PROJECT REPORT

PROJECT TITLE

DEVELOPING A FLIGHT DELAY PREDICTION MODEL USING MACHINE LEARNING

TEAM ID : PNT2022TMID32458

TEAM MEMBERS : SUREN GOPAL D (TEAM LEAD)

SUNDARESAN M

SUNDARESVAR P

THULASI VEERARAGAVAN L

TABLE OF CONTENTS :

1.INTRODUCTION :

1.1 PROJECT OVERVIEW :

In the past two decades, air travel has become increasingly popular and has become increasingly accessible to people all over the world. Aviation has evolved to become one of the most important forms of transportation, with its efficiency and reliability making it the preferred choice for long-distance travel. However, flight delays are a major problem in the aviation industry, and they are becoming more and more common. In the United States, the average delay has increased by 30% since 2000, and the cost of delays has risen to $32 billion per year. There are many factors that can contribute to flight delays, such as weather, air traffic control, and maintenance. However, the most common cause of delays is simply that the plane is not ready to take off on time. This is usually due to the fact that the plane is not fully loaded with passengers, baggage, and fuel. It can also be due to technical problems with the plane itself. The goal of this project is to develop a machine learning model that can predict flight delays. The model will be trained on a dataset of historical flight data, and it will be used to predict the delay of a flight before it even takes off.

1.2 PURPOSE :

The purpose of this project is to develop a machine learning model that can predict flight delays. The model will be trained on a dataset of flight information, and will be used to predict the arrival delay of flights. The project is divided into two parts:

1. Data pre-processing and feature engineering
2. Model training and testing

In the first part, the data will be pre-processed and features will be engineered. This part will be focused on cleaning the data and making sure that the features are suitable for training the machine learning model. In the second part, the machine learning model will be trained and tested. This part will focus on tuning the model to get the best performance possible.

2. LITERATURE SURVEY :

2.1 EXISTING PROBLEM :

Flight planning, as one of the challenging issue in the industrial world, is faced with many uncertain conditions. One such condition is delay occurrence, which stems from various factors and imposes considerable costs

on airlines, operators, and travelers. With these considerations in mind, we implemented flight delay prediction through proposed approaches that are based on machine learning algorithms. Parameters that enable the effective estimation of delay are identified, after which Bayesian modeling, decision tree, cluster classification, random forest, and hybrid method are applied to estimate the occurrences and magnitude of delay in a network. These methods were tested on a U.S. flight dataset and then refined for a large Iranian airline network. Results showed that the parameters affecting delay in US networks are visibility, wind, and departure time, whereas those affecting delay in Iranian airline flights are fleet age and aircraft type. The proposed approaches exhibited an accuracy of more than 70% in calculating delay occurance and magnitude in both the whole-network US and Iranian. It is hoped that the techniques put forward in this work will enable airline companies to accurately predict delays, improve flight planning, and prevent delay propagation.

2.2 REFERENCES :

Thiagarajan B, et al. A machine learning approach for prediction of on-time performance of flights. In 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC). New York: IEEE. 2017.

Esmaeilzadeh, E., & Mokhtarimousavi, S. (2020). Machine learning approach for flight departure delay prediction and analysis. Transportation Research Record: Journal of the Transportation Research Board, *2674*(8), 145–159.

Yu, B., Guo, Z., Asian, S., Wang, H., & Chen, G. (2019). Flight delay prediction for commercial air transport: A deep learning approach. Transportation Research Part E: Logistics and Transportation Review, *125*, 203–221.

Ye, B., Liu, B., Tian, Y., & Wan, L. (2020). A methodology for predicting aggregate flight departure delays in airports based on supervised learning. Sustainability, *12*(7), 2749.

Liu YJ, Cao WD, Ma S. Estimation of arrival flight delay and delay propagation in a busy hub-airport. In 2008

Fourth International Conference on Natural Computation. New York: IEEE. 2008.

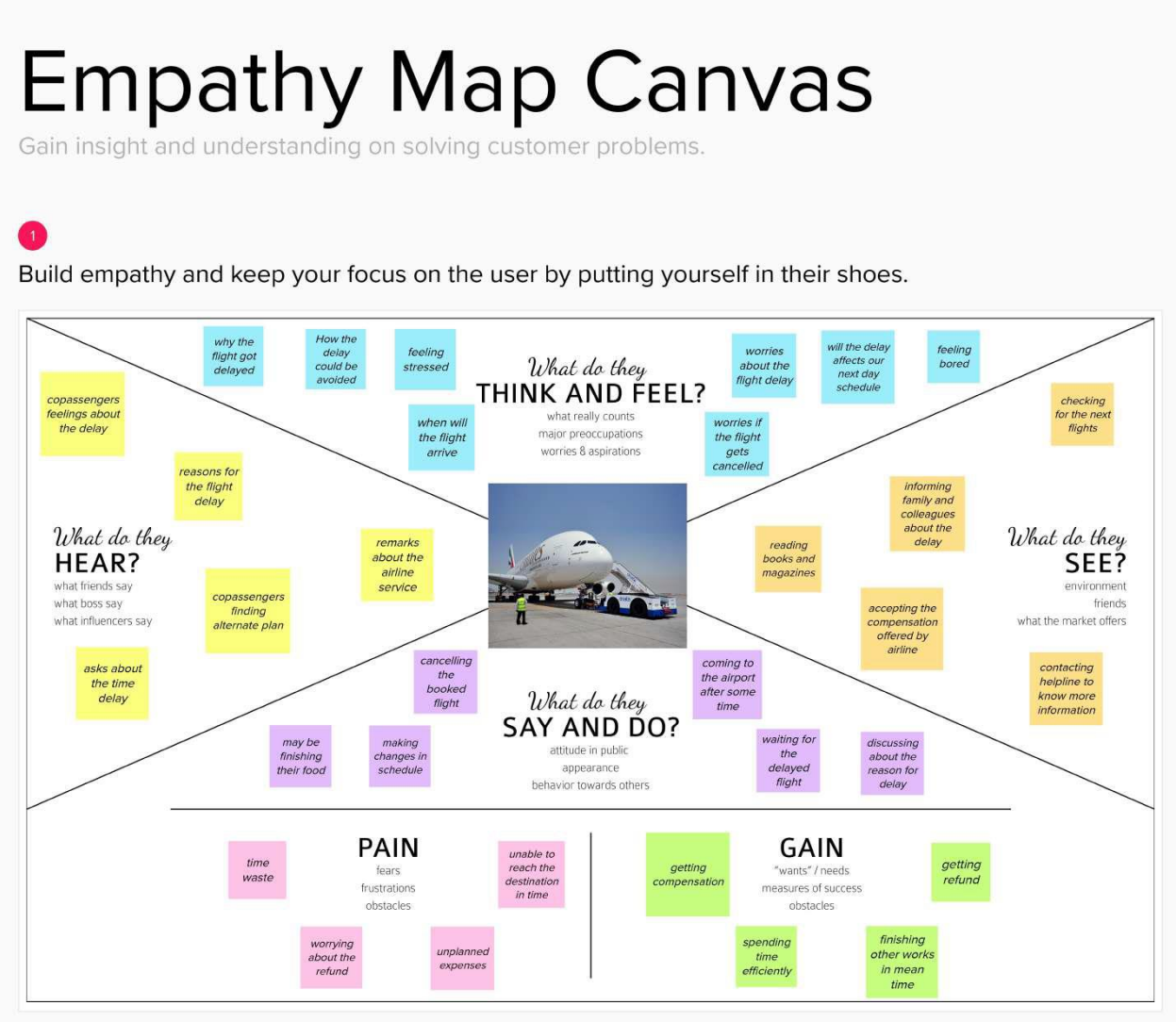
2.3 PROBLEM STATEMENT :

Operating a flight is challenging on so many different levels, largely because of all the different people involved.On the one hand, there are factors that are under the direct control of the carrier, such as aircraft turnarounds between flights, passenger punctuality, technical and crew performance, etc. On the other hand, there are perhaps even more factors that are outside of the airline's control, such as weather, air traffic control, security, airport conditions, etc.The reality is such that so long as airplanes continue flying, flight delays will be a part of the experience. According to the Bureau of Statistics, about 20% of all flights are delayed by 15 minutes or more. The aim of the project is to predict the time delay of the flight which reaches the destination due to various problems such as air traffic ,adverse weather conditions ,knock-on effect due to a delayed aircraft ,waiting for passengers ,waiting for baggages etc… But in this project we are considering only the important data such as departure time ,arrival time ,time delay of the flight and several other data by using the

machine learning algorithms such as Decision Tree ,XG Boost ,regression and classification algorithms.

3. IDEATION AND PROPOSED SOLUTION :

3.1 EMPATHY MAP CANVAS :



3.2 IDEATION AND BRAINSTORMING :

1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

DEVELOPING A FLIGHT DELAY PREDICTION MODEL USING MACHINE LEARNING

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

using the dataset which contains the necessary data

Taking the distance between the airports and departure time

SUREN GOPAL D

Taking arrival and departure time as data

Using weather data for processing

Referring research papers related to the project

Using Data pre processing and data cleaning

SUNDARESAN M

Referring to models which are already in use

Using feature engineering to process the data

Visualizing the dataset with various plots

Using Exploratory Data Analaysis

SUNDARESVAR P

Using a proper methodology for predicting the model

Using efficient training and testing methodologies

Using XG Boost algorithm

Using Decision Tree Algorithm

THULASI VEEERARAGAVAN L

Using Random Forest Algorithm

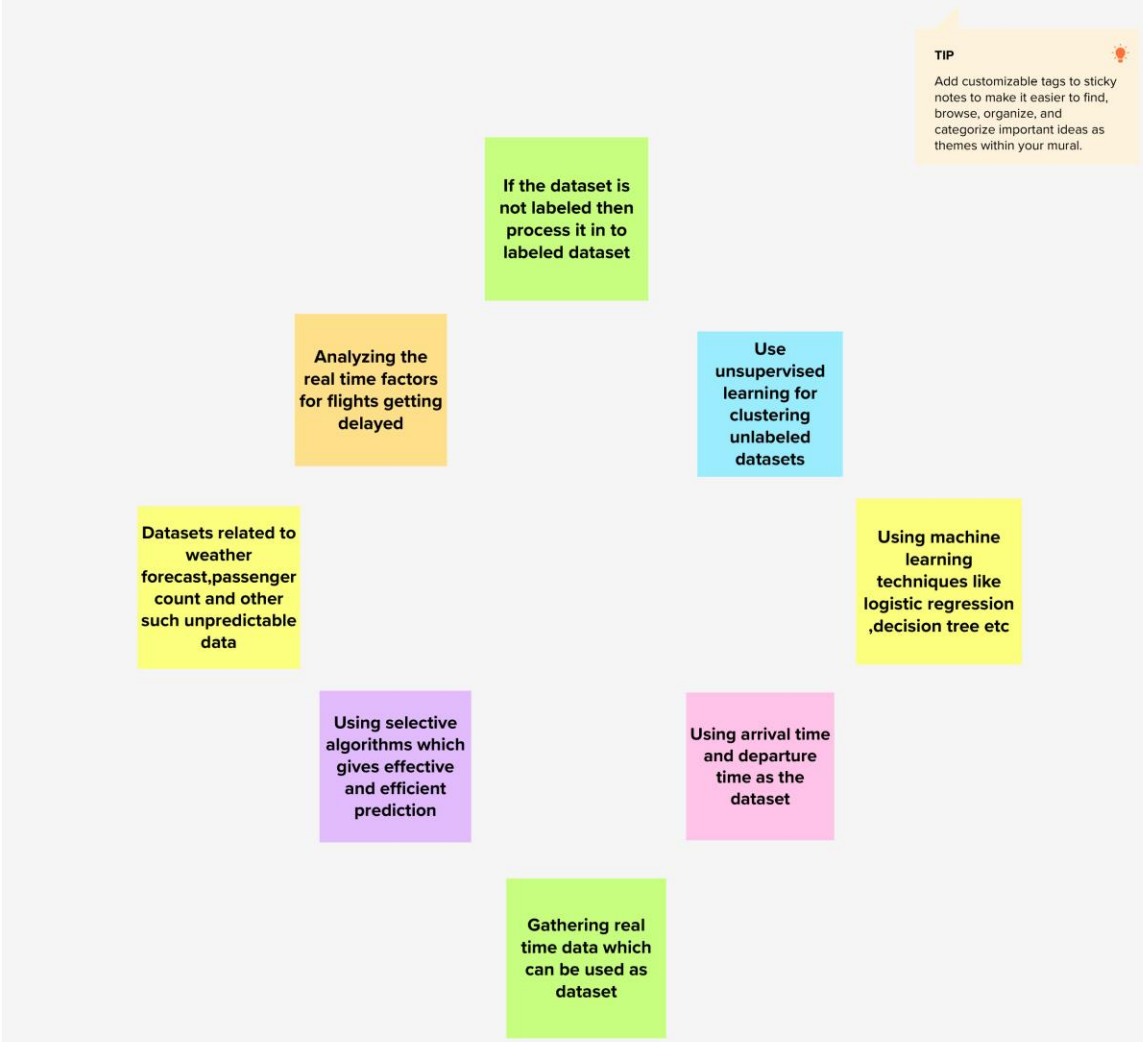
Using Deep Neural Network Modelling

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

3.3 PROPOSED SOLUTION :

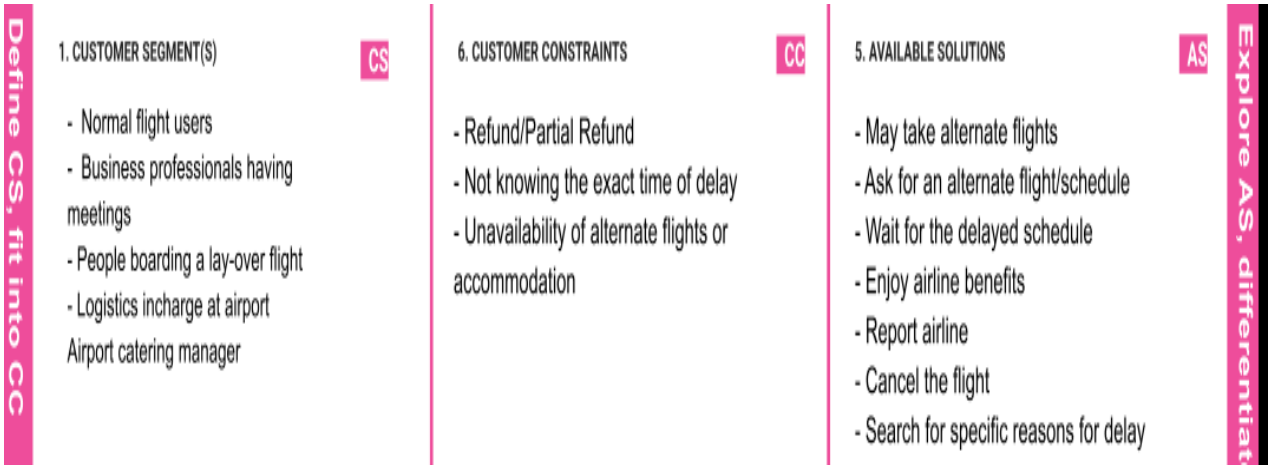
S.NO	PARAMETERS	DESCRIPTION
1.	Problem Statement (Problem to be solved)	Flight delays have been the most challenging area for airlines to improve. They have been affecting the air industry directly and indirectly causing unforeseen expenses thereby reducing the reputation of the industry and the airlines. Thus, knowing if a flight would be delayed beforehand can let passengers and airlines be prepared for the circumstances. This solution aims at making it possible by predicting arrival and departure delays using Machine learning.

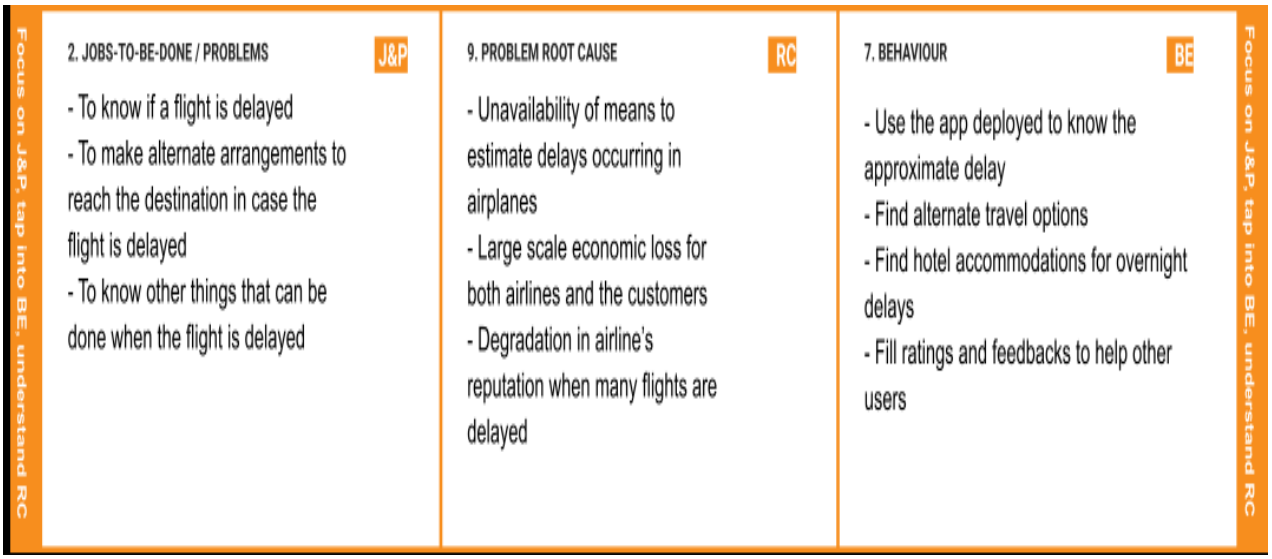
2.	Idea / Solution description	Building an application interface for customers(passengers and airlines) to know if a flight is delayed by implementing a machine learning based model to predict departure and arrival delays of an aircraft considering spatial, temporal and other dependencies causing the delay.
3.	Novelty / Uniqueness	The solution takes into account all possible reasons for delay(crew delys, weather, air traffic, aircraft type) to provide an accurate prediction. Apart from predicting arrival delays, departure delays are also predicted in order for the passengers to prepare accordingly and for the airline to make arrangements suitably.
4.	Social Impact / Customer Satisfaction	A lot of time and money can be saved for the customers and the loyalty and trust of customers towards the company increases. Improves airline operations by letting the company prepare in prior to adversaries (like crew illness, timeouts, rescheduling) leading to passenger

		satisfaction which will result positively on the economy and brand value.
5.	Business Model (Revenue Model)	Business to Consumer model The solution is a low-cost airline model planned to be created as an application with which the consumers can interact directly to know the details of their flight. It follows a non-monetary revenue model where the consumers aren't charged for what they get but are asked to provide their flight details and ratings which can be used to improve the model and shared with the airline in return for airline's flight data.
6.	Scalability of the Solution	The present solution is drafted with the aim of experimenting with airlines based out of the United States of America. If there is a possibility to acquire data of a broader region (say North America, other continents), then the solution can be developed to benefit a wider range of people. International flight dependencies in both temporal and spatial focus can be derived from that data to provide more accurate predictions.

		Presence of ADS-B data can further increase the efficiency of system making it reach global audience and live time tracking of flights.

3.4 PROBLEM SOLUTION FIT :





Identify strong TR & EM

3. TRIGGERS

TR

- Cancellation of flights
- Extreme boredom
- Guilt of wasting time
- Thought of missing important meetings
- Missing layover flight
- Uncertainty in deciding if the flight is delayed when they start late for the airport

10. OUR SOLUTION

SL

- The aim is to develop an applicatio predicts flight delays using a super machine learning model (a decisio classifier) with the data of flights and d so far and estimate the time of delay t spatial dependencies of flights into acc

8.CHANNELS of BEHAVIOUR

CH

8.1 ONLINE

- Check if a particular flight will be delayed and the estimated time of arrival
- Giving ratings and feedbacks for various flights so as to improve the app's performance in predicting further delays
- Check for other specific reasons for delay

Identify strong TR & EM

4 EFORE / AFTER

EM

Before:

- Worried
- About missing important events
- About missing layover flights
- If the flight is gonna be canceled
- Frustrated
- About the unexpected delay/cancellation
- Not knowing the news of delay beforehand
- About the weather
- Bored
- Don't know how to make use of time

After:

- Gets to enjoy the airline benefits
- Stay relaxed after getting a proper update from the airline
- Relieved if an alternate solution can be found

4. Advantages/Disadvantages

EM **Advantages:**

- Save Time
- Reduced Human Errors - Save Money

Disadvantages:

- Installation Cost is High
- Prediction cannot be completely Accurate
- If a external issue such as network issue arises the facility may not work properly

4. REQUIREMENT ANALYSIS :

FUNCTIONAL REQUIREMNETS :

S.NO	Functional Requirement	Description
1.	User Registration	Registration through Gmail
2.	User Confirmation	Confirmation via Email Confirmation via OTP
3.	User requirements	Collecting informations like date of travel, Departing & arrival destination, flight number or booking number.
4.	User friendliness	User Interface is simple and understandable.

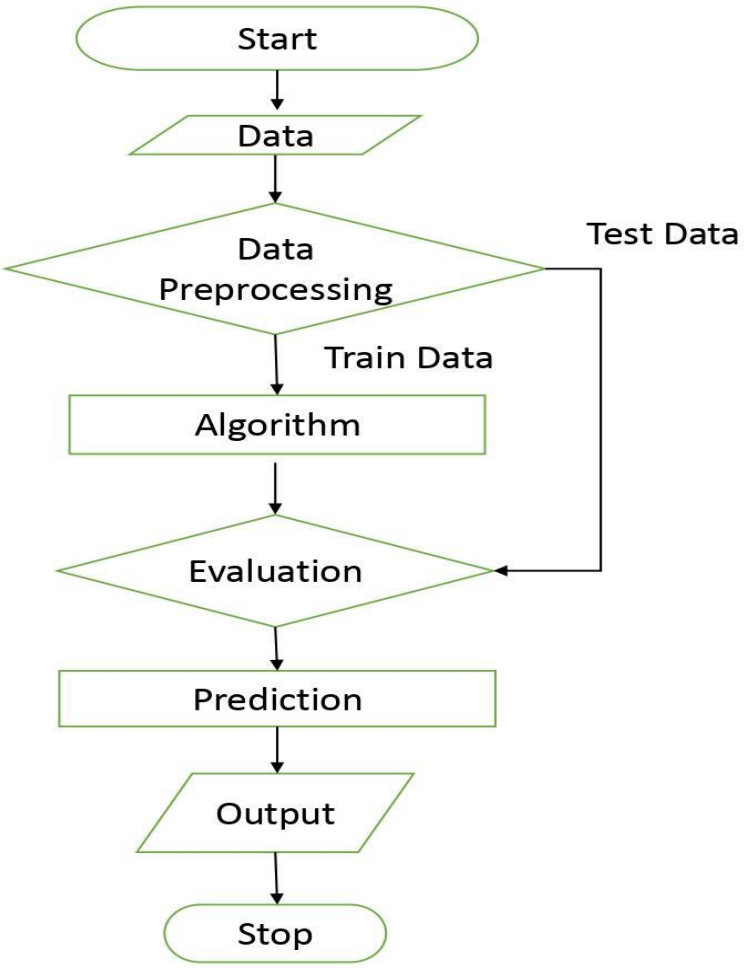
4.2 NON – FUNCTIONAL REQUIREMENTS :

S.NO	Non-Functional Requirement	
1.	Usability	Customer should finds it to be simple to interact.
2.	Security	Security's part will be protected against malware attacks or unauthorized access. But there's a catch. The lion's share of security non-functional requirements can be translated into concrete functional counterparts. If you want to protect the admin panel from unauthorized access, youwould define the login flow and different user roles as system behavior or user actions.
3.	Reliability	Reliability specifies how likely the system element would run without a failure for a given period of time under predefined conditions. Traditionally, this probability is expressed in percentages. For instance, if the system has 85 percent reliabilityfor a month, this means that during this month, under normal usage conditions, there's an 85 percent chance that the system won't experience critical failure.

4.	Performance	Performance defines how fast a software system or a particular piece of it responds to certain users' actions under a certain workload.In most cases, this metric explains how long a user must wait before the target operation happens (the page renders, a transaction is processed, etc.) given the overall number of users at the moment. But it's not always like that. Performance requirements may describe background processes invisible to users, e.g. backup. But let's focus on user-centric performance.
5.	Availability	Availability describes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, you may also define it as a percentage of time the system is accessible for operation during sometime period. For instance, the system may be available 98 percent of the time during a month. Availability is perhaps the most businesscritical requirement, but to define it,you also must have estimations for reliability and maintainability. Every request should be responded in time. Information which is provided should be accurate and should be updated according to the flight arrival and departure time.
6.	Scalability	Scalability assesses the highest workloads under which the system will still meet the performance requirements. There are two ways to enable your system scale as the workloads get higher horizontal and vertical scaling.

5. PROJECT DESIGN :

5.1 DATA FLOW DIAGRAM :



5.2 SOLUTION ARCHITECTURE :

ARCHITECTURAL WORKFLOW:

User view:

1. User enters flight details in the UI
2. Entered input is sent to the classifier model deployed through IBM Watson.
3. The model predicts the estimated time of departure/arrival delay and sendsit to the UI.
4. The predicted value is displayed to the UI

Model view:

1. The dataset is preprocessed for handling missing/categorical values.
2. Spatial and other features are extracted.
3. The features are split into training and test set.
4. Various algorithms are built and trained with the training data.
5. The model is evaluated using testing data.
6. The trained model is deployed in IBM Watson.

Template

Customer experience journey map

This map illustrates the complete journey of the customer from entry to the exit while using the flight delay prediction application

Created in partnership with

Product School

Share template feedback

Document an existing experience

Narrow your focus to a specific scenario or process within an existing product or service. In the **Steps** row, document the step-by-step process someone typically experiences, then add detail to each of the other rows.

SCENARIO: Navigating through the flight delay prediction application and finding the estimated delay	Entice How does someone initially become aware of this process?	Enter What do people experience as they begin the process?	Engage In the core moments in the process, what happens?	Exit What do people typically experience as the process finishes?	Extend What happens after the experience is over?
Steps What does the person (or group) typically experience?					
Interactions What interactions do they have at each step along the way? • **People:** Who do they see or talk to? • **Places:** Where are they? • **Things:** What digital touchpoints or physical objects would they use?					
Goals & motivations At each step, what is a person's primary goal or motivation? ("Help me…" or "Help me avoid…")					
Positive moments What steps does a typical person find enjoyable, productive, fun, motivating, delightful, or exciting?					
Negative moments What steps does a typical person find frustrating, confusing, angering, costly, or time-consuming?					
Areas of opportunity How might we make each step better? What ideas do we have? What have others suggested?					

6. PROJECT PLANNING AND SCHEDULING :

6.1 SPRINT DELIVERY PALN :

Product Backlog, Sprint Schedule, and Estimation

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points
Sprint-1	Data Collection	USN-1	Data is collected in the form of dataset for processing.	2
Sprint-1	Data Cleaning	USN-2	Duplicate values in the data is cleaned.	3
Sprint-1	Outlier Detection	USN-3	Outliers in the dataset are removed.	2

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Training	USN-4	Data is trained to build a model.	2	Medium	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-1	Data Prediction	USN-5	Result is predicted based on the trained data.	3	Medium	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-2	Model Building	USN-6	Model is built using several algorithms.	2	Medium	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-2	Model Training	USN-7	Model is trained .	2	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-2	Model Testing	USN-8	Model is tested.	2	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-2	Model Deployment	USN-9	Model is deployed in IBM Cloud.	2	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Login Page	USN-10	A login page is created .	3	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-3	Home Page	USN-11	A home page is created.	4	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-3	Results Page	USN-12	A separate page is created in which the information about the flights are displayed.	3	High	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-4	Application Building	USN-14	An application is built with a basic login page and home page.	2	Medium	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L
Sprint-4	Application Testing	USN-15	The developed application is tested.	3	Medium	SUNDARESVAR P, SUREN GOPAL D, SUNDARESAN M, THULASI VEERA RAGAVAN L

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	6 Days	24 Oct 2022	29 Oct 2022	15	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	15	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	15	07 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	15	14 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

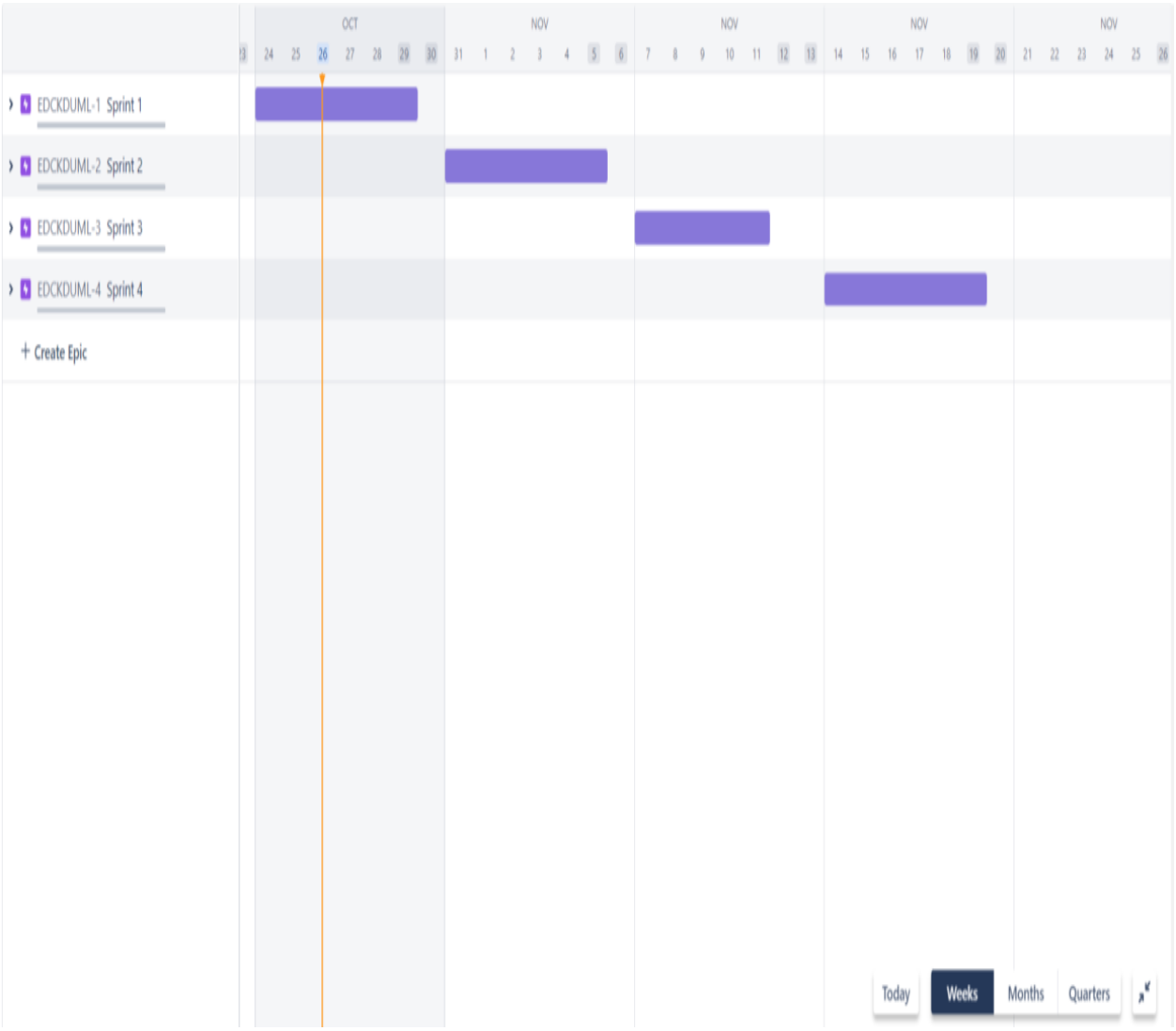
Sprint 1 AV = Sprint duration/velocity =

Sprint 2 AV = Sprint duration/velocity =

Sprint 3 AV = Sprint duration/velocity =

Sprint 4 AV = Sprint duration/velocity =

Burndown Chart:



7. CODING AND SOLUTIONING :

7.1 REQUIREMENTS AND FEATURES :

REQUIREMENTS :

Anaconda Navigator

Jupyter Notebook

Spyder

Anaconda Prompt

IBM Cloud

IBM Watson Studio

Object Cloud Storage

Watson Machine Learning

Node Red

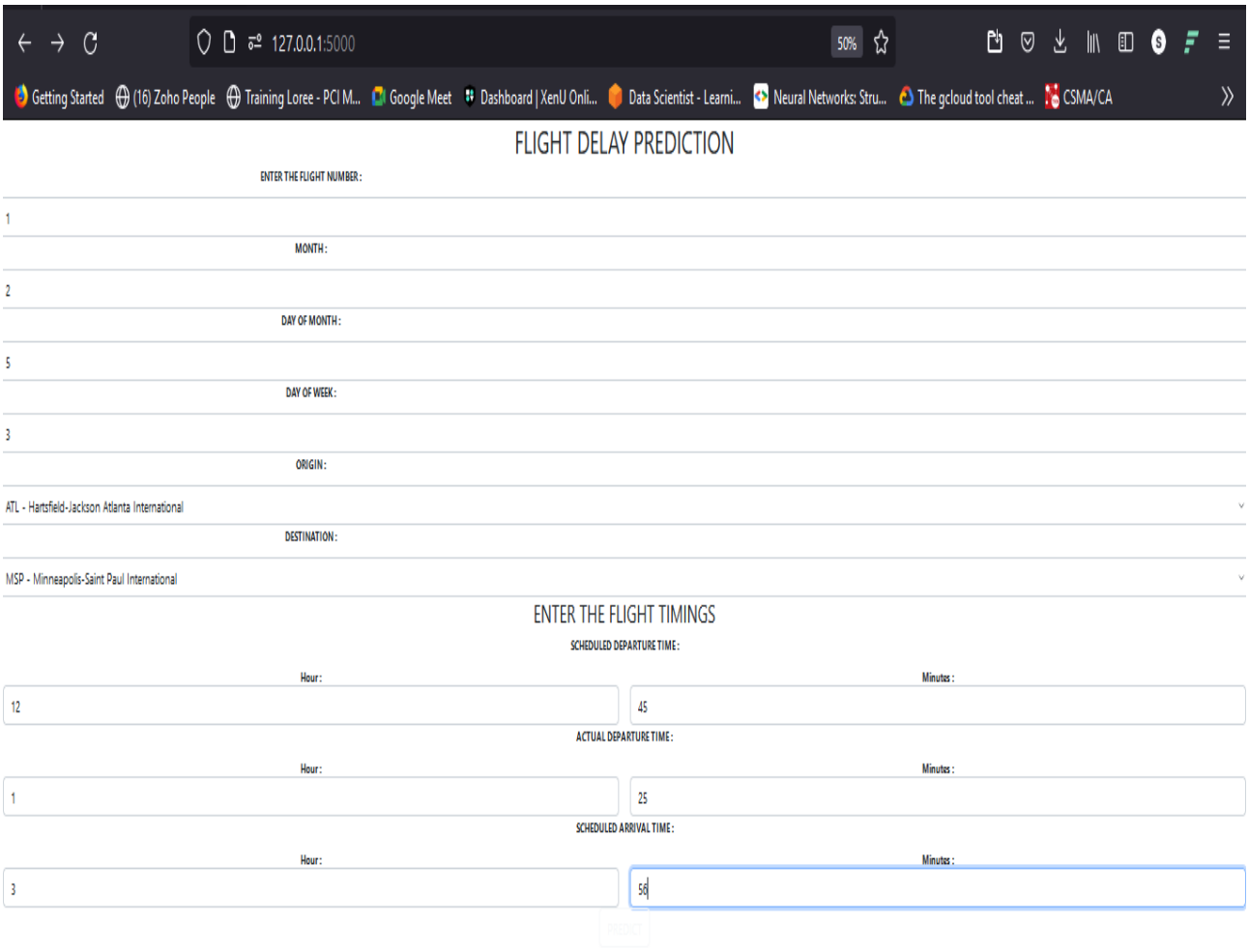
FEATURES :

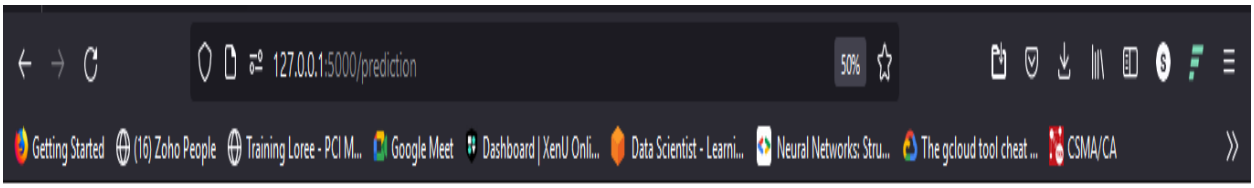
Index page

Results page

HTML and CSS codes are integrated with flask app coding

8. TESTING AND RESULTS :





THE FLIGHT WILL BE ON TIME

9.ADVANTAGES AND DISADVANTAGES :

ADVANTAGES :

• Machine learning can predict flight delays with a high degree of accuracy.

• Machine learning can help identify causes of flight delays.

• Machine learning can help reduce the number of flight delays.

• Machine learning can help improve the efficiency of airport operations.

DISADVANTAGES :

• Machine learning models can be complex and difficult to understand.

• Machine learning models require a large amount of data to train and can be time-consuming to develop.

• Machine learning models can be prone to overfitting, meaning they may not generalize well to new data.

• Machine learning models can be expensive to develop and maintain.

10.CONCLUSION :

In this project, we use flight data, weather, and demand data to predict flight departure delay. Our result shows that the Decision Tree method yields the best performance compared to the SVM model. Somehow the SVM model is very time consuming and does not necessarily produce better results. In the end, our model correctly predicts 91% of the non-delayed flights. However, the delayed flights are only correctly predicted 41% of time. As a result, there can be additional features related to the causes of flight delay that are not yet discovered using our existing data sources. In the second part of the project, we can see that it is possible to predict flight delay patterns from just the volume of concurrently published tweets, and their sentiment and objectivity. This is not unreasonable; people tend to post about airport delays on Twitter; it stands to reason that these posts would become more frequent, and more profoundly emotional, as the delays get worse. Without more data, we cannot make a robust model and find out the role of related factors and chance on these results. However, as a proof of concept, there is potential for these results. It may be possible to routinely use tweets to ascertain an understanding of concurrent airline delays and traffic patterns, which could be useful in a variety of circumstances.

FUTURE SCOPE :

This project is based on data analysis from year 2008. A large dataset is available from 1987-2008 but handling a bigger dataset requires a great amount of preprocessing and cleaning of the data. Therefore, the future work of this project includes incorporating a larger dataset. There are many different ways to preprocess a larger dataset like running a Spark cluster over a server or using a cloud-based services like AWS and Azure to process the data. With the new advancement in the field of deep learning, we can use Neural Networks algorithm on the flight and weather data. Neural Network works on the pattern matching methodology. It is divided into three basic parts for data modelling that includes feed forward networks, feedback networks, and selforganization network. Feed-forward and feedback networks are generally used in the areas of prediction, pattern recognition, associative memory, and optimization calculation, whereas self-organization networks are generally used in cluster analysis. Neural Network offers distributed computer architecture with important learning abilities to represent nonlinear relationships.

Also, the scope of this project is very much confined to flight and weather data of United States, but we can include more countries like China, India, and Russia. Expanding the scope of this project, we can also add the flight data from international flights and not just restrict our self to the domestic flights.

APPENDIX :

SOURCE CODE :

MODEL TRAINING :

```
import pandas as pd
import pickle
import joblib
import seaborn as sb
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.ensemble import VotingClassifier
```

```
dataset = pd.read_csv('flightdata.csv')
dataset.head()
```

```
dataset.columns
```

```
dataset.dtypes
```

```
# removing irrelevant data
dataset = dataset.drop(['Unnamed: 25', 'UNIQUE_CARRIER'], axis=1)
```

```
# find exact value counts of all fields
cat_cols = dataset.select_dtypes(include=object).columns.tolist()
(pd.DataFrame(
    dataset[cat_cols]
    .melt(var_name='column', value_name='value')
    .value_counts())
.rename(columns={0: 'counts'})
```

```
.sort_values(by=['column', 'counts']))
```

```
#plotting correlations (implication - almost linear relationship)
sb.jointplot(data=dataset, x="CRS_ARR_TIME", y="ARR_TIME")
```

```
# creating a copy of the dataset for visualization

dataset_visualization = dataset.copy()
```

```
dataset_visualization.dtypes
```

```
list_str_obj_cols = dataset_visualization.columns[dataset_visualization.dtypes ==
"object"].tolist()
for str_obj_col in list_str_obj_cols:
    dataset_visualization[str_obj_col] =
pd.to_numeric(dataset_visualization[str_obj_col], errors='coerce')
dataset_visualization.dtypes
```

```
correlations = dataset_visualization.corr()
sb.heatmap(correlations)
```

```
# to identify the correlations between arrival delay and other fields
correlations['ARR_DEL15']
```

```
# based on the correlation factors
refined_dataset = dataset.drop(['YEAR', 'QUARTER', 'DAY_OF_WEEK', 'TAIL_NUM',
'FL_NUM',
                                'ORIGIN_AIRPORT_ID', 'DEST_AIRPORT_ID',
'CRS_ELAPSED_TIME',
                                'ACTUAL_ELAPSED_TIME', 'DISTANCE', 'ORIGIN',
                                'DEST', 'ARR_TIME', 'ARR_DELAY'], axis=1)
refined_dataset
```

```
# check datatypes of columns
refined_dataset.dtypes
```

```
# finding missing values
refined_dataset.isna().sum()
```

```
# imputing missing values with mean
refined_dataset.dropna(inplace=True)
refined_dataset
```

```
refined_dataset.shape
```

```
# converting float values to categorical
columns = ['DEP_DEL15', 'CANCELLED', 'DIVERTED', 'ARR_DEL15']
for col in columns:
    refined_dataset[col] = refined_dataset[col].astype('int').astype('category')
refined_dataset.dtypes
```

```
# converting float values to categorical
columns = ['DEP_TIME', 'DEP_DELAY']
for col in columns:
    refined_dataset[col] = refined_dataset[col].astype('int')
refined_dataset.dtypes
```

```
# split dependent and independent variables
X = refined_dataset[['MONTH', 'DAY_OF_MONTH', 'CRS_DEP_TIME', 'DEP_TIME',
'DEP_DELAY',
                     'DEP_DEL15', 'CRS_ARR_TIME', 'CANCELLED', 'DIVERTED']]
Y = refined_dataset[['ARR_DEL15']]
```

```
# splitting into train and test data
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33,
random_state=42)
y_train.value_counts()
```

```
# decision tree classifier
decision_tree_classifier = DecisionTreeClassifier()
decision_tree_classifier = decision_tree_classifier.fit(X_train,y_train)
decision_tree_prediction = decision_tree_classifier.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(decision_tree_prediction, y_test))
print("Classification report\n", classification_report(decision_tree_prediction,
y_test))
print("Accuracy score\n", accuracy_score(decision_tree_prediction, y_test))
```

```
# svc model
SVC_model = SVC()
SVC_model.fit(X_train, y_train)
SVC_prediction = SVC_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(SVC_prediction, y_test))
print("Classification report\n", classification_report(SVC_prediction, y_test))
print("Accuracy score\n", accuracy_score(SVC_prediction, y_test))
```

```
# knn model
KNN_model = KNeighborsClassifier(n_neighbors=5)
KNN_model.fit(X_train, y_train)
KNN_prediction = KNN_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(KNN_prediction, y_test))
print("Classification report\n", classification_report(KNN_prediction, y_test))
print("Accuracy score\n", accuracy_score(KNN_prediction, y_test))
```

```
# gaussian naive bayes model
GNB_model = GaussianNB()
GNB_model.fit(X_train, y_train)
GNB_prediction = GNB_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(GNB_prediction, y_test))
print("Classification report\n", classification_report(GNB_prediction, y_test))
print("Accuracy score\n", accuracy_score(GNB_prediction, y_test))
```

```
# ensemble model of best 3 peforming model - gnb, knn, svc
ensemble = VotingClassifier(estimators=[('gnb', GNB_model), ('knn', KNN_model),
('svc', SVC_model)], voting='hard')
```

```
ensemble.fit(X_train, y_train)
ensemble_prediction = ensemble.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(ensemble_prediction, y_test))
print("Classification report\n", classification_report(ensemble_prediction,
y_test))
print("Accuracy score\n", accuracy_score(ensemble_prediction, y_test))
```

```
joblib.dump(ensemble, 'flight.pkl')
```

MODEL DEPLOYMENT IN IBM :

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='9sZqBJtG4-b-g5lWT82rc0E7XBL1FW5oIXCYWiqWXGnC',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'flightdelayprediction-donotdelete-pr-clt0bq9r7uf8bs'
object_key = 'flightdata.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__,
body )

dataset = pd.read_csv(body)
dataset.head()
```

```
dataset.head(10)
```

```
dataset.columns
dataset.dtypes
```

```
dataset.describe()
```

```
dataset.info()
```

```
# removing irrelevant data
dataset = dataset.drop(['Unnamed: 25', 'UNIQUE_CARRIER'], axis=1)
```

```
# find exact value counts of all fields
cat_cols = dataset.select_dtypes(include=object).columns.tolist()
(pd.DataFrame(
    dataset[cat_cols]
    .melt(var_name='column', value_name='value')
    .value_counts())
.rename(columns={0: 'counts'})
.sort_values(by=['column', 'counts']))
```

```
#plotting correlations (implication - almost linear relationship)
import seaborn as sb
sb.jointplot(data=dataset, x="CRS_ARR_TIME", y="ARR_TIME")
```

```
# creating a copy of the dataset for visualization

dataset_visualization = dataset.copy()
```

```
dataset_visualization.dtypes
```

```
list_str_obj_cols = dataset_visualization.columns[dataset_visualization.dtypes ==
"object"].tolist()
for str_obj_col in list_str_obj_cols:
    dataset_visualization[str_obj_col] =
pd.to_numeric(dataset_visualization[str_obj_col], errors='coerce')
dataset_visualization.dtypes
```

```
correlations = dataset_visualization.corr()
sb.heatmap(correlations)
```

```
# to identify the correlations between arrival delay and other fields
correlations['ARR_DEL15']
```

```
# based on the correlation factors
refined_dataset = dataset.drop(['YEAR', 'QUARTER', 'DAY_OF_WEEK', 'TAIL_NUM',
'FL_NUM',
                                'ORIGIN_AIRPORT_ID', 'DEST_AIRPORT_ID',
'CRS_ELAPSED_TIME',
                                'ACTUAL_ELAPSED_TIME', 'DISTANCE', 'ORIGIN',
                                'DEST', 'ARR_TIME', 'ARR_DELAY'], axis = 1)
```

```
# check datatypes of columns
refined_dataset
```

```
# finding missing values
refined_dataset.isna().sum()
```

```
# imputing missing values with mean
refined_dataset.dropna(inplace = True)
refined_dataset
```

```
refined_dataset.shape
```

```
# converting float values to categorical
columns = ['DEP_DEL15', 'CANCELLED', 'DIVERTED', 'ARR_DEL15']
for col in columns:
    refined_dataset[col] = refined_dataset[col].astype('int').astype('category')
refined_dataset.dtypes
```

```
# converting float values to categorical
columns = ['DEP_TIME', 'DEP_DELAY']
for col in columns:
    refined_dataset[col] = refined_dataset[col].astype('int')
refined_dataset.dtypes
```

```
# split dependent and independent variables
X = refined_dataset[['MONTH', 'DAY_OF_MONTH', 'CRS_DEP_TIME', 'DEP_TIME',
'DEP_DELAY',
                     'DEP_DEL15', 'CRS_ARR_TIME', 'CANCELLED', 'DIVERTED']]
Y = refined_dataset[['ARR_DEL15']]
```

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.ensemble import VotingClassifier
```

```
# splitting into train and test data
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.33,
random_state=42)
y_train.value_counts()
```

```
# decision tree classifier
decision_tree_classifier = DecisionTreeClassifier()
decision_tree_classifier = decision_tree_classifier.fit(X_train,y_train)
decision_tree_prediction = decision_tree_classifier.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(decision_tree_prediction, y_test))
print("Classification report\n", classification_report(decision_tree_prediction,
y_test))
print("Accuracy score\n", accuracy_score(decision_tree_prediction, y_test))
```

```
# svc model
SVC_model = SVC()
SVC_model.fit(X_train, y_train)
SVC_prediction = SVC_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(SVC_prediction, y_test))
print("Classification report\n", classification_report(SVC_prediction, y_test))
print("Accuracy score\n", accuracy_score(SVC_prediction, y_test))
```

```
# knn model
KNN_model = KNeighborsClassifier(n_neighbors=5)
KNN_model.fit(X_train, y_train)
KNN_prediction = KNN_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(KNN_prediction, y_test))
print("Classification report\n", classification_report(KNN_prediction, y_test))
print("Accuracy score\n", accuracy_score(KNN_prediction, y_test))
```

```
# gaussian naive bayes model
GNB_model = GaussianNB()
GNB_model.fit(X_train, y_train)
GNB_prediction = GNB_model.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(GNB_prediction, y_test))
print("Classification report\n", classification_report(GNB_prediction, y_test))
print("Accuracy score\n", accuracy_score(GNB_prediction, y_test))
```

```
# ensemble model of best 3 peforming model - gnb, knn, svc
ensemble = VotingClassifier(estimators=[('gnb', GNB_model), ('knn', KNN_model),
('svc', SVC_model)], voting='hard')
ensemble.fit(X_train, y_train)
ensemble_prediction = ensemble.predict(X_test)
# performance metrics
print("Confusion matrix\n", confusion_matrix(ensemble_prediction, y_test))
print("Classification report\n", classification_report(ensemble_prediction,
y_test))
print("Accuracy score\n", accuracy_score(ensemble_prediction, y_test))
```

```
import pickle
import joblib
```

```
joblib.dump(ensemble, 'flight.pkl')
```

```
!pip install ibm_watson_machine_learning
```

```
from ibm_watson_machine_learning import APIClient
import json
wml_credentials = {
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"XSDHlDlMa-kb8AfvLH_4X8c2iFCyp_JBnp5ZgwU_7Rfn"
}
```

```
wml_client = APIClient(wml_credentials)
wml_client.spaces.list()
```

```
SPACE_ID = "109e03e3-31b6-439b-95da-02a7bbf53e4b"
wml_client.set.default_space(SPACE_ID)
wml_client.software_specifications.list(500)
```

```
import sklearn
sklearn.__version__
MODEL_NAME = 'model'
DEPLOYMENT_NAME = 'model'
DEMO_MODEL = ensemble
```

```
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-
22.1-py3.9')
```

```
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

```
model_details = wml_client.repository.store_model(
    model=DEMO_MODEL,
    meta_props=model_props,
    training_data=X_train,
    training_target=y_train
)
```

```
model_id = wml_client.repository.get_model_id(model_details)
model_id
```

```
deployment_props = {
    wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

```
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)
```

```
#######################################################################################
###### Synchronous deployment creation for uid: 'e174c368-31ba-4bbe-a17d-
cc0e08fe7c80' started
#######################################################################################
###### initializing Note: online_url is deprecated and will be removed in a
future release. Use serving_urls instead. ready ---------------------------------
------------------------------------------------------------- Successfully
finished deployment creation, deployment_uid='28b6245d-29ab-4f08-9a5e-
e6f975945288' ---------------------------------------------------------------------
-----------------------------
```

HTML CODES :

INDEX PAGE :

```
<html>
    <head>
        <!-- CSS only -->
            <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
            <link rel="stylesheet"
href="{{url_for('static',filename='css/main.css')}}"
        </head>
    <body>
        <center>
            <h1>FLIGHT DELAY PREDICTION</h1>


            <div id="form">
            <form class="form-group" action="/prediction" method="post" >
                <div class="row">
                    <div class="col-md-12">
                <div class="row ">
                        <p class="fw-bolder  col-6">ENTER THE FLIGHT NUMBER :
</p>
                        <input class="form-control-lg form-control-inline form-
control " type="text" name="name" />
                </div>

                <div class="row input-feilds">
                    <p class="fw-bolder col-6">MONTH : </p>
                    <input class="form-control-lg form-control-inline form-
control" type="text" name="month" />
                </div>

                <div class="row input-feilds">
                    <p class="fw-bolder col-6">DAY OF MONTH : </p>
                    <input class="form-control-lg form-control-inline form-
control" type="text" name="dayofmonth" />
                </div>

                <div class="row input-feilds">
```

```
                    <p class="fw-bolder col-6">DAY OF WEEK : </p>
                    <input class="form-control-lg form-control-inline form-
control" class="form-control-inline form-control" type="text" name="dayofweek" />
                </div>

                <div class="row input-feilds">
                <p class="fw-bolder col-6">ORIGIN : </p>
                <select class="form-select form-select-lg form-control-inline "
name="origin">
                    <option value="alt">ATL - Hartsfield-Jackson Atlanta
International</option>
                    <option value="dtw">DTW - Detroit Metropolitan Wayne County
</option>
                    <option value="sea">SEA - Seattle-Tacoma
International</option>
                    <option value="msp">MSP - Minneapolis-Saint Paul
International</option>
                    <option value="jfk">JFK - John F. Kennedy
International</option>
                </select>
                </div>
                <div class="row input-feilds">
                <p class="fw-bolder col-6">DESTINATION : </p>
                <select class="form-select form-select-lg form-control-inline"
name="destination">
                    <option value="alt">ATL - Hartsfield-Jackson Atlanta
International</option>
                    <option value="dtw">DTW - Detroit Metropolitan Wayne County
</option>
                    <option value="sea">SEA - Seattle-Tacoma
International</option>
                    <option value="msp">MSP - Minneapolis-Saint Paul
International</option>
                    <option value="jfk">JFK - John F. Kennedy
International</option>
                </select>
            </div>
            <h2 class="flight-time">ENTER THE FLIGHT TIMINGS</h2>
            <div class="row time input-feilds " >
                <p class="fw-bolder sdt">SCHEDULED DEPARTURE TIME : </p>
                <div class="col-sm-6 time2">
                <label class="fw-bolder">Hour : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="depthr" />
                </div>
```

```
                <div class="col-sm-6">
                <label class="fw-bolder">Minutes : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="deptmin" />
                </div>
            </div>
            <div class="row time">

                <p class="fw-bolder">ACTUAL DEPARTURE TIME : </p>
                <div class="col-sm-6 time2">
                <label class="fw-bolder">Hour : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="actdepthr" />
                </div>
                <div class="col-sm-6 ">
                <label class="fw-bolder">Minutes : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="actdeptmin" />
                </div>
            </div>

            <div class="row time">
                <p class="fw-bolder">SCHEDULED ARRIVAL TIME : </p>
                <div class="col-sm-6 time2">
                <label class="fw-bolder">Hour : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="arrtimehr" />
                </div>
                <div class="col-sm-6">
                <label class="fw-bolder">Minutes : </label>
                <input placeholder="00" class="form-control-lg form-control-
inline form-control" type="text" name="arrtimemin" />
                </div>
            </div>
                <button type="submit" class="input-feilds btn btn-outline-light
btn-lg">PREDICT</button>
            </div>
            </div>
            </form>
            <p>{{y}}</p>
        </div>
        </center>

    </body>
</html>
```

RESULTS PAGE :

```
<html>
    <head>
        <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
        <link rel="stylesheet"
href="{{url_for('static',filename='css/main.css')}}">
    </head>
    <body>
        <center>
        <div id="form" class="result">
            <h1 class="result-text">{{y}}</h1>
        </div>
    </center>
    </body>
</html>
```

CSS PAGE :

MAIN PAGE :

```
body{

    background-repeat:no-repeat ;
    background-size: cover;
    background-position: center;
}

h1{
    color:white;
    font-size: 70px;
    margin-top: 30px;
    margin-bottom: 30px;
}

p{
    color:white;
    font-size: 25px;
```

```css
}

#form{
    border:5px outset white ;
    border-bottom: 3px solid white;
    width:fit-content;

    background-repeat:no-repeat ;
    background-size: cover;
    background-position: center;
    margin-bottom: 30px;
    padding: 20px;
}

label{
    color:white
}

.time{
    margin: 10px;
}

.time2{
    margin-bottom: 4px;
}

.form-control-inline{
    min-width: 0;
    width: auto;
    display: inline;
}

.input-feilds{
    margin-top: 30px;
    padding:6px
}

label{
    font-size: 20px;

}

.result{
    margin-top: 250px;
}
```

```css
.result-text{
    font-size: 120px;
    color:azure
}

.flight-time{

    color:white;
    font-size: 50px;
    margin-top: 30px;
    margin-bottom: 40px;
}
```

FLASK APP CODE :

```python
from flask import Flask,render_template,request
import pickle


model=pickle.load(open('flightclf.pkl','rb'))


app=Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/prediction',methods=["POST"])
def predict():
    if request.method=="POST":
        name=request.form["name"]
        month=request.form["month"]
        if(int(month)>12):
            ans="Please Enter the correct Month"
            return render_template("index.html" ,y=ans)

        dayofmonth=request.form["dayofmonth"]
        if(int(dayofmonth)>31):
            ans="Please Enter the correct Day of Month"
            return render_template("index.html" ,y=ans)

        dayofweek=request.form["dayofweek"]
```

```
        if(int(dayofweek)>7):
            ans="Please Enter the correct Day of Week"
            return render_template("index.html" ,y=ans)


        origin=request.form["origin"]
        destination=request.form['destination']

        if(origin==destination):
            ans="Origin airport and destination airport can't be same"
            return render_template("index.html" ,y=ans)

        if(origin=="msp"):
            origin1,origin2,origin3,origin4,origin5=0,0,0,1,0
        if(origin=="dtw"):
            origin1,origin2,origin3,origin4,origin5=0,1,0,0,0
        if(origin=="jfk"):
            origin1,origin2,origin3,origin4,origin5=0,0,1,0,0
        if(origin=="sea"):
            origin1,origin2,origin3,origin4,origin5=0,0,0,0,1
        if(origin=="alt"):
            origin1,origin2,origin3,origin4,origin5=1,0,0,0,0



        if(destination=="msp"):
            destination1,destination2,destination3,destination4,destination5=0,0,
0,1,0
        if(destination=="dtw"):
            destination1,destination2,destination3,destination4,destination5=0,1,
0,0,0
        if(destination=="jfk"):
            destination1,destination2,destination3,destination4,destination5=0,0,
1,0,0
        if(destination=="sea"):
            destination1,destination2,destination3,destination4,destination5=0,0,
0,0,1
        if(destination=="alt"):
           destination1,destination2,destination3,destination4,destination5=1,0,0
,0,0


        depthr=request.form['depthr']
        deptmin=request.form['deptmin']
        if(int(depthr)>23 or int(deptmin)>59):
            ans="Please enter the correct Departure time"
```

```
            return render_template("index.html" ,y=ans)
        else:
            dept=depthr+deptmin

        actdepthr=request.form['actdepthr']
        actdeptmin=request.form['actdeptmin']
        if(int(actdepthr)>23 or int(actdeptmin)>59):
            ans="Please enter the correct Actual Departure time"
            return render_template("index.html" ,y=ans)
        else:
            actdept=actdepthr+actdeptmin



        arrtimehr=request.form['arrtimehr']
        arrtimemin=request.form['arrtimemin']
        if(int(arrtimehr)>23 or int(arrtimemin)>59):
            ans="Please enter the correct Arrival time"
            return render_template("index.html" ,y=ans)
        else:
            arrtime=arrtimehr+arrtimemin


        if((int(actdept)-int(dept))<15):
            dept15=0
        else:
            dept15=1

        print(dept15)
        total=[[month,dayofmonth,dayofweek,origin1,origin2,origin3,origin4,origin
5,destination1,destination2,destination3,destination4,destination5,dept,actdept,d
ept15,arrtime]]

        value=model.predict(total)
        print(value)
        if(value==[0.]):
            ans="THE FLIGHT WILL BE ON TIME"
        else:
            ans="THE FLIGHT WILL BE DELAYED"

    return render_template("results.html" ,y=ans)


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

FLASK INTEGRATION IN IBM :

```
from flask import Flask,render_template,request
import requests
# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.


# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = "XSDHlDlMa-kb8AfvLH_4X8c2iFCyp_JBnp5ZgwU_7Rfn"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}

import pickle
model=pickle.load(open('flightclf.pkl','rb'))


app=Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/prediction',methods=["POST"])
def predict():
    if request.method=="POST":
        name=request.form["name"]
        month=request.form["month"]
        if(int(month)>12):
            ans="Please Enter the correct Month"
            return render_template("index.html" ,y=ans)

        dayofmonth=request.form["dayofmonth"]
        if(int(dayofmonth)>31):
            ans="Please Enter the correct Day of Month"
            return render_template("index.html" ,y=ans)

        dayofweek=request.form["dayofweek"]
        if(int(dayofweek)>7):
```

```
            ans="Please Enter the correct Day of Week"
            return render_template("index.html" ,y=ans)


        origin=request.form["origin"]
        destination=request.form['destination']

        if(origin==destination):
            ans="Origin airport and destination airport can't be same"
            return render_template("index.html" ,y=ans)

        if(origin=="msp"):
            origin1,origin2,origin3,origin4,origin5=0,0,0,1,0
        if(origin=="dtw"):
            origin1,origin2,origin3,origin4,origin5=0,1,0,0,0
        if(origin=="jfk"):
            origin1,origin2,origin3,origin4,origin5=0,0,1,0,0
        if(origin=="sea"):
            origin1,origin2,origin3,origin4,origin5=0,0,0,0,1
        if(origin=="alt"):
            origin1,origin2,origin3,origin4,origin5=1,0,0,0,0



        if(destination=="msp"):
            destination1,destination2,destination3,destination4,destination5=0,0,
0,1,0
        if(destination=="dtw"):
            destination1,destination2,destination3,destination4,destination5=0,1,
0,0,0
        if(destination=="jfk"):
            destination1,destination2,destination3,destination4,destination5=0,0,
1,0,0
        if(destination=="sea"):
            destination1,destination2,destination3,destination4,destination5=0,0,
0,0,1
        if(destination=="alt"):
            destination1,destination2,destination3,destination4,destination5=1,0,
0,0,0

        depthr=request.form['depthr']
        deptmin=request.form['deptmin']
        if(int(depthr)>23 or int(deptmin)>59):
            ans="Please enter the correct Departure time"
            return render_template("index.html" ,y=ans)
```

```
        else:
            dept=depthr+deptmin

        actdepthr=request.form['actdepthr']
        actdeptmin=request.form['actdeptmin']
        if(int(actdepthr)>23 or int(actdeptmin)>59):
            ans="Please enter the correct Actual Departure time"
            return render_template("index.html" ,y=ans)
        else:
            actdept=actdepthr+actdeptmin



        arrtimehr=request.form['arrtimehr']
        arrtimemin=request.form['arrtimemin']
        if(int(arrtimehr)>23 or int(arrtimemin)>59):
            ans="Please enter the correct Arrival time"
            return render_template("index.html" ,y=ans)
        else:
            arrtime=arrtimehr+arrtimemin


        if((int(actdept)-int(dept))<15):
            dept15=0
        else:
            dept15=1

        print(dept15)
        total=[[month,dayofmonth,dayofweek,origin1,origin2,origin3,origin4,origin
5,destination1,destination2,destination3,destination4,destination5,dept,actdept,d
ept15,arrtime]]

        # NOTE: manually define and pass the array(s) of values to be scored in
the next line
        payload_scoring = {"input_data": [{"fields":
["f0","f1","f2","f3","f4","f5","f6","f7","f8","f9","f10","f11","f12","f13","f14",
"f15","f16"], "values": total}]}

        response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/28b6245d-29ab-4f08-9a5e-
e6f975945288/predictions?version=2022-11-21', json=payload_scoring,
        headers={'Authorization': 'Bearer ' + mltoken})

        print("Scoring response")
        print(response_scoring.json())
```

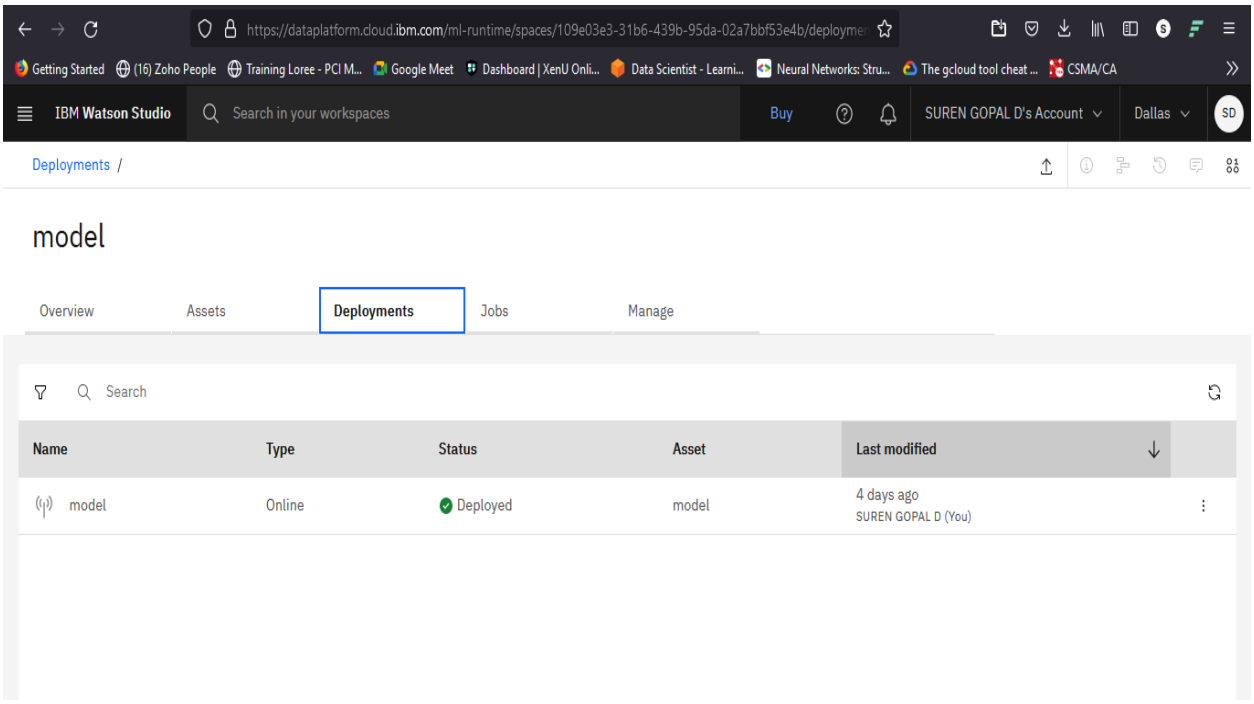
```
        pred = response_scoring.json()
        value = pred['predictions'][0]['values'][0][0]

        print(value)
        if(value==[0.]):
            ans="THE FLIGHT WILL BE ON TIME"
        else:
            ans="THE FLIGHT WILL BE DELAYED"

    return render_template("results.html" ,y=ans)


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

MODEL DEPLOYMENT STATUS :



GITHUB REPOSITORY LINK :

https://github.com/IBM-EPBL/IBM-Project-13366-1659517326

MODEL ID :

'e174c368-31ba-4bbe-a17d-cc0e08fe7c80'

DEPLOYMENT UID :

'28b6245d-29ab-4f08-9a5e-e6f975945288'

DEMO VIDEO LINK :

https://drive.google.com/file/d/1WUmVwsbei6PjZQ5j11el9ObfSW7zffRx/view?usp=share_link