

IDENTIFYING SATISFACTION OF AIRLINE PASSENGERS

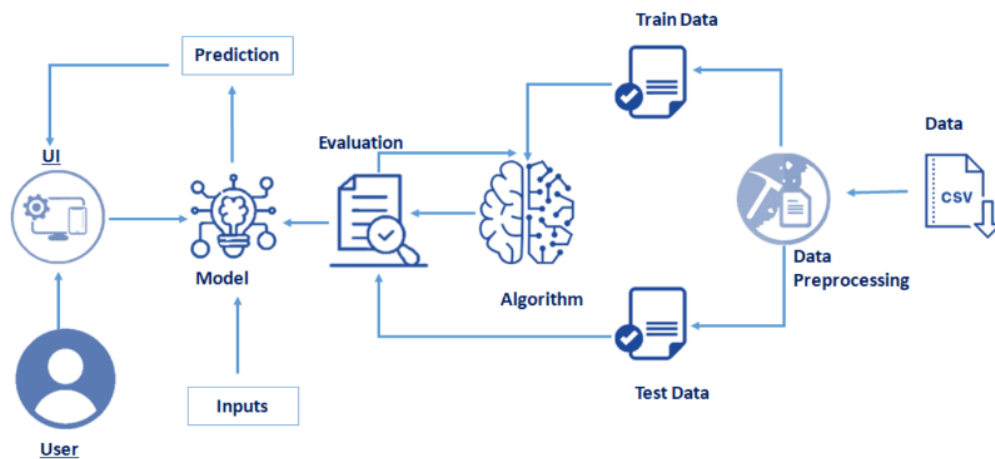
The primary objective in identifying the satisfaction of airline passengers is to cultivate an environment of continual improvement and excellence in the air travel experience. Airlines aim to understand and meet the expectations of their passengers comprehensively, recognizing that content and satisfied customers are more likely to remain loyal and contribute positively to the airline's reputation. them into one of the three categories. One key aspect is predictive analysis, where AI algorithms can anticipate passenger needs and preferences based on historical data. By analyzing past interactions, travel patterns, and feedback, AI systems can predict potential pain points and areas for improvement, allowing airlines to proactively address issues and enhance passenger satisfaction.

Identifying the satisfaction of airline passengers is driven by several compelling reasons that are pivotal for the success and sustainability of airlines:

- 1)Customer Loyalty and Retention: Satisfied passengers are more likely to remain loyal to an airline, choosing it for future travels and contributing to long-term customer retention.
- 2)Reputation Management: Positive passenger experiences lead to favourable reviews and recommendations, enhancing the airline's reputation and attracting new customers.
- 3)Competitive Advantage: Airlines with higher passenger satisfaction levels often outperform competitors, gaining a competitive edge in the industry.
- 4)Operational Efficiency: Feedback from passengers helps identify specific areas of improvement, allowing airlines to optimize processes and enhance overall operational efficiency.
- 5)Revenue Growth: Satisfied passengers are more likely to make repeat purchases, use additional services, and contribute to increased revenue for the airline.
- 6) Cost Reduction: Proactively addressing passenger concerns and enhancing satisfaction can reduce costs associated with customer complaints, compensation, and service failures.

Let us look at the Technical Architecture of the project.

Technical Architecture:



Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI to accomplish this, we have to complete all the activities listed below,

➤ Data Collection & Preparation

- o Collect the dataset
- o Data Preparation
- o Exploratory Data Analysis

➤ Descriptive statistical

- o Visual Analysis

➤ Model Building

- o Training the model in multiple algorithms
- o Testing the model

➤ Performance Testing

o Testing model with multiple evaluation metrics

➤ Model Deployment

o Save the best model

o Integrate with Web Framework

➤ Project Demonstration & Documentation

o Record explanation Video for project end to end solution

Project Objectives:

By the end of this project, you will:

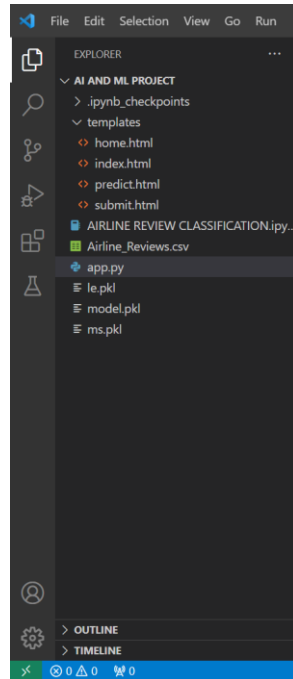
a) Understand the fundamental concepts and techniques of predictive analysis in the context of identifying satisfaction of airline passengers. b) Gain a comprehensive understanding of the factors influencing passenger satisfaction and how they can be analyzed using predictive modelling.

c) Acquire knowledge on preprocessing and cleaning diverse datasets related to airline passenger feedback, employing various data preprocessing techniques tailored to predictive analysis.

d) Develop proficiency in building and training predictive models that can forecast passenger satisfaction levels based on historical and real-time data.

Project Structure:

Create the Project folder which contains files as shown below:



Define Problem / Problem Understanding Activity 1: Specify the business problem:

Identifying satisfaction of airline passengers

Activity 2: Business requirements:

Here are some potential business requirements for identifying the satisfaction of airline passengers:

1)Comprehensive Passenger Feedback System:

Implementation of a robust system for collecting feedback from passengers across all stages of the travel experience. Objective: By capturing insights into booking, check-in, boarding, in-flight services, and post-flight experiences, the airline can identify specific areas for improvement and enhance overall satisfaction.

2)Real-time Feedback Collection:

Integration of mechanisms for instant feedback during various travel stages, including inflight and post-flight. Objective: Swift response to passenger concerns, allowing the airline to make immediate improvements and adapt to evolving customer expectations in real-time.

3)Sentiment Analysis:

Utilization of sentiment analysis tools to assess passenger comments and feedback. Objective: Rapid identification of both areas of concern and positive

aspects, enabling targeted improvements and amplification of successful elements for an enhanced travel experience.

Activity 3: Literature Survey (Student Will Write)

A literature survey would involve researching and reviewing existing studies, articles, and other publications on the topic of project. The survey would aim to gather information on current systems, their strengths and weaknesses, and any gaps in knowledge that the project could address. The literature survey would also look at the methods and techniques used in previous projects, and any relevant data or findings that could inform the design and implementation of the current project.

Activity 4: Social or Business Impact:

1)Enhanced Customer Experience:

A focus on passenger satisfaction leads to an improved overall travel experience for individuals. This positively influences social well-being by reducing stress and frustration associated with air travel, contributing to more positive and enjoyable journeys.

2)Increased Loyalty and Positive Reputation:

Satisfied passengers are more likely to become repeat customers and brand advocates. By consistently meeting or exceeding passenger expectations, airlines can build loyalty, leading to positive word-of-mouth and an enhanced reputation in the market.

3)Competitive Advantage:

Airlines that prioritize passenger satisfaction gain a competitive edge. Positive customer experiences differentiate an airline in a crowded market, attracting new customers and retaining existing ones in an industry where service quality plays a pivotal role in consumer choices.

Activity 1: IMPORTING THE LIBRARIES:

IMPORTING THE LIBRARIES

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
from imblearn.over_sampling import SMOTE
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report
```

Here we are importing the pandas library as pd, numpy as np, matplotlib.pyplot as plt, seaborn as sns, and various modules from scikit-learn and imbalanced-learn libraries for the tasks such as model selection, preprocessing, and machine learning.

Activity 1.1: Reading the dataset:

```
In [2]: data=pd.read_csv("Airline_Reviews.csv") #Reading the dataset
```

```
In [3]: data.shape #getting the no.of rows and columns
```

```
Out[3]: (23171, 20)
```

i) The data.shape command in Pandas is used to display the number of rows and columns.

ii) The data.head() command is used to display the first few rows of the Pandas DataFrame named data.

```
In [4]: data.head() #reading the dataset
```

```
Out[4]:
```

	Unnamed: 0	Airline Name	Overall_Rating	Review_Title	Review Date	Verified	Review	Aircraft	Type Of Traveller	Seat Type	Route	Date Flown	Seat Comfort	Cabin Staff Service	Fi Beve
0	0	AB Aviation	9	"pretty decent airline"	11th November 2019	True	Moroni to Moheli. Turned out to be a pretty ...	NaN	Solo Leisure	Economy Class	Moroni to Moheli	November 2019	4.0	5.0	
1	1	AB Aviation	1	"Not a good airline"	25th June 2019	True	Moroni to Anjouan. It is a very small airline...	E120	Solo Leisure	Economy Class	Moroni to Anjouan	June 2019	2.0	2.0	
2	2	AB Aviation	1	"flight was fortunately short"	25th June 2019	True	Anjouan to Dzaoudzi. A very small airline an...	Embraer E120	Solo Leisure	Economy Class	Anjouan to Dzaoudzi	June 2019	2.0	1.0	
3	3	Adria Airways	1	"I will never fly again with Adria"	28th September 2019	False	Please do a favor yourself and do not fly wi...	NaN	Solo Leisure	Economy Class	Frankfurt to Pristina	September 2019	1.0	1.0	
4	4	Adria Airways	1	"it ruined our last days of holidays"	24th September 2019	True	Do not book a flight with this airline! My fr...	NaN	Couple Leisure	Economy Class	Sofia to Amsterdam via Ljubljana	September 2019	1.0	1.0	

iii) The data.info() command in Pandas provides a concise summary of the DataFrame.

```
In [5]: data.info() #getting the info of data set

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23171 entries, 0 to 23170
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  --
0   Unnamed: 0            23171 non-null  int64
1   Airline Name          23171 non-null  object
2   Overall_Rating        23171 non-null  object
3   Review_Title          23171 non-null  object
4   Review Date          23171 non-null  object
5   Verified              23171 non-null  bool
6   Review               23171 non-null  object
7   Aircraft              7129 non-null   object
8   Type Of Traveller    19433 non-null  object
9   Seat Type            22075 non-null  object
10  Route                19343 non-null  object
11  Date Flown           19417 non-null  object
12  Seat Comfort         19016 non-null  float64
13  Cabin Staff Service  18911 non-null  float64
14  Food & Beverages     14500 non-null  float64
15  Ground Service       18378 non-null  float64
16  Inflight Entertainment 10829 non-null  float64
17  Wifi & Connectivity  5920 non-null   float64
18  Value For Money      22105 non-null  float64
19  Recommended          23171 non-null  object
dtypes: bool(1), float64(7), int64(1), object(11)
memory usage: 3.4+ MB
```

iv) dropping the columns to get efficient model.

In: `df = data.drop(['Inflight Entertainment', 'Wifi & Connectivity', 'Aircraft', 'Value For Money', 'Cabin Staff Service', 'Unnamed: 0', 'Review Date', 'Review_Title', 'Review'], axis=1)`

The above mentioned columns are dropped from the dataset.

`df.head()` of the modified dataset:

```
In [7]: df.head() #modified dataset

Out[7]:
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Date Flown	Seat Comfort	Food & Beverages	Ground Service	Recommended
0	AB Aviation	9	True	Solo Leisure	Economy Class	Moroni to Moheli	November 2019	4.0	4.0	4.0	yes
1	AB Aviation	1	True	Solo Leisure	Economy Class	Moroni to Anjouan	June 2019	2.0	1.0	1.0	no
2	AB Aviation	1	True	Solo Leisure	Economy Class	Anjouan to Dzaoudzi	June 2019	2.0	1.0	1.0	no
3	Adria Airways	1	False	Solo Leisure	Economy Class	Frankfurt to Pristina	September 2019	1.0	NaN	1.0	no
4	Adria Airways	1	True	Couple Leisure	Economy Class	Sofia to Amsterdam via Ljubljana	September 2019	1.0	1.0	1.0	no

v) `df.info()` command in Pandas provides a concise summary of the DataFrame.[to get info of updated dataset]

```
In [8]: df.info() #to get info of updated dataset

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23171 entries, 0 to 23170
Data columns (total 11 columns):
#   Column                Non-Null Count  Dtype
---  --
0   Airline Name          23171 non-null  object
1   Overall_Rating        23171 non-null  object
2   Verified              23171 non-null  bool
3   Type Of Traveller    19433 non-null  object
4   Seat Type            22075 non-null  object
5   Route                19343 non-null  object
6   Date Flown           19417 non-null  object
7   Seat Comfort         19016 non-null  float64
8   Food & Beverages     14500 non-null  float64
9   Ground Service       18378 non-null  float64
10  Recommended          23171 non-null  object
dtypes: bool(1), float64(3), object(7)
memory usage: 1.8+ MB
```

Activity 2: DATA PREPARATION

Activity 2.1: CHECKING FOR NULL VALUES AND HANDLING NULL VALUES:

CHECKING FOR NULL VALUES:

i) `df.isnull().any()`:

The `df.isnull().any()` expression is used to check if there are any null (missing) values in each column of the DataFrame `df`.

```
In [9]: df.isnull().any() #checking for null values

Out[9]: Airline Name      False
Overall_Rating      False
Verified            False
Type Of Traveller    True
Seat Type           True
Route              True
Date Flown          True
Seat Comfort        True
Food & Beverages     True
Ground Service      True
Recommended         False
dtype: bool
```

ii) `df.isnull().sum()`:

The `df.isnull().sum()` expression is used to calculate the total number of null (missing) values in each column of the DataFrame `df`.

```
In [10]: df.isnull().sum() #checking for the sum of null values

Out[10]: Airline Name      0
Overall_Rating      0
Verified            0
Type Of Traveller    3738
Seat Type           1096
Route              3828
Date Flown          3754
Seat Comfort        4155
Food & Beverages     8671
Ground Service      4793
Recommended         0
dtype: int64
```

By observation, looks like there are some null values, so to ensure accuracy and reliability we handle those null values.

HANDLING NULL VALUES:

iii) There are null values in some columns so we have to modify the dataset categorical values are replace with mode and numerical are replace with mean.

e.g; `df["Type Of Traveller"].fillna(df["Type Of Traveller"].mode()[0], inplace=True)`

```
In [11]: #There is null values in some columns so we have to modify the dataset categorical values are replace with mode and numerical are
df["Type Of Traveller"].fillna(df["Type Of Traveller"].mode()[0],inplace=True)
df["Seat Type"].fillna(df["Seat Type"].mode()[0],inplace=True)
df["Route"].fillna(df["Route"].mode()[0],inplace=True)
df["Date Flown"].fillna(df["Date Flown"].mode()[0],inplace=True)
df["Seat Comfort"].fillna(df["Seat Comfort"].mean(),inplace=True)
df["Food & Beverages"].fillna(df["Food & Beverages"].mean(),inplace=True)
df["Ground Service"].fillna(df["Ground Service"].mean(),inplace=True)
```

CHECKING FOR NULL VALUES:

We have to check whether the null values are present or not in the modified data.

iv) `df.isnull().any()`:

```
In [12]: df.isnull().any() #checking again there is any null values
Out[12]: Airline Name      False
Overall_Rating    False
Verified           False
Type Of Traveller  False
Seat Type          False
Route             False
Date Flown         False
Seat Comfort       False
Food & Beverages   False
Ground Service     False
Recommended        False
dtype: bool
```

Activity 2.2: CONVERTING THE CATEGORICAL COLUMN TO NUMERIC USING LABEL ENCODING:

i) `le=LabelEncoder()`, introducing a new object for label encoding:

```
In [13]: le=LabelEncoder() #creating an object for label encoding
```

ii) `df.head()`:

The `df.head()` command in Pandas is used to display the first few rows of the DataFrame `df`.

```
In [14]: df.head()
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Date Flown	Seat Comfort	Food & Beverages	Ground Service	Recommended
0	AB Aviation	9	True	Solo Leisure	Economy Class	Moroni to Moheli	November 2019	4.0	4.000000	4.0	yes
1	AB Aviation	1	True	Solo Leisure	Economy Class	Moroni to Anjouan	June 2019	2.0	1.000000	1.0	no
2	AB Aviation	1	True	Solo Leisure	Economy Class	Anjouan to Dzaoudzi	June 2019	2.0	1.000000	1.0	no
3	Adria Airways	1	False	Solo Leisure	Economy Class	Frankfurt to Pristina	September 2019	1.0	2.553586	1.0	no
4	Adria Airways	1	True	Couple Leisure	Economy Class	Sofia to Amsterdam via Ljubljana	September 2019	1.0	1.000000	1.0	no

iii) `df[['Month flown','Year flown']]=df['Date Flown'].str.split(expand=True)`:

To split the values in the 'Date Flown' column of the DataFrame `df` and then assigning the resulting substrings to new columns 'Month flown' and 'Year flown'.

```
In [15]: df[['Month flown','Year flown']]=df['Date Flown'].str.split(expand=True) #splitting the string
```

iv) `df=df.drop(["Date Flown"],axis=1)`:

To remove the "Date Flown" column from the DataFrame `df`.

```
In [16]: df=df.drop(["Date Flown"],axis=1) #dropping the date flown column
```

v) order=['Airline Name','Overall_Rating','Verified','Type Of Traveller','Seat Type','Route','Month flown','Year flown','Seat Comfort','Food & Beverages','Ground Service','Recommended']

df=df.reindex(columns=order):

To reorder the columns of the DataFrame df based on the specified list of column names in the order list.

```
In [18]: #we are converting the columns by using Label Encoder
df['Airline Name']=le.fit_transform(df['Airline Name'])
df['Overall_Rating']=le.fit_transform(df['Overall_Rating'])
df['Verified']=le.fit_transform(df['Verified'])
df['Type Of Traveller']=le.fit_transform(df['Type Of Traveller'])
df['Seat Type']=le.fit_transform(df['Seat Type'])
df['Route']=le.fit_transform(df['Route'])
df['Month flown']=le.fit_transform(df['Month flown'])
df['Recommended']=le.fit_transform(df['Recommended'])
```

vi) df.head():

The df.head() command in Pandas is used to display the first few rows of the DataFrame df.

```
In [19]: df.head()
Out[19]:
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Year flown	Seat Comfort	Food & Beverages	Ground Service	Recommended
0	0	8	1	3	1	8585	9	2019	4.0	4.000000	4.0	1
1	0	0	1	3	1	8584	6	2019	2.0	1.000000	1.0	0
2	0	0	1	3	1	482	6	2019	2.0	1.000000	1.0	0
3	4	0	0	3	1	4256	11	2019	1.0	2.553586	1.0	0
4	4	0	1	1	1	11778	11	2019	1.0	1.000000	1.0	0

Exploratory Data Analysis

Activity 1: VISUALIZATION OF DATA:

i)df.describe():

The df.describe() is a Pandas DataFrame method that provides statistical summaries of the numerical columns in the DataFrame.

```
In [20]: df.describe() #it gives the statical calculation of data
```

```
Out[20]:
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Seat Comfort	Food & Beverages	Ground Service	Recomr
count	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171.000000	23171
mean	245.027880	2.393423	0.531785	2.008675	0.973242	7016.380130	5.495188	2.618321	2.553586	2.353738	0
std	144.313766	3.104117	0.498999	1.077595	0.457584	3612.341934	3.019861	1.327017	1.207396	1.421145	0
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0
25%	124.000000	0.000000	0.000000	1.000000	1.000000	4101.500000	3.000000	1.000000	2.000000	1.000000	0
50%	237.000000	0.000000	1.000000	2.000000	1.000000	8093.000000	6.000000	2.618321	2.553586	2.353738	0
75%	373.000000	5.000000	1.000000	3.000000	1.000000	9482.000000	7.000000	4.000000	3.000000	3.000000	1
max	496.000000	9.000000	1.000000	3.000000	3.000000	13606.000000	11.000000	5.000000	5.000000	5.000000	1

ii) `df.corr(numeric_only=True)`:

The `df.corr()` method in Pandas is used to calculate the correlation coefficients between numerical columns in a DataFrame.

```
In [21]: df.corr(numeric_only=True) #corretion between the columns
```

```
Out[21]:
```

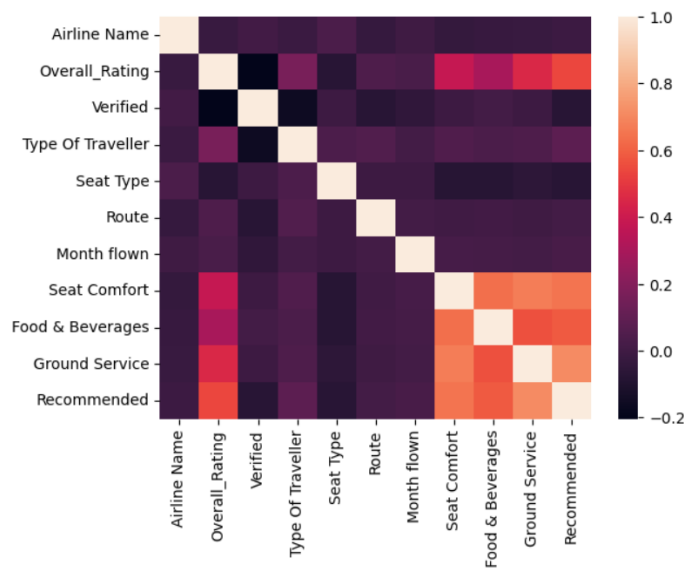
	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Seat Comfort	Food & Beverages	Ground Service	Recommended
Airline Name	1.000000	-0.022610	0.006762	-0.018686	0.032295	-0.030981	-0.006680	-0.035857	-0.026203	-0.023924	-0.015614
Overall_Rating	-0.022610	1.000000	-0.204288	0.163941	-0.076057	0.041676	0.027643	0.382214	0.295038	0.450949	0.534467
Verified	0.006762	-0.204288	1.000000	-0.159073	-0.008561	-0.076355	-0.053120	-0.009543	0.009813	-0.009349	-0.073313
Type Of Traveller	-0.018686	0.163941	-0.159073	1.000000	0.036095	0.049901	0.009462	0.048675	0.033289	0.041037	0.087046
Seat Type	0.032295	-0.076057	-0.008561	0.036095	1.000000	-0.008572	-0.007745	-0.078122	-0.080341	-0.061260	-0.076048
Route	-0.030981	0.041676	-0.076355	0.049901	-0.008572	1.000000	0.012067	-0.000635	0.006327	-0.003002	0.010862
Month flown	-0.006680	0.027643	-0.053120	0.009462	-0.007745	0.012067	1.000000	0.021298	0.015861	0.011897	0.025007
Seat Comfort	-0.035857	0.382214	-0.009543	0.048675	-0.078122	-0.000635	0.021298	1.000000	0.635971	0.672536	0.651380
Food & Beverages	-0.026203	0.295038	0.009813	0.033289	-0.080341	0.006327	0.015861	0.635971	1.000000	0.558308	0.581473
Ground Service	-0.023924	0.450949	-0.009349	0.041037	-0.061260	-0.003002	0.011897	0.672536	0.558308	1.000000	0.709446
Recommended	-0.015614	0.534467	-0.073313	0.087046	-0.076048	0.010862	0.025007	0.651380	0.581473	0.709446	1.000000

iii) `sns.heatmap(df.corr(numeric_only=True))`:

The `sns.heatmap()` function from the Seaborn library is used to visualize the correlation matrix as a heatmap. The heatmap visually represents the strength and direction of the relationships between numerical columns in the DataFrame.

```
In [22]: sns.heatmap(df.corr(numeric_only=True)) #it will gives relation between the columns
```

```
Out[22]: <Axes: >
```

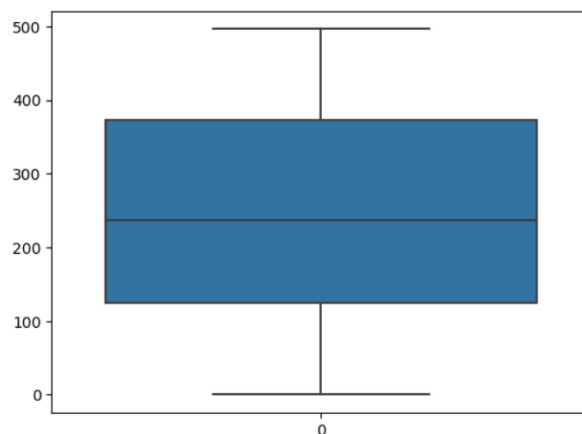


iv) `sns.boxplot(df["Airline Name"])`:

The `sns.boxplot()` function from the Seaborn library is used to create a boxplot for the distribution of the “Airline Name” column variable.

```
In [23]: sns.boxplot(df["Airline Name"])
```

```
Out[23]: <Axes: >
```



v) `df.head()`:

The `df.head()` command in Pandas is used to display the first few rows of the DataFrame `df`.

```
In [24]: df.head()
```

```
Out[24]:
```

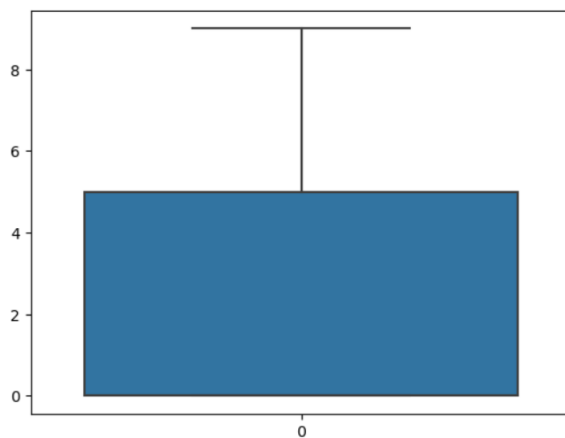
	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Year flown	Seat Comfort	Food & Beverages	Ground Service	Recommended
0	0	8	1	3	1	8585	9	2019	4.0	4.000000	4.0	1
1	0	0	1	3	1	8584	6	2019	2.0	1.000000	1.0	0
2	0	0	1	3	1	482	6	2019	2.0	1.000000	1.0	0
3	4	0	0	3	1	4256	11	2019	1.0	2.553586	1.0	0
4	4	0	1	1	1	11778	11	2019	1.0	1.000000	1.0	0

vi) `sns.boxplot(df["Overall_Rating"]):`

The `sns.boxplot()` function from the Seaborn library is used to create the boxplot for the “Overall_Rating” column variable.

```
In [25]: sns.boxplot(df["Overall_Rating"]) #to find outliers we are plotting the boxplot
```

```
Out[25]: <Axes: >
```

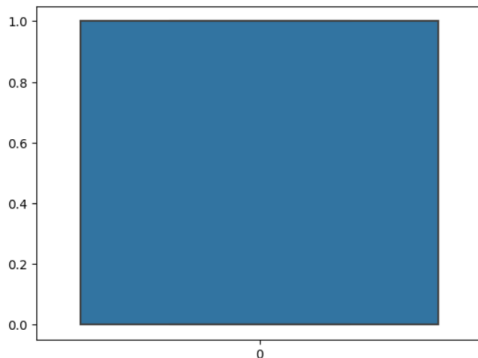


vii) `sns.boxplot(df['Verified']):`

The `sns.boxplot()` function from the Seaborn library is used to create the boxplot for the “Verified” column variable.

```
In [26]: sns.boxplot(df['Verified'])
```

```
Out[26]: <Axes: >
```

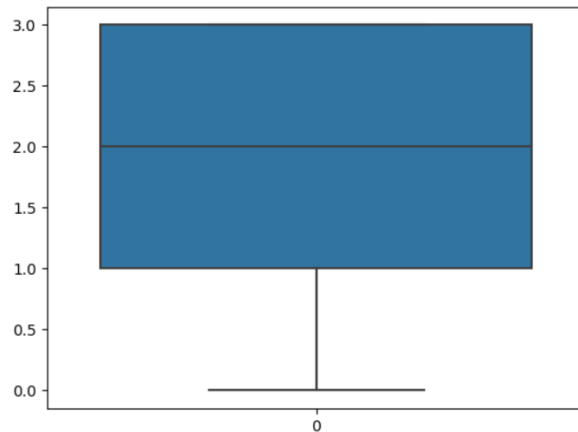


vii) `sns.boxplot(df['Type Of Traveller'])`

The `sns.boxplot()` function from the Seaborn library is used to create the boxplot for the “Type Of Traveller” column variable.

```
In [27]: sns.boxplot(df['Type Of Traveller'])
```

```
Out[27]: <Axes: >
```

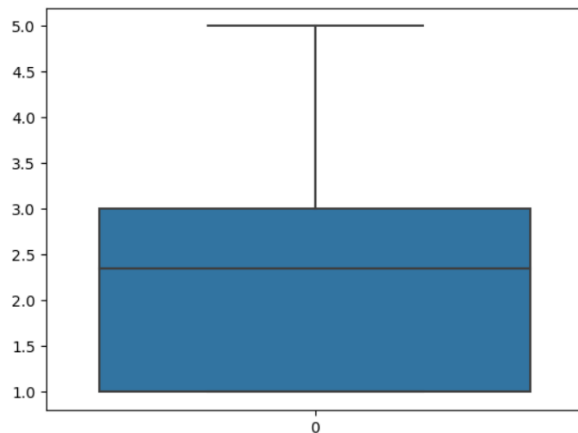


viii) `sns.boxplot(df['Ground Service'])`

The `sns.boxplot()` function from the Seaborn library is used to create the boxplot for the “Ground Service” column variable.

```
In [28]: sns.boxplot(df['Ground Service'])
```

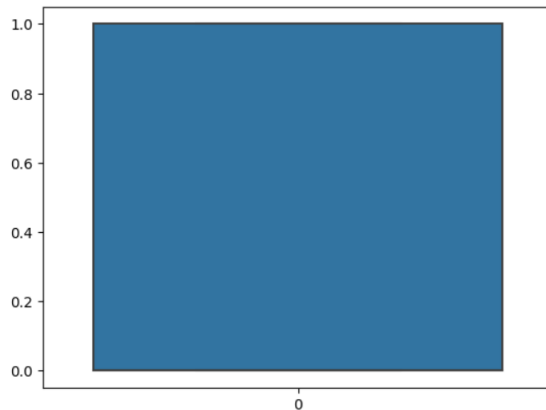
```
Out[28]: <Axes: >
```



ix) `sns.boxplot(df['Recommended'])`

The `sns.boxplot()` function from the Seaborn library is used to create the boxplot for the “Recommended” column variable.

```
In [29]: sns.boxplot(df["Recommended"])
Out[29]: <Axes: >
```



Activity 2: SPLITTING THE INDEPENDENT AND DEPENDENT VARIABLE:

i)x=df.iloc[:,11] #independent variable
y=df["Recommended"] #dependent variable

```
In [30]: x=df.iloc[:,11] #independent variable
y=df["Recommended"] #dependent variable
```

ii) x.head():

The x.head() is used to display the first few rows of the DataFrame x.

```
In [31]: x.head()
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Year flown	Seat Comfort	Food & Beverages	Ground Service
0	0	8	1	3	1	8585	9	2019	4.0	4.000000	4.0
1	0	0	1	3	1	8584	6	2019	2.0	1.000000	1.0
2	0	0	1	3	1	482	6	2019	2.0	1.000000	1.0
3	4	0	0	3	1	4256	11	2019	1.0	2.553586	1.0
4	4	0	1	1	1	11778	11	2019	1.0	1.000000	1.0

iii) y.head()

The y.head() is used to display the first few rows of the DataFrame y.

```
In [32]: y.head()
Out[32]: 0    1
1    0
2    0
3    0
4    0
Name: Recommended, dtype: int32
```

iv)x.shape()

The x.shape is used to determine the number of rows and columns of the DataFrame x.

```
In [33]: x.shape
Out[33]: (23171, 11)
```

v)y.shape()

The y.shape is used to determine the number of rows and columns of the DataFrame y.

```
In [34]: y.shape
Out[34]: (23171,)
```

vi) y.value_counts():

The y.value_counts() is used to get a count of unique values in the DataFrame column represented by the variable y.

```
In [35]: y.value_counts()
Out[35]: 0    15364
         1     7807
         Name: Recommended, dtype: int64
```

vii) ms=MinMaxScaler()

x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns) #feature scaling

x_scaled:

To standardize the range of independent variables.

```
In [36]: ms=MinMaxScaler()
         x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns) #feature scaling
         x_scaled

Out[36]:
```

	Airline Name	Overall_Rating	Verified	Type Of Traveller	Seat Type	Route	Month flown	Year flown	Seat Comfort	Food & Beverages	Ground Service
0	0.000000	0.888889	1.0	1.000000	0.333333	0.630972	0.818182	0.636364	0.800000	0.800000	0.750000
1	0.000000	0.000000	1.0	1.000000	0.333333	0.630898	0.545455	0.636364	0.400000	0.200000	0.000000
2	0.000000	0.000000	1.0	1.000000	0.333333	0.035426	0.545455	0.636364	0.400000	0.200000	0.000000
3	0.008065	0.000000	0.0	1.000000	0.333333	0.312803	1.000000	0.636364	0.200000	0.510717	0.000000
4	0.008065	0.000000	1.0	0.333333	0.333333	0.865648	1.000000	0.636364	0.200000	0.200000	0.000000
...
23166	0.981855	0.000000	0.0	0.333333	0.333333	0.094517	0.545455	0.909091	0.400000	0.510717	0.000000
23167	0.981855	0.000000	1.0	1.000000	0.333333	0.863296	0.545455	0.909091	0.523664	0.510717	0.338435
23168	0.981855	0.222222	1.0	0.000000	0.333333	0.094517	0.727273	0.909091	0.400000	0.400000	0.000000
23169	0.981855	0.555556	1.0	0.000000	0.000000	0.927826	0.727273	0.909091	0.600000	0.600000	0.000000
23170	0.981855	0.000000	1.0	1.000000	0.333333	0.863296	0.727273	0.909091	0.523664	0.510717	0.000000

23171 rows x 11 columns

Activity 3: Splitting data into train and test and validation sets.

Now let's split the Dataset into train, test and validation sets. First split the dataset into train and test sets.

i)x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.2, random_state=None):

The `train_test_split` function from `scikit-learn` to split the scaled feature matrix (`x_scaled`) and the target variable (`y`) into training and testing sets.

```
In [37]: #splitting the training data and testing data
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.2,random_state=None)
```

ii) `smote=SMOTE()`

`x_train_smote,y_train_smote=smote.fit_resample(x_train,y_train):`

"we are dealing with imbalanced data

so to balance that data we use smote technique"

SMOTE technique is a common approach when dealing with imbalanced datasets, where one class is underrepresented compared to the other.

```
In [38]: '''we are dealing with imbalanced data
so to balance that data we use smote technique'''
smote=SMOTE()
x_train_smote,y_train_smote=smote.fit_resample(x_train,y_train)
```

iii) `x_train_smote.shape:`

The `x_train_smote.shape` is used to check the number of rows and columns of the feature matrix `x_train_smote` after applying the SMOTE.

```
In [39]: x_train_smote.shape
Out[39]: (24572, 11)
```

iv) `y_train_smote.shape:`

The `y_train_smote.shape` is used to check the number of rows and columns of the feature matrix `y_train_smote` after applying the SMOTE.

```
In [40]: y_train_smote.shape
Out[40]: (24572,)
```

v) `y_train_smote.value_counts():`

Now the data is balanced.

```
In [41]: y_train_smote.value_counts() #now the data is balanced
Out[41]: 0    12286
         1    12286
         Name: Recommended, dtype: int64
```

Model Building

i) `knn=KNeighborsClassifier()`

`knn.fit(x_train_smote,y_train_smote):`

The k-Nearest Neighbors (KNN) classifier from scikit-learn to train the model on the resampled training data.

```
In [42]: knn=KNeighborsClassifier()
         knn.fit(x_train_smote,y_train_smote)

Out[42]: KNeighborsClassifier()
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.
```

ii) `pred=knn.predict(x_test):`

To make predictions on the test set (`x_test`).

```
In [43]: pred=knn.predict(x_test)
```

iii) `pred:`

```
In [44]: pred
Out[44]: array([1, 1, 0, ..., 1, 1, 0])
```

iv) `y_test:`

```
In [45]: y_test
Out[45]: 12130    1
         5371    1
         7832    0
         15735   0
         17239   1
         ..
         14642   0
         2328    1
         12056    1
         20284    1
         18881    0
         Name: Recommended, Length: 4635, dtype: int32
```

Model Deployment

i) `accuracy_score(pred,y_test):`

To compute the accuracy of the machine learning model's predictions.

```
In [46]: accuracy_score(pred,y_test)
Out[46]: 0.9376483279395901
```

ii) `print(classification_report(pred,y_test)):`

To generate a detailed classification report for your machine learning model's predictions.

```
In [47]: print(classification_report(pred,y_test))
```

	precision	recall	f1-score	support
0	0.94	0.97	0.95	2975
1	0.94	0.88	0.91	1660
accuracy			0.94	4635
macro avg	0.94	0.93	0.93	4635
weighted avg	0.94	0.94	0.94	4635

import pickle

```
In [48]: import pickle
```

pickle.dump(knn,open('ar_knn.pkl','wb'))

```
In [50]: pickle.dump(knn,open('ar_knn.pkl','wb'))
```

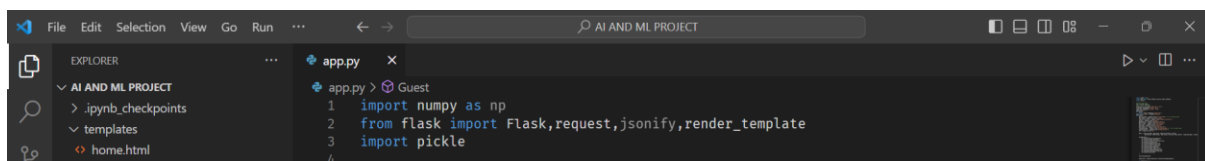
Activity 2: Integrate with Web Framework In this section,

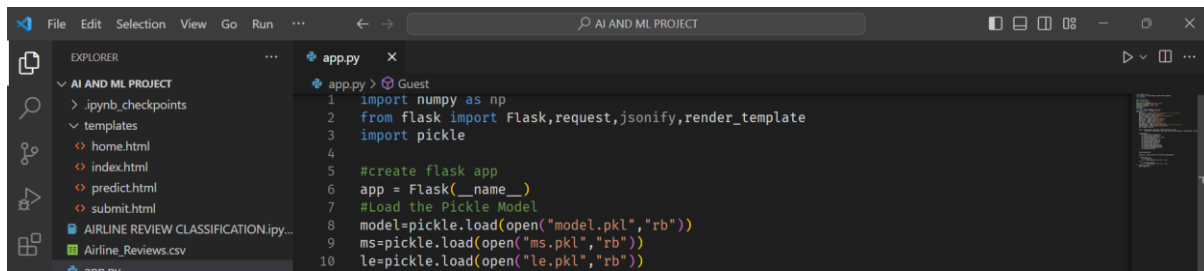
we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI. This section has the following tasks

- Building HTML Pages
- Building server-side script
- Run the web application

Activity 2.2: Build Python code:

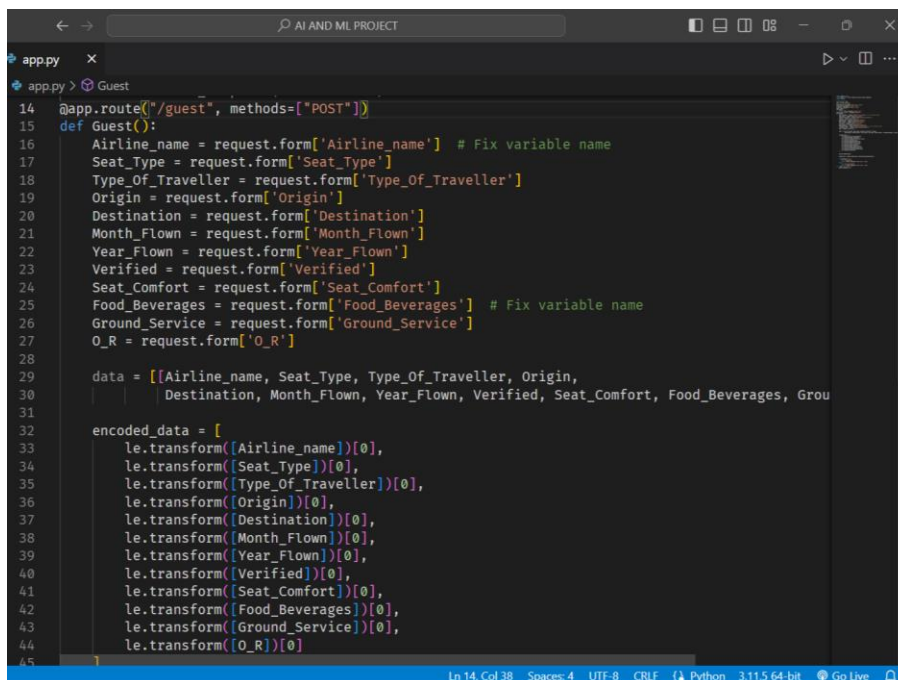
Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (name) as argument





```
File Edit Selection View Go Run ... AI AND ML PROJECT
EXPLORER
  AI AND ML PROJECT
    .ipynb_checkpoints
    templates
      home.html
      index.html
      predict.html
      submit.html
    AIRLINE REVIEW CLASSIFICATION.ipynb
    Airline_Reviews.csv
    app.py
  app.py
    app.py
      1 import numpy as np
      2 from flask import Flask, request, jsonify, render_template
      3 import pickle
      4
      5 #create flask app
      6 app = Flask(__name__)
      7 #Load the Pickle Model
      8 model=pickle.load(open("model.pkl","rb"))
      9 ms=pickle.load(open("ms.pkl","rb"))
      10 le=pickle.load(open("le.pkl","rb"))
```

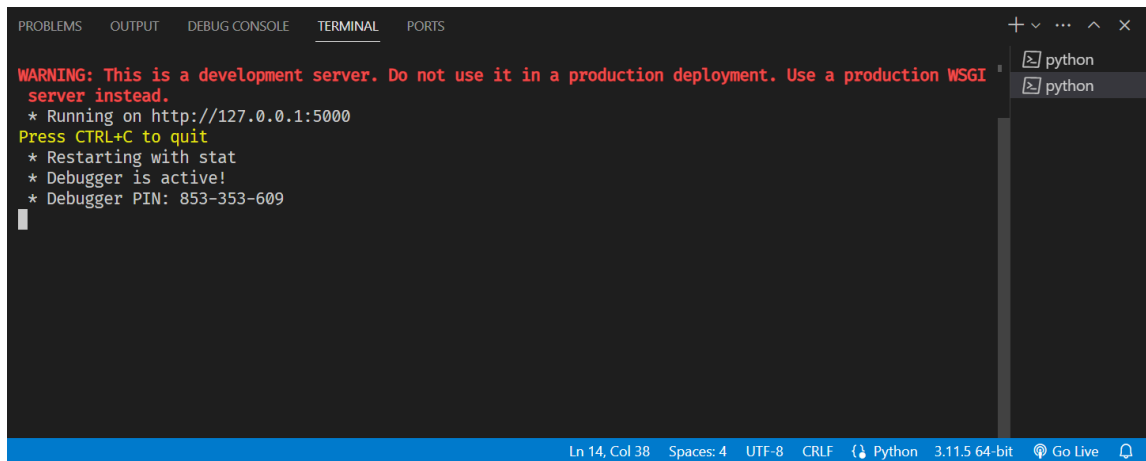
Here we will be using a declared constructor to route to the HTML page which we have created earlier. In the above example, '/' URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method. Retrieves the value from UI:



```
app.py
app.py > Guest
14 @app.route("/guest", methods=["POST"])
15 def Guest():
16     Airline_name = request.form['Airline_name'] # Fix variable name
17     Seat_Type = request.form['Seat_Type']
18     Type_Of_Traveller = request.form['Type_Of_Traveller']
19     Origin = request.form['Origin']
20     Destination = request.form['Destination']
21     Month_Flowm = request.form['Month_Flowm']
22     Year_Flowm = request.form['Year_Flowm']
23     Verified = request.form['Verified']
24     Seat_Comfort = request.form['Seat_Comfort']
25     Food_Beverages = request.form['Food_Beverages'] # Fix variable name
26     Ground_Service = request.form['Ground_Service']
27     O_R = request.form['O_R']
28
29     data = [[Airline_name, Seat_Type, Type_Of_Traveller, Origin,
30             Destination, Month_Flowm, Year_Flowm, Verified, Seat_Comfort, Food_Beverages, Ground_Service, O_R]]
31
32     encoded_data = [
33         le.transform([Airline_name])[0],
34         le.transform([Seat_Type])[0],
35         le.transform([Type_Of_Traveller])[0],
36         le.transform([Origin])[0],
37         le.transform([Destination])[0],
38         le.transform([Month_Flowm])[0],
39         le.transform([Year_Flowm])[0],
40         le.transform([Verified])[0],
41         le.transform([Seat_Comfort])[0],
42         le.transform([Food_Beverages])[0],
43         le.transform([Ground_Service])[0],
44         le.transform([O_R])[0]
45     ]
```

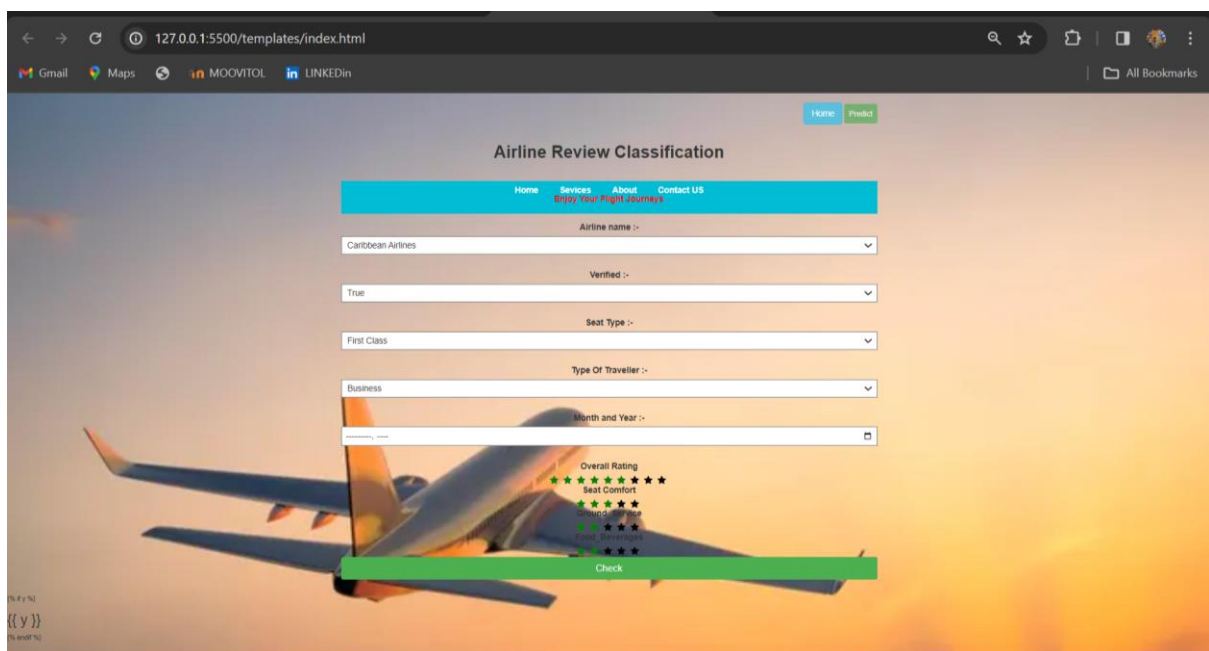
Here we are routing our app to Guest() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the index.html page. Set app.run(debug=True) so that we can edit

Running the Application:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI
server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 853-353-609
Ln 14, Col 38 Spaces: 4 UTF-8 CRLF Python 3.11.5 64-bit Go Live
```

Now, Go the web browser and write the localhost URL (<http://127.0.0.1:5000>) to get the below result



127.0.0.1:5500/templates/index.html

Gmail Maps MOOVITOL LINKEDIN All Bookmarks

Home Product

Airline Review Classification

Home Services About Contact US
Enjoy Your Flight Journeys

Airline name :-
Caribbean Airlines

Verified :-
True

Seat Type :-
First Class

Type Of Traveller :-
Business

Month and Year :-
January 2023

Overall Rating
★★★★★
Seat Comfort
★★★★★
Food & Beverage
★★★★★
Check