In [1]:

```
#IMPORT REQUIRED LIBRARIES
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
import matplotlib
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
train=pd.read_excel('Data_Train.xlsx')
train.head()
```

Out[2]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops
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

In [3]:

```
train.shape
```

Out[3]:

```
(10683, 11)
```

In [4]:

```
train.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]:

```
train.columns
```

Out[5]:

```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info', 'Price'],
      dtype='object')
```

In [6]:

```
train.describe()
```

Out[6]:

	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

This column shows the statistical values for the numerical data like mean,standard deviation and percentiles,minimum and maximum.

In [7]:

```
#checking for missing values
train.isnull().sum()
```

Out[7]:

```
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 [8]:

```
train.dropna(inplace=True)
train
```

Out[8]:

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

10682 rows × 11 columns

In [9]:

```
train.shape
```

Out[9]:

```
(10682, 11)
```

In [10]:

```
train.isnull().sum()
```

Out[10]:

```
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 [11]:

```
train['Date_of_Journey'].value_counts
```

Out[11]:

```
<bound method IndexOpsMixin.value_counts of 0        24/03/2019
1         1/05/2019
2         9/06/2019
3        12/05/2019
4        01/03/2019
            ...
10678     9/04/2019
10679    27/04/2019
10680    27/04/2019
10681    01/03/2019
10682     9/05/2019
Name: Date_of_Journey, Length: 10682, dtype: object>
```

In [12]:

```
train['Date_of_Journey'].unique
```

Out[12]:

```
<bound method Series.unique of 0        24/03/2019
1         1/05/2019
2         9/06/2019
3        12/05/2019
4        01/03/2019
            ...
10678     9/04/2019
10679    27/04/2019
10680    27/04/2019
10681    01/03/2019
10682     9/05/2019
Name: Date_of_Journey, Length: 10682, dtype: object>
```

EDA

In [13]:

```
train["journey_day"]=pd.to_datetime(train.Date_of_Journey,format="%d/%m/%Y").dt.day
```

In [14]:

```
train['journey_month']=pd.to_datetime(train.Date_of_Journey,format="%d/%m/%Y").dt.mont
```

In [15]:

```
train.head()
```

Out[15]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops
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

As model only understand numeric values instead strings so we have converted the string into day and month using date time format.

In [16]:

```
#As we have coverted date_of_journey column into integers so we are now goig to drop
train.drop(['Date_of_Journey'],axis=1,inplace=True)
```

In [17]:

```
train["Departure_hr"]=pd.to_datetime(train['Dep_Time']).dt.hour
```

In [18]:

```
train["Departure_min"]=pd.to_datetime(train['Dep_Time']).dt.minute
```

Now,similarly we have converted the departure time into integer format also and droped the Dep_Time column as it is of no use.

In [19]:

```
#As we have coverted Dep_Time column into integers so we are now going to drop it as
train.drop(['Dep_Time'],axis=1,inplace=True)
```

In [20]:

```
train.head()
```

Out[20]:

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

In [21]:

```
train["Arrival_hr"]=pd.to_datetime(train['Arrival_Time']).dt.hour          #converted
train["Arrival_min"]=pd.to_datetime(train['Arrival_Time']).dt.minute       #converted
train.drop(['Arrival_Time'],axis=1,inplace=True)                           #droped the
```

In [22]:

```
train.head()
```

Out[22]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	journey_day	jou
0	IndiGo	Banglore	New Delhi	BLR → DEL	2h 50m	non-stop	No info	3897	24	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	7h 25m	2 stops	No info	7662	1	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	19h	2 stops	No info	13882	9	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	5h 25m	1 stop	No info	6218	12	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	4h 45m	1 stop	No info	13302	1	

In [23]:

```
# Assigning and converting Duration column into list
duration = list(train["Duration"])

for i in range(len(duration)):
    if len(duration[i].split()) != 2:                   # Check if duration contains
        if "h" in duration[i]:
            duration[i] = duration[i].strip() + " 0m"   # Adds 0 minute
        else:
            duration[i] = "0h " + duration[i]           # Adds 0 hour

duration_hours = []
duration_mins = []
for i in range(len(duration)):
    duration_hours.append(int(duration[i].split(sep = "h")[0]))               # Extrac
    duration_mins.append(int(duration[i].split(sep = "m")[0].split()[-1]))    # Extrac
```

In [24]:

```
train['duration_hours']=duration_hours
train['duration_mins']=duration_mins
```

In [25]:

```
train.head()
```

Out[25]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	journey_day	jou
0	IndiGo	Banglore	New Delhi	BLR → DEL	2h 50m	non-stop	No info	3897	24	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	7h 25m	2 stops	No info	7662	1	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	19h	2 stops	No info	13882	9	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	5h 25m	1 stop	No info	6218	12	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	4h 45m	1 stop	No info	13302	1	

In [26]:

```
train.drop(['Duration'],axis=1,inplace=True)
```

In [27]:

```
train.head()
```

Out[27]:

	Airline	Source	Destination	Route	Total_Stops	Additional_Info	Price	journey_day	journey_montl
0	IndiGo	Banglore	New Delhi	BLR → DEL	non-stop	No info	3897	24	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	2 stops	No info	7662	1	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	2 stops	No info	13882	9	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	1 stop	No info	6218	12	
4	IndiGo	Banglore	New Delhi	BLR → NAG → DEL	1 stop	No info	13302	1	

In [28]:

```
train['Airline'].value_counts()
```

Out[28]:

```
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 [29]:

```
sns.catplot(x='Airline',y='Price',data=train,kind='box',height=5,aspect=4)
```

Out[29]:

```
<seaborn.axisgrid.FacetGrid at 0x12850f310>
```

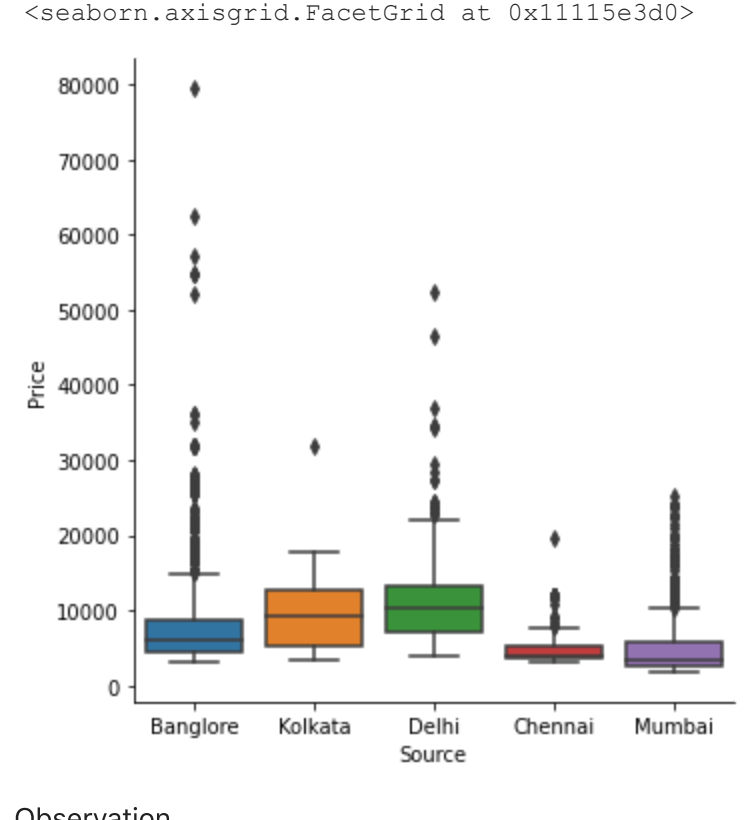
Observation As we can see the fare price is highest for jet airways business.

In [30]:

```
sns.catplot(x='Source',y='Price',data=train,kind='box',height=5,aspect=1)
```

Out[30]:

```
<seaborn.axisgrid.FacetGrid at 0x11115e3d0>
```



Observation

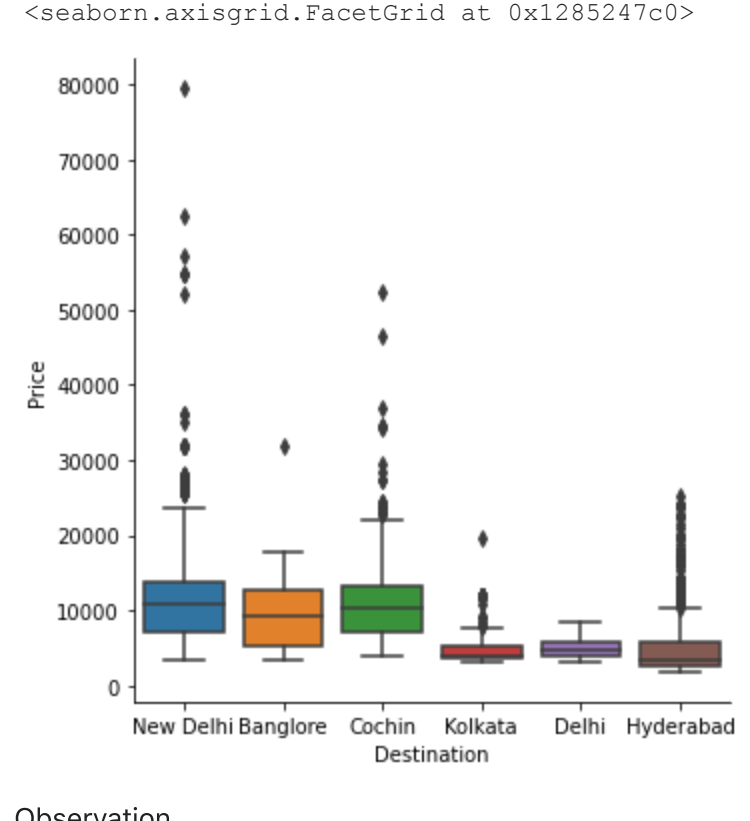
Outliers are present very much in bangalore.

In [31]:

```
sns.catplot(x='Destination',y='Price',data=train,kind='box',height=5,aspect=1)
```

Out[31]:

```
<seaborn.axisgrid.FacetGrid at 0x1285247c0>
```



Observation

Outliers are present very much in new delhi.

In [33]:

```
Airline=train[['Airline']]
Airline=pd.get_dummies(Airline,drop_first=True)
Airline.head()
```

Out[33]:

	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airli
0	0	0	1	0	0	0	0	
1	1	0	0	0	0	0	0	
2	0	0	0	1	0	0	0	
3	0	0	1	0	0	0	0	
4	0	0	1	0	0	0	0	

In [35]:

```
train['Source'].unique()
```

Out[35]:

```
array(['Banglore', 'Kolkata', 'Delhi', 'Chennai', 'Mumbai'], dtype=object)
```

In [36]:

```
train['Source'].value_counts()
```

Out[36]:

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

In [40]:

```
Source=train[['Source']]
Source=pd.get_dummies(Source,drop_first=True)
Source.head()
```

Out[40]:

	Source_Chennai	Source_Delhi	Source_Kolkata	Source_Mumbai
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	0

In [41]:

```
train['Destination'].value_counts()
```

Out[41]:

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

In [42]:

```
Destination=train[['Destination']]
Destination=pd.get_dummies(Destination,drop_first=True)
Destination.head()
```

Out[42]:

	Destination_Cochin	Destination_Delhi	Destination_Hyderabad	Destination_Kolkata	Destination_New Delhi
0	0	0	0	0	1
1	0	0	0	0	0
2	1	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	1

In [43]:

```
train['Route']
```

Out[43]:

```
0                    BLR → DEL
1        CCU → IXR → BBI → BLR
2        DEL → LKO → BOM → COK
3              CCU → NAG → BLR
4              BLR → NAG → DEL
                 ...
10678                CCU → BLR
10679                CCU → BLR
10680                BLR → DEL
10681                BLR → DEL
10682    DEL → GOI → BOM → COK
Name: Route, Length: 10682, dtype: object
```

In [44]:

```
train.drop(['Route','Additional_Info'],axis=1,inplace=True)
```

In [45]:

```
train.head()
```

Out[45]:

	Airline	Source	Destination	Total_Stops	Price	journey_day	journey_month	Departure_hr	Depart
0	IndiGo	Banglore	New Delhi	non-stop	3897	24	3	22	
1	Air India	Kolkata	Banglore	2 stops	7662	1	5	5	
2	Jet Airways	Delhi	Cochin	2 stops	13882	9	6	9	
3	IndiGo	Kolkata	Banglore	1 stop	6218	12	5	18	
4	IndiGo	Banglore	New Delhi	1 stop	13302	1	3	16	

In [46]:

```
train['Total_Stops'].value_counts()
```

Out[46]:

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

In [47]:

```
train.replace({'non-stop':0,'1 stop':1,'2 stops':2,'3 stops':3,'4 stops':4},inplace=Tr
```

In [48]:

```
train.head()
```

Out[48]:

	Airline	Source	Destination	Total_Stops	Price	journey_day	journey_month	Departure_hr	Depart
0	IndiGo	Banglore	New Delhi	0	3897	24	3	22	
1	Air India	Kolkata	Banglore	2	7662	1	5	5	
2	Jet Airways	Delhi	Cochin	2	13882	9	6	9	
3	IndiGo	Kolkata	Banglore	1	6218	12	5	18	
4	IndiGo	Banglore	New Delhi	1	13302	1	3	16	

In [49]:

```
data_train=pd.concat([train,Airline,Source,Destination],axis=1)
```

In [50]:

```
data_train.head()
```

Out[50]:

	Airline	Source	Destination	Total_Stops	Price	journey_day	journey_month	Departure_hr	Depart
0	IndiGo	Banglore	New Delhi	0	3897	24	3	22	
1	Air India	Kolkata	Banglore	2	7662	1	5	5	
2	Jet Airways	Delhi	Cochin	2	13882	9	6	9	
3	IndiGo	Kolkata	Banglore	1	6218	12	5	18	
4	IndiGo	Banglore	New Delhi	1	13302	1	3	16	

5 rows × 33 columns

In [51]:

```
data_train.drop(['Airline','Source','Destination'],axis=1,inplace=True)
```

In [52]:

```
data_train.head()
```

Out[52]:

	Total_Stops	Price	journey_day	journey_month	Departure_hr	Departure_min	Arrival_hr	Arrival_min
0	0	3897	24	3	22	20	1	10
1	2	7662	1	5	5	50	13	15
2	2	13882	9	6	9	25	4	25
3	1	6218	12	5	18	5	23	30
4	1	13302	1	3	16	50	21	35

5 rows × 30 columns

In [53]:

```
data_train.shape
```

Out[53]:

```
(10682, 30)
```

In [54]:

```
test=pd.read_excel('Test_set.xlsx')
test.head()
```

Out[54]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stop
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL → BOM → COK	17:30	04:25 07 Jun	10h 55m	1 stop
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU → MAA → BLR	06:20	10:20	4h	1 stop
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL → BOM → COK	19:15	19:00 22 May	23h 45m	1 stop
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL → BOM → COK	08:00	21:00	13h	1 stop
4	Air Asia	24/06/2019	Banglore	Delhi	BLR → DEL	23:55	02:45 25 Jun	2h 50m	non-stop

In [55]:

```
test.shape
```

Out[55]:

```
(2671, 10)
```

In [57]:

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2671 entries, 0 to 2670
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          2671 non-null   object
 1   Date_of_Journey  2671 non-null   object
 2   Source           2671 non-null   object
 3   Destination      2671 non-null   object
 4   Route            2671 non-null   object
 5   Dep_Time         2671 non-null   object
 6   Arrival_Time     2671 non-null   object
 7   Duration         2671 non-null   object
 8   Total_Stops      2671 non-null   object
 9   Additional_Info  2671 non-null   object
dtypes: object(10)
memory usage: 208.8+ KB
```

In [59]:

```
test.columns
```

Out[59]:

```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
       'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
       'Additional_Info'],
      dtype='object')
```

In [61]:

```
test.describe()
```

Out[61]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_S
count	2671	2671	2671	2671	2671	2671	2671	2671	
unique	11	44	5	6	100	199	704	320	
top	Jet Airways	9/05/2019	Delhi	Cochin	DEL → BOM → COK	10:00	19:00	2h 50m	1
freq	897	144	1145	1145	624	62	113	122	

In [62]:

```
#checking for missing values
test.isnull().sum()
```

Out[62]:

```
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
dtype: int64
```

In [66]:

```
test['Date_of_Journey'].value_counts()
```

Out[66]:

```
9/05/2019     144
12/06/2019    135
18/05/2019    129
6/06/2019     127
9/06/2019     119
21/05/2019    118
15/05/2019    106
15/06/2019    105
6/03/2019      97
21/03/2019     93
3/06/2019      92
1/06/2019      88
24/06/2019     85
24/03/2019     78
1/04/2019      78
27/06/2019     75
6/05/2019      73
24/05/2019     71
12/05/2019     68
27/03/2019     65
27/05/2019     65
3/03/2019      62
1/05/2019      62
9/03/2019      55
12/03/2019     43
18/03/2019     41
01/03/2019     34
15/03/2019     33
06/03/2019     28
3/04/2019      28
3/05/2019      27
03/03/2019     26
9/04/2019      24
21/06/2019     24
15/04/2019     23
18/06/2019     22
21/04/2019     22
09/03/2019     21
24/04/2019     21
27/04/2019     15
6/04/2019      14
18/04/2019     12
1/03/2019      12
12/04/2019     11
Name: Date_of_Journey, dtype: int64
```

In [67]:

```
test['Date_of_Journey'].unique()
```

Out[67]:
```
array(['6/06/2019', '12/05/2019', '21/05/2019', '24/06/2019',
       '12/06/2019', '12/03/2019', '1/05/2019', '15/03/2019',
       '18/05/2019', '21/03/2019', '15/06/2019', '15/05/2019',
       '3/06/2019', '06/03/2019', '24/03/2019', '6/03/2019', '9/05/2019',
       '18/03/2019', '6/04/2019', '1/06/2019', '3/03/2019', '27/03/2019',
       '9/06/2019', '3/05/2019', '1/04/2019', '18/06/2019', '15/04/2019',
       '6/05/2019', '9/03/2019', '3/04/2019', '27/06/2019', '21/06/2019',
       '21/04/2019', '18/04/2019', '9/04/2019', '24/05/2019',
       '01/03/2019', '09/03/2019', '27/05/2019', '03/03/2019',
       '27/04/2019', '1/03/2019', '24/04/2019', '12/04/2019'],
      dtype=object)
```

EDA TEST DATA

In [68]:
```
test["journey_day"]=pd.to_datetime(test.Date_of_Journey,format="%d/%m/%Y").dt.day
```

In [69]:
```
test['journey_month']=pd.to_datetime(test.Date_of_Journey,format="%d/%m/%Y").dt.month
```

In [70]:
```
test.head()
```

Out[70]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stop
0	Jet Airways	6/06/2019	Delhi	Cochin	DEL → BOM → COK	17:30	04:25 07 Jun	10h 55m	1 stop
1	IndiGo	12/05/2019	Kolkata	Banglore	CCU → MAA → BLR	06:20	10:20	4h	1 stop
2	Jet Airways	21/05/2019	Delhi	Cochin	DEL → BOM → COK	19:15	19:00 22 May	23h 45m	1 stop
3	Multiple carriers	21/05/2019	Delhi	Cochin	DEL → BOM → COK	08:00	21:00	13h	1 stop
4	Air Asia	24/06/2019	Banglore	Delhi	BLR → DEL	23:55	02:45 25 Jun	2h 50m	non-stop

In [71]:
```
#As we have coverted date_of_journey column into integers so we are now goig to drop
test.drop(['Date_of_Journey'],axis=1,inplace=True)
```

In [72]:
```
test["Departure_hr"]=pd.to_datetime(test['Dep_Time']).dt.hour
```

In [73]:
```
test["Departure_min"]=pd.to_datetime(test['Dep_Time']).dt.minute
```

In [74]:
```
#As we have coverted Dep_Time column into integers so we are now going to drop it as
test.drop(['Dep_Time'],axis=1,inplace=True)
```

In [75]:
```
test.head()
```

Out[75]:

	Airline	Source	Destination	Route	Arrival_Time	Duration	Total_Stops	Additional_Info	journey_da
0	Jet Airways	Delhi	Cochin	DEL → BOM → COK	04:25 07 Jun	10h 55m	1 stop	No info	
1	IndiGo	Kolkata	Banglore	CCU → MAA → BLR	10:20	4h	1 stop	No info	
2	Jet Airways	Delhi	Cochin	DEL → BOM → COK	19:00 22 May	23h 45m	1 stop	In-flight meal not included	
3	Multiple carriers	Delhi	Cochin	DEL → BOM → COK	21:00	13h	1 stop	No info	
4	Air Asia	Banglore	Delhi	BLR → DEL	02:45 25 Jun	2h 50m	non-stop	No info	

In [76]:
```
test["Arrival_hr"]=pd.to_datetime(test['Arrival_Time']).dt.hour        #converted ar
test["Arrival_min"]=pd.to_datetime(test['Arrival_Time']).dt.minute     #converted ar
test.drop(['Arrival_Time'],axis=1,inplace=True)
```

In [77]:
```
test.head()
```

Out[77]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	journey_day	journey_mo
0	Jet Airways	Delhi	Cochin	DEL → BOM → COK	10h 55m	1 stop	No info	6	
1	IndiGo	Kolkata	Banglore	CCU → MAA → BLR	4h	1 stop	No info	12	
2	Jet Airways	Delhi	Cochin	DEL → BOM → COK	23h 45m	1 stop	In-flight meal not included	21	
3	Multiple carriers	Delhi	Cochin	DEL → BOM → COK	13h	1 stop	No info	21	
4	Air Asia	Banglore	Delhi	BLR → DEL	2h 50m	non-stop	No info	24	

In [78]:
```
# Assigning and converting Duration column into list
duration = list(test["Duration"])

for i in range(len(duration)):
    if len(duration[i].split()) != 2:                  # Check if duration contains
        if "h" in duration[i]:
            duration[i] = duration[i].strip() + " 0m"   # Adds 0 minute
        else:
            duration[i] = "0h " + duration[i]           # Adds 0 hour

duration_hours = []
duration_mins = []
for i in range(len(duration)):
    duration_hours.append(int(duration[i].split(sep = "h")[0]))              # Extrac
    duration_mins.append(int(duration[i].split(sep = "m")[0].split()[-1]))   # Extrac
```

In [79]:
```
test['duration_hours']=duration_hours
test['duration_mins']=duration_mins
```

In [80]:
```
test.head()
```

Out[80]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	journey_day	journey_mo
0	Jet Airways	Delhi	Cochin	DEL → BOM → COK	10h 55m	1 stop	No info	6	
1	IndiGo	Kolkata	Banglore	CCU → MAA → BLR	4h	1 stop	No info	12	
2	Jet Airways	Delhi	Cochin	DEL → BOM → COK	23h 45m	1 stop	In-flight meal not included	21	
3	Multiple carriers	Delhi	Cochin	DEL → BOM → COK	13h	1 stop	No info	21	
4	Air Asia	Banglore	Delhi	BLR → DEL	2h 50m	non-stop	No info	24	

In [81]:
```
test.drop(['Duration'],axis=1,inplace=True)
```

In [82]:
```
test.head()
```

Out[82]:

	Airline	Source	Destination	Route	Total_Stops	Additional_Info	journey_day	journey_month	Depa
0	Jet Airways	Delhi	Cochin	DEL → BOM → COK	1 stop	No info	6	6	
1	IndiGo	Kolkata	Banglore	CCU → MAA → BLR	1 stop	No info	12	5	
2	Jet Airways	Delhi	Cochin	DEL → BOM → COK	1 stop	In-flight meal not included	21	5	
3	Multiple carriers	Delhi	Cochin	DEL → BOM → COK	1 stop	No info	21	5	
4	Air Asia	Banglore	Delhi	BLR → DEL	non-stop	No info	24	6	

In [84]:
```
Airline=test[['Airline']]
Airline=pd.get_dummies(Airline,drop_first=True)
Airline.head()
```

Out[84]:

	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airli
0	0	0	0	1	0	0	0	
1	0	0	1	0	0	0	0	
2	0	0	0	1	0	0	0	
3	0	0	0	0	0	1	0	
4	0	0	0	0	0	0	0	

In [87]:
```
test['Source'].value_counts()
```

Out[87]:
```
Delhi       1145
Kolkata      710
Banglore     555
Mumbai       186
Chennai       75
Name: Source, dtype: int64
```

In [86]:
```
test['Source'].unique()
```

Out[86]:
```
Delhi       1145
Kolkata      710
Banglore     555
Mumbai       186
Chennai       75
Name: Source, dtype: int64
```

In [88]:
```
Source=test[['Source']]
Source=pd.get_dummies(Source,drop_first=True)
Source.head()
```

Out[88]:

	Source_Chennai	Source_Delhi	Source_Kolkata	Source_Mumbai
0	0	1	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	0	0
4	0	0	0	0

In [89]:
```
test['Destination'].value_counts()
```

Out[89]:
```
Cochin       1145
Banglore      710
Delhi         317
New Delhi     238
Hyderabad     186
Kolkata        75
Name: Destination, dtype: int64
```

In [90]:
```
Destination=test[['Destination']]
Destination=pd.get_dummies(Destination,drop_first=True)
Destination.head()
```

Out[90]:

	Destination_Cochin	Destination_Delhi	Destination_Hyderabad	Destination_Kolkata	Destination_New Delhi
0	1	0	0	0	0
1	0	0	0	0	0
2	1	0	0	0	0
3	1	0	0	0	0
4	0	1	0	0	0

In [91]:
```
test['Route']
```

Out[91]:
```
0       DEL → BOM → COK
1       CCU → MAA → BLR
2       DEL → BOM → COK
3       DEL → BOM → COK
4             BLR → DEL
             ...
2666    CCU → DEL → BLR
2667          CCU → BLR
2668    DEL → BOM → COK
2669    DEL → BOM → COK
2670    DEL → BOM → COK
Name: Route, Length: 2671, dtype: object
```

In [92]:
```
test.drop(['Route','Additional_Info'],axis=1,inplace=True)
```

In [93]:
```
test.head()
```

Out[93]:

	Airline	Source	Destination	Total_Stops	journey_day	journey_month	Departure_hr	Departure_min
0	Jet Airways	Delhi	Cochin	1 stop	6	6	17	30
1	IndiGo	Kolkata	Banglore	1 stop	12	5	6	20
2	Jet Airways	Delhi	Cochin	1 stop	21	5	19	15
3	Multiple carriers	Delhi	Cochin	1 stop	21	5	8	0
4	Air Asia	Banglore	Delhi	non-stop	24	6	23	55

In [94]:
```
test['Total_Stops'].value_counts()
```

Out[94]:
```
1 stop      1431
non-stop     849
2 stops      379
3 stops       11
4 stops        1
Name: Total_Stops, dtype: int64
```

In [95]:
```
test.replace({'non-stop':0,'1 stop':1,'2 stops':2,'3 stops':3,'4 stops':4},inplace=Tru
```

In [96]:
```
test.head()
```

Out[96]:

	Airline	Source	Destination	Total_Stops	journey_day	journey_month	Departure_hr	Departure_min
0	Jet Airways	Delhi	Cochin	1	6	6	17	30
1	IndiGo	Kolkata	Banglore	1	12	5	6	20
2	Jet Airways	Delhi	Cochin	1	21	5	19	15
3	Multiple carriers	Delhi	Cochin	1	21	5	8	0
4	Air Asia	Banglore	Delhi	0	24	6	23	55

In [97]:
```
data_test=pd.concat([test,Airline,Source,Destination],axis=1)
```

In [98]:
```
data_test.head()
```

Out[98]:

	Airline	Source	Destination	Total_Stops	journey_day	journey_month	Departure_hr	Departure_min
0	Jet Airways	Delhi	Cochin	1	6	6	17	30
1	IndiGo	Kolkata	Banglore	1	12	5	6	20
2	Jet Airways	Delhi	Cochin	1	21	5	19	15
3	Multiple carriers	Delhi	Cochin	1	21	5	8	0
4	Air Asia	Banglore	Delhi	0	24	6	23	55

5 rows × 31 columns

In [99]:
```
data_test.drop(['Airline','Source','Destination'],axis=1,inplace=True)
```

In [100…
```
data_test.head()
```

Out[100…

	Total_Stops	journey_day	journey_month	Departure_hr	Departure_min	Arrival_hr	Arrival_min	duratio
0	1	6	6	17	30	4	25	
1	1	12	5	6	20	10	20	
2	1	21	5	19	15	19	0	
3	1	21	5	8	0	21	0	
4	0	24	6	23	55	2	45	

5 rows × 28 columns

In [101…
```
data_test.shape
```

Out[101…
```
(2671, 28)
```

In [102…
```
data_train.shape
```

Out[102…
```
(10682, 30)
```

In [103…
```
data_train.columns
```

Out[103…
```
Index(['Total_Stops', 'Price', 'journey_day', 'journey_month', 'Departure_hr',
       'Departure_min', 'Arrival_hr', 'Arrival_min', 'duration_hours',
       'duration_mins', 'Airline_Air India', 'Airline_GoAir', 'Airline_IndiGo',
       'Airline_Jet Airways', 'Airline_Jet Airways Business',
       'Airline_Multiple carriers',
       'Airline_Multiple carriers Premium economy', 'Airline_SpiceJet',
       'Airline_Trujet', 'Airline_Vistara', 'Airline_Vistara Premium economy',
       'Source_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_Mumbai',
       'Destination_Cochin', 'Destination_Delhi', 'Destination_Hyderabad',
       'Destination_Kolkata', 'Destination_New Delhi'],
      dtype='object')
```

In [105…
```
X=data_train.loc[:,['Total_Stops', 'journey_day', 'journey_month', 'Departure_hr',
       'Departure_min', 'Arrival_hr', 'Arrival_min', 'duration_hours',
       'duration_mins', 'Airline_Air India', 'Airline_GoAir', 'Airline_IndiGo',
       'Airline_Jet Airways', 'Airline_Jet Airways Business',
       'Airline_Multiple carriers',
       'Airline_Multiple carriers Premium economy', 'Airline_SpiceJet',
       'Airline_Trujet', 'Airline_Vistara', 'Airline_Vistara Premium economy',
       'Source_Chennai', 'Source_Delhi', 'Source_Kolkata', 'Source_Mumbai',
       'Destination_Cochin', 'Destination_Delhi', 'Destination_Hyderabad',
       'Destination_Kolkata', 'Destination_New Delhi']]
X.head()
```

Out[105…

	Total_Stops	journey_day	journey_month	Departure_hr	Departure_min	Arrival_hr	Arrival_min	duratio
0	0	24	3	22	20	1	10	
1	2	1	5	5	50	13	15	
2	2	9	6	9	25	4	25	
3	1	12	5	18	5	23	30	
4	1	1	3	16	50	21	35	

5 rows × 29 columns

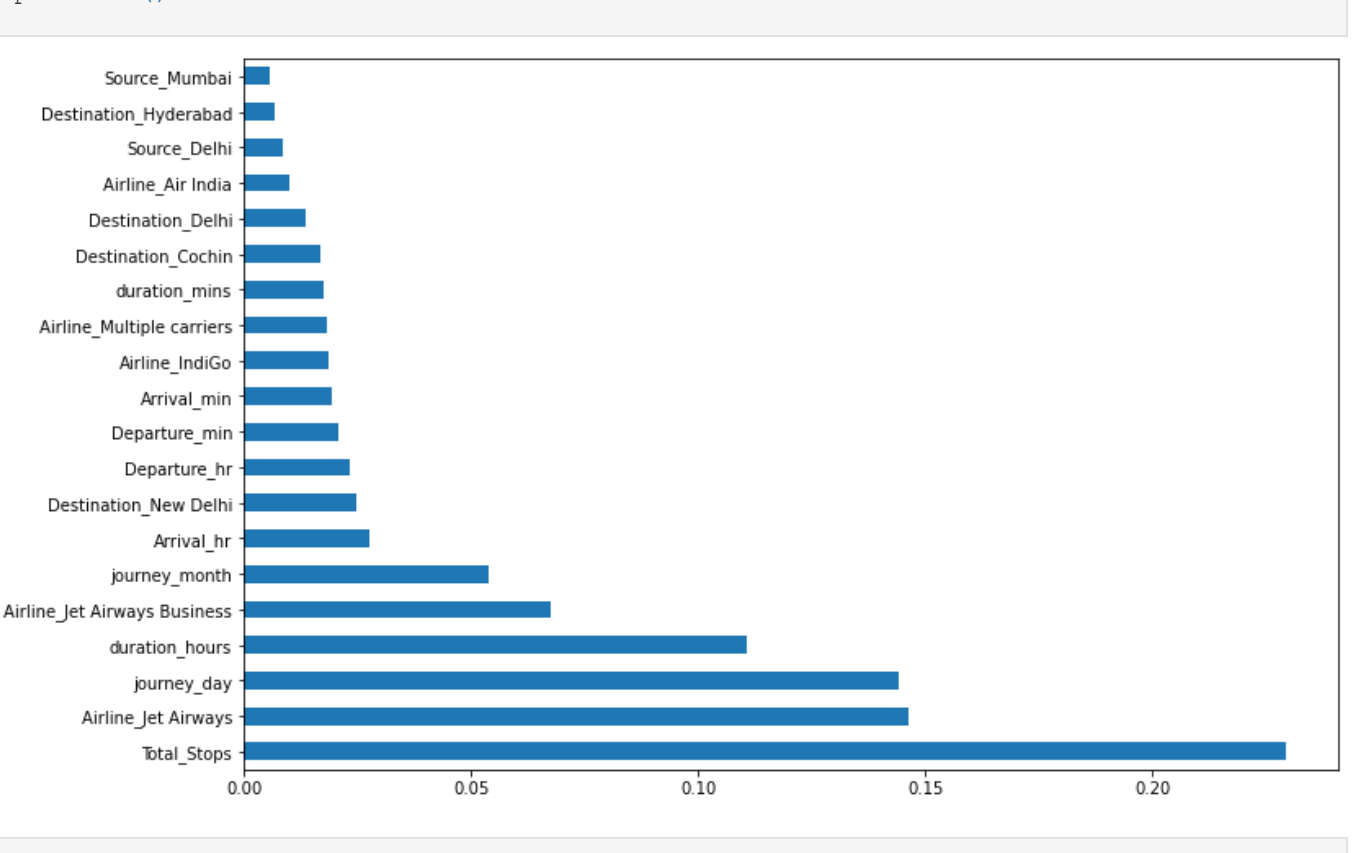
In [107…
```
y=data_train.iloc[:,1]
y.head()
```

Out[107…
```
0     3897
1     7662
2    13882
3     6218
4    13302
Name: Price, dtype: int64
```

In [108…
```
from sklearn.ensemble import ExtraTreesRegressor
select=ExtraTreesRegressor()
select.fit(X,y)
```

Out[108…
```
ExtraTreesRegressor()
```

In [111…
```
#plot graph for feature improtance for better visualization
plt.figure(figsize=(12,8))
feature_imp=pd.Series(select.feature_importances_,index=X.columns)
feature_imp.nlargest(20).plot(kind='barh')
plt.show()
```



In [184…
```
from sklearn.model_selection import train_test_split
Xtrain,Xtest,ytrain,ytest=train_test_split(X,y,test_size=0.2,random_state=42)
```

In [185…
```
from sklearn.ensemble import RandomForestRegressor
rfr=RandomForestRegressor()
rfr.fit(Xtrain,ytrain)
```

Out[185…
```
RandomForestRegressor()
```

In [186…
```
y_pred=rfr.predict(Xtest)
y_pred
```

Out[186…
```
array([16901.46      ,  5412.28      ,  8885.34      , ...,
        6593.63      , 12771.25733333, 13025.40588095])
```

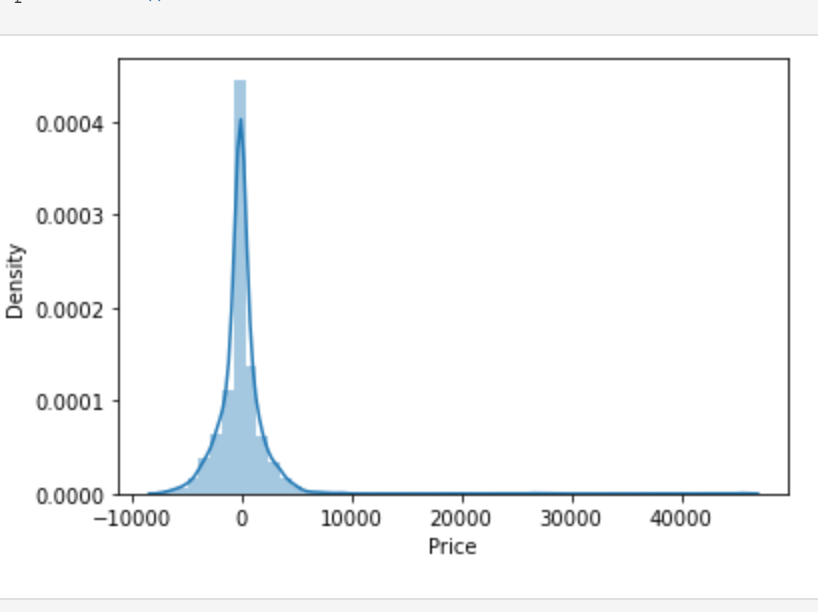
In [187…
```
rfr.score(Xtrain,ytrain)
```

Out[187…
```
0.9536482720994266
```

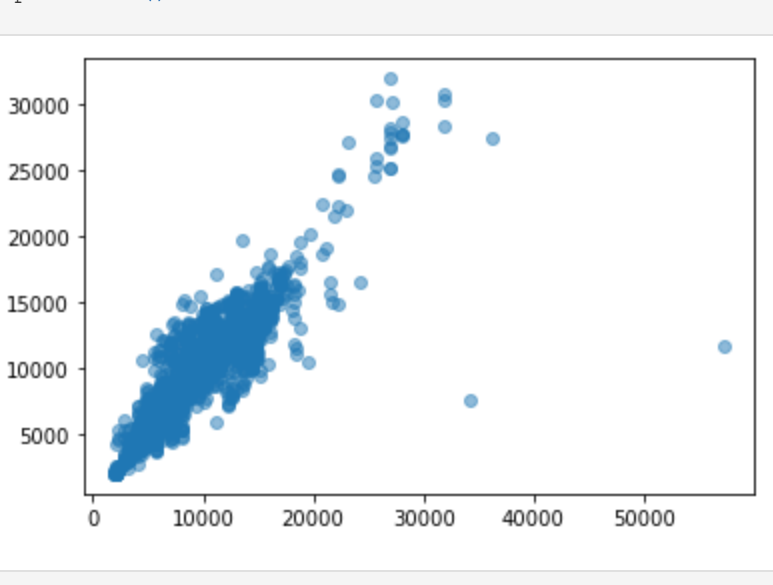
In [188…
```
rfr.score(Xtest,ytest)
```

Out[188…
```
0.797266194635126
```

In [189…
```
sns.distplot(ytest-y_pred)
plt.show()
```



In [190…
```
plt.scatter(ytest,y_pred,alpha=0.5)
plt.show()
```



In [120…
```
from sklearn import metrics
```

In [124…
```
print('MAE:',metrics.mean_absolute_error(ytest,y_pred))
print('MSE:',metrics.mean_squared_error(ytest,y_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(ytest,y_pred)))
```

```
MAE: 1280.9632984987247
MSE: 5452391.279202733
RMSE: 2335.0356055535285
```

In [125…
```
#hyperparameter tuning
from sklearn.model_selection import RandomizedSearchCV
```

In [191…
```
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1200, num = 12)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(5, 30, num = 6)]
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10, 15, 100]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 5, 10]
```

In [192…
```
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf}
```

In [193…
```
rf_random=RandomizedSearchCV(estimator=rfr,param_distributions=random_grid,scoring='ne
```

In [195…
```
rf_random.fit(Xtrain,ytrain)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
[CV] END max_depth=10, max_features=sqrt, min_samples_leaf=5, min_samples_split=5, n_e
stimators=900; total time=   4.8s
[CV] END max_depth=10, max_features=sqrt, min_samples_leaf=5, min_samples_split=5, n_e
stimators=900; total time=   4.4s
[CV] END max_depth=10, max_features=sqrt, min_samples_leaf=5, min_samples_split=5, n_e
stimators=900; total time=   5.0s
[CV] END max_depth=10, max_features=sqrt, min_samples_leaf=5, min_samples_split=5, n_e
stimators=900; total time=   4.5s
[CV] END max_depth=10, max_features=sqrt, min_samples_leaf=5, min_samples_split=5, n_e
stimators=900; total time=   4.5s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_
estimators=1100; total time=   6.6s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_
estimators=1100; total time=   8.4s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_
estimators=1100; total time=   7.5s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_
estimators=1100; total time=   7.7s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_
estimators=1100; total time=   7.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=100, n
_estimators=300; total time=   4.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=100, n
_estimators=300; total time=   4.2s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=100, n
_estimators=300; total time=   4.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=100, n
_estimators=300; total time=   3.8s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=100, n
_estimators=300; total time=   4.0s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=5, n_e
stimators=400; total time=   7.1s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=5, n_e
stimators=400; total time=   6.9s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=5, n_e
stimators=400; total time=   7.5s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=5, n_e
stimators=400; total time=   7.5s
[CV] END max_depth=15, max_features=auto, min_samples_leaf=5, min_samples_split=5, n_e
stimators=400; total time=   7.2s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=10, min_samples_split=5, n_
estimators=700; total time=  11.3s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=10, min_samples_split=5, n_
estimators=700; total time=  14.3s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=10, min_samples_split=5, n_
estimators=700; total time=  12.8s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=10, min_samples_split=5, n_
estimators=700; total time=  13.3s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=10, min_samples_split=5, n_
estimators=700; total time=  13.3s
[CV] END max_depth=25, max_features=sqrt, min_samples_leaf=1, min_samples_split=2, n_e
stimators=1000; total time=  14.7s
[CV] END max_depth=25, max_features=sqrt, min_samples_leaf=1, min_samples_split=2, n_e
stimators=1000; total time=  15.4s
[CV] END max_depth=25, max_features=sqrt, min_samples_leaf=1, min_samples_split=2, n_e
stimators=1000; total time=  14.8s
[CV] END max_depth=25, max_features=sqrt, min_samples_leaf=1, min_samples_split=2, n_e
stimators=1000; total time=  12.2s
[CV] END max_depth=25, max_features=sqrt, min_samples_leaf=1, min_samples_split=2, n_e
stimators=1000; total time=  13.6s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=10, min_samples_split=15, n_
estimators=1100; total time=   5.2s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=10, min_samples_split=15, n_
estimators=1100; total time=   5.8s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=10, min_samples_split=15, n_
estimators=1100; total time=   5.7s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=10, min_samples_split=15, n_
estimators=1100; total time=   5.7s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=10, min_samples_split=15, n_
estimators=1100; total time=   5.4s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=1, min_samples_split=15, n_
estimators=300; total time=   2.5s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=1, min_samples_split=15, n_
estimators=300; total time=   2.7s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=1, min_samples_split=15, n_
estimators=300; total time=   2.8s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=1, min_samples_split=15, n_
estimators=300; total time=   2.4s
[CV] END max_depth=15, max_features=sqrt, min_samples_leaf=1, min_samples_split=15, n_
estimators=300; total time=   2.4s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_e
stimators=700; total time=   3.4s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_e
stimators=700; total time=   3.1s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_e
stimators=700; total time=   3.1s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_e
stimators=700; total time=   3.0s
[CV] END max_depth=5, max_features=sqrt, min_samples_leaf=2, min_samples_split=10, n_e
stimators=700; total time=   3.3s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=1, min_samples_split=15, n_
estimators=700; total time=  17.8s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=1, min_samples_split=15, n_
estimators=700; total time=  20.0s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=1, min_samples_split=15, n_
estimators=700; total time=  16.3s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=1, min_samples_split=15, n_
estimators=700; total time=  16.7s
[CV] END max_depth=20, max_features=auto, min_samples_leaf=1, min_samples_split=15, n_
estimators=700; total time=  18.8s
```

Out[195…
```
RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(), n_jobs=1,
                   param_distributions={'max_depth': [5, 10, 15, 20, 25, 30],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 2, 5, 10],
                                        'min_samples_split': [2, 5, 10, 15,
                                                              100],
                                        'n_estimators': [100, 200, 300, 400,
                                                         500, 600, 700, 800,
                                                         900, 1000, 1100,
                                                         1200]},
                   random_state=42, scoring='neg_mean_squared_error',
                   verbose=2)
```

In [169…
```
rf_random.best_params
```

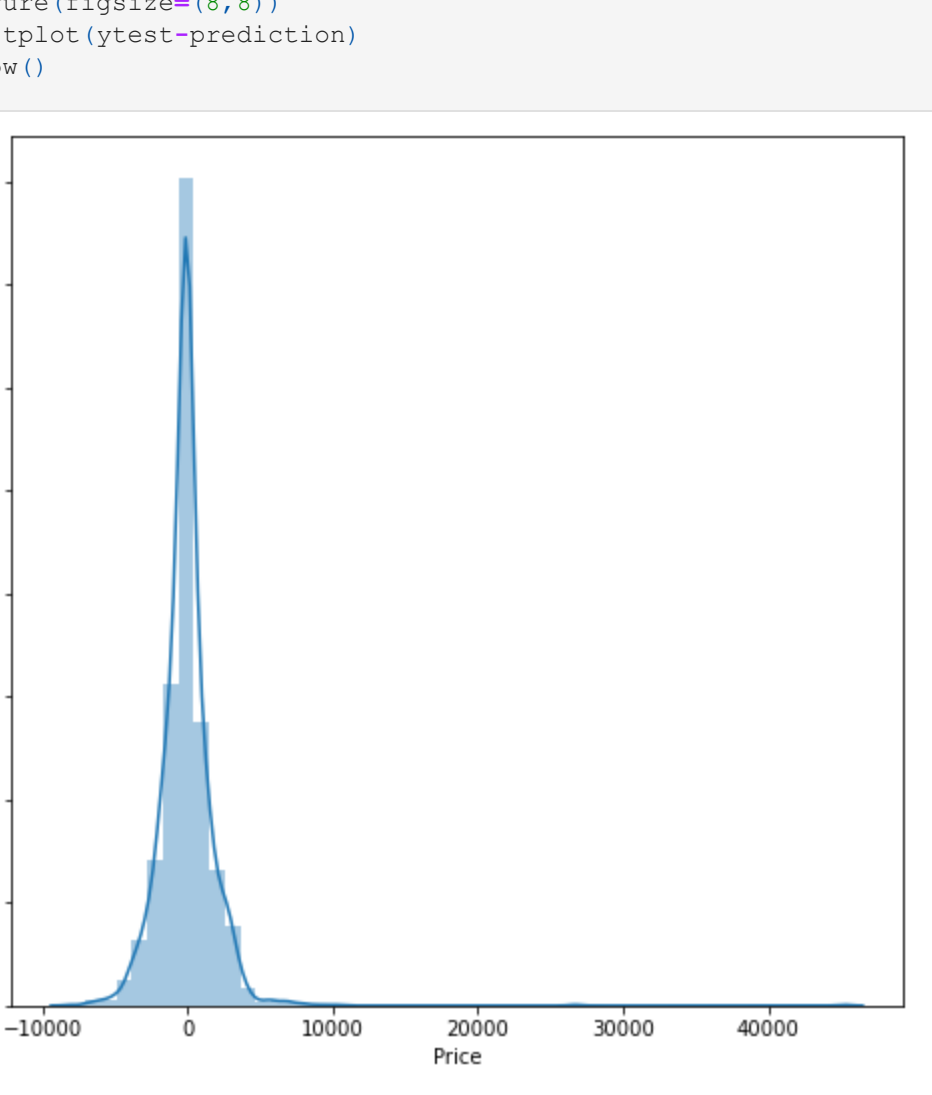
Out[169…
```
dict_keys(['bootstrap', 'ccp_alpha', 'criterion', 'max_depth', 'max_features', 'max_le
af_nodes', 'max_samples', 'min_impurity_decrease', 'min_samples_leaf', 'min_samples_sp
lit', 'min_weight_fraction_leaf', 'n_estimators', 'n_jobs', 'oob_score', 'random_stat
e', 'verbose', 'warm_start'])
```

In [197…
```
rf_random.best_params_
```

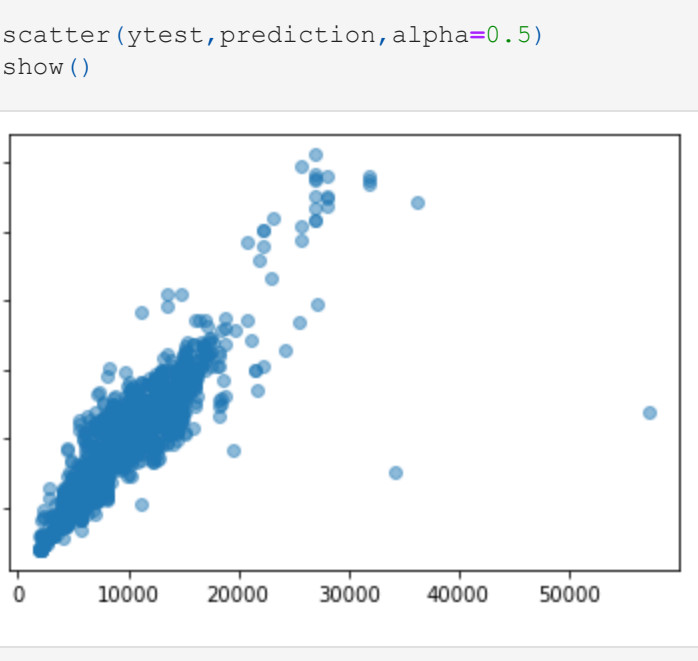
Out[197…
```
{'n_estimators': 700,
 'min_samples_split': 15,
 'min_samples_leaf': 1,
 'max_features': 'auto',
 'max_depth': 20}
```

In [198…
```
prediction=rf_random.predict(Xtest)
```

In [200…
```
plt.figure(figsize=(8,8))
sns.distplot(ytest-prediction)
plt.show()
```



In [201…
```
plt.scatter(ytest,prediction,alpha=0.5)
plt.show()
```



In [202…
```
print('MAE:',metrics.mean_absolute_error(ytest,prediction))
print('MSE:',metrics.mean_squared_error(ytest,prediction))
print('RMSE:',np.sqrt(metrics.mean_squared_error(ytest,prediction)))
```

```
MAE: 1166.4167956715494
MSE: 4054929.6104338802
RMSE: 2013.6855788414139
```

Save a model

In [203…
```
import pickle
#open a file where you want to store the data
file=open('flight_rf.pkl','wb')
#dump information to that file
pickle.dump(rfr,file)
```

In [205…
```
model=open('flight_rf.pkl','rb')
forest=pickle.load(model)
```

In [207…
```
y_prediction = forest.predict(Xtest)
```

In [208…
```
metrics.r2_score(ytest,y_prediction)
```

Out[208…
```
0.797266194635126
```

In [ ]: