DEPARTMENT OF COMPUTER APPLICATION

ARIGNAR ANNA GOVERNMENT
ARTS AND SCIENCE COLLEGE
, CHEYYAR-604 407
(Affiliated to THIRUVALLUVAR UNIVERSITY)

FLIGHT DELAY PREDICTION FOR AVIATION INDUSTRY USING MACHINE LEARNING

Team Leader : EMIMA. P(20120U09020)

Team Members: KIRUBAKARAN. N(20120U09033)

KISHOREKUMAR. V(20120U09034)

LAKSHMAN NARAYANAN. K(20120U09037)

INTRODUCTION

Overview:

Due to its quickness and occasional comfort, air travel has become more and more popular among tourists during the past 20 years. The result has been a spectacular increase in land traffic and air traffic. Massive levels of aircraft delays on the ground and in the air have also been brought on by an increase in air traffic. There have been significant monetary and environmental losses as a result of these delays. To optimize flight operations and reduce delays, the model's primary goal is to estimate flight delays accurately.

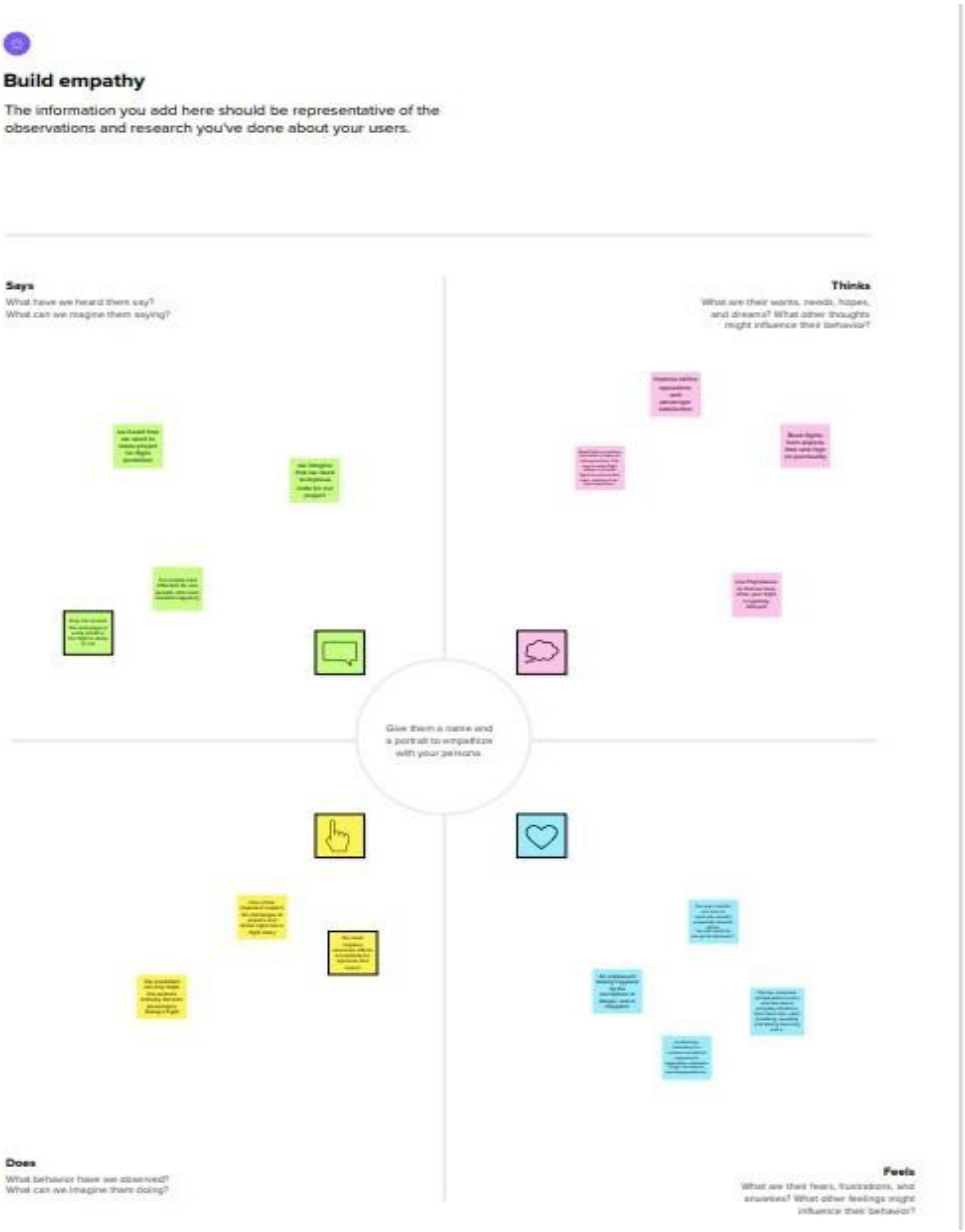
Flight arrival delays can be predicted using a machine learning algorithm. Rows of feature vectors, such as departure date, delay, travel time between the two airports, and scheduled arrival time, provide the input to our algorithm. The Support Vector Machine is then used to determine whether or not 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.

Purpose:

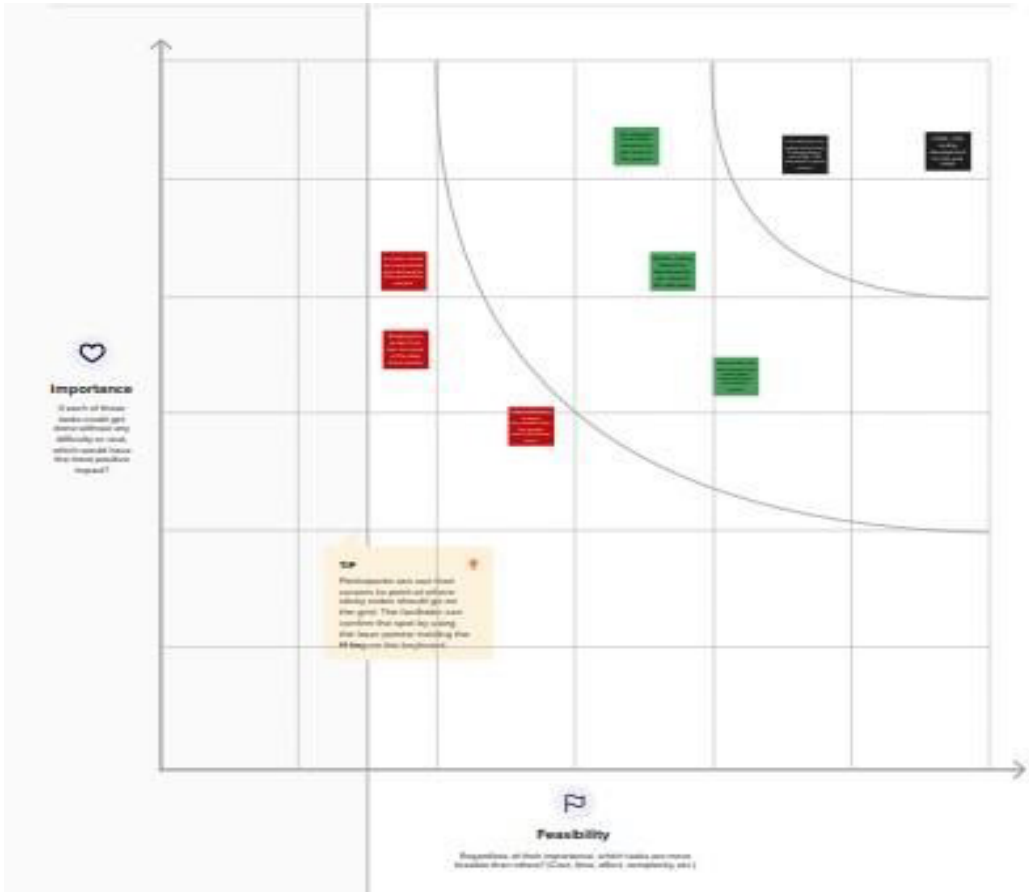
Flight delay forecasting can enhance airline operations and passenger happiness, which will boost the economy. Comparing the effectiveness of machine learning classification systems for predicting flight delays is the major objective. Flight delays are unavoidable, and they significantly affect the carriers' profits and losses. The traveler's ability to plan ahead and avoid wasting vital time can be greatly aided by this delay prediction. For airlines, estimating flight delays correctly is essential since the data may be used to boost client happiness and revenue for airline agencies.

Problem Definition & Design Thinking

Empathy Map:



Ideation & Brainstorming Map:



RESULT

Performance metrics:

Logistic Regression performance values

There is no big variation in the training and testing accuracy. Therefore, the Logistic Regression model is not overfit or underfit.

```
[42]
log_train_acc = accuracy_score(Y_train, Y_pred_log_train)
log_test_acc = accuracy_score(Y_test, Y_pred_log_test)
print('Training Accuracy =', log_train_acc)
print('Testing Accuracy =', log_test_acc)

Training Accuracy = 0.9206366874443455
Testing Accuracy = 0.9194481530930129
```

Decision tree Classifier

There is a variation in the training and testing accuracy. The Decision tree classifier model is overfit. So, it is not chosen for best results.

```
[46]
clf_train_acc = accuracy_score(Y_train, Y_pred_clf_train)
clf_test_acc = accuracy_score(Y_test, Y_pred_clf_test)
print('Training Accuracy =', clf_train_acc)
print('Testing Accuracy =', clf_test_acc)

Training Accuracy = 1.0
Testing Accuracy = 0.8722741433021807
```

KNN Classifier

There is no big variation in the training and testing accuracy. Therefore, the KNN Classifier model is not overfit or underfit.

```
[50]
knn_train_acc = accuracy_score(Y_train, Y_pred_knn_train)
knn_test_acc = accuracy_score(Y_test, Y_pred_knn_test)
print('Training Accuracy =', knn_train_acc)
print('Testing Accuracy =', knn_test_acc)

Training Accuracy = 0.8876892252894034
Testing Accuracy = 0.8629283489096573
```

Random Forest Classifier

There is no big variation in the training and testing accuracy. Therefore, the Random Forest Classifier model is not overfit or underfit.

```
[54]
    rf_train_acc = accuracy_score(Y_train, Y_pred_rf_train)
    rf_test_acc = accuracy_score(Y_test, Y_pred_rf_test)
    print('Training Accuracy =', rf_train_acc)
    print('Testing Accuracy =', rf_test_acc)

...  Training Accuracy = 0.9019367764915405
     Testing Accuracy = 0.894971072541166
```

On comparing the four models built, based on the performance metrics it is clear that logistic regression gives the highest performance. Hence, that model is chosen for deployment.

FINAL OUTPUT:

ADVANTAGES & DISADVANTAGES

Advantages:

- The application is fast and offers great accuracy in predicting the flight delay.
- Less maintenance is required.
- It is user friendly.
- It helps in reducing the tension of the passengers in knowing how long they will have to wait and lets passengers plan their schedule accordingly, thus in a way saving their time.

Disadvantages:

- It requires an internet connection for the website to work.

APPLICATIONS

- An accurate estimation of flight delay is critical for airlines because the results can be applied to increase customer satisfaction and incomes of airline agencies.
- Flight delays hurt airlines, airports, and passengers. Their prediction is crucial during the decision-making process for all players of commercial aviation.

CONCLUSION

Flight delays not only anger and disturb air travellers plans, but they also reduce efficiency, raise capital costs, reallocate flight crews and aircraft, and add to crew costs. The goal of the flight delay prediction model is to forecast aircraft delays caused as a lot of passengers have become dependent on flights these days for their mode of transportation.

The dataset which has all important information about the flights and its delay is made use for developing the model. A lot of steps are performed right from importing the data, then pre-processing it till training and testing the model. First the necessary packages were imported then the missing values in the data were handled and it was checked for outliers and then one hot encoding was performed and scaling was done. Then the data was split and given for training.

Four different models were used for training and out of it the best one was chosen based on the performance metric which is the Logistic regression model. Once the model was built it was integrated along with the Flask framework so that the users can enter their flight details and see if the flight would be on time or get delayed. Then this model is trained and deployed in the IBM Cloud.

As a result, anticipating delays can enhance airline operations and passenger satisfaction, which will benefit the economy and bring a positive impact.

FUTURE SCOPE

In the future, the application can be included with an user authentication model. Apart from checking if the flight would get delayed or not, their search history can be

maintained and personalized flight recommendations can be done. A section where the users can give their feedback can also be implemented.

APPENDIX

SOURCE CODE:

FLASK FILE:

app. py

```
from flask import Flask , render_template , request
import pandas as pd
import joblib
import numpy as np


app =  Flask (_name_)


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


@ app . route ( '/result' , methods = [ 'POST' ])
def predict ():
        fl_num = int ( request . form. get( 'fno' ))
        month = int ( request . form. get( 'month' ))
        dayofmonth = int ( request . form. get( 'daym' ))
        dayofweek = int ( request . form. get( 'dayw' ))
```

```
        sdeptime = request . form. get( 'sdt' )
        adeptime = request . form. get( 'adt' )
        arrtime = int ( request . form. get( 'sat' ))
        depdelay = int ( adeptime ) -  int ( sdeptime )
        inputs = list ()
```

```
        inputs.append(fl_num)
        inputs.append(month)
        inputs.append(dayofmonth)
        inputs.append(dayofweek)
        if(depdelay<15):
                inputs.append(0)
        else:
                inputs.append(1)
        inputs.append(arrtime)
        origin = str(request.form.get("org"))
        dest = str(request.form.get("dest"))
        if(origin=="ATL"):
                a=[1,0,0,0,0]
                inputs.extend(a)
        elif(origin=="DTW"):
                a=[0,1,0,0,0]
                inputs.extend(a)
        elif(origin=="JFK"):
                a=[0,0,1,0,0]
                inputs.extend(a)
        elif(origin=="MSP"):
                a=[0,0,0,1,0]
                inputs.extend(a)
        elif(origin=="SEA"):
                a=[0,0,0,0,1]
                inputs.extend(a)


        if(dest=="ATL"):
                b=[1,0,0,0,0]
                inputs.extend(b)
        elif(dest=="DTW"):
```

```
                b=[0,1,0,0,0]
```

```
            inputs.extend(b)
        elif(dest=="JFK"):
            b=[0,0,1,0,0]
            inputs.extend(b)
        elif(dest=="MSP"):
            b=[0,0,0,1,0]
            inputs.extend(b)
        elif(dest=="SEA"):
            b=[0,0,0,0,1]
            inputs.extend(b)


        prediction=preprocessAndPredict(inputs)
        #Pass prediction to prediction template
        print(inputs)
        return render_template('/result.html',prediction=prediction
       )


def preprocessAndPredict(inputs):
        test_data=np.array(inputs).reshape((1,16))


        model_file=open('H:\\New folder (2)\\Final Deliverables\\Local
Deployment\\model.pkl','rb')


        trained_model=joblib.load(model_file)


           df=pd.DataFrame(data=test_data[0:,0:],columns=[
                                                    'FL_NUM','MONTH',
'DAY_OF_MONTH',          'DAY_OF_WEEK',          'DEP_DEL15',          'CRS_ARR_TIME',
'ORIGIN_ATL',
'ORIGIN_DTW','ORIGIN_JFK','ORIGIN_MSP','ORIGIN_SEA','DEST_ATL',
'DEST_DTW','DEST_JFK','DEST_MSP','DEST_SEA'])
```

```
        data = df . values



        result = trained_model . predict( data )



        print ( result )
        return result



if _name_ == '__main_' :
        app . run ( debug = True )
```

HTML JS and CSS files:

FLIGHT DELAY. html

```
<! DOCTYPE html >
< html lang = "en" >
< head >
        < meta charset = "UTF-8" >
        < meta http-equiv = "X-UA-Compatible" >
                                                < meta name = "viewport" content = "width=device-width, initial-
                                                                                    scale=1. 0" >
        < link rel = "stylesheet" href = " {{
url_for ( 'static' , filename = 'styles/styles. css' ) }} " >
        < script src = " {{ url_for ( 'static' ,
filename = 'styles/delaypredict. js' )}} " ></ script >
        < title > Flight Delay Prediction </ title >
</ head >
< body id = "flight-form" >
< h2 id = "main-head" class = "centered-head" > FLIGHT DELAY PREDICTION </ h2 >

< img src = " {{ url_for ( 'static',                    filename =
'styles/images/Flight. png' )}} "
id = "bgimg" >
< form name = "flightForm" action = "/result" method = "POST" target
= "_blank" >
        < div id = "form-content" >
                < div id = "block1" >
                        < div class = "detail-container" >
```

```
                    < label for = "fno" class = "label-item" > Enter the Flight
Number </ label >
                    < br >
                        < input type = "number" id = "fno" name = "fno"
class = "text-input" >
                </ div >
                < div class = "detail-container" >
                    < label for = "month" class = "label-item" > Month </ label >
                    < br >
```

```
                    < input type = "number" id = "month" name = "month"
class = "text-input" onblur = "checkValid('month');" placeholder = "Enter the
Month Number" >
                    < div class = "alert-text" id = "month-valid" > Enter a valid
month between 1 to 12. </ div >
                </ div >
                < div class = "detail-container" >
                                    < label for = "daym" class = "label-item" > Day of Month </ label >
                    < br >
                        < input type = "number" id = "daym" name = "daym"
class = "text-input" onblur = "checkValid('daym');" >
                                        < div class = "alert-text" id = "daym-valid" > Enter a valid day
of month. </ div >
                </ div >
                < div class = "detail-container" >
    < label for = "dayw" class = "label-item" > Day of Week </ label >
                    < br >
                        < input type = "number" id = "dayw" name = "dayw"
class = "text-input" onblur = "checkValid('dayw');" >
                                        < div class = "alert-text" id = "dayw-valid" > Enter a valid day
between 1 to 7. </ div >
                </ div >
```

```
                    < div class = "detail-container" >
                        < label for = "org" class = "label-item" > Origin </ label >
                        < br >
                        < select id = "org" name = "org" class = "select- input" >
                                                    < option value = "ATL" class = "option-item" > ATL </ option >
                                                    < option value = "SEA" class = "option-item" > SEA </ option >
                                                   < option value = "DTW" class = "option-item" > DTW </ option >
                                                    < option value = "MSP" class = "option-item" > MSP </ option >
                                                    < option value = "JFK" class = "option-item" > JFK </ option >
                        </ select >
                    </ div >
                    < div class = "detail-container" >
                                                   < label for = "dest" class = "label-item" > Destination </ label >
                        < br >
```

```
                        < select id = "dest" name = "dest" class = "select- input"
onblur = "checkValid('dest'); " >
                                                    < option value = "ATL" class = "option-item" > ATL </ option >
                                                    < option value = "SEA" class = "option-item" > SEA </ option >
                                                   < option value = "DTW" class = "option-item" > DTW </ option >
                                                    < option value = "MSP" class = "option-item" > MSP </ option >
                                                    < option value = "JFK" class = "option-item" > JFK </ option >
                        </ select >
        < div class = "alert-text" id = "dest-valid" > Enter different
Origin and Destination. </ div >
                    </ div >
                </ div >
                < div id = "block2" >
                    < div class = "detail-container" >
        < label for = "sdt" class = "label-item" > Scheduled Departure
Time </ label >
                        < br >
```

```
                                    < input type = "number" id = "sdt" name = "sdt"
class = "text-input" onblur = "checkValid('sdt'); " placeholder = "Enter in the
format HHMM" >
                                                                < div class = "alert-text" id = "sdt-valid" > Enter a valid time
between 500 to 2359. </ div >
                        </ div >
                        < div class = "detail-container" >
                                < label for = "sat" class = "label-item" > Scheduled Arrival
Time </ label >
                                < br >
                                        < input type = "number" id = "sat" name = "sat"
class = "text-input" onblur = "checkValid('sat'); " placeholder = "Enter in the
format HHMM" >
                                                                < div class = "alert-text" id = "sat-valid" > Enter a valid time
between 500 to 2359. </ div >
                        </ div >
                        < div class = "detail-container" >
                                < label for = "adt" class = "label-item" > Actual Departure
Time </ label >
                                < br >
                                        < input type = "number" id = "adt" name = "adt"
class = "text-input" onblur = "checkValid('adt'); " placeholder = "Enter in the
format HHMM" >
                                                                < div class = "alert-text" id = "adt-valid" > Enter a valid time
between 500 to 2359. </ div >
                        </ div >
                </ div >
        </ div >
        < div id = "submit-button" >
                < input  type = "submit" value = "Submit" id = "submit" class
            = "button"
onclick = "validateForm()" >
```

```
        </ div >
</ form >
</ body >
</ html >
```

STYLE CSS:

```
body {
        font-family : Arial , Helvetica , sans-serif ;
        margin : 0 ;
}
. content {
        padding : 10px ;
        display : block ;
}
. content-head {
        text-align : center ;
        font-weight : bold ;
        font-size : 36px ;
}
. button {
        background-color : #1C55A2 ;
        color :  aliceblue ;
        padding : 10px ;
        border-radius :  10px ;
```

```
                    border-color : #0E0E0F ;
        border-width : 1.5px ;
}
. button a {
        color :  aliceblue ;
                        text-decoration : none ;
        font-weight : bold ;
}
#feedback-button {
        margin-top :  10px ;
```

```
}
#feedback-button-section {
        text-align : center ;
}
#bgimg {
        position : fixed ;
        z-index : -1 ;
        opacity : 0.5 ;
        width : 100% ;
        height : 100% ;
        padding : 0 ;
        margin : 0 ;
        top : 0 ;
}
.centered-head {
        text-align : center ;
        color : #1C55A2 ;
        font-weight : bold ;
}
.label-item {
        color : #2E547F ;
        font-weight : bold ;
}
.detail-container {
        padding-bottom : 10px ;
        padding-top : 10px ;
```

```
}
.text-input {
        margin-top : 5px ;
        border-color : #1C55A2 ;
        border-width : 1.5px ;
        border-radius :  10px ;
        width :  75% ;
```

```
        height : 20px ;
        padding-left : 5px ;
        padding-right : 5px ;
        padding-top : 2px ;
        padding-bottom :  2px ;
}
. select-input {
        margin-top : 5px ;
        border-color : #0E0E0F ;
        border-width : 1. 5px ;
        border-radius :  10px ;
        width :  40% ;
        height : 30px ;
        background-color : #1C55A2 ;
        color :  aliceblue ;
        font-weight : bold ;
        cursor : pointer ;
}
#form-content {
        display : flex ;
        justify-content : space-evenly ;
        flex-direction :  row ;
        padding-left : 10% ;
}
#block1 {
        display : block ;
        width :  50% ;
        padding : 20px ;
}
```

```
#block2 {
        display : block ;
        width :  50% ;
        padding : 20px ;
```

```
}
#review {
        height : 100px ;
        padding-top : 5px ;
        font-family : Arial , Helvetica , sans-serif ;
}
#submit-button {
        text-align : center ;
        align-items : center ;
        display : block ;
}
#submit {
        background-color : #1C55A2 ;
        color : aliceblue ;
        font-weight : bold ;
}

#submit:hover {
        cursor : pointer ;
}

.choose-item {
        font-weight : 600 ;
}

input [ type = "radio" ], input [ type = "checkbox" ] {
        cursor : pointer ;
}

.alert-text {
        color : rgb ( 255 , 79 , 47 );
        font-size : small ;
```

```
        padding-left : 10px ;
        display : none ;
}
```

delaypredict. js

```
function validateForm () {
        var fno = document . forms [ "flightForm" ][ "fno" ]. value ;
        var month = document . forms [ "flightForm" ][ "month" ]. value ;
        var daym = document . forms [ "flightForm" ][ "daym" ]. value ;
        var dayw = document . forms [ "flightForm" ][ "dayw" ]. value ;
        var org = document . forms [ "flightForm" ][ "org" ]. value ;
        var dest = document . forms [ "flightForm" ][ "dest" ]. value ;
        var sdt = document . forms [ "flightForm" ][ "sdt" ]. value ;
        var sat = document . forms [ "flightForm" ][ "sat" ]. value ;
        var adt = document . forms [ "flightForm" ][ "adt" ]. value ;
        if ( fno == "" || fno == null || month == "" || month
      == null || daym
 == "" || daym == null || dayw == "" || dayw == null || org
== "" || org ==
 null || dest == "" || dest == null || sdt == "" || sdt == null || sat  ==

 "" || sat == null || adt == "" || adt == null ) {
                alert ( "The given fields must be filled out" );
                event . preventDefault ();
        }
        if ( month < 1 || month > 12 )
        {
                alert ( "Enter a valid month" );
                event . preventDefault ();
        }
        if ( month == 2 )
        {
                if ( daym < 1 || daym >= 29 )
                {
                        alert ( "Enter a valid day of month" );
                        event . preventDefault ();
```

```
            }
        }
```

```
                    else if ( month == 1 || month == 3 || month == 5 || month == 7
                                || month ==8                ||
month == 10 || month == 12 )
        {
            if ( daym < 1 || daym > 31 )
            {
                alert ( "Enter a valid day of month" );
                event . preventDefault ();
            }
        }
        else if ( month == 4 || month == 6 || month == 9 || month == 11 )
        {
            if ( daym < 1 || daym > 30 )
            {
                alert ( "Enter a valid day of month" );
                event . preventDefault ();
            }
        }
        if ( dayw < 1 || dayw > 7 )
        {
            alert ( "Enter a valid day of week" );
            event . preventDefault ();
        }
        if ( org == dest )
        {
            alert ( "Enter different origin and destination" );
            event . preventDefault ();
        }
        if ( sdt < 500 || sdt > 2400 )
        {
          alert ( "Enter a valid Departure time between 500 to 2400" );
```

```
            event . preventDefault ();
        }
        if ( sat < 500 || sat > 2400 )
        {
           alert ( "Enter a valid Arrival time between 500 to 2400" );
            event . preventDefault ();
        }
        if ( sdt == sat )
        {
            alert ( "Departure and Arrival time must differ by atleast 1 hr"
          );
            event . preventDefault ();
        }
        if ( adt < 500 || adt > 2400 )
        {
           alert ( "Enter a valid Departure time between 500 to 2400" );
            event . preventDefault ();
        }
}
function checkValid ( element )
{
        var obj = document . getElementById ( element );
        var valid_obj = document . getElementById ( element + "-valid"
      );
        if ( element == 'month' )
        {
            if ( obj . value < 1 || obj . value > 12 )
            {
                                obj . style . borderColor = "rgb(255, 79, 47)"
                valid_obj . style . display = "block" ;
            }
            else {
                obj . style . borderColor = "#1C55A2" ;
                valid_obj . style . display = "none" ;
```

```
            }
        }
        if ( element == 'daym' )
        {
                                var monobj = document . getElementByld ( 'month' );
            if ( monobj . value == 2 )
            {
                if ( obj . value < 1 || obj . value >= 29 )
                {
                                    obj . style . borderColor = "rgb(255, 79, 47)"
```

```
                                    valid_obj . style . display = "block" ;
                }
                else {
                                    obj . style . borderColor = "#1C55A2" ;
                                    valid_obj . style . display = "none" ;
                }
            }
            else if ( monobj . value == 1 || monobj . value == 3 || monobj . value == 5 ||
 monobj . value == 7 || monobj . value == 8 || monobj . value == 10
||
 monobj . value == 12 )
            {
                if ( obj . value < 1 || obj . value > 31 )
                {
                                    obj . style . borderColor = "rgb(255, 79, 47)"
                                    valid_obj . style . display = "block" ;
                }
                else {
                                    obj . style . borderColor = "#1C55A2" ;
                                    valid_obj . style . display = "none" ;
                }
            }
```

```
            else if ( monobj . value == 4 || monobj . value == 6 || monobj . value == 9  ||
monobj . value == 11 )
            {
                  if ( obj . value < 1 || obj . value > 30 )
                  {
                              obj . style . borderColor = "rgb(255, 79, 47)"
                                    valid_obj . style . display = "block" ;
                  }
                  else {
                                    obj . style . borderColor = "#1C55A2" ;
                                     valid_obj . style . display = "none" ;
                  }
            }
            else
            {
```

```
                              obj . style . borderColor = "rgb(255, 79, 47)"
                  valid_obj . style . display = "block" ;
            }
      }
      if ( element == 'dayw' )
      {
            if ( obj . value < 1 || obj . value > 7 )
            {
                              obj . style . borderColor = "rgb(255, 79, 47)"
                  valid_obj . style . display = "block" ;
            }
            else {
                  obj . style . borderColor = "#1C55A2" ;
                  valid_obj . style . display = "none" ;
            }
      }
      if ( element == 'dest' )
```

```
{
                        var origin_obj = document . getElementById ( 'org' );
        if ( obj . value == origin_obj . value )
        {
                            obj . style . borderColor = "rgb(255, 79, 47)"
                valid_obj . style . display = "block" ;
        }
        else
        {
                obj . style . borderColor = "#1C55A2" ;
                valid_obj . style . display = "none" ;
        }
}
if ( element == 'sdt' )
{
        if ( obj . value < 500 || obj . value > 2400 )
        {
                            obj . style . borderColor = "rgb(255, 79, 47)"
                valid_obj . style . display = "block" ;
```

```
        }
        else {
                obj . style . borderColor = "#1C55A2" ;
                valid_obj . style . display = "none" ;
        }
}
if ( element == 'sat' )
{
        if ( obj . value < 500 || obj . value > 2400 )
        {
                            obj . style . borderColor = "rgb(255, 79, 47)"
                valid_obj . style . display = "block" ;
        }
        else {
```

```
                obj.style.borderColor = "#1C55A2";
                valid_obj.style.display = "none";
            }
        }
        if (element == 'adt')
        {
            if (obj.value < 500 || obj.value > 2400)
            {
                                obj.style.borderColor = "rgb(255, 79, 47)"
                valid_obj.style.display = "block";
            }
            else {
                obj.style.borderColor = "#1C55A2";
                valid_obj.style.display = "none";
            }
        }
}
```

result.html

```
<!doctype html>
<html>
<head>
    <title>Flight Delay Prediction - Result</title>
    <link rel="stylesheet" href="{{
url_for('static', filename='styles/result_styles.css')}}">
</head>
<body>
                                <img src="{{url_for('static',                                filename=
                                                                                'styles/images/Flight.png')}}"
id="bgimg">
    {% if prediction[0] == 0.0 %}
    <div class="pred_result" id="result_0">Your flight will likely be on
time</div>
    {% endif %}
    {% if prediction[0] == 1.0 %}
```

```
< div class = "pred_result" id = "result_1" > Your flight is likely to be

delayed </ div >
        {% endif %}
</ body >
</ html >
```