PROBLEM STATEMENT:- TO PREDICT THE INSURANCE CHARGES BASED ON VARIOUS FEATURES OF THE DATASET

IMPORTING THE ESSENTIAL LIBRARIES:-

In [2]:

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

LOADING TRAINING DATASET:-

In [3]:

```
train_df=pd.read_csv(r"C:\Users\hp\Documents\Data_Train.csv")
train_df
```

Out[3]:

	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 [4]:

```
test_df=pd.read_csv(r"C:\Users\hp\Documents\Test_set.csv")
train_df
```

Out[4]:

	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

DATAPREPROCESSING:-

In [5]:

```
train_df.head()
```

Out[5]:

	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 [6]:

```
train_df.tail()
```

Out[6]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Dura
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

In [7]:

```
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 [8]:

```
train_df.isnull().any()
```

Out[8]:

```
Airline            False
Date_of_Journey    False
Source             False
Destination        False
Route               True
Dep_Time           False
Arrival_Time       False
Duration           False
Total_Stops         True
Additional_Info    False
Price              False
dtype: bool
```

In [10]:

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

Out[10]:

```
220
```

In [12]:

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

Out[12]:

```
26
```

In [11]:

```
train_df.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 [13]:

```
test_df.describe()
```

Out[13]:

	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 [14]:

```
train_df.columns
```

Out[14]:

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

In [15]:

```
test_df.columns
```

Out[15]:

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

In [16]:

```
test_df.isnull().any()
```

Out[16]:

```
Airline            False
Date_of_Journey    False
Source             False
Destination        False
Route              False
Dep_Time           False
Arrival_Time       False
Duration           False
Total_Stops        False
Additional_Info    False
dtype: bool
```

In [17]:

```
train_df.shape
```

Out[17]:

```
(10683, 11)
```

In [18]:

```
test_df.shape
```

Out[18]:

```
(2671, 10)
```

In [20]:

```
train_df['Airline'].value_counts()
```

Out[20]:

```
Jet Airways                          3849
IndiGo                               2053
Air India                            1752
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: Airline, dtype: int64
```

In [22]:

```
train_df['Source'].value_counts()
```

Out[22]:

```
Delhi       4537
Kolkata     2871
Banglore    2197
Mumbai       697
Chennai      381
Name: Source, dtype: int64
```

In [24]:

```
train_df['Destination'].value_counts()
```

Out[24]:

```
Cochin       4537
Banglore     2871
Delhi        1265
New Delhi     932
Hyderabad     697
Kolkata       381
Name: Destination, dtype: int64
```

In [25]:

```
train_df['Total_Stops'].value_counts()
```

Out[25]:

```
1 stop      5625
non-stop    3491
2 stops     1520
3 stops       45
4 stops        1
Name: Total_Stops, dtype: int64
```

In [26]:

```
train_df['Price'].value_counts()
```

Out[26]:

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

In [27]:

```
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[27]:

	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 [29]:

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

Out[29]:

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

10683 rows × 11 columns

In [31]:

```
destination={"Destination":{"New Delhi":0,"Banglore":1,"Cochin":2,
"Banglore":3,"Hyderabad":4,"Kolkata":5,"Delhi":6}}
train_df=train_df.replace(destination)
train_df
```

Out[31]:

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

10683 rows × 11 columns

In [32]:

```
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[32]:

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

10683 rows × 11 columns

In [36]:

```
del train_df['Additional_Info']
```

In [37]:

```
train_df
```

Out[37]:

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

10683 rows × 10 columns

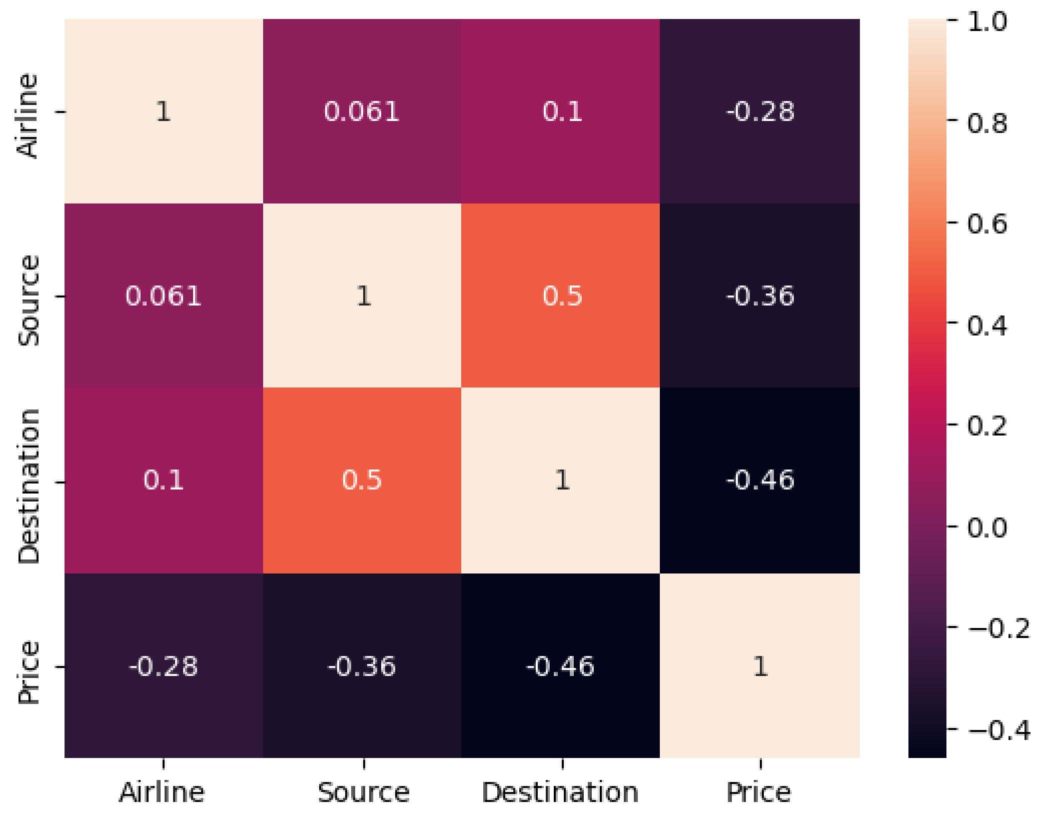
EXPLORATARY DATA ANALYSIS:-

In [49]:

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

Out[49]:

```
<Axes: >
```



In [51]:

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

LINEAR REGRESSION:-

In [52]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
```

In [53]:

```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train,y_train)
print(reg.intercept_)
coeff_=pd.DataFrame(reg.coef_,x.columns,columns=['coefficient'])
coeff_
```

```
13649.498863520994
```

Out[53]:

	coefficient
Airline	-555.125491
Source	-694.326269
Destination	-1040.440782

In [55]:

```
score=reg.score(X_test,y_test)
print(score)
```

```
0.3116327086259708
```

In [57]:

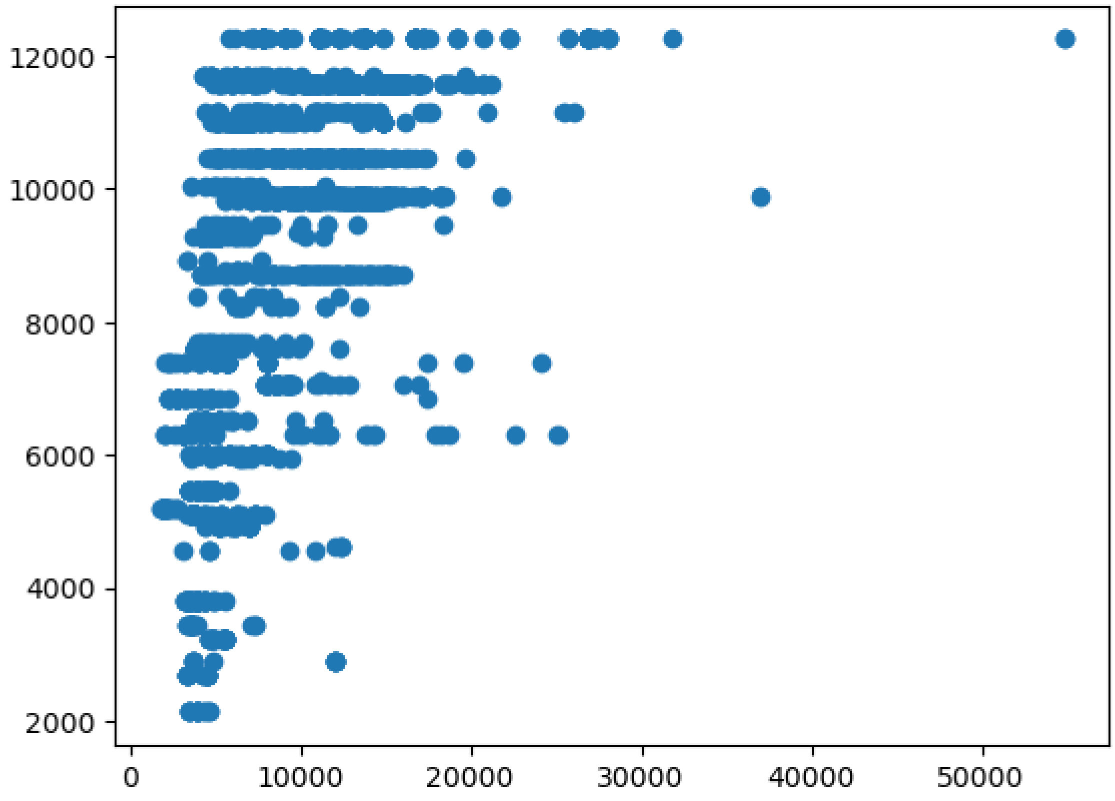
```
predictions=reg.predict(X_test)
```

In [58]:

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

Out[58]:

```
<matplotlib.collections.PathCollection at 0x2aa0db5ffd0>
```



In [60]:

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

```
C:\Users\hp\AppData\Local\Temp\ipykernel_2596\3357360649.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-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)
  tr_df.dropna(inplace=True)
```

In [62]:

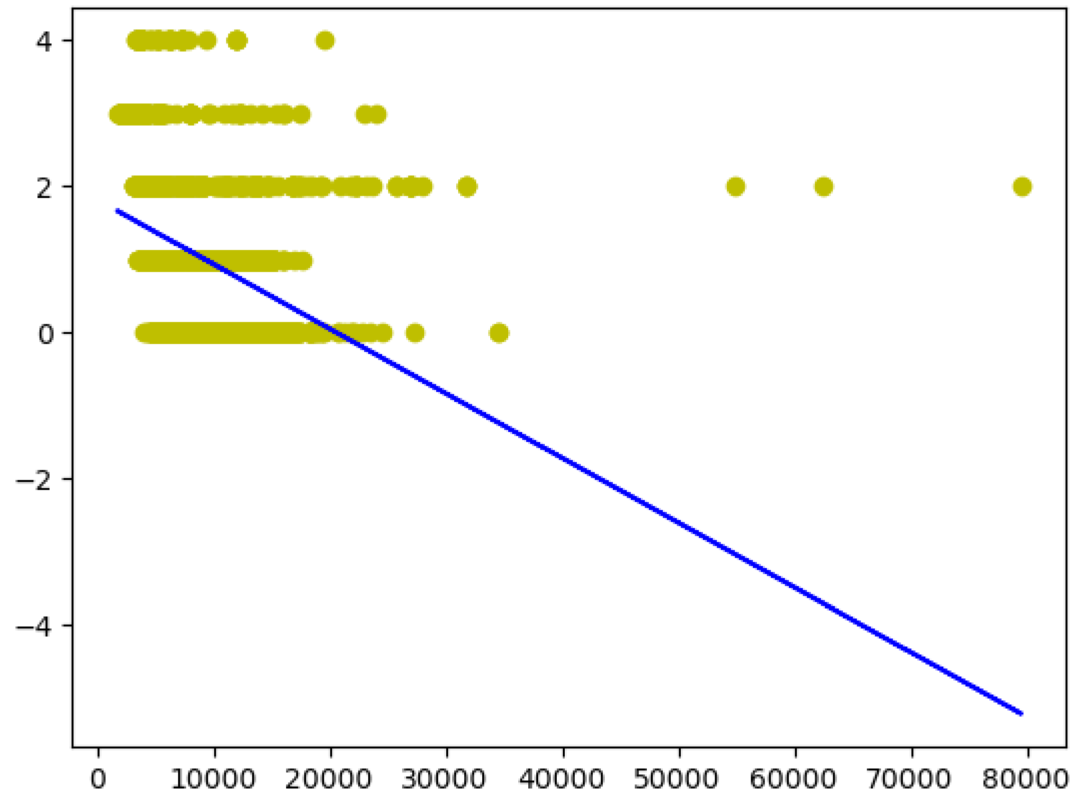
```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
reg.fit(X_train,y_train)
reg.fit(X_train,y_train)
```

Out[62]:

```
▾ LinearRegression
LinearRegression()
```

In [64]:

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



LOGISTIC REGRESSION:-

In [67]:

```
x=np.array(fdf['Price']).reshape(-1,1)
y=np.array(fdf['Source']).reshape(-1,1)
fdf.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\hp\AppData\Local\Temp\ipykernel_2596\2658472267.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-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)
  fdf.dropna(inplace=True)
```

In [68]:

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

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\utils\validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using r
avel().
  y = column_or_1d(y, warn=True)
```

Out[68]:

```
▾        LogisticRegression
LogisticRegression(max_iter=10000)
```

In [69]:

```
score=lr.score(X_test,y_test)
print(score)
```

```
0.431201248049922
```

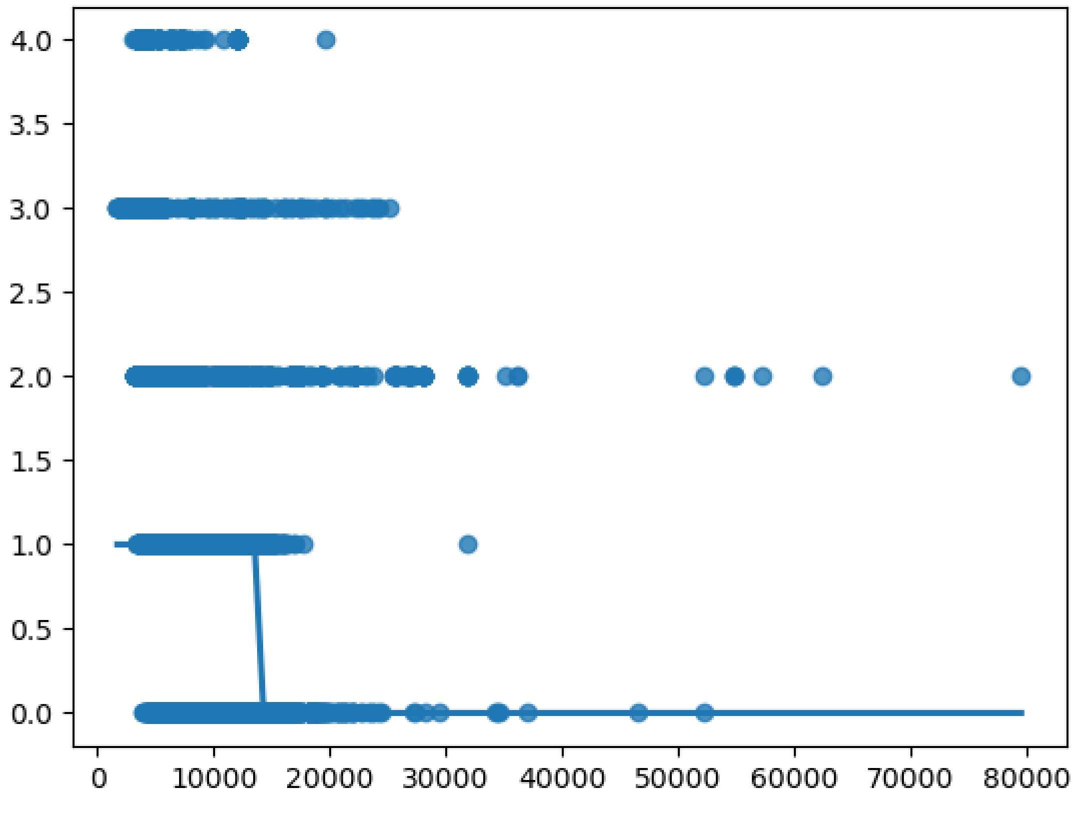
In [70]:

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

```
C:\Users\hp\anaconda3\lib\site-packages\statsmodels\genmod\families\links.
py:187: RuntimeWarning: overflow encountered in exp
  t = np.exp(-z)
```

Out[70]:

```
<Axes: >
```



DESICION TREE:-

In [71]:

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

Out[71]:

```
▾       DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [72]:

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

```
0.646801872074883
```

RANDOM FOREST:-

In [73]:

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

```
C:\Users\hp\AppData\Local\Temp\ipykernel_2596\4104924521.py:3: DataConvers
ionWarning: A column-vector y was passed when a 1d array was expected. Ple
ase change the shape of y to (n_samples,), for example using ravel().
  rfc.fit(X_train,y_train)
```

Out[73]:

```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [74]:

```
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 [75]:

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

In [76]:

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

```
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\hp\anaconda3\lib\site-packages\sklearn\model_selection\_valida
tion.py:686: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), fo
r example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\hp\anaconda3\lib\site-packages\sklearn\model_selection\_valida
tion.py:686: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), fo
r example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\hp\anaconda3\lib\site-packages\sklearn\model_selection\_valida
tion.py:686: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), fo
r example using ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\hp\anaconda3\lib\site-packages\sklearn\model_selection\_valida
tion.py:686: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), fo
r example using ravel().
```

In [77]:

```
grid_search.best_score_
```

Out[77]:

```
0.8280063109322571
```

In [78]:

```
rf_best=grid_search.best_estimator_
rf_best
```

Out[78]:

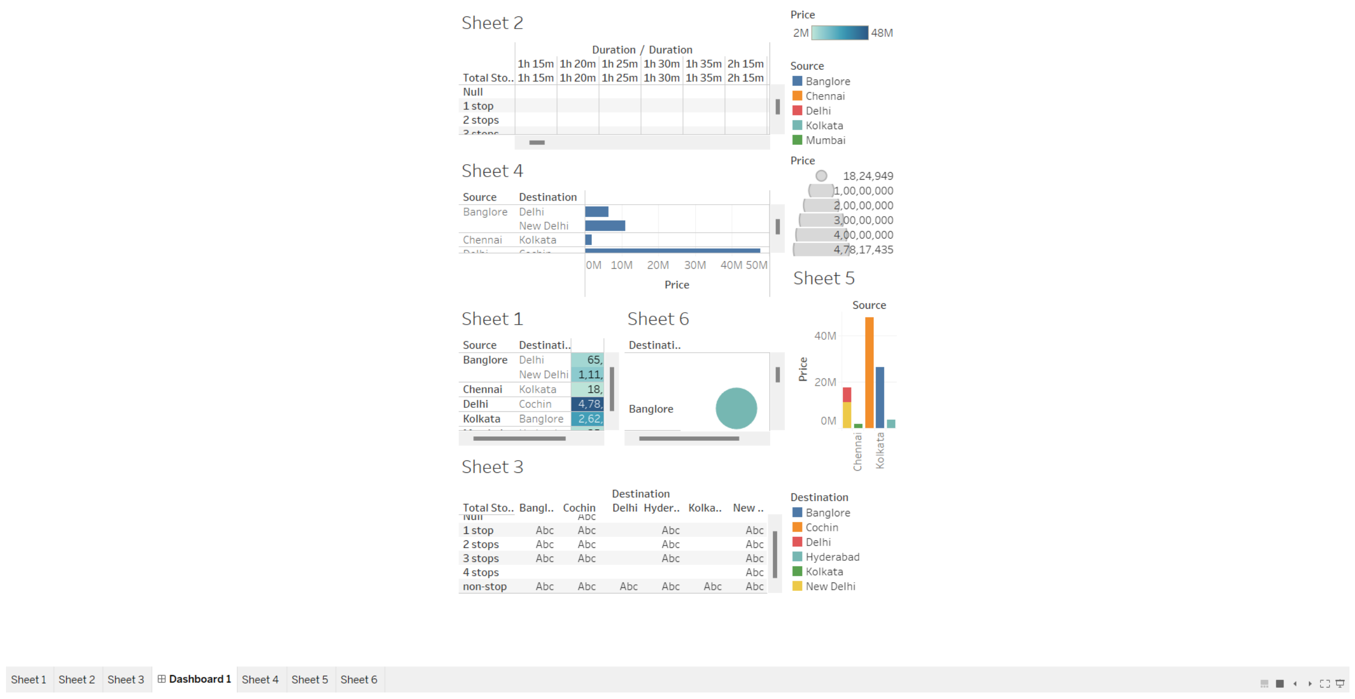
```
▾                          RandomForestClassifier
RandomForestClassifier(max_depth=20, min_samples_leaf=5, n_estimators=20
0)
```

CONCLUSION:-

```
THE SCORE OF LINEAR REGRESSION IS:- 0.3116327086259708
THE SCORE OF LOGISTIC REGRESSION IS:- 0.431201248049922
THE SCORE OF DECISION TREE IS:- 0.646801872074883
THE SCORE OF RANDOM FOREST IS:- 0.8280063109322571
```

AMONG ALL MODELS RANDOM FOREST YEILD HIGHEST ACCURACY.SO, WE PREFER RANDOM FOREST.

DASH BOARD USING TEABLEAU:-



Source	Destinati..	
Banglore	Delhi	65,
	New Delhi	1,11,
Chennai	Kolkata	18,
Delhi	Cochin	4,78,
Kolkata	Banglore	2,62,

Total Sto..	Bangl..	Cochin	Delhi	Hyder..	Kolka..	New ..
Null		Abc				
1 stop	Abc	Abc		Abc		Abc
2 stops	Abc	Abc		Abc		Abc
3 stops	Abc	Abc		Abc		Abc
4 stops						Abc
non-stop	Abc	Abc	Abc	Abc	Abc	Abc