Flight Price Prediction

Linear Regression ¶

Problem Statement: Based on the Total_stops how the price is varying.

In [157]:
```
# importing libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [5]:
```
# Data Collection
df=pd.read_excel(r"C:\Users\yoshitha lakshmi\OneDrive\Desktop\Data_Train.xlsx")
df
```

Out[5]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302
...	...	...	...	...	...	...	...	...	...	...	...
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU → BLR	19:55	22:25	2h 30m	non-stop	No info	4107
10679	Air India	27/04/2019	Kolkata	Banglore	CCU → BLR	20:45	23:20	2h 35m	non-stop	No info	4145
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR → DEL	08:20	11:20	3h	non-stop	No info	7229
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR → DEL	11:30	14:10	2h 40m	non-stop	No info	12648
10682	Air India	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	10:55	19:15	8h 20m	2 stops	No info	11753

10683 rows × 11 columns

In [ ]:
```
# Data cleaning
```

In [6]:
```
df.head()
```

Out[6]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302

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

Out[7]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU → BLR	19:55	22:25	2h 30m	non-stop	No info	4107
10679	Air India	27/04/2019	Kolkata	Banglore	CCU → BLR	20:45	23:20	2h 35m	non-stop	No info	4145
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR → DEL	08:20	11:20	3h	non-stop	No info	7229
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR → DEL	11:30	14:10	2h 40m	non-stop	No info	12648
10682	Air India	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	10:55	19:15	8h 20m	2 stops	No info	11753

In [8]:

```
df.describe()
```

Out[8]:

	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 [9]:

```
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 [11]:
```
df.shape
```

Out[11]: (10683, 11)

In [120]:
```
convert={'Total_Stops':{'non-stop':0,'1 stop':1,'2 stops':2,'3 stops':3,'4 stops':4}}
df=df.replace(convert)
df
```

Out[120]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	1.0	1	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	3.0	1	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	3.0	1	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	2.0	1	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	2.0	1	13302
...	...	...	...	...	...	...	...	...	...	...	...
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU → BLR	19:55	22:25	2h 30m	1.0	1	4107
10679	Air India	27/04/2019	Kolkata	Banglore	CCU → BLR	20:45	23:20	2h 35m	1.0	1	4145
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR → DEL	08:20	11:20	3h	1.0	1	7229
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR → DEL	11:30	14:10	2h 40m	1.0	1	12648
10682	Air India	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	10:55	19:15	8h 20m	3.0	1	11753

10683 rows × 11 columns

In [124]:
```
features=df['Total_Stops']
target=df.columns[-1]
```

In [125]:
```
df=df[['Total_Stops','Price']]
df.columns=['TS','prc']
```

In [126]:
```
df.fillna(method='ffill',inplace=True)
```

```
C:\Users\yoshitha lakshmi\AppData\Local\Temp\ipykernel_16700\4116506308.py:1: 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-docs/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)
  df.fillna(method='ffill',inplace=True)
```

In [127]:
```
X = np.array(df['TS']).reshape(-1,1)
y = np.array(df['prc']).reshape(-1,1)
```

In [159]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [130]:
```
X_train,x_test,y_train,y_test = train_test_split(X,y,train_size=0.9)
regr = LinearRegression()
regr.fit(X_train,y_train)
print(regr.score(x_test, y_test))
```

```
0.4034661319970495
```

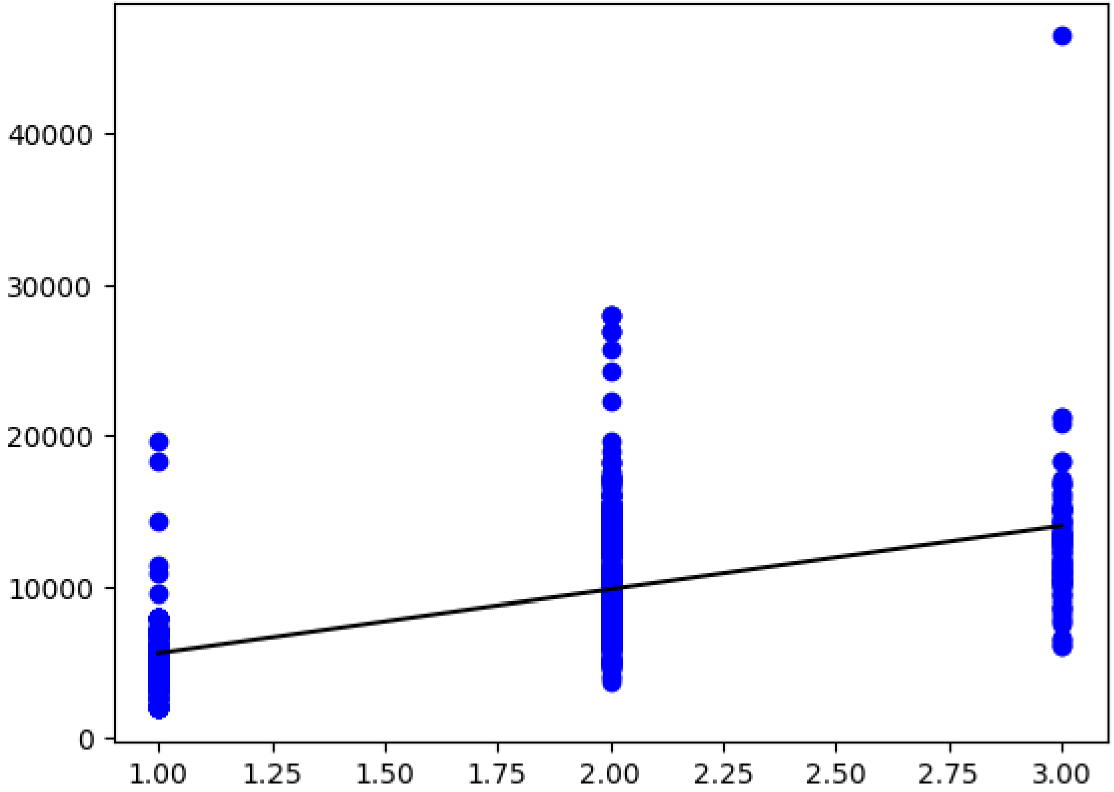
In [134]:

```
y_pred = regr.predict(x_test)

plt.scatter(x_test, y_test, color ='b')

plt.plot(x_test, y_pred, color ='k')

plt.show()
```



In [145]:
```
coeff_df=pd.DataFrame(regr.coef_)
coeff_df
```

Out[145]:

	0
0	1978.124921

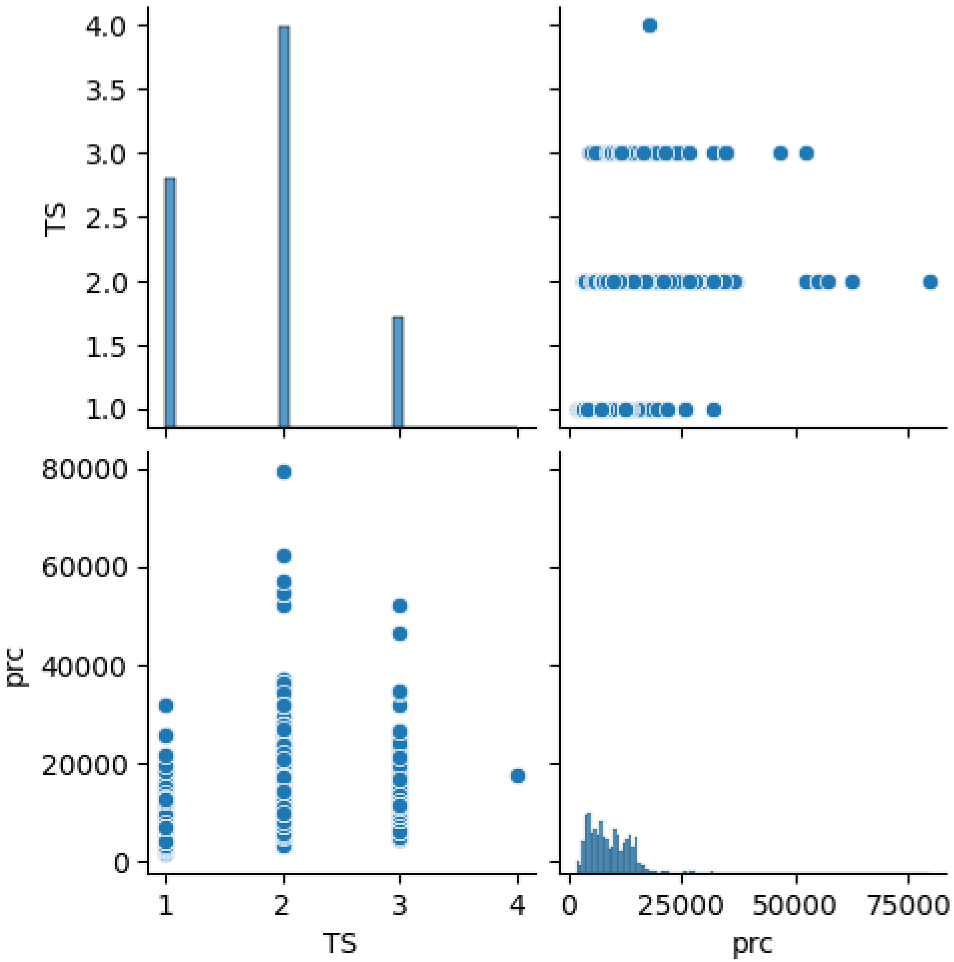
Conclusion

For Linear Regression the accuracy is 40%.

Exploratory data analysis

In [161]:
```
sns.pairplot(df)
```

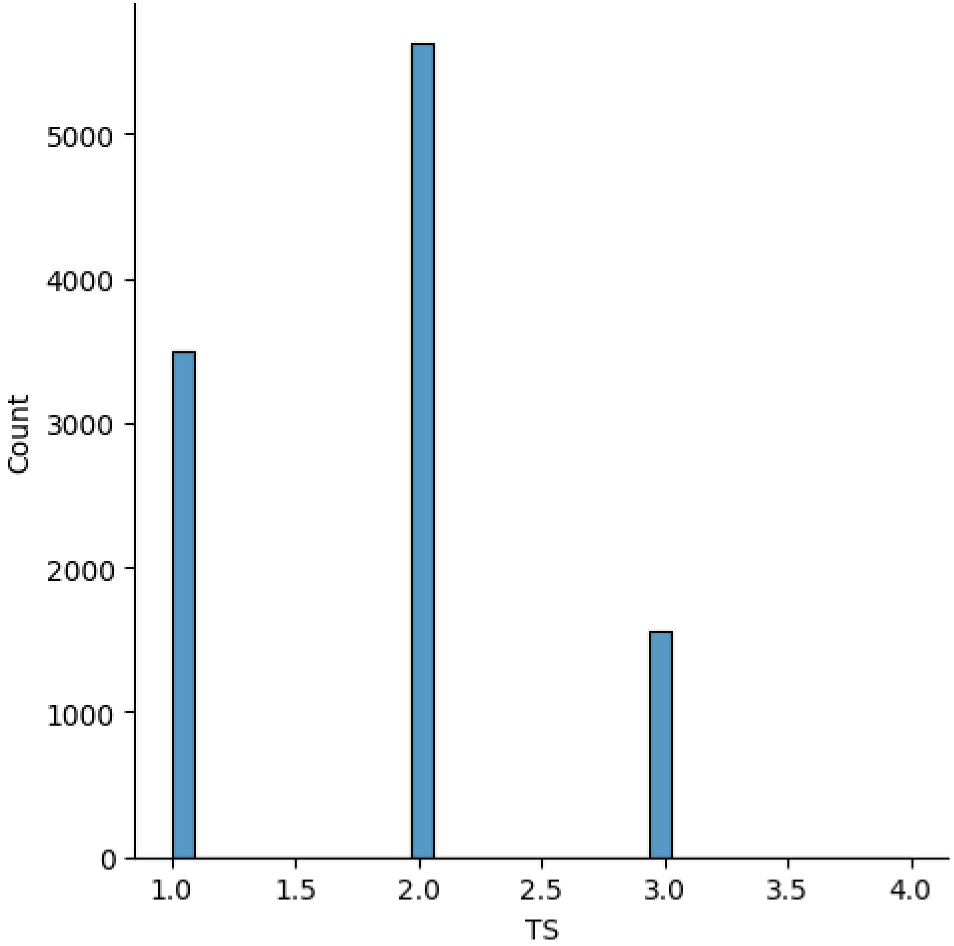
Out[161]: <seaborn.axisgrid.PairGrid at 0x231042b3550>



In [164]:

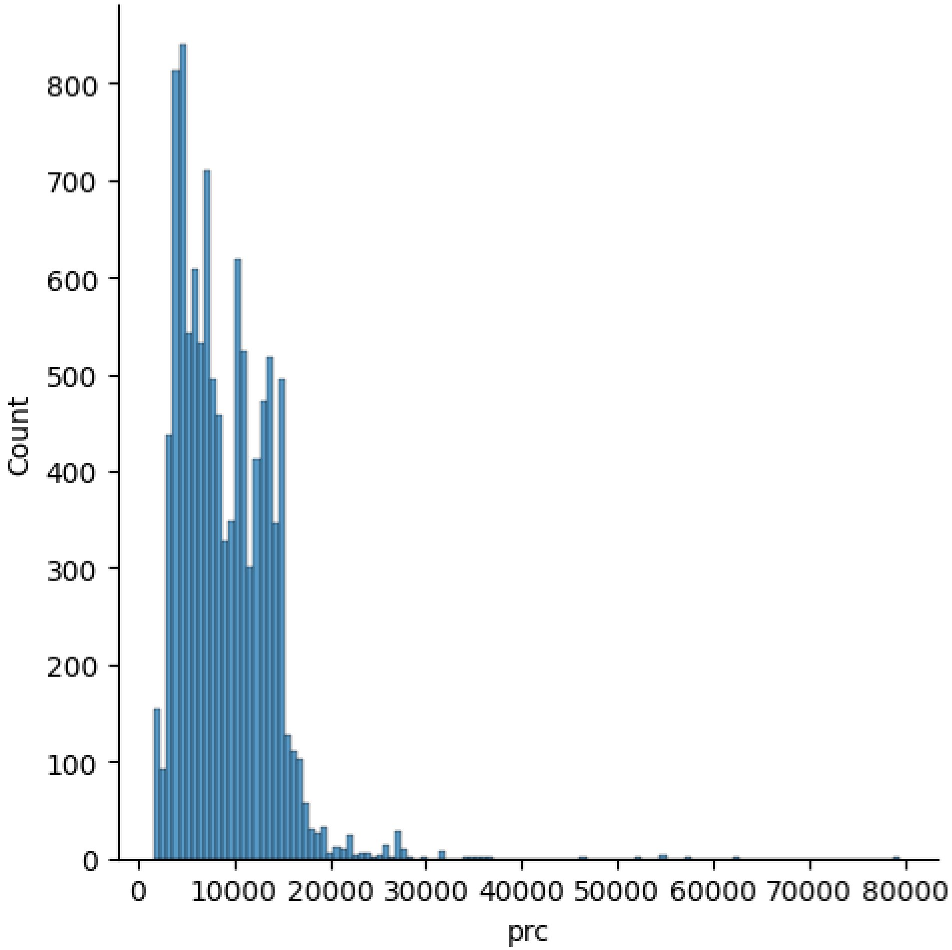
```
sns.displot(df['TS'])
```

Out[164]: <seaborn.axisgrid.FacetGrid at 0x23105b69f30>



In [166]:
```
sns.displot(df['prc'])
```

Out[166]: <seaborn.axisgrid.FacetGrid at 0x23103f7a470>



Ridge and Lasso, Elastic Net

In [135]:

```
from sklearn.linear_model import Ridge,RidgeCV,Lasso
```

In [139]:

```
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 0.36592181213396213
Test score for ridge model is 0.4033942075452617
```

In [140]:

```
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('\nLasso Model\n')
print('Train score for lasso model is {}'.format(train_score_lasso))
print('Test score for lasso model is {}'.format(test_score_lasso))
```

```
Lasso Model

Train score for lasso model is 0.3659131768191236
Test score for lasso model is 0.40329509993506674
```

In [148]:
```
# Elastic Net
from sklearn.linear_model import ElasticNet
regr = ElasticNet()
regr.fit(X,y)
print(regr.coef_)
print(regr.intercept_)
y_pred_elastic = regr.predict(X_train)
mean_squared_error = np.mean((y_pred_elastic-y_train)**2)
print('Mean squared error on test set',mean_squared_error)
regr.score(X_train,y_train)
```

```
[1978.1249211]
[5487.25820546]
Mean squared error on test set 23004190.962821722
```

Out[148]:
```
0.26334799863145886
```

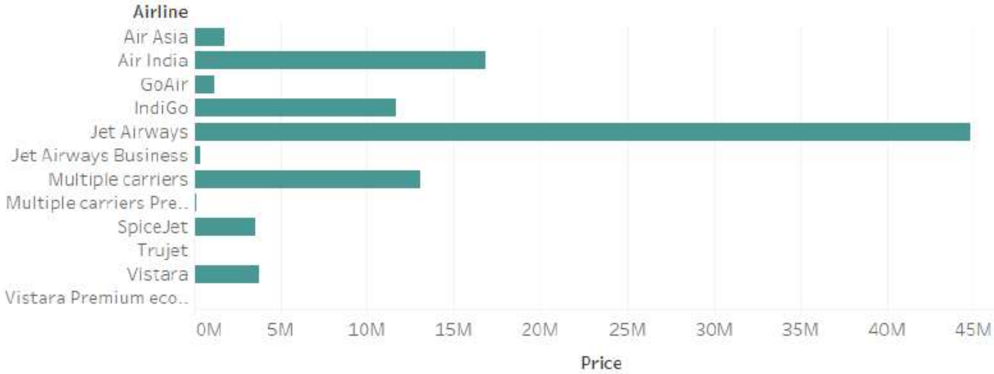
Conclusion

For Ridge and Lasso Regression the score is comparatively same but for Elastic Net the score is low compared to both Ridge and Lasso.

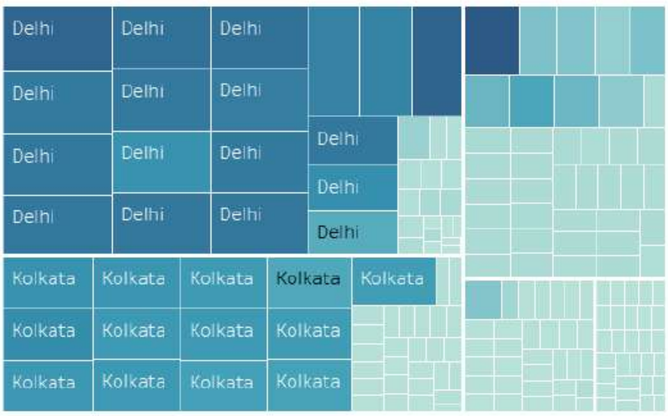
In [ ]:
```

```

Sheet 1



Sheet 2



Price

8,635 — 33,26,651

Sheet 3

Route	Airline	
Null	Air India	7,480
BLR → AMD → DEL	Air India	76,488
	IndiGo	28,746
	Vistara	95,181
BLR → BBI → DEL	Air India	57,430
BLR → BDQ → DEL	Jet Airways	92,404
BLR → BOM → AMD ..	Air India	73,758
BLR → BOM → BHO ..	Air India	1,89,327
BLR → BOM → DEL	Air India	1,15,125
	IndiGo	52,529
	Jet Airways	59,58,894
	Jet Airways Business	1,94,168
BLR → BOM → IDR →..	Air India	76,803
BLR → BOM → IDR →..	Air India	26,774
BLR → BOM → IXC →..	Air India	13,303
BLR → BOM → JDH →	Air India	38,491
DEL	Jet Airways	50,584
BLR → BOM → NAG ..	Air India	1,44,946
BLR → BOM → UDR ..	Air India	35,813
BLR → CCU → BBI → ..	Air India	45,775
BLR → CCU → BBI → ..	Air India	25,427
BLR → CCU → BBI → ..	Air India	17,686
BLR → CCU → DEL	Air India	1,34,863
BLR → CCU → GAU →..	Air India	1,29,910
BLR → COK → DEL	Air India	1,37,087
BLR → DEL	Air Asia	4,07,111
	Air India	10,62,122
	GoAir	4,24,266
	IndiGo	24,43,515
	Jet Airways	24,83,273
	SpiceJet	7,44,280
	Vistara	10,48,521