Problem Statement:Which model is suitable for Dataset ¶

Importing All The Required Libraries

In [69]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [70]:

```
train_df=pd.read_csv(r"C:\Users\mural\Downloads\Copy of Data_Train.csv")
train_df
```

Out[70]:

	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 [71]:

```
test_df=pd.read_csv(r"C:\Users\mural\Downloads\Copy of Test_set.csv")
test_df
```

Out[71]:

	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 Preprocessing And Cleaning

In [72]:

```
train_df.head()
```

Out[72]:

	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 [73]:

```
test_df.head()
```

Out[73]:

	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 [74]:

```
train_df.tail()
```

Out[74]:

	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 [75]:

```
test_df.tail()
```

Out[75]:

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

In [76]:

```
train_df.describe()
```

Out[76]:

	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 [77]:

```
test_df.describe()
```

Out[77]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Dura
count	2671	2671	2671	2671	2671	2671	2671	
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 [78]:

```
#checking for null values
train_df.isnull().sum()
```

Out[78]:

```
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 [79]:

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

Out[79]:

```
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 [81]:

```
#Removing null values
train_df.dropna(inplace=True)
```

In [82]:

```
#checking for duplicate values
train_df.duplicated().sum()
```

Out[82]:

```
220
```

In [83]:

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

Out[83]:

```
26
```

In [84]:

```
#Removing duplicate values
train_df=train_df.drop_duplicates()
```

In [85]:

```
test_df=test_df.drop_duplicates()
```

In [86]:

```
train_df.shape
```

Out[86]:

```
(10462, 11)
```

In [87]:

```
test_df.shape
```

Out[87]:

```
(2645, 10)
```

Conversion of datatype of values from String to Numerical Values

In [90]:

```
airline={"Airline":{"Jet Airways":0,"IndiGo":1,"Air India":2,"Multiple carriers":3,"Spic
                    "Air Asia":6,"GoAir":7,"Multiple carriers Premium economy":8,"Jet Ai
                    "Vistara Premium economy":10,"Trujet":11}}
train_df=train_df.replace(airline)
train_df
```

Out[90]:

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

10462 rows × 11 columns

In [91]:

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

Out[91]:

	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(

10462 rows × 11 columns

In [94]:

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

Out[94]:

	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 5(
1	2	1/05/2019	1	1	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 2(
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 2(
4	1	01/03/2019	2	3	BLR ? NAG ? DEL	16:50	21:35	4h 4(
...	...	...	...	...	...	...	...	
10678	6	9/04/2019	1	1	CCU ? BLR	19:55	22:25	2h 3(
10679	2	27/04/2019	1	1	CCU ? BLR	20:45	23:20	2h 3(
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 4(
10682	2	9/05/2019	0	0	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 2(

10462 rows × 11 columns

In [95]:

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

Out[95]:

	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 5(
1	2	1/05/2019	1	1	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 2(
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 2(
4	1	01/03/2019	2	3	BLR ? NAG ? DEL	16:50	21:35	4h 4(
...	...	...	...	...	...	...	...	
10678	6	9/04/2019	1	1	CCU ? BLR	19:55	22:25	2h 3(
10679	2	27/04/2019	1	1	CCU ? BLR	20:45	23:20	2h 3(
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 4(
10682	2	9/05/2019	0	0	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 2(

10462 rows × 11 columns

In [96]:

```
train_df
```

Out[96]:

	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 5(
1	2	1/05/2019	1	1	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 2[
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 2[
4	1	01/03/2019	2	3	BLR ? NAG ? DEL	16:50	21:35	4h 4[
...	...	...	...	...	...	...	...	
10678	6	9/04/2019	1	1	CCU ? BLR	19:55	22:25	2h 3(
10679	2	27/04/2019	1	1	CCU ? BLR	20:45	23:20	2h 3[
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 4(
10682	2	9/05/2019	0	0	DEL ? GOI ? BOM ? COK	10:55	19:15	8h 2(

10462 rows × 11 columns

Feature Scaling :To Split the data into training data and test data

In [97]:

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

In [109]:

```
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)
```

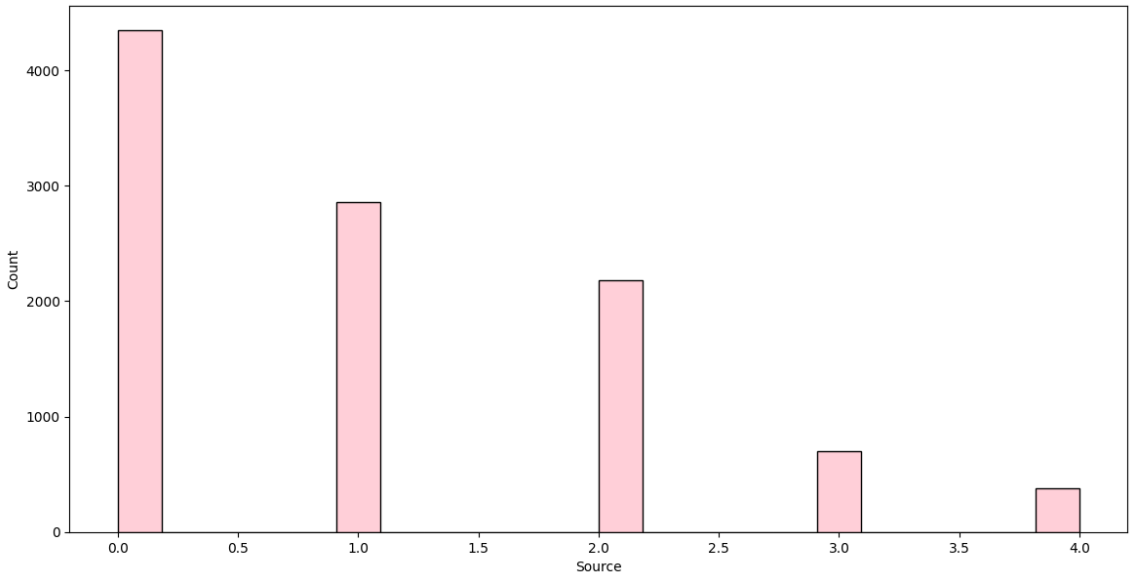
Data Visualization

In [130]:

```
plt.figure(figsize=(14,7))
sns.histplot(data=train_df, x='Source',color='pink')
```

Out[130]:

```
<Axes: xlabel='Source', ylabel='Count'>
```

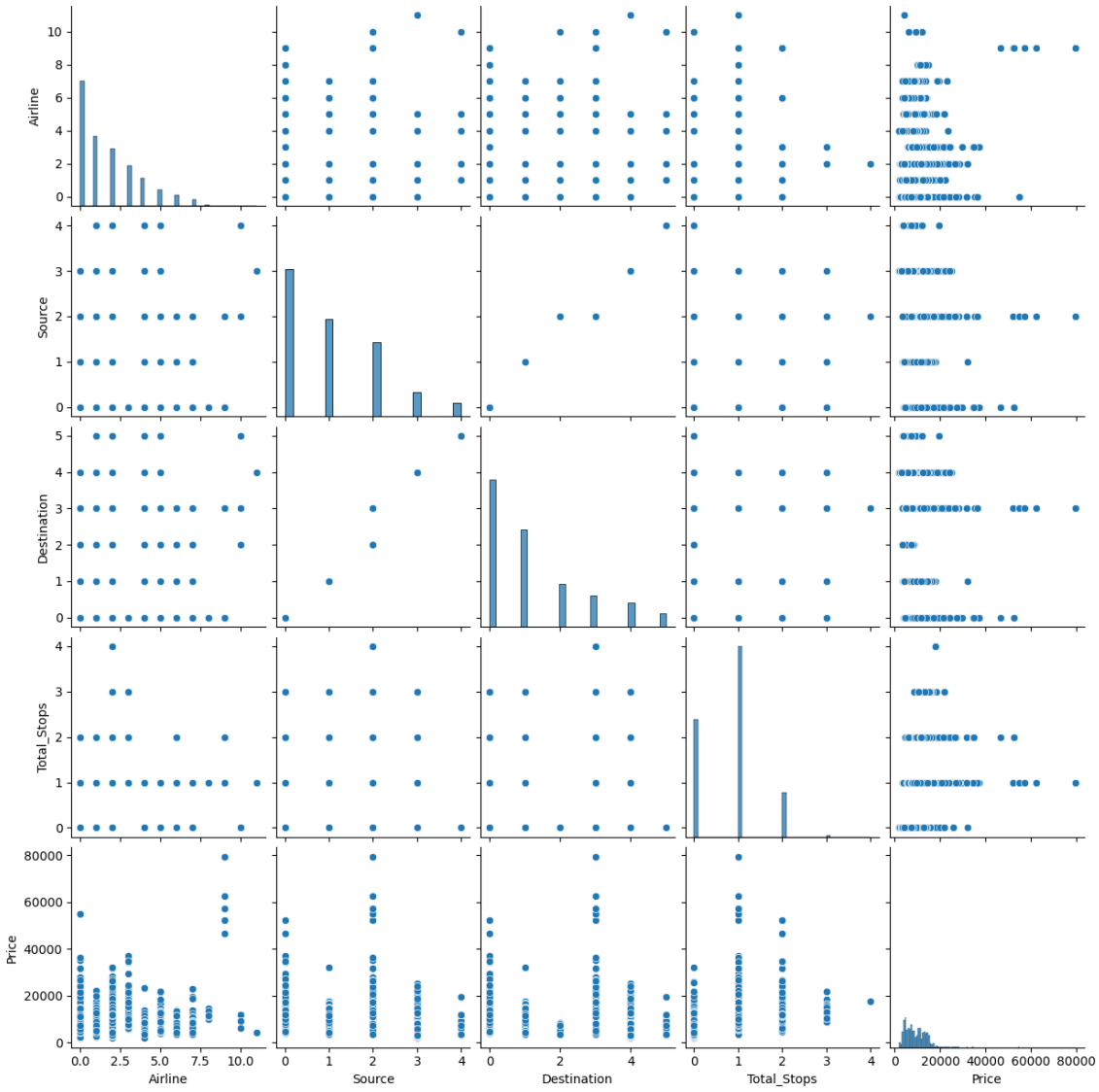


In [111]:

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

Out[111]:

```
<seaborn.axisgrid.PairGrid at 0x292a68c5890>
```



In [112]:

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

Out[112]:

<Axes: >

LinearRegression

In [113]:

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

Out[113]:

```
LinearRegression
LinearRegression()
```

In [114]:

```
y_pred = regressor.predict(x_test)
```

In [115]:

```
from sklearn.metrics import r2_score
score = r2_score(y_test, y_pred)
```

In [116]:

```
score
```

Out[116]:

```
0.3866735537606394
```

Since we did not get the good accuracy for LinearRegression we are going to implement LogisticRegression

LogisticRegression

In [117]:

```
x=np.array(df['Price']).reshape(-1,1)
y=np.array(df['Total_Stops']).reshape(-1,1)
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)
import warnings
warnings.simplefilter(action='ignore')
```

In [118]:

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

Out[118]:

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

In [119]:

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

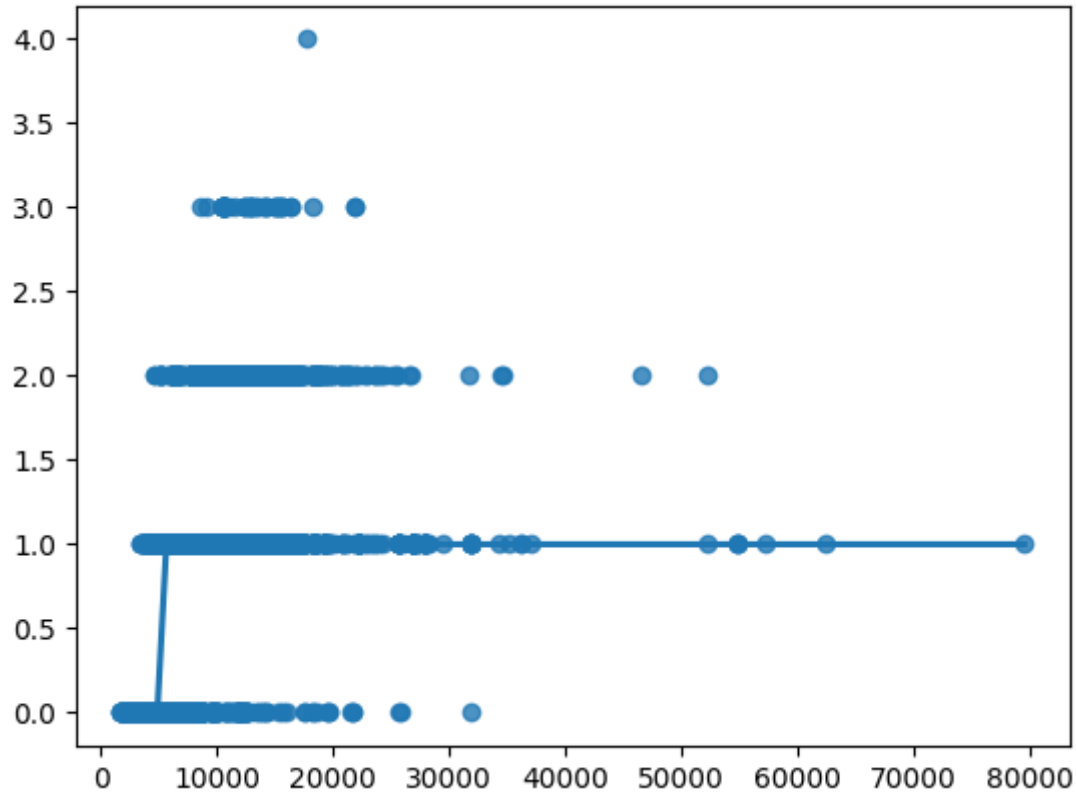
```
0.7311245619624084
```

In [120]:

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

Out[120]:

```
<Axes: >
```



Since we did not get the accuracy for Logistic Regression we are going to implement Decision Tree andRandom Forest and make a comparative study for finding the best model for the dataset

Decision tree

In [121]:

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

Out[121]:

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

In [122]:

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

```
0.9327811404906021
```

Random Forest

In [123]:

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

Out[123]:

RandomForestClassifier

RandomForestClassifier()

In [124]:

```
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 [125]:

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

Out[125]:

GridSearchCV

estimator: RandomForestClassifier

RandomForestClassifier

In [126]:

```
grid_search.best_score_
```

Out[126]:

```
0.5381674464080405
```

In [127]:

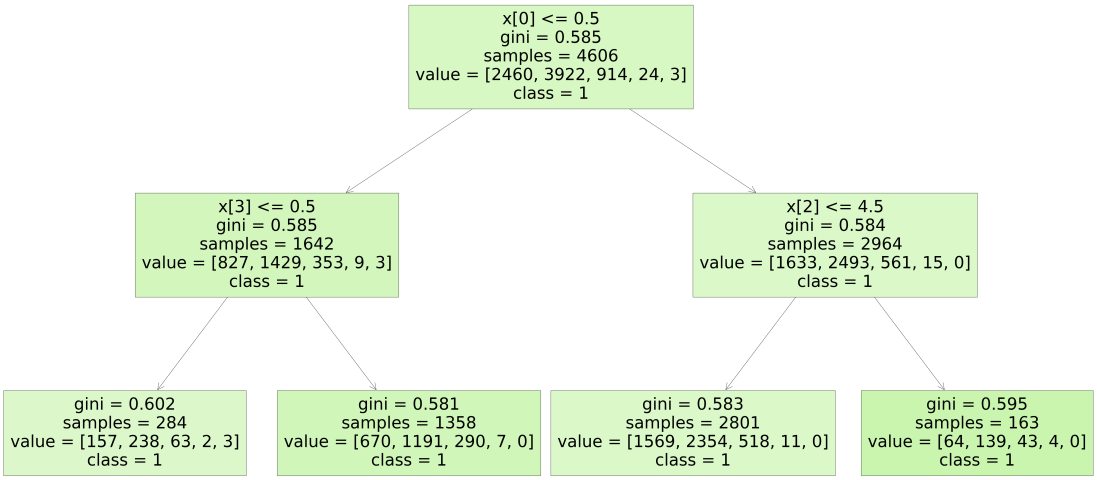
```
rf_best=grid_search.best_estimator_
rf_best
```

Out[127]:

```
RandomForestClassifier(max_depth=2, min_samples_leaf=5, n_estimators=10)
```

In [128]:

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



In [129]:

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

```
0.5297865562280981
```

CONCLUSION : Based on accuracy scores of all models that were implemented we can conclude that"Decision Tree" is the best model for the givendataset

In [ ]:

```

```