```
In [2]:   1  import pandas as pd
          2  import numpy as np
          3  import matplotlib.pyplot as plt
          4  import seaborn as sns
```

```
In [3]:   1  traindf=pd.read_csv(r"C:\Users\91628\Downloads\Data_Train1.csv")
          2  traindf
```

Out[3]:

	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 [5]:   1  testdf=pd.read_csv(r"C:\Users\91628\Downloads\Test_set1.csv")
          2  testdf
```

Out[5]:

	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 [6]:
```
1 testdf=pd.read_csv(r"C:\Users\91628\Downloads\Test_set1.csv")
2 testdf
```

Out[6]:

	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 [7]: `1  traindf.head()`

Out[7]:

	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 [8]: `1  testdf.head()`

Out[8]:

	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 [9]:   1 traindf.tail()

Out[9]:

	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 [10]:   1 testdf.tail()

Out[10]:

	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 [11]:   1 traindf.describe()

Out[11]:

	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 [12]:   1 testdf.describe()

Out[12]:

	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 [13]:   1 traindf.shape

Out[13]: (10683, 11)

In [14]:   1 testdf.shape

Out[14]: (2671, 10)

In [15]:   1   traindf.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 [16]:   1   traindf.duplicated().sum()

Out[16]: 220

In [17]:   1   testdf.duplicated().sum()

Out[17]: 26

In [18]:   1   traindf.columns

Out[18]: Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
                 'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
                 'Additional_Info', 'Price'],
               dtype='object')

In [19]:   1   traindf.columns

Out[19]: Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
                 'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
                 'Additional_Info', 'Price'],
               dtype='object')

In [20]:
```python
1 traindf.isnull().sum()
```

Out[20]:
```
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 [21]:
```python
1 testdf.isnull().sum()
```

Out[21]:
```
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 [22]:
```python
1 traindf.dropna(inplace=True)
```

In [23]:
```python
1 traindf.isnull().sum()
```

Out[23]:
```
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 [24]:
```python
1 traindf.shape
```

Out[24]: (10682, 11)

In [25]:
```python
1  traindf['Airline'].value_counts()
```

Out[25]:
```
Airline
Jet Airways                          3849
IndiGo                               2053
Air India                            1751
Multiple carriers                    1196
SpiceJet                              818
Vistara                              479
Air Asia                             319
GoAir                                194
Multiple carriers Premium economy    13
Jet Airways Business                  6
Vistara Premium economy               3
Trujet                                1
Name: count, dtype: int64
```

In [26]:
```python
1  traindf['Source'].value_counts()
2
```

Out[26]:
```
Source
Delhi      4536
Kolkata    2871
Banglore   2197
Mumbai      697
Chennai     381
Name: count, dtype: int64
```

In [27]:
```python
1  traindf['Destination'].value_counts()
```

Out[27]:
```
Destination
Cochin      4536
Banglore    2871
Delhi       1265
New Delhi    932
Hyderabad    697
Kolkata      381
Name: count, dtype: int64
```

In [28]:
```python
1  traindf['Total_Stops'].value_counts()
```

Out[28]:
```
Total_Stops
1 stop      5625
non-stop    3491
2 stops     1520
3 stops       45
4 stops        1
Name: count, dtype: int64
```

In [29]:
```python
airline={"Airline":{"Jet Airways":0,"IndiGo":1,"Air India":2,"Multiple ca
"SpiceJet":4,"Vistara":5,"Air Asia":6,"GoAir":7,
"Multiple carriers Premium economy":8,
"Jet Airways Business":9,"Vistara Premium economy":10,"Trujet":11}}
traindf=traindf.replace(airline)
traindf
```

Out[29]:

	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

10682 rows × 11 columns

In [30]:
```python
city={"Source":{"Delhi":0,"Kolkata":1,"Banglore":2,
"Mumbai":3,"Chennai":4}}
traindf=traindf.replace(city)
traindf
```

Out[30]:

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

10682 rows × 11 columns

In [31]:
```python
destination={"Destination":{"Cochin":0,"Banglore":1,"Delhi":2,
"New Delhi":3,"Hyderabad":4,"Kolkata":5}}
traindf=traindf.replace(destination)
traindf
```

Out[31]:

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

10682 rows × 11 columns

In [32]:
```python
1  stops={"Total_Stops":{"non-stop":0,"1 stop":1,"2 stops":2,
2  "3 stops":3,"4 stops":4}}
3  traindf=traindf.replace(stops)
4  traindf
```

Out[32]:

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

10682 rows × 11 columns

In [33]:   1   traindf

Out[33]:

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

10682 rows × 11 columns

In [34]:
```python
fdf=traindf[['Airline','Source','Destination','Total_Stops','Price']]
sns.heatmap(fdf.corr(),annot=True)
```

Out[34]: <Axes: >

In [35]:
```python
x=fdf[['Airline','Source','Destination','Total_Stops']]
y=fdf['Price']
```

Linear Regression

In [36]:
```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_s
```

```
In [37]:   1  from sklearn.linear_model import LinearRegression
           2  regr=LinearRegression()
           3  regr.fit(X_train,y_train)
           4  print(regr.intercept_)
           5  coeff_df=pd.DataFrame(regr.coef_,x.columns,columns=['coefficient'])
           6  coeff_df
```

7211.098088897471

Out[37]:

	coefficient
Airline	-418.483922
Source	-3275.073380
Destination	2505.480291
Total_Stops	3541.798053

```
In [38]:   1  score=regr.score(X_test,y_test)
           2  print(score)
```

0.41083048909283415

```
In [39]:   1  predictions=regr.predict(X_test)
```

```
In [40]:   1  plt.scatter(y_test,predictions)
```

Out[40]:   <matplotlib.collections.PathCollection at 0x2258f598650>
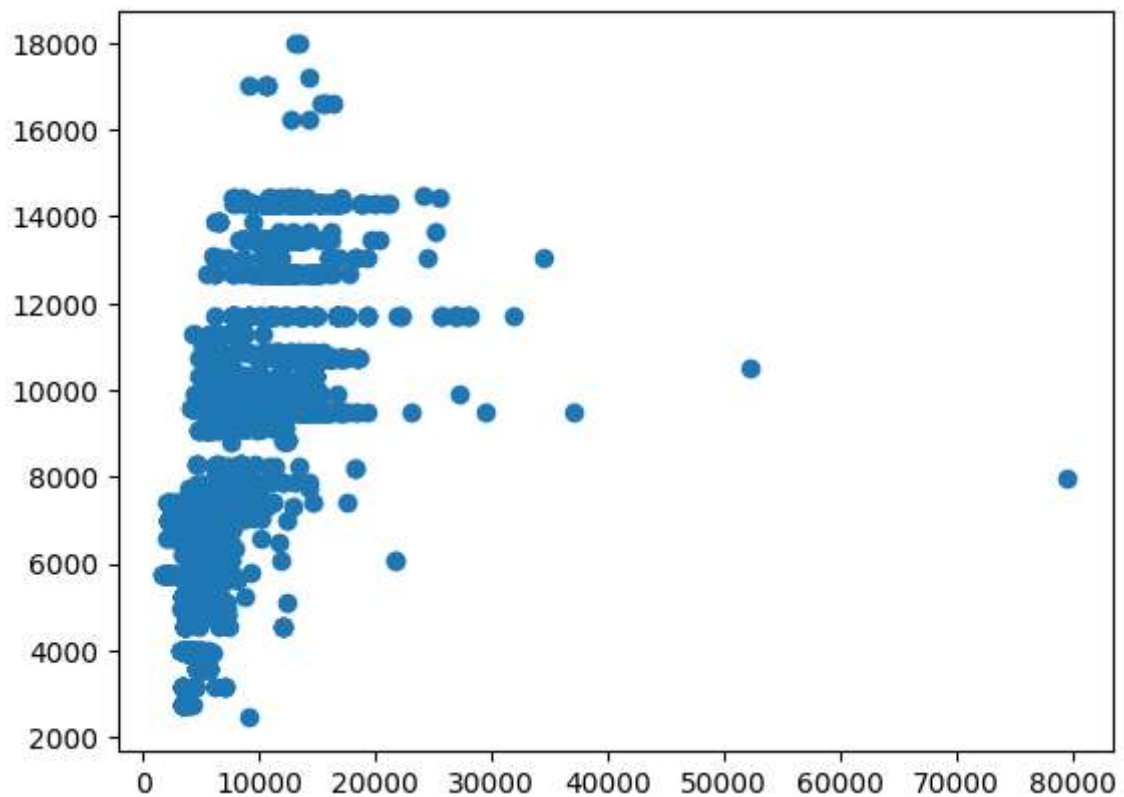
In [41]:
```python
1  x=np.array(fdf['Price']).reshape(-1,1)
2  y=np.array(fdf['Total_Stops']).reshape(-1,1)
3  fdf.dropna(inplace=True)
```

```
C:\Users\91628\AppData\Local\Temp\ipykernel_17176\521034954.py:3: SettingWith
CopyWarning:
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)
  fdf.dropna(inplace=True)
```

In [42]:
```python
1  X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
2  regr.fit(X_train,y_train)
3  regr.fit(X_train,y_train)
```

Out[42]: 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 [43]:
```python
1  y_pred=regr.predict(X_test)
2  plt.scatter(X_test,y_test,color='y')
3  plt.plot(X_test,y_pred,color='b')
4  plt.show()
```
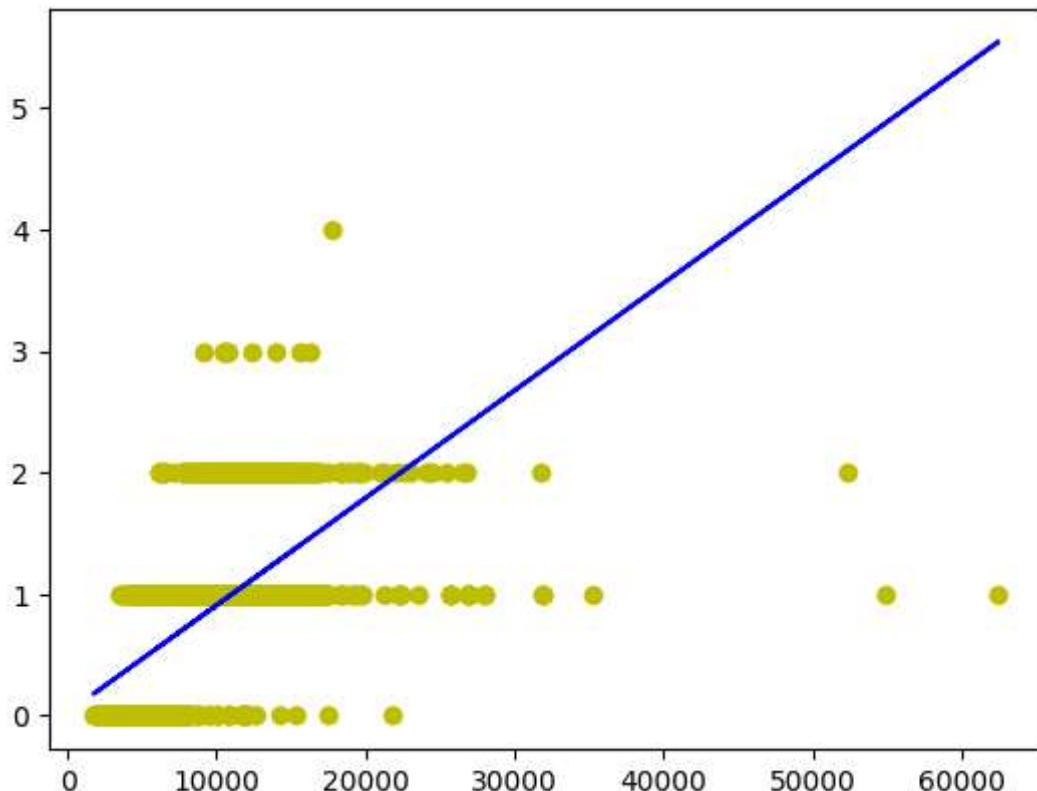
Logistic Regression

In [44]:
```python
1  x=np.array(fdf['Price']).reshape(-1,1)
2  y=np.array(fdf['Total_Stops']).reshape(-1,1)
3  fdf.dropna(inplace=True)
4  x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_s
5  from sklearn.linear_model import LogisticRegression
6  lr=LogisticRegression(max_iter=10000)
```

C:\Users\91628\AppData\Local\Temp\ipykernel_17176\497261869.py:3: SettingWith
CopyWarning:
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)
  fdf.dropna(inplace=True)

In [45]:
```python
1  lr.fit(x_train,y_train)
```

C:\Users\91628\AppData\Local\Programs\Python\Python311\Lib\site-packages\skle
arn\utils\validation.py:1143: DataConversionWarning: A column-vector y was pa
ssed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
  y = column_or_1d(y, warn=True)

Out[45]: 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 [46]:
```python
1  score=lr.score(x_test,y_test)
2  print(score)
```

0.7160686427457098

In [47]:
```python
1  sns.regplot(x=x,y=y,data=fdf,logistic=True,ci=None)
```

```
C:\Users\91628\AppData\Local\Programs\Python\Python311\Lib\site-packages\stat
smodels\genmod\families\links.py:198: RuntimeWarning: overflow encountered in
exp
  t = np.exp(-z)
```

Out[47]: <Axes: >
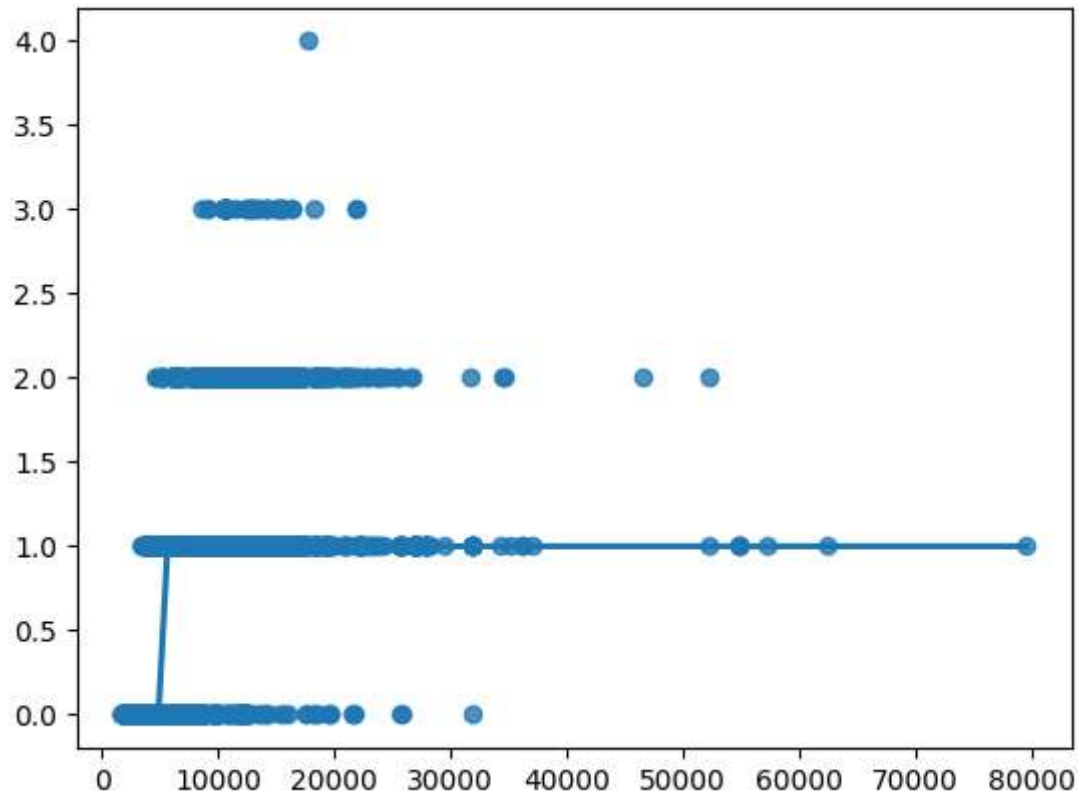
In [48]:
```python
1  from sklearn.tree import DecisionTreeClassifier
2  clf=DecisionTreeClassifier(random_state=0)
3  clf.fit(x_train,y_train)
```

Out[48]: DecisionTreeClassifier(random_state=0)

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 [49]:
```python
1  score=clf.score(x_test,y_test)
2  print(score)
```

```
0.9369734789391576
```

Random Classifier

In [50]:
```python
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(X_train,y_train)
```

C:\Users\91628\AppData\Local\Temp\ipykernel_17176\4104924521.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[50]: RandomForestClassifier()

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 [51]:
```python
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 [52]:
```python
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="ac
```

In [56]:
```python
grid_search.fit(X_train,y_train)
```

C:\Users\91628\AppData\Local\Programs\Python\Python311\Lib\site-packages\s
klearn\model_selection_split.py:700: UserWarning: The least populated cla
ss in y has only 1 members, which is less than n_splits=2.
  warnings.warn(
C:\Users\91628\AppData\Local\Programs\Python\Python311\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\91628\AppData\Local\Programs\Python\Python311\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\91628\AppData\Local\Programs\Python\Python311\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)

In [57]:
```python
grid_search.best_score_
```

Out[57]: 0.523605715699528

In [59]:
```python
1  rf_best=grid_search.best_estimator_
2  rf_best
```

Out[59]: RandomForestClassifier(max_depth=2, min_samples_leaf=10, n_estimators=10)

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 [60]:
```python
1  from sklearn.tree import plot_tree
2  plt.figure(figsize=(80,40))
3  plot_tree(rf_best.estimators_[4],class_names=['0','1','2','3','4'],filled=
```

Out[60]: [Text(0.5, 0.8333333333333334, 'x[0] <= 17053.0\ngini = 0.597\nsamples = 4663\nvalue = [2420, 3932, 1090, 35, 0]\nclass = 1'),
 Text(0.25, 0.5, 'x[0] <= 13941.5\ngini = 0.595\nsamples = 4521\nvalue = [234 7, 3847, 1037, 35, 0]\nclass = 1'),
 Text(0.125, 0.16666666666666666, 'gini = 0.594\nsamples = 4004\nvalue = [203 3, 3428, 931, 34, 0]\nclass = 1'),
 Text(0.375, 0.16666666666666666, 'gini = 0.596\nsamples = 517\nvalue = [314, 419, 106, 1, 0]\nclass = 1'),
 Text(0.75, 0.5, 'x[0] <= 17263.5\ngini = 0.655\nsamples = 142\nvalue = [73, 85, 53, 0, 0]\nclass = 1'),
 Text(0.625, 0.16666666666666666, 'gini = 0.638\nsamples = 23\nvalue = [15, 7, 13, 0, 0]\nclass = 0'),
 Text(0.875, 0.16666666666666666, 'gini = 0.643\nsamples = 119\nvalue = [58, 78, 40, 0, 0]\nclass = 1')]
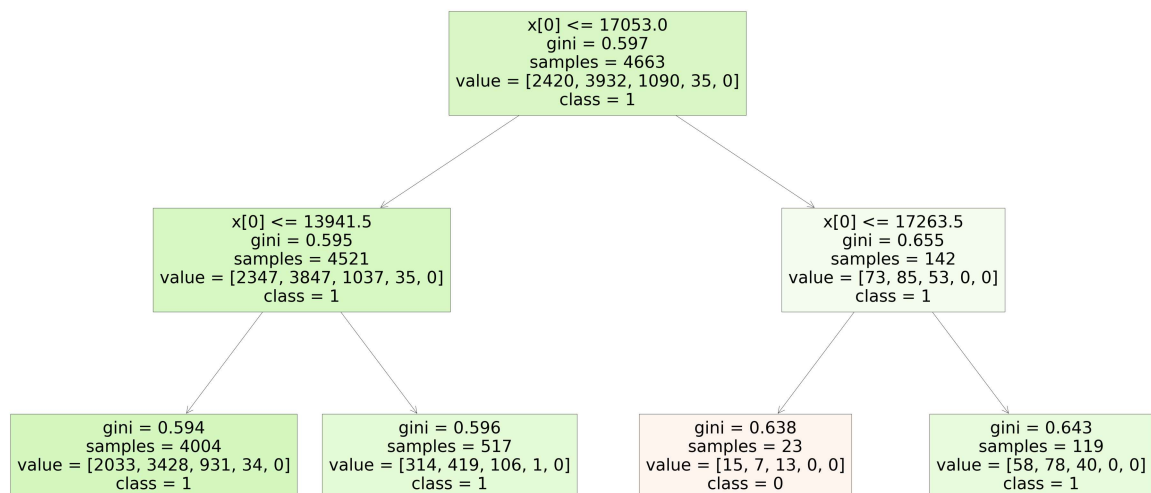
In [61]:
```python
1  score=rfc.score(x_test,y_test)
2  print(score)
```

0.4642745709828393

Conclusion

For the above Dataset we use different Types of Models,For that each and every model we get different Types of Accuracies.Based on that accuracies we can conclude which model is best fit for my our Dataset. Here we get different Types of accuracies For That Different Types of Accuracies Decision Tree is get more accuracy among all the models.So,that we can Conclude that for our Model Decision Tree is Best Fit.

In [ ]:   1