Optimizing Flight Booking Decisions through Machine Learning Price Predictions

1.INTRODUCTION:

1.1 OVERVIEW:

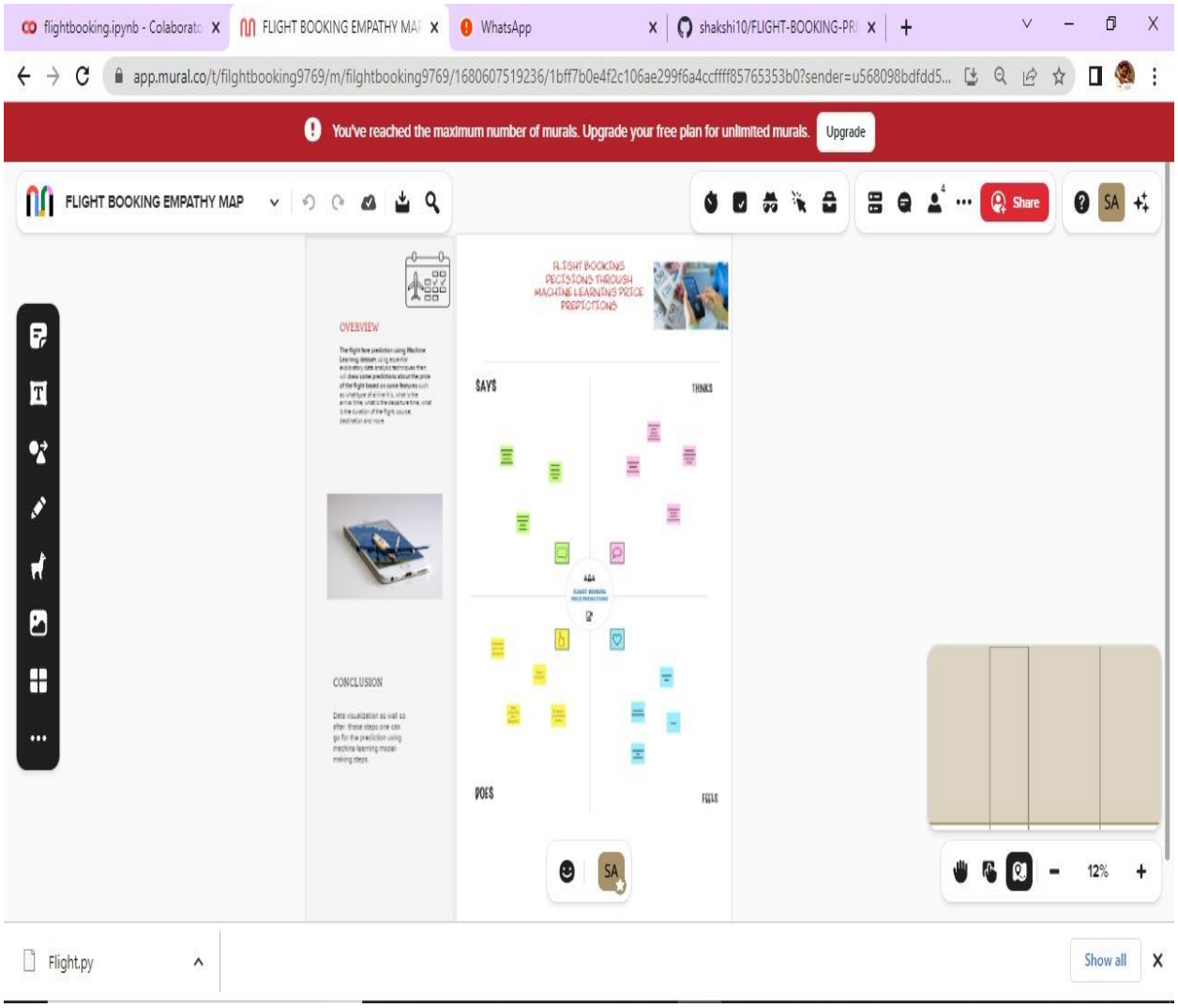
In this article, we will be analyzing the flight fare prediction using Machine Learning dataset using essential exploratory data analysis techniques then will draw some predictions about the price of the flight based on some features such as what type of airline it is, what is the arrival time, what is the departure time, what is the duration of the flight, source, destination and more.

1.2 PURPOSE

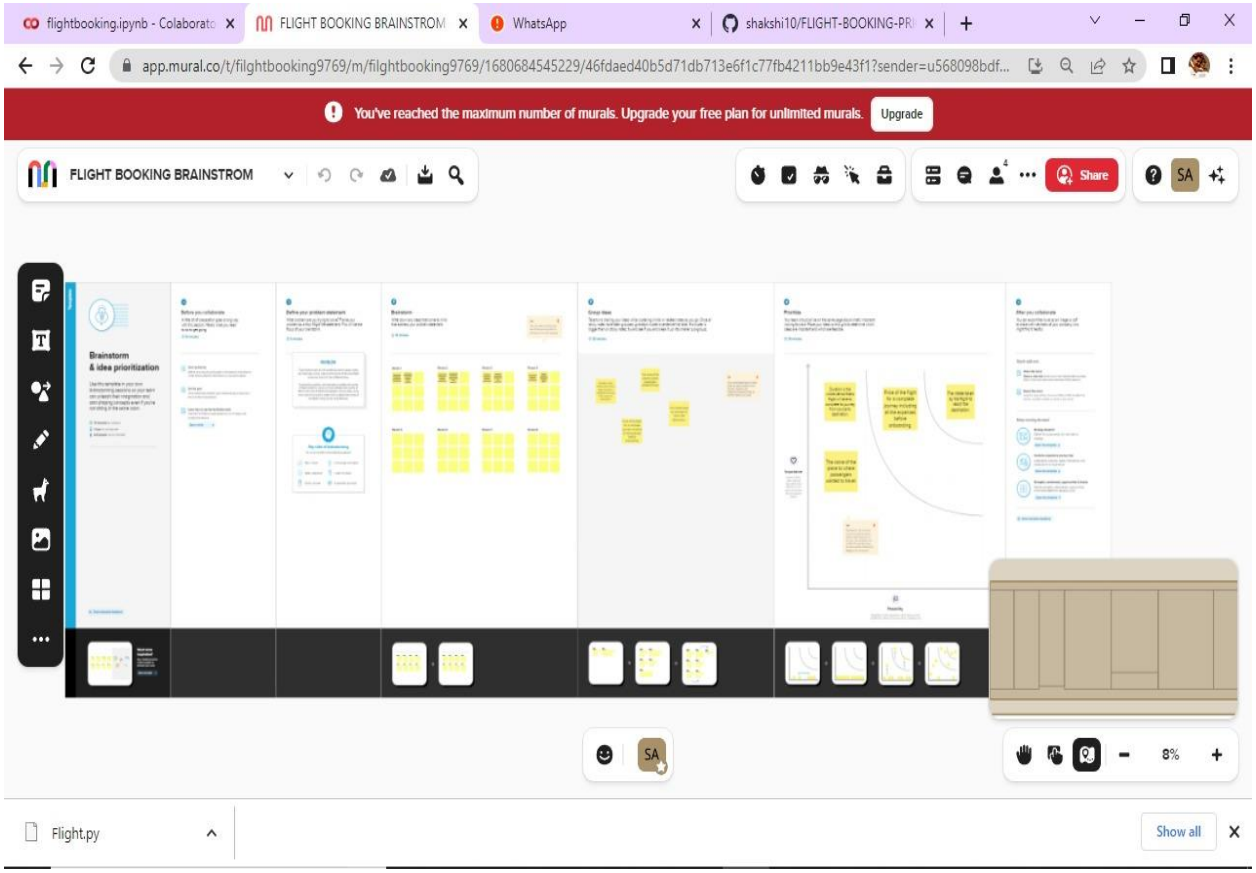
Optimizing Flight Booking Decisions through Machine Learning Price Predictions to provide convenient to passenger.

2. Problem Definition & Design Thinking

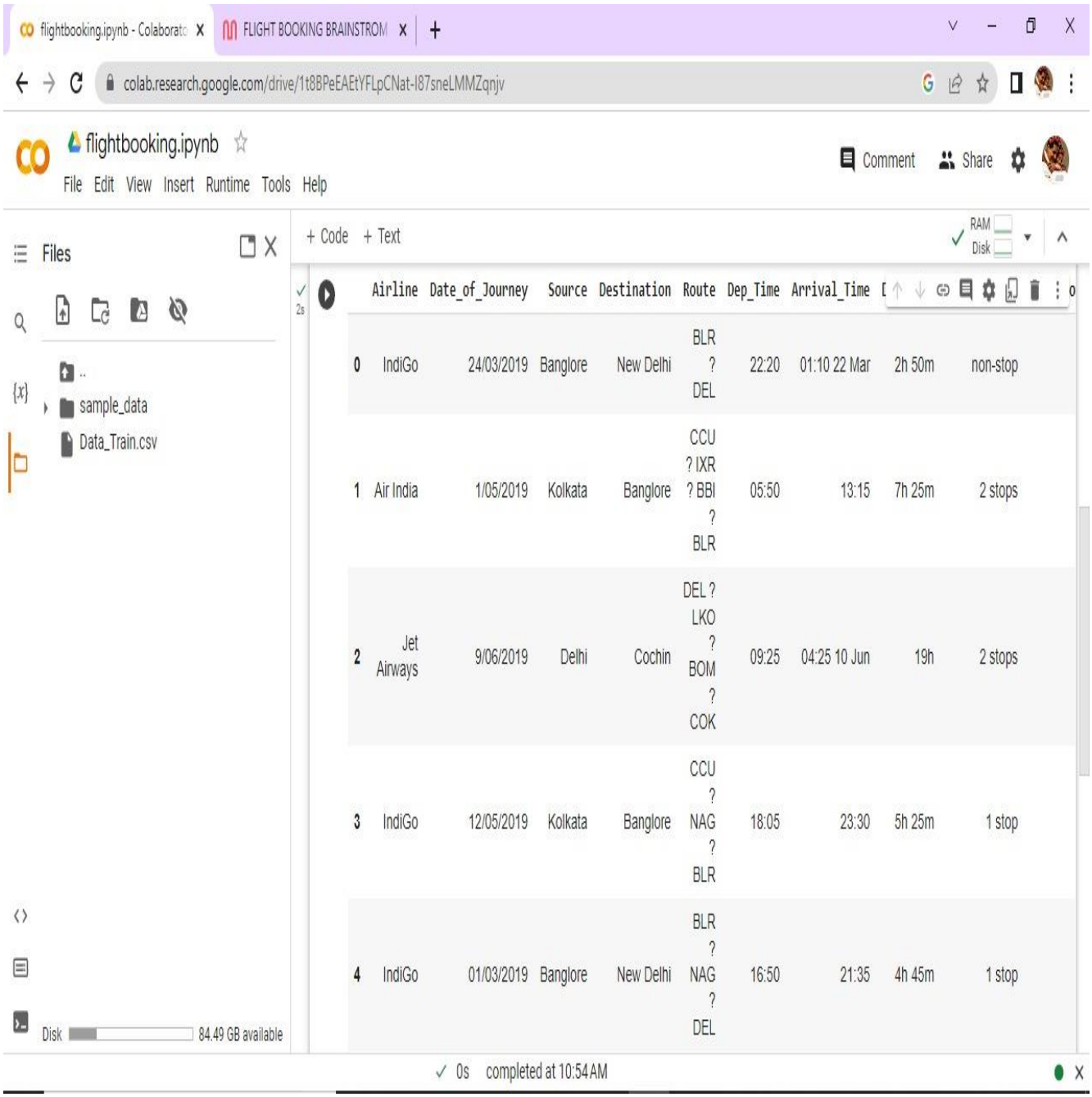
2.1 Empathy Map:



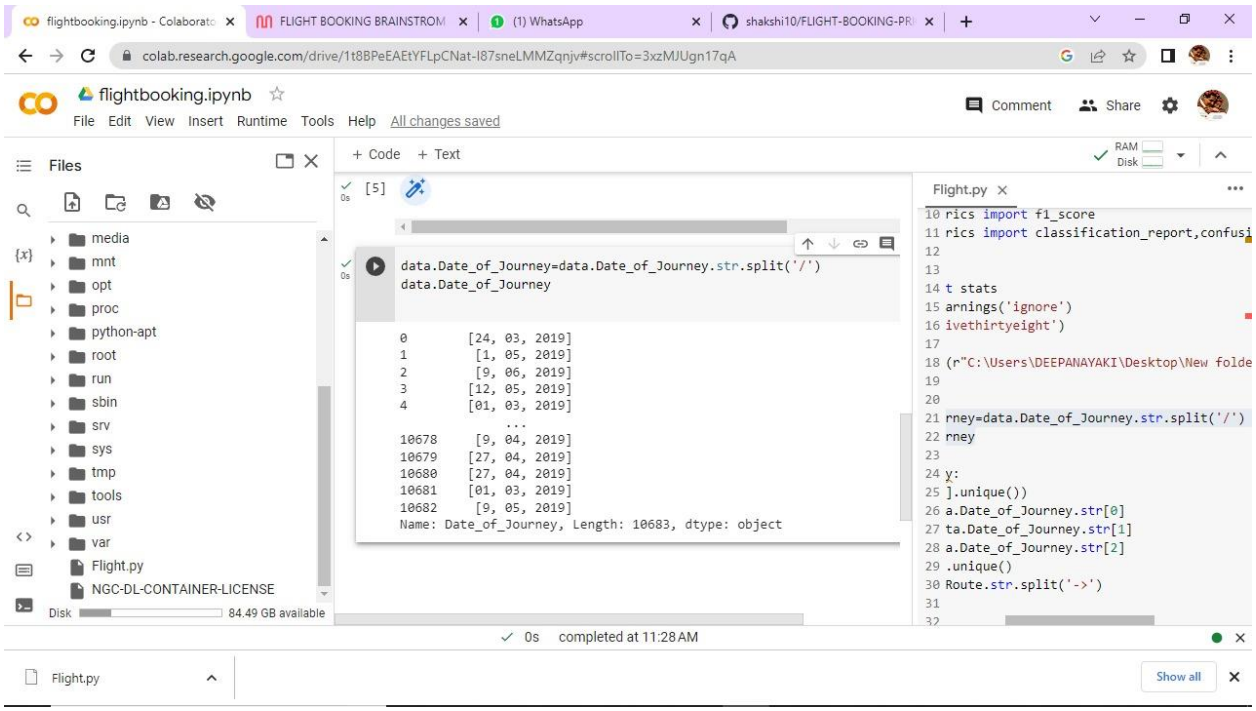
2.2 Ideation & Brainstorming Map:

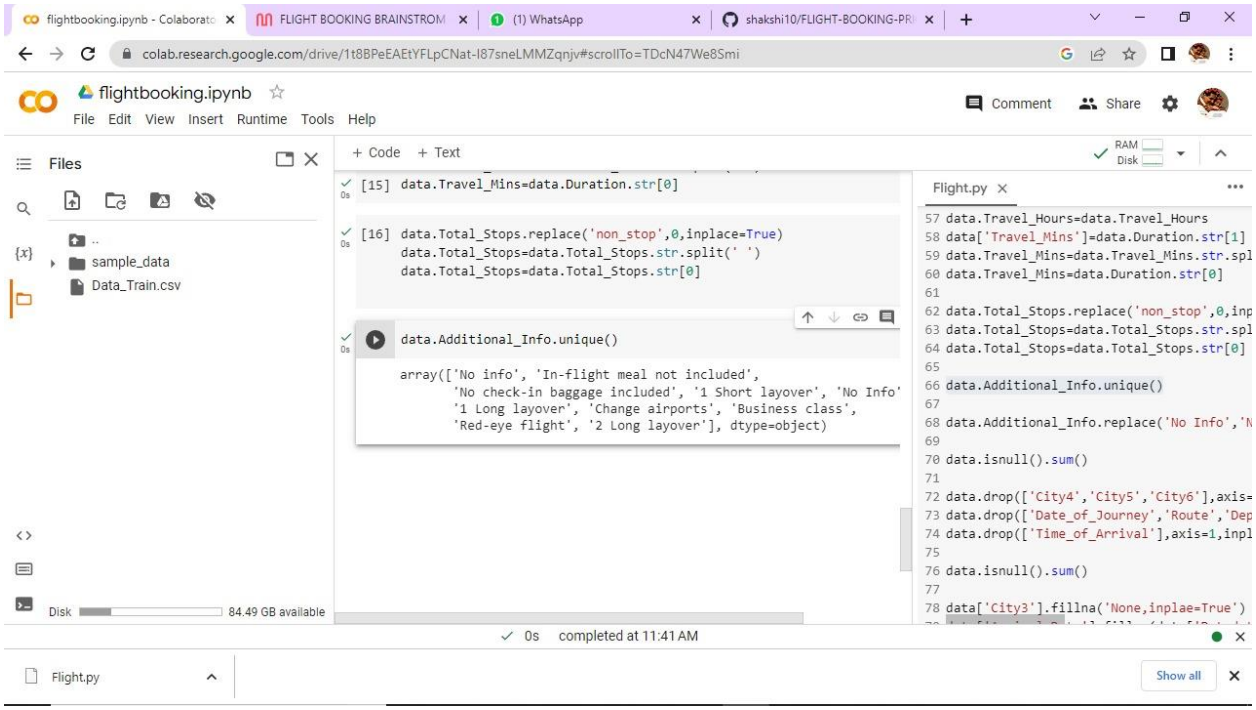


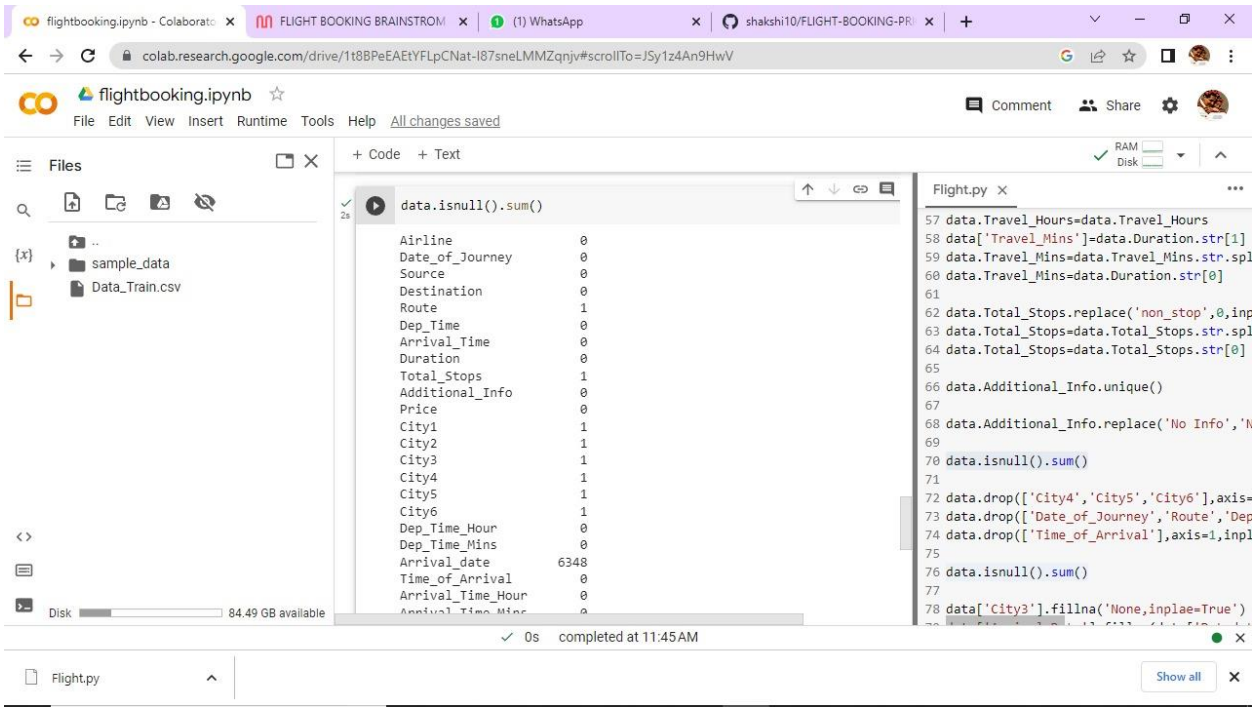
3.**RESULT:**

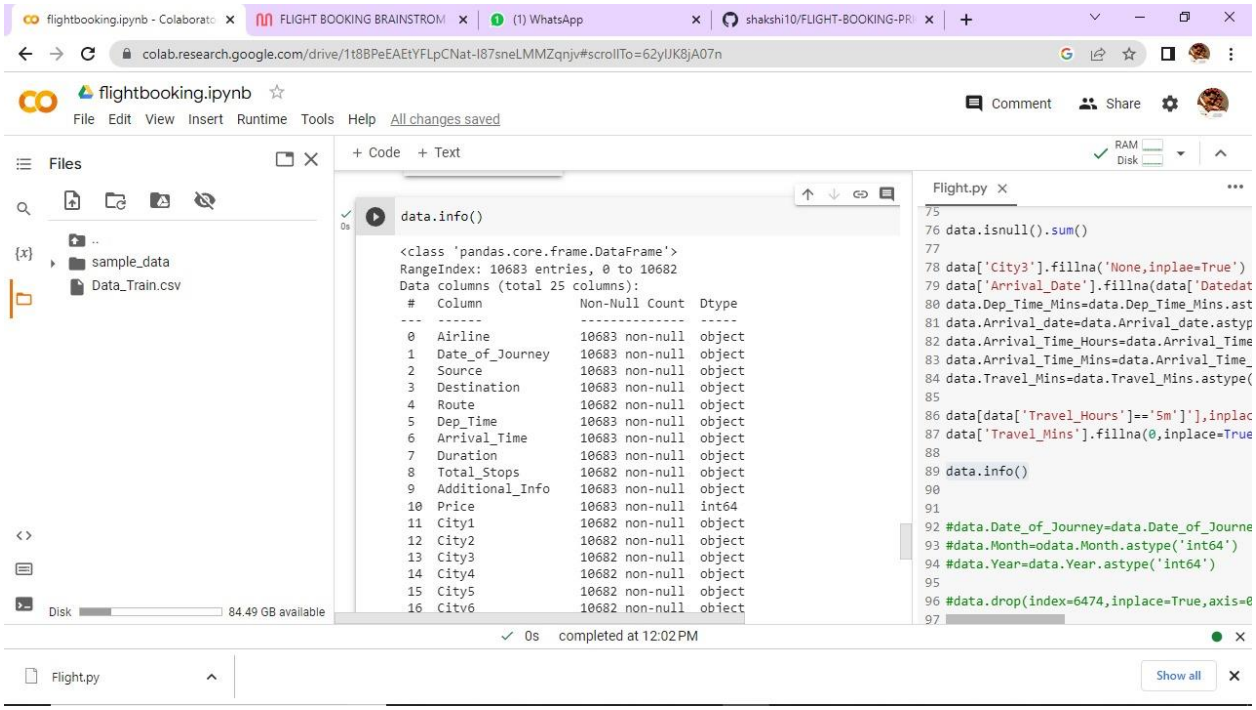


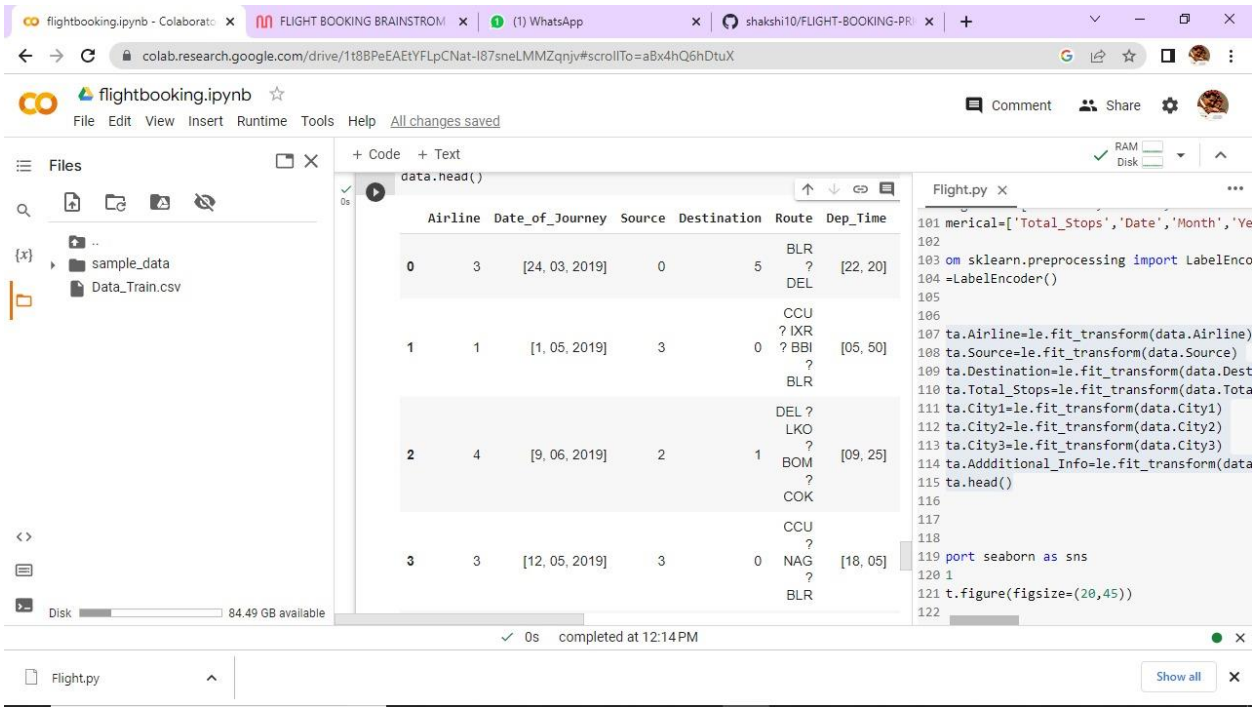
	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h 50m	non-stop
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m	2 stops
2	Jet Airways	9/06/2019	Delhi	Cochin	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

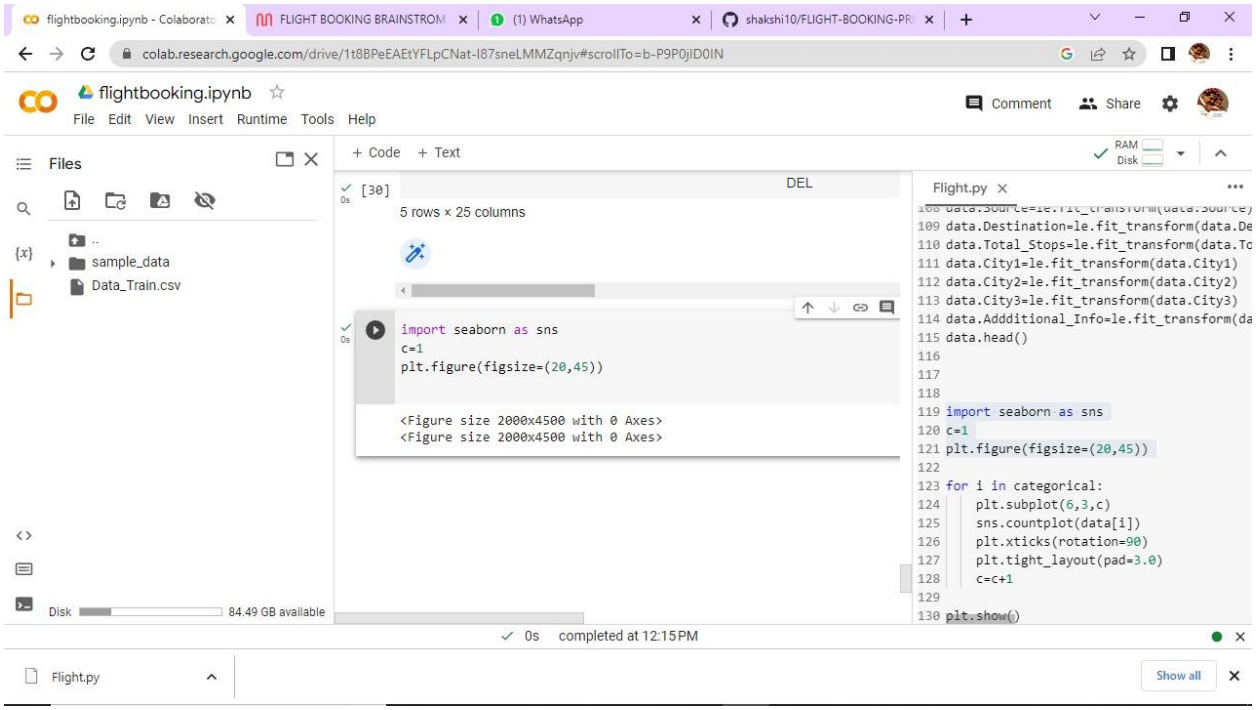

flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=3xzMJUgn17qA
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
media
mnt
opt
proc
python-apt
root
run
sbin
srv
sys
tmp
tools
usr
var
Flight.py
NGC-DL-CONTAINER-LICENSE
Disk 84.49 GB available
+ Code + Text
RAM
Disk
[5]
data.Date_of_Journey=data.Date_of_Journey.str.split('/')
data.Date_of_Journey
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: 10683, dtype: object
Flight.py
10 rics import f1_score
11 rics import classification_report,confusi
12
13
14 t stats
15 arnings('ignore')
16 ivethirtyeight')
17
18 (r"C:\Users\DEEPANAYAKI\Desktop\New folde
19
20
21 rney=data.Date_of_Journey.str.split('/')
22 rney
23
24 y:
25].unique())
26 a.Date_of_Journey.str[0]
27 ta.Date_of_Journey.str[1]
28 a.Date_of_Journey.str[2]
29 .unique()
30 Route.str.split('->')
31
32
0s completed at 11:28 AM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=TDcN47We8Smi
flightbooking.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
..
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
[15] data.Travel_Mins=data.Duration.str[0]
[16] data.Total_Stops.replace('non_stop',0,inplace=True)
data.Total_Stops=data.Total_Stops.str.split(' ')
data.Total_Stops=data.Total_Stops.str[0]
data.Additional_Info.unique()
array(['No info', 'In-flight meal not included',
'No check-in baggage included', '1 Short layover', 'No Info'
'1 Long layover', 'Change airports', 'Business class',
'Red-eye flight', '2 Long layover'], dtype=object)
Flight.py
57 data.Travel_Hours=data.Travel_Hours
58 data['Travel_Mins']=data.Duration.str[1]
59 data.Travel_Mins=data.Travel_Mins.str.spl
60 data.Travel_Mins=data.Duration.str[0]
61
62 data.Total_Stops.replace('non_stop',0,inp
63 data.Total_Stops=data.Total_Stops.str.spl
64 data.Total_Stops=data.Total_Stops.str[0]
65
66 data.Additional_Info.unique()
67
68 data.Additional_Info.replace('No Info','N
69
70 data.isnull().sum()
71
72 data.drop(['City4','City5','City6'],axis=
73 data.drop(['Date_of_Journey','Route','Dep
74 data.drop(['Time_of_Arrival'],axis=1,inpl
75
76 data.isnull().sum()
77
78 data['City3'].fillna('None,inplae=True')
0s completed at 11:41 AM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=JSy1z4An9HwV
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
RAM
Disk
sample_data
Data_Train.csv
data.isnull().sum()
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
City1 1
City2 1
City3 1
City4 1
City5 1
City6 1
Dep_Time_Hour 0
Dep_Time_Mins 0
Arrival_date 6348
Time_of_Arrival 0
Arrival_Time_Hour 0
Flight.py
data.Travel_Hours=data.Travel_Hours
data['Travel_Mins']=data.Duration.str[1]
data.Travel_Mins=data.Travel_Mins.str.spl
data.Travel_Mins=data.Duration.str[0]
data.Total_Stops.replace('non_stop',0,inp
data.Total_Stops=data.Total_Stops.str.spl
data.Total_Stops=data.Total_Stops.str[0]
data.Additional_Info.unique()
data.Additional_Info.replace('No Info','N
data.isnull().sum()
data.drop(['City4','City5','City6'],axis=
data.drop(['Date_of_Journey','Route','Dep
data.drop(['Time_of_Arrival'],axis=1,inpl
data.isnull().sum()
data['City3'].fillna('None,inplae=True')
Disk 84.49 GB available
0s completed at 11:45 AM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=62yIJK8jA07n
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
RAM
Disk
sample_data
Data_Train.csv
data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 25 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
11 City1 10682 non-null object
12 City2 10682 non-null object
13 City3 10682 non-null object
14 City4 10682 non-null object
15 City5 10682 non-null object
16 City6 10682 non-null object
Flight.py
data.isnull().sum()
data['City3'].fillna('None,inplae=True')
data['Arrival_Date'].fillna(data['Datedat
data.Dep_Time_Mins=data.Dep_Time_Mins.ast
data.Arrival_date=data.Arrival_date.astyp
data.Arrival_Time_Hours=data.Arrival_Time
data.Arrival_Time_Mins=data.Arrival_Time_
data.Travel_Mins=data.Travel_Mins.astype(
data[data['Travel_Hours']=='5m']],inplac
data['Travel_Mins'].fillna(0,inplace=True
data.info()
#data.Date_of_Journey=data.Date_of_Journe
#data.Month=odata.Month.astype('int64')
#data.Year=data.Year.astype('int64')
#data.drop(index=6474,inplace=True,axis=0
Disk 84.49 GB available
0s completed at 12:02 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=aBx4hQ6hDtuX
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
data.head()
Airline Date_of_Journey Source Destination Route Dep_Time
0 3 [24, 03, 2019] 0 5 BLR ? DEL [22, 20]
1 1 [1, 05, 2019] 3 0 CCU ? IXR ? BBI ? BLR [05, 50]
2 4 [9, 06, 2019] 2 1 DEL ? LKO ? BOM ? COK [09, 25]
3 3 [12, 05, 2019] 3 0 CCU ? NAG ? BLR [18, 05]
Flight.py
merical=['Total_Stops','Date','Month','Ye
om sklearn.preprocessing import LabelEnco
=LabelEncoder()
ta.Airline=le.fit_transform(data.Airline)
ta.Source=le.fit_transform(data.Source)
ta.Destination=le.fit_transform(data.Dest
ta.Total_Stops=le.fit_transform(data.Tota
ta.City1=le.fit_transform(data.City1)
ta.City2=le.fit_transform(data.City2)
ta.City3=le.fit_transform(data.City3)
ta.Addditional_Info=le.fit_transform(data
ta.head()
port seaborn as sns
1
t.figure(figsize=(20,45))
0s completed at 12:14 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=b-P9P0jlD0IN
flightbooking.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
[30]
DEL
5 rows × 25 columns
import seaborn as sns
c=1
plt.figure(figsize=(20,45))
<Figure size 2000x4500 with 0 Axes>
<Figure size 2000x4500 with 0 Axes>
Flight.py
data.Destination=le.fit_transform(data.De
data.Total_Stops=le.fit_transform(data.To
data.City1=le.fit_transform(data.City1)
data.City2=le.fit_transform(data.City2)
data.City3=le.fit_transform(data.City3)
data.Addditional_Info=le.fit_transform(da
data.head()
import seaborn as sns
c=1
plt.figure(figsize=(20,45))
for i in categorical:
plt.subplot(6,3,c)
sns.countplot(data[i])
plt.xticks(rotation=90)
plt.tight_layout(pad=3.0)
c=c+1
plt.show()
0s completed at 12:15 PM
Flight.py
Show all

flightbooking.ipynb - Colaborato | FLIGHT BOOKING BRAINSTROM | (1) WhatsApp | shakshi10/FLIGHT-BOOKING-PR

colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=e6g9Jw6nEUPU

flightbooking.ipynb

File Edit View Insert Runtime Tools Help Saving...

Comment Share

Files

..
sample_data
Data_Train.csv

Disk 84.49 GB available

+ Code + Text

RAM Disk

```
data['Date']=data.Date_of_Journey.str[0]
data['Month']=data.Date_of_Journey.str[1]
data['Year']=data.Date_of_Journey.str[2]
data.Total_Stops.unique()
data.Route=data.Route.str.split('->')
data.Route
```

```
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: 10683, dtype: object
```

```
[10] data['City1']=data.Route.str[0]
     data['City2']=data.Route.str[1]
```

Flight.py

```
109 data.Destination=le.fit_transform(data.De
110 data.Total_Stops=le.fit_transform(data.To
111 data.City1=le.fit_transform(data.City1)
112 data.City2=le.fit_transform(data.City2)
113 data.City3=le.fit_transform(data.City3)
114 data.Addditional_Info=le.fit_transform(da
115 data.head()
116
117
118
119 import seaborn as sns
120 c=1
121 plt.figure(figsize=(20,45))
122
123 for i in categorical:
124     plt.subplot(6,3,c)
125     sns.countplot(data[i])
126     plt.xticks(rotation=90)
127     plt.tight_layout(pad=3.0)
128     c=c+1
129
130 plt.show()
```

0s completed at 12:17 PM

Flight.py Show all

flightbooking.ipynb - Colaborato | FLIGHT BOOKING BRAINSTROM | (1) WhatsApp | shakshi10/FLIGHT-BOOKING-PR

colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=e6g9Jw6nEUPU

flightbooking.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Files

..
sample_data
Data_Train.csv

Disk 84.49 GB available

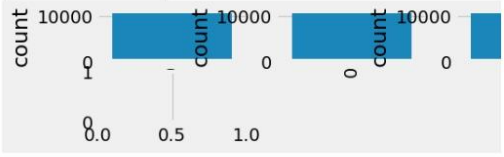
+ Code + Text

RAM Disk

```
[32]
    870                 dtype='datetime64[ns]')
    871         """
--> 872         return np.asarray(self._values, dtype)
    873
    874     # ----------------------------------------------------
---

ValueError: could not convert string to float: 'No info'
```

SEARCH STACK OVERFLOW

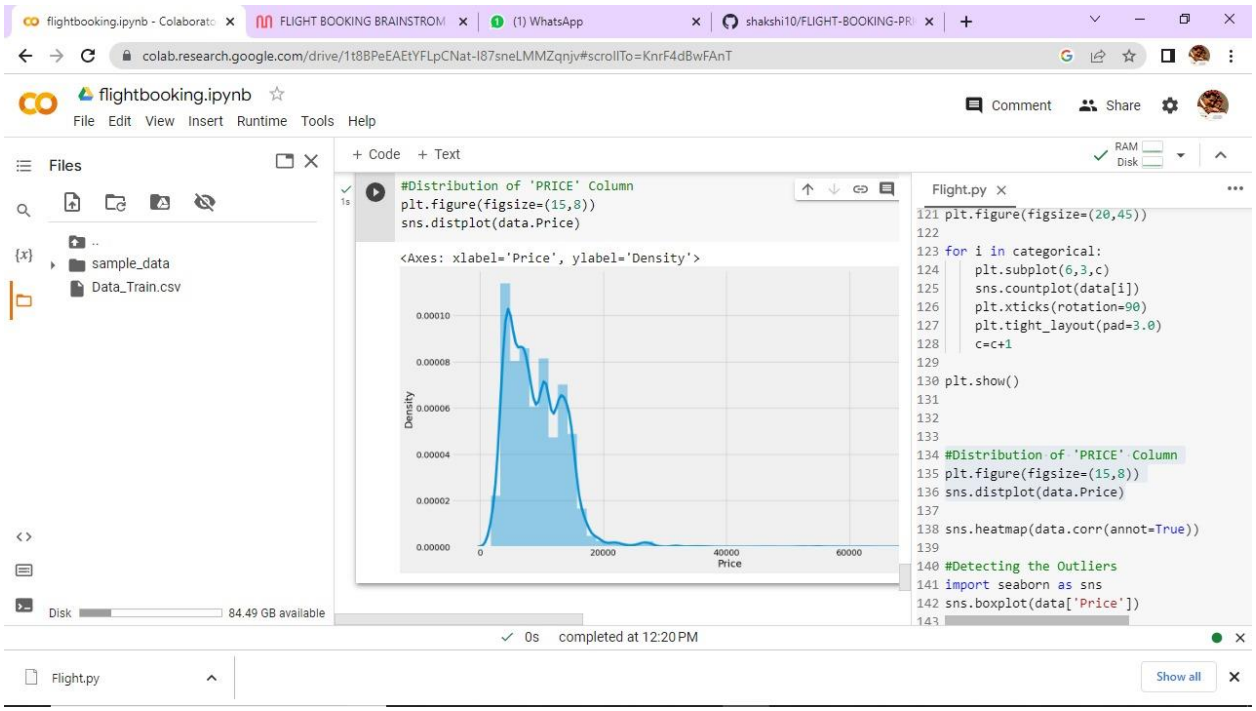


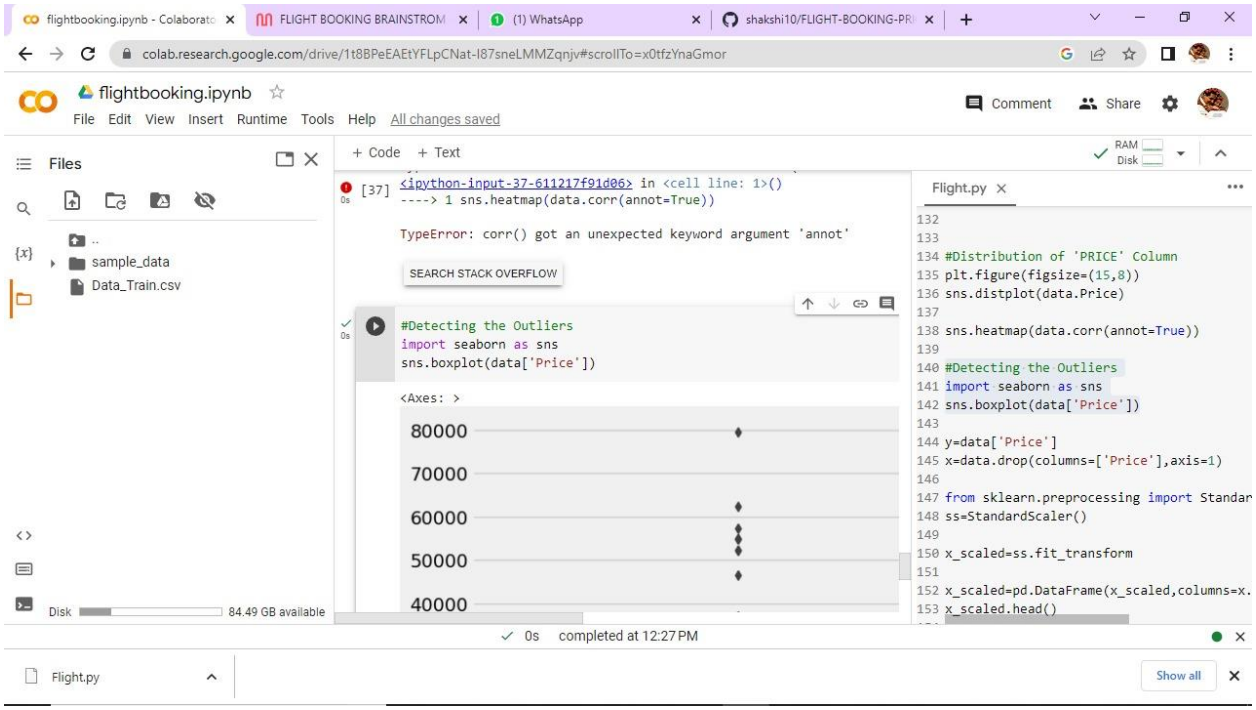
Flight.py

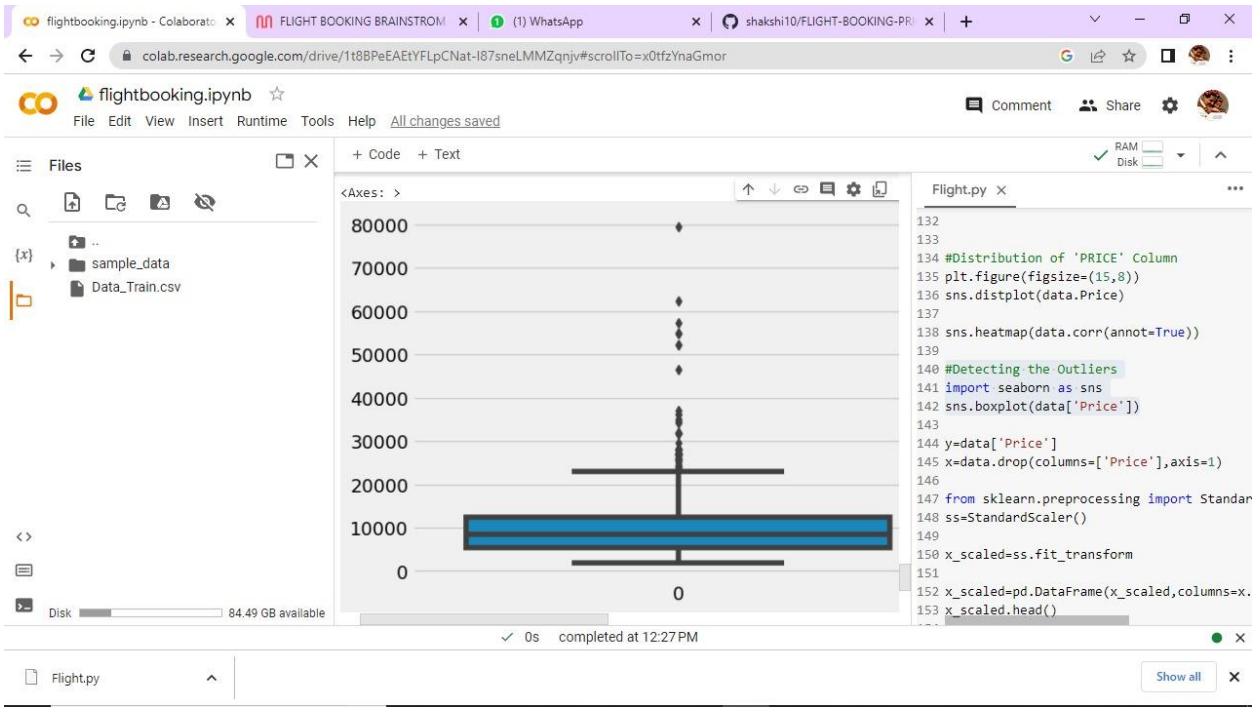
```
109 data.Destination=le.fit_transform(data.De
110 data.Total_Stops=le.fit_transform(data.To
111 data.City1=le.fit_transform(data.City1)
112 data.City2=le.fit_transform(data.City2)
113 data.City3=le.fit_transform(data.City3)
114 data.Addditional_Info=le.fit_transform(da
115 data.head()
116
117
118
119 import seaborn as sns
120 c=1
121 plt.figure(figsize=(20,45))
122
123 for i in categorical:
124     plt.subplot(6,3,c)
125     sns.countplot(data[i])
126     plt.xticks(rotation=90)
127     plt.tight_layout(pad=3.0)
128     c=c+1
129
130 plt.show()
```

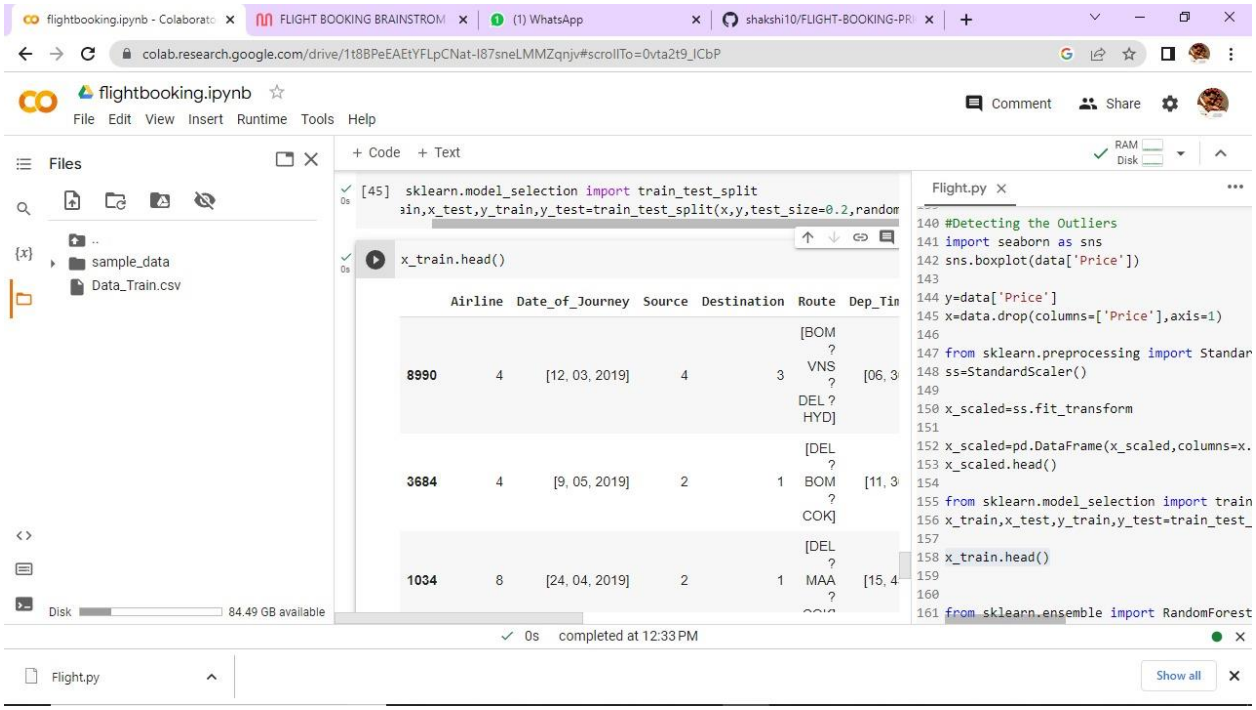
0s completed at 12:17 PM

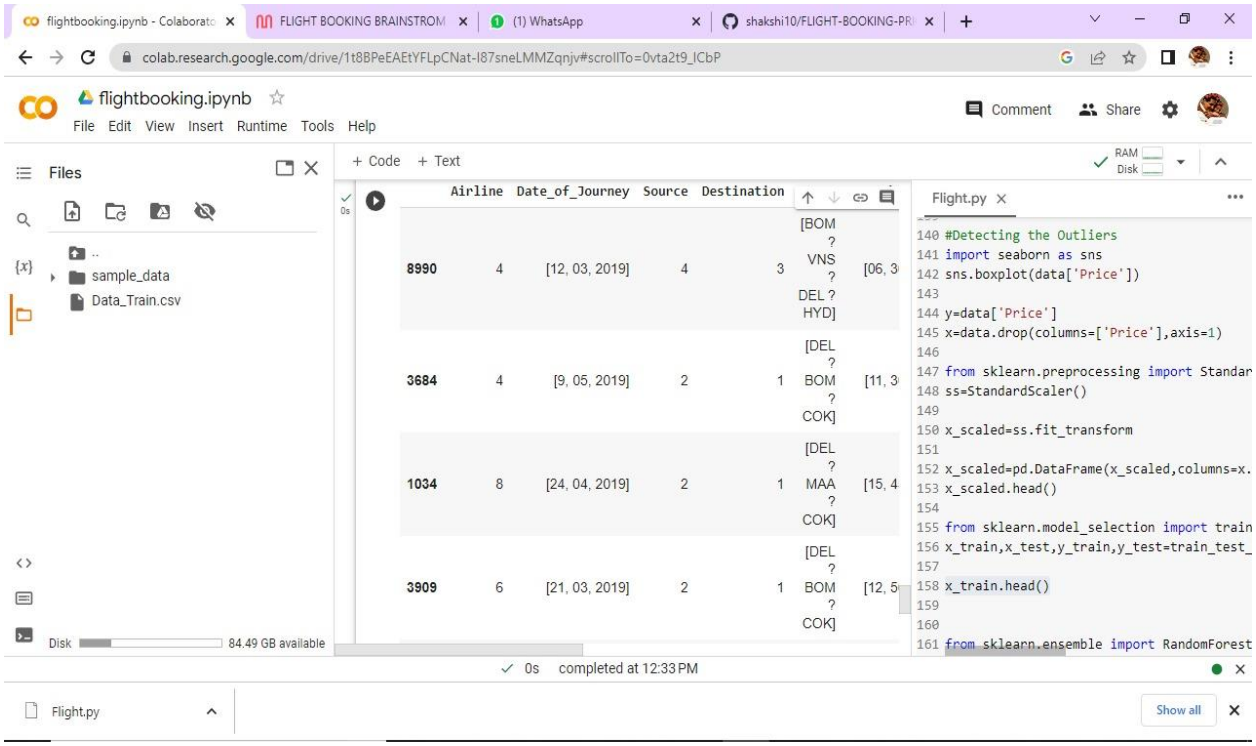
Flight.py Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=KnrF4dBwFAnT
flightbooking.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
#Distribution of 'PRICE' Column
plt.figure(figsize=(15,8))
sns.distplot(data.Price)
<Axes: xlabel='Price', ylabel='Density'>
Density
Price
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
0
20000
40000
60000
Flight.py
121 plt.figure(figsize=(20,45))
122
123 for i in categorical:
124 plt.subplot(6,3,c)
125 sns.countplot(data[i])
126 plt.xticks(rotation=90)
127 plt.tight_layout(pad=3.0)
128 c=c+1
129
130 plt.show()
131
132
133
134 #Distribution of 'PRICE' Column
135 plt.figure(figsize=(15,8))
136 sns.distplot(data.Price)
137
138 sns.heatmap(data.corr(annot=True))
139
140 #Detecting the Outliers
141 import seaborn as sns
142 sns.boxplot(data['Price'])
0s completed at 12:20 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=x0tfzYnaGmor
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
[37] <ipython-input-37-611217f91d06> in <cell line: 1>()
----> 1 sns.heatmap(data.corr(annot=True))
TypeError: corr() got an unexpected keyword argument 'annot'
SEARCH STACK OVERFLOW
#Detecting the Outliers
import seaborn as sns
sns.boxplot(data['Price'])
<Axes: >
80000
70000
60000
50000
40000
Flight.py
132
133
134 #Distribution of 'PRICE' Column
135 plt.figure(figsize=(15,8))
136 sns.distplot(data.Price)
137
138 sns.heatmap(data.corr(annot=True))
139
140 #Detecting the Outliers
141 import seaborn as sns
142 sns.boxplot(data['Price'])
143
144 y=data['Price']
145 x=data.drop(columns=['Price'],axis=1)
146
147 from sklearn.preprocessing import Standar
148 ss=StandardScaler()
149
150 x_scaled=ss.fit_transform
151
152 x_scaled=pd.DataFrame(x_scaled,columns=x.
153 x_scaled.head()
0s completed at 12:27 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=x0tfzYnaGmor
flightbooking.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
RAM
Disk
sample_data
Data_Train.csv
Disk 84.49 GB available
<Axes: >
80000
70000
60000
50000
40000
30000
20000
10000
0
0
Flight.py
#Distribution of 'PRICE' Column
plt.figure(figsize=(15,8))
sns.distplot(data.Price)
sns.heatmap(data.corr(annot=True))
#Detecting the Outliers
import seaborn as sns
sns.boxplot(data['Price'])
y=data['Price']
x=data.drop(columns=['Price'],axis=1)
from sklearn.preprocessing import Standar
ss=StandardScaler()
x_scaled=ss.fit_transform
x_scaled=pd.DataFrame(x_scaled,columns=x.
x_scaled.head()
0s completed at 12:27 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=0vta2t9_ICbP
flightbooking.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
+ Code + Text
RAM
Disk
sample_data
Data_Train.csv
Disk 84.49 GB available
[45] sklearn.model_selection import train_test_split
ain,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random
x_train.head()
Airline Date_of_Journey Source Destination Route Dep_Tim
8990 4 [12, 03, 2019] 4 3 [BOM ? VNS ? DEL ? HYD] [06, 3
3684 4 [9, 05, 2019] 2 1 [DEL ? BOM ? COK] [11, 3
1034 8 [24, 04, 2019] 2 1 [DEL ? MAA ? [15, 4
Flight.py
#Detecting the Outliers
import seaborn as sns
sns.boxplot(data['Price'])
y=data['Price']
x=data.drop(columns=['Price'],axis=1)
from sklearn.preprocessing import Standar
ss=StandardScaler()
x_scaled=ss.fit_transform
x_scaled=pd.DataFrame(x_scaled,columns=x.
x_scaled.head()
from sklearn.model_selection import train
x_train,x_test,y_train,y_test=train_test_
x_train.head()
from sklearn.ensemble import RandomForest
0s completed at 12:33 PM
Flight.py
Show all


flightbooking.ipynb - Colaborato
FLIGHT BOOKING BRAINSTROM
(1) WhatsApp
shakshi10/FLIGHT-BOOKING-PR
colab.research.google.com/drive/1t8BPeEAEtYFLpCNat-I87sneLMMZqnjv#scrollTo=0vta2t9_ICbP
flightbooking.ipynb
File Edit View Insert Runtime Tools Help
Comment
Share
Files
sample_data
Data_Train.csv
Disk 84.49 GB available
+ Code + Text
RAM
Disk
Airline Date_of_Journey Source Destination
8990 4 [12, 03, 2019] 4 3 [BOM ? VNS ? DEL ? HYD] [06, 3
3684 4 [9, 05, 2019] 2 1 [DEL ? BOM ? COK] [11, 3
1034 8 [24, 04, 2019] 2 1 [DEL ? MAA ? COK]
3909 6 [21, 03, 2019] 2 1 [DEL ? BOM ? COK] [12, 5
Flight.py
140 #Detecting the Outliers
141 import seaborn as sns
142 sns.boxplot(data['Price'])
143
144 y=data['Price']
145 x=data.drop(columns=['Price'],axis=1)
146
147 from sklearn.preprocessing import Standar
148 ss=StandardScaler()
149
150 x_scaled=ss.fit_transform
151
152 x_scaled=pd.DataFrame(x_scaled,columns=x.
153 x_scaled.head()
154
155 from sklearn.model_selection import train
156 x_train,x_test,y_train,y_test=train_test_
157
158 x_train.head()
159
160
161 from sklearn.ensemble import RandomForest
0s completed at 12:33 PM
Flight.py
Show all

4. ADVANTAGE :

- Booking airline flights early can give passengers cheaper deals for travel because it gives them time to find and track the best price.
- It can also allow them to begin planning their trip early.
- While this may not appear to be a huge benefit for airports, cheaper airline tickets can lead to increased flight travel.

DISADVANTAGES:

- You an influx of new customers.
- Not all online booking for need internet access. Reliable internet access is required to check reservations and add bookings that are made over the phone.
- You need to be ready systems are created equal.

5. APPLICATION:

Flight booking applications help the airline industry automate the booking process. Users worldwide can book flights on the go using the simple apps, which include features such as quick flight search, download tickets, check and modify booking details, one-tap check-in, and many more.

6.CONCLUSION :

Data visualization as well so after these steps one can go for the prediction using machine learning model making steps.

7 .FUTURE SCOPE:

Airline reservation system make the life of passengers very easy as they don't need to stand in queues for getting their seats reserved and they can easily make reservations on any airline just from a single system.

8. APPENDIX:

A. Source Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
#from sklearn.ensemble import RandomForestclassifier,GradientBoosting
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import f1_score
from sklearn.metrics import classification_report,confusion_matrix
import warnings
import pickle
from scipy import stats
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
```

```
data=pd.read_csv(r"C:\Users\DEEPANAYAKI\Desktop\New folder\Data_Train.csv"
)
data.head()

data.Date_of_Journey=data.Date_of_Journey.str.split('/')
data.Date_of_Journey

for i in category:
  print(i,data[i].unique())
data['Date']=data.Date_of_Journey.str[0]
data['Month']=data.Date_of_Journey.str[1]
data['Year']=data.Date_of_Journey.str[2]
data.Total_Stops.unique()
data.Route=data.Route.str.split('->')
data.Route

data['City1']=data.Route.str[0]
data['City2']=data.Route.str[1]
data['City3']=data.Route.str[2]
data['City4']=data.Route.str[3]
data['City5']=data.Route.str[4]
data['City6']=data.Route.str[5]

data.Dep_Time=data.Dep_Time.str.split(':')
data['Dep_Time_Hour']=data.Dep_Time.str[0]
data['Dep_Time_Mins']=data.Dep_Time.str[1]

data.Arrival_Time=data.Arrival_Time.str.split(' ')

data['Arrival_date']=data.Arrival_Time.str[1]
data['Time_of_Arrival']=data.Arrival_Time.str[0]
data['Time_of_Arrival']=data.Time_of_Arrival.str.split(':')
data['Arrival_Time_Hour']=data.Time_of_Arrival.str[0]
data['Arrival_Time_Mins']=data.Time_of_Arrival.str[1]

data.Duration=data.Duration.str.split(' ')

data['Travel_Hours']=data.Duration.str[0]
data['Travel_Hours']=data['Travel_Hours'].str.split('h')
data['Travel_Hours']=data['Travel_Hours'].str[0]
data.Travel_Hours=data.Travel_Hours
data['Travel_Mins']=data.Duration.str[1]
data.Travel_Mins=data.Travel_Mins.str.split('m')
data.Travel_Mins=data.Travel_Mins.str.split('m')
data.Travel_Mins=data.Duration.str[0]
```

```
data.Total_Stops.replace('non_stop',0,inplace=True)
data.Total_Stops=data.Total_Stops.str.split(' ')
data.Total_Stops=data.Total_Stops.str[0]

data.Additional_Info.unique()

data.Additional_Info.replace('No Info','No Info',inplace=True)

data.isnull().sum()

data.drop(['City4','City5','City6'],axis=1,inplace=True)
data.drop(['Date_of_Journey','Route','Dep_Time','Arrival_Time','Duration']
,axis=1,inplace=True)
data.drop(['Time_of_Arrival'],axis=1,inplace=True)

data.isnull().sum()

data['City3'].fillna('None,inplae=True')
data['Arrival_Date'].fillna(data['Datedata.Dep_Time_Hur=data.Dep_Time_Hour
.astype('int64')
data.Dep_Time_Mins=data.Dep_Time_Mins.astype('int64')
data.Arrival_date=data.Arrival_date.astype('int64')
data.Arrival_Time_Hours=data.Arrival_Time_Hours.astype('int64')
data.Arrival_Time_Mins=data.Arrival_Time_Mins.astype('int64')
data.Travel_Mins=data.Travel_Mins.astype('int64')

data[data['Travel_Hours']=='5m']'],inplace=True)
data['Travel_Mins'].fillna(0,inplace=True)

data.info()


#data.Date_of_Journey=data.Date_of_Journey.astype('int64')
#data.Month=odata.Month.astype('int64')
#data.Year=data.Year.astype('int64')

#data.drop(index=6474,inplace=True,axis=0)

data.Travel_Hours=data.Travel_Hours.astype('int64')

categorical=['Airline','Source','Destination','Additional_Info','City1']
numerical=['Total_Stops','Date','Month','Year','Dep_Time_Hour','Dep_Time_M
in','Arrival_date','Arrival_Time_Hour','Arrival_Time_Mins','Travel_Hours',
'Travel_Mins']
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()


data.Airline=le.fit_transform(data.Airline)
data.Source=le.fit_transform(data.Source)
data.Destination=le.fit_transform(data.Destination)
data.Total_Stops=le.fit_transform(data.Total_Stops)
data.City1=le.fit_transform(data.City1)
data.City2=le.fit_transform(data.City2)
data.City3=le.fit_transform(data.City3)
data.Addditional_Info=le.fit_transform(data.Additional_Info)
data.head()



import seaborn as sns
c=1
plt.figure(figsize=(20,45))

for i in categorical:
    plt.subplot(6,3,c)
    sns.countplot(data[i])
    plt.xticks(rotation=90)
    plt.tight_layout(pad=3.0)
    c=c+1

plt.show()



#Distribution of 'PRICE' Column
plt.figure(figsize=(15,8))
sns.distplot(data.Price)

sns.heatmap(data.corr(annot=True))

#Detecting the Outliers
import seaborn as sns
sns.boxplot(data['Price'])

y=data['Price']
x=data.drop(columns=['Price'],axis=1)
```

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()

x_scaled=ss.fit_transform

x_scaled=pd.DataFrame(x_scaled,columns=x.columns)
x_scaled.head()

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_st
ate=42)

x_train.head()


from sklearn.ensemble import RandomForestRegressor,GradientBoostingRegress
or,AdaBoostRegressor
rfr=RandomForestsRegressor()
gb=GradientBoostingRegressor()
ad=AdaBoostRegressor()

from sklearn.metrics import r2_Score,mean_absolute_error,mean_squared_erro
r

for i in [rfr,gb,ad]:
    i.fit(x_train,y_train)
    y_pred=i.predict(x_test)
    test_score=r2_score(y_test,y_pred)
    train_score=r2_score(y_train,i.predict(x_train))
    if abs(train_score-test_score)<=0.2:
       print(i)

       print("R2 score is",r2_score(y_test,y_pred))
       print("R2 for train data",r2_score(y_train,i.predict(x_train))
       print("Mean Absolue Error is",mean_absolute_error(y_pred,y_test))
       print("Mean Squared Error is",mean_squared_error(y_pred,y_test))
       print("Root Mean Sqaured Error is",(mean_Squared_error(y_pred,y_tes
t,squared-False)))
```

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_erro
r

knn=KNeighborsRegressor()
svr=SVR()
dt=DecisionTreeRegressor()

for i in [knn,svr,dt]:
    i.fit(x_train,y_train)
    y_pred-i.predict(x_test)
    test_score=r2_score(y_test,y_pred)
    train_score=r2score(y_train,i.predict(x_train))
    if abs(train_score-test_score)<=0.1:
       print(i)
print('R2 Score is',r2_score(y_test,y_pred))
print('R2 Score for train data',r2_score(y_train,i,predict(x_train)))
print('Mean Absolute Error is',mean_absolute_error(y_test,y_pred))
print('Mean squared Error is',mean_squared_error(y_test,y_pred))
print('Root Mean Squared Error is',(mean_squared_error(y_test,y_pred,squar
ed=False)))
```

```
from sklearn.model_selection import cross_val_score
for i in range(2,5):
    cv=cross_val_score(rfr,x,y,cv=i)
    print(rfr,cv.mean())
```

```
rfr=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=No
ne)
rfr.fit(x_train,y_train)
y_train_pred=rfr.predict(x_train)
y_test_pred=rfr.predict(x_test)
print("train accurancy",r2_score(y_train_pred,y_train))
print("test accurancy",r2_score(y_test_pred,y_test))
```

```
knn=KNeighboursRegressor(n_neighbors=2,algorithm='auto',metric_params=None
,n_jobs=-1)
Knn.fit(x_train,y_train)
y_train_pred=knn.predict (x_train)
y_test_pred=Knn.predict(x_test)
print("train accurancy",r2_score(y_train_pred,y_train))
print("test accurancy",r2_score(y_test_pred,y_test))

import pickle
pickle.dump(rfr,open('model1.pkl','wb'))


from Flask import Flask render_template,'request'
import numpy as np
import pickle

model=pickle.load(open(r"mode[].prl",'rb'))

@app,route("/home")
def home():
    return render_template('home.html')

@app.route("/predict")
def home1():
    return render_template('predict.html')

@app.route("/pred",methods=['POST','GET'])
def predict():
    x=[[int(x) for xin request.form.values()]]
    print(x)

    x=np.array(x)
    print(x.shape)

    print(x)
    pred=model.predict(x)
    print(pred)
    return render_template('submit.html',predicition_text=pred)

if __name__ == "__main__":
    app.run(debug=False)

from tensortflow.kers.models import Sequential
from tensortflow.keras.layers import Dense
```

```
#Fitting the model to the training sets
model = Sequential()

x_train.shape
 (4457, 7163)

model.add(Dense(units =x_train_res.shape[1],activation="relu",kernel_initi
alizer="random_uniform"))

model.add(Dense(units=100,activation="relu",kernel_initializer="random_uni
form"))

model.add(Dense(units=100,activation="relu",kernel_initializer="random_uni
form"))

model.add(Dense(units*1,activation="sigmoid"))

model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accura
cy'])

generator=model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch=len(
x_train_res)//64)


from sklearn.naive_bayes import MultinomialNB
model=multinomialNB()

#Fitting the model to the training sets
model.fit(x_train_res,y_train_res)
```