1.INTRODUCTION
1.1 Project Overview
1.2 Purpose
2. LITERATURE SURVEY
2.1 Existing problem
2.2 References
2.3 Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
3.1 Empathy Map Canvas
3.2 Ideation & Brainstorming
3.3 Proposed Solution
3.4 Problem Solution fit
4. REQUIREMENT ANALYSIS
4.1 Functional requirement
4.2 Non-Functional requirements
5. PROJECT DESIGN
5.1 Data Flow Diagrams
5.2 Solution & Technical Architecture
5.3 User Stories
6. PROJECT PLANNING & SCHEDULING
6.1 Sprint Planning & Estimation
6.2 Sprint Delivery Schedule
6.3 Reports from JIRA
7. CODING & SOLUTIONING (Explain the features added in the project along with code)
7.1 Feature 1
7.2 Feature 2
7.3 Database Schema (if Applicable)
8. TESTING
8.1 Test Cases
8.2 User Acceptance Testing
9. RESULTS
9.1 Performance Metrics
10. ADVANTAGES & DISADVANTAGES
11. CONCLUSION
12. FUTURE SCOPE
13. APPENDIX
13.1 Source Code
13.2 GitHub & Project Demo Link

INTRODUCTION

CHAPTER 1
INTRODUCTION

Travellers have begun to favour air travel more and more over the past 20 years,primarily due to its quickness and occasional comfort. Both on the ground and in the air,as a result, have experienced amazing growth. Massive amounts of ground and airborne aircraft delays have also been brought on by an increase in air traffic. Large economic and environmental losses are the result of these delays. The model's primary goal is to correctly forecast flight delays in order to improve aircraft operations and reduce delays

1.1. PROJECT OVERVIEW

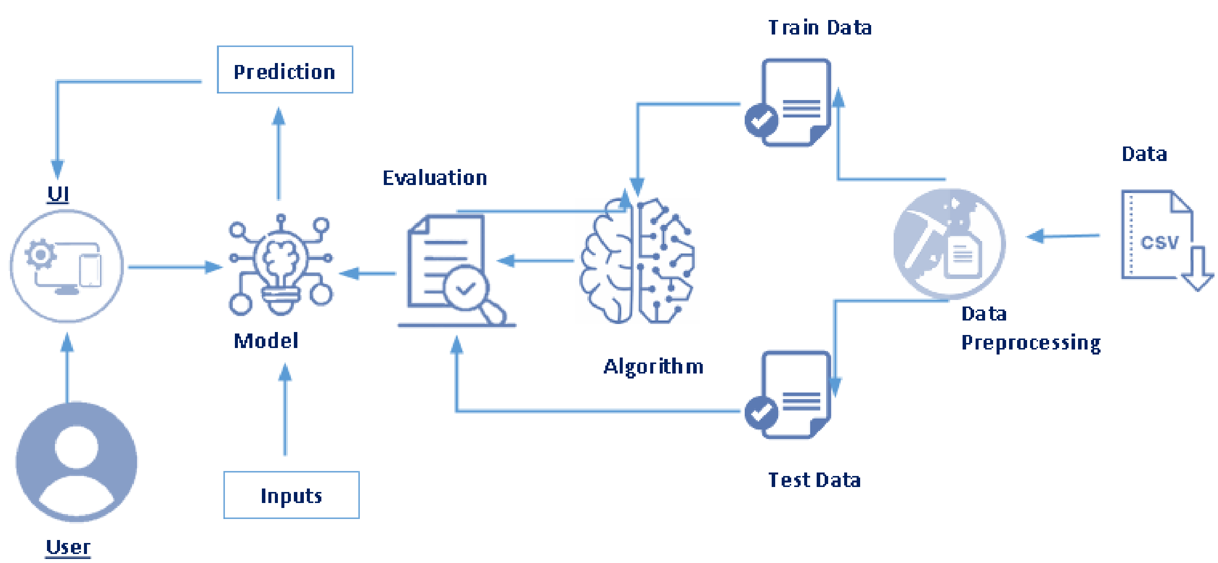


Figure 1.1. Technical Architecture

Flight arrival delays can be predicted using a machine learning algorithm. Rows of feature vectors, such as departure date, delay, travel time between the two airports, and scheduled arrival time, provide the input to our algorithm. The decision tree classifier is then used to determine whether or not the flight arrival will be delayed. When there is more than a 15-minute gap between the scheduled and actual arrival timings, a flight is deemed to be delayed. For various figures of merit, we contrast the decision tree classifier with logistic regression and a straightforward neural network.Flight arrival delays can be predicted using a machine learning algorithm. Rows of feature vectors, such as departure date, delay, travel time between input to our algorithm. The decision tree classifier is then used to determine whether or not the flight arrival will be delayed. When there is more than a 15-minute gap between the scheduled and actual arrival timings, a flight is deemed to be delayed.

1.2. PURPOSE

The main goal of this project is to predict the flight delay using machine learning algorithms. Flight planning is one of the difficulties in the industrial environment because there are many un predictabilities. One such condition is the incidence of delays, which can result from a variety of causes and impose significant expenses on airlines, operators, and passengers. Delays in departure can be brought on by inclement weather, seasonal and holiday demands, airline policies, technical issues with airport infrastructure, baggage handling, and mechanical equipment, and a build-up of delays from earlier flights. Hence Predicting flight delays can improve airline operations and passenger satisfaction, which will result in a positive impact on the economy.

CHAPTER 2
LITERATURE SURVEY

AUTHOR:Mohamed

YEAR:2020

The main concern of the researchers and analysts is to predict the reasons for flight delays and for that they have put in their efforts on collecting data about flight and the weather. He have studied the pattern of arrival delay for non-stop domestic flights at the Orlando International Airport. They focused primarily on the cyclic variations that happen in the air travel demand and the weather at that particular airport.

AUTHOR:Adrian

YEAR:2020

He have created a data mining model which enables the flight delays by observing the weather conditions. They have used WEKA and R to build their models by selecting different classifiers and choosing the one with the best results. They have used different machine learning techniques like Naive Bayes and Linear Discriminant Analysis classifier.

AUTHOR:Choi

YEAR:2020

They have employed Long Short-Term Memory RNN architecture trying to prove that the accuracy increases with deeper architectures. To train the model,stochastic gradient descent (SGD) algorithm is utilized. Use of SGDbhelped prevent Overfitting and increase general performance. The comparison of accuracies obtained with different number of layers has been formulated to support the claim of accuracy increasing with the increase in number of layers. The accuracy further improved with increasing epochs. The model has then been used to calculate and compare the delay of individual flights which manifests promising results.

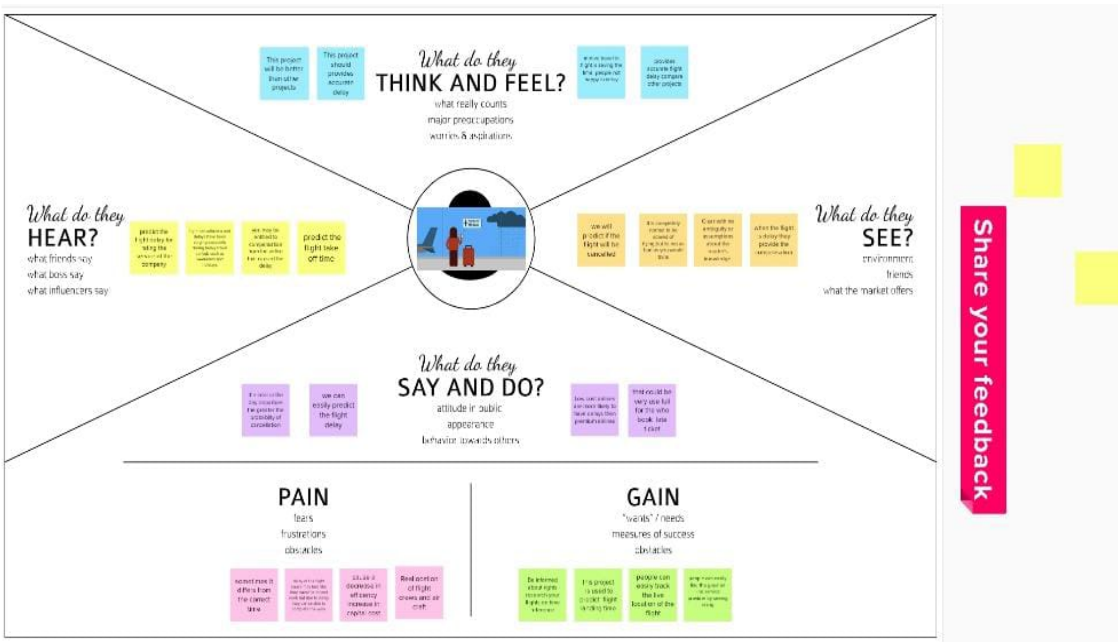
AUTHOR:Navoneel chakrabarty

YEAR:2020

Applied Gradient Boosting Classifier to analyse and predict possible arrival delay. Data balancing is done using Randomized SMOTE technique which in turn helped improve validation accuracy. A 200% Randomized-SMOTE is done on the dataset to reduce the imbalance between classes. There have been two strategies followed and compared throughout the paper. In strategy 1, the data imbalance removal step has been skipped and in strategy 2 it has been followed.

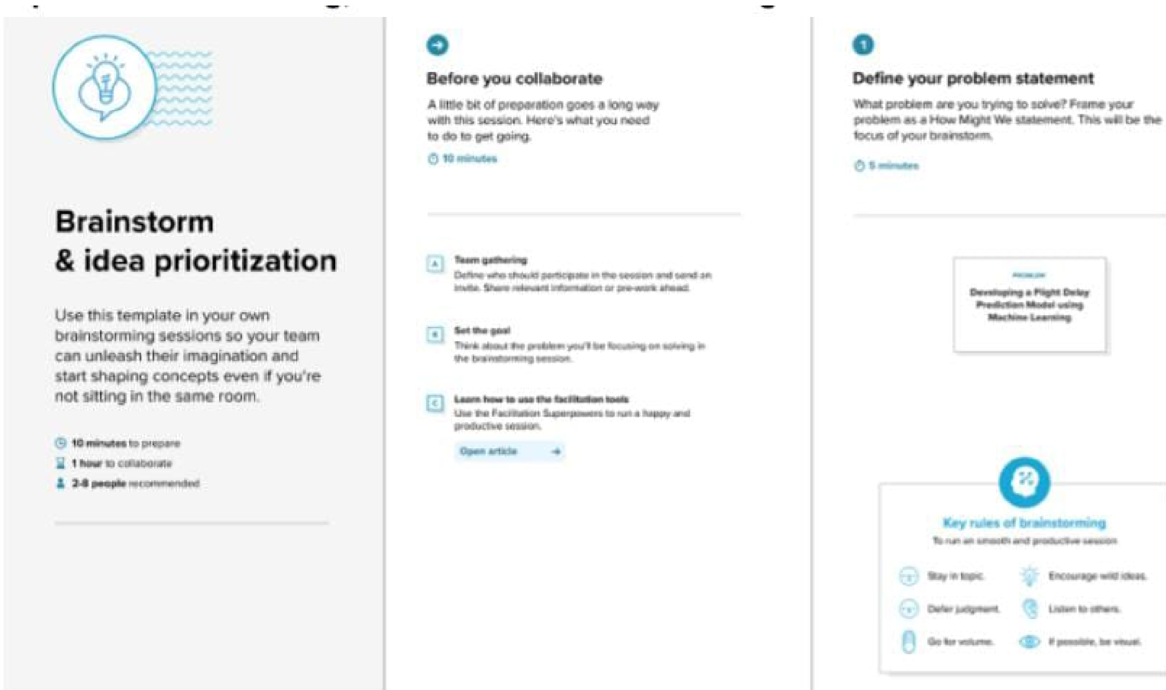
CHAPTER 3
IDEATION & PROPOSED SOLUTION

3.1.EMPATHY MAP CANVAS

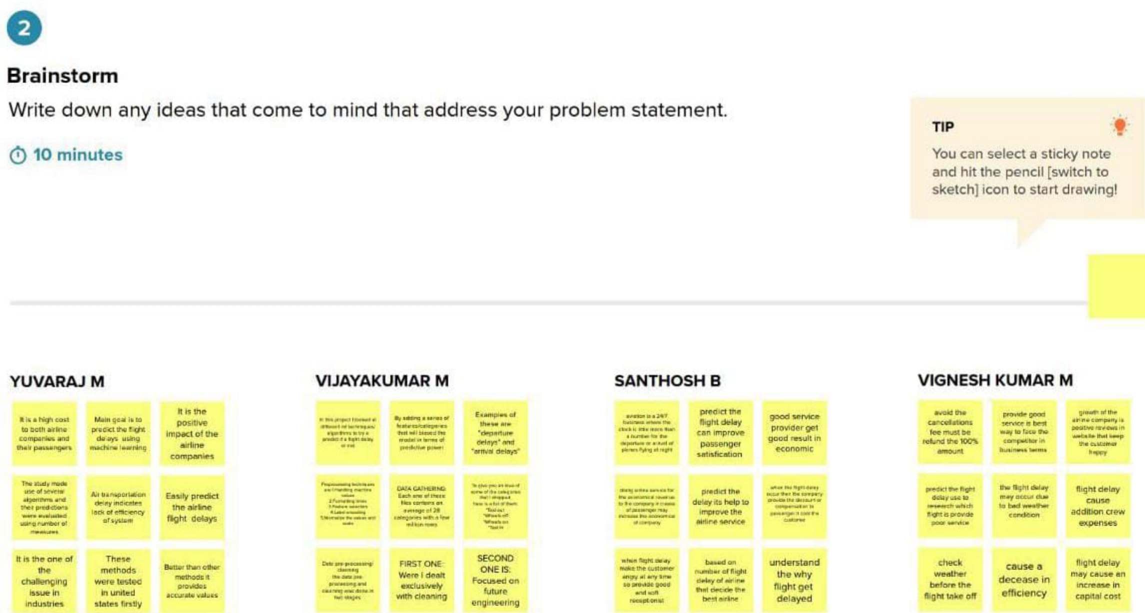


3.2. IDEATION & BRAINSTORMING

Step 1 - Team Gathering, Collaboration and Selecting the Problem Statement



STEP 2-Brainstorm,Idea Listing and Grouping

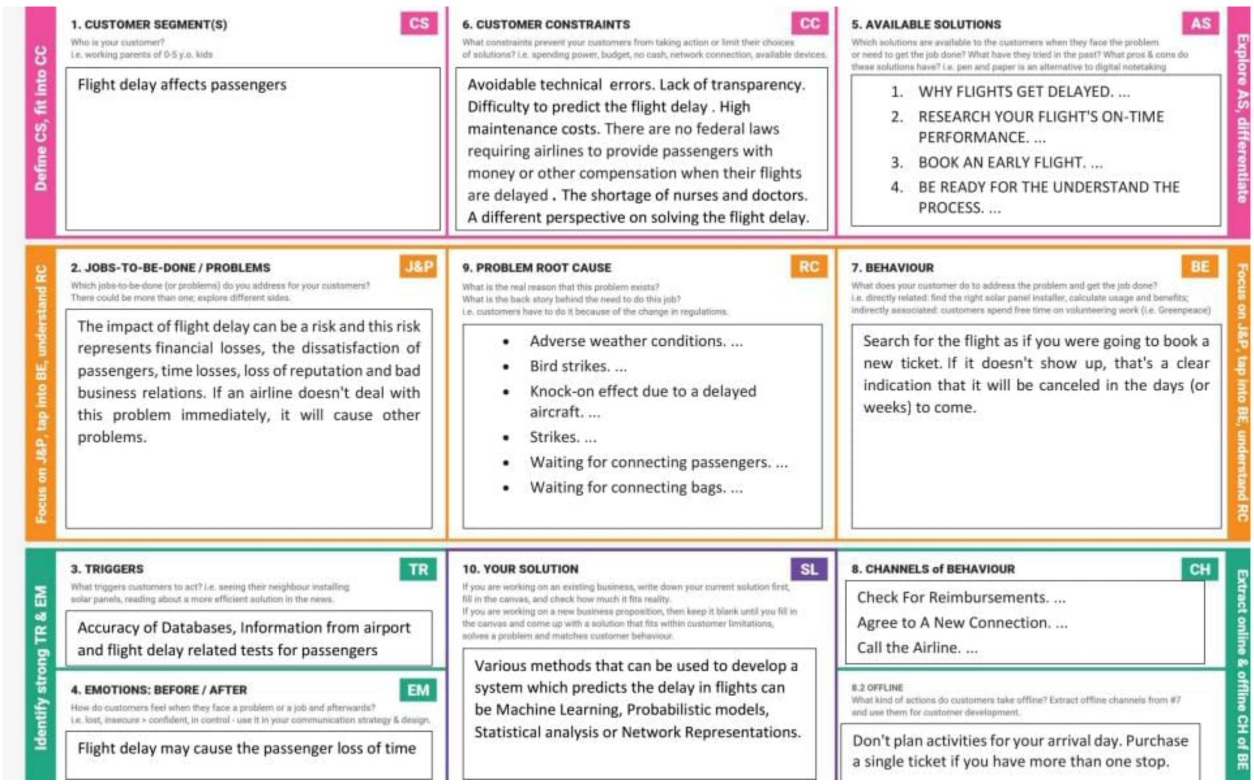


3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Flight delays can be very annoying toairlines, airports, andpassengers. Moreover, the development of accurateprediction models for flight delays became very difficult dueto the complexity of air transportation flight data. In thisproject, we try to resolve this problem with approaches usedto build flight delay prediction models using BPN and RadialBasis Function. Flight delays are gradually increasing and bring more financial difficulties and customer dissatisfaction to airline companies. To resolve this situation, supervised machine learning models were implemented to predict flight delays. The data set that records information of flights departing from JFK airport during one year was used for the prediction. Seven algorithms (Logistic Regression, K-Nearest Neighbor, Gaussian Naïve Bayes, Decision Tree, Support Vector Machine, Random Forest, and Gradient Boosted Tree) were trained and tested to complete the binary classification of flight delays. The evaluation of algorithms was fulfilled by comparing the values of four measures: accuracy, precision, recall, and f1-score. These measures were weighted to adjust the imbalance of the selected data set. The comparative analysis showed that the Decision Tree algorithm has the best performance with an accuracy of 0.9777, and the KNN algorithm has the worst performance with an f1-score of 0.8039. Tree-based ensemble classifiers generally have better performance over other base classifiers.
2.	Idea / Solution description	As discussed, weather condition plays an important role in proper and timely functioning of flights. We propose a flight delay prediction system which focuses mainly on predicting delay of a flight based on the weather situation. To make the system more scalable it is necessary to

		choose an algorithm which considers all the parameters to be independent.
3.	Novelty / Uniqueness	predicting flight delays can improve airline operations and passenger satisfaction, which will result in a positive impact on the economy. In this study, the main goal is to compare the performance of machine learning classification algorithms when predicting flight delays.
4.	Social Impact / Customer Satisfaction	Flight delays not only irritate air passengers and disrupt their schedules but also cause a decrease in efficiency, an increase in capital costs, reallocation of flight crews and aircraft, and additional crew expenses.So using this flight deley prediction will be used for peoples or passengers
5.	Business Model (Revenue Model)	Business model requires analysis tools that estimates the probability of an event based on the historic data. The estimated outcome is given in form of a distribution function of the probability. The factor of randomness always makes an impact on the decision or the outcome produced by the business model.
6.	Scalability of the Solution	The results show a high accuracy in prediction of delays above a given threshold.For instance, with a delay threshold of 60 minutes we achieve an accuracy of 85.8% and a delay recall of 86.9%. We also consider the effects on performance of varying model parameters, such as the classification threshold or the number of weather observations used. The goal of this work is to implement a predictor of the arrival delay of a scheduled flight due to weather conditions.

3.4 PROBLEM SOLUTION FIT



CHAPTER 4
REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN Registration through Facebook
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Data Management	The administrator of the mobile portal can add, amend, or remove the department data.
FR-4	Data Storage	• Application system under consideration manages the archival, retrieval, and retention of historical data • Sufficient information comprising these
FR-5	Process for Exporting Flight Data	A procedure where mobile client users obtain flight data from a web server for web client analysis.
FR-6	Flight Data	The administrator of the mobile portal can add, amend, or remove passenger's data.
FR-7	Regulatory Conditions	This paper proposes a model for predicting the flight delay based on the decision tree. A decision tree is a supervised machine learning tool that may be used to classify or forecast data based on how queries from the past have been answered. The model is supervised learning in nature, which means that it is trained and tested using data sets that contain the required categorisation.

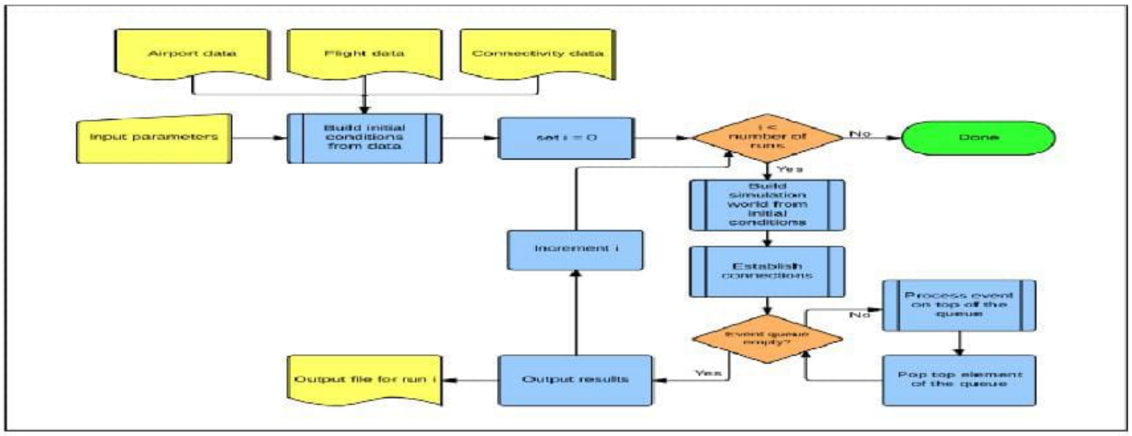
4.2. NON-FUNCTIONAL REQUIREMENTS

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Knowing when the flight will be delayed enables improved operational planning at the airport of destination based on anticipated flight delay at origin.
NFR-2	**Security**	It is highly secure and the passengers who log in to the application will be able to view the status
NFR-3	**Reliability**	As we train with more data the model will be reliable.
NFR-4	**Performance**	• This was done statistically, and the delay time was thought to be shorter. • Using variables that occur close to the destination's arrival time, it has projected the delay at the destination.
NFR-5	**Availability**	The mobile application must be accessible to users in India 99.98% of the time each month between EST and IST business hours.
NFR-6	**Scalability**	• The main problem for airlines and travellers is flight delay. • According to the flight schedule, the anticipated arrival delay considers both flight information and the weather at the airports of origin and destination.

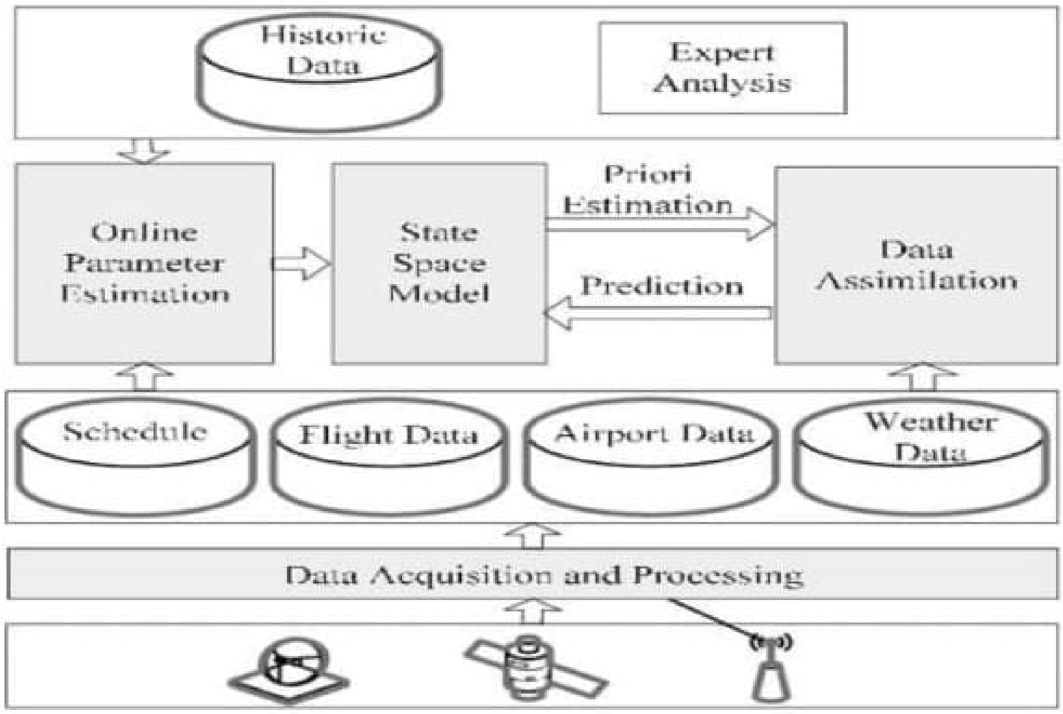
CHAPTER 5
PROJECT DESIGN

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2. SOLUTION & TECHNICAL ARCHITECTURE



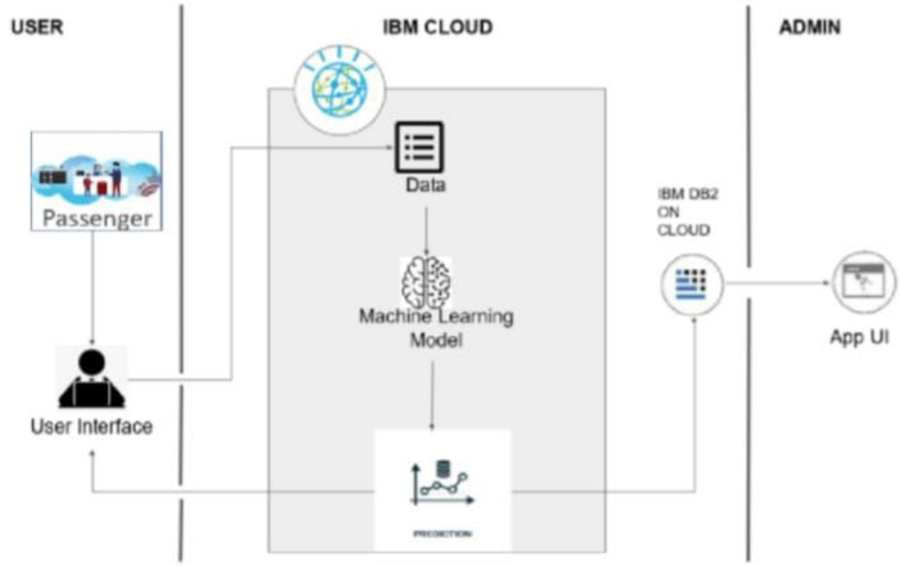


Figure 5.3. Technology Stack

Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	Web Application to interact with the user.	Flask
2.	Login/Sign up	Login/ Sign up – The user can enter the details and get them validated	Python
3.	Database	The Database to store the login details of the user	MySQL
4.	Cloud Database	The database to keep track of the flight details from the travel agency, input to the Machine Learning Model	Firebase
5.	Machine Learning Model	To Predict whether the flight will get delayed or not.	SVM, KNN Classifier, Logistic Regression, Decision Trees
6.	Deep Learning Model	To Predict whether the flight will get delayed or not	Fully Connected Neural Networks
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	IBM Cloud

APPLICATION CHARACTERITICS

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Web application – Flask ML – Sklearn, Tensorflow, Keras API	Deep Learning, Python
2.	Security Implementations	The data is secured that it is encrypted in IBM cloud	AES (256-bit)
3.	Scalable Architecture	Can be scaled upto many airports, many users with more training	Firebase
4.	Availability	The status will be updated frequently	IBM Cloud
5.	Performance	Can make as many number of requests per second to get the prediction	IBM Cloud

USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register & access the dashboard with Gmail Login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login & access the dashboard	High	Sprint-1
	Core	USN-6	As a user, I can enter my flight details	I can feed the inputs to the system	High	Sprint-2
		USN-7	As a user, I can look at the flight details	I can see whether my flight is getting delayed or not	High	Sprint-3

CHAPTER 6
PLANNING PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
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	M Yuvaraj
Sprint-1	User Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the web application.	1	Medium	B Santhosh M Vijayakumar
Sprint-1	Login	USN-3	As a user, I can login to the application by entering my email & password.	1	High	M Vigneshkumar

6.2 SPRINT DELIVERY SCHEDULE

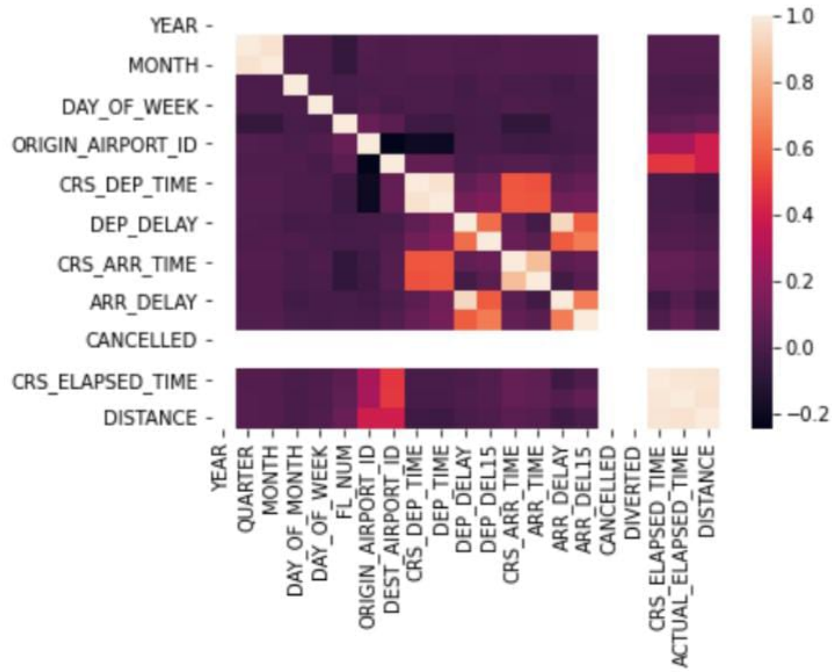
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	4 Days	24 Oct 2022	27 Oct 2022	20	29 Oct 2022
Sprint-2	20	5 Days	28 Oct 2022	01 Nov 2022	20	04 Nov 2022
Sprint-3	20	8 Days	02 Nov 2022	09 Nov 2022	20	11 Nov 2022
Sprint-4	20	9 Days	10 Nov 2022	18 Nov 2022	20	19 Nov 2022

Figure 6.1 - Burndown Chart

CHAPTER 7
CODING AND SOLUTION
7.1.FEATURE 1-CORRELATION BETWEEN THE VARIABLES IN DATASET

```
In [19]: sns.heatmap(data.corr())

Out[19]: <AxesSubplot:>
```



This will help us to find out the correlation between the variables in the dataset which would help us to find out the columns that are unnecessary and hence to be dropped.

7.2.FEATURE 2 -ONE HOT ENCODING

```
In [39]: data=pd.get_dummies(data,columns=['ORIGIN','DEST'])

In [40]: data['ARR_DEL15'].value_counts()

Out[40]: 0.0    9668
         1.0    1375
         Name: ARR_DEL15, dtype: int64

In [41]: data.tail()

Out[41]:
```

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15	ORIGIN_0	ORIGIN_1	ORIGIN_2	ORIGIN_3	ORIGIN_4
11226	1715	12	30	5	12	0.0	0.0	0	1	0	0	0
11227	1770	12	30	5	20	1.0	0.0	0	0	0	0	1
11228	1823	12	30	5	22	0.0	0.0	0	1	0	0	0
11229	1901	12	30	5	18	0.0	0.0	1	0	0	0	0
11230	2005	12	30	5	9	0.0	0.0	1	0	0	0	0

The cities in both Origin and Destination are one-hot encoded using the above code.

7.3.FEATURE 3-SAVING THE MODEL WEIGHTS FOR DEPLOYEMENT

SAVING THE MODEL

```
In [63]: pickle.dump(classifier,open('flight_new.pk1','wb'))

In [64]: from sklearn.metrics import confusion_matrix
         confusion_matrix(predicted, y_test)

Out[64]: array([[1825,  129],
                [ 138,  117]], dtype=int64)

In [66]: from sklearn.metrics import classification_report
         print(classification_report(predicted, y_test, labels=[1, 2]))

                  precision    recall  f1-score   support

               1       0.48      0.46      0.47       255
               2       0.00      0.00      0.00         0

       micro avg       0.48      0.46      0.47       255
       macro avg       0.24      0.23      0.23       255
    weighted avg       0.48      0.46      0.47       255
```

The above code will save the model weights for further deployment in IBM Cloud and also measure the performance metrics.

7.4. FEATURE 4-FLASK INTERFACE-UI

```
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
import pickle
import os
model = pickle.load(open('flight_new.pk1','rb'))
app = Flask(_name_)
@app.route('/')
def home():
return render_template("mainpage.html")
@app.route('/prediction',methods=['GET','POST'])
def predict():
name = request.form['fname']
month = request.form['month']
dayofmonth = request.form['daymonth']
dayofweek = request.form['dayweek']
origin = request.form['origin']
```

```
if(origin == "msp"):
origin1, origin2, origin3, origin4, origin5 = 0,0,0,0,1
if(origin == "dtw"):
origin1, origin2, origin3, origin4, origin5 = 1,0,0,0,0
if(origin == "jfk"):
origin1, origin2, origin3, origin4, origin5 = 0,0,1,0,0,
if(origin == "sea"):
origin1, origin2, origin3, origin4, origin5 = 0,1,0,0,0
if(origin == "atl"):
origin1, origin2, origin3, origin4, origin5 = 0,0,0,1,0
destination = request.form['destination']
if(destination == "msp"):
destination1,destination2,destination3,destination4,destination5 = 0,0,0,0,1
if(destination == "dtw"):
destination1,destination2,destination3,destination4,destination5 = 1,0,0,0,0
if(destination == "jfk"):
destination1,destination2,destination3,destination4,destination5 = 0,0,1,0,0
if(destination == "sea"):
destination1,destination2,destination3,destination4,destination5 = 0,1,0,0,0
if(destination == "atl"):
destination1,destination2,destination3,destination4,destination5 = 0,0,0,1,0
dept = request.form['sdeparttime']
arrtime = request.form['sarrivaltime']
actdept = request.form['adeparttime']
dept15 = int(dept)-int(actdept)
total =
[[name,month,dayofmonth,dayofweek,arrtime,dept15,origin1,origin2,origin3,origin4,origi
n5,destination1,destination2,destination3,destination4,destination5]]
y_pred = model.predict(total)
print(y_pred)
if(y_pred == [0.]):
ans = "The Flight will be on time"
else:
ans = "The Flight will be delayed"
return render_template("index.html",data = ans)

app.run(debug=True)
```

Explanation:

The above code will be able to get the details of the flight from the user in the respective text fields created using the HTML, scale the inputs and give the inputs to the model which has been developed already. The predictions are shown in another HTML page.

7.5.FEATURE 5-HTML PAGES FOR FRONTEND DESIGN

```
 <html>
<div align="center" class="logbg">
<head>
<meta charset="UTF-8">
<center>
<table>
<tr>
<td><h1><br>Prediction of Flight Delay<br><br></h1></td>
</tr>
</table>
</center>
</head>
<body background='C:\Users\Public\project\templates\flight_4.jpg'>
<form action="http://localhost:5000/prediction" method="POST" >
<center>
<table>
<tr>
<td>Enter the flight number:</td>
<td><input type="number" name="fname"><br></td>
</tr>
<tr>
<td>Month:</td>
<td><input type="number" name="month"><br></td>
</tr>
<tr>
<td>Day of Month:</td>
<td><input type="number" name="daymonth"><br></td>
</tr>
<tr>
<td>Day of Week:</td>
<td><input type="number" name="dayweek"><br></td>
</tr>
<tr>
<td>Origin:</td>
```

```
<td><select name="origin">
<option value="atl">ATL</option>
<option value="dtw">DTW</option>
<option value="sea">SEA</option>
<option value="msp">MSP</option>
<option value="jfk">JFK</option>
</select></td>
<tr>
<tr>
<td>Destination:</td>
<td><select name="destination">
<option value="atl">ATL</option>
<option value="dtw">DTW</option>
<option value="sea">SEA</option>
<option value="msp">MSP</option>
<option value="jfk">JFK</option>
</select></td>
<tr>
<tr>
<td>Scheduled Departure Time:</td>
<td><input type="number" name="sdeparttime"><br></td>
</tr>
<tr>
<td>Scheduled Arrival Time:</td>
<td><input type="number" name="sarrivaltime"><br></td>
</tr>
<tr>
<td>Actual Departure Time:</td>
<td><input type="number" name="adeparttime"><br></td>
</tr>
<tr>
<td><br><input type="submit" class="btn" value="SUBMIT"></br>
</tr>
</table>
</center>
</form>
</body>
</div>
</html>
```

CHAPTER-8
TESTING

8.1.TEST

User No	Flight No	Month	Day of month	Day of week	Origin	Destination	Scheduled Departure Time	Scheduled Arrival Time	Actual Departure Time	Actual Inputs
1	1232	1	1	1	ATL	MSP	1905	2305	1945	Delayed
2	1399	1	1	1	ATL	SEA	1805	2410	1855	Delayed
3	2351	1	2	3	ATL	DTW	1305	2305	1305	Not Delayed
4	2637	2	1	3	DTW	ATL	1500	2410	1505	Not Delayed

8.2.USER ACCEPTANCE TESTING

This report shows the number of test cases that have passed and failed

User No	Flight No	Month	Day Of Month	Day Of Week	Origin	Destin -ation	Scheduled Departure Time	Scheduled Arrival Time	Actual Departure Time	Actual Output	Predict -ed Output	Correct-ne ss
1	1232	1	1	1	ATL	MSP	1905	2305	1945	Delayed	Delayed	Correct
2	1399	1	1	1	ATL	SEA	1805	2410	1855	Delayed	Delayed	Correct
3	2351	1	2	3	ATL	DTW	1305	2305	1305	Not Delayed	Not Delayed	Correct
4	2637	2	1	3	DTW	ATL	1500	2410	1505	Not Delayed	Not Delayed	Correct

CHAPTER 9
RESULTS

9.1.PERFORMANCE METRICES

Training Accuracy

Model Evaluation

```
acc=accuracy_score(predicted,y_test)
```

```
acc
```

```
0.8791308284291535
```

Confusion Matrix

```
from sklearn.metrics import confusion_matrix
confusion_matrix(predicted, y_test)
```

```
array([[1825,  129],
       [ 138,  117]], dtype=int64)
```

Classification Model

```
from sklearn.metrics import classification_report
print(classification_report(predicted, y_test, labels=[1, 2, 3]))
```

```
              precision    recall  f1-score   support

           1       0.48      0.46      0.47       255
           2       0.00      0.00      0.00         0
           3       0.00      0.00      0.00         0

   micro avg       0.48      0.46      0.47       255
   macro avg       0.16      0.15      0.16       255
weighted avg       0.48      0.46      0.47       255
```

CHAPTER 10
ADVANTAGES AND DISADVANTAGES

Advantages

- Customers are happy
- The available flights are easily identified
- Prior information will be sent if in case the flight is delayed
- The current status of the flight can be tracked

Disadvantages

- Wrong prediction due to noise of input data
- If the prediction is wrong, then there will be extra expenses for the agencies,passengers and airport
- Passengers with medical emergencies gets affected

CHAPTER 11
CONCLUSION

In this project, we use flight data, weather, and demand data to predict light-hearted delay. In the end, our model correctly predicts the delayed and non-delayed flights correctly. As a result, there can be additional features related to the causes off light delay that are not yet discovered using our existing data sources.

CHAPTER 12
FUTURE SCOPE

Based on data analysis from the year 2008, this project. There is a sizable dataset accessible from 1987 to 2008, but managing a larger dataset necessitates extensive preprocessing and purification of the data Therefore, adding a larger dataset is a part of this project's future effort. Preprocessing a bigger dataset can be done in a variety of methods, such as establishing a Spark cluster on a computer or using cloud services like AWS and Azure. Now that deep learning has advanced, we can employ neural networks algorithms to analyse aviation and meteorological data. Neural networks employ a form of pattern matching.The project's focus is primarily on flight and weather data for India, but we can also include data from other nations like China, the United States, and Russia. We can broaden the project's scope by including flight information from international flights rather than just domestic flights.

CHAPTER 13

APPENDIX

13.1 Source codes

13.1.2 Exploratory Data Analysis

```
#!/usr/bin/env python
# coding: utf-8

# **Importing all the libraries**
# In[1]:

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
get_ipython().run_line_magic('matplotlib', 'inline')
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics

# **Importing the dataset**

# In[2]:

data=pd.read_csv("flightdata.csv")

# In[3]:

data.head()

# In[4]:
```

```
data.info()

# In[5]:

data=data.drop('Unnamed: 25',axis=1)

# In[6]:

data.info()

# In[7]:

data.describe()

# In[ ]:

# *Handling Missing Values*

# In[8]:

data=data.dropna()

# In[9]:

data.info()

# *Analysis
*
# In[10]:

plt.scatter(data.index,data['ARR_TIME'])
plt.ylabel('Arrival Time')
plt.title('Distribution of the Arrival Time')

# In[11]:

plt.hist(data['FL_NUM'])

# In[12]:
```

columns=list(data.columns)

In[13]:

sns.scatterplot(x='ARR_DELAY',y='ARR_DEL15',data=data)

In[14]:

sns.catplot(x='ARR_DELAY',y='ARR_DEL15',data=data,kind='bar')

In[15]:

data['ARR_DEL15'].nunique()

In[16]:

In[17]:

data.describe()

Dropping off unnecessary columns

In[18]:

data.corr()['ARR_DEL15']

In[19]:

sns.heatmap(data.corr())

In[20]:

new_data=data.drop(['ORIGIN_AIRPORT_ID','DEST_AIRPORT_ID','FL_NUM','YEAR','
CANCELLED','DIVERTED','DISTANCE','DAY_OF_MONTH','QUARTER','MONTH','DAY
_OF_WEEK','UNIQUE_CARRIER','TAIL_NUM'],axis=1)

In[21]:

new_data.head()

```
# *Label Encoding*
# In[22]:
cities=new_data['ORIGIN'].unique()

# In[23]:

cities

# In[24]:

new_data['DEST'].unique()

# In[25]:

city_map={cities[i]:i for i in range(0,len(cities))}

# In[26]:

city_map

# In[27]:

def encode(c):
  return city_map[c]

# In[28]:

new_data['ORIGIN']=new_data['ORIGIN'].apply(encode)

# In[29]:

new_data['DEST']=new_data['DEST'].apply(encode)

# In[30]:

new_data.head()

# In[31]:

new_data.corr()['ARR_DEL15']
```

In[32]:

```
#data=data.drop('Unnamed: 25',axis=1)
data.isnull().sum()
```

In[33]:

```
data=data[["FL_NUM","MONTH","DAY_OF_MONTH","DAY_OF_WEEK","ORIGIN","DE
ST","CRS_ARR_TIME","DEP_DEL15","ARR_DEL15"]]
data.isnull().sum()
```

#

In[34]:

```
data=data.fillna({'ARR_DEL15': 1})
data=data.fillna({'DEP_DEL15': 0})
data.iloc[177:185]
```

In[35]:

```
import math
for index, row in data.iterrows():
  data.loc[index,'CRS_ARR_TIME'] = math.floor(row['CRS_ARR_TIME'] / 100)
data.head()
```

In[36]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['DEST'] = le.fit_transform(data['DEST'])
data['ORIGIN'] = le.fit_transform(data['ORIGIN'])
```

In[37]:

```
data.head()
```

In[38]:

```
from sklearn.preprocessing import OneHotEncoder
oh = OneHotEncoder()
z=oh.fit_transform(data['ORIGIN'].values.reshape(-1,1)).toarray()
t=oh.fit_transform(data['DEST'].values.reshape(-1,1)).toarray()

# In[ ]:

# In[ ]:

# In[39]:

data=pd.get_dummies(data,columns=['ORIGIN','DEST'])

# In[40]:

data['ARR_DEL15'].value_counts()

# In[41]:

data.tail()

# **Split the data into dependent and independent variables**
#
# In[42]:

x=data[[i for i in data.columns if i!='ARR_DEL15']].values
y=data[[i for i in data.columns if i=='ARR_DEL15']].values

# In[43]:

x.shape

# In[44]:

y.shape

# In[ ]:
```

CHAPTER 13

APPENDIX

13.1. SOURCE CODE

13.1.1. Train the ML Model

```
# # SPRINT-2

# **TRAIN-TEST-SPLIT**

# In[45]:

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

# In[46]:

x_test.shape

# In[47]:

x_train.shape

# In[48]:

y_test.shape

# In[49]:

y_train.shape

# **Scaling**

# In[50]:

sc = StandardScaler()

# In[51]:
```

```
x_train=sc.fit_transform(x_train)
# In[52]:

x_test=sc.fit_transform(x_test)

# **Model Building**

# In[53]:

classifier = DecisionTreeClassifier(random_state=0)

# In[54]:

classifier.fit(x_train,y_train)

# In[55]:

predicted = classifier.predict(x_test)

# In[56]:

predicted

# In[57]:

y_test

# **MODEL EVALUATION**

# In[58]:

acc=accuracy_score(predicted,y_test)

# In[59]:

acc

# In[ ]:
```

```
# In[60]:
data[data['ARR_DEL15']>0].iloc[33].values

# In[61]:

sample=[[1.187e+03, 1.000e+00, 1.500e+01, 5.000e+00, 1.900e+01, 1.000e+00,
0.000e+00, 0.000e+00, 0.000e+00, 1.000e+00, 0.000e+00,
       0.000e+00, 0.000e+00, 0.000e+00, 0.000e+00, 1.000e+00]]

# In[62]:

classifier.predict(sample)

# **SAVING THE MODEL**

# In[63]:

pickle.dump(classifier,open('flight_new.pk1','wb'))

# In[64]:

from sklearn.metrics import confusion_matrix
confusion_matrix(predicted, y_test)

# In[66]:

from sklearn.metrics import classification_report
print(classification_report(predicted, y_test, labels=[1, 2]))

# In[ ]:
```

13.1.2. Main page – HTML Code

```html
<html>
<div align="center" class="logbg">
<head>
<meta charset="UTF-8">
<center>
<table>
<tr>
<td><h1><br>Prediction of Flight Delay<br><br></h1></td>
</tr>
</table>
</center>
</head>
<body background='C:\Users\Public\project\templates\flight_4.jpg'>
<form action="http://localhost:5000/prediction" method="POST" >
<center>
<table>
<tr>
<td>Enter the flight number:</td>
<td><input type="number" name="fname"><br></td>
</tr>
<tr>
<td>Month:</td>
<td><input type="number" name="month"><br></td>
</tr>
<tr>
<td>Day of Month:</td>
<td><input type="number" name="daymonth"><br></td>
</tr>
<tr>
<td>Day of Week:</td>
<td><input type="number" name="dayweek"><br></td>
</tr>
<tr>
<td>Origin:</td>
<td><select name="origin">
<option value="atl">ATL</option>
<option value="dtw">DTW</option>
```

```
<option value="sea">SEA</option>
<option value="msp">MSP</option>
<option value="jfk">JFK</option>
</select></td>
<tr>
<tr>
<td>Destination:</td>
<td><select name="destination">
<option value="atl">ATL</option>
<option value="dtw">DTW</option>
<option value="sea">SEA</option>
<option value="msp">MSP</option>
<option value="jfk">JFK</option>
</select></td>
<tr>
<tr>
<td>Scheduled Departure Time:</td>
<td><input type="number" name="sdeparttime"><br></td>
</tr>
<tr>
<td>Scheduled Arrival Time:</td>
<td><input type="number" name="sarrivaltime"><br></td>
</tr>
<tr>
<td>Actual Departure Time:</td>
<td><input type="number" name="adeparttime"><br></td>
</tr>
<tr>
<td><br><input type="submit" class="btn" value="SUBMIT"></br>
</tr>
</table>
</center>
</form>
</body>
</div>
</html>
```

13.1.3 Prediction Page - HTML Code

```
<!doctype html>
<html>
  <body background="C:\Users\Public\project\templates\flight_2.jpg">
        <center>
     <h1><strong>Thanks for asking</strong></h1>
     <h2>{{data}}</h2>
     <a href='/'>Go back to home page</a>
        </center>
  </body>
</html>
```

13.1.4. Flask Application

```
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
import pickle
import os
model = pickle.load(open('flight_new.pk1','rb'))
app = Flask(__name__)
@app.route('/')
def home():
return render_template("mainpage.html")

@app.route('/prediction',methods=['GET','POST'])
def predict():
  name = request.form['fname']
  month = request.form['month']
  dayofmonth = request.form['daymonth']
  dayofweek = request.form['dayweek']
  origin = request.form['origin']
  if(origin == "msp"):
     origin1, origin2, origin3, origin4, origin5 = 0,0,0,0,1
  if(origin == "dtw"):
     origin1, origin2, origin3, origin4, origin5 = 1,0,0,0,0
  if(origin == "jfk"):
      origin1, origin2, origin3, origin4, origin5 = 0,0,1,0,0,
  if(origin == "sea"):
```

```
        origin1, origin2, origin3, origin4, origin5 = 0,1,0,0,0
if(origin == "atl"):
     origin1, origin2, origin3, origin4, origin5 = 0,0,0,1,0
destination = request.form['destination']
 if(destination == "msp"):
destination1,destination2,destination3,destination4,destination5 = 0,0,0,0,1
 if(destination == "dtw"):
destination1,destination2,destination3,destination4,destination5 = 1,0,0,0,0
 if(destination == "jfk"):
 destination1,destination2,destination3,destination4,destination5 = 0,0,1,0,0
 if(destination == "sea"):
 destination1,destination2,destination3,destination4,destination5 = 0,1,0,0,0
 if(destination == "atl"):
 destination1,destination2,destination3,destination4,destination5 = 0,0,0,1,0
dept = request.form['sdeparttime']
arrtime = request.form['sarrivaltime']
actdept = request.form['adeparttime']
dept15 = int(dept)-int(actdept)
total =
[[name,month,dayofmonth,dayofweek,arrtime,dept15,origin1,origin2,origin3,origin4,origi
n5,destination1,destination2,destination3,destination4,destination5]]
    y_pred = model.predict(total)
    print(y_pred)
    if(y_pred == [0.]):
        ans = "The Flight will be on time"
    else:
        ans = "The Flight will be delayed"
    return render_template("index.html",data = ans)

app.run(debug=True)
```

13.2 GITHUB AND PROJECT DEMO LINK

Github link :

https://github.com/IBM-EPBL/IBM-Project-45648-1660731447

project demo link :

https://drive.google.com/file/d/1QWxzs9N8vwgU2WbKKkLPWNhi5EjYDNJL/view?usp=drivesdk