1 INTRODUCTION

1.1 Overview

Delay is one of the most remembered performance indicators of any transportation system. Notably, commercial aviation players understand delay as the period by which a flight is late or postponed. Thus, a delay may be represented by the difference between scheduled and real times of departure or arrival of a plane. Country regulator authorities have a multitude of indicators related to tolerance thresholds for flight delays. Indeed, flight delay is an essential subject in the context of air transportation systems. In 2013, 36% of flights delayed by more than five minutes in Europe, 31.1% of flights delayed by more than 15 minutes in the United States. This indicates how relevant this indicator is and how it affects no matter the scale of airline meshes. To better understand the entire flight ecosystems, vast volumes of data from commercial aviation is collected every moment and stored in databases. In this context, the procedure of comprehending the domain, managing data and applying a model is known as Data Science, a trend in solving challenging problems related to Big Data. In this project, we've performed an extensive data analysis in order to extract the important attributes/factors that are responsible for the delay of flight.

1.2 Purpose

The project aims to predict the flight status by implementing different Machine Learning techniques such as KNN, Decision Tree and Random Forest. From these models the Best performing model is selected and saved. Here we will be building a flask application that uses a machine learning model to get the prediction of heart stroke. We will also train our model on IBM Cloud and deployment on IBM Cloud

2 LITERATURE SURVEY

2.1 Existing Problem

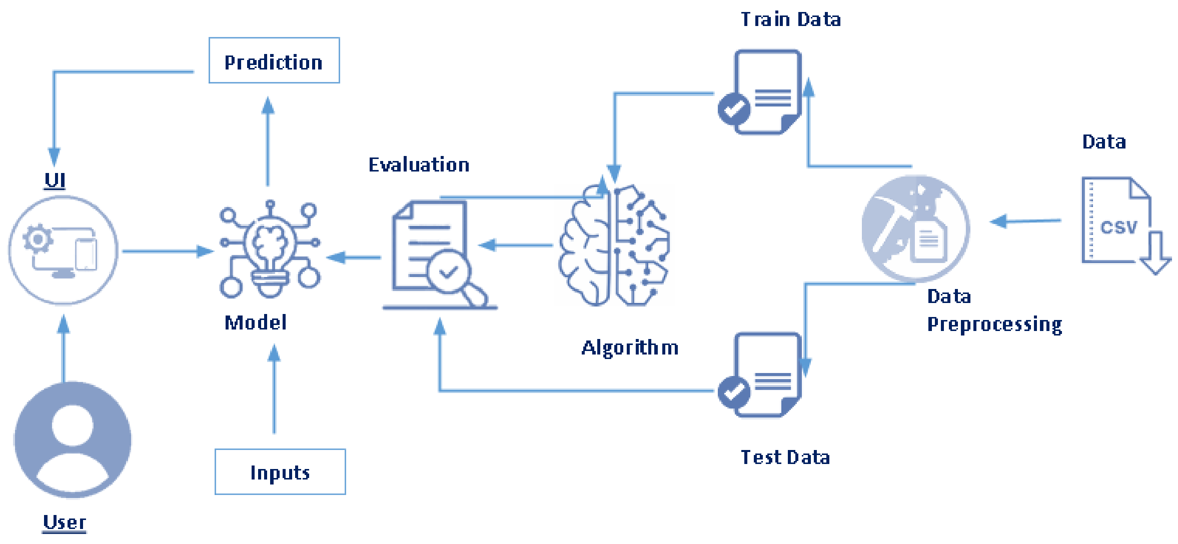
Over the last twenty years, air travel has been increasingly preferred among travellers, mainly because of its speed and in some cases comfort. This has led to phenomenal growth in air traffic and on the ground. An increase in air traffic growth has also resulted in massive levels of aircraft delays on the ground and in the air. These delays are responsible for large economic and environmental losses. The main objective of the model is to predict flight delays accurately in order to optimize flight operations and minimize delays. Using a machine learning model, we can predict flight arrival delays. The input to our algorithm is rows of feature vector like departure date, departure delay, distance between the two airports, scheduled arrival time etc. We then use decision tree classifier to predict if the flight arrival will be delayed or not. A flight is considered to be delayed when difference between scheduled and actual arrival times is greater than 15 minutes. Furthermore, we compare decision tree classifier with logistic regression and a simple neural network for various figures of merit.

2.2 Proposed Solution

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 air trafic. To make the system more scalable it is necessary to choose an algorithm which considers all the parameters to be independent. Supervised learning as the name indicates a presence of supervisor as teacher. Essentially supervised learning could be a learning that within which we tend to teach or train the machine exploitation data which is well tagged which means some data is already labelled with correct answer. After that, machine is given new set of examples(data) so supervised learning algorithmic rule analyses the coaching knowledge (set of training examples) and produces a correct outcome from tagged data Using supervised machine learning approach, the labelled data gives it authenticity. NaÃ¯ve bayed model is one of the algorithms which is proven to be efficient for real time prediction as well as the fact that it considers every attribute to be independent from each other makes it an apt algorithm for the concerned project

3 THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software Designing

- **Tools**

  For application development, the following Software Requirements are:

  1. **Operating System:** Windows 7 and above
  2. **Language**: Python, Html
  3. **Tools**: Microsoft Excel (Optional).
  4. **Technologies used**: Flask application

- **Software Requirements**
  1. **Operating System**: Any OS with clients to access the internet
  2. **Network**: Wi-Fi Internet or Cellular Network
  3. **Visual Studio**: Create and design Data Flow and Block Diagaram
  4. **GitHub**: Versioning Control
  5. **Google Chrome**: Medium to display and run the html code

- Hardware Requirements
  1. **Processor:** Intel or AMD
  2. **Ram:** 4Gb
  3. **Space on disk**: minimum 10GB

4. **For running the application**: Device: Any device that can access the internet.
5. Minimum space to execute: 20 MB

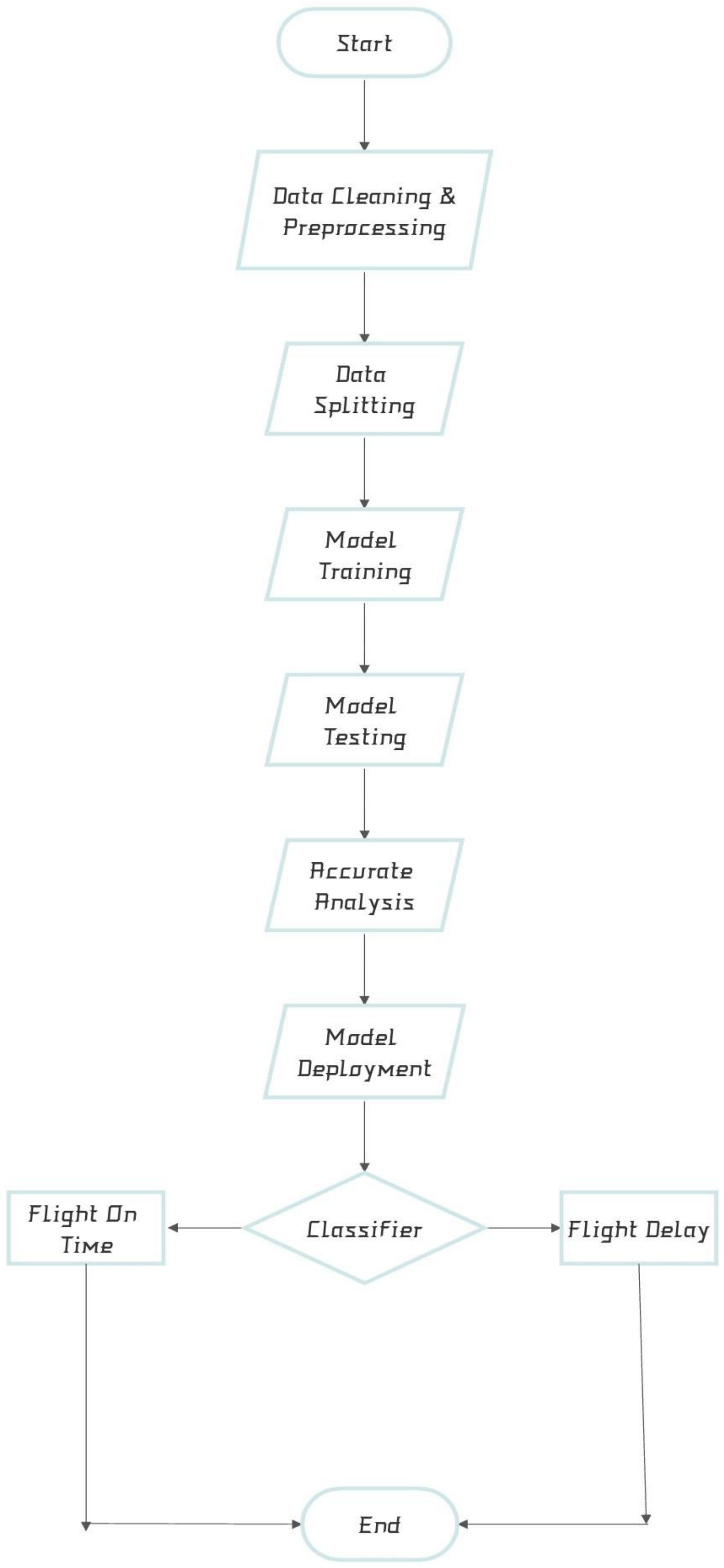
4 EXPIREMENTAL INVESTIGATIONS

In our research, we used a web technique to get all information about flights from official airport websites, and besides that, the weather condition has been collected too. All experiments were conducted under the same conditions. The same training dataset has been used for all algorithms. To avoid that some algorithms that cannot deal with the categorical data, the integer encoding has been used for training and new data. In the training of each model process, the grid search has been implemented. The most common parameters have been analyzed, and the bounds and the best-founded parameters have been presented.

The proposed solution is based on the factors of a flight number, month ,day of month, day of week, orgin, destination, scheduled departure time, actual departure time which classifies that particular flight into a specific category which may be either one of the following :

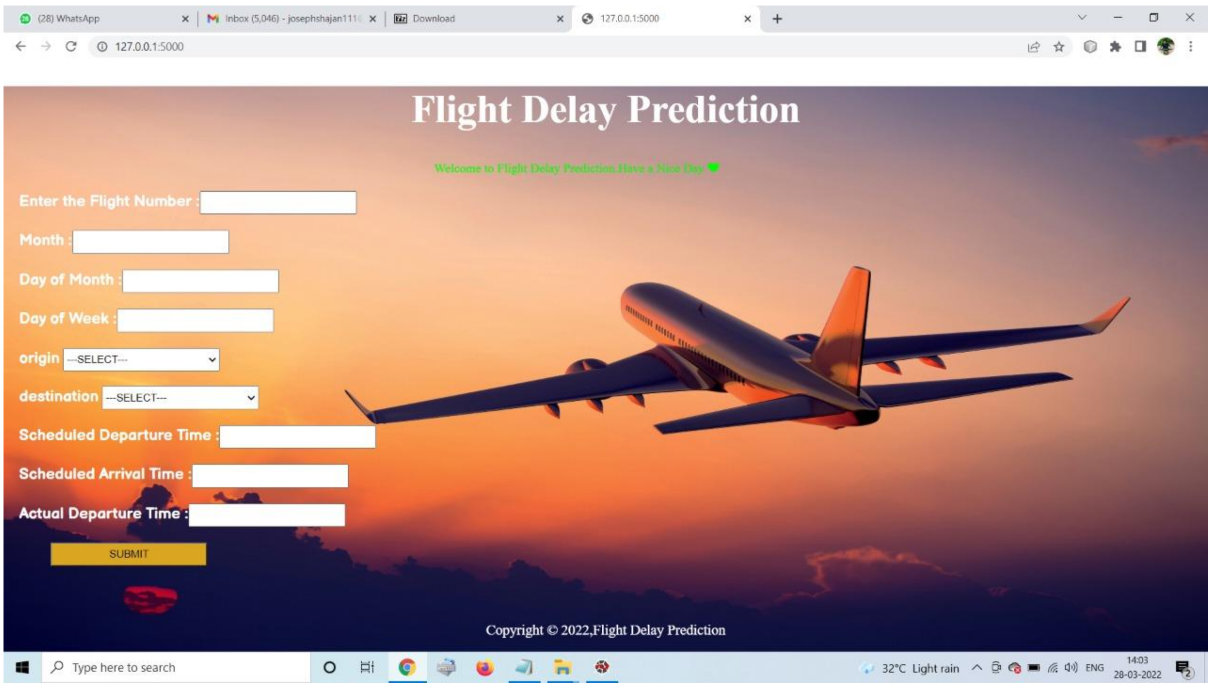
1. Flight on time
2. Flight might be delay.

5 FLOW CHART


Start
Data Cleaning & Preprocessing
Data Splitting
Model Training
Model Testing
Accurate Analysis
Model Deployment
Classifier
Flight On Time
Flight Delay
End

6 RESULT

6.1 Input Page



Flight Delay Prediction

Welcome to Flight Delay Prediction Have a Nice Day

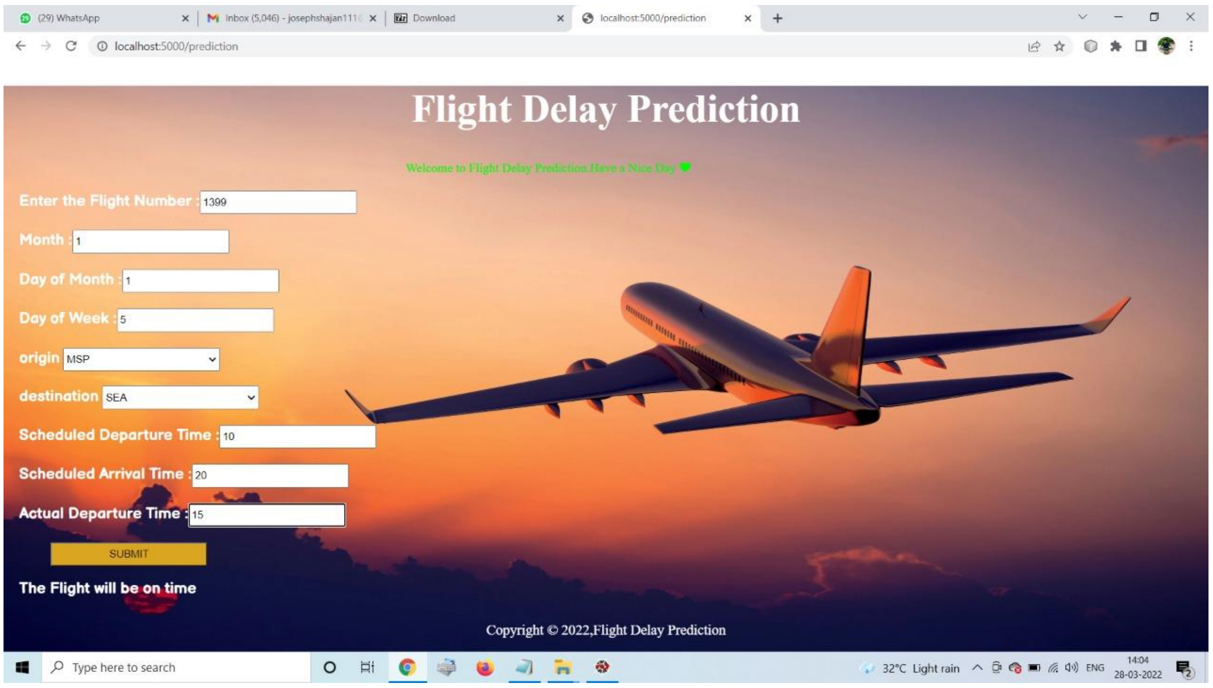
Enter the Flight Number :
Month :
Day of Month :
Day of Week :
origin ---SELECT---
destination ---SELECT---
Scheduled Departure Time :
Scheduled Arrival Time :
Actual Departure Time :

SUBMIT

Copyright © 2022,Flight Delay Prediction

All the 10 fields as shown in above picture are independent variables used to predict the flight delay.

6.2 Output Page



Flight Delay Prediction

Welcome to Flight Delay Prediction Have a Nice Day

Enter the Flight Number : 1399
Month : 1
Day of Month : 1
Day of Week : 5
origin MSP
destination SEA
Scheduled Departure Time : 10
Scheduled Arrival Time : 20
Actual Departure Time : 15

SUBMIT

The Flight will be on time

Copyright © 2022,Flight Delay Prediction

After giving all the inputs the predictions is shown in red color in the picture above which says that "Flight On Time".

7 ADVANTAGES & DISADVANTAGES

- **Advantages**
  - Reduce time complexity
  - Handles roughest(enormous) amount of data using random forest algorithm and feature selection.
  - Cost effective.
- **Disadvantages**
  - Prediction of flight delay results is not accurate.
  - Data mining techniques does not help to provide effective decision making.

8 APPLICATIONS

We can apply this application in Air Travel Area. The proposed working model can also help in knowing the current status of flight. The model can also serve the purpose of training tool for business man and all other people who want to travel in flight.

9 CONCLUSION

The application gives details about the range of different methodology that is used or can be used to find out the delay in flights. As flight delay cost a lot to the airlines as well as passengers in financial and environmental terms, flight delay is the talk of the hour. Flight delay causes surging of prices by costing a lot on operational purpose They may increase prices to customers and operational prices to airlines. As the outcome is directly associated with the passenger and the airlines which in turn is liked to another set of airline and passengers it is very crucial to get real time delay for each player within the air transport system. hence there is a requirement to develop a system to predict the delay in flights to scale back monetary loss and for the higher and smooth operation.

10 FUTURE SCOPE

The project can be further enhanced by deploying the machine learning model obtained using a web application and a larger dataset could be used for prediction to give higher accuracy and produce better results.

This project helps to predict the status of flight and know the flight was on time or delay.

11 BIBILOGRAPHY

- https://www.ijert.org/flight-delay-prediction-system
- https://www.researchgate.net/publication/341872407_Flight_Delay_Prediction_System

12 APPENDIX

- **Source code of python**

```
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.
client_e8c92023d6b1438585a0fb4d54b4dc84 =
ibm_boto3.client(service_name='s3',
    ibm_api_key_id='8pEYICKE7x7LDK1bjPIAZ_I2BfBd-p9HCnWRcp9WXLms',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu.cloud-object-
storage.appdomain.cloud')

body =
client_e8c92023d6b1438585a0fb4d54b4dc84.get_object(Bucket='flightdela
yprediction-donotdelete-pr-
arjthz3g6v4jyr',Key='flightdata.csv')['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.isnull().any()
dataset.isnull().sum()
dataset['DEST'].unique()
dataset = dataset.drop('Unnamed: 25', axis=1)
dataset.isnull().sum()
import seaborn as sns
%matplotlib inline
flight_data = pd.read_csv(body)
flight_data.describe()
sns.scatterplot(x='ARR_DELAY',y='ARR_DEL15',data=flight_data)
sns.catplot(x="ARR_DEL15",y="ARR_DELAY",kind='bar',data=flight_data)
sns.heatmap(dataset.corr())
#filter the dataset to eliminate columns that aren't relevant to a
predictive model.
dataset = dataset[["FL_NUM", "MONTH", "DAY_OF_MONTH", "DAY_OF_WEEK",
"ORIGIN", "DEST", "CRS_ARR_TIME","DEP_DEL15", "ARR_DEL15"]]
dataset.isnull().sum()
dataset[dataset.isnull().any(axis=1)].head(10)
dataset['DEP_DEL15'].mode()
#replace the missing values with 1s.
dataset = dataset.fillna({'ARR_DEL15': 1})
dataset = dataset.fillna({'DEP_DEL15': 0})
dataset.iloc[177:185]
import math

for index, row in dataset.iterrows():
    dataset.loc[index, 'CRS_ARR_TIME'] =
math.floor(row['CRS_ARR_TIME'] / 100)
dataset.head()
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
dataset['DEST'] = le.fit_transform(dataset['DEST'])
dataset['ORIGIN'] = le.fit_transform(dataset['ORIGIN'])
dataset['ORIGIN'].unique()
dataset = pd.get_dummies(dataset, columns=['ORIGIN', 'DEST'])
dataset.head()
x = dataset.iloc[:, 0:8].values
y = dataset.iloc[:, 8:9].values
x
y
x.shape
y.shape
from sklearn.preprocessing import OneHotEncoder
oh = OneHotEncoder()
z=oh.fit_transform(x[:,4:5]).toarray()
t=oh.fit_transform(x[:,5:6]).toarray()
#x=np.delete(x,[4,7],axis=1)
z
t
x=np.delete(x,[4,5],axis=1)
x.shape
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.2,random_state=0)
x_test.shape
x_train.shape
y_test.shape
y_train.shape
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(random_state = 0)
classifier.fit(x_train,y_train)
decisiontree = classifier.predict(x_test)
decisiontree
from sklearn.metrics import accuracy_score
```

```
desacc = accuracy_score(y_test,decisiontree)
desacc
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,decisiontree)
cm
import sklearn.metrics as metrics
fpr1 ,tpr1 ,threshold1 =metrics.roc_curve(y_test,decisiontree)
roc_auc1 = metrics.auc(fpr1,tpr1)
fpr1
tpr1
threshold1
import matplotlib.pyplot as plt
plt.title("roc")
plt.plot(fpr1,tpr1,'b',label = 'Auc = %0.2f'% roc_auc1)
plt.legend(loc = 'lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([0,1])
plt.ylim([0,1])
plt.xlabel('tpr')
plt.ylabel('fpr')
plt.show()
import pickle
pickle.dump(classifier,open('flight.pkl','wb'))
import ibm_watson_machine_learning
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
wml_credentials={"url":"https://eu-gb.ml.cloud.ibm.com",
                 "apikey":"AI_l6st4kEKyuQT-
D5JEudm3X_kLBjZL98srb6t1ytmI"}
wml_client=APIClient(wml_credentials)
wml_client.spaces.list()
SPACE_ID="ed90ea5b-01e0-4de1-bc34-1baf34d5e4e0"
wml_client.set.default_space(SPACE_ID)
MODEL_NAME = 'Flightmodel'
DEPLOYMENT_NAME = 'flight_deploy'
bestmodel = classifier
# Set Python Version
software_spec_uid =
wml_client.software_specifications.get_id_by_name('default_py3.8')
# Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_0.23',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:
software_spec_uid
}
#Save model
model_details = wml_client.repository.store_model(
    model= bestmodel,
    meta_props= model_props,
    training_data= x_train,
    training_target=y_train,
)
print(model_details)
software_spec_uid
model_uid=wml_client.repository.get_model_id(model_details)
```

```
model_uid
#set meta
deployment_props= {

wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE:{}
}
deployment = wml_client.deployments.create(
    artifact_uid=model_uid,
    meta_props=deployment_props
)
deployment_uid =wml_client.deployments.get_uid(deployment)
deployment_uid
payload = {"input_data":
           [
               {"fields":x_test.tolist(), "values":x_test.tolist()}
           ]
          }
result = wml_client.deployments.score(deployment_uid, payload)
import sklearn
sklearn.__version__
```

❖ **Flask Application Coding**

```
#Importing Libraries

from flask import Flask,render_template,request

import pickle
import numpy as np

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 = "35jnX8LPiSw8ysLmGDh5Evb2_A646U_rjQH2rIPg9kkh"
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}

# NOTE: manually define and pass the array(s) of values to be
scored in the next line
```

```
#model = pickle.load(open('flight.pkl','rb'))

app = Flask(__name__)



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


@app.route('/prediction',methods =['GET','POST'])
def predict():
    name = request.form['name']
    month = request.form['month']
    dayofmonth = request.form['dayofmonth']
    dayofweek = request.form['dayofweek']
    origin = request.form['origin']
    if(origin == "msp"):
        origin1,origin2,origin3,origin4,orgin5 = 0,0,0,0,1
    if(origin == "dtw"):
        origin1,origin2,origin3,origin4,orgin5 = 1,0,0,0,0
    if(origin == "jfk"):
         origin1,origin2,origin3,origin4,orgin5 = 0,0,1,0,0
    if(origin == "sea"):
         origin1,origin2,origin3,origin4,orgin5 = 0,1,0,0,0
    if(origin == "alt"):
         origin1,origin2,origin3,origin4,orgin5 = 0,0,0,1,0

    destination = request.form['destination']
    if(destination == "msp"):

destination1,destination2,destination3,destination4,destinatio
n5 = 0,0,0,0,1
    if(destination == "dtw"):

destination1,destination2,destination3,destination4,destinatio
n5 = 1,0,0,0,0
    if(destination == "jfk"):

destination1,destination2,destination3,destination4,destinatio
n5 = 0,0,1,0,0
    if(destination == "sea"):

destination1,destination2,destination3,destination4,destinatio
n5 = 0,1,0,0,0
    if(destination == "alt"):

destination1,destination2,destination3,destination4,destinatio
n5 = 0,0,0,1,0
```

```
    dept = request.form['dept']
    arrtime = request.form['arrtime']
    actdept = request.form['actdept']
    dept15=int(dept)-int(actdept)
    total =
[[name,month,dayofmonth,dayofweek,origin1,origin2,origin3,orig
in4,orgin5,destination1,destination2,destination3,destination4
,destination5,int(arrtime),int(dept15)]]
    #print(total)

    payload_scoring = {"input_data": [{"field":
['name','month','dayofmonth','dayofweek','origin1','origin2','
origin3','origin4','orgin5','destination1','destination2','des
tination3','destination4','destination5','arrtime','dept15'],
"values":total }]}
    response_scoring =  requests.post('https://eu-
gb.ml.cloud.ibm.com/ml/v4/deployments/445aa1d5-d1cb-4f9a-9e0d-
9720be5445d5/predictions?version=2022-03-30',
json=payload_scoring, headers={'Authorization': 'Bearer ' +
mltoken})
    print("Scoring response")
    print(response_scoring.json())
    predictions =response_scoring.json()

    print(predictions)

    print('Final Prediction
Result',predictions['predictions'][0]['values'][0][0])
    y_pred = predictions['predictions'][0]['values'][0][0]
    #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",showcase = ans)

if __name__ == '__main__':
    app.run(debug = False)
```

GitHub: Source file