Mini Project on Flite

1.Problem Statement:which model is suitable best fit model for Flite

In [1]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing,svm
```

Data Collection

Read the Data

In [2]:

```
train_df=pd.read_csv(r"C:\Users\RAMADEVI SURIPAKA\OneDrive\Documents\train data.csv")
train_df
```

Out[2]:

	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 ? DEL	16:50	21:35	4h
...	...	...	...	...	...	...	...	
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU ? BLR	19:55	22:25	2h
10679	Air India	27/04/2019	Kolkata	Banglore	CCU ? BLR	20:45	23:20	2h
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR ? DEL	08:20	11:20	
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR ? DEL	11:30	14:10	2h
10682	Air India	9/05/2019	Delhi	Cochin	DEL ? GOI ? BOM ? COK	10:55	19:15	8h

10683 rows × 11 columns

In [3]:

```python
test_df=pd.read_csv(r"C:\Users\RAMADEVI SURIPAKA\Downloads\Test_set.csv")
test_df
```

Out[3]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Durat
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL ? BOM ? COK	17:30	04:25 07 Jun	10h 5
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? MAA ? BLR	06:20	10:20	
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	19:15	19:00 22 May	23h 4
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	08:00	21:00	
4	Air Asia	24/06/2019	Banglore	Delhi	BLR ? DEL	23:55	02:45 25 Jun	2h 5
...	...	...	...	...	...	...	...	
2666	Air India	6/06/2019	Kolkata	Banglore	CCU ? DEL ? BLR	20:30	20:25 07 Jun	23h 5
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU ? BLR	14:20	16:55	2h 3
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	21:50	04:25 07 Mar	6h 3
2669	Air India	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	04:00	19:15	15h 1
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL ? BOM ? COK	04:55	19:15	14h 2

2671 rows × 10 columns

Data Cleaning and preprocessing

In [4]:

```python
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 [5]:

```python
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 [6]:

```python
train_df.describe()
```

Out[6]:

	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 [7]:

```python
test_df.describe()
```

Out[7]:

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

In [8]:

```python
train_df.columns
```

Out[8]:

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

In [9]:

```python
test_df.columns
```

Out[9]:

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

In [10]:

```
train_df.head()
```

Out[10]:

	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

In [11]:

```
test_df.head()
```

Out[11]:

	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

In [12]:

```
train_df.shape
```

Out[12]:

(10683, 11)

In [13]:

```
test_df.shape
```

Out[13]:

(2671, 10)

To find duplicate values

In [14]:

```
train_df.duplicated().sum()
```

Out[14]:

220

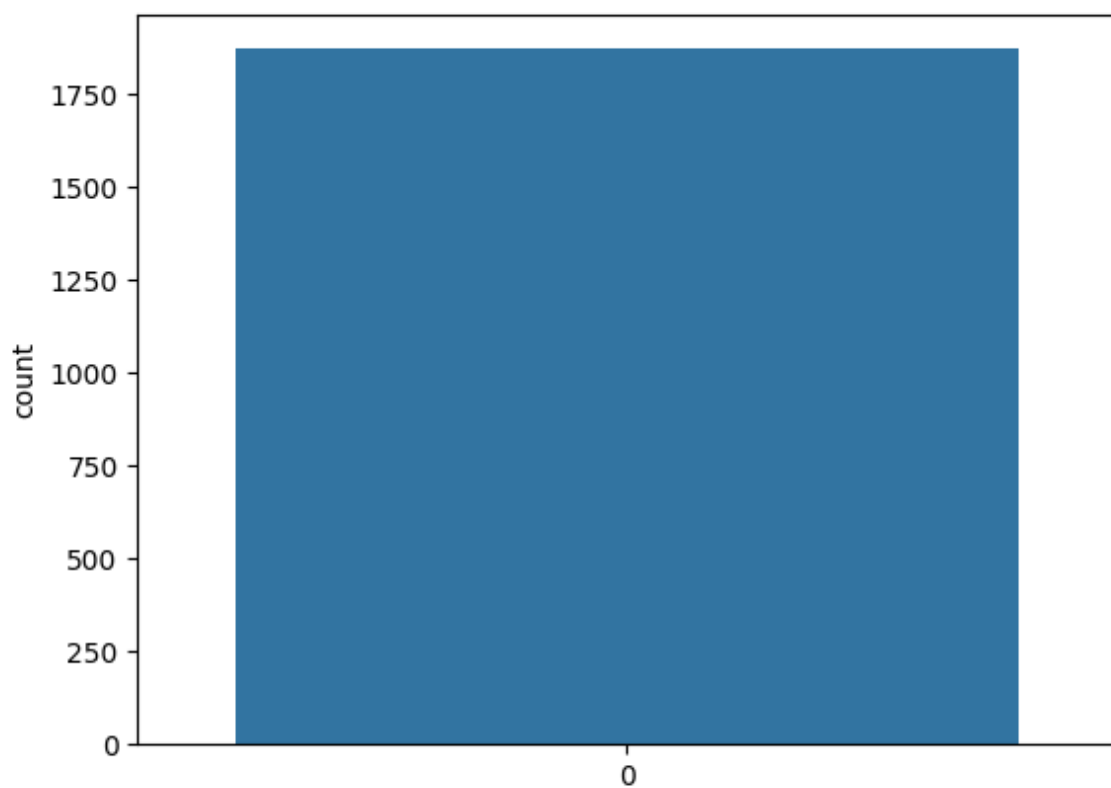In [15]:

```
test_df.duplicated().sum()
```

Out[15]:

26

3.Data Visualization:Visualize the unique counts

In [16]:

```python
sns.countplot(train_df['Price'].unique())
```
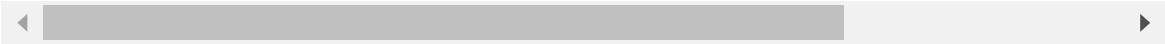
Out[16]:

```
<Axes: ylabel='count'>
```

In [17]:

```python
additionalinfo={"Additional_Info":{"No info":0,"In-flight meal not included":1,"No check
test_df=test_df.replace(additionalinfo)
test_df
```

Out[17]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Durat
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL ? BOM ? COK	17:30	04:25 07 Jun	10h 5
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? MAA ? BLR	06:20	10:20	
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	19:15	19:00 22 May	23h 4
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL ? BOM ? COK	08:00	21:00	
4	Air Asia	24/06/2019	Banglore	Delhi	BLR ? DEL	23:55	02:45 25 Jun	2h 5
...	...	...	...	...	...	...	...	
2666	Air India	6/06/2019	Kolkata	Banglore	CCU ? DEL ? BLR	20:30	20:25 07 Jun	23h 5
2667	IndiGo	27/03/2019	Kolkata	Banglore	CCU ? BLR	14:20	16:55	2h 3
2668	Jet Airways	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	21:50	04:25 07 Mar	6h 3
2669	Air India	6/03/2019	Delhi	Cochin	DEL ? BOM ? COK	04:00	19:15	15h 1
2670	Multiple carriers	15/06/2019	Delhi	Cochin	DEL ? BOM ? COK	04:55	19:15	14h 2

2671 rows × 10 columns

find null values

Drop the column

In [ ]:

```python
train_df.dropna(inplace=True)
```

In [19]:

```python
train_df.isnull().sum()
```

Out[19]:

```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              1
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        1
Additional_Info    0
Price              0
dtype: int64
```

In [20]:

```python
test_df.isnull().sum()
```

Out[20]:

```
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 [21]:

```python
airline={"Airline":{"Jet Airways":0,"IndiGo":1,"Air India":2,"Multiple carriers":3,
"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[21]:

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

10683 rows × 11 columns

In [22]:

```python
city={"Source":{"Delhi":0,"Kolkata":1,"Banglore":2,
"Mumbai":3,"Chennai":4}}
train_df=train_df.replace(city)
train_df
```

Out[22]:

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

10683 rows × 11 columns

In [23]:

```python
destination={"Destination":{"Cochin":0,"Banglore":1,"Delhi":2,
"New Delhi":3,"Hyderabad":4,"Kolkata":5}}
train_df=train_df.replace(destination)
train_df
```

Out[23]:

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

10683 rows × 11 columns

In [24]:

```python
stops={"Total_Stops":{"non-stop":0,"1 stop":1,"2 stops":2,"3 stops":3,"4 stops":4}}
train_df=train_df.replace(stops)
train_df
```

Out[24]:

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

10683 rows × 11 columns

In [25]:

```
train_df
```

Out[25]:

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

10683 rows × 11 columns

In [26]:

```python
train_df=train_df[['Airline','Source','Destination','Total_Stops','Price']]
sns.heatmap(train_df.corr(),annot=True)
```

Out[26]:

```
<Axes: >
```

In [29]:

```python
train_df500=train_df[:][:500]
sns.lmplot(x="Price",y="Source",data=train_df500,order=1,ci=None)
```

Out[29]:

```
<seaborn.axisgrid.FacetGrid at 0x154bab89d20>
```
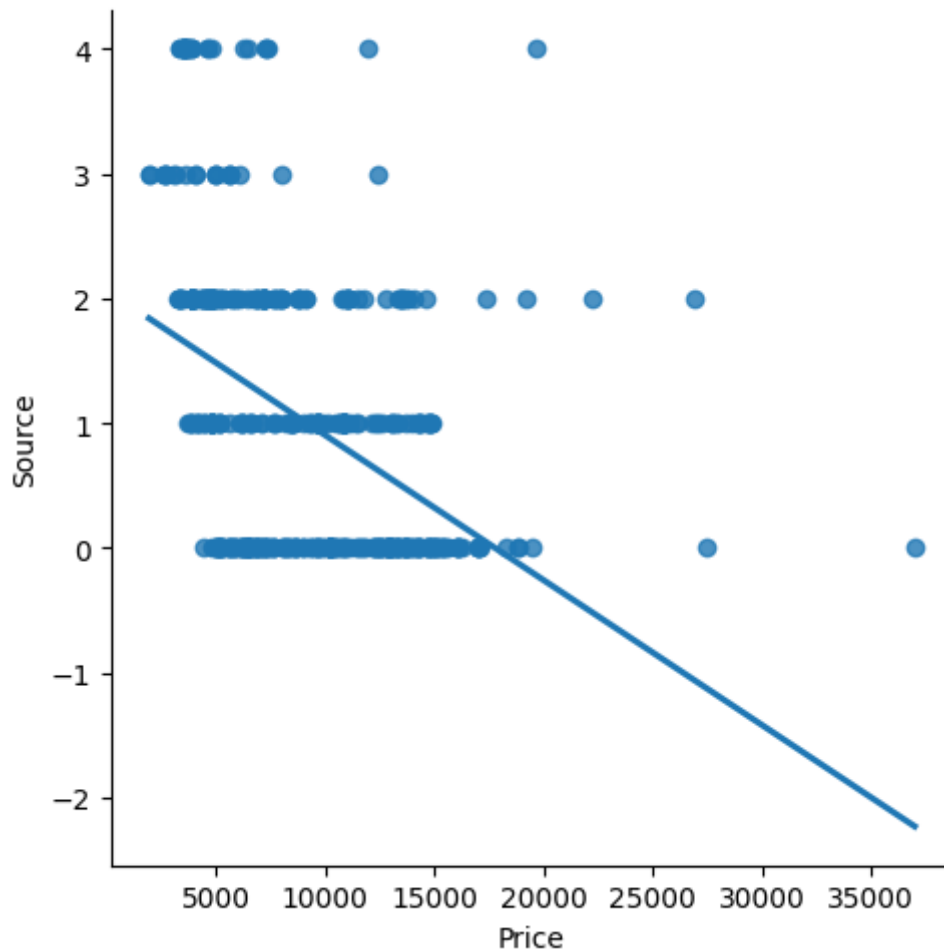
In [31]:

```
sns.pairplot(train_df)
```

Out[31]:

<seaborn.axisgrid.PairGrid at 0x154bdddc9d0>
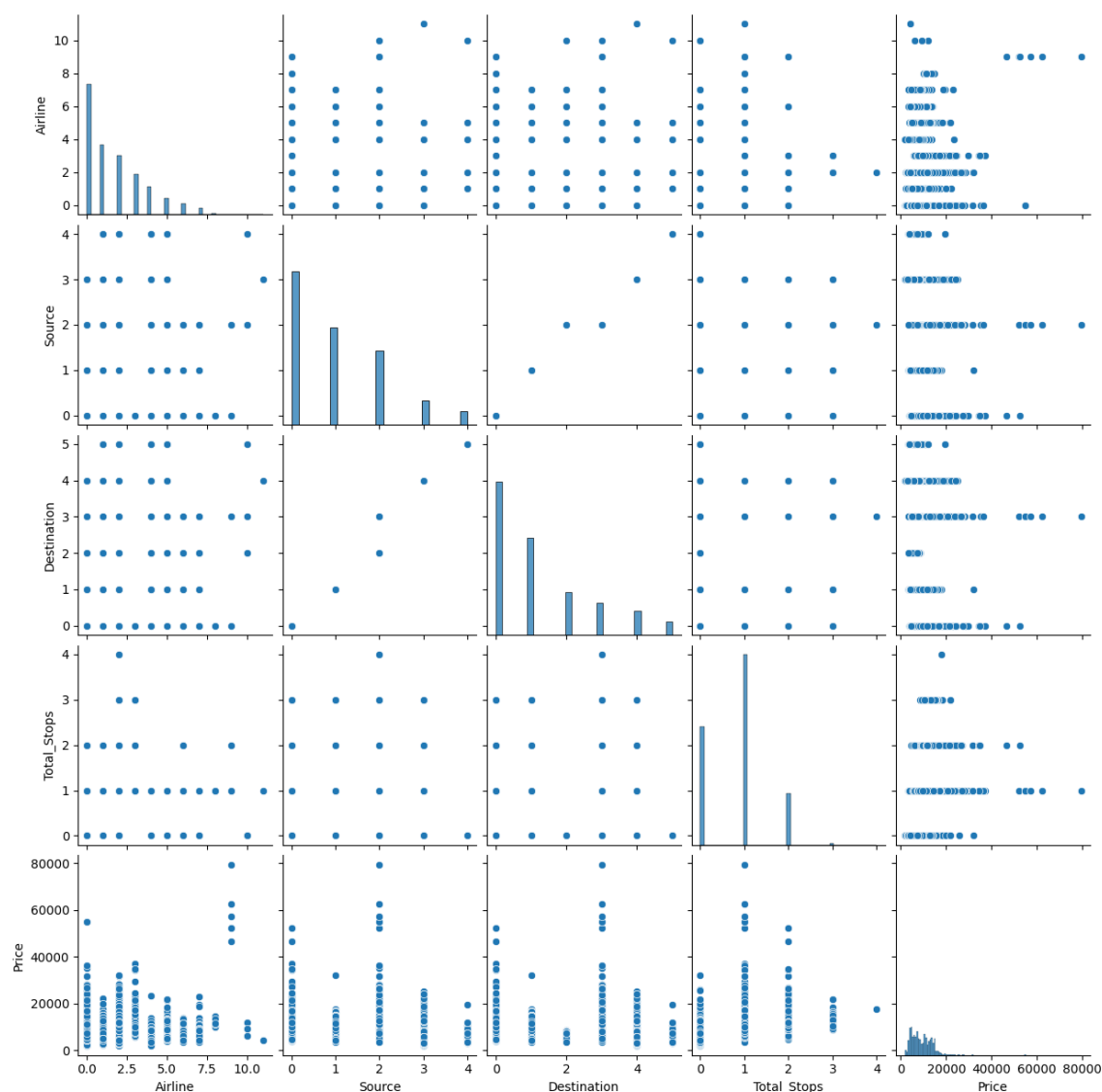
In [35]:

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

In [ ]:

4.Data Modeling:Using LinearRegression

In [55]:

```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=100)
```

In [80]:

```python
from sklearn.linear_model import LinearRegression
regr=LinearRegression()
regr.fit(x_train,y_train)
```

Out[80]:

LinearRegression()

**In a Jupyter environment, please rerun this cell to show the HTML representation or
trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page
with nbviewer.org.**

In [69]:

```python
score=regr.score(x_test,y_test)
print(score)
```

0.38389415387222314

In [70]:

```python
predictions=regr.predict(x_test)
```

In [71]:

```python
plt.scatter(y_test,predictions)
```

Out[71]:
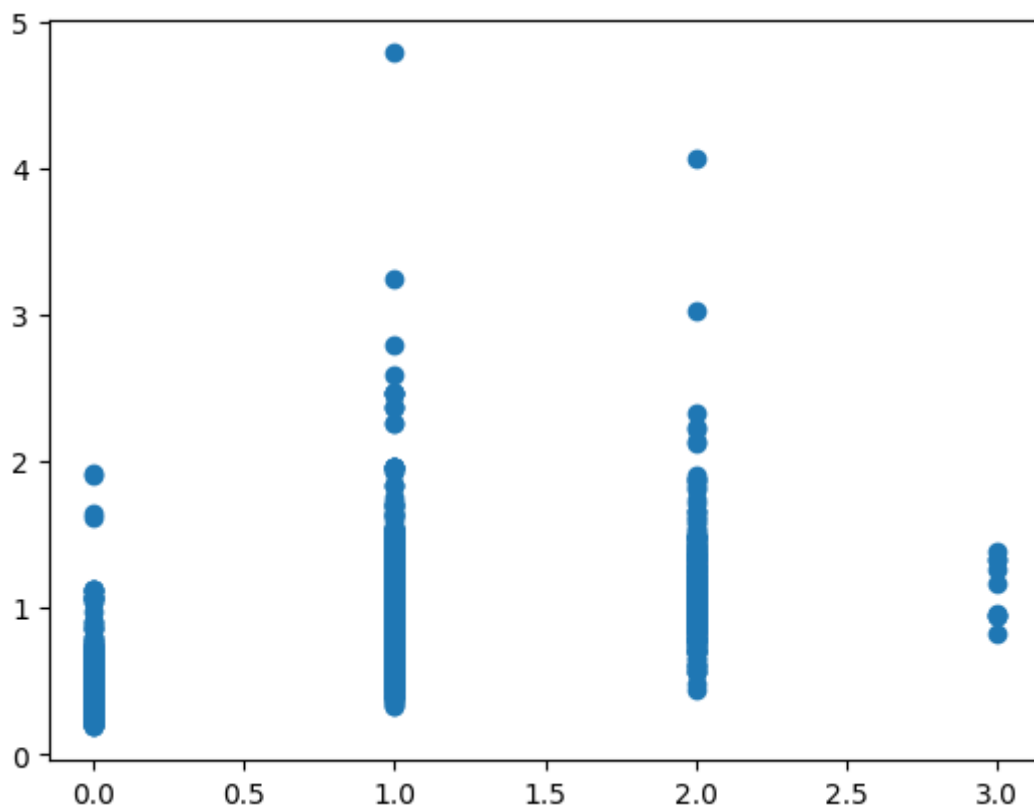
```
<matplotlib.collections.PathCollection at 0x154c347dc90>
```


In [76]:

```python
x=np.array(train_df['Price']).reshape(-1,1)
y=np.array(train_df['Total_Stops']).reshape(-1,1)
train_df.dropna(inplace=True)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_10164\143116723.p
y:3: SettingWithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  train_df.dropna(inplace=True)
```

In [73]:

```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
regr.fit(X_train,y_train)
regr.fit(X_train,y_train)
```

Out[73]:

```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [74]:

```
y_pred=regr.predict(X_test)
plt.scatter(X_test,y_test,color='y')
plt.plot(X_test,y_pred,color='b')
plt.show()
```
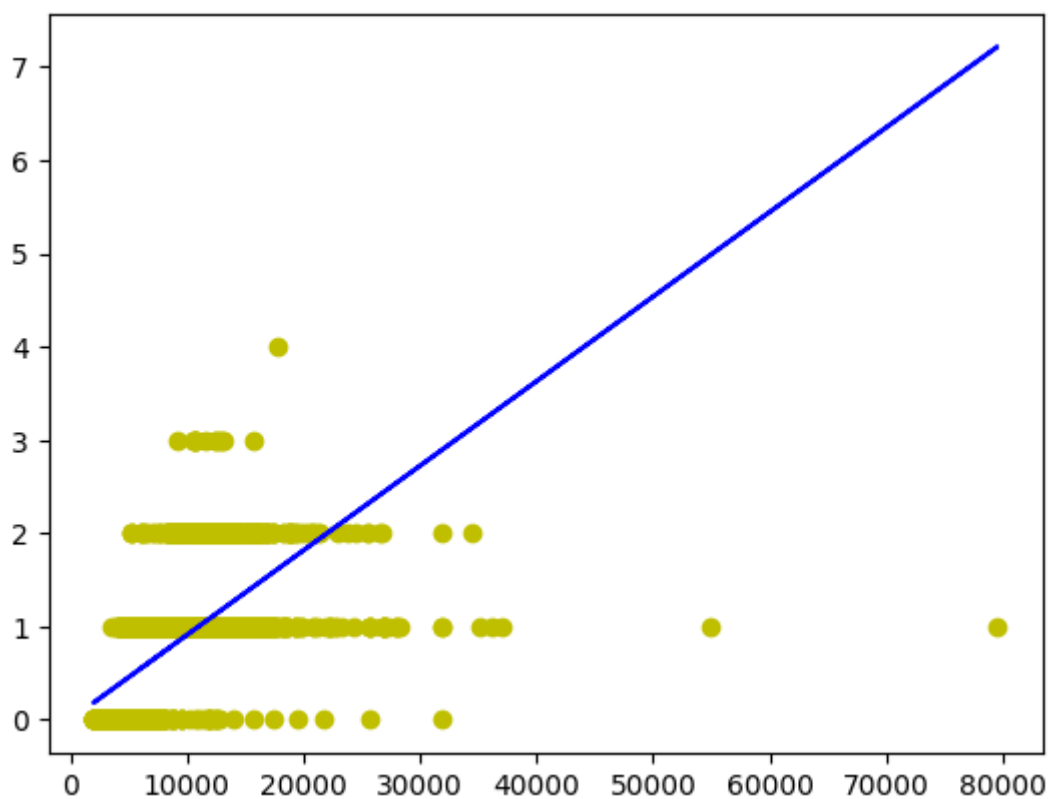

Logistic Regression

In [52]:

```python
#Logistic Regression
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=1)
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(max_iter=10000)
```

C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_10164\601506973.p
y:4: SettingWithCopyWarning:
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-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  train_df.dropna(inplace=True)

In [77]:

```python
lr.fit(x_train,y_train)
```

C:\Users\RAMADEVI SURIPAKA\AppData\Local\Programs\Python\Python310\lib\sit
e-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A colu
mn-vector y was passed when a 1d array was expected. Please change the sha
pe of y to (n_samples,), for example using ravel().
  y = column_or_1d(y, warn=True)

Out[77]:

LogisticRegression(max_iter=10000)
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [65]:

```python
score=lr.score(x_test,y_test)
print(score)
```
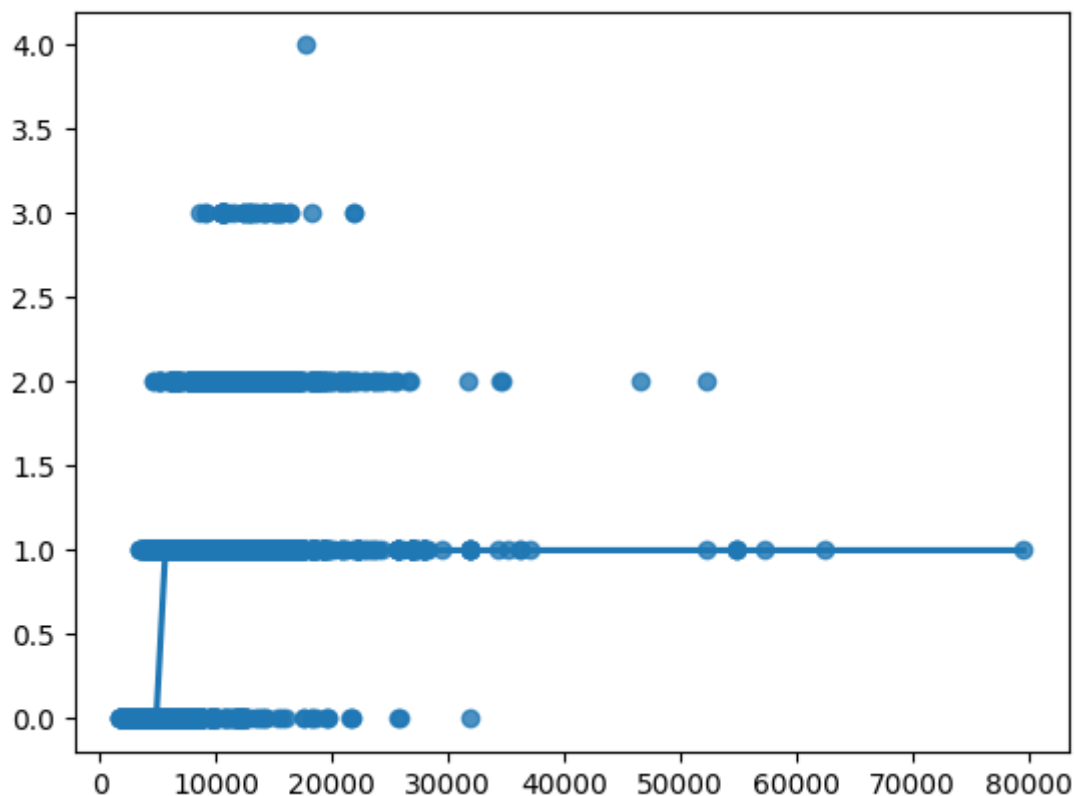
0.7160686427457098

In [67]:

```
sns.regplot(x=x,y=y,data=train_df,logistic=True,ci=None)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Programs\Python\Python310\lib\sit
e-packages\statsmodels\genmod\families\links.py:198: RuntimeWarning: overf
low encountered in exp
  t = np.exp(-z)
```

Out[67]:

```
<Axes: >
```


Decision Tree

In [81]:

```python
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier(random_state=0)
clf.fit(x_train,y_train)
```

Out[81]:

```
▼        DecisionTreeClassifier

DecisionTreeClassifier(random_state=0)
```

In [82]:

```
score=clf.score(x_test,y_test)
print(score)
```

0.4449297971918877

Random Forest

In [83]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(X_train,y_train)
```

```
C:\Users\RAMADEVI SURIPAKA\AppData\Local\Temp\ipykernel_10164\4104924521.p
y:3: DataConversionWarning: A column-vector y was passed when a 1d array w
as expected. Please change the shape of y to (n_samples,), for example usi
ng ravel().
  rfc.fit(X_train,y_train)
```

Out[83]:

```
▼ RandomForestClassifier
RandomForestClassifier()
```

In [84]:

```
params={'max_depth':[2,3,5,10,20],
'min_samples_leaf':[5,10,20,50,100,200],
'n_estimators':[10,25,30,50,100,200]}
```

In [85]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="accuracy")
```

In [86]:

```
grid_search.fit(X_train,y_train)
```

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

In [87]:

```
grid_search.best_score_
```

Out[87]:

```
0.8790962138842513
```

In [88]:

```
rf_best=grid_search.best_estimator_
rf_best
```
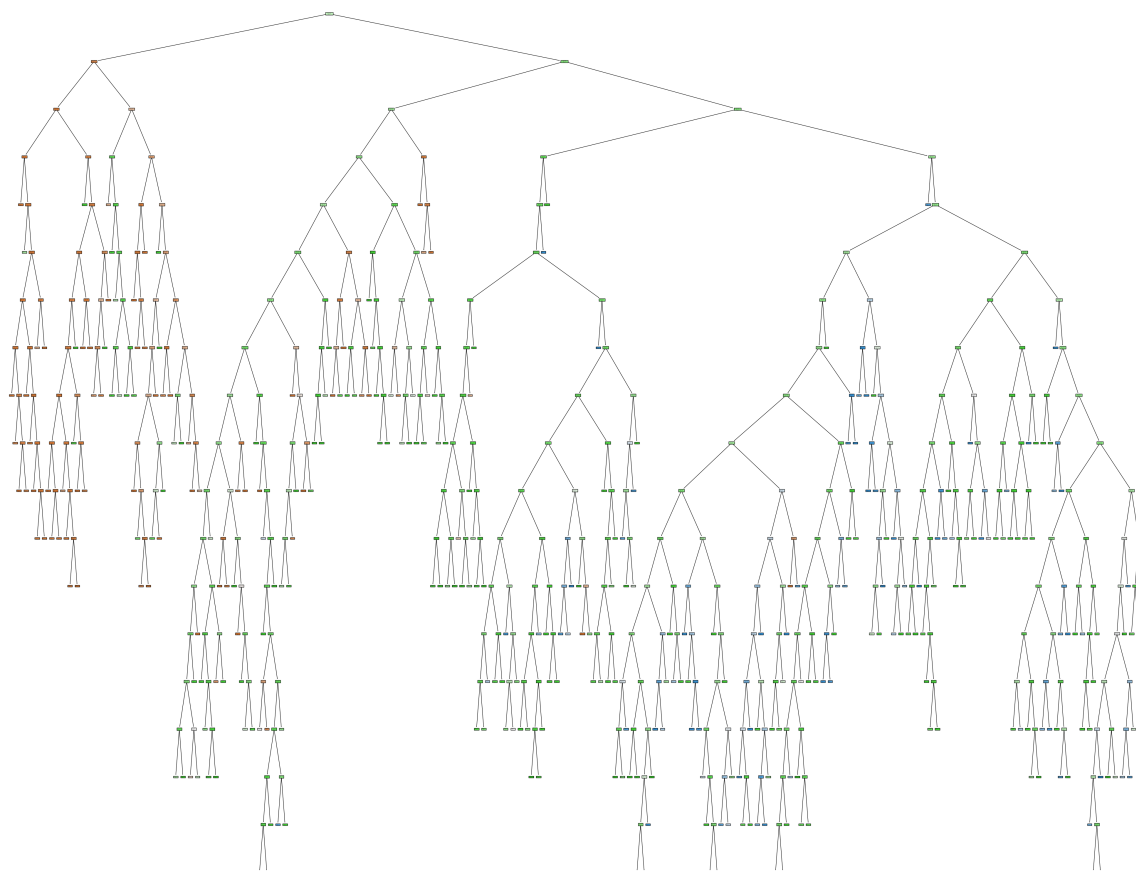
Out[88]:

```
                    RandomForestClassifier
RandomForestClassifier(max_depth=20, min_samples_leaf=5, n_estimators=25)
```

In [96]:

```python
from sklearn.tree import plot_tree
plt.figure(figsize=(50,40))
plot_tree(rf_best.estimators_[5],class_names=['0','1','2'],filled=True);
```


5.Prediction and Evaluation

In [93]:

```python
prediction=lr.predict(X_test)
```

In [94]:

```python
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: 0.3513581227540905

Conclusion

Finallyn In the project I have done Lenear,Logistic,Decision Tree,Random Forest models.I got the different types of accuracys, I concluded that random Forest model was the best fit model for the Flite Price I got the accuracy value 87.

In [ ]: