Part 1

Busniess problem
Importing the data
Exploring the data

In [1]:
```
#import required library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_excel(r'C:\Users\cws\OneDrive\Desktop\Corico Data\Data_Train.xlsx')

df.head()
```

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

In [3]:
```
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]:
```
df.shape
```

Out[4]:
```
(10683, 11)
```

In [5]:
```
df.count()
```

Out[5]:
```
Airline            10683
Date_of_Journey    10683
Source             10683
Destination        10683
Route              10682
Dep_Time           10683
Arrival_Time       10683
Duration           10683
Total_Stops        10682
Additional_Info    10683
Price              10683
dtype: int64
```

In [6]:
```
df.dtypes
```

Out[6]:
```
Airline            object
Date_of_Journey    object
Source             object
Destination        object
Route              object
Dep_Time           object
Arrival_Time       object
Duration           object
Total_Stops        object
Additional_Info    object
Price               int64
dtype: object
```

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

Out[7]:

	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 [8]:
```
#missing value in dataset
df.isna()
```

Out[8]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...
10678	False	False	False	False	False	False	False	False	False	False	False
10679	False	False	False	False	False	False	False	False	False	False	False
10680	False	False	False	False	False	False	False	False	False	False	False
10681	False	False	False	False	False	False	False	False	False	False	False
10682	False	False	False	False	False	False	False	False	False	False	False

10683 rows × 11 columns

In [9]:
```
#missing value in dataset sum
df.isna().sum()
```

Out[9]:
```
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 [10]:
```
#find missing value for specific flight
df[df['Route'].isna()|df['Total_Stops'].isna()]
```

Out[10]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
9039	Air India	6/05/2019	Delhi	Cochin	NaN	09:45	09:25 07 May	23h 40m	NaN	No info	7480

In [11]:
```
#drop missing value
df.dropna(inplace = True)
```

In [12]:
```
df.isna().sum()
```

Out[12]:
```
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 [13]:
```
df.count()
```

Out[13]:
```
Airline            10682
Date_of_Journey    10682
Source             10682
Destination        10682
Route              10682
Dep_Time           10682
Arrival_Time       10682
Duration           10682
Total_Stops        10682
Additional_Info    10682
Price              10682
dtype: int64
```

Part 2

EDA
Feature Engineering
Analytics

In [14]:
```
df.head()
```

Out[14]:

	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

Exploratoring data analysis & feature engineering

1.Duration

2.Departure & Arrival time

3.Date of Journey

4.Toatl stops

5.Additional Info

6.Airline

7.Source & Destination

8.Route

Duration

In [15]:
```
def convert_duration(duration):
    if len(duration.split())  == 2:
        hours = int(duration.split()[0][: -1])
        minutes = int(duration.split()[1][: -1])
        return hours * 60 + minutes
    else:
        return int(duration[: -1]) * 60
```

In [16]:
```
df['Duration'] = df['Duration'].apply(convert_duration)
df.head()
```

Out[16]:

	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	170	non-stop	No info	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	445	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	1140	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	325	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	285	1 stop	No info	13302

DEPARTURE AND ARRIVAL TIME

In [17]:
```
df['Dep_Time'] = pd.to_datetime(df['Dep_Time'])
df['Arrival_Time'] = pd.to_datetime(df['Arrival_Time'])
df.dtypes
```

Out[17]:
```
Airline                    object
Date_of_Journey            object
Source                     object
Destination                object
Route                      object
Dep_Time           datetime64[ns]
Arrival_Time       datetime64[ns]
Duration                    int64
Total_Stops                object
Additional_Info            object
Price                       int64
dtype: object
```

In [18]:
```
df['Dep_Time_in_hours'] = df['Dep_Time'].dt.hour
df['Dep_Time_in_minutes'] = df['Dep_Time'].dt.minute
df['Arrival_Time_in_hours'] = df['Arrival_Time'].dt.hour
df['Arrival_Time_in_minutes'] = df['Arrival_Time'].dt.minute

df.head()
```

Out[18]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Dep_Time_
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	2023-10-13 22:20:00	2023-03-22 01:10:00	170	non-stop	No info	3897	
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	2023-10-13 05:50:00	2023-10-13 13:15:00	445	2 stops	No info	7662	
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	2023-10-13 09:25:00	2023-06-10 04:25:00	1140	2 stops	No info	13882	
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	2023-10-13 18:05:00	2023-10-13 23:30:00	325	1 stop	No info	6218	
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	2023-10-13 16:50:00	2023-10-13 21:35:00	285	1 stop	No info	13302	

In [19]:
```
df.drop(['Dep_Time', 'Arrival_Time'], axis = 1, inplace = True)
df.head()
```

Out[19]:

	Airline	Date_of_Journey	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Dep_Time_in_hours	Dep_Time_in_m
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	170	non-stop	No info	3897	22	
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2 stops	No info	7662	5	
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2 stops	No info	13882	9	
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	325	1 stop	No info	6218	18	
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	285	1 stop	No info	13302	16	

DATE OF JOURNEY

In [20]:
```
df['Date_of_Journey'] = pd.to_datetime(df['Date_of_Journey'])
df.head()
```

```
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1671732385.py:1: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consistent parsing.
  df['Date_of_Journey'] = pd.to_datetime(df['Date_of_Journey'])
```

Out[20]:

	Airline	Date_of_Journey	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Dep_Time_in_hours	Dep_Time_in_m
0	IndiGo	2019-03-24	Banglore	New Delhi	BLR → DEL	170	non-stop	No info	3897	22	
1	Air India	2019-01-05	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2 stops	No info	7662	5	
2	Jet Airways	2019-09-06	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2 stops	No info	13882	9	
3	IndiGo	2019-12-05	Kolkata	Banglore	CCU → NAG → BLR	325	1 stop	No info	6218	18	
4	IndiGo	2019-01-03	Banglore	New Delhi	BLR → NAG → DEL	285	1 stop	No info	13302	16	

In [21]:
```
df['Date_of_Journey'].dt.year.unique()
```

Out[21]:
```
array([2019], dtype=int64)
```

In [22]:
```
df['Day'] = df['Date_of_Journey'].dt.day
df['Month'] = df['Date_of_Journey'].dt.month

df.head()
```

Out[22]:

	Airline	Date_of_Journey	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Dep_Time_in_hours	Dep_Time_in_m
0	IndiGo	2019-03-24	Banglore	New Delhi	BLR → DEL	170	non-stop	No info	3897	22	
1	Air India	2019-01-05	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2 stops	No info	7662	5	
2	Jet Airways	2019-09-06	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2 stops	No info	13882	9	
3	IndiGo	2019-12-05	Kolkata	Banglore	CCU → NAG → BLR	325	1 stop	No info	6218	18	
4	IndiGo	2019-01-03	Banglore	New Delhi	BLR → NAG → DEL	285	1 stop	No info	13302	16	

In [23]:
```
df.drop('Date_of_Journey', axis = 1, inplace = True)
df.head()
```

Out[23]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Tin
0	IndiGo	Banglore	New Delhi	BLR → DEL	170	non-stop	No info	3897	22	20	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2 stops	No info	7662	5	50	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2 stops	No info	13882	9	25	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325	1 stop	No info	6218	18	5	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	285	1 stop	No info	13302	16	50	

TOTAL STOPS

In [24]:
```
df['Total_Stops'].value_counts()
```

Out[24]:
```
1 stop      5625
non-stop    3491
2 stops     1520
3 stops       45
4 stops        1
Name: Total_Stops, dtype: int64
```

In [25]:
```
df['Total_Stops'] = df['Total_Stops'].map({
    'non-stop': 0,
    '1 stop': 1,
    '2 stops': 2,
    '3 stops': 3,
    '4 stops': 4
})

df.head()
```

Out[25]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Tin
0	IndiGo	Banglore	New Delhi	BLR → DEL	170	0	No info	3897	22	20	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	No info	7662	5	50	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	No info	13882	9	25	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325	1	No info	6218	18	5	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	285	1	No info	13302	16	50	

ADDITIONAL INFO

In [26]:
```
df['Additional_Info'].value_counts()
```

Out[26]:
```
No info                         8344
In-flight meal not included     1982
No check-in baggage included     320
1 Long layover                    19
Change airports                    7
Business class                     4
No Info                            3
1 Short layover                    1
Red-eye flight                     1
2 Long layover                     1
Name: Additional_Info, dtype: int64
```

In [27]:
```
df.drop('Additional_Info', axis = 1, inplace = True)
df.head()
```

Out[27]:

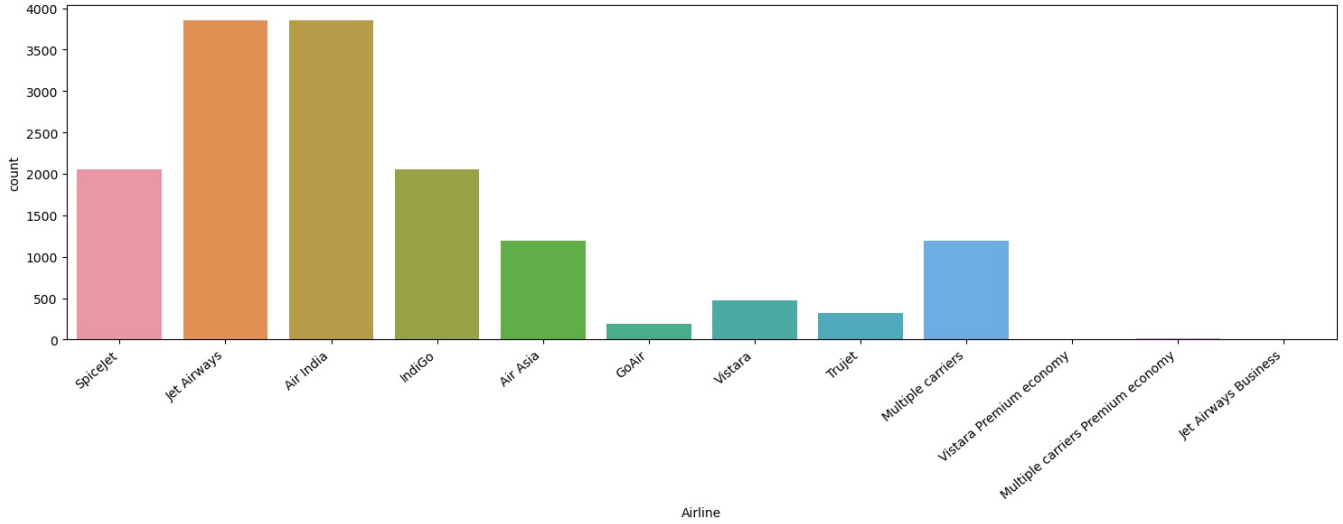
	Airline	Source	Destination	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arr
0	IndiGo	Banglore	New Delhi	BLR → DEL	170	0	3897	22	20	1	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	7662	5	50	13	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	13882	9	25	4	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325	1	6218	18	5	23	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	285	1	13302	16	50	21	

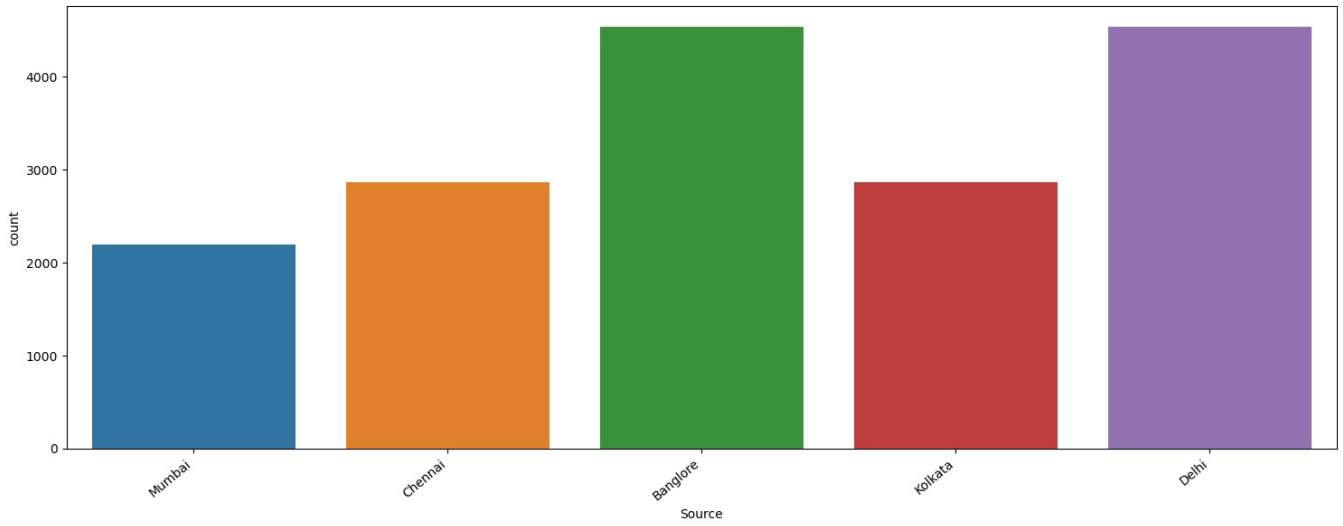
In [28]:
```
df.select_dtypes(['object']).columns
```

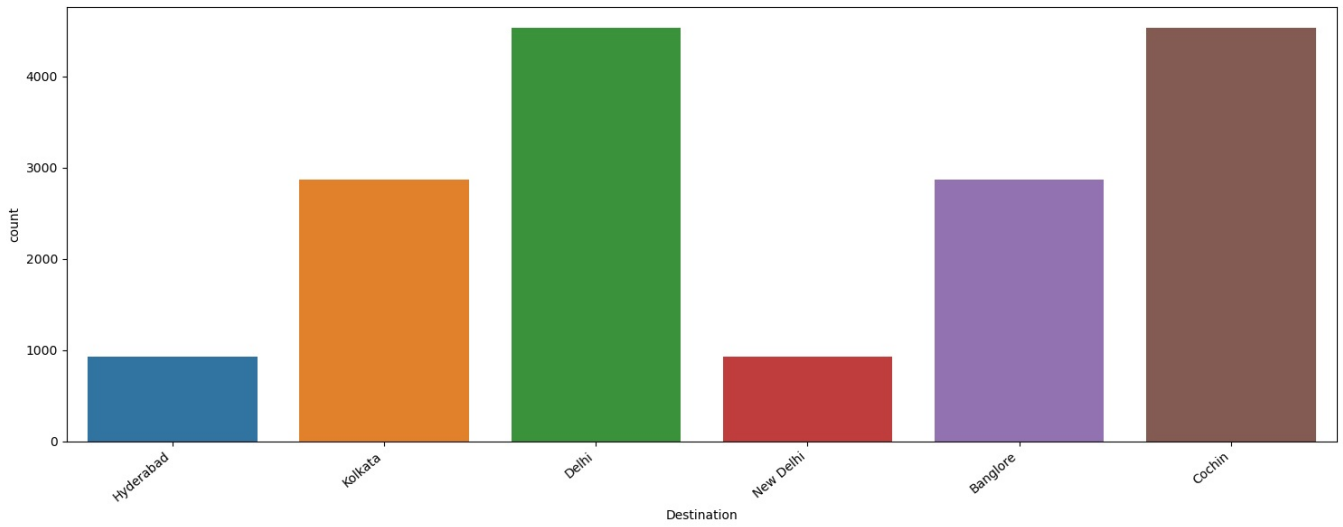
Out[28]:
```
Index(['Airline', 'Source', 'Destination', 'Route'], dtype='object')
```

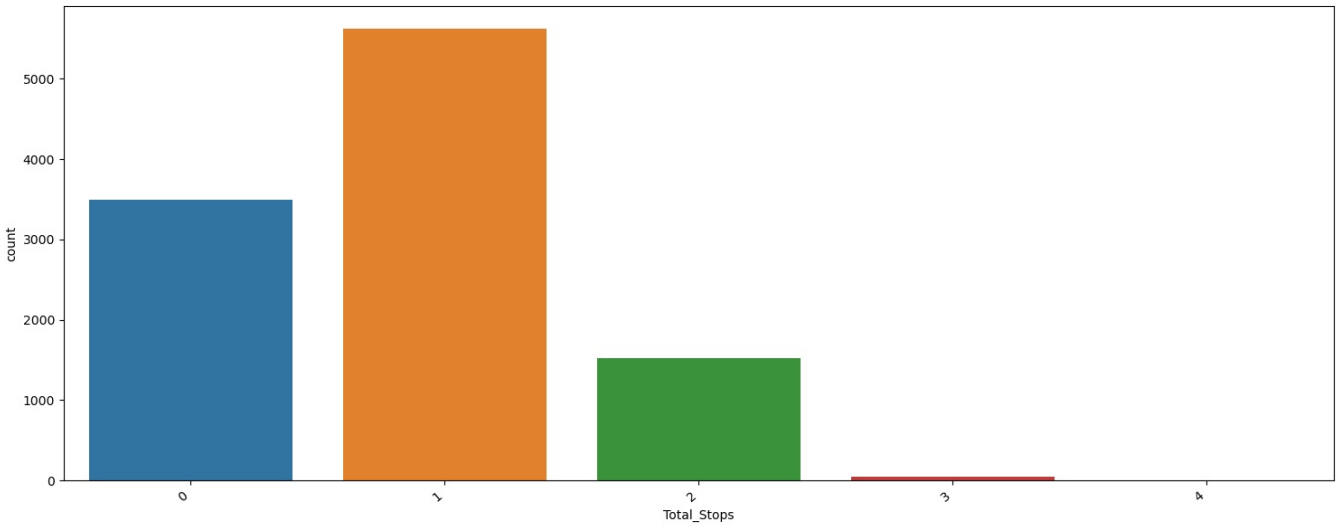
In [30]:
```
for i in ['Airline', 'Source', 'Destination', 'Total_Stops']:
```

```
In [29]: for i in ['Airline', 'Source', 'Destination', 'Total_Stops']:
             plt.figure(figsize = (15, 6))
             sns.countplot(data = df, x = i)
             ax = sns.countplot(x = i, data = df.sort_values('Price', ascending = True))
             ax.set_xticklabels(ax.get_xticklabels(), rotation = 40, ha = 'right')
             plt.tight_layout()
             plt.show()
             print('\n\n')
```







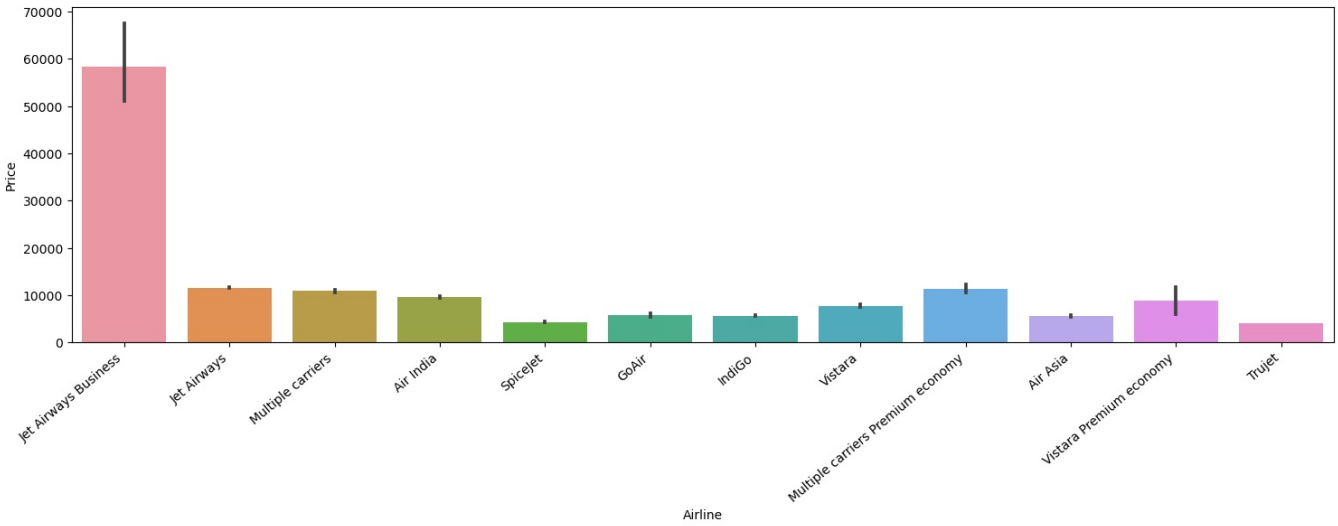


AIRLINE

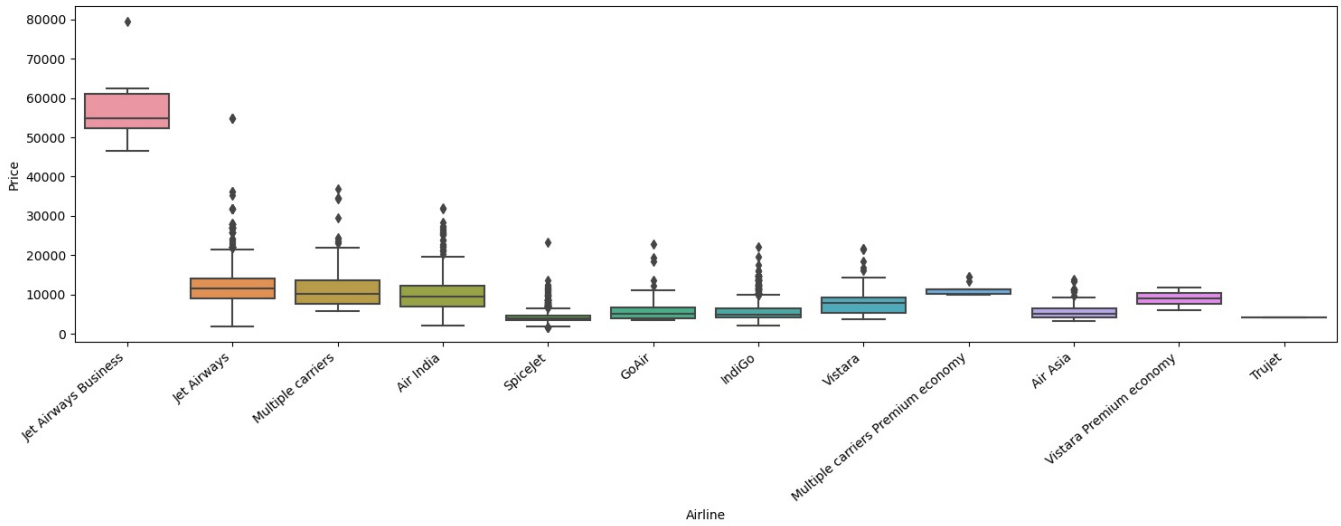
In [30]:
```
df['Airline'].value_counts()
```

Out[30]:
```
Jet Airways                          3849
IndiGo                               2053
Air India                            1751
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 [31]:
```
plt.figure(figsize = (15, 6))
ax = sns.barplot(x = 'Airline', y = 'Price', data = df.sort_values('Price', ascending = False))
ax.set_xticklabels(ax.get_xticklabels(), rotation = 40, ha = 'right')
plt.tight_layout()
plt.show()
```



In [32]:
```
plt.figure(figsize = (15, 6))
ax = sns.boxplot(x = 'Airline', y = 'Price', data = df.sort_values('Price', ascending = False))
ax.set_xticklabels(ax.get_xticklabels(), rotation = 40, ha = 'right')
plt.tight_layout()
plt.show()
```



In [33]:
```
df.groupby('Airline').describe()['Price'].sort_values('mean', ascending = False)
```

Out[33]:

Airline	count	mean	std	min	25%	50%	75%	max
Jet Airways Business	6.0	58358.666667	11667.596748	46490.0	52243.0	54747.0	61122.50	79512.0
Jet Airways	3849.0	11643.923357	4258.940578	1840.0	9134.0	11467.0	14151.00	54826.0
Multiple carriers Premium economy	13.0	11418.846154	1717.153936	9845.0	10161.0	11269.0	11269.00	14629.0
Multiple carriers	1196.0	10902.678094	3721.234997	5797.0	7723.0	10197.0	13587.00	36983.0
Air India	1751.0	9612.427756	3901.734561	2050.0	6891.0	9443.0	12219.00	31945.0
Vistara Premium economy	3.0	8962.333333	2915.405518	5969.0	7547.0	9125.0	10459.00	11793.0
Vistara	479.0	7796.348643	2914.298578	3687.0	5403.0	7980.0	9345.00	21730.0
GoAir	194.0	5861.056701	2703.585767	3398.0	3898.0	5135.0	6811.25	22794.0
IndiGo	2053.0	5673.682903	2264.142168	2227.0	4226.0	5000.0	6494.00	22153.0
Air Asia	319.0	5590.260188	2027.362290	3383.0	4282.0	5162.0	6451.00	13774.0
SpiceJet	818.0	4338.284841	1849.922514	1759.0	3574.5	3873.0	4760.00	23267.0
Trujet	1.0	4140.000000	NaN	4140.0	4140.0	4140.0	4140.00	4140.0

In [34]:
```
Airline = pd.get_dummies(df['Airline'], drop_first = True)
Airline.head()
```

Out[34]:

	Air India	GoAir	IndiGo	Jet Airways	Jet Airways Business	Multiple carriers	Multiple carriers Premium economy	SpiceJet	Trujet	Vistara	Vistara Premium economy
0	0	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0

In [35]:
```
df = pd.concat([df, Airline], axis = 1)
df.head()
```

Out[35]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	...
0	IndiGo	Banglore	New Delhi	BLR → DEL	170	0	3897	22	20	1	...
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	7662	5	50	13	...
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	13882	9	25	4	...
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325	1	6218	18	5	23	...
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	285	1	13302	16	50	21	...

5 rows × 24 columns

In [36]:
```
df.drop('Airline', axis = 1, inplace = True)
df.head()
```

Out[36]:

	Source	Destination	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arrival_Time
0	Banglore	New Delhi	BLR → DEL	170	0	3897	22	20	1	
1	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	7662	5	50	13	
2	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	13882	9	25	4	
3	Kolkata	Banglore	CCU → NAG → BLR	325	1	6218	18	5	23	
4	Banglore	New Delhi	BLR → NAG → DEL	285	1	13302	16	50	21	

5 rows × 23 columns

SOURCE AND DESTINATION

In [37]:
```
list1 = ['Source', 'Destination']
for l in list1:
    print(df[[l]].value_counts(), '\n')
```

```
Source
Delhi       4536
Kolkata     2871
Banglore    2197
Mumbai       697
Chennai      381
dtype: int64

Destination
Cochin          4536
Banglore        2871
Delhi           1265
New Delhi        932
Hyderabad        697
Kolkata          381
dtype: int64
```

In [38]:
```
df = pd.get_dummies(data = df, columns = list1, drop_first = True)
df.head()
```

Out[38]:

	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arrival_Time_in_minutes	Day	Mor
0	BLR → DEL	170	0	3897	22	20	1	10	24	
1	CCU → IXR → BBI → BLR	445	2	7662	5	50	13	15	5	
2	DEL → LKO → BOM → COK	1140	2	13882	9	25	4	25	6	
3	CCU → NAG → BLR	325	1	6218	18	5	23	30	5	
4	BLR → NAG → DEL	285	1	13302	16	50	21	35	3	

5 rows × 30 columns

ROUTE

In [39]:
```
route = df[['Route']]
route.head()
```

Out[39]:

	Route
0	BLR → DEL
1	CCU → IXR → BBI → BLR
2	DEL → LKO → BOM → COK
3	CCU → NAG → BLR
4	BLR → NAG → DEL

In [40]:
```
df['Total_Stops'].value_counts()
```

Out[40]:
```
1    5625
0    3491
2    1520
3      45
4       1
Name: Total_Stops, dtype: int64
```

In [41]:
```
route['Route_1'] = route['Route'].str.split('→').str[0]
route['Route_2'] = route['Route'].str.split('→').str[1]
route['Route_3'] = route['Route'].str.split('→').str[2]
route['Route_4'] = route['Route'].str.split('→').str[3]
route['Route_5'] = route['Route'].str.split('→').str[4]
```

```
route.head()
```

```
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2109748827.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  route['Route_1'] = route['Route'].str.split('→').str[0]
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2109748827.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  route['Route_2'] = route['Route'].str.split('→').str[1]
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2109748827.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  route['Route_3'] = route['Route'].str.split('→').str[2]
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2109748827.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  route['Route_4'] = route['Route'].str.split('→').str[3]
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2109748827.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  route['Route_5'] = route['Route'].str.split('→').str[4]
```

Out[41]:

	Route	Route_1	Route_2	Route_3	Route_4	Route_5
0	BLR → DEL	BLR	DEL	NaN	NaN	NaN
1	CCU → IXR → BBI → BLR	CCU	IXR	BBI	BLR	NaN
2	DEL → LKO → BOM → COK	DEL	LKO	BOM	COK	NaN
3	CCU → NAG → BLR	CCU	NAG	BLR	NaN	NaN
4	BLR → NAG → DEL	BLR	NAG	DEL	NaN	NaN

In [42]:

```
route.fillna('None', inplace = True)
route.head()
```

```
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2171952904.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
  route.fillna('None', inplace = True)
```

Out[42]:

	Route	Route_1	Route_2	Route_3	Route_4	Route_5
0	BLR → DEL	BLR	DEL	None	None	None
1	CCU → IXR → BBI → BLR	CCU	IXR	BBI	BLR	None
2	DEL → LKO → BOM → COK	DEL	LKO	BOM	COK	None
3	CCU → NAG → BLR	CCU	NAG	BLR	None	None
4	BLR → NAG → DEL	BLR	NAG	DEL	None	None

In [43]:

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
for i in range(1, 6):
    col = 'Route_' + str(i)
    route[col] = le.fit_transform(route[col])

route.head()
```

```
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1382904615.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  route[col] = le.fit_transform(route[col])
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1382904615.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  route[col] = le.fit_transform(route[col])
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1382904615.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  route[col] = le.fit_transform(route[col])
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1382904615.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  route[col] = le.fit_transform(route[col])
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\1382904615.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#ret
urning-a-view-versus-a-copy
  route[col] = le.fit_transform(route[col])
```

Out[43]:

	Route	Route_1	Route_2	Route_3	Route_4	Route_5
0	BLR → DEL	0	13	29	13	5
1	CCU → IXR → BBI → BLR	2	25	1	3	5
2	DEL → LKO → BOM → COK	3	32	4	5	5
3	CCU → NAG → BLR	2	34	3	13	5
4	BLR → NAG → DEL	0	34	8	13	5

In [44]:

```python
route.drop('Route', axis = 1, inplace = True)
route.head()
```

```
C:\Users\cws\AppData\Local\Temp\ipykernel_3736\2499507917.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#ret
urning-a-view-versus-a-copy
  route.drop('Route', axis = 1, inplace = True)
```

Out[44]:

	Route_1	Route_2	Route_3	Route_4	Route_5
0	0	13	29	13	5
1	2	25	1	3	5
2	3	32	4	5	5
3	2	34	3	13	5
4	0	34	8	13	5

In [45]:

```python
df = pd.concat([df, route], axis = 1)
df.head()
```

Out[45]:

	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arrival_Time_in_minutes	Day	Mor
0	BLR → DEL	170	0	3897	22	20	1	10	24	
1	CCU → IXR → BBI → BLR	445	2	7662	5	50	13	15	5	
2	DEL → LKO → BOM → COK	1140	2	13882	9	25	4	25	6	
3	CCU → NAG → BLR	325	1	6218	18	5	23	30	5	
4	BLR → NAG → DEL	285	1	13302	16	50	21	35	3	

5 rows × 35 columns

In [46]:
```
df.drop('Route', axis = 1, inplace = True)
df.head()
```

Out[46]:

	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arrival_Time_in_minutes	Day	Month	A Indi
0	170	0	3897	22	20	1	10	24	3	
1	445	2	7662	5	50	13	15	5	1	
2	1140	2	13882	9	25	4	25	6	9	
3	325	1	6218	18	5	23	30	5	12	
4	285	1	13302	16	50	21	35	3	1	

5 rows × 34 columns

Building the Machine Learning Model(s) & Evaluating them

In [47]:
```
temp_col = df.columns.to_list()
print(temp_col, '\n')

new_col = temp_col[: 2] + temp_col[3:]
new_col.append(temp_col[2])
print(new_col, '\n')

df = df.reindex(columns = new_col)
df.head()
```

```
['Duration', 'Total_Stops', 'Price', 'Dep_Time_in_hours', 'Dep_Time_in_minutes', 'Arrival_Time_in_hours', 'Arri
val_Time_in_minutes', 'Day', 'Month', 'Air India', 'GoAir', 'IndiGo', 'Jet Airways', 'Jet Airways Business', 'M
ultiple carriers', 'Multiple carriers Premium economy', 'SpiceJet', 'Trujet', 'Vistara', 'Vistara Premium econo
my', 'Source_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_Mumbai', 'Destination_Cochin', 'Destination_De
lhi', 'Destination_Hyderabad', 'Destination_Kolkata', 'Destination_New Delhi', 'Route_1', 'Route_2', 'Route_3',
'Route_4', 'Route_5']

['Duration', 'Total_Stops', 'Dep_Time_in_hours', 'Dep_Time_in_minutes', 'Arrival_Time_in_hours', 'Arrival_Time_
in_minutes', 'Day', 'Month', 'Air India', 'GoAir', 'IndiGo', 'Jet Airways', 'Jet Airways Business', 'Multiple c
arriers', 'Multiple carriers Premium economy', 'SpiceJet', 'Trujet', 'Vistara', 'Vistara Premium economy', 'Sou
rce_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_Mumbai', 'Destination_Cochin', 'Destination_Delhi', 'De
stination_Hyderabad', 'Destination_Kolkata', 'Destination_New Delhi', 'Route_1', 'Route_2', 'Route_3', 'Route_4
', 'Route_5', 'Price']
```

Out[47]:

	Duration	Total_Stops	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Arrival_Time_in_minutes	Day	Month	Air India	GoA
0	170	0	22	20	1	10	24	3	0	
1	445	2	5	50	13	15	5	1	1	
2	1140	2	9	25	4	25	6	9	0	
3	325	1	18	5	23	30	5	12	0	
4	285	1	16	50	21	35	3	1	0	

5 rows × 34 columns

In [48]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df = scaler.fit_transform(df)

df[0]
```

Out[48]:
```
array([-0.93160111, -1.22066609,  1.65415376, -0.2349499 , -1.80043628,
       -0.8900139 ,  1.28553644, -0.84844966, -0.44278513, -0.13600489,
        2.05015058, -0.75053033, -0.02370671, -0.35507822, -0.03490678,
       -0.28797191, -0.00967596, -0.21667251, -0.01676082, -0.19231927,
       -0.85909313, -0.60626609, -0.2642058 , -0.85909313, -0.36651266,
       -0.2642058 , -0.19231927,  3.23440464, -1.67418972,  0.13765097,
        1.39512392,  0.40974412,  0.06420744, -1.12553455])
```

In [49]:
```
from sklearn.model_selection import train_test_split as tts

x = df[:, : -1]
y = df[:, -1]
```

In [50]:
```
x_train, x_test, y_train, y_test = tts(x, y, test_size = 0.1, random_state = 69)

print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(9613, 33)
(1069, 33)
(9613,)
(1069,)
```

Linear Regression

In [51]:
```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(x_train, y_train)
```

Out[51]:
```
▾ LinearRegression
LinearRegression()
```

In [52]:
```
from sklearn.metrics import mean_squared_error, r2_score

def metrics(y_true, y_pred):
    print(f'RMSE:', mean_squared_error(y_true, y_pred) ** 0.5)
    print(f'R_Squared value:', r2_score(y_true, y_pred))

def accuracy(y_true, y_pred):
    errors = abs(y_true - y_pred)
    mape = 100 * np.mean(errors/y_true)
    accuracy = 100 - mape
    return accuracy
```

In [53]:
```
y_pred = model.predict(x_test)
```

In [54]:
```
metrics(y_test, y_pred)
```

```
RMSE: 0.5363712927002792
R_Squared value: 0.6458004370526309
```

In [55]:
```
accuracy(y_test, y_pred)
```

Out[55]:
```
70.24984132253009
```

Random Forest

In [56]:
```
from sklearn.ensemble import RandomForestRegressor

model_random_forest = RandomForestRegressor(n_estimators = 500, min_samples_split = 3)
model_random_forest.fit(x_train, y_train)
```

Out[56]:
```
RandomForestRegressor
RandomForestRegressor(min_samples_split=3, n_estimators=500)
```

In [57]:
```
pred_rf = model_random_forest.predict(x_test)
```

In [58]:
```
metrics(y_test, pred_rf)
```

```
RMSE: 0.39249595924460284
R_Squared value: 0.8103349327024327
```

In [59]:
```
accuracy(y_test, pred_rf)
```

Out[59]:
```
99.0544819939845
```