

Problem Statement:-

For each Airline mostly having how many stops

In [1]: ▶

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from sklearn.linear_model import LinearRegression
6 from sklearn.model_selection import train_test_split
```

In [2]: ▶

```
1 df = pd.read_excel(r"C:\Users\HP\Downloads\Data_Train.xlsx")
2 df
```

Out[2]:

	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 [3]: ▶

```
1 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 [4]: 1 df.isnull().sum()
```

```
Out[4]: 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 [5]: 1 df.fillna(method='ffill',inplace=True)
```

```
In [6]: 1 df.isnull().sum()
```

```
Out[6]: 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 [7]: 1 df['Airline'].value_counts()
```

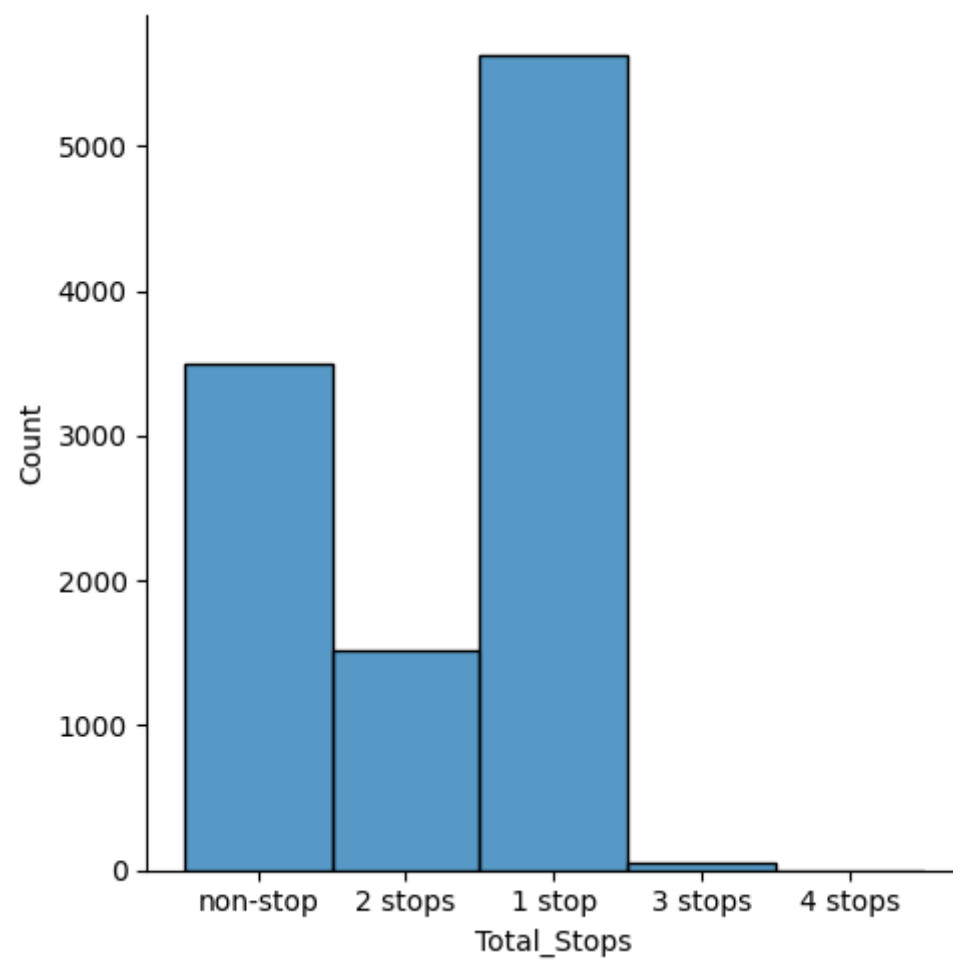
```
Out[7]: Airline
Jet Airways      3849
IndiGo           2053
Air India        1752
Multiple carriers 1196
SpiceJet          818
Vistara           479
Air Asia         319
GoAir            194
Multiple carriers Premium economy 13
Jet Airways Business 6
Vistara Premium economy 3
Trujet           1
Name: count, dtype: int64
```

```
In [8]: 1 df['Total_Stops'].value_counts()
```

```
Out[8]: Total_Stops
1 stop      5625
non-stop    3492
2 stops     1520
3 stops      45
4 stops       1
Name: count, dtype: int64
```

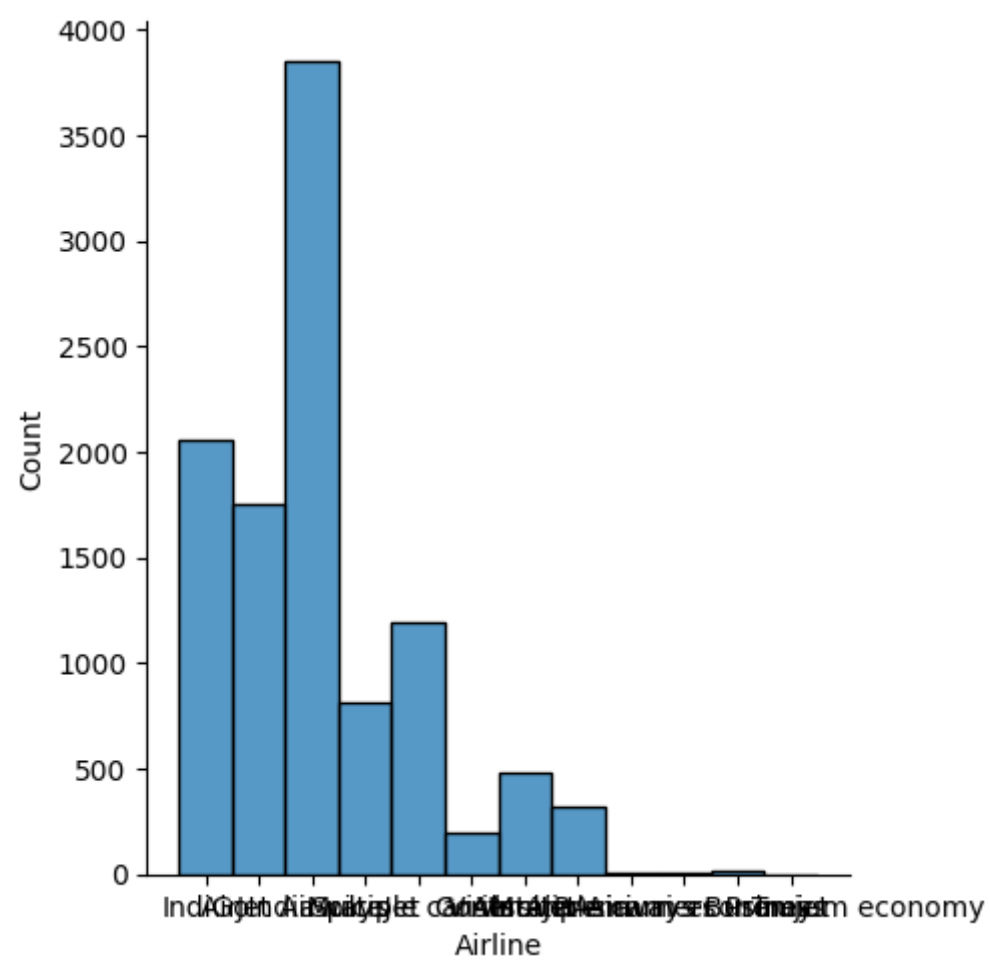
```
1 sns.displot(df['Total_Stops'])
```

```
Out[9]: <seaborn.axisgrid.FacetGrid at 0x1c9db51ffa0>
```



```
1 sns.displot(df['Airline'])
2
```

```
Out[10]: <seaborn.axisgrid.FacetGrid at 0x1c9db726bc0>
```



```
In [11]: 1 s={'Airline':{'Jet Airways':1,'IndiGo':2,'Air India':3,'Multiple carriers':4,'SpiceJet':5,'Vistara':6,'Air As
2 df = df.replace(s)
3 print(df)
```

	Airline	Date_of_Journey	Source	Destination	Route	
0	2	24/03/2019	Banglore	New Delhi	BLR → DEL	\
1	3	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	
2	1	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	
3	2	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	
4	2	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	
...	...	...	...	...	...	...
10678	7	9/04/2019	Kolkata	Banglore	CCU → BLR	
10679	3	27/04/2019	Kolkata	Banglore	CCU → BLR	
10680	1	27/04/2019	Banglore	Delhi	BLR → DEL	
10681	6	01/03/2019	Banglore	New Delhi	BLR → DEL	
10682	3	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	

	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897
1	05:50	13:15	7h 25m	2 stops	No info	7662
2	09:25	04:25 10 Jun	19h	2 stops	No info	13882
3	18:05	23:30	5h 25m	1 stop	No info	6218
4	16:50	21:35	4h 45m	1 stop	No info	13302
...	...	...	...	...	...	...
10678	19:55	22:25	2h 30m	non-stop	No info	4107
10679	20:45	23:20	2h 35m	non-stop	No info	4145
10680	08:20	11:20	3h	non-stop	No info	7229
10681	11:30	14:10	2h 40m	non-stop	No info	12648
10682	10:55	19:15	8h 20m	2 stops	No info	11753

[10683 rows x 11 columns]

```
In [12]: 1 t={'Total_Stops':{'1 stop':1,'2 stops':2,'3 stops':3,'4 stops':4,'non-stop':5}}
2 df = df.replace(t)
3 print(df)
```

	Airline	Date_of_Journey	Source	Destination	Route	
0	2	24/03/2019	Banglore	New Delhi	BLR → DEL	\
1	3	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	
2	1	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	
3	2	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	
4	2	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	
...	...	...	...	...	...	...
10678	7	9/04/2019	Kolkata	Banglore	CCU → BLR	
10679	3	27/04/2019	Kolkata	Banglore	CCU → BLR	
10680	1	27/04/2019	Banglore	Delhi	BLR → DEL	
10681	6	01/03/2019	Banglore	New Delhi	BLR → DEL	
10682	3	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	

	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	22:20	01:10 22 Mar	2h 50m	5	No info	3897
1	05:50	13:15	7h 25m	2	No info	7662
2	09:25	04:25 10 Jun	19h	2	No info	13882
3	18:05	23:30	5h 25m	1	No info	6218
4	16:50	21:35	4h 45m	1	No info	13302
...	...	...	...	...	...	...
10678	19:55	22:25	2h 30m	5	No info	4107
10679	20:45	23:20	2h 35m	5	No info	4145
10680	08:20	11:20	3h	5	No info	7229
10681	11:30	14:10	2h 40m	5	No info	12648
10682	10:55	19:15	8h 20m	2	No info	11753

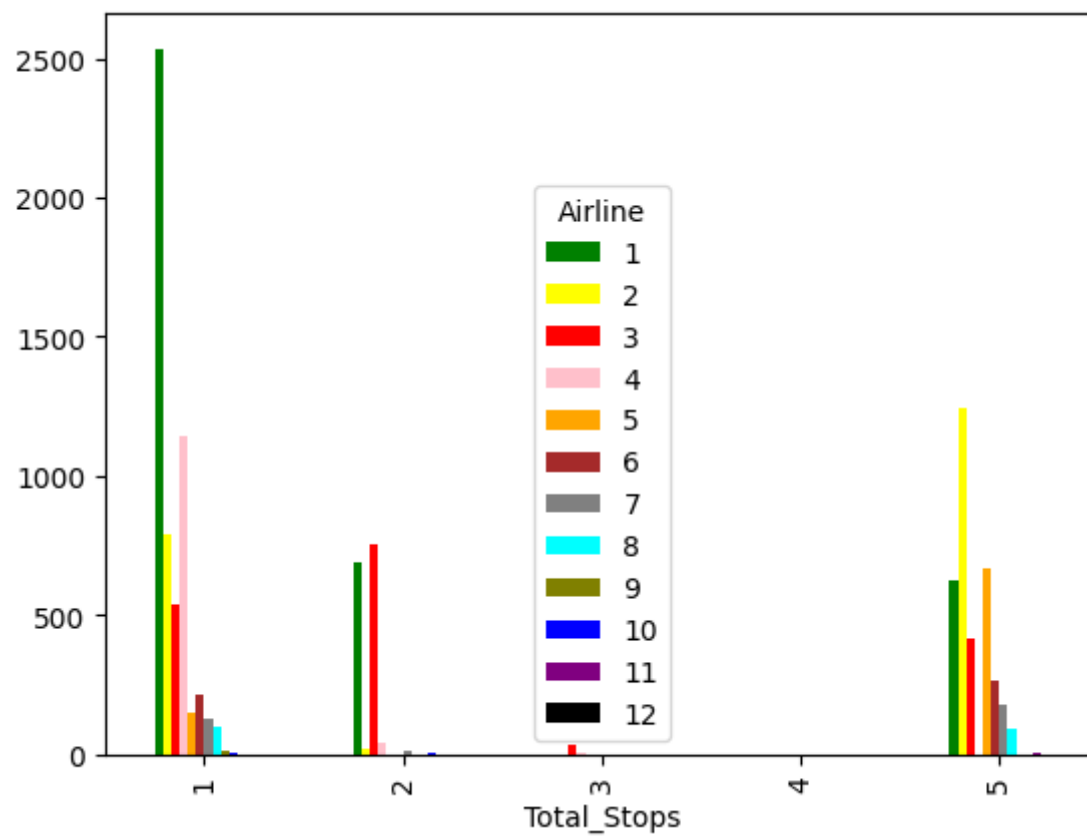
[10683 rows x 11 columns]

```
In [13]: 1 r=pd.crosstab(df['Total_Stops'],df['Airline'])
2 print(r)
```

Airline	1	2	3	4	5	6	7	8	9	10	11	12
Total_Stops												
1	2535	793	540	1145	148	215	129	102	13	4	0	1
2	691	19	756	43	0	0	9	0	0	2	0	0
3	0	0	37	8	0	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0	0
5	623	1241	418	0	670	264	181	92	0	0	3	0

```
In [14]: 1 r.plot(kind='bar', stacked=False, color=['green', 'yellow', 'red', 'pink', 'orange', 'brown', 'gray', 'cyan', 'olive']
```

Out[14]: <Axes: xlabel='Total\_Stops'>



```
In [15]: 1 x=df[['Total_Stops','Airline']]
2 y=df[['Price']]
```

```
In [16]: 1 x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

```
In [17]: 1 lr=LinearRegression()
2 lr.fit(x_train,y_train)
```

Out[17]: 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 [18]: 1 print(lr.score(x_train,y_train))
2 print(lr.score(x_test,y_test))
```

0.3588965505895584  
0.3547193238050983

```
In [19]: 1 from sklearn.linear_model import LogisticRegression
2 import warnings
3 warnings.simplefilter(action='ignore')
```

```
In [20]: 1 lg=LogisticRegression()
2 lg.fit(x_train,y_train)
3 print(lg.score(x_test,y_test))
```

0.08985959438377535

```
In [28]: 1 from sklearn.ensemble import RandomForestClassifier
2 rfc=RandomForestClassifier()
3 rfc.fit(x_train,y_train)
4 print(rfc.score(x_test,y_test))
5 print(rfc.score(x_train,y_train))
```

0.10514820592823713  
0.11152714629580102

```
In [30]: 1 from sklearn.tree import DecisionTreeClassifier
2 dt = DecisionTreeClassifier()
3 dt.fit(x_train,y_train)
4 print(dt.score(x_test,y_test))
5 print(dt.score(x_train,y_train))
```

0.10546021840873634  
0.11152714629580102

```
In [21]: 1 from sklearn.linear_model import Lasso,Ridge,LassoCV,RidgeCV
```

```
In [22]: 1 lasso = Lasso(alpha=10)
2 lasso.fit(x_train,y_train)
3 print(lasso.score(x_test,y_test))
4 print(lasso.score(x_train,y_train))
```

```
0.3547509900593633
0.3588942010344687
```

```
In [23]: 1 ridge = Ridge(alpha=10)
2 ridge.fit(x_train,y_train)
3 print(ridge.score(x_test,y_test))
4 print(ridge.score(x_train,y_train))
```

```
0.3547234452726441
0.3588965011593923
```

```
In [24]: 1 lasso_cv = LassoCV(alphas=[2,5,10,20,40,50])
2 lasso_cv.fit(x_train,y_train)
3 print(lasso_cv.score(x_test,y_test))
4 print(lasso_cv.score(x_train,y_train))
```

```
0.35472602049387103
0.35889645718648144
```

```
In [26]: 1 ridge_cv = RidgeCV(alphas=[2,5,10,20,40,50])
2 ridge_cv.fit(x_train,y_train)
3 print(ridge_cv.score(x_test,y_test))
4 print(ridge_cv.score(x_train,y_train))
```

```
0.3547274620637978
0.35889635302276923
```

```
In [31]: 1 from sklearn.linear_model import ElasticNet
2 en=ElasticNet()
3 en.fit(x_train,y_train)
4 print(en.score(x_train,y_train))
5 print(en.score(x_test,y_test))
```

```
0.35360949118514984
0.3506565913735198
```

## Conclusion:-

For Jet Airways Airline mostly having Single stop  
For Air India Airline mostly having 2-stops  
only Air India Airline having 3 stops  
IndiGo this Airline is non-stop

```
In [ ]: 1
```