FLIP ROBO

Flight Price Prediction

Submitted by:

Manoj.I.V

ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped you and guided you in completion of the project.

INTRODUCTION

- ## Business Problem Framing

  Describe the business problem and how this problem can be related to the real world.

  Answer: The travel from one country to another has become common these days. These algorithms are required so that it would be every cost effective if a price of the flight can be predicted.

- ## Conceptual Background of the Domain Problem

  Describe the domain related concepts that you think will be useful for better understanding of the project.

  Answer: Type of Airlines, time of flight, destination, source, price of the fuel, number of stops.

- ## Review of Literature

  This is a comprehensive summary of the research done on the topic. The review should enumerate, describe, summarize, evaluate and clarify the research done.

  Answer: Data was scraped at different websites like make my trip, yatra.com, skyscanner.com, official websites of airlines. So the different costs of the airlines were taken into consideration.

- ## Motivation for the Problem Undertaken

  Describe your objective behind to make this project, this domain and what the motivation is behind.

  Answer: The objective is to build economic model which can predict cost of the flight. This model is to save money during travel for far places.

Analytical Problem Framing

- ## Mathematical/ Analytical Modeling of the Problem

Describe the mathematical, statistical and analytics modelling done during this project along with the proper justification.

Answer: Different models used in the project are linear regression, logistic regression, Random Forest Regressor, XGBRegressor, AdaBoostRegressor, KNeighborsRegressor, SVR, Gradient Boosting Regressor.

- ## Data Sources and their formats

What are the data sources, their origins, their formats and other details that you find necessary? They can be described here. Provide a proper data description. You can also add a snapshot of the data.

Answer: Data was scraped at different websites like make my trip, yatra.com, skyscanner.com, official websites of airlines.

In [120]: f

Out[120]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	66h 00m	58,201	DEL to LON to LIS
1	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
2	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
3	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,414	DEL to LON to LIS
4	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
5	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
6	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
7	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,794	DEL to LON to LIS
8	Air India	18 Sept 22	New Delhi	Toronto	1 Stop	30h 14m	65,079	DEL to LIS
9	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	65,188	DEL to LON to LIS

In [3]:
```
a
```
Out[3]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	66h 00m	58,201	DEL to LON to LIS
1	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
2	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
3	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,414	DEL to LON to LIS
4	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
5	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
6	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
7	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,794	DEL to LON to LIS
8	Air India	18 Sept 22	New Delhi	Toronto	1 Stop	30h 14m	65,079	DEL to LIS
9	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	65,188	DEL to LON to LIS

In [4]:
```
z=pd.read_csv('Set2')
```
In [5]:
```
z
```
Out[5]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Srilankan Airlines	29 Sept 22	New Delhi	Paris	1 stop	33 h 30 m	27,264	DEL to RML to CDG
1	Gulf Air	29 Sept 22	New Delhi	Paris	1 stop	31 h 30 m	31,492	DEL to BAH to CDG
2	Vistara, Air France	29 Sept 22	New Delhi	Paris	1 stop	15 h	32,065	DEL to BOM to CDG
3	Vistara, Air France	29 Sept 22	New Delhi	Paris	1 stop	15 h 55 m	32,065	DEL to BOM to CDG
4	Vistara, Air France	29 Sept 22	New Delhi	Paris	1 stop	16 h 40 m	32,065	DEL to BOM to CDG
5	Vistara, Air France	29 Sept 22	New Delhi	Paris	1 stop	17 h 55 m	32,065	DEL to BOM to CDG
6	Vistara, Air France	29 Sept 22	New Delhi	Paris	1 stop	18 h 45 m	32,065	DEL to BOM to CDG

In [6]:
```
n=pd.read_csv('Set3')
```
In [7]:
```
n
```
Out[7]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Vistara, United Airlines	29 Oct 22	Bengaluru	Toronto	2 stop	26 h 37 m	72,375	BLR to DEL to EWR to LIS
1	United Airlines	29 Oct 22	Bengaluru	Toronto	2 stop	25 h 19 m	78,959	BLR to DEL to ORD to LIS
2	Lufthansa	29 Oct 22	Bengaluru	Toronto	1 stop	25 h 50 m	82,822	BLR to FRA to LIS
3	Qatar Airways, Air Canada	29 Oct 22	Bengaluru	Toronto	2 stop	26 h 27 m	84,373	BLR to DIA to NYC to LIS
4	Qatar Airways, American Airlines	29 Oct 22	Bengaluru	Toronto	2 stop	28 h 14 m	84,373	BLR to DIA to PHL to LIS
...	...	...	...	...	...	...	...	...
131	Delta Air Lines	29 Oct 22	Bengaluru	Toronto	2 stop	49 h 15 m	3,02,418	BLR to CDG to NYC to LIS
132	Delta Air Lines	29 Oct 22	Bengaluru	Toronto	2 stop	49 h 15 m	3,02,418	BLR to CDG to NYC to LIS
133	Lufthansa	29 Oct 22	Bengaluru	Toronto	2 stop	27 h 55 m	3,07,323	BLR to FRA to YVR to LIS
134	Etihad Airways, Finnair	29 Oct 22	Bengaluru	Toronto	2 stop	24 h 45 m	3,96,695	BLR to AUH to LHR to LIS
135	Etihad Airways, Finnair	29 Oct 22	Bengaluru	Toronto	2 stop	24 h 45 m	3,96,695	BLR to AUH to LHR to LIS

136 rows × 8 columns

In [8]:
```
w=pd.read_csv('Set4')
```
In [9]:
```
w
```
Out[9]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Vistara, Air France	27 Nov 22	New Delhi	Toronto	2 stop	30 h 35 m	66,380	DEL to BLR to CDG to LIS
1	Vistara, Air France	27 Nov 22	New Delhi	Toronto	2 stop	31 h 25 m	66,380	DEL to BLR to CDG to LIS
2	Vistara, Air France	27 Nov 22	New Delhi	Toronto	2 stop	33 h 40 m	66,380	DEL to BLR to CDG to LIS

```
In [10]: s=pd.read_csv('Set5')
```

```
In [11]: s
```

Out[11]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Go First	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,366	BLR to DEL
1	Go First	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,368	BLR to DEL
2	Akasa Air	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,371	BLR to DEL
3	SpiceJet	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,680	BLR to DEL
4	AirAsia	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,682	BLR to DEL
5	AirAsia	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,682	BLR to DEL
6	AirAsia	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,682	BLR to DEL
7	AirAsia	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,682	BLR to DEL
8	AirAsia	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,682	BLR to DEL
9	IndiGo	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
10	IndiGo	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
11	Air India	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
12	Vistara	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 40 m	7,684	BLR to DEL
13	IndiGo	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
14	Vistara	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 40 m	7,684	BLR to DEL
15	IndiGo	30 Nov 22	Bengaluru	New Delhi	Non st	03 h	7,684	BLR to DEL
16	Vistara	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,684	BLR to DEL
17	IndiGo	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,684	BLR to DEL
18	Air India	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
19	Air India	30 Nov 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL

```
In [12]: u=pd.read_csv('Set6')
```

```
In [13]: u
```

Out[13]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Go First	21 sep 22	Bengaluru	New Delhi	Non st	02 h 40 m	7,682	BLR to DEL
1	SpiceJet	21 sep 22	Bengaluru	New Delhi	Non st	02 h 35 m	7,682	BLR to DEL
2	Go First	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,682	BLR to DEL
3	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
4	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
5	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
6	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
7	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
8	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,684	BLR to DEL
9	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
10	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
11	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
12	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	7,684	BLR to DEL
13	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	7,684	BLR to DEL
14	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,684	BLR to DEL
15	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,684	BLR to DEL
16	AirAsia	21 sep 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,685	BLR to DEL
17	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 55 m	7,802	BLR to DEL
18	IndiGo	21 sep 22	Bengaluru	New Delhi	Non st	02 h 45 m	8,157	BLR to DEL
19	Air India	21 sep 22	Bengaluru	New Delhi	Non st	02 h 50 m	8,209	BLR to DEL

```
In [14]: o=pd.read_csv('Set7')
```

```
In [15]: o
```

Out[15]:

	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	Qatar Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	13 h 30 m	38,192	BLR to DIA to FCO
1	Etihad Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	28 h 40 m	38,798	BLR to AUH to FCO
2	Qatar Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	13 h 05 m	42,386	BLR to DIA to FCO
3	Qatar Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	30 h 30 m	42,674	BLR to DIA to FCO
4	Gulf Air	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	11 h 45 m	44,773	BLR to BAH to FCO
...	...	...	...	...	...	...	...	...
123	Air France, Kenya Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	16 h 05 m	2,55,714	BLR to CDG to FCO
124	Vistara, Etihad Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	23 h 50 m	2,56,401	BLR to BOM to AUH to FCO
125	Air France, Alitalia	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	17 h 20 m	2,67,712	BLR to CDG to LIN to FCO
126	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	3,11,355	BLR to BOM to SIN to FCO
127	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	3,11,355	BLR to BOM to SIN to FCO

128 rows × 8 columns

```
In [16]: dt=pd.concat([a,z,n,w,s,u,o],ignore_index=True)
```

```
In [17]: dt
```

Out[17]:

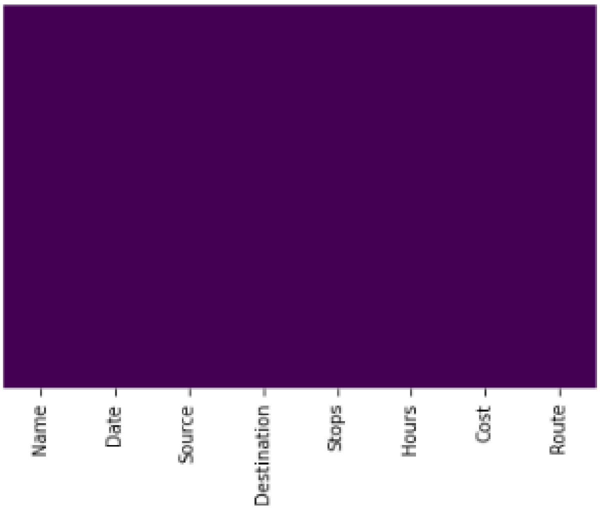
	Name	Date	Source	Destination	Stops	Hours	Cost	Route
0	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	66h 00m	58,201	DEL to LON to LIS
1	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
2	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS
3	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,414	DEL to LON to LIS
4	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS
...	...	...	...	...	...	...	...	...
510	Air France, Kenya Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	16 h 05 m	2,55,714	BLR to CDG to FCO
511	Vistara, Etihad Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	23 h 50 m	2,56,401	BLR to BOM to AUH to FCO
512	Air France, Alitalia	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	17 h 20 m	2,67,712	BLR to CDG to LIN to FCO
513	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	3,11,355	BLR to BOM to SIN to FCO
514	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	3,11,355	BLR to BOM to SIN to FCO

515 rows × 8 columns

```
In [21]: # To find the data type of the dataset
for col in dt:
    print ('This column', col ,'has', dt[col].unique(),'unique elements')
    print ('*'*100)
```

```
This column Name has ['TAP Air Portugal' 'Air India' 'Srilankan Airlines' 'Gulf Air'
 'Vistara, Air France' 'Etihad Airways' 'Air France' 'Qatar Airways'
 'Emirates' 'Vistara, United Airlines' 'United Airlines' 'Lufthansa'
 'Qatar Airways, Air Canada' 'Qatar Airways, American Airlines'
 'Emirates, WestJet' 'Delta Air Lines' 'British Airways, Air Canada'
 'Qatar Airways, Porter Airlines' 'Qatar Airways, WestJet' 'Air Canada'
 'British Airways' 'Delta Air Lines, WestJet' 'American Airlines'
 'Air France, Air Canada' 'Vistara, KLM Royal Dutch' 'Vistara, Air Canada'
 'Vistara, United Airlines, Air Canada' 'Ethiopian Airlines' 'Swiss'
 'Vistara, Lufthansa' 'Lufthansa, Air Canada' 'British Airways, Lufthansa'
 'Etihad Airways, Finnair' 'Japan Airlines'
 'Japan Airlines, American Airlines' 'KLM Royal Dutch'
 'KLM Royal Dutch, Air France, Delta Air Lines'
 'United Airlines, Air Canada' 'Emirates, Air Canada' 'Korean Air'
 'British Airways, Finnair' 'All Nippon Airways' 'Go First' 'Akasa Air'
 'SpiceJet' 'AirAsia' 'IndiGo' 'Vistara' 'Turkish Airlines'
 'Singapore Airlines' 'Vistara, Qatar Airways' 'Vistara, Emirates'
 'Lufthansa, Alitalia' 'Lufthansa, Air Dolomiti, Alitalia'
 'KLM Royal Dutch, Alitalia' 'Air France, Alitalia' 'Air India, Lufthansa'
 'Air India, Air France' 'Air France, KLM Royal Dutch'
 'Lufthansa, Royal Air Maroc' 'Air France, Kenya Airways'
 'Vistara, Etihad Airways' 'Vistara, Singapore Airlines'] unique elements
****************************************************************************************************
This column Date has ['18 Sept 22' '29 Sept 22' '29 Oct 22' '27 Nov 22' '30 Nov 22' '21 sep 22'
 '28 Oct 22'] unique elements
****************************************************************************************************
This column Source has ['New Delhi' 'Bengaluru'] unique elements
****************************************************************************************************
This column Destination has ['Toronto' 'Paris' 'New Delhi' 'Rome - Fiumicino Apt'] unique elements
****************************************************************************************************
This column Stops has ['2 Stops' '1 Stop' '1 stop' 'Non st' '2 stop'] unique elements
```

```
In [22]: sns.heatmap(dt.isnull(), yticklabels = False, cbar = False, cmap ='viridis')
Out[22]: <AxesSubplot:>
```




```
In [23]: # Getting information on the dataset
         dt.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 515 entries, 0 to 514
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Name         515 non-null    object
 1   Date         515 non-null    object
 2   Source       515 non-null    object
 3   Destination  515 non-null    object
 4   Stops        515 non-null    object
 5   Hours        515 non-null    object
```

- Data Preprocessing Done

What were the steps followed for the cleaning of the data? What were the assumptions done and what were the next actions steps over that?

Answer: Converting data formats from object to float etc. Removing junk data.

```
In [26]: dt['Cost']=dt['Cost'].replace('₹', '')

In [27]: dt['Rate']=dt['Cost'].replace('₹','')

In [28]: dt
Out[28]:
```

	Name	Date	Source	Destination	Stops	Hours	Cost	Route	Rate
0	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	66h 00m	58,201	DEL to LON to LIS	58,201
1	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS	61,020
2	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,020	DEL to LON to LIS	61,020
3	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	61,414	DEL to LON to LIS	61,414
4	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	64,768	DEL to LON to LIS	64,768
...	...	...	...	...	...	...	...	...	...
510	Air France, Kenya Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	16 h 05 m	2,55,714	BLR to CDG to FCO	2,55,714
511	Vistara, Etihad Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	23 h 50 m	2,56,401	BLR to BOM to AUH to FCO	2,56,401
512	Air France, Alitalia	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	17 h 20 m	2,67,712	BLR to CDG to LIN to FCO	2,67,712

```
In [31]: dt['Price']=dt['Rate'].str.replace('₹', ' ')

In [32]: dt.drop('Rate',axis=1,inplace=True)

In [33]: dt
```

Out[33]:

	Name	Date	Source	Destination	Stops	Hours	Route	Price
0	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	66h 00m	DEL to LON to LIS	58,201
1	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	DEL to LON to LIS	61,020
2	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	DEL to LON to LIS	61,020
3	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	49h 05m	DEL to LON to LIS	61,414
4	TAP Air Portugal	18 Sept 22	New Delhi	Toronto	2 Stops	42h 00m	DEL to LON to LIS	64,768
...	...	...	...	...	...	...	...	...
510	Air France, Kenya Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	1 stop	16 h 05 m	BLR to CDG to FCO	2,55,714
511	Vistara, Etihad Airways	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	23 h 50 m	BLR to BOM to AUH to FCO	2,56,401
512	Air France, Alitalia	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	17 h 20 m	BLR to CDG to LIN to FCO	2,67,712
513	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	BLR to BOM to SIN to FCO	3,11,355
514	Vistara, Singapore Airlines	28 Oct 22	Bengaluru	Rome - Fiumicino Apt	2 stop	39 h	BLR to BOM to SIN to FCO	3,11,355

515 rows × 8 columns

```
In [34]: dt['Price'] = dt['Price'].apply(lambda x: float(x.split()[0].replace(',', '')))
```

```
In [37]: dt['Date'].replace('21 sep 22','21 Sept 22',inplace=True)

In [38]: dt['Date'].unique()

Out[38]: array(['18 Sept 22', '29 Sept 22', '29 Oct 22', '27 Nov 22', '30 Nov 22',
               '21 Sept 22', '28 Oct 22'], dtype=object)

In [39]: dt['Journey_day'] = dt['Date'].str.split(' ').str[0]
         dt['Journey_month'] = dt['Date'].str.split(' ').str[1]

In [40]: dt.drop('Date',axis=1,inplace=True)
```

- Data Inputs- Logic- Output Relationships

  Describe the relationship behind the data input, its format, the logic in between and the output. Describe how the input affects the output.

  Answer: Normally we use corr(), describe() to get better relation between input and output data. Basically every website considers the following Type of Airlines, time of flight, destination, source, price of the fuel, number of stops for the decision of the price.

- State the set of assumptions (if any) related to the problem under consideration

  Here, you can describe any presumptions taken by you.

  Answer: As Type of Airlines, time of flight, destination, source, price of the fuel, number of stops for the decision of the price in almost all the websites it is also taken to be input variable for the prediction.

- Hardware and Software Requirements and Tools Used

Listing down the hardware and software requirements along with the tools, libraries and packages used. Describe all the software tools used along with a detailed description of tasks done with those tools.

Answer: To analyse the present project we have used python libraries like numpy, pandas, seaborn, matplotlib, sklearn etc.

Numpy and pandas for converting the data to data frame, datacleaning etc.

Seaborn and matplotlib for explorative data analysis.

Sklearn for models, trainsplit and training etc.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Describe the approaches you followed, both statistical and analytical, for solving of this problem.

Answer: We have used different models from data provided from the website. Data was cleaned and trained by the model.

- Testing of Identified Approaches (Algorithms)

Listing down all the algorithms used for the training and testing.

Answer: : Different models used in the project are linear regression, logistic regression, Random Forest Regressor, XGBRegressor, AdaBoostRegressor, KNeighborsRegressor, SVR, Gradient Boosting Regressor.

- Run and Evaluate selected models

Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

In [146]:
```
for i in range(1,2000):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    li=LinearRegression()
    li.fit(x_train, y_train)
    predtrain=li.predict(x_train)
    predtest=li.predict(x_test)
    print(f"At random state, {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state, 1, the training accuracy is :-0.184004429229506
At random state, 1, the testing accuracy is :0.49636484620467636


At random state, 2, the training accuracy is :0.09868165605276036
At random state, 2, the testing accuracy is :0.2770979920386363


At random state, 3, the training accuracy is :-0.1292615653340754
At random state, 3, the testing accuracy is :0.4230684379655758


At random state, 4, the training accuracy is :-0.1469750544436994
At random state, 4, the testing accuracy is :0.44838070079027825


At random state, 5, the training accuracy is :-0.22568109292470595
At random state, 5, the testing accuracy is :0.5311125303735689
```

NONE

In [147]:
```
for i in range(1,400):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    re=RandomForestRegressor()
    re.fit(x_train, y_train)
    predtrain=re.predict(x_train)
    predtest=re.predict(x_test)
    print(f"At random state, {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state, 1, the training accuracy is :0.930266131789451
At random state, 1, the testing accuracy is :0.7568367288148827


At random state, 2, the training accuracy is :0.9094545688232664
At random state, 2, the testing accuracy is :0.6333538574794695


At random state, 3, the training accuracy is :0.933107756555103
At random state, 3, the testing accuracy is :0.7077814628708637


At random state, 4, the training accuracy is :0.9382638573200353
At random state, 4, the testing accuracy is :0.6536136996509664


At random state, 5, the training accuracy is :0.9294972124363576
At random state, 5, the testing accuracy is :0.6541962923427079
```

At random state, 256, the training accuracy is :0.9189197232257813 At random state, 256, the testing accuracy is :0.8621895961389421

In [148]:
```
for i in range(1,400):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    xg=XGBRegressor()
    xg.fit(x_train, y_train)
    predtrain=xg.predict(x_train)
    predtest=xg.predict(x_test)
    print(f"At random state {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state 1, the training accuracy is :0.9846603728353472
At random state, 1, the testing accuracy is :0.7677274959328132


At random state 2, the training accuracy is :0.9775537520920989
At random state, 2, the testing accuracy is :0.703364090120919


At random state 3, the training accuracy is :0.9825447709856042
At random state, 3, the testing accuracy is :0.7200772334895229


At random state 4, the training accuracy is :0.9883836058850252
At random state, 4, the testing accuracy is :0.6315336648550252


At random state 5, the training accuracy is :0.9887416200317186
At random state, 5, the testing accuracy is :0.6358478365727995
```

At random state 213, the training accuracy is :0.9851006393159201 At random state, 213, the testing accuracy is :0.8670978935518964

In [149]:
```
for i in range(1,400):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    ad=AdaBoostRegressor()
    ad.fit(x_train, y_train)
    predtrain=ad.predict(x_train)
    predtest=ad.predict(x_test)
    print(f"At random state, {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state, 1, the training accuracy is :0.10608099102454693
At random state, 1, the testing accuracy is :0.3930547937163056


At random state, 2, the training accuracy is :0.18785345766463202
At random state, 2, the testing accuracy is :0.37908089313222726


At random state, 3, the training accuracy is :0.4252801478826177
At random state, 3, the testing accuracy is :0.5570837850981857


At random state, 4, the training accuracy is :0.619154308722431
At random state, 4, the testing accuracy is :0.5486286620423351


At random state, 5, the training accuracy is :0.48466804284055964
At random state, 5, the testing accuracy is :0.549464798611041
```

NONE

In [150]:
```
for i in range(1,400):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    kn=KNeighborsRegressor()
    kn.fit(x_train, y_train)
    predtrain=kn.predict(x_train)
    predtest=kn.predict(x_test)
    print(f"At random state {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state 1, the training accuracy is :0.6047311278089675
At random state, 1, the testing accuracy is :0.7150451100431534


At random state 2, the training accuracy is :0.6570336947387765
At random state, 2, the testing accuracy is :0.521194438397144


At random state 3, the training accuracy is :0.6371038007571714
At random state, 3, the testing accuracy is :0.5750468658645398


At random state 4, the training accuracy is :0.6039175317813147
At random state, 4, the testing accuracy is :0.6588411608386406


At random state 5, the training accuracy is :0.6180816650336347
At random state, 5, the testing accuracy is :0.5417746633884482
```

At random state 152, the training accuracy is :0.6451172815728641 At random state, 152, the testing accuracy is :0.6406373444013416

In [151]:
```
for i in range(1,200):
    x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
    sr=SVR()
    sr.fit(x_train, y_train)
    predtrain=sr.predict(x_train)
    predtest=sr.predict(x_test)
    print(f"At random state {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
    print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
    print("\n")
```

```
At random state 7, the training accuracy is :-7156626.248233519
At random state, 7, the testing accuracy is :-0.01856376802774662


At random state 8, the training accuracy is :-6791605.215493745
At random state, 8, the testing accuracy is :-0.05874135112511203


At random state 9, the training accuracy is :-8703627.51804664
At random state, 9, the testing accuracy is :-0.035872574583808214


At random state 10, the training accuracy is :-7273582.383233338
At random state, 10, the testing accuracy is :-0.01772201300597165


At random state 11, the training accuracy is :-8040318.636410757
```

NONE

```
In [152]: for i in range(1,400):
              x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=i)
              gr=GradientBoostingRegressor()
              gr.fit(x_train, y_train)
              predtrain=gr.predict(x_train)
              predtest=gr.predict(x_test)
              print(f"At random state, {i}, the training accuracy is :{r2_score(predtrain,y_train)}")
              print(f"At random state, {i}, the testing accuracy is :{r2_score(y_test, predtest)}")
              print("\n")
```

```
At random state, 1, the training accuracy is :0.8369798765026053
At random state, 1, the testing accuracy is :0.7249225882351606


At random state, 2, the training accuracy is :0.8388619105932106
At random state, 2, the testing accuracy is :0.6879291845283011


At random state, 3, the training accuracy is :0.852644436769887
At random state, 3, the testing accuracy is :0.6743731092176486


At random state, 4, the training accuracy is :0.8456440416657527
At random state, 4, the testing accuracy is :0.6592085055592627


At random state, 5, the training accuracy is :0.8482966623149224
At random state, 5, the testing accuracy is :0.6666523198132595
```

```
At random state, 107, the training accuracy is :0.8109132521855902
At random state, 107, the testing accuracy is :0.80601231330102
```

```
In [153]: x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=256)
          re=RandomForestRegressor()
          re.fit(x_train, y_train)
          predtrain=re.predict(x_train)
          predtest=re.predict(x_test)
          print(f"At random state, {256}, the training accuracy is :{r2_score(predtrain,y_train)}")
          print(f"At random state, {256}, the testing accuracy is :{r2_score(y_test, predtest)}")
```

```
At random state, 256, the training accuracy is :0.9186982017256481
At random state, 256, the testing accuracy is :0.8667305553271245
```

```
In [154]: # Calculating the MSE, RMSE and MAE for K-NN model
          mse=mean_absolute_error(y_test, predtest)
          rmse=np.sqrt(mse)
          print('The MAE is', mean_absolute_error(y_test, predtest))
          print ('The MSE is', mse, 'and RMSE is', rmse)
```

```
The MAE is 12570.23404381541
The MSE is 12570.23404381541 and RMSE is 112.1170550978548
```

```
In [155]: x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=213)
          xg=XGBRegressor()
          xg.fit(x_train, y_train)
          predtrain=xg.predict(x_train)
          predtest=xg.predict(x_test)
          print(f"At random state {213}, the training accuracy is :{r2_score(predtrain,y_train)}")
          print(f"At random state, {213}, the testing accuracy is :{r2_score(y_test, predtest)}")
```

```
At random state 213, the training accuracy is :0.9851006393159201
At random state, 213, the testing accuracy is :0.8670978935518964
```

```
In [156]: # Calculating the MSE, RMSE and MAE for K-NN model
mse=mean_absolute_error(y_test, predtest)
rmse=np.sqrt(mse)
print('The MAE is', mean_absolute_error(y_test, predtest))
print ('The MSE is', mse, 'and RMSE is', rmse)

The MAE is 16838.357445577974
The MSE is 16838.357445577974 and RMSE is 129.7626966642493
```

```
In [157]: x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=152)
kn=KNeighborsRegressor()
kn.fit(x_train, y_train)
predtrain=kn.predict(x_train)
predtest=kn.predict(x_test)
print(f"At random state {152}, the training accuracy is :{r2_score(predtrain,y_train)}")
print(f"At random state, {152}, the testing accuracy is :{r2_score(y_test, predtest)}")

At random state 152, the training accuracy is :0.6451172815728641
At random state, 152, the testing accuracy is :0.6406373444013416
```

```
In [158]: # Calculating the MSE, RMSE and MAE for K-NN model
mse=mean_absolute_error(y_test, predtest)
rmse=np.sqrt(mse)
print('The MAE is', mean_absolute_error(y_test, predtest))
print ('The MSE is', mse, 'and RMSE is', rmse)

The MAE is 26187.456310679616
The MSE is 26187.456310679616 and RMSE is 161.8253883377995
```

```
In [159]: x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=107)
gr=GradientBoostingRegressor()
gr.fit(x_train, y_train)
predtrain=gr.predict(x_train)
predtest=gr.predict(x_test)
print(f"At random state, {107}, the training accuracy is :{r2_score(predtrain,y_train)}")
print(f"At random state, {107}, the testing accuracy is :{r2_score(y_test, predtest)}")

At random state, 107, the training accuracy is :0.8109132521855902
At random state, 107, the testing accuracy is :0.8061442234434835
```

```
In [160]: # Calculating the MSE, RMSE and MAE for K-NN model
mse=mean_absolute_error(y_test, predtest)
rmse=np.sqrt(mse)
print('The MAE is', mean_absolute_error(y_test, predtest))
print ('The MSE is', mse, 'and RMSE is', rmse)

The MAE is 18851.063292273397
The MSE is 18851.063292273397 and RMSE is 137.29917440492275
```

```
In [163]: #Highest accuracy is achieved in XGBoost
x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=256)
re=RandomForestRegressor()
re.fit(x_train, y_train)
predtrain=re.predict(x_train)
predtest=re.predict(x_test)
print(f"At random state, {256}, the training accuracy is :{r2_score(predtrain,y_train)}")
print(f"At random state, {256}, the testing accuracy is :{r2_score(y_test, predtest)}")
# Calculating the MSE, RMSE and MAE for K-NN model
mse=mean_absolute_error(y_test, predtest)
rmse=np.sqrt(mse)
print('The MAE is', mean_absolute_error(y_test, predtest))
print ('The MSE is', mse, 'and RMSE is', rmse)
```

```
In [165]: #Hyperparameter tuning
          from sklearn.model_selection import GridSearchCV
          parameters={'n_estimators' : range(2,15),
                     'n_jobs': range(2,10),
                     'max_depth': range(2,15)}
```

```
In [167]: c=GridSearchCV(RandomForestRegressor(),param_grid=parameters)
          c.fit(x_train, y_train)
          print ('The best parameters are', c.best_params_)

          The best parameters are {'max_depth': 14, 'n_estimators': 10, 'n_jobs': 5}
```

```
In [169]: x_train,x_test,y_train, y_test=train_test_split(X_scaled,y,test_size=.20, random_state=256)
          re=RandomForestRegressor(max_depth=14, n_estimators=10, n_jobs=5)
          re.fit(x_train, y_train)
          predtrain=re.predict(x_train)
          predtest=re.predict(x_test)
          print(f"At random state, {256}, the training accuracy is :{r2_score(predtrain,y_train)}")
          print(f"At random state, {256}, the testing accuracy is :{r2_score(y_test, predtest)}")
          # Calculating the MSE, RMSE and MAE for K-NN model
          mse=mean_absolute_error(y_test, predtest)
          rmse=np.sqrt(mse)
          print('The MAE is', mean_absolute_error(y_test, predtest))
          print ('The MSE is', mse, 'and RMSE is', rmse)

          At random state, 256, the training accuracy is :0.9130650810656163
          At random state, 256, the testing accuracy is :0.8812360857622321
          The MAE is 12974.838366156499
          The MSE is 12974.838366156499 and RMSE is 113.90714800290849
```

- Key Metrics for success in solving problem under consideration

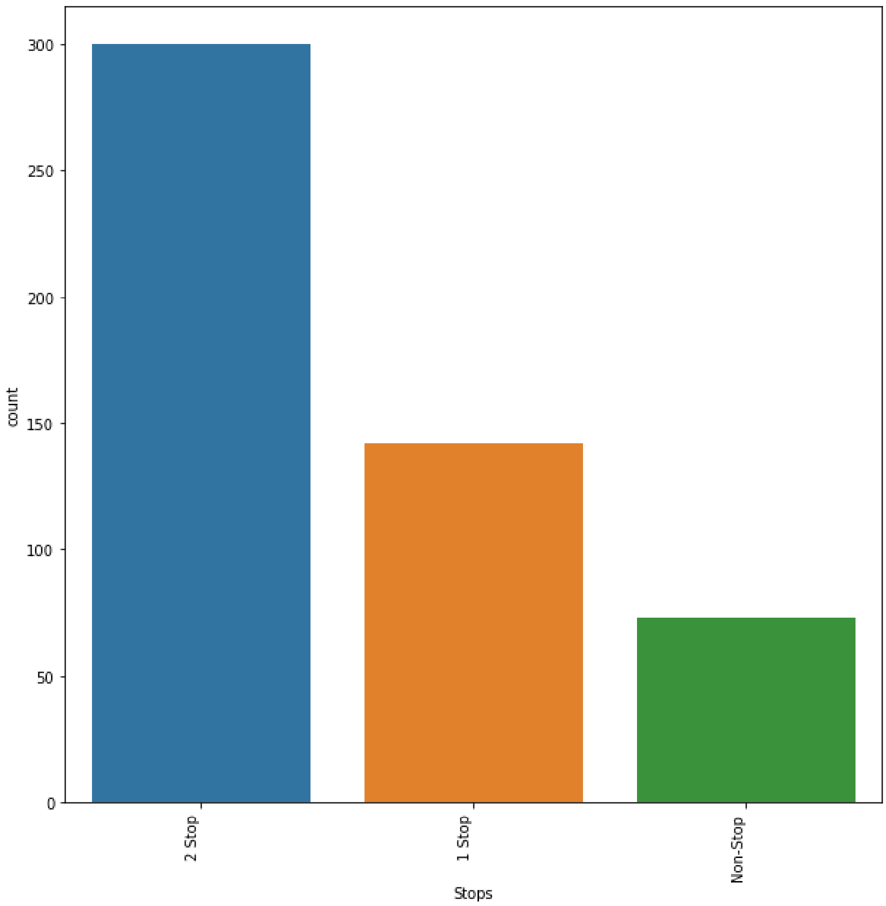
  What were the key metrics used along with justification for using it? You may also include statistical metrics used if any.
  Answer: The metrics used were The R2score, MAE, MSE and RMSE.

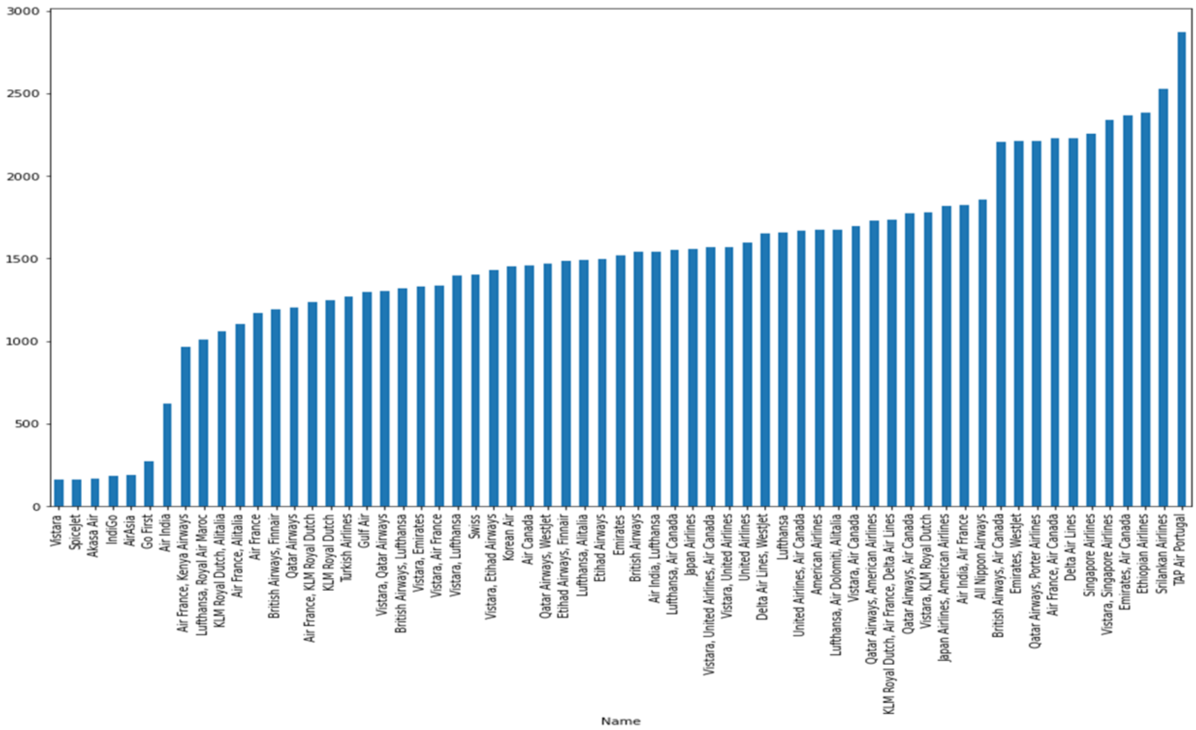
- Visualizations

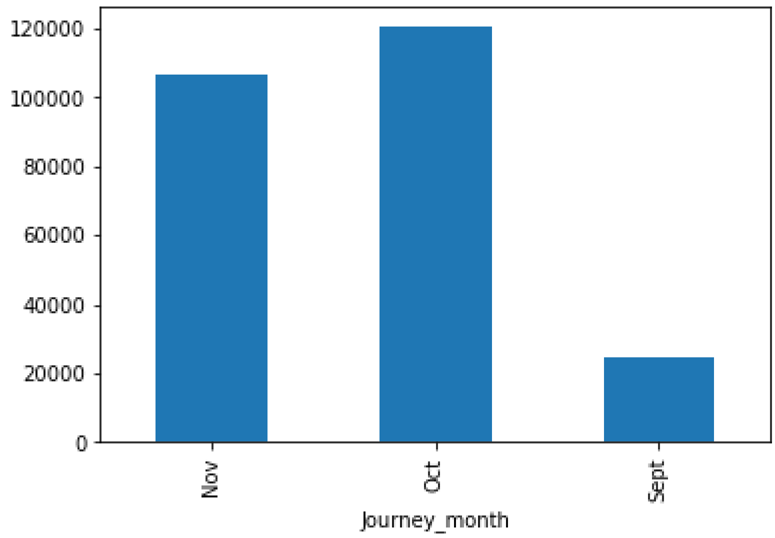
  Mention all the plots made along with their pictures and what were the inferences and observations obtained from those. Describe them in detail.

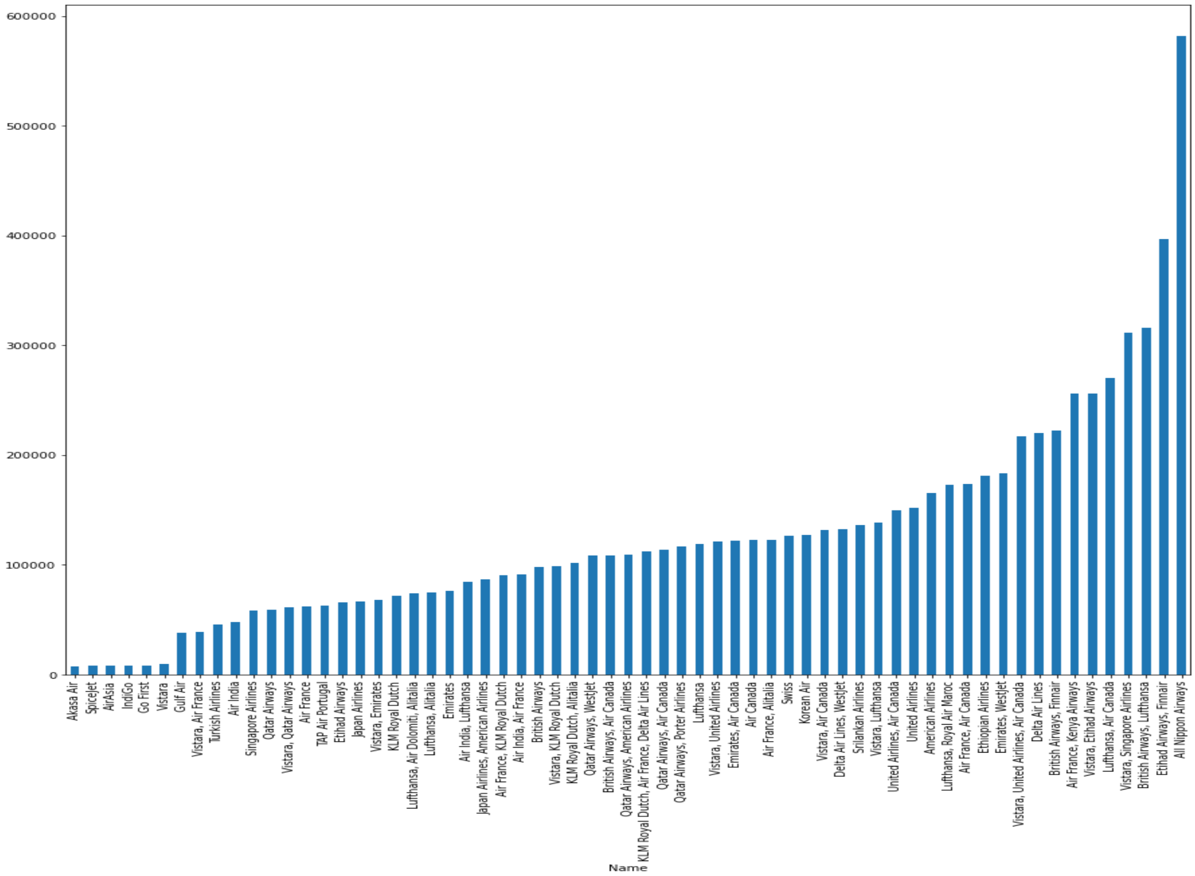
  If different platforms were used, mention that as well.

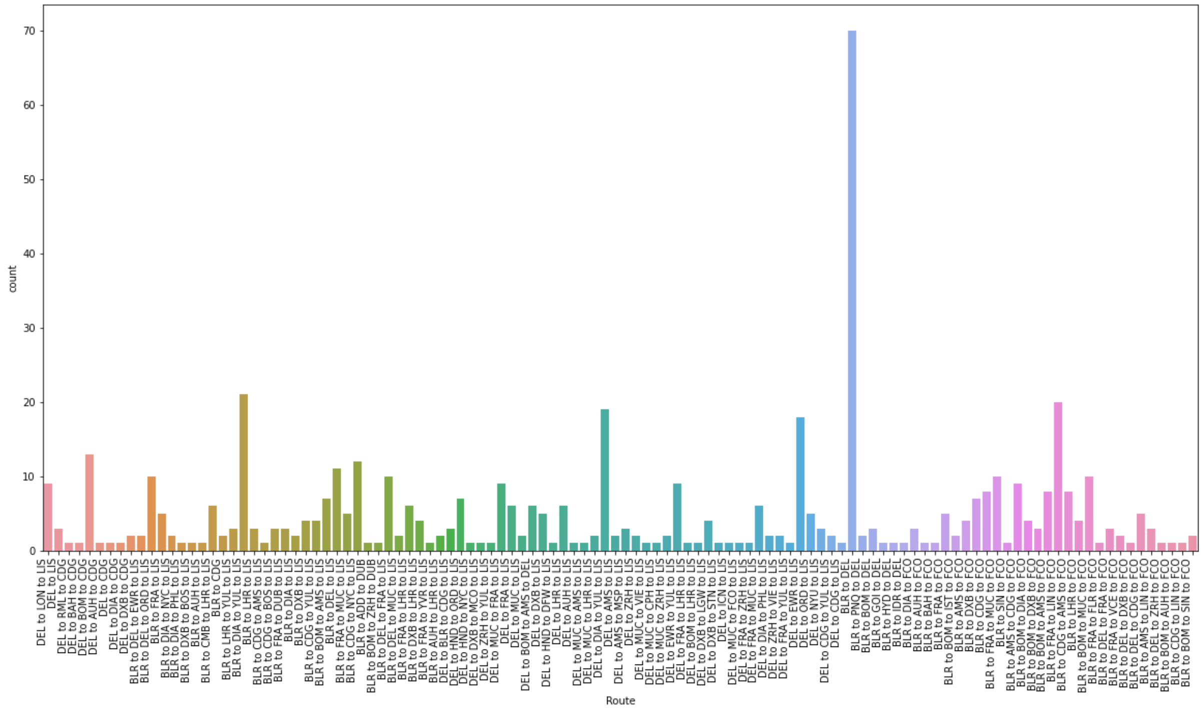
  Answer: The different plots were made to predict the cost, number of stops, destination, source etc. only python was used for explorative data analysis.

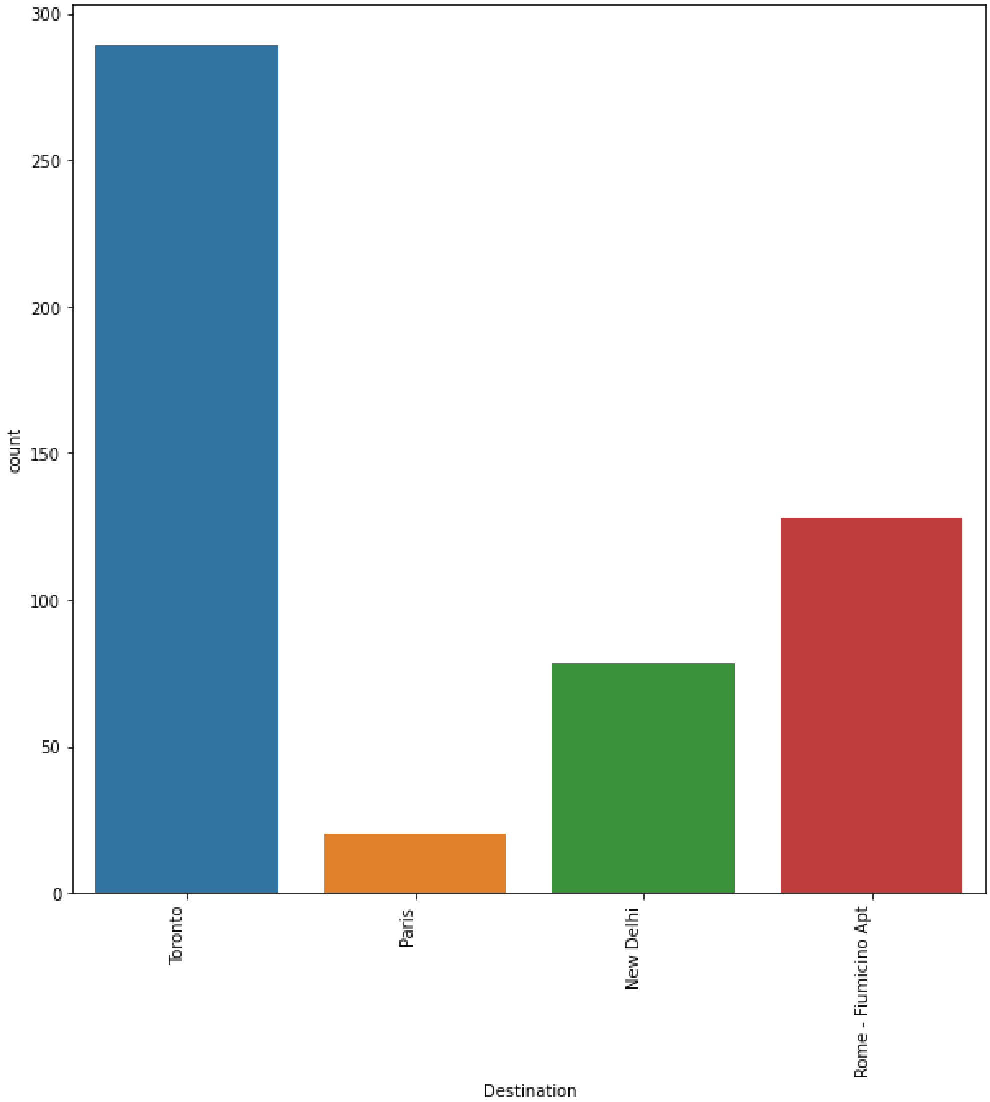

300
250
200
150
100
50
0
count
2 Stop
1 Stop
Non-Stop
Stops

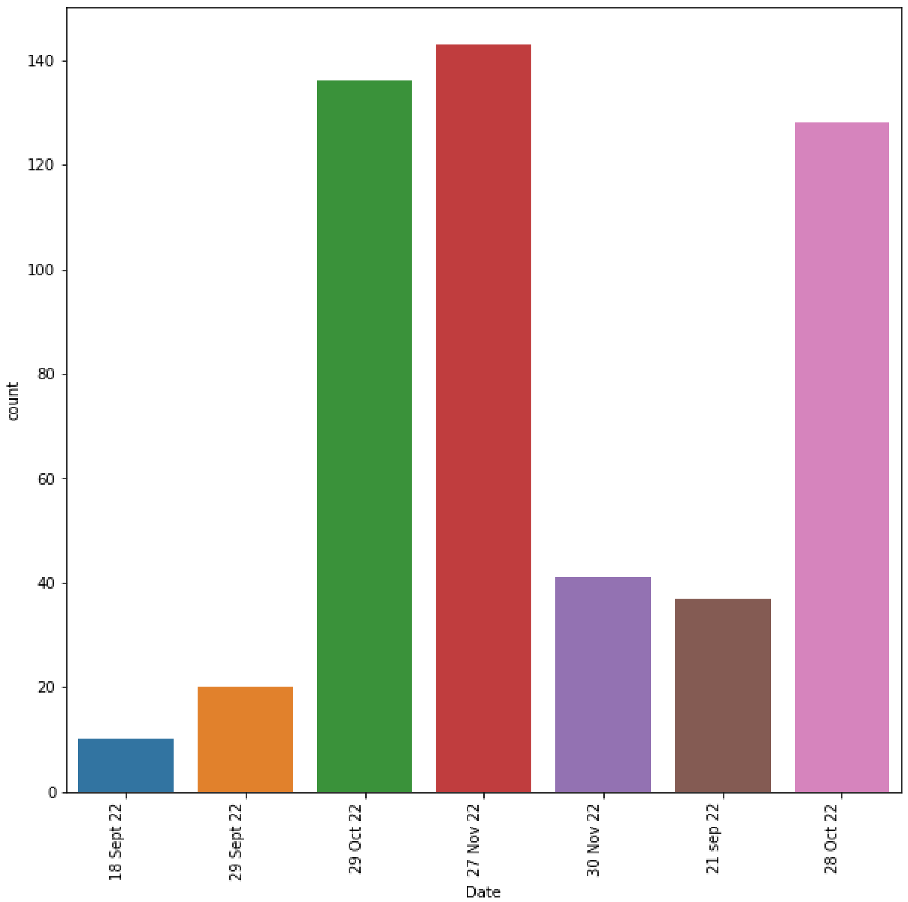

3000
2500
2000
1500
1000
500
0
Vistara
SpiceJet
Akasa Air
IndiGo
AirAsia
Go First
Air India
Air France, Kenya Airways
Lufthansa, Royal Air Maroc
KLM Royal Dutch, Alitalia
Air France, Alitalia
Air France
British Airways, Finnair
Qatar Airways
Air France, KLM Royal Dutch
KLM Royal Dutch
Turkish Airlines
Gulf Air
Vistara, Qatar Airways
British Airways, Lufthansa
Vistara, Emirates
Vistara, Air France
Vistara, Lufthansa
Swiss
Vistara, Etihad Airways
Korean Air
Air Canada
Qatar Airways, WestJet
Etihad Airways, Finnair
Lufthansa, Alitalia
Etihad Airways
Emirates
British Airways
Air India, Lufthansa
Lufthansa, Air Canada
Japan Airlines
Vistara, United Airlines, Air Canada
Vistara, United Airlines
United Airlines
Delta Air Lines, WestJet
Lufthansa
United Airlines, Air Canada
American Airlines
Lufthansa, Air Dolomiti, Alitalia
Vistara, Air Canada
Qatar Airways, American Airlines
KLM Royal Dutch, Air France, Delta Air Lines
Qatar Airways, Air Canada
Vistara, KLM Royal Dutch
Japan Airlines, American Airlines
Air India, Air France
All Nippon Airways
British Airways, Air Canada
Emirates, WestJet
Qatar Airways, Porter Airlines
Air France, Air Canada
Delta Air Lines
Singapore Airlines
Vistara, Singapore Airlines
Emirates, Air Canada
Ethiopian Airlines
Srilankan Airlines
TAP Air Portugal
Name


120000
100000
80000
60000
40000
20000
0
Nov
Oct
Sept
Journey_month

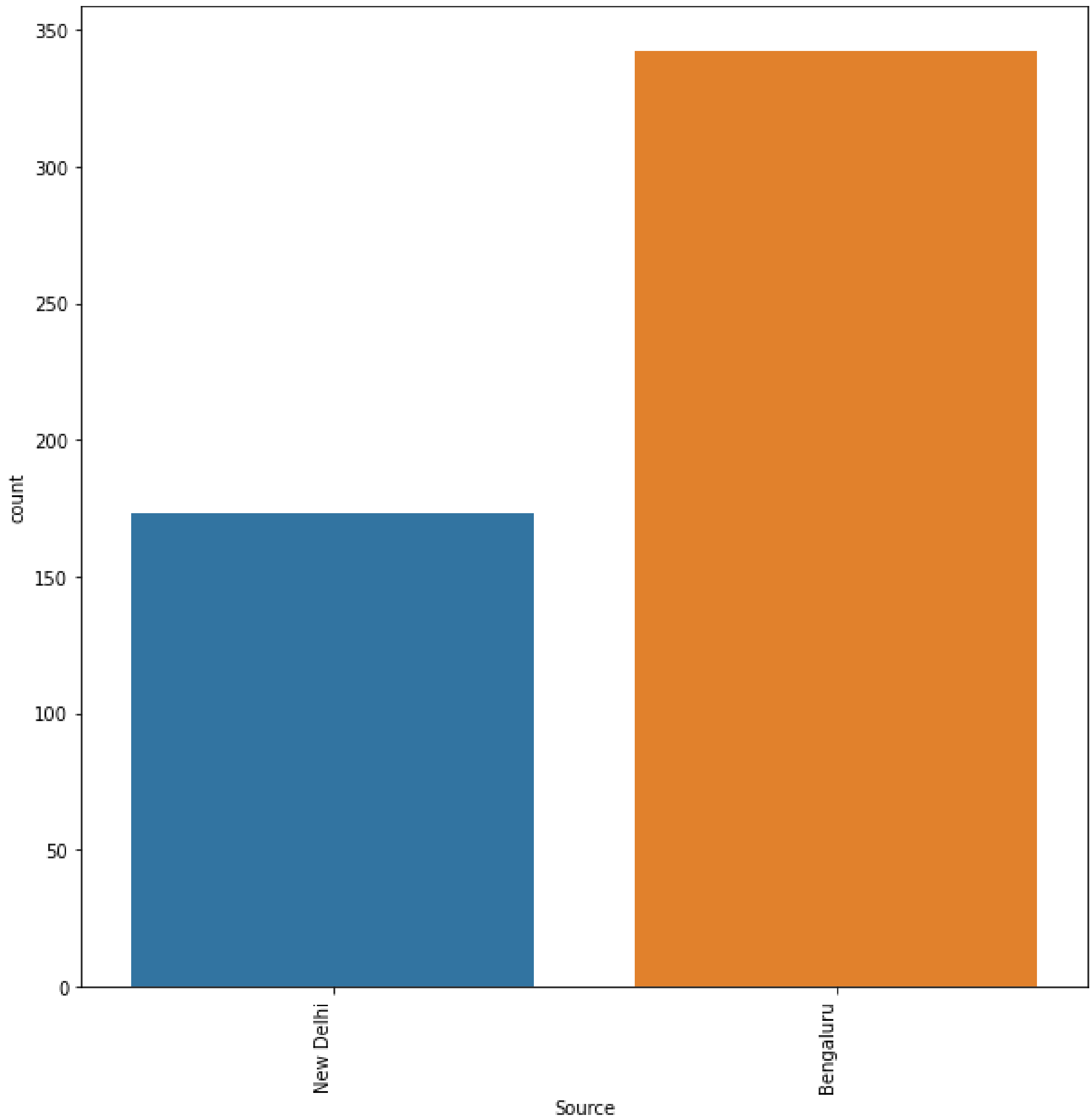

600000
500000
400000
300000
200000
100000
0
Akasa Air
SpiceJet
AirAsia
IndiGo
Go First
Vistara
Gulf Air
Vistara, Air France
Turkish Airlines
Air India
Singapore Airlines
Qatar Airways
Vistara, Qatar Airways
Air France
TAP Air Portugal
Etihad Airways
Japan Airlines
Vistara, Emirates
KLM Royal Dutch
Lufthansa, Air Dolomiti, Alitalia
Lufthansa, Alitalia
Emirates
Air India, Lufthansa
Japan Airlines, American Airlines
Air France, KLM Royal Dutch
Air India, Air France
British Airways
Vistara, KLM Royal Dutch
KLM Royal Dutch, Alitalia
Qatar Airways, WestJet
British Airways, Air Canada
Qatar Airways, American Airlines
KLM Royal Dutch, Air France, Delta Air Lines
Qatar Airways, Air Canada
Qatar Airways, Porter Airlines
Lufthansa
Vistara, United Airlines
Emirates, Air Canada
Air Canada
Air France, Alitalia
Swiss
Korean Air
Vistara, Air Canada
Delta Air Lines, WestJet
Srilankan Airlines
Vistara, Lufthansa
United Airlines, Air Canada
United Airlines
American Airlines
Lufthansa, Royal Air Maroc
Air France, Air Canada
Ethiopian Airlines
Emirates, WestJet
Vistara, United Airlines, Air Canada
Delta Air Lines
British Airways, Finnair
Air France, Kenya Airways
Vistara, Etihad Airways
Lufthansa, Air Canada
Vistara, Singapore Airlines
British Airways, Lufthansa
Etihad Airways, Finnair
All Nippon Airways
Name


count
0
10
20
30
40
50
60
70
Route


count
0
50
100
150
200
250
300
Toronto
Paris
New Delhi
Rome - Fiumicino Apt
Destination



140
120
100
80
60
40
20
0
count
18 Sept 22
29 Sept 22
29 Oct 22
27 Nov 22
30 Nov 22
21 sep 22
28 Oct 22
Date



- Interpretation of the Results

Give a summary of what results were interpreted from the visualizations, preprocessing and modelling.

Answer: It was seen that TAP AIR RORTUGAL was the airways having highest amount of travelling time. The month of October most travelling occurred according to the dataset. Most of them have 2 stops at least to commute. ALL NIPPON AIRWAYS is the most costliest. Most travelling occurred on 27 NOV 2022. From the dataset it was seen that TORONTO was the most destination place.

CONCLUSION

- ## Key Findings and Conclusions of the Study

Describe the key findings, inferences, observations from the whole problem.

Answer: It was seen that TAP AIR RORTUGAL was the airways having highest amount of travelling time. The month of October most travelling occurred according to the dataset. Most of them have 2 stops at least to commute. ALL NIPPON AIRWAYS is the most costliest. Most travelling occurred on 27 NOV 2022. From the dataset it was seen that TORONTO was the most destination place.

- ## Learning Outcomes of the Study in respect of Data Science

List down your learnings obtained about the power of visualization, data cleaning and various algorithms used. You can describe which algorithm works best in which situation and what challenges you faced while working on this project and how did you overcome that.

Answer: From the graphs obtained by the python different conclusions were drawn. As the data was supposed to be taken from the different websites there were challenges in scraping them. Joining and Cleaning of data was also the next challenge as there were many column and junk data.

Limitations of this work and Scope for Future Work

What are the limitations of this solution provided, the future scope? What all steps/techniques can be followed to further extend this study and improve the results.

Answer: Different models were tested but it was seen that after hyperparameter tuning it was found to have at random state, 256, the training accuracy is :0.9130650810656163. At random state, 256, the testing accuracy is :0.8812360857622321. The MAE is 12974.838366156499. The MSE is 12974.838366156499 and RMSE is 113.90714800290849. in order to increase this accuracy different dataset have to be still collected for different destinations. Different deep learning methods can also be employed.