FLIGHT PRICE PREDICTION

LINEAR REGRESSION

PROBLEM STATEMENT: TO PREDICT AND ANALYZE THE SCORE FOR EACH LINEAR, RIDGE AND LASSO REGRESSION.

In [1]:
```
import pandas as pd
import numpy as np
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
```

In [4]:
```
df=pd.read_excel(r"C:\Users\91756\Documents\Data_Train.xlsx")
df
```

Out[4]:

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

10683 rows × 11 columns

In [5]:
```
convert={"Total_Stops":{"non-stop":0,"1 stop":1,"2 stops":2,"3 stops":3,"4 sto
df=df.replace(convert)
df
```

Out[5]:

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

10683 rows × 11 columns

Data Cleaning

In [6]:
```
df=df[['Total_Stops','Price']]
df.columns=['ts','pr']
```

In [7]:
```
df.head()
```

Out[7]:

	ts	pr
0	0.0	3897
1	2.0	7662
2	2.0	13882
3	1.0	6218
4	1.0	13302

In [8]:
```
df.describe()
```

Out[8]:

	ts	pr
count	10682.000000	10683.000000
mean	0.824190	9087.064121
std	0.675229	4611.359167
min	0.000000	1759.000000
25%	0.000000	5277.000000
50%	1.000000	8372.000000
75%	1.000000	12373.000000
max	4.000000	79512.000000

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   ts      10682 non-null  float64
 1   pr      10683 non-null  int64  
dtypes: float64(1), int64(1)
memory usage: 167.0 KB
```

In [10]:
```
features=['Total_Stops']
```

In [11]:
```
target=df.columns[-1]
```

In [12]:
```
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\91756\AppData\Local\Temp\ipykernel_1376\4116506308.py:1: 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/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  df.fillna(method='ffill',inplace=True)
```

In [13]:
```
X = np.array(df['ts']).reshape(-1,1)
y = np.array(df['pr']).reshape(-1,1)
```

In [14]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,train_size=0.7)
```

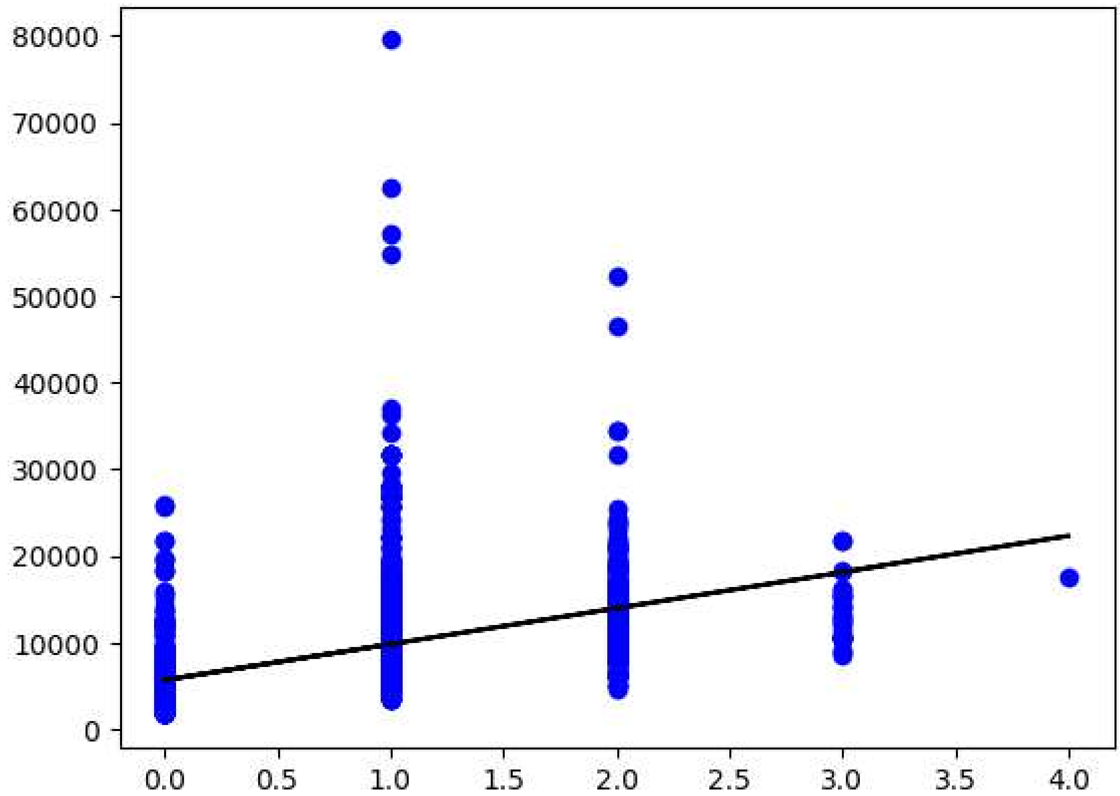
In [15]:
```
lm=LinearRegression()
lm.fit(X_train,y_train)
```

Out[15]:
```
▾ LinearRegression
LinearRegression()
```

In [16]:
```
lm.score(X_train,y_train)
```

Out[16]: 0.3641871145264679

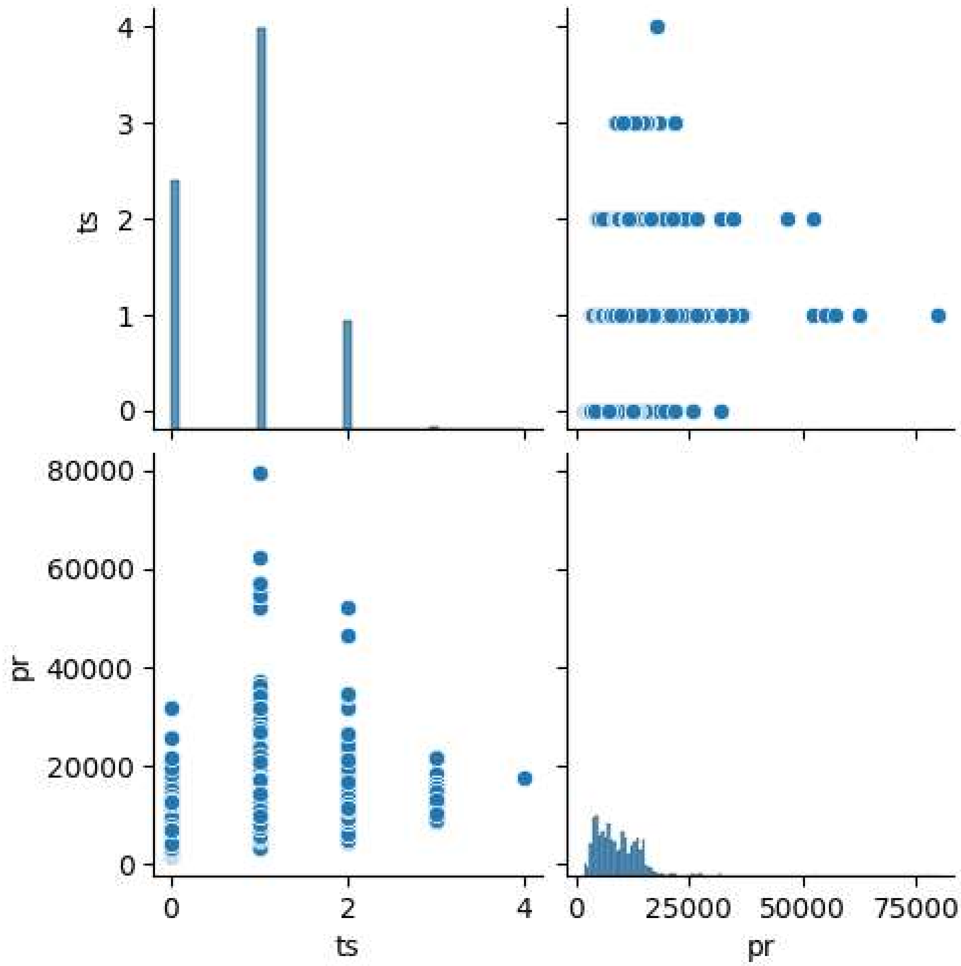
In [17]:
```
y_pred=lm.predict(X_train)
plt.scatter(X_train,y_train,color='b')
plt.plot(X_train,y_pred,color='k')
plt.show()
```



EDA REPORT

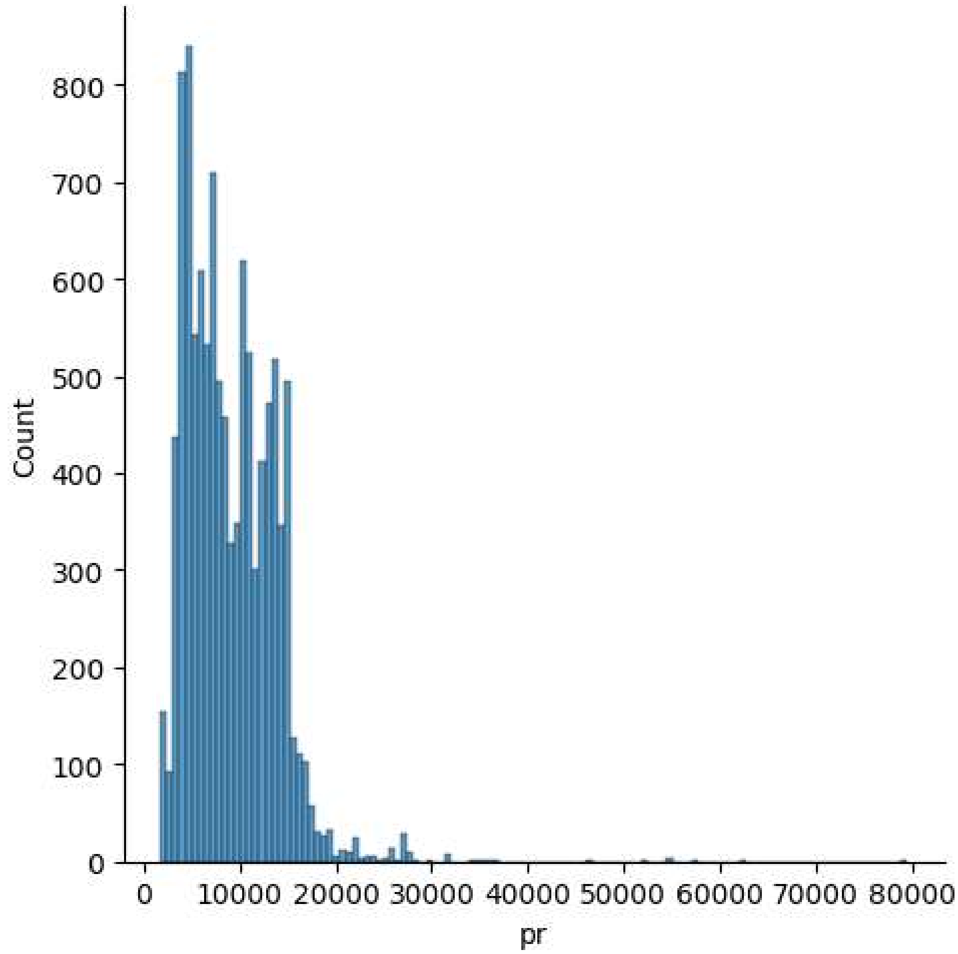
In [18]:
```
sns.pairplot(df)
```

Out[18]: <seaborn.axisgrid.PairGrid at 0x1c63c3b2680>



In [19]:
```
sns.displot(df['pr'])
```

Out[19]: <seaborn.axisgrid.FacetGrid at 0x1c63b07af20>



In [21]:
```
df.isnull().sum()
```

Out[21]:
```
ts    0
pr    0
dtype: int64
```

In [22]:
```
lm.intercept_
```

Out[22]:
```
array([5675.4619719])
```

Ridge Regression

In [23]:
```
from sklearn.linear_model import Ridge
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

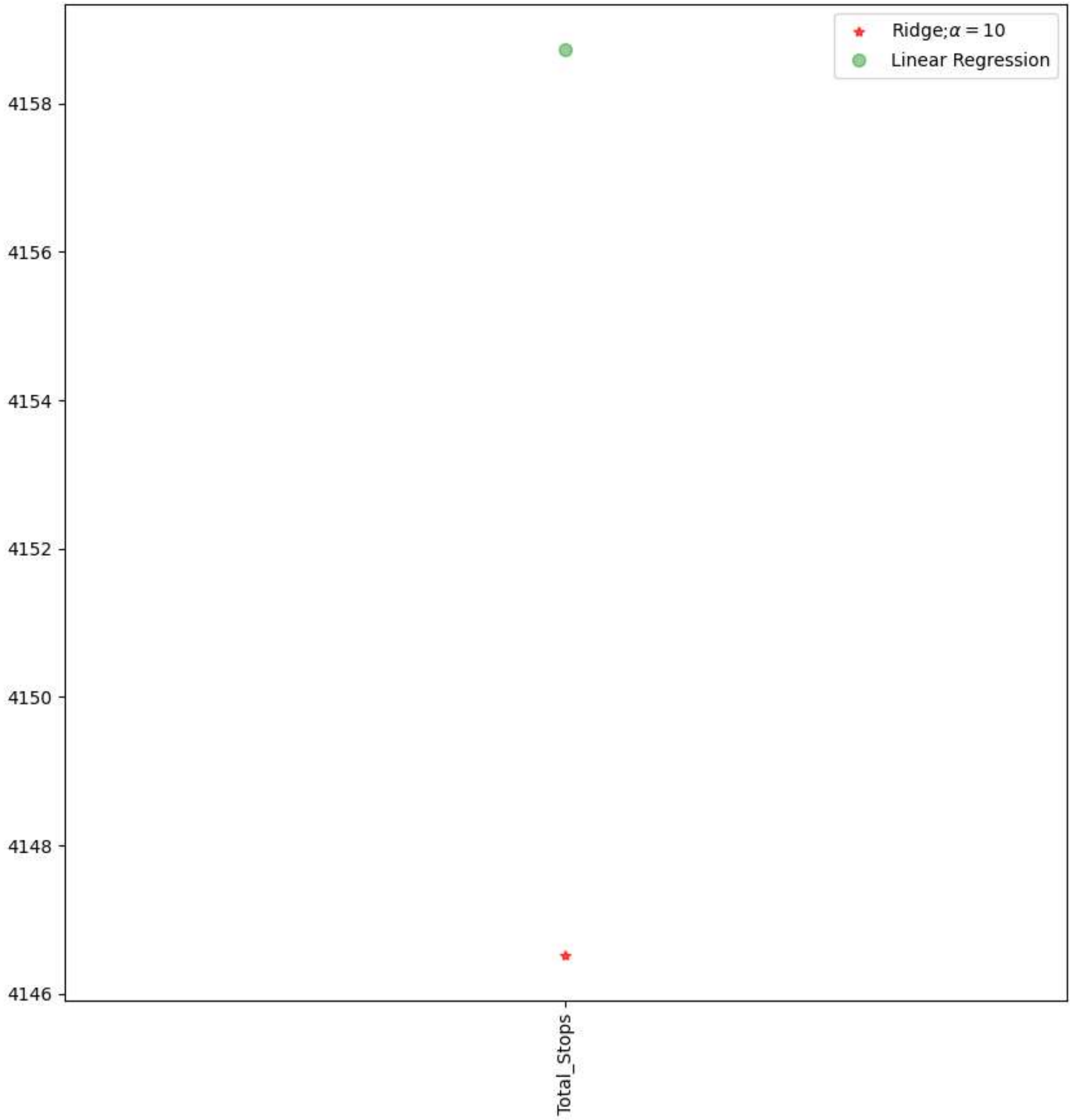
In [24]:
```
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(X_train,y_train)
train_score_ridge = ridgeReg.score(X_train,y_train)
test_score_ridge = ridgeReg.score(X_test,y_test)
print('\nRidge Model\n')
print('Train score for ridge model is {}' .format(train_score_ridge))
print('Test score for ridge model is {}' .format(test_score_ridge))
```

```

Ridge Model

Train score for ridge model is 0.3641839775336505
Test score for ridge model is 0.36572470289062164
```

In [25]:
```
plt.figure(figsize = (10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markers
plt.plot(features,lm.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



LASSO REGRESSION

In [26]:
```
from sklearn.linear_model import Lasso
```

In [27]:
```
lassoReg = Lasso(alpha=10)
lassoReg.fit(X_train,y_train)
train_score_lasso = lassoReg.score(X_train,y_train)
test_score_lasso = lassoReg.score(X_test,y_test)
print('\nRidge Model\n')
print('Train score for lasso model is {}'.format(train_score_lasso))
print('Test score for lasso model is {}'.format(test_score_lasso))
```

```
Ridge Model

Train score for lasso model is 0.36417691179849243
Test score for lasso model is 0.365767423447269
```

ELASTICNET

In [28]:
```
from sklearn.linear_model import ElasticNet
```

In [29]:
```
regr=ElasticNet()
regr.fit(X,y)
```

Out[29]:
```
▾ ElasticNet
ElasticNet()
```

In [30]:
```
regr.coef_
```

Out[30]:
```
array([1966.41150845])
```

In [31]:
```
regr.intercept_
```

Out[31]:
```
array([7466.51868198])
```

In [32]:
```
regr.predict(X_train)
```

Out[32]:
```
array([9432.93019043, 7466.51868198, 9432.93019043, ..., 7466.51868198,
       9432.93019043, 7466.51868198])
```

In [34]:
```
regr.score(X_train,y_train)
```

Out[34]:
```
0.2629640506723836
```

CONCLUSION

FROM THE ABOVE DATA FRAME, THE SCORE OF LINEAR REGRESSION, RIDGE REGRESSION AND LASSO REGRESSION ARE 36% AND ELASTICNET IS 26%. COMPARE TO THE ELASTICNET ALL LINEAR, RIDGE AND LASSO WERE HIGHEST.