

# Project Report (Team Id:592974)

## Airline Review Classification Using Machine Learning

### Table of Contents

1. INTRODUCTION
  1. Project Overview
  2. Purpose
2. LITERATURE SURVEY
  1. Existing problem
  2. References
  3. Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
  1. Empathy Map Canvas
  2. Ideation & Brainstorming
4. REQUIREMENT ANALYSIS
  1. Functional requirement
  2. Non-Functional requirements
5. PROJECT DESIGN
  1. Data Flow Diagrams & User Stories
  2. Solution Architecture
6. PROJECT PLANNING & SCHEDULING
  1. Technical Architecture
  2. Sprint Planning & Estimation
  3. Sprint Delivery Schedule
7. CODING & SOLUTIONING
  1. Feature 1
  2. Feature 2
  3. Database Schema (if Applicable)
8. PERFORMANCE TESTING
  1. Performance Metrics
9. RESULTS
  1. Output Screenshots
10. ADVANTAGES & DISADVANTAGES
11. CONCLUSION
12. FUTURE SCOPE

[Source Code](#)

[GitHub & Project Demo Link](#)

# 1. Introduction:

## Project Overview

In today's interconnected world, the airline industry serves as a critical catalyst for global travel and business. As air travel becomes increasingly accessible, the quality of service provided by airlines plays a pivotal role in shaping passenger experiences. This project focuses on the development of an airline review classification system using Classification models such as Decision Tree Classifier, Random Forest Classifier, XGBoost Classifier, etc.

## Purpose

The proliferation of social media platforms, travel websites, and online forums has given rise to a wealth of user-generated content, including airline reviews. Extracting actionable insights from this vast pool of unstructured text data has the potential to provide airlines with valuable information for refining their services and elevating passenger satisfaction.

Throughout this report, we will delve into the methodology employed to preprocess the raw text data, the process of selecting pertinent features, the training and evaluation of the classification model, and the subsequent interpretation of the obtained results.

## 2. Literature Survey

### Existing Problem

The airline industry faces challenges related to passenger satisfaction and feedback analysis. Existing reviews are often unstructured and require effective classification and sentiment analysis to extract actionable insights.

## References

Dataset used:

[SkyRatings: Unleashing 23K+ Airline Reviews!](#)

## Problem Statement Definition

Goal is to develop a classification model that can effectively analyze airline reviews and categorize them into relevant classes, such as positive, neutral, or negative sentiments. The primary objectives of this project are as follows:

1. **Preprocess Raw Text Data:** Develop a robust preprocessing methodology to clean and prepare the raw text data extracted from various sources, including social media platforms, travel websites, and online forums.
2. **Feature Selection:** Identify and select pertinent features that are crucial for sentiment analysis and classification. This involves extracting relevant information from the unstructured text data.
3. **Classification Model Development:** Utilize classification models such as Decision Tree Classifier, Random Forest Classifier, XGBoost Classifier, and others to build a predictive model that can classify airline reviews into predefined categories.
4. **Sentiment Analysis:** Perform sentiment analysis on the reviews to determine the overall sentiment expressed by passengers, whether it's positive, neutral, or negative.
5. **Actionable Insights:** Interpret the results obtained from the classification model and sentiment analysis to provide airlines with valuable information that can aid in improving their services and enhancing passenger satisfaction.

The ultimate goal of this project is to harness the power of machine learning and natural language processing to help the airline industry address the challenges related to passenger satisfaction and feedback analysis by efficiently categorizing and analyzing the wealth of unstructured text data available in airline reviews.

### **3. Ideation & Proposed Solution**

#### **Empathy Map Canvas**

##### **- WHO are we empathizing with?**

We are empathizing with airline passengers who might be frequent flyers, business travelers, vacationers, or any individual using airline services. The passenger is experiencing end-to-end airline service, which encompasses booking ticket, arriving at the airport, security check in, boarding etc. They may encounter scenarios like flight delays, in-flight services, interaction with flight staff and other passengers.

##### **- What do they HEAR?**

- Complaints about delays or cancellations.
- Appreciation for onboard services or in-flight entertainment.
- Worries about luggage handling.
- Conversations about legroom or seating comfort.

##### **- What do they DO?**

- Using in-flight entertainment system.
- Buying snacks and drinks on-board.
- Boarding, de-boarding and collecting the luggage.

##### **- What do they need to DO?**

- Post a review of their experience with the flight.
- Collect the luggage securely.
- Respect other passengers boundaries.

##### **- What do they SEE?**

- Crowded boarding areas.
- Long lines at check-in or security.

- Digital screens showing flight information.
- Crew interactions with passengers.

**- What do they SAY?**

- Complementing or complaining to cabin crew.
- Asking for in-flight services.
- Discussing future travelling plans based on current experience.

**- PAINS**

- Unpredictable flight delays
- Rescheduling of flights
- Overpriced on-board food
- Miscommunication or lack
- of information
- Uncomfortable seats for long hauls-flight.

**- GAINS**

- A quick way of traveling.
- Courteous staff and good on-board services.
- A smooth and punctual journey.
- Clear and regular communication .
- Responsive airlines with good customer services.

## Ideation & Brainstorming

- Data Collection Strategy
- Model Selection and Hyperparameter Tuning
- Privacy and Ethical Consideration
- Data Preprocessing and Cleaning
- Scalability Planning
- Interpretability Techniques
- Feature Extraction and Sentiment Analysis
- User Feedback Integration
- Evaluation Metrics and Model Performance Analysis
- Proper Documentation
- Web App Development
- Privacy and Ethical Considerations

## 4.Requirement Analysis:

### Functional Requirements:

- User Authentication and Authorization:

Description: Users should have unique accounts with appropriate access levels.

Features:

User registration.

User login/logout.

User roles (admin, regular user).

- Input Data Submission:

Description: Users should be able to submit their reviews.

Features:

Form to input various parameters (airline name, rating, type of traveler, etc.).

Option to upload files (e.g., JSON for bulk reviews).

- Review Classification:

Description: The system should classify reviews into categories (Recommended/Not Recommended).

Features:

Integration with machine learning models for classification.

Real-time or batch processing of reviews.

- **Data Visualization:**

Description: Provide visual representations of review data.

Features:

Graphs and charts displaying overall ratings.

Distribution of recommendations.

- **Search and Filtering:**

Description: Users should be able to search and filter reviews.

Features:

Search by airline name, origin, destination, etc.

Filters based on time, type of traveler, etc.

- **Admin Dashboard:**

Description: An admin interface to manage users and system settings.

Features:

User management (add, remove, modify).

Model retraining options.

System configuration settings.

- **Feedback Mechanism:**

Description: Users should be able to provide feedback on the classification.

Features:

Option to dispute the classification.

Feedback form for users to suggest improvements.

## **Non-Functional Requirements:**

- **Performance:**

Description: The system should handle a large number of reviews efficiently.

Requirements:

Response time for review classification should be within a few seconds.

- **Scalability:**

Description: The system should easily scale to accommodate a growing number of users and reviews.

Requirements:

Ability to handle increased traffic without performance degradation.

- **Reliability:**

Description: The system should be reliable and available whenever users need to access it.



Requirements:

High availability (e.g., 99.9% uptime).

- Security:

Description: The system should ensure the confidentiality and integrity of user data.

Requirements:

Secure user authentication and authorization.

Data encryption in transit and at rest.

- Usability:

Description: The system should be user-friendly.

Requirements:

Intuitive user interfaces.

Clear instructions for data submission and retrieval.

- Maintainability:

Description: The system should be easy to maintain and update.

Requirements:

Well-documented code and system architecture.

Version control for software changes

.

- Model Accuracy:

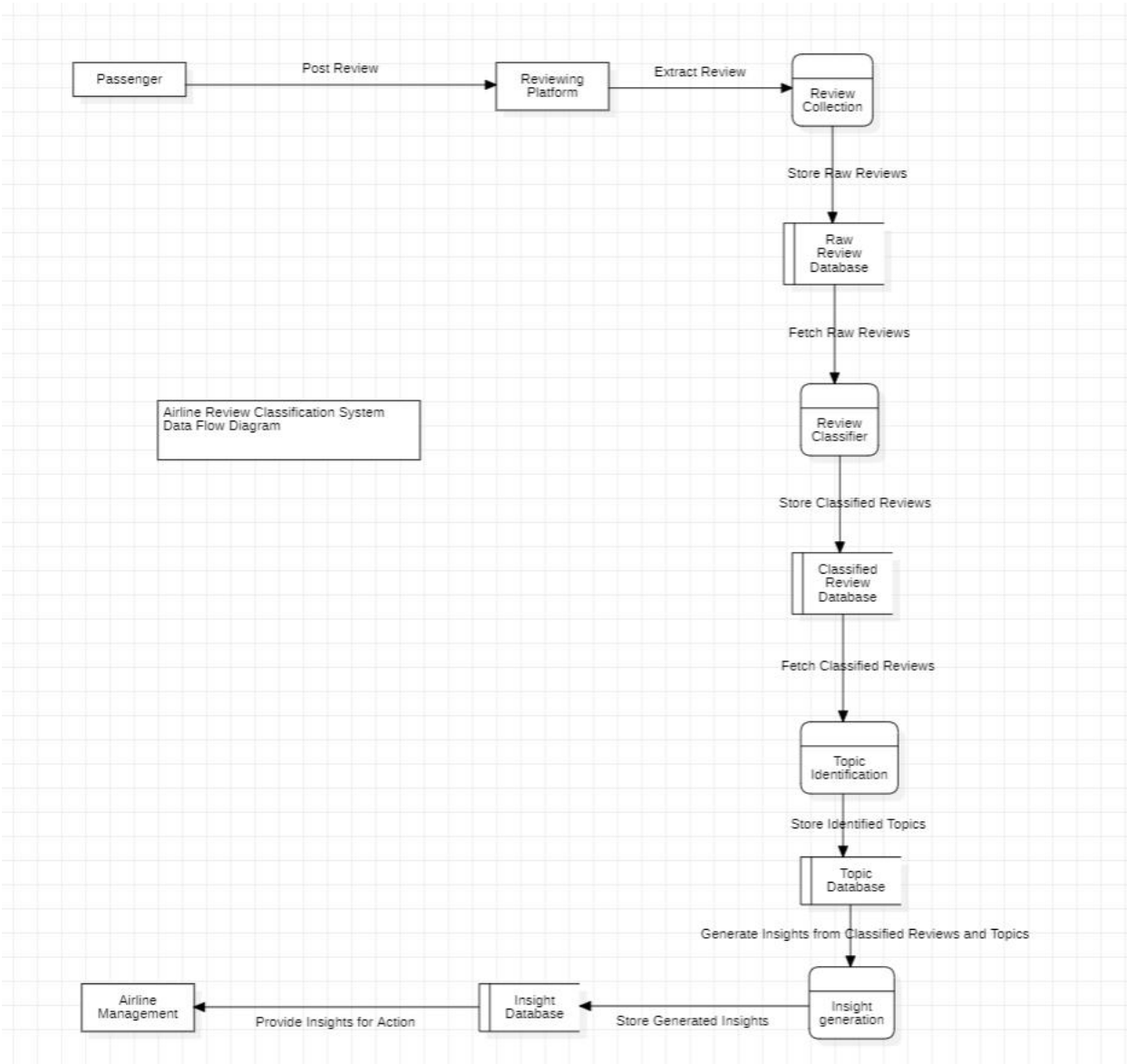
Description: The machine learning model should provide accurate classifications.

Requirements:

Regular model retraining based on new data.

## 5. Project Design

### Data Flow Diagrams & User Stories:



### Solution Architecture:

#### Components:

- User Interface (UI):

Description: The front-end interface accessible to users for submitting and viewing reviews.

Technology: HTML, CSS, JavaScript, and a front-end framework like React or Angular.

- Application Logic:

Description: Handles user requests, manages data, and interacts with the machine learning model.

Technology: Flask (Python) for the backend logic.

- Machine Learning Model:

Description: Classifies reviews into recommended or not recommended categories.

Technology: Python with scikit-learn or TensorFlow for model training and deployment.

- Database:

Description: Stores user data, reviews, and system configuration settings.

Technology: Relational database (e.g., PostgreSQL) for structured data.

- User Authentication and Authorization:

Description: Manages user accounts, authentication, and authorization.

Technology: OAuth for third-party authentication, JWT tokens for session management.

- Admin Dashboard:

Description: Interface for administrators to manage users and system settings.

Technology: A web-based dashboard using the same front-end technology as the user interface.

- Feedback Mechanism:

Description: Allows users to provide feedback on review classifications.

Technology: Integrated form within the UI, storing feedback data in the database.

- Data Processing:

Description: Handles preprocessing of input data before it goes to the machine learning model.

Technology: Python scripts for data transformation and validation.

- External Interfaces:

Description: Interfaces with third-party services or APIs for additional data or functionalities.

Technology: RESTful APIs for communication.

- Data Storage:

Description: Stores both structured and unstructured data.

Technology: Cloud-based storage (e.g., Amazon S3) for file uploads, and a relational database for structured data.

- Scalability and Load Balancing:

Description: Ensures the system can handle increased loads.

Technology: Containerization (e.g., Docker), orchestration (e.g., Kubernetes), and load balancing.

- Security Layer:

Description: Protects the system against security threats and vulnerabilities.

Technology: SSL for secure communication, firewall, regular security audits.

### **Workflow:**

- User submits a review:

The UI captures user input.

Data is sent to the Application Logic.

- Data Processing:

Application Logic processes the input data.

Validates and transforms the data.

- Machine Learning Classification:

Preprocessed data is sent to the Machine Learning Model.

Model classifies the review.

- Storage and Logging:

Classified data is stored in the database.

System logs record the transaction.

- **User Interaction:**

Users can view their own reviews and reviews from others.

Admins use the dashboard to manage users and system settings.

- **Feedback Mechanism:**

Users can provide feedback on classification results.

Feedback data is stored for analysis.

**Considerations:**

- **Cloud Deployment:**

Deploy components on a cloud platform for scalability and accessibility.

- **Continuous Integration/Continuous Deployment (CI/CD):**

Implement CI/CD pipelines for automated testing and deployment.

- **Monitoring and Logging:**

Use monitoring tools to track system performance.

Implement logging for auditing and troubleshooting.

- **Data Privacy and Compliance:**

Ensure compliance with data protection regulations (e.g., GDPR).

- **Machine Learning Model Maintenance:**

Schedule regular model retraining based on new data.

## 6. Project Planning & Scheduling:

### Technical Architecture:

- Data Collection Layer:

Data Sources: Connect to various data sources such as airline websites, social media, and third-party review platforms (e.g., TripAdvisor).

Data Ingestion: Use data ingestion tools or connectors to collect reviews and store them in a raw data storage layer.

- Data Preprocessing Layer:

Data Cleaning: Clean and preprocess the raw data to remove noise, special characters, and convert text to lowercase.

Tokenization: Tokenize text data into words or phrases.

Stop Words Removal: Eliminate common stop words that do not carry much meaning.

Feature Engineering: Convert text data into numerical representations (e.g., TF-IDF, Word Embeddings).

- Machine Learning Layer:

Classification Model: Train a machine learning model for sentiment analysis, which can be based on algorithms like Logistic Regression, Naive Bayes, or deep learning techniques like LSTM.

Topic Modeling: Employ topic modeling techniques (e.g., Latent Dirichlet Allocation) to identify common themes in reviews.

Real-time Learning: Implement a mechanism for the model to learn from user feedback and continuously improve its accuracy.

- Data Storage Layer:

Raw Review Database: Store the original reviews for reference.

Classified Review Database: Store reviews along with their sentiment classification (positive, negative, neutral).

Topic Database: Maintain information about identified topics or themes in the reviews.

Model Parameters Storage: Store model parameters for deployment and model version control.

- API and Services Layer:

RESTful API: Develop APIs to serve predictions and insights to end-users.

Web Interface: Create a user-friendly web interface for airlines and passengers to interact with the system.

Feedback Mechanism: Include a feature for passengers to provide feedback or corrections on sentiment and topic classifications.

Real-time Dashboard: Provide dashboards for airlines to view real-time insights, sentiment trends, and common issues.

- Cloud Services:

Cloud Infrastructure: Deploy the system on a cloud platform like AWS, Azure, or Google Cloud for scalability and redundancy.

Containerization: Use containerization tools like Docker for easy deployment and management.

Auto-Scaling: Configure auto-scaling to handle varying workloads.

- Security and Compliance:

Implement security measures to protect sensitive passenger data.

Ensure compliance with data protection regulations (e.g., GDPR).

- Monitoring and Logging:

Implement logging to track system behavior and user interactions.

Use monitoring tools to detect issues and performance bottlenecks.

- Deployment and Scaling Strategy:

Deploy the system on multiple servers or containers for load balancing and redundancy.

Implement horizontal scaling to handle increasing data volumes and user loads.

- Maintenance and Continuous Improvement:

Regularly update the machine learning model with new data.

Monitor model performance and retrain it as needed.

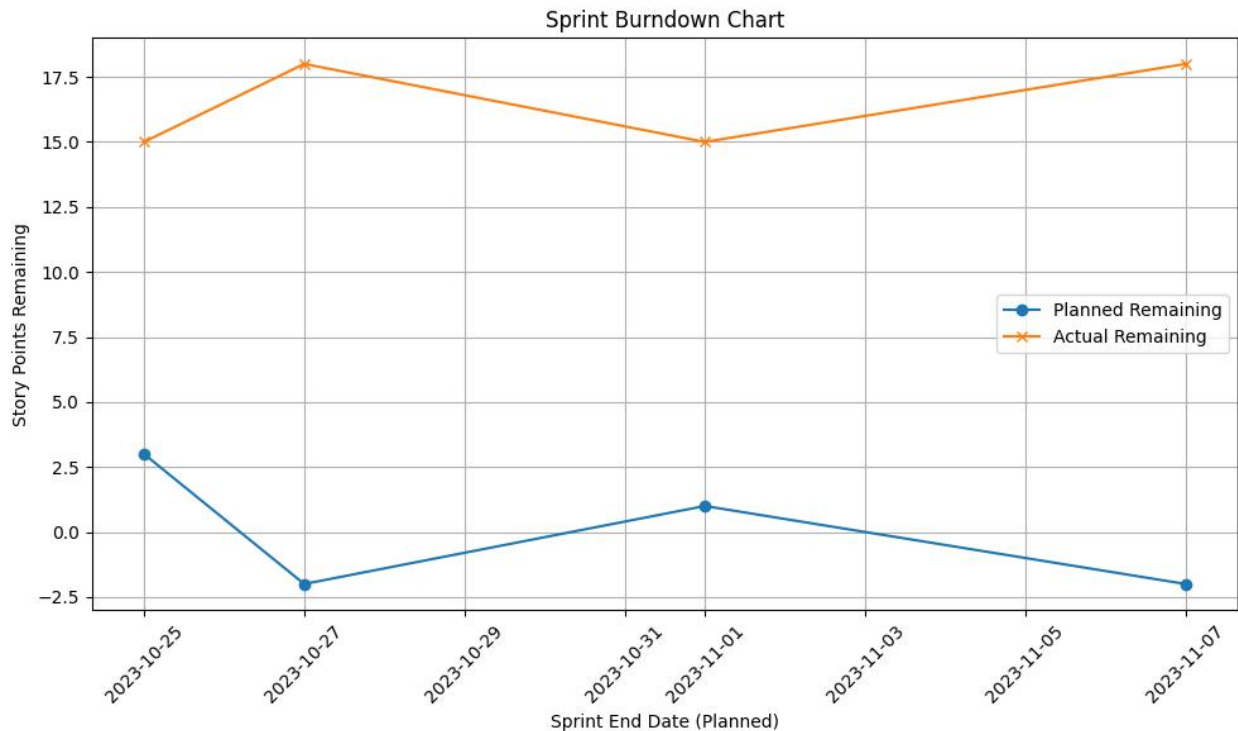
Stay informed about the latest advancements in NLP and machine learning to improve the system over time.

## Sprint Planning & Estimation

Sprint	Functional Requirement	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High
Sprint-2		USN-2	As a user, I will receive a confirmation email once I have registered for the application.	1	High
Sprint-3		USN-4	As a user, I can register for the application through Gmail.	2	Medium
Sprint-4	Login	USN-5	As a user, I can log into the application by entering email & password.	1	High
Sprint-5		USN-3	As a user, I can register for the application through Facebook.	2	Low

## Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed	Sprint Release Date (Actual)
Sprint-1	15	5 Days	20-Oct-2023	25-Oct-2023	12	25-Oct-2023
Sprint-2	18	2 Days	25-Oct-2023	27-Oct-2023	20	28-Oct-2023
Sprint-3	15	5 Days	27-Oct-2023	01-Nov-2023	14	03-Nov-2023
Sprint-4	18	6 Days	01-Nov-2023	07-Nov-2023	20	10-Nov-2023





## 7. Coding & Solutioning

- Importing Necessary Libraries:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, roc_curve, auc
import pickle
from scipy import stats
import warnings
warnings.filterwarnings('ignore')
print('All libraries imported. \nSetup complete.')
```

- Filling Null Values:

```
ar1['Type Of Traveller']=ar1['Type Of Traveller'].fillna(ar1['Type Of Traveller'].mode()[0])
ar1['Type Of Traveller'].isnull().sum()

ar1['Seat Type']=ar1['Seat Type'].fillna(ar1['Seat Type'].mode()[0])
ar1['Seat Type'].isnull().sum()
ar1['Route']=ar1['Route'].fillna(ar1['Route'].mode()[0])
ar1['Route'].isnull().sum()
ar1['Date Flown']=ar1['Date Flown'].fillna(ar1['Date Flown'].mode()[0])
ar1['Date Flown'].isnull().sum()
ar1['Seat Comfort']=ar1['Seat Comfort'].fillna(ar1['Seat Comfort'].mode()[0])
ar1['Seat Comfort'].isnull().sum()
ar1['Cabin Staff Service']=ar1['Cabin Staff Service'].fillna(ar1['Cabin Staff Service'].mode()[0])
ar1['Cabin Staff Service'].isnull().sum()
ar1['Food & Beverages']=ar1['Food & Beverages'].fillna(ar1['Food & Beverages'].mode()[0])
ar1['Food & Beverages'].isnull().sum()
ar1['Ground Service']=ar1['Ground Service'].fillna(ar1['Ground Service'].mode()[0])
ar1['Ground Service'].isnull().sum()
```

- Encoding:

```
from sklearn.preprocessing import LabelEncoder

le1=LabelEncoder()
le2=LabelEncoder()
le3=LabelEncoder()
le4=LabelEncoder()
le5=LabelEncoder()
le6=LabelEncoder()
le7=LabelEncoder()
le8=LabelEncoder()
le9=LabelEncoder()

ar1['Airline Name']=le1.fit_transform(ar1['Airline Name'])
ar1['Verified']=le2.fit_transform(ar1['Verified'])
ar1['Type Of Traveller']=le3.fit_transform(ar1['Type Of Traveller'])
ar1['Seat Type']=le4.fit_transform(ar1['Seat Type'])
ar1['Origin']=le5.fit_transform(ar1['Origin'])
ar1['Destination']=le6.fit_transform(ar1['Destination'])
ar1['Month Flown']=le7.fit_transform(ar1['Month Flown'])
ar1['Year Flown']=le8.fit_transform(ar1['Year Flown'])
ar1['Recommended']=le9.fit_transform(ar1['Recommended'])

ar1.head()
```

- Analysis:

```
ar1.describe()

fig=sns.lineplot(x=ar1.index,y=ar1['Type Of Traveller'],markevery=1,marker='d',hue=ar1['Seat Type'])
sns.set(rc={'figure.figsize':[10,10]})
fig.set(xlabel='index')

ar1['Seat Type'].value_counts().plot.bar()

plt.figure(figsize=(5,5))
plt.pie(ar1['Seat Type'].value_counts(),startangle=90,autopct='%0.3f',shadow=True,labels=['Economy','Buisness','Premium','First'])

sns.barplot(data=ar1,x='Type Of Traveller',y='Seat Type')

sns.heatmap(ar1.corr(),annot=True)
```

- Splitting the dataset:

```
x=ar1.loc[:,'Airline Name':'Ground Service']
y=ar1['Recommended']

smote=SMOTE(sampling_strategy='auto',random_state=50)
x,y=smote.fit_resample(x,y)

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

- Scaling and Model Building:

```
ss=StandardScaler()

x_train=ss.fit_transform(x_train)
x_test=ss.fit_transform(x_test)
import pickle
pickle.dump(ss,open('ss1.pkl','wb'))

model1=DecisionTreeClassifier(criterion='entropy',random_state=0)
model1.fit(x_train,y_train)
pred1=model1.predict(x_test)
pred1

from sklearn.metrics import classification_report,confusion_matrix,accuracy_score,roc_auc_score
print(classification_report(y_test,pred1))
print(accuracy_score(y_test,pred1))
print(confusion_matrix(y_test,pred1))
print(roc_auc_score(y_test,pred1))
```

```
model2=RandomForestClassifier(criterion='entropy',n_estimators=20,random_state=0)
model2.fit(x_train,y_train)
pred2=model2.predict(x_test)
pred2
print(classification_report(y_test,pred2))
print(accuracy_score(y_test,pred2))
print(confusion_matrix(y_test,pred2))
print(roc_auc_score(y_test,pred2))
```

```
model3= LogisticRegression()
model3.fit(x_train,y_train)
pred3=model3.predict(x_test)
print(classification_report(y_test,pred3))
print(accuracy_score(y_test,pred3))
print(confusion_matrix(y_test,pred3))
print(roc_auc_score(y_test,pred3))
```

```
model4=GaussianNB()
model4.fit(x_train,y_train)
pred4=model4.predict(x_test)
print(classification_report(y_test,pred4))
print(accuracy_score(y_test,pred4))
print(confusion_matrix(y_test,pred4))
print(roc_auc_score(y_test,pred4))
```

```
model5=KNeighborsClassifier(n_neighbors=10)
model5.fit(x_train,y_train)
pred5=model5.predict(x_test)
print(classification_report(y_test,pred5))
print(accuracy_score(y_test,pred5))
print(confusion_matrix(y_test,pred5))
print(roc_auc_score(y_test,pred5))
print(confusion_matrix(y_test,pred5))
```

```
model6= SVC()
model6.fit(x_train,y_train)
pred6=model6.predict(x_test)
print(classification_report(y_test,pred6))
print(accuracy_score(y_test,pred6))
print(confusion_matrix(y_test,pred6))
print(roc_auc_score(y_test,pred6))
```

```
model7=XGBClassifier(n_estimators=6,max_depth=8)
model7.fit(x_train,y_train)
pred7=model7.predict(x_test)
print(classification_report(y_test,pred7))
print(accuracy_score(y_test,pred7))
print(confusion_matrix(y_test,pred7))
print(roc_auc_score(y_test,pred7))
```

```
per=pd.DataFrame({'Model':['DecisionTreeClassifier','Random Forest
Classifier','LogisticRegression','NaiveBayesClassifier','KNN Classifier','SVC','XGB'],
                  'roc_auc':[roc_auc_score(y_test,pred1),roc_auc_score(y_test,pred2),roc_auc_score(y_test,
pred3),roc_auc_score(y_test,pred4),roc_auc_score(y_test,pred5),roc_auc_score(y_test,pred6),roc_auc_score(y
_test,pred7)],
                  'acc':[accuracy_score(y_test,pred1),accuracy_score(y_test,pred2),accuracy_score(y_test,p
red3),accuracy_score(y_test,pred4),accuracy_score(y_test,pred5),accuracy_score(y_test,pred6),accuracy_scor
e(y_test,pred7)]})
per
```

```
##KNN got the highest accuracy and roc_auc score
pickle.dump(model5,open('knn_model5.pkl','wb'))
```

## 8. Performance Testing:

Classification Model	Accuracy Score	Precision	Recall	F1-Score	True Positives	True Negatives	False Positives	False Negatives
KNeighborsClassifier	0.943	0.94	0.95	0.94	2861	2937	192	156
SVC	0.929	0.92	0.94	0.93	2813	2899	240	194
XGBClassifier	0.912	0.89	0.94	0.92	2701	2907	352	186
RandomForestClassifier	0.895	0.84	0.97	0.9	2491	3010	562	83
LogisticRegression	0.878	0.87	0.89	0.88	2639	2759	414	334
GaussianNB	0.855	0.87	0.84	0.85	2664	2589	389	504
DecisionTreeClassifier	0.847	0.79	0.95	0.86	2253	2759	800	142

**KNN performed best.**

Detailed performance of KNN:

Classification

```
              precision    recall  f1-score   support

0               0.95         0.94         0.94       3053
report: 1               0.94         0.95         0.94       3093
```

Accuracy score:

```
accuracy              0.94       6146
macro avg             0.94         0.94         0.94       6146
weighted avg          0.94         0.94         0.94       6146
```

Confusion matrix:

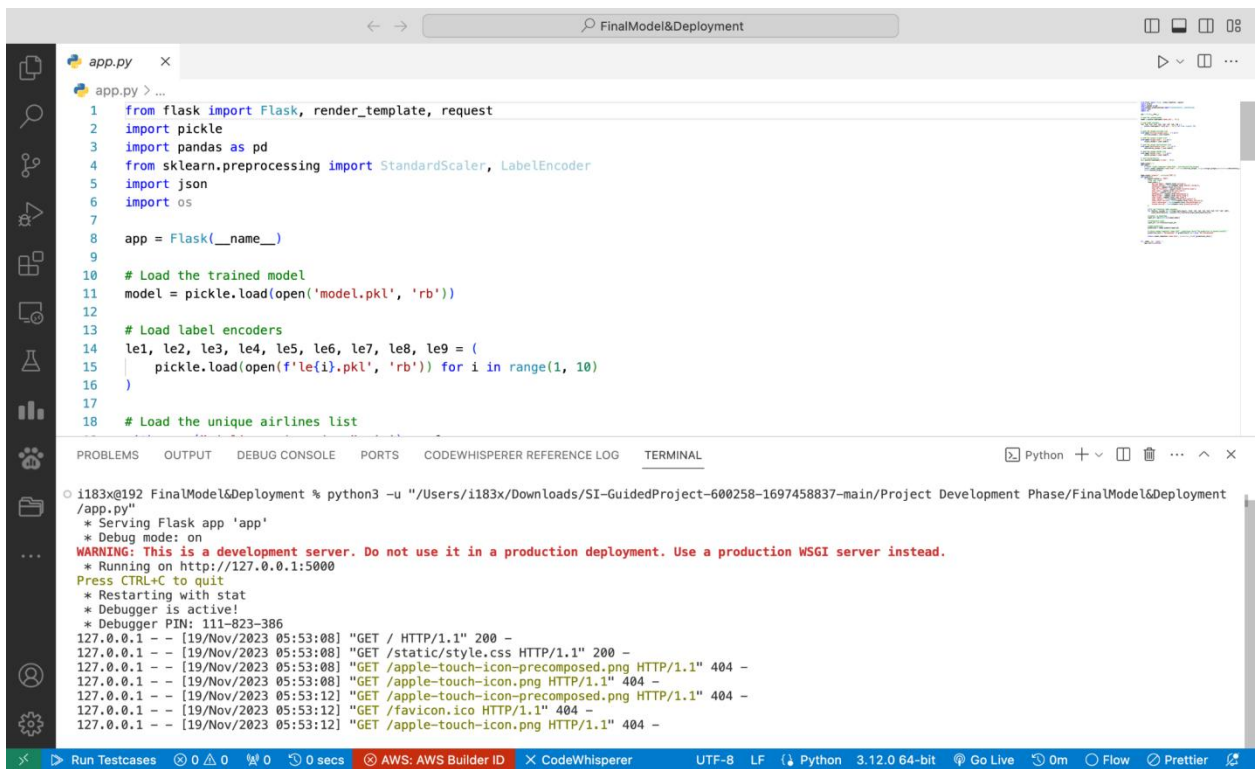
```
0.943377806703547
[[2861  192]
 [ 156 2937]]
```

ROC AUC (Receiver Operating Characteristic - Area Under the Curve) score:

**0.9433372844379111**

## 9. Results

Running app.py:



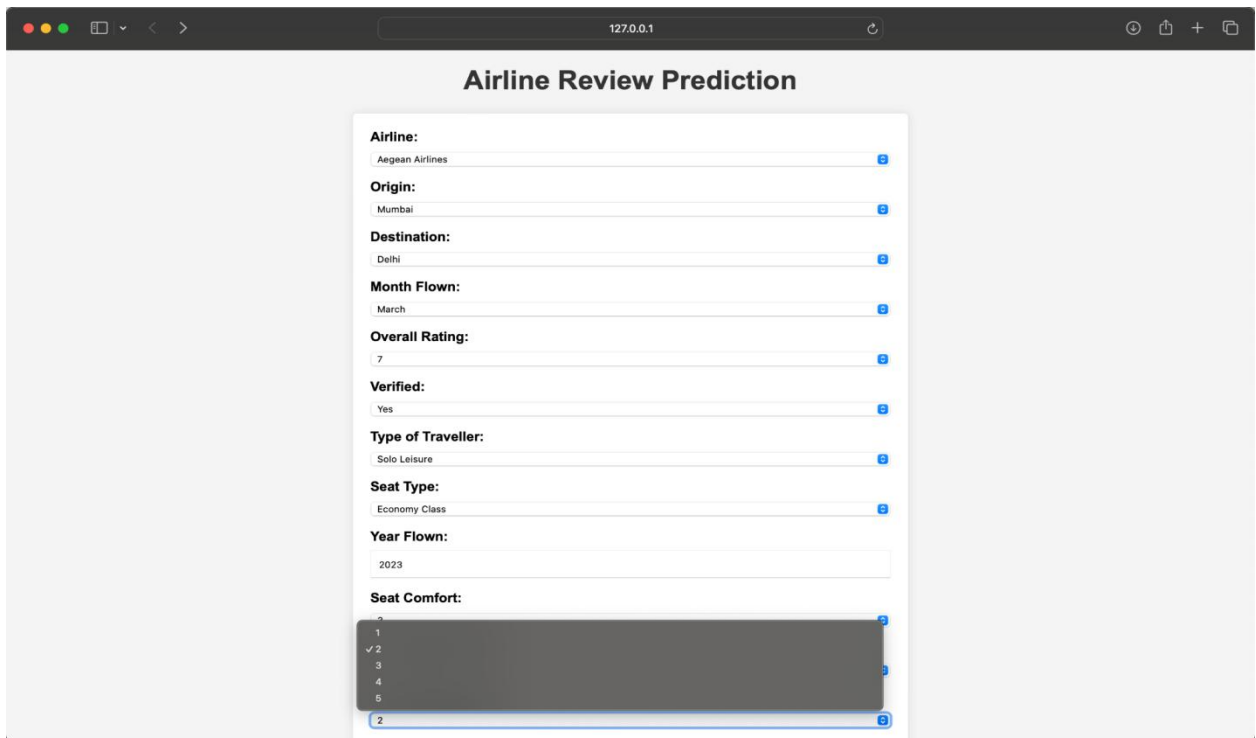
The screenshot shows a VS Code editor with a file named `app.py` open. The code is a Flask application that loads a trained model and encoders, and serves a web interface. The terminal output shows the server running on `http://127.0.0.1:5000` and handling several GET requests, including `/static/style.css` and `/apple-touch-icon.png`.

```
1 from flask import Flask, render_template, request
2 import pickle
3 import pandas as pd
4 from sklearn.preprocessing import StandardScaler, LabelEncoder
5 import json
6 import os
7
8 app = Flask(__name__)
9
10 # Load the trained model
11 model = pickle.load(open('model.pkl', 'rb'))
12
13 # Load label encoders
14 le1, le2, le3, le4, le5, le6, le7, le8, le9 = (
15     pickle.load(open(f'le{i}.pkl', 'rb')) for i in range(1, 10)
16 )
17
18 # Load the unique airlines list
```

Terminal Output:

```
python3 -u "/Users/i183x/Downloads/SI-GuidedProject-600258-1697458837-main/Project Development Phase/FinalModel&Deployment /app.py"
* Serving Flask app 'app'
* Debug mode: on
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: 111-823-386
127.0.0.1 -- [19/Nov/2023 05:53:08] "GET / HTTP/1.1" 200 -
127.0.0.1 -- [19/Nov/2023 05:53:08] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 -- [19/Nov/2023 05:53:08] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 -
127.0.0.1 -- [19/Nov/2023 05:53:08] "GET /apple-touch-icon.png HTTP/1.1" 404 -
127.0.0.1 -- [19/Nov/2023 05:53:12] "GET /apple-touch-icon-precomposed.png HTTP/1.1" 404 -
127.0.0.1 -- [19/Nov/2023 05:53:12] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 -- [19/Nov/2023 05:53:12] "GET /apple-touch-icon.png HTTP/1.1" 404 -
```

Filling the information:



The screenshot shows a web application titled "Airline Review Prediction". It contains a form with the following fields:

- Airline: Aegean Airlines
- Origin: Mumbai
- Destination: Delhi
- Month Flown: March
- Overall Rating: 7
- Verified: Yes
- Type of Traveller: Solo Leisure
- Seat Type: Economy Class
- Year Flown: 2023
- Seat Comfort: 2

Result:

The screenshot shows a web application interface with a form titled "Recommended". The form contains the following fields:

- Destination: (empty)
- Month Flown: (empty)
- Overall Rating: (0)
- Verified: (Yes)
- Type of Traveller: (Solo Leisure)
- Seat Type: (Economy Class)
- Year Flown: (empty)
- Seat Comfort: (1)
- Cabin Staff Service: (1)
- Food & Beverages: (1)
- Ground Service: (1)

A green "Predict" button is located at the bottom of the form. Below the form, the word "Recommended" is displayed in a large, bold font. A "Display a menu" button is visible in the bottom left corner.

'Recommended' according to previous input.

Filling the information:

The screenshot shows the same web application interface, but with the form fields filled with information:

- Airline: (Air Algerie)
- Origin: (Dallas)
- Destination: (Delhi)
- Month Flown: (March)
- Overall Rating: (5)
- Verified: (Yes)
- Type of Traveller: (Solo Leisure)
- Seat Type: (Economy Class)
- Year Flown: (2023)
- Seat Comfort: (2)
- Cabin Staff Service: (3)
- Food & Beverages: (2)
- Ground Service: (4)

A green bar is visible at the bottom of the form. A "Display a menu" button is visible in the bottom left corner.

Result:

The screenshot shows a web application interface with a dark grey header bar. The header contains a browser address bar with '127.0.0.1' and several icons on the right. Below the header is a light grey background. In the center, there is a white form with various input fields and a green 'Predict' button. The form fields are labeled as follows: 'Destination:', 'Month Flown:', 'Overall Rating:' (with a value of 0), 'Verified:' (with a value of Yes), 'Type of Traveller:' (with a value of Solo Leisure), 'Seat Type:' (with a value of Economy Class), 'Year Flown:', 'Seat Comfort:' (with a value of 1), 'Cabin Staff Service:' (with a value of 1), 'Food & Beverages:' (with a value of 1), and 'Ground Service:' (with a value of 1). Below the form, the text 'Not Recommended' is displayed in a bold, black font. In the bottom left corner, there is a small button labeled 'Display a menu'.

Destination:

Month Flown:

Overall Rating:

0

Verified:

Yes

Type of Traveller:

Solo Leisure

Seat Type:

Economy Class

Year Flown:

Seat Comfort:

1

Cabin Staff Service:

1

Food & Beverages:

1

Ground Service:

1

Predict

Not Recommended

Display a menu

**'Not Recommended'** according to previous input.



## 10. Advantages & Disadvantages

### **Advantages:**

- Improved Customer Experience: Users can make informed decisions about airlines based on reviews.
- Efficient Review Processing: Automation of review classification streamlines the review analysis process.
- Data-Driven Decision Making: Insights from visualizations help airlines make data-driven improvements.
- User Engagement: Feedback mechanisms encourage user engagement and continuous improvement.
- Scalability: Cloud deployment and containerization support scalability.
- Enhanced Security: Security measures, including SSL and user authentication, protect user data.
- Admin Control: The admin dashboard provides control over user management and system settings.
- Flexibility: External interfaces allow integration with third-party services or APIs.
- Continuous Improvement: Regular model retraining supports ongoing improvement in classification accuracy.

### **Disadvantages:**

- Initial Model Training Complexity: Developing and training an accurate machine learning model can be complex and time-consuming.
- Dependency on Data Quality: The effectiveness of the system relies on the quality and diversity of the training data.
- User Adoption Challenges: Users may be hesitant to adopt a new system for submitting reviews.
- Feedback Processing Overhead: Managing and responding to user feedback requires additional resources.
- Maintenance Effort: Regular maintenance is necessary, including model retraining and system updates.
- Data Privacy Concerns: Storing user reviews requires robust privacy measures to comply with regulations.
- System Complexity: The integration of various components may increase system complexity.

- Resource Intensive:Machine learning model deployment and maintenance demand computational resources.
- Learning Curve:Users and administrators may require time to adapt to the new system.
- Potential Bias in Models:Machine learning models may exhibit biases based on training data, impacting review classifications.
- Costs:Cloud services and maintenance may incur ongoing operational costs.
- Potential for User Disputes:Users may dispute review classifications, necessitating a resolution process.

## 11. Conclusion

- **Review Classification Accuracy:**

Assess the accuracy of the machine learning model in classifying airline reviews into "Recommended" or "Not Recommended." Evaluate its performance on both training and testing datasets.

- **User Engagement and Feedback:**

Analyze user engagement with the system, including the frequency of review submissions and feedback provided. Examine the nature of user feedback to identify areas for improvement.

- **System Performance:**

Evaluate the overall performance of the system, including response times for review classification, data processing efficiency, and any potential bottlenecks.

- **Security and Privacy Compliance:**

Verify the system's adherence to security and privacy standards, especially concerning user data protection. Ensure compliance with regulations like GDPR or other applicable data protection laws.

- **Maintenance and Model Retraining:**

Assess the effectiveness of the model maintenance strategy, including the frequency of model retraining and its impact on classification accuracy over time.

## 12. Future Scope:

- **Enhanced User Experience:**  
Improve the user interface to provide a more intuitive and visually appealing experience for users, encouraging greater engagement.
- **Natural Language Processing (NLP):**  
Integrate advanced NLP techniques to extract more nuanced insights from user reviews, considering sentiment analysis and context for more accurate classifications.
- **Dynamic Recommendation System:**  
Develop a recommendation system that dynamically adapts to changing trends and user preferences, providing personalized suggestions for users.
- **Multimodal Analysis:**  
Explore the incorporation of image and multimedia analysis for reviews that include visual content, enhancing the system's ability to understand diverse forms of feedback.
- **Real-time Review Processing:**  
Implement real-time review processing to reduce latency and provide users with immediate feedback on their submissions.
- **Social Media Integration:**  
Integrate with social media platforms to capture and analyze additional user reviews and sentiments, expanding the scope of the system.

Source Code(Click to view)

[Python Collaborative Notebook](#)

[KNN Model](#)

GitHub & Project Demo Link

[GitHub repository](#)

Demo link(<https://youtu.be/qZlBr0n7V2Y>)