```
[-919.85211142]
[10095.55046327]
```

Out[141]:
```
0.14206509261929623
```

In [142]:
```
y_pred_elastic=regr.predict(x_train)
```

In [143]:
```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
Mean Squared Error on test set 21873193.74657491
```

LOGISTIC REGRESSION

In [144]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [145]:
```
print('This DataFrame has %d Rows and %d columns'%(train_df.shape))
```

```
This DataFrame has 10683 Rows and 4 columns
```

In [146]:
```
x=np.array(train_df['Price']).reshape(-1,1)
y=np.array(train_df['Total_Stops']).reshape(-1,1)
train_df.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(max_iter=10000)
```

```
C:\Users\rubin\AppData\Local\Temp\ipykernel_17248\1798174072.py:3: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  train_df.dropna(inplace=True)
```

In [147]:
```
lr.fit(x_train,y_train)
```

```
C:\Users\rubin\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[147]:
```
LogisticRegression
LogisticRegression(max_iter=10000)
```

In [148]:
```
score=lr.score(x_test,y_test)
print(score)
```

```
0.7101404056162246
```

Decision Tree

In [149]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [150]:
```
train_df['Price'].value_counts()
```

Out[150]:
```
Price
10262    258
10844    212
7229     162
4804     160
4823     131
        ...
14153      1
8488       1
7826       1
6315       1
12648      1
Name: count, Length: 1870, dtype: int64
```

In [151]:
```
train_df['Total_Stops'].value_counts()
```

Out[151]:
```
Total_Stops
2    5625
1    3492
3    1520
4      45
5       1
Name: count, dtype: int64
```

In [152]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [153]:
```
clf.fit(x_train,y_train)
```

Out[153]:
```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [154]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.9351014040561623
```

Random Forest

In [155]:
```
import pandas as pd
import numpy as ny
import matplotlib.pyplot as plt,seaborn as sns
```

In [156]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
C:\Users\rubin\AppData\Local\Temp\ipykernel_17248\2210184639.py:3: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples,), for example using ravel().
  rfc.fit(x_train,y_train)
```

Out[156]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [157]:
```
rf=RandomForestClassifier()
```

In [158]:
```
params={'max_depth':[2,3,5,10,20],'min_samples_leaf':[5,10,20,50,100,200],'n_e
```

In [159]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="accurac
```

In [160]:
```
grid_search.fit(x_train,y_train)
```

```
C:\Users\rubin\AppData\Local\Programs\Python\Python310\lib\site-packages\s
klearn\model_selection\_validation.py:686: DataConversionWarning: A column
-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rubin\AppData\Local\Programs\Python\Python310\lib\site-packages\s
klearn\model_selection\_validation.py:686: DataConversionWarning: A column
-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rubin\AppData\Local\Programs\Python\Python310\lib\site-packages\s
klearn\model_selection\_validation.py:686: DataConversionWarning: A column
-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rubin\AppData\Local\Programs\Python\Python310\lib\site-packages\s
klearn\model_selection\_validation.py:686: DataConversionWarning: A column
-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
```

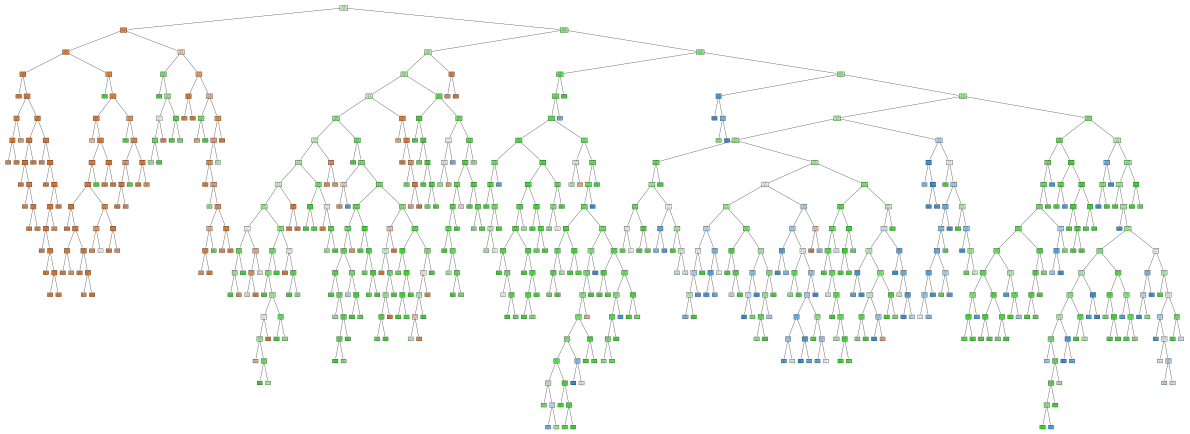
In [161]:
```
grid_search.best_score_
```

Out[161]:
```
0.8690826424177588
```

In [162]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=20, min_samples_leaf=5)
```

In [167]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,30))
plot_tree(rf_best.estimators_[5],class_names=['1','0','2','3','4'],filled=True
```



In [168]:
```
rf_best.feature_importances_
```

Out[168]:
```
array([1.])
```

In [166]:
```
score=rfc.score(x_test,y_test)
print(score)
```

```
0.9338533541341654
```

Conclusion: For the given data set we have performed different types of models ,and have got different accuracies .Based on the highest accuracy we can classify that which model best suits for the given dataset. Now we can conclude that Random Forest got the more accuracy among all models. Therefore,Random Forest model best fit for the dataframe.

In [ ]:
```

```