In [1]:

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

```
C:\Users\himan\Anaconada3\lib\site-packages\scipy\__init__.py:146: UserWarning: A NumPy versi
on >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.24.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:

```
data = pd.read_excel(r'C:\Users\himan\Downloads\archive (25)\Data_Train.xlsx')
data
```

Out[2]:

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

10683 rows × 11 columns

In [3]:
```
data.head()
```

Out[3]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additiona
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h 50m	non-stop	N
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	N
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	19h	2 stops	N
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	N
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	N

In [4]:
```
data.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 [5]:
```
data.describe()
```

Out[5]:

	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 [6]:
```
data.shape
```

Out[6]:
```
(10683, 11)
```

In [7]:
```
data.count()
```

Out[7]:
```
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 [8]:
```
data.dtypes
```

Out[8]:
```
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 [9]:
```
data.isnull().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]:
```
#fillter data raws missing doing conditional value, missing values fatching
data[data['Route'].isna() | data['Total_Stops'].isna()]
```

Out[10]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Addition
9039	Air India	6/05/2019	Delhi	Cochin	NaN	09:45	09:25 07 May	23h 40m	NaN	

In [11]:
```
data.dropna(inplace = True)
```

In [12]:
```
data.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
```

EDA & Feature Engineering

1.Duration

2.Departure and Arrival time

3.Data of journey

4.Total Stops

5.Additional info

6.Airline

7.Source and destination

8.Route

In [13]:
```
#Duration
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 [14]:
```
data['Duration'] = data['Duration'].apply(convert_duration)
data.head()
```

Out[14]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additiona
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	170	non-stop	N
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	445	2 stops	N
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	1140	2 stops	N
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	325	1 stop	N
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	285	1 stop	N

In [15]:
```
#Departure and Arrival Time
data['Dep_Time'] = pd.to_datetime(data['Dep_Time'])
data['Arrival_Time'] = pd.to_datetime(data['Dep_Time'])
data.dtypes
```

Out[15]:
```
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 [16]: data['Dep_Time_in_hours'] = data['Dep_Time'].dt.hour
         data['Dep_Time_in_minutes'] = data['Dep_Time'].dt.minute
         data['Arrival_Time_in_hours'] = data['Arrival_Time'].dt.hour
         data['Arrival_Time_in_hours'] = data['Arrival_Time'].dt.minute

         data.head()
```

Out[16]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additiona
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	2024-06-24 22:20:00	2024-06-24 22:20:00	170	non-stop	N
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	2024-06-24 05:50:00	2024-06-24 05:50:00	445	2 stops	N
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	2024-06-24 09:25:00	2024-06-24 09:25:00	1140	2 stops	N
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	2024-06-24 18:05:00	2024-06-24 18:05:00	325	1 stop	N
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	2024-06-24 16:50:00	2024-06-24 16:50:00	285	1 stop	N

```
In [17]: data.drop(['Dep_Time', 'Arrival_Time'], axis = 1, inplace = True)
         data.head()
```

Out[17]:

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

```
In [18]: #Date Of Journey
         data['Date_of_Journey'] = pd.to_datetime(data['Date_of_Journey'])
         data.head()
```

```
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '24/03/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '24/06/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '27/05/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '18/04/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '24/04/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '15/04/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '21/03/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '15/05/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '18/06/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '15/06/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '18/05/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '27/06/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '21/05/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '15/03/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '24/05/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '21/04/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '21/06/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '27/03/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
```

```
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '18/03/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
C:\Users\himan\Anaconada3\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarning:
Parsing '27/04/2019' in DD/MM/YYYY format. Provide format or specify infer_datetime_format=Tr
ue for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
```

Out[18]:

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

In [19]:
```
data['Date_of_Journey'].dt.year.unique()
```

Out[19]:
```
array([2019], dtype=int64)
```

In [20]:
```python
#create new cloumn of days and months
data['Day'] = data['Date_of_Journey'].dt.day
data['Month'] = data['Date_of_Journey'].dt.month

data.head()
```

Out[20]:

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

In [21]:
```python
data.drop('Date_of_Journey', axis = 1, inplace = True)
data.head()
```

Out[21]:

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

In [22]:
```
#Total Stops
data['Total_Stops'].value_counts()
```
Out[22]:
```
1 stop      5625
non-stop    3491
2 stops     1520
3 stops       45
4 stops        1
Name: Total_Stops, dtype: int64
```
In [23]:
```
data['Total_Stops'] = data['Total_Stops'].map({
    'non-stop': 0,
    '1 stop': 1,
    '2 stops': 2,
    '3 stops': 3,
    '4 stops': 4
})
```
In [24]:
```
data.head()
```
Out[24]:

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

In [25]:
```
#Additional info
data['Additional_Info'].value_counts()
```
Out[25]:
```
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 [26]: data.drop('Additional_Info', axis=1, inplace = True)
```

```
In [27]: data.head()
```

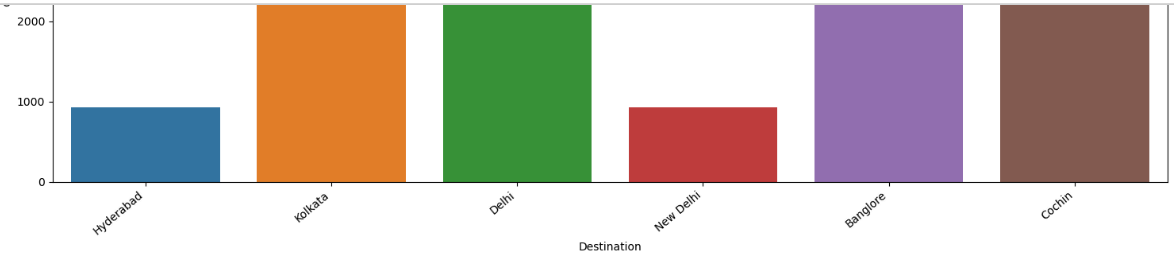
Out[27]:

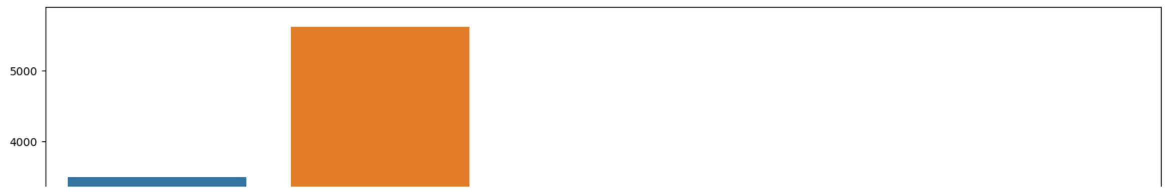
	Airline	Source	Destination	Route	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	A
0	IndiGo	Banglore	New Delhi	BLR → DEL	170	0	3897	22	20	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	7662	5	50	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	13882	9	25	
3	IndiGo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

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

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

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

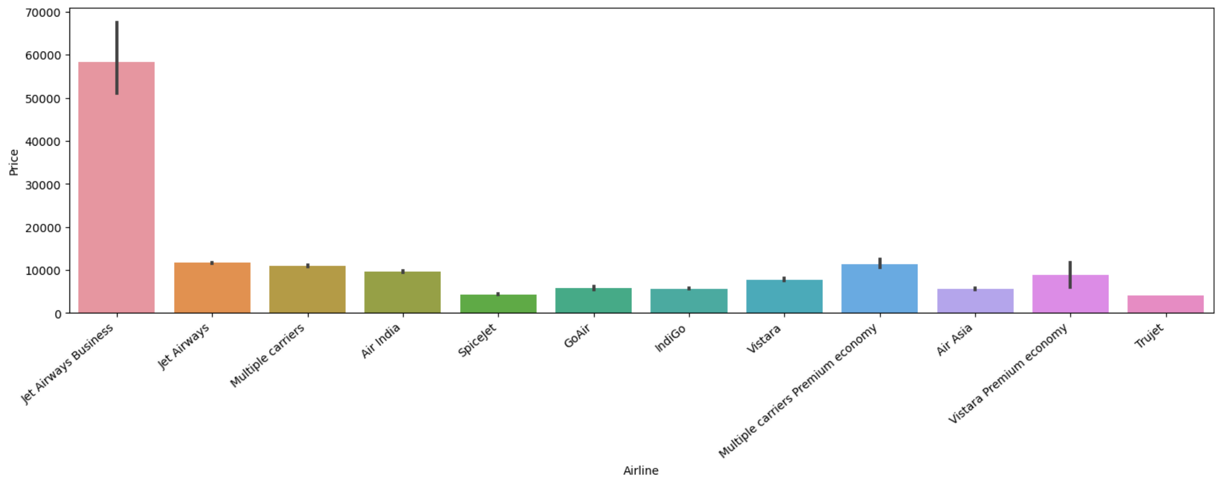




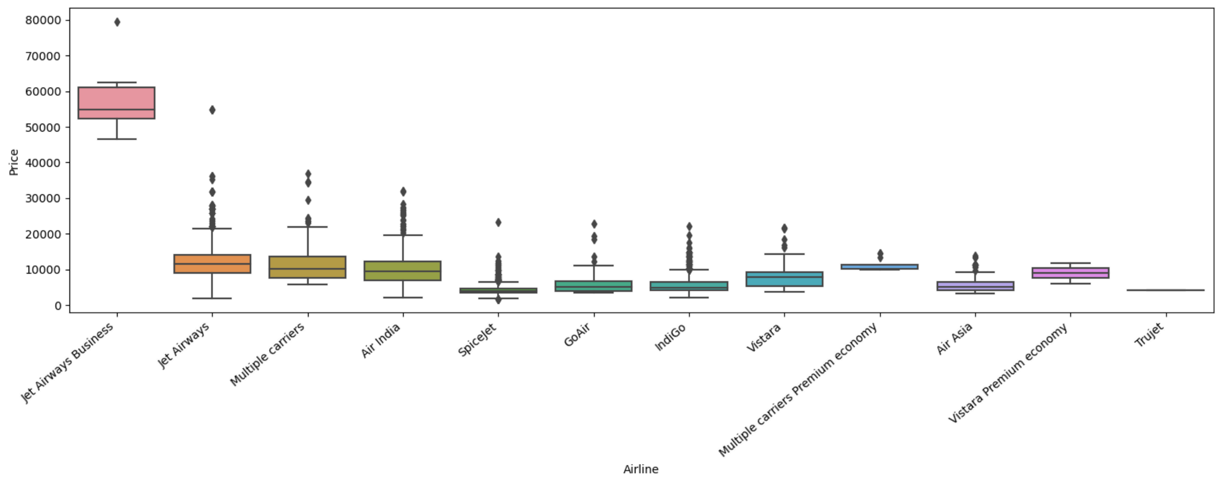
In [30]:
```
#Airline
data['Airline'].value_counts()
```

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 = data.sort_values('Price', ascending = Fals
ax.set_xticklabels(ax.get_xticklabels(), rotation = 40, ha = 'right') #used for rotaion value
plt.tight_layout()
plt.show()
print('\n\n')
```



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



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

Out[33]:

	count	mean	std	min	25%	50%	75%	max
Airline								
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]:
```python
Airline = pd.get_dummies(data['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]:
```
data = pd.concat([data, Airline], axis = 1)
data.head()
```

Out[35]:

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

5 rows × 23 columns

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

Out[36]:

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

5 rows × 22 columns

In [37]:
```
#Soure and Destination
list1 = ['Source', 'Destination']
for l in list1:
    print(data[[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]:
```python
data = pd.get_dummies(data = data, columns = list1, drop_first = True)
data.head()
```

Out[38]:

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

5 rows × 29 columns

In [39]:
```python
#Route
route = data[['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]:
```python
data['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\himan\AppData\Local\Temp\ipykernel_22976\401271215.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_guid
e/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)
  route['Route_1'] = route['Route'].str.split('->').str[0]
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\401271215.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_guid
e/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)
  route['Route_2'] = route['Route'].str.split('->').str[1]
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\401271215.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_guid
e/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)
  route['Route_3'] = route['Route'].str.split('->').str[2]
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\401271215.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_guid
e/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)
  route['Route_4'] = route['Route'].str.split('->').str[3]
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\401271215.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_guid
e/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)
  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	NaN
1	CCU → IXR → BBI → BLR	CCU → IXR → BBI → BLR	NaN	NaN	NaN	NaN
2	DEL → LKO → BOM → COK	DEL → LKO → BOM → COK	NaN	NaN	NaN	NaN
3	CCU → NAG → BLR	CCU → NAG → BLR	NaN	NaN	NaN	NaN
4	BLR → NAG → DEL	BLR → NAG → DEL	NaN	NaN	NaN	NaN

In [42]:
```python
route.fillna('None', inplace = True)
route.head()
```

```
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\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_guid
e/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)
  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	None
1	CCU → IXR → BBI → BLR	CCU → IXR → BBI → BLR	None	None	None	None
2	DEL → LKO → BOM → COK	DEL → LKO → BOM → COK	None	None	None	None
3	CCU → NAG → BLR	CCU → NAG → BLR	None	None	None	None
4	BLR → NAG → DEL	BLR → NAG → DEL	None	None	None	None

In [43]:
```python
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\himan\AppData\Local\Temp\ipykernel_22976\959327169.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_guid
e/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)
  route[col] = le.fit_transform(route[col])
```

Out[43]:

	Route	Route_1	Route_2	Route_3	Route_4	Route_5
0	BLR → DEL	18	0	0	0	0
1	CCU → IXR → BBI → BLR	84	0	0	0	0
2	DEL → LKO → BOM → COK	118	0	0	0	0
3	CCU → NAG → BLR	91	0	0	0	0
4	BLR → NAG → DEL	29	0	0	0	0

In [44]:
```
route.drop('Route', axis = 1, inplace = True)
route.head()
```

```
C:\Users\himan\AppData\Local\Temp\ipykernel_22976\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_guid
e/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)
  route.drop('Route', axis = 1, inplace = True)
```

Out[44]:

	Route_1	Route_2	Route_3	Route_4	Route_5
0	18	0	0	0	0
1	84	0	0	0	0
2	118	0	0	0	0
3	91	0	0	0	0
4	29	0	0	0	0

In [45]:
```
data = pd.concat([data, route], axis=1)
data.head()
```

Out[45]:

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

5 rows × 34 columns

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

Out[46]:

	Duration	Total_Stops	Price	Dep_Time_in_hours	Dep_Time_in_minutes	Arrival_Time_in_hours	Day	Month	Ai Indi
0	170	0	3897	22	20	20	24	3	
1	445	2	7662	5	50	50	5	1	
2	1140	2	13882	9	25	25	6	9	
3	325	1	6218	18	5	5	5	12	
4	285	1	13302	16	50	50	3	1	

5 rows × 33 columns

Building the Machine Learning Model(s) & Evaluating them

In [47]:
```
temp_col = data.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')

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

```
['Duration', 'Total_Stops', 'Price', 'Dep_Time_in_hours', 'Dep_Time_in_minutes', 'Arrival_Tim
e_in_hours', 'Day', 'Month', 'Air India', 'GoAir', 'IndiGo', 'Jet Airways', 'Jet Airways Busi
ness', 'Multiple carriers', 'Multiple carriers Premium economy', 'SpiceJet', 'Trujet', 'Vista
ra', 'Vistara Premium economy', 'Source_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_M
umbai', 'Destination_Cochin', 'Destination_Delhi', 'Destination_Hyderabad', 'Destination_Kolk
ata', '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_hour
s', 'Day', 'Month', 'Air India', 'GoAir', 'IndiGo', 'Jet Airways', 'Jet Airways Business', 'M
ultiple carriers', 'Multiple carriers Premium economy', 'SpiceJet', 'Trujet', 'Vistara', 'Vis
tara Premium economy', 'Source_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_Mumbai',
'Destination_Cochin', 'Destination_Delhi', 'Destination_Hyderabad', 'Destination_Kolkata', 'D
estination_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	Day	Month	Air India	GoAi
0	170	0	22	20	20	24	3	0	
1	445	2	5	50	50	5	1	1	
2	1140	2	9	25	25	6	9	0	
3	325	1	18	5	5	5	12	0	
4	285	1	16	50	50	3	1	0	

5 rows × 33 columns

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

scaler = StandardScaler()
data = scaler.fit_transform(data)

data[0]
```

Out[48]:
```
array([-0.93160111, -1.22066609,  1.65415376, -0.2349499 , -0.2349499 ,
        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, -1.5470817 ,  0.        ,  0.        ,
        0.        ,  0.        , -1.12553455])
```

In [49]:
```
#Split entire dataset
from sklearn.model_selection import train_test_split as tts

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

In [50]:
```
#split properly
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, 32)
(1069, 32)
(9613,)
(1069,)
```

Linear Regression

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

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

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

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

def metrics(y_true, y_pred):
    print(f'RMS:', 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)
```

```
RMS: 0.5383497886215094
R_Squared value: 0.6431825681355884
```

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

```
Out[55]: 59.8179827678726
```

Ramdom 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(min_samples_split=3, n_estimators=500)
```

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

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

```
RMS: 0.4041627475047418
R_Squared value: 0.7988919144073189
```

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

```
Out[59]: 86.17028522338039
```

```
In [ ]:
```