

**Kamarajar Government Arts College,
Surandai
Tenkasi (Dt.) - 627859.**



MANONMANIAM SUNDARANAR UNIVERSITY

APRIL - 2023

**Optimizing Flight Booking Decisions
through Machine Learning Price Prediction**

Team ID : NM2023TMID22068

Team Leader : SELVARAJ. N

Reg. No : 20201061506138

Team Member : SELVAMOORTHY. C

Reg. No : 20201061506136

Team Member : VENKATESH. M

Reg. No : 20201061506147

Team Member : KARTHICK KUMAR. V

Reg. No : 20201061506109

INDEX

S.No:	TITLE	Pg.No:
01	Introduction	04
02	Problem Definition & Design Thinking	07
03	Result	09
04	Advantages &Disadvantages	10
05	Application	12
06	Conclusion	13
07	Future Scope	14
08	Appendix	16

INTRODUCTION:

Welcome to the world of flight ticket booking optimization through machine learning. In this project, we will be exploring how we can leverage machine learning techniques to predict flight ticket prices and help customers save money on their air travel expenses.

Airline ticket prices are highly dynamic and can fluctuate rapidly based on various factors such as seasonality, demand, competition, and availability. By using machine learning algorithms, we can analyze historical data and predict future prices with a high degree of accuracy.

Our goal is to build a predictive model that can accurately forecast future ticket prices and provide recommendations to users on the optimal time to book their flights. This project will involve data collection, data cleaning, feature engineering, model selection, and performance evaluation.

Through this project, we aim to help travelers make informed decisions about their flight bookings and save money by booking at the right time. We look forward to embarking on this exciting journey with you! Learning models will become even more accurate and effective in predicting flight delays, enabling the industry to operate more efficiently and reliably.

Optimizing flight ticket booking through machine learning price prediction is a project that aims to improve the process of purchasing airline tickets by leveraging machine learning algorithms to predict ticket prices. The goal is to

enable travelers to make informed decisions about when to book their tickets and how much to pay for them.

The project will involve collecting and analyzing historical data on flight prices, as well as information about various factors that affect ticket prices, such as time of year, airline, and route. Using this data, the machine learning model will be trained to accurately predict future ticket prices.

The benefits of this project are numerous. For travelers, it can help them save money by identifying the optimal time to book tickets. For airlines, it can improve their revenue management by allowing them to better predict demand and adjust prices accordingly. Overall, this project has the potential to revolutionize the way people purchase airline tickets, making it more convenient and cost-effective.

Overview:

Flight booking decisions can be difficult, with travelers often struggling to balance their budget constraints with their preferred travel dates and airline preferences. Machine learning (ML) can help travelers make more informed decisions by predicting flight prices based on historical data and current market trends.

ML algorithms can analyze large amounts of data from various sources, such as airline websites, travel agencies, and social media platforms, to identify patterns and trends that influence flight prices. By using these patterns, algorithms can make accurate predictions on how prices will change in the future, enabling travelers to make more informed decisions on when and where to book their flights.

With the help of ML-powered price prediction tools, travelers can optimize their flight booking decisions and save money by booking at the right time and choosing the best airline and travel dates for their needs. Additionally, airlines and travel agencies can benefit from these tools by improving their pricing strategies and increasing customer satisfaction.

Overall, the use of ML in flight booking can greatly improve the travel experience for both travelers and businesses, making it easier and more efficient to book flights and travel to new destinations.

Purpose:

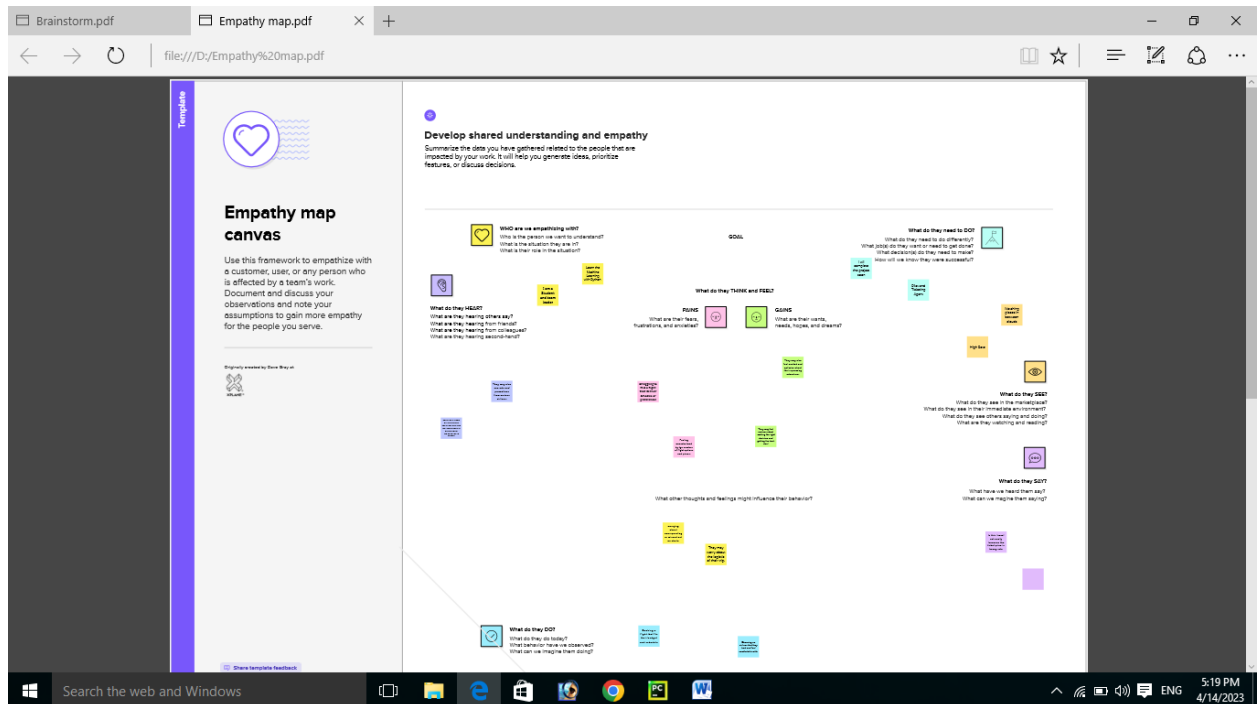
The purpose of using machine learning for optimizing flight booking decisions through price predictions is to help travelers make more informed decisions about their flights, leading to cost savings and a better travel experience. The project aims to provide accurate predictions of flight prices based on historical data and current market trends, which can help travelers determine the best time to book their flights and choose the most cost-effective travel dates and airlines.

The use of machine learning in this project also benefits airlines and travel agencies by enabling them to optimize their pricing strategies, increase customer satisfaction, and gain a competitive edge in the industry.

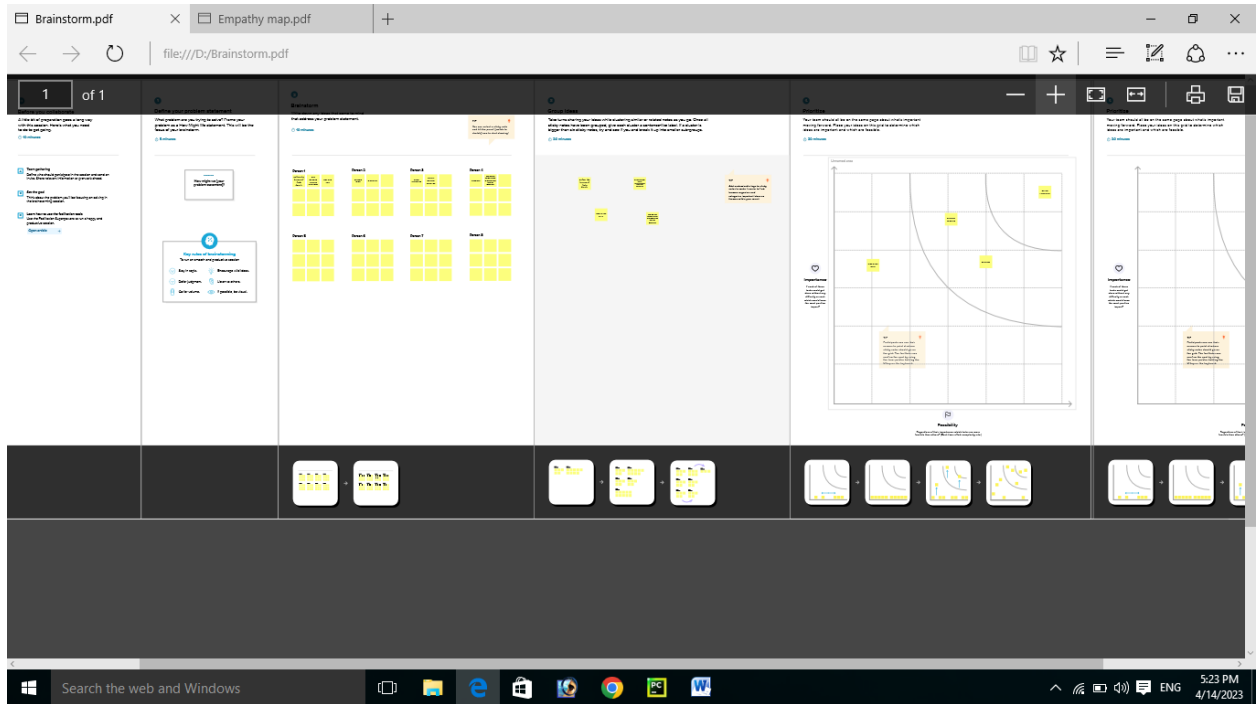
Overall, the purpose of this project is to leverage the power of machine learning to improve the travel experience for both travelers and businesses, making it easier and more efficient to book flights and travel to new destinations.

Problem definition & designing thinking:

Empathy map:



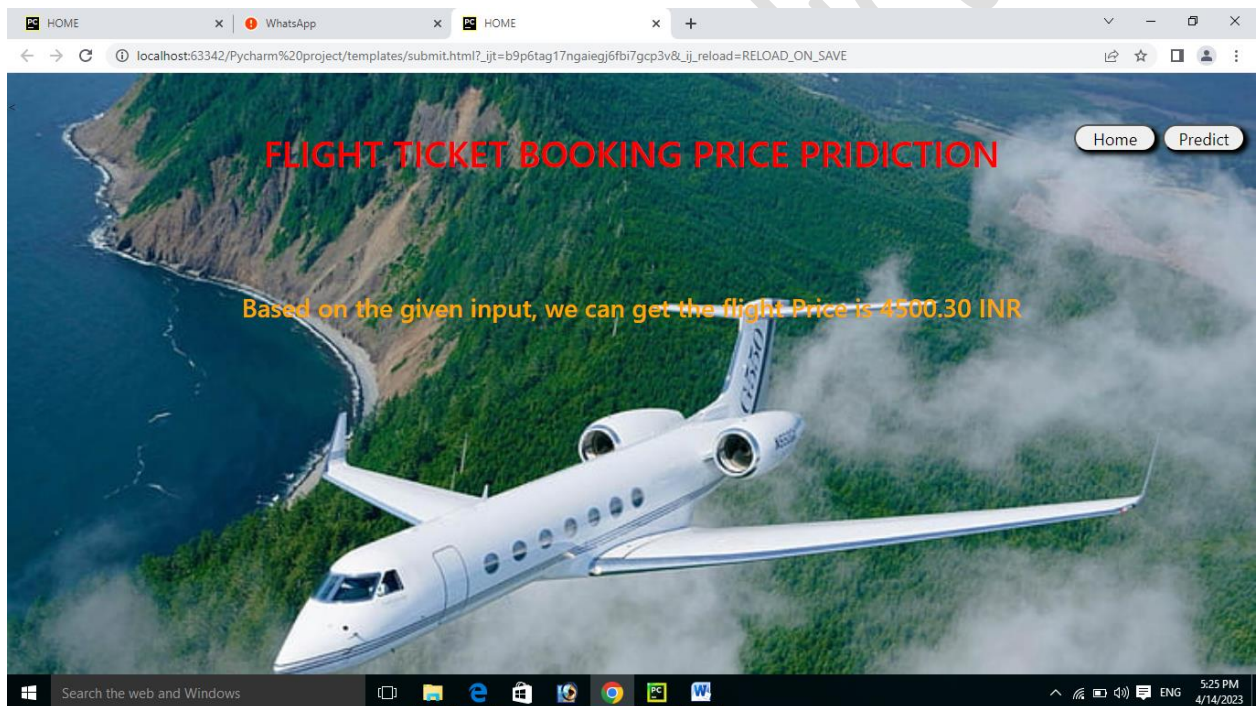
Brainstorming Map:



Result:

The use of machine learning for optimizing flight booking decisions through price prediction has the potential to revolutionize the travel industry. By analysing large amounts of data from various sources, machine learning algorithms can make accurate price predictions and help travellers make more informed decisions about their flights.

Output: (final output)



Advantages:

Cost savings:

Machine learning price prediction models can analyze a vast amount of data and predict the best time to buy tickets, which can help customers save money on their flight bookings.

Improved customer satisfaction:

If customers can book flights at lower prices, they are more likely to be satisfied with the booking experience and the airline brand.

Competitive advantage:

Airlines that offer accurate price predictions through machine learning can gain a competitive advantage in the market, as customers will be more likely to choose their services over others.

Efficient use of resources:

By predicting the demand for flights, airlines can optimize their resources, including planes and crew, reducing the risk of flying with empty seats.

Disadvantages:

Limited accuracy:

Machine learning algorithms are only as good as the data they are trained on, and there is always a risk of inaccurate predictions. This can lead to customer frustration and dissatisfaction if they miss out on cheaper flights.

Limited data availability:

Access to real-time data on flight bookings and cancellations can be limited, which can impact the accuracy of machine learning models.

Ethical concerns:

There is a risk that airlines may use the information they gather through machine learning algorithms to engage in price discrimination or other unethical practices.

Technical challenges:

Implementing a machine learning algorithm can be technically challenging, requiring significant investment in data analysis, infrastructure, and expertise.

Applications:

The application of this project is in the travel industry, particularly in the areas of online travel agencies, airline websites, and travel search engines. By implementing machine learning algorithms for flight booking decisions, companies can provide their customers with more accurate and personalized recommendations for flights, resulting in cost savings and an enhanced travel experience.

This also extends to airlines and travel agencies, as they can use machine learning algorithms to optimize their pricing strategies and improve their revenue management. By analyzing data on customer behavior and market trends, companies can adjust their prices in real-time to attract more customers and increase profitability.

By using machine learning algorithms to analyze data from multiple sources, companies can provide their customers with more comprehensive travel recommendations and an overall better travel experience.

In travel industry, where machine learning algorithms can be used to optimize flight booking decisions, improve revenue management, and enhance the travel experience for both businesses and customers.

Conclusion:

By analyzing large amounts of data from various sources, machine learning algorithms can make accurate price predictions and help travelers make more informed decisions about their flights.

The benefits of using machine learning for flight booking decisions include increased accuracy, time and cost savings, customization, and improved pricing strategies for airlines and travel agencies. However, there are potential drawbacks, such as the need for accurate and up-to-date data, technical expertise, unforeseen events, and privacy concerns.

Despite these challenges, the application of machine learning for flight booking decisions is a promising development for the travel industry. Companies that adopt these technologies are likely to gain a competitive advantage by providing better pricing strategies and personalized recommendations to their customers, resulting in increased customer satisfaction and loyalty.

Future Scope:

The future scope of this project is vast and promising, with the potential to further improve the travel experience for both travelers and businesses.

Integration with augmented reality:

The integration of machine learning algorithms with augmented reality could help travelers to visualize and interact with their travel plans in real-time, providing a more immersive and personalized travel experience.

Expansion to other modes of transportation:

The application of machine learning for optimizing travel decisions can be extended beyond flight bookings to include other modes of transportation such as trains, buses, and taxis. This would allow travelers to make more informed decisions about their travel plans and choose the most cost-effective and efficient modes of transportation.

Integration with smart cities:

The integration of machine learning algorithms with smart city infrastructure could help travelers to navigate their destinations more efficiently, providing real-time traffic information, parking recommendations, and other helpful information.

Personalized travel recommendations:

As machine learning algorithms become more advanced, they will be able to provide more personalized travel recommendations based on a traveler's preferences and past travel history. This would allow companies to offer customized travel packages and improve customer loyalty.

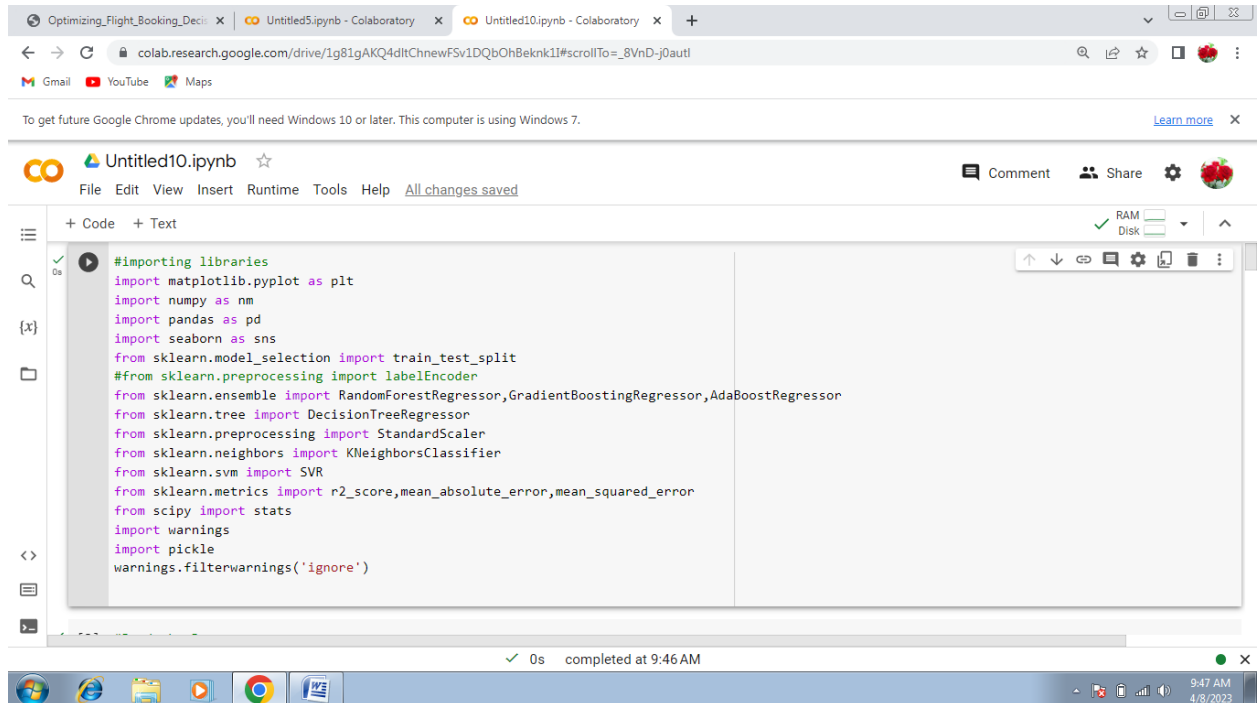
Increased use of natural language processing: The use of natural language processing in conjunction with machine learning algorithms could

enable travelers to interact with virtual travel assistants using natural language, providing a more intuitive and user-friendly experience.

Flight Price Prediction

Appendix :

Source Code :



Optimizing_Flight_Booking_Decis x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=_8VnD-j0autl

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

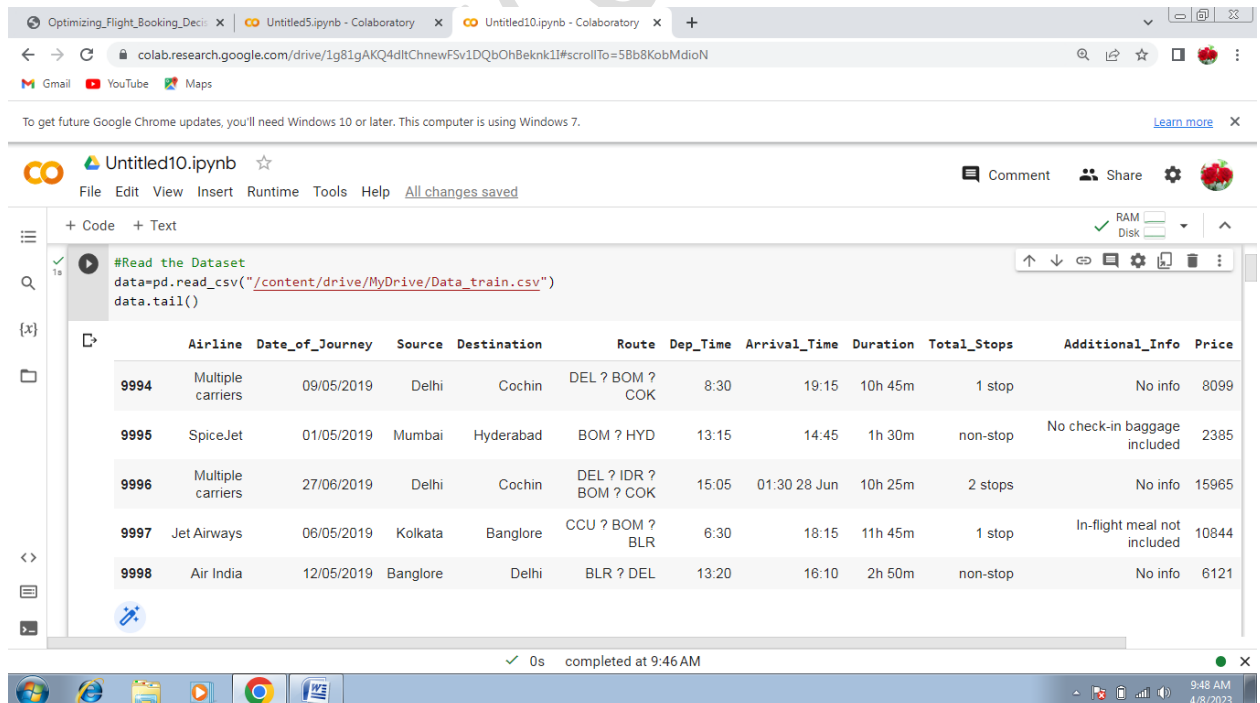
Untitled10.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
#importing libraries
import matplotlib.pyplot as plt
import numpy as nm
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
#from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from scipy import stats
import warnings
import pickle
warnings.filterwarnings('ignore')
```

0s completed at 9:46 AM



Optimizing_Flight_Booking_Decis x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=5Bb8KobMdioN

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
#Read the Dataset
data=pd.read_csv("/content/drive/MyDrive/Data_train.csv")
data.tail()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
9994	Multiple carriers	09/05/2019	Delhi	Cochin	DEL ? BOM ? COK	8:30	19:15	10h 45m	1 stop	No info	8099
9995	SpiceJet	01/05/2019	Mumbai	Hyderabad	BOM ? HYD	13:15	14:45	1h 30m	non-stop	No check-in baggage included	2385
9996	Multiple carriers	27/06/2019	Delhi	Cochin	DEL ? IDR ? BOM ? COK	15:05	01:30 28 Jun	10h 25m	2 stops	No info	15965
9997	Jet Airways	06/05/2019	Kolkata	Banglore	CCU ? BOM ? BLR	6:30	18:15	11h 45m	1 stop	In-flight meal not included	10844
9998	Air India	12/05/2019	Banglore	Delhi	BLR ? DEL	13:20	16:10	2h 50m	non-stop	No info	6121

0s completed at 9:46 AM

Optimizing_Flight_Booking_Dec... x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk11#scrollTo=7GoiNC3ne8mz

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

data.columns

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

data.info

```
<bound method DataFrame.info of
0      IndiGo      24/03/2019  Bangalore  New Delhi
1      Air India    01/05/2019  Kolkata    Bangalore
2      Jet Airways  09/06/2019  Delhi     Cochin
3      IndiGo      12/05/2019  Kolkata    Bangalore
4      IndiGo      01/03/2019  Bangalore New Delhi
...
9994 Multiple carriers 09/05/2019  Delhi     Cochin
9995 SpiceJet         01/05/2019  Mumbai   Hyderabad
9996 Multiple carriers 27/06/2019  Delhi     Cochin
9997 Jet Airways      06/05/2019  Kolkata    Bangalore
```

0s completed at 9:48 AM

Optimizing_Flight_Booking_Dec... x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk11#scrollTo=LX9FSm1xfNN2

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

data.info

```
<bound method DataFrame.info of
0      IndiGo      24/03/2019  Bangalore  New Delhi
1      Air India    01/05/2019  Kolkata    Bangalore
2      Jet Airways  09/06/2019  Delhi     Cochin
3      IndiGo      12/05/2019  Kolkata    Bangalore
4      IndiGo      01/03/2019  Bangalore New Delhi
...
9994 Multiple carriers 09/05/2019  Delhi     Cochin
9995 SpiceJet         01/05/2019  Mumbai   Hyderabad
9996 Multiple carriers 27/06/2019  Delhi     Cochin
9997 Jet Airways      06/05/2019  Kolkata    Bangalore
9998 Air India       12/05/2019  Bangalore Delhi

Route Dep_Time Arrival_Time Duration Total_Stops \
0      BLR ? DEL  22:20  01:10 22 Mar  2h 50m  non-stop
1      CCU ? IXR ? BBT ? BLR  5:50  13:15  7h 25m  2 stops
2      DEL ? LKO ? BOM ? COK  9:25  04:25 10 Jun  19h  2 stops
3      CCU ? NAG ? BLR  18:05  22:30  5h 25m  1 stop
4      BLR ? NAG ? DEL  16:50  21:35  4h 45m  1 stop
...
9994 DEL ? BOM ? COK  8:30  19:15 10h 45m  1 stop
9995 BOM ? HYD  13:15  14:45  1h 30m  non-stop
9996 DEL ? IDR ? BOM ? COK  15:05  01:30 28 Jun 10h 25m  2 stops
9997 CCU ? BOM ? BLR  6:30  18:15 11h 45m  1 stop
9998 BLR ? DEL  13:20  16:10  2h 50m  non-stop

Additional_Info Price
0      No info  3897
1      No info  7662
2      No info 13082
3      No info  6218
4      No info 13302
...
9994 No info  8959
```

0s completed at 9:48 AM

Optimizing_Flight_Booking_Dec... x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk11#scrollTo=BUSdTUEci3E-

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[] data.describe()

(x)

	Price
count	9999.000000
mean	9088.149915
std	4591.304024
min	1759.000000
25%	5276.500000
50%	6372.000000
75%	12373.000000
max	79512.000000

data.isnull().head()

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False

completed at 9:48 AM

9:49 AM 4/8/2023

Optimizing_Flight_Booking_Dec... x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk11#scrollTo=F6hw5QQciVPz

Gmail YouTube Maps

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 10:01 AM

+ Code + Text

```
for i in data:
    print(i,data[i].unique())
```

(x)

Airline ['Indigo' 'Air India' 'Jet Airways' 'SpiceJet' 'Multiple carriers' 'Goair' 'Vistara' 'Air Asia' 'Vistara Premium economy' 'Jet Airways Business' 'Multiple carriers Premium economy' 'Trujet']

Date_of_Journey ['24/03/2019' '01/05/2019' '09/06/2019' '12/05/2019' '01/03/2019' '24/06/2019' '12/03/2019' '27/05/2019' '01/06/2019' '18/04/2019' '09/05/2019' '24/04/2019' '03/03/2019' '15/04/2019' '12/06/2019' '06/03/2019' '21/03/2019' '03/04/2019' '06/05/2019' '15/06/2019' '18/06/2019' '15/06/2019' '06/04/2019' '18/05/2019' '27/06/2019' '21/05/2019' '03/06/2019' '15/03/2019' '03/05/2019' '09/03/2019' '06/06/2019' '24/05/2019' '01/04/2019' '21/06/2019' '27/03/2019' '18/03/2019' '12/04/2019' '09/04/2019' '27/04/2019']

Source ['Bangalore' 'Kolkata' 'Delhi' 'Chennai' 'Mumbai']

Destination ['New Delhi' 'Bangalore' 'Cochin' 'Kolkata' 'Delhi' 'Hyderabad']

Route ['BLR > DEL' 'CCU > IXR > BBI > BLR' 'DEL > LKO > BOM > COK' 'CCU > NAG > BLR' 'BLR > NAG > DEL' 'CCU > BLR' 'BLR > BOM > DEL' 'DEL > BOM > COK' 'DEL > BLR > COK' 'MAA > CCU' 'CCU > BOM > BLR' 'DEL > AMD > BOM > COK' 'DEL > PNQ > COK' 'DEL > CCU > BOM > COK' 'BLR > COK > DEL' 'DEL > IDR > BOM > COK' 'DEL > LKO > COK' 'CCU > GAU > DEL > BLR' 'DEL > NAG > BOM > COK' 'CCU > MAA > BLR' 'DEL > HYD > COK' 'CCU > HYD > BLR' 'DEL > COK' 'CCU > DEL > BLR' 'BLR > BOM > AMD > DEL' 'BOM > DEL > HYD' 'DEL > MAA > COK' 'BOM > HYD' 'DEL > BHO > BOM > COK' 'DEL > JAI > BOM > COK' 'DEL > ATQ > BOM > COK' 'DEL > JDM > BOM > COK' 'CCU > BBI > BOM > BLR' 'BLR > MAA > DEL' 'DEL > GOI > BOM > COK' 'DEL > BQJ > BOM > COK' 'CCU > JAI > BOM > BLR' 'CCU > BBI > BLR' 'BLR > HYD > DEL' 'DEL > TRV > COK' 'CCU > IXR > DEL > BLR' 'DEL > IXU > BOM > COK' 'CCU > IXB > BLR' 'BLR > BOM > JDM > DEL' 'DEL > UDR > BOM > COK' 'DEL > HYD > MAA > COK' 'CCU > BOM > COK > BLR' 'BLR > CCU > DEL' 'CCU > BOM > GOI > BLR' 'DEL > RPR > NAG > BOM > COK' 'DEL > HYD > BOM > COK' 'CCU > DEL > AMD > BLR' 'CCU > PNQ > BLR' 'BLR > CCU > GAU > DEL' 'CCU > DEL > COK > BLR' 'BLR > PNQ > DEL' 'BOM > JDM > DEL > HYD' 'BLR > BOM > BHO > DEL' 'DEL > AMD > COK' 'BLR > LKO > DEL' 'CCU > GAU > BLR' 'BOM > GOI > HYD' 'CCU > BOM > AMD > BLR'

10:03 AM 4/8/2023

Optimizing_Flight_Booking_Deci x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=rJEK_jAti5aN

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

+ Code + Text

```
data["Airline"].unique()

array(['IndiGo', 'Air India', 'Jet Airways', 'SpiceJet',
       'Multiple carriers', 'GoAir', 'Vistara', 'Air Asia',
       'Vistara Premium economy', 'Jet Airways Business',
       'Multiple carriers Premium economy', 'Trujet'], dtype=object)

[ ] data["Source"].unique()

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

[ ] data["Destination"].unique()

array(['New Delhi', 'Bangalore', 'Cochin', 'Kolkata', 'Delhi', 'Hyderabad'],
      dtype=object)

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'], dtype=object)
```

0s completed at 9:48 AM

9:50 AM 4/8/2023

Optimizing_Flight_Booking_Deci x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=OG_NLgskqH8A

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

+ Code + Text

```
'Red-eye flight'], dtype=object)

#split the Date Column
data.Date_of_Journey=data.Date_of_Journey.str.split('/')
data.Date_of_Journey

0      [24, 03, 2019]
1      [01, 05, 2019]
2      [09, 06, 2019]
3      [12, 05, 2019]
4      [01, 03, 2019]
...
9994   [09, 05, 2019]
9995   [01, 05, 2019]
9996   [27, 06, 2019]
9997   [06, 05, 2019]
9998   [12, 05, 2019]
Name: Date_of_Journey, Length: 9999, dtype: object

[ ] data.Route=data.Route.str.split('->')
```

0s completed at 9:56 AM

9:57 AM 4/8/2023

Optimizing_Flight_Booking_Dec... x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=mEFi5u5kHpGI

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
#Treating the Data Column
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.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3
0	IndiGo	[24, 03, 2019]	Banglore	New Delhi	[BLR ? DEL]	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897	24	03	2019	BLR ? DEL	NaN	NaN
1	Air India	[01, 05, 2019]	Kolkata	Banglore	[CCU ? IXR ? BBI ? BLR]	5:50	13:15	7h 25m	2 stops	No info	7662	01	05	2019	CCU ? IXR ? BBI ? BLR	NaN	NaN
2	Jet Airways	[09, 06, 2019]	Delhi	Cochin	[DEL ? LKO ? BOM ? COK]	9:25	04:25 10 Jun	19h	2 stops	No info	13882	09	06	2019	DEL ? LKO ? BOM ? COK	NaN	NaN
3	IndiGo	[12, 05, 2019]	Kolkata	Banglore	[CCU ? NAG ? BLR]	18:05	23:30	5h 25m	1 stop	No info	6218	12	05	2019	CCU ? NAG ? BLR	NaN	NaN
4	IndiGo	[01, 03, 2019]	Banglore	New Delhi	[BLR ? NAG ? DEL]	16:50	21:35	4h 45m	1 stop	No info	13302	01	03	2019	BLR ? NAG ? DEL	NaN	NaN

[25] data.Route=data.Route.str.split('->')

0s completed at 9:58 AM

WhatsApp x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x Copy of Copy of Copy of Copy x Optimizing_Flight_Booking_De... x +

colab.research.google.com/drive/1WDG9L3JX9J9uRzvNG9w55g4DvEaiQbVD#scrollTo=v1dZdEMWcRPI

Gmail YouTube Maps

Copy of Copy of Copy of Copy of Welcome To Colaboratory

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[6] data.Route=data.Route.str.split('->')
data.Route
data.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	[BLR ? DEL]	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897
1	Air India	01/05/2019	Kolkata	Banglore	[CCU ? IXR ? BBI ? BLR]	5:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	09/06/2019	Delhi	Cochin	[DEL ? LKO ? BOM ? COK]	9:25	04:25 10 Jun	19h	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	[CCU ? NAG ? BLR]	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	[BLR ? NAG ? DEL]	16:50	21:35	4h 45m	1 stop	No info	13302

0s completed at 11:30 AM

Optimizing_Flight_Booking_Deci x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=tbx89xjwnWn1

Gmail YouTube Maps

To get future Google Chrome updates, you'll need Windows 10 or later. This computer is using Windows 7. [Learn more](#)

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
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.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3
0	IndiGo	[24, 03, 2019]	Banglore	New Delhi	[BLR ? DEL]	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897	24	03	2019	BLR ? DEL	NaN	NaN
1	Air India	[01, 05, 2019]	Kolkata	Banglore	[CCU ? IXR ? BBI ? BLR]	5:50	13:15	7h 25m	2 stops	No info	7662	01	05	2019	CCU ? IXR ? BBI ? BLR	NaN	NaN
2	Jet Airways	[09, 06, 2019]	Delhi	Cochin	[DEL ? LKO ? BOM ? COK]	9:25	04:25 10 Jun	19h	2 stops	No info	13882	09	06	2019	DEL ? LKO ? BOM ? COK	NaN	NaN
3	IndiGo	[12, 05, 2019]	Kolkata	Banglore	[CCU ? NAG ? BLR]	18:05	23:30	5h 25m	1 stop	No info	6218	12	05	2019	CCU ? NAG ? BLR	NaN	NaN
4	IndiGo	[01, 03, 2019]	Banglore	New Delhi	[BLR ? NAG ? DEL]	16:50	21:35	4h 45m	1 stop	No info	13302	01	03	2019	BLR ? NAG ? DEL	NaN	NaN

0s completed at 9:58 AM

Optimizing_Flight_Booking_Deci x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=dbLnnXjWp8oE

Gmail YouTube Maps

Untitled10.ipynb ☆

File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

```
raise AttributeError("Can only use .str accessor with string values!")
return inferred_dtype

AttributeError: Can only use .str accessor with string values!
```

```
#In the similar manner, we split the Dep_time column, and create separate columns for departure hours and minutes
data['Dep_hour'] = pd.to_datetime(data['Dep_Time']).dt.hour
data['Dep_min'] = pd.to_datetime(data['Dep_Time']).dt.minute
data.drop('Dep_Time',axis=1,inplace=True)
data['Arrival_hour'] = pd.to_datetime(data['Arrival_Time']).dt.hour
data['Arrival_min'] = pd.to_datetime(data['Arrival_Time']).dt.minute
data.drop('Arrival_Time',axis=1,inplace=True)
data.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	Date	Month	Year	City1	City2	City3	Dep_hour	Dep_min	Arrival_hour	Arrival_min
0	IndiGo	[24, 03, 2019]	Banglore	New Delhi	[BLR ? DEL]	2h 50m	non-stop	No info	3897	24	03	2019	BLR ? DEL	NaN	NaN	22	20	1	10
1	Air India	[01, 05, 2019]	Kolkata	Banglore	[CCU ? IXR ? BBI ? BLR]	7h 25m	2 stops	No info	7662	01	05	2019	CCU ? IXR ? BBI ? BLR	NaN	NaN	5	50	13	15
2	Jet Airways	[09, 06, 2019]	Delhi	Cochin	[DEL ? LKO ? BOM ? COK]	19h	2 stops	No info	13882	09	06	2019	DEL ? LKO ? BOM ? COK	NaN	NaN	9	25	4	25
3	IndiGo	[12, 05, 2019]	Kolkata	Banglore	[CCU ? NAG ? BLR]	5h 25m	1 stop	No info	6218	12	05	2019	CCU ? NAG ? BLR	NaN	NaN	18	5	23	30
4	IndiGo	[01, 03, 2019]	Banglore	New Delhi	[BLR ? NAG ? DEL]	4h 45m	1 stop	No info	13302	01	03	2019	BLR ? NAG ? DEL	NaN	NaN	16	50	21	35

```
[ ] data.Duration=data.Duration.str.split('')
data['Travel_Hours']=data.Duration.str[1]
```

0s completed at 10:03 AM

Optimizing_Flight_Booking_Deci x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x +

colab.research.google.com/drive/1g81gAKQ4dltChnewFsv1DQbOhBeknk1I#scrollTo=7XD1JN2rJf3

Google Gmail YouTube Maps

Untitled10.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk

```
+ Code + Text
except TypeError:
    # If we have a listlike key, _check_indexing_error will raise
    KeyError: 'Arrival_date'
SEARCH STACK OVERFLOW

[37] #Filling Travel Mins as Zero
data['Travel_Mins'].fillna(0,inplace=True)

[34] #List of different type of columns
categorical=['Airline','Source','Destination','Additional_Info','City1']
numerical=['Total_Stops','Date','Month','Year','Dep_Time_Hour','Dep_Time_Mins','Arrival_Date','Arrival_Time_Hour','Arrival_Time_Mins','Travel_Hou

[ ] #Label Encoder
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()

data.Airline=le.fit_transform(data.Airline)
data.Source=le.fit_transform(data.Source)
```

0s completed at 10:10 AM

(1) WhatsApp x Untitled5.ipynb - Colaboratory x Untitled10.ipynb - Colaboratory x Copy of Copy of Copy of Copy x Optimizing_Flight_Booking_Deci x +

colab.research.google.com/drive/1WDG9L3JX9J9uRvzNG9w55g4DvEaiQbVD#scrollTo=ePNhFxsTfw0o

Google Gmail YouTube Maps

Copy of Copy of Copy of Copy of Welcome To Colaborat...

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

RAM Disk

```
+ Code + Text
#Label Encoder
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.Additional_Info=le.fit_transform(data.Additional_Info)
data.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	City1	City2	City3	Travel_Hours	Travel_Mins
0	3	24/03/2019	0	5	[BLR ? DEL]	22:20	01:10 22 Mar	[2 h , 5 m]	4	7 3897	18	0	0	2	NaN	
1	1	01/05/2019	3	0	[CCU ? IXR ? BBI ? BLR]	5:50	13:15	[7 h , 25 m]	1	7 7662	84	0	0	7	NaN	
2	4	09/06/2019	2	1	[DEL ? LKO ? BOM ? COK]	9:25	04:25 10 Jun	[1 h , 9 m]	1	7 13882	118	0	0	1	NaN	
3	3	12/05/2019	3	0	[CCU ? NAG ? BLR]	18:05	23:30	[5 h , 25 m]	0	7 6218	91	0	0	5	NaN	
4	3	01/03/2019	0	5	[BLR ? NAG ? DEL]	16:50	21:35	[4 h , 45 m]	0	7 13302	29	0	0	4	NaN	

0s completed at 11:42 AM

Untitled5.ipynb - Colaboratory

colab.research.google.com/drive/1mie4LtvZLaGBqcFB51cfvO6yJxOpHm0P#scrollTo=Q0kH8Ho2LFEU

File Edit View Insert Runtime Tools Help

Code Text

```
[6] data=data[['Airline','Source','Destination','Date','Month','Year','Dep_hour','Dep_min','Arrival_hour','Arrival_min','Price']]
data.head()
```

	Airline	Source	Destination	Date	Month	Year	Dep_hour	Dep_min	Arrival_hour	Arrival_min	Price
0	3	0	5	24	03	2019	22	20	1	10	3897
1	1	3	0	01	05	2019	5	50	13	15	7662
2	4	2	1	09	06	2019	9	25	4	25	13882
3	3	3	0	12	05	2019	18	5	23	30	6218
4	3	0	5	01	03	2019	16	50	21	35	13302

```
[4] #Label Encoder
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

Executing (29s) <cell line: 138> predict() > _dense_predict()

11:52 AM 4/8/2023

Untitled10.ipynb - Colaboratory

colab.research.google.com/drive/1g81gAKQ4dtChnewFSv1DQbOhBeknk1I#scrollTo=TBR092v8tOeM

File Edit View Insert Runtime Tools Help All changes saved

Code Text

```
[18] #MILESTONE 3
#Visual Analysis
plt.figure(figsize = (5,5))
plt.title('Count of flights month wise')
ax=sns.countplot(x = 'Month', data = data)
plt.xlabel('Month')
plt.ylabel('Count of flights')

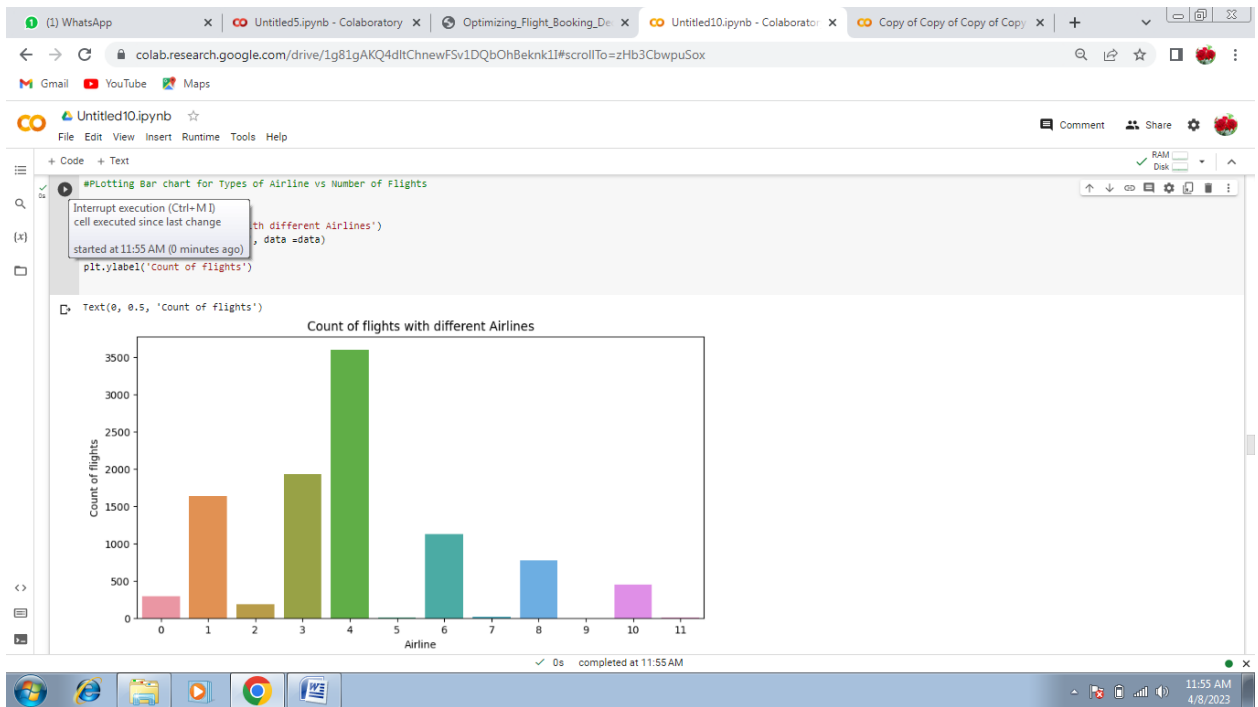
Text(0, 0.5, 'Count of flights')
```

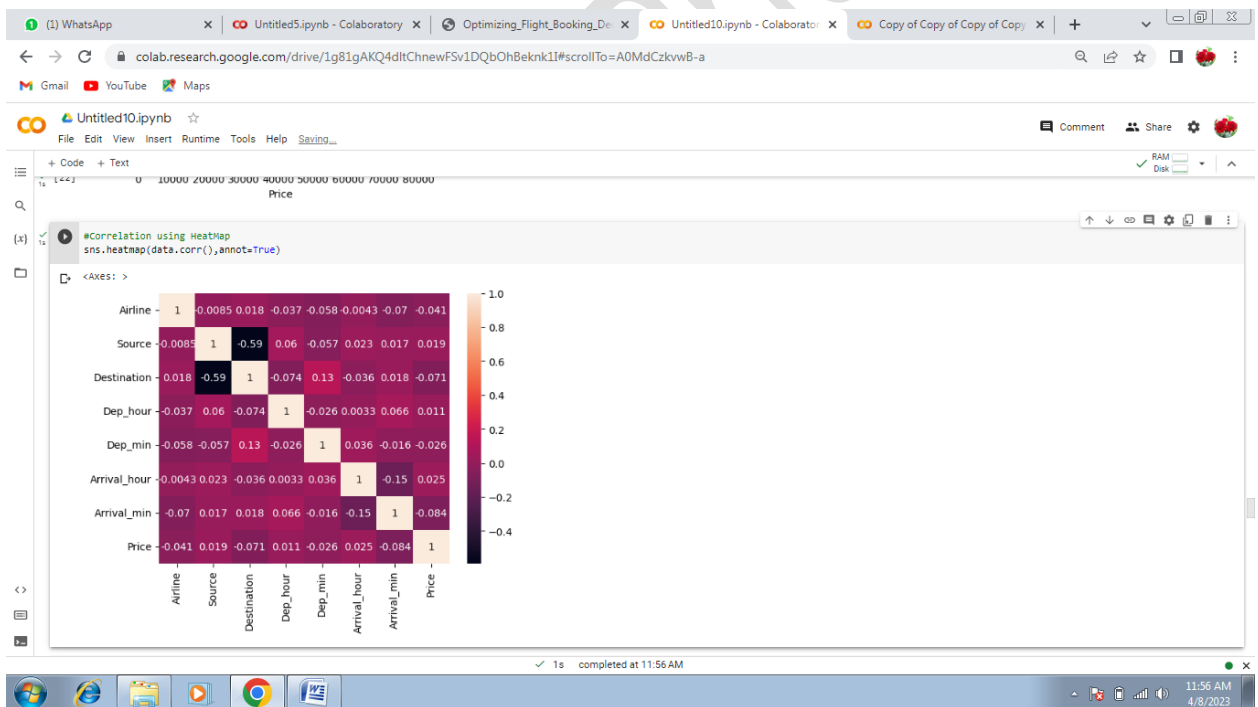
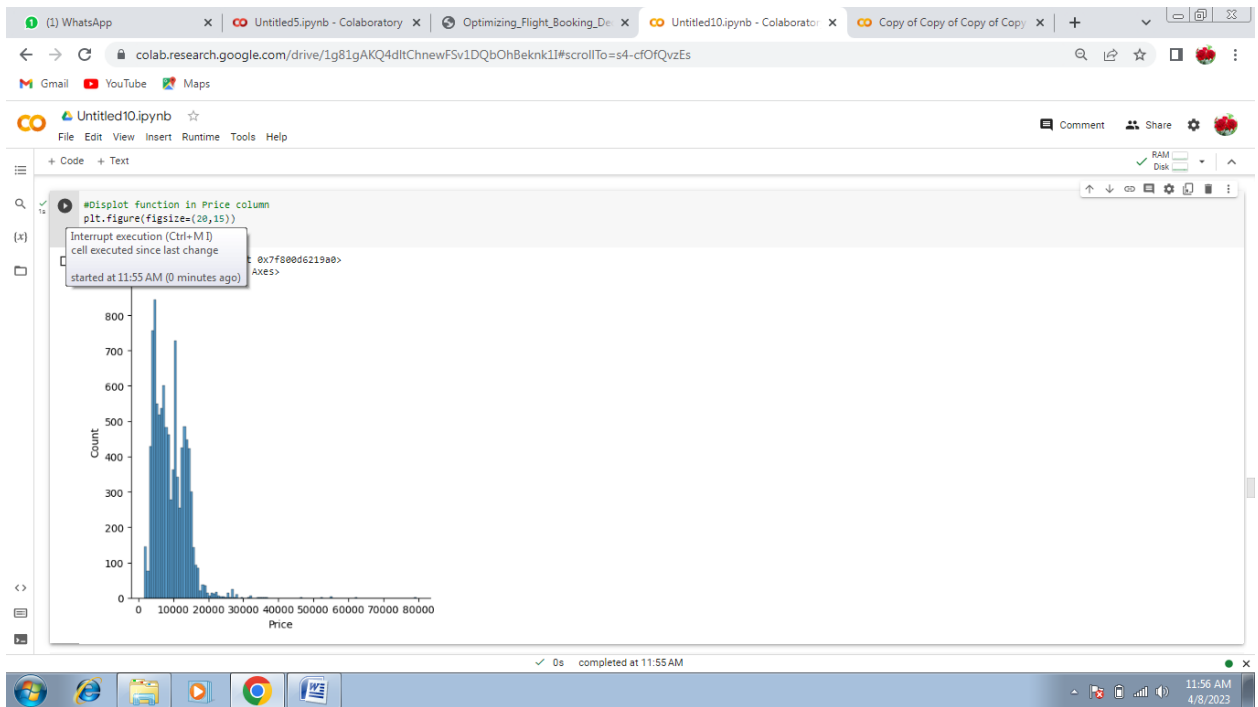
Count of flights month wise

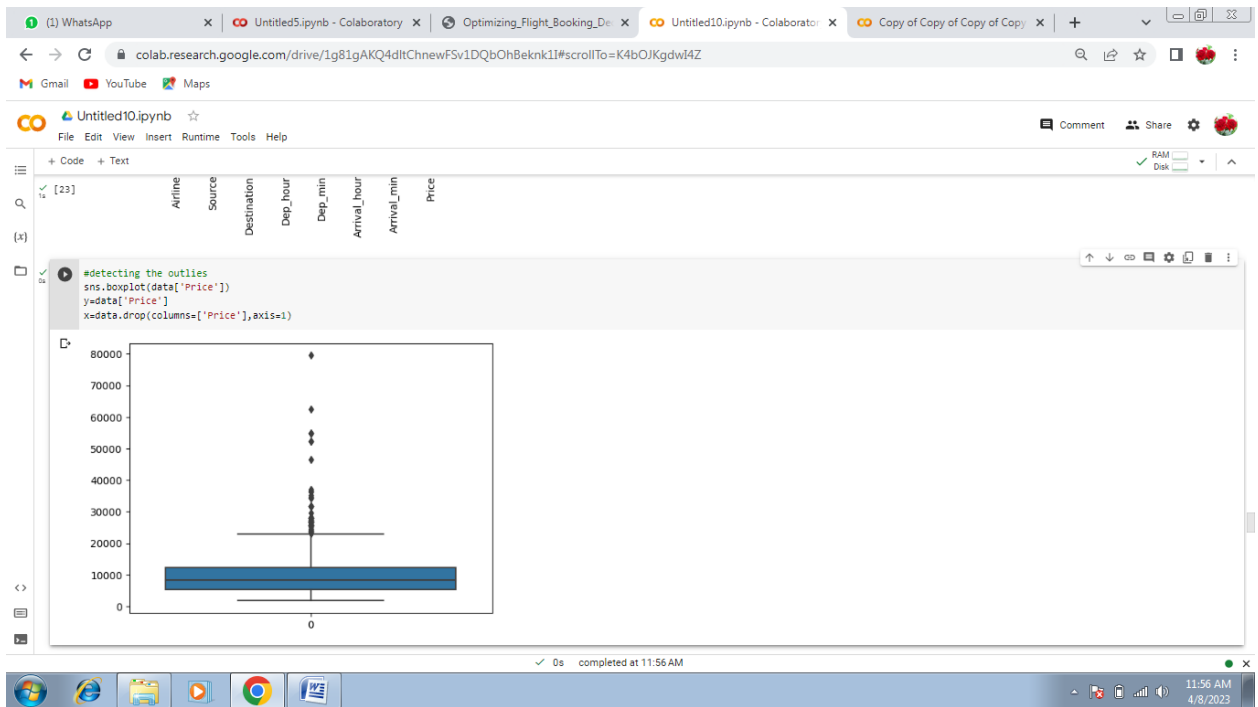
Month	Count of flights
4	1000
1	2000
9	1300
2	900
7	1100
8	800
3	800
5	900
6	1200

completed at 11:53 AM

11:53 AM 4/8/2023







Colaboratory interface showing a Jupyter Notebook with the following code:

```
#scaling the data
ss=StandardScaler()
x_scaled=ss.fit_transform(x)
x_scaled=pd.DataFrame(x_scaled,columns=x.columns)
x_scaled.head()
```

The output is a DataFrame showing the scaled data for the first 5 rows (index 0 to 4). The columns are: Airline, Source, Destination, Date, Month, Year, Dep_hour, Dep_min, Arrival_hour, and Arrival_min.

	Airline	Source	Destination	Date	Month	Year	Dep_hour	Dep_min	Arrival_hour	Arrival_min
0	-0.413142	-1.657964	2.421216	1.237154	-1.468868	0.0	1.660471	-0.234232	-1.798676	-0.889296
1	-1.263757	0.893018	-0.975021	-1.471757	0.249135	0.0	-1.300192	1.365738	-0.048712	-0.586814
2	0.012165	0.042691	-0.295774	-0.529527	1.108137	0.0	-0.603565	0.032429	-1.361185	0.018151
3	-0.413142	0.893018	-0.975021	-0.176191	0.249135	0.0	0.963844	-1.034218	1.409591	0.320633
4	-0.413142	-1.657964	2.421216	-1.471757	-1.468868	0.0	0.615531	1.365738	1.117931	0.623115

The following code is also visible in the notebook:

```
[6] data=data[['Airline','Source','Destination','Date','Month','Year','Dep_hour','Dep_min','Arrival_hour','Arrival_min','Price']]
data.head()
```

Colab interface showing a Jupyter Notebook with a table of flight data. The table has columns: Airline, Source, Destination, Date, Month, Year, Dep_hour, Dep_min, Arrival_hour, Arrival_min. The data is as follows:

Airline	Source	Destination	Date	Month	Year	Dep_hour	Dep_min	Arrival_hour	Arrival_min
1023	4	0	2	12	05	2019	8 20	11	20
263	3	2	1	09	03	2019	4 55	16	10
6235	4	2	1	03	06	2019	19 45	19	0
4049	1	2	1	27	05	2019	3 50	19	15
8451	4	0	5	01	03	2019	22 50	5	5

Code cell output: #Split Train and Test, x_train, x_test, y_train, y_test=train_test_split(x,y), x_train.head(). The output shows the first 5 rows of the training data.

Colab interface showing a Jupyter Notebook with code for building a model. The code is as follows:

```
#MILESTONE 4
#Building the Model
#using ensemble techniques
rfr=RandomForestRegressor()
gb=GradientBoostingRegressor()
ad=AdaBoostRegressor()

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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
        print("Mean Squared Error is->",mean_squared_error(y_pred,y_test))
        print("Root Mean Squared Error is->",(mean_squared_error(y_pred,y_test,squared=False)))
```

The code cell output shows the results of the model building process, including the R2 Score, Mean Absolute Error, and Mean Squared Error for the training and testing data.

Colaboratory interface showing a Jupyter Notebook titled "Untitled5.ipynb". The code cell contains the following Python code:

```
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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
print("MEan Squared Error is->",mean_squared_error(y_pred,y_test))
print("Root Mean Squared Error is->",(mean_squared_error(y_pred,y_test,squared=False)))

GradientBoostingRegressor()
R2 Score is-> 0.6962464685140564
R2 for train Data-> 0.7506500526743249
Mean Absolute Error is-> 1764.36293515023
MEan Squared Error is-> 6671407.67705831
Root Mean Squared Error is-> 2582.90682701841
AdaBoostRegressor()
R2 Score is-> 0.36337321658388244
R2 for train Data-> 0.3889853383076247
Mean Absolute Error is-> 3032.594404315625
MEan Squared Error is-> 13982378.376067594
Root Mean Squared Error is-> 3739.3018567732124
```

The execution status shows "3s completed at 12:01 PM". The system tray at the bottom indicates the time is 12:01 PM on 4/8/2023.

Colaboratory interface showing a Jupyter Notebook titled "Untitled5.ipynb". The code cell contains the following Python code:

```
#Regression Model
#KNN and Svm
knn=KNeighborsClassifier()
svm=SVR()
dt=DecisionTreeRegressor()
for i in[knn,svm,dt]:
    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)<=0.1:
        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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
        print("Mean Squared Error is->",mean_squared_error(y_pred,y_test))
        print("Root Mean Squared Error is->",(mean_squared_error(y_pred,y_test,squared=False)))
```

The execution status shows "11s completed at 12:02 PM". The system tray at the bottom indicates the time is 12:03 PM on 4/8/2023.

Colaboratory interface showing a Jupyter Notebook with the following code and output:

```
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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
print("Mean Squared Error is->",mean_squared_error(y_pred,y_test))
print("Root Mean Squared Error is->",(mean_squared_error(y_pred,y_test,squared=False)))

SVR()
R2 Score is-> -0.024847225947665974
R2 for train Data-> -0.024178300394269492
Mean Absolute Error is-> 3667.6124125047313
Mean Squared Error is-> 22508951.970211305
Root Mean Squared Error is-> 4744.360016926551
```

Execution completed at 12:02 PM.

Colaboratory interface showing a Jupyter Notebook with the following code and output:

```
#Checking cross validaation
from sklearn.model_selection import cross_val_score
for i in range(2,5):
    cv=cross_val_score(rf,x,y,cv=i)
    print(rf,cv.mean())

RandomForestClassifier(n_estimators=10) 0.36313594718943787
RandomForestClassifier(n_estimators=10) 0.36853685368536854
RandomForestClassifier(n_estimators=10) 0.36473505402160866
```

Execution completed at 12:04 PM.

Colaboratory interface showing a Jupyter Notebook session. The browser tabs include WhatsApp, Untitled5.ipynb - Colaboratory, Untitled10.ipynb - Colaboratory, Optimizing Flight Booking De, and Copy of Copy of Copy of Copy. The URL is colab.research.google.com/drive/1mie4LtvZLaGBqcF851cfvO6yJxOpHm0P#scrollTo=i4xDmDg_k8pj. The notebook is titled "Untitled5.ipynb" and has a menu bar with File, Edit, View, Insert, Runtime, Tools, and Help. The code cell contains the following Python code:

```
#Accuracy
rf=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
rf.fit(x_train,y_train)
y_train_pred=rf.predict(x_train)
y_test_pred=rf.predict(x_test)
print("Test Accuracy",r2_score(y_train_pred,y_train))
print("Train Accuracy",r2_score(y_test_pred,y_test))
```

The output of the code cell shows:

```
Test Accuracy 0.9343220194838787
Train Accuracy 0.6824694229574373
```

The status bar indicates the code was completed at 12:06 PM.

Colaboratory interface showing a Jupyter Notebook session. The browser tabs include WhatsApp, Untitled5.ipynb - Colaboratory, Untitled10.ipynb - Colaboratory, Optimizing Flight Booking De, and Copy of Copy of Copy of Copy. The URL is colab.research.google.com/drive/1mie4LtvZLaGBqcF851cfvO6yJxOpHm0P#scrollTo=7SWmaQNvW4. The notebook is titled "Untitled5.ipynb" and has a menu bar with File, Edit, View, Insert, Runtime, Tools, Help, and [All changes saved](#). The code cell contains the following Python code:

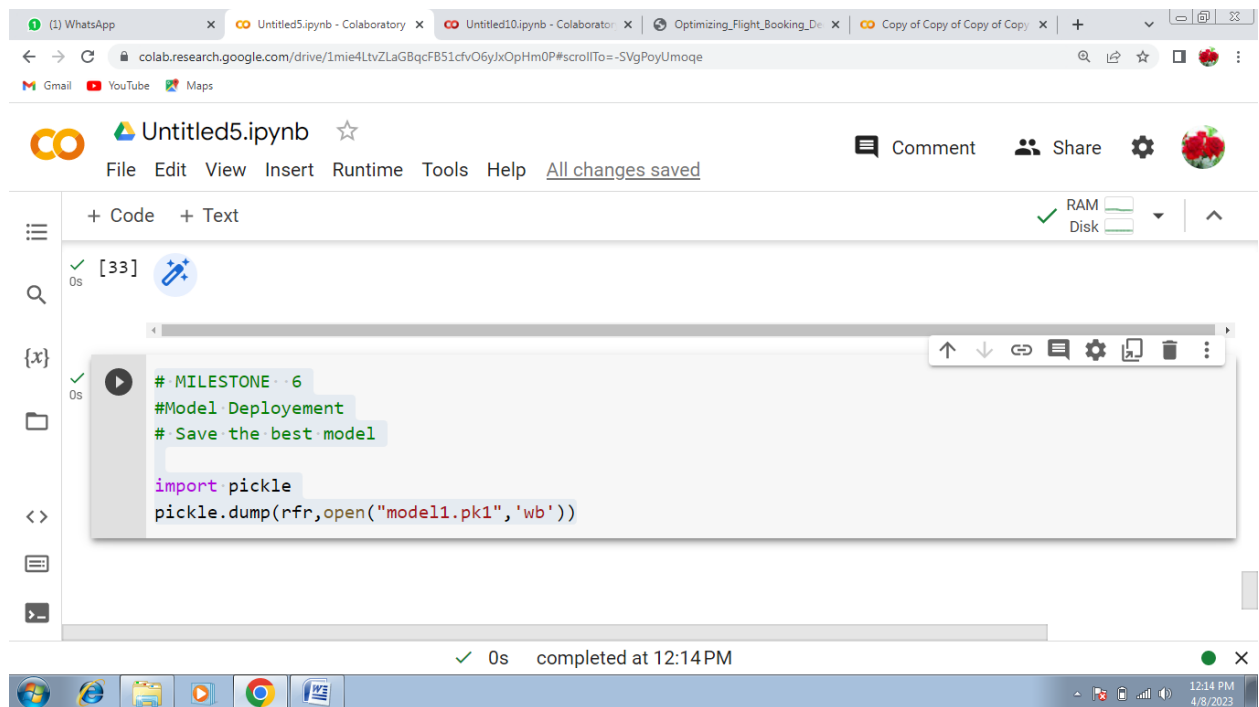
```
knn=KNeighborsClassifier(n_neighbors=2,algorithm='auto',metric_params=None,n_jobs=1)
knn.fit(x_train,y_train)
y_train_pred=knn.predict(x_train)
y_test_pred=knn.predict(x_test)
print("Test Accuracy",r2_score(y_train_pred,y_train))
print("Train Accuracy",r2_score(y_test_pred,y_test))
```

The output of the code cell shows:

```
Test Accuracy 0.9343220194838787
Train Accuracy 0.6824694229574373
```

The status bar indicates the code was completed at 12:07 PM.

```
import pickle
pickle.dump(rfr,open("model1.pk1",'wb'))
```



#importing libraries

```
import matplotlib.pyplot as plt
import numpy as nm
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
#from sklearn.preprocessing import labelEncoder
from sklearn.ensemble import
RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from scipy import stats
import warnings
```

```
import pickle

warnings.filterwarnings('ignore')
```

#Read the Dataset

```
data=pd.read_csv("/content/drive/MyDrive/Data_train.csv")

data.head()
```

```
for i in data:
```

```
    print(i,data[i].unique())
```

#Checking value in Destination

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

data.info()

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

data.Date_of_Journey
```

#Treating the Data Column

```
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.head()

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.head()

#data.dropna(inplace=True)

#data.isnull().sum()

#In the similar manner, we split the Dep_time column, and create separate
columns for departure hours and minutes
```



```
data['Dep_hour'] = pd.to_datetime(data['Dep_Time']).dt.hour
data['Dep_min'] = pd.to_datetime(data['Dep_Time']).dt.minute
data.drop('Dep_Time',axis=1,inplace=True)
```

```
data['Arrival_hour'] = pd.to_datetime(data['Arrival_Time']).dt.hour
data['Arrival_min'] = pd.to_datetime(data['Arrival_Time']).dt.minute
data.drop('Arrival_Time',axis=1,inplace=True)
data.head()
```

```
data.Duration=data.Duration.str.split("")
data['Travel_Hours']=data.Duration.str[1]
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[0]
data['Travel_Mins']=data.Travel_Mins.str.split('m')
data['Travel_Mins']=data.Travel_Mins.str[1]
data.head()
data.Additional_Info.unique()
data.isnull().sum()
categorical=['Airline','Source','Destination','Additional_Info','City1']
numerical=['Total_Stops','Date','Month','Year','Dep_Time_Hour','Dep_Time_Mins','Arrival_Date','Arrival_Time_Hour','Arrival_Time_Mins','Travel_Hours','Travel_Mins']
data.head()
```

#Label Encoder

```
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.Additional_Info=le.fit(data.Additional_Info)
data.head(10)

data=data[['Airline','Source','Destination','Date','Month','Year','Dep_hour','Dep_min','
Arrival_hour','Arrival_min','Price']]

data.head()
```

#Descriptive Stastical

```
data.describe()
```

#Visual Analysis

```
c=1
plt.figure(figsize=(20,45))
categorical=['Airline','Source','Destination','Additional_Info']
#for i in categorical:
# plt.subplot(6,3,c)
#sns.countplot(x=data[i])
#plt.xticks(rotation=90)
#plt.tight_layout(pad=3.0)
#c=c+1
#plt.show()
#Displot function in Price column
plt.figure(figsize=(15,8))
sns.displot(data.Price)
```

#Correlation using HeatMap

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

#detecting the outliers

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

#scalling the data

```
ss=StandardScaler()
```

```
x_scaled=ss.fit_transform(x)
x_scaled=pd.DataFrame(x_scaled,columns=x.columns)
x_scaled.head()
```

#Split Train and Test

```
x_train,x_test,y_train,y_test=train_test_split(x,y)
x_train.head()
```

#using ensemble techniques

```
rf=RandomForestRegressor()
gb=GradientBoostingRegressor()
ad=AdaBoostRegressor()
```

```
for i in[rf,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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
        print("MEan Squared Error is->",mean_squared_error(y_pred,y_test))
        print("Root Mean Squared Error is-
>",(mean_squared_error(y_pred,y_test,squared=False)))
```

#KNN and Svm

```
knn=KNeighborsClassifier()
```

```

svm=SVR()
dt=DecisionTreeRegressor()
for i in[knn,svm,dt]:
    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)<=0.1:
        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 Absolute Error is->",mean_absolute_error(y_pred,y_test))
        print("Mean Squared Error is->",mean_squared_error(y_pred,y_test))
        print("Root Mean Squared Error is-
>",(mean_squared_error(y_pred,y_test,squared=False)))

```

```

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

```

#Accuracy

```

rf=RandomForestRegressor(n_estimators=10,max_features='sqrt',max_depth=None)
rf.fit(x_train,y_train)
y_train_pred=rf.predict(x_train)
y_test_pred=rf.predict(x_test)
print("Test Accuracy",r2_score(y_train_pred,y_train))
print("Train Accuracy",r2_score(y_test_pred,y_test))

```

```

knn=KNeighborsClassifier(n_neighbors=2,algorithm='auto',metric_params=None,n_j
obs=1)
knn.fit(x_train,y_train)

```

```
y_train_pred=rf.predict(x_train)
y_test_pred=rf.predict(x_test)
print("Test Accuracy",r2_score(y_train_pred,y_train))
print("Train Accuracy",r2_score(y_test_pred,y_test))

#Evaluating the performance

#price_list=pd.DataFrame({'Price':data})

#price_list

#Save the model

pickle.dump(rf,open('model1.pkl','wb'))

data.head()
```