OPTIMIZING FLIGHT BOOKING DECISIONS THROUGH MACHINE LEARNING PRICE PREDICTIONS

PROJECT REPORT

Submitted by

TEAM ID: NM2023TMID25456

PUGALENTHI S

NAVEENKUMAR M

SARANKUMAR P

YUVARAJ A

In partial fulfilment of the requirements for the award of the degree of Bachelor of computer science of Bharathiar University,Coimbatore -46.

Under the Guidance of

Dr.A.GEETHA

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE

GOVERNMENT ARTS AND SCIENCE COLLEGE (CO-ED)

(Affiliated to Bharathiar University, Coimbatore)

AVINASHI-641 654

NAAN MUDHALVAN PROJECT WORK

GOVERNMENT ARTS AND SCIENCE COLLEGE (CO-ED)

(Affiliated to Bharathiar University, Coimbatore)

AVINASHI-641 654

OPTIMIZING FLIGHT BOOKING DECISIONS THROUGH MACHINE LEARNING PRICE PREDICTIONS

This is to certify that this is a bonafide record of work done by the above students of III B.Sc. (CS) Degree **NAAN MUDHALVAN PROJECT** during the year 2022-2023

Submitted for the Naan Mudhalvan project work held on…….20

Class Mentor	**Head of Department**
Dr.A.GEETHA	**Prof. B. HEMALATHA**

CONTENTS

1. INTRODUCTION

1.1 Overview

A person who already has reserved a ticket for a flight realizes how powerfully the price of the ticket switches. Airline utilizes progressed techniques considered Revenue Management to accomplish a characteristic esteeming technique. The most affordable ticket available changes over a course of time. The expense of the booking may be far and wide. This esteeming technique normally alters the cost according to the different times in a day namely forenoon, evening, or night. Expenses for the flight may similarly alter according to the different seasons in a year like summers, rainy and winters, also during the period of festivals. The buyers would be looking for the cheapest ticket. While the outrageous objective of the transporter would be generating more and more revenue. Travelers for the most part attempt to buy the ticket ahead of their departure day. The reason would be their belief that the prices might be the highest when they would make a booking much nearer to the day of their flight but conventionally this isn't verifiable. The buyer might wrap up paying more than they should for a comparable seat. Considering the challenges faced by the travellers for getting an affordable seat, various strategies are utilized which will extract a particular day on which the fare will be the least. For this purpose, Machine Learning comes into the picture. Gini and Groves developed a model using PLSR, to predict the appropriate time to book the seats.

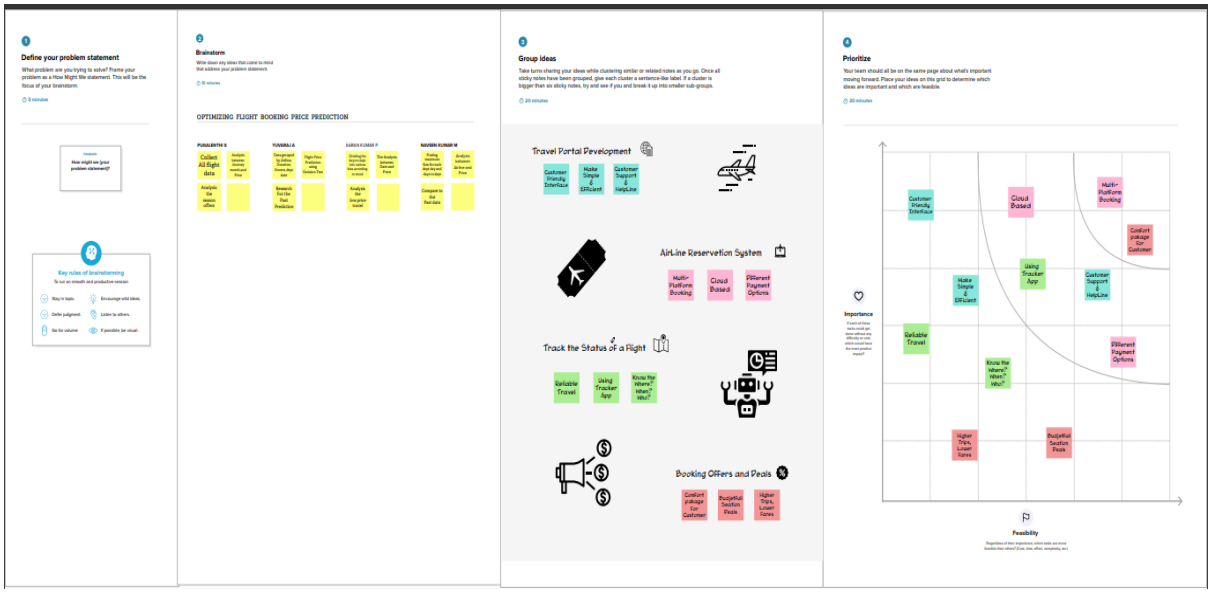
1.2 Purpose

People who work frequently travel through flight will have better knowledge on best discount and right time to buy the ticket. For the business purpose many airline companies change prices according to the seasons or time duration. They will increase the price when people travel more. Estimating the highest prices of the airlines data for the route is collected with features such as Duration, Source, Destination, Arrival and Departure. Features are taken from chosen dataset and in the price wherein the airline price ticket costs vary overtime. We have implemented flight price prediction for users by using KNN, decision tree and random forest algorithms. Random Forest shows the best accuracy of 80% for predicting the flight price. Also, we have done correlation tests and metrics for the statistical analysis

2. PROBLEM DEFINITION & DESIGN THINKING

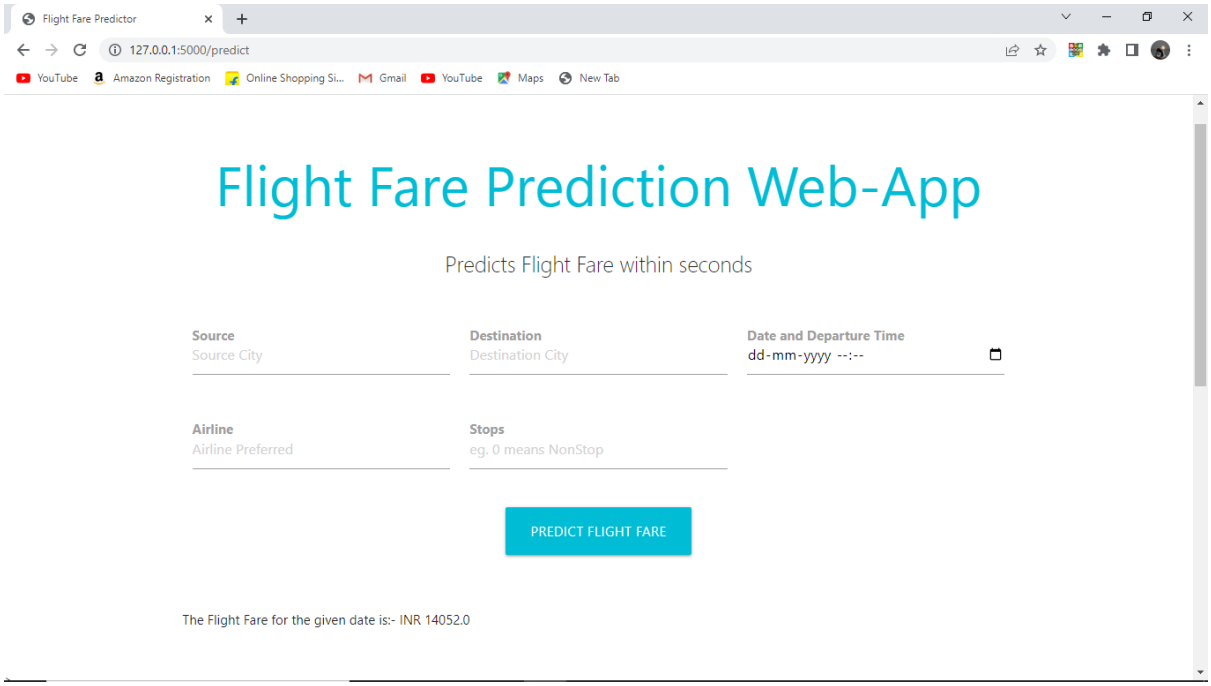
2.1 Empathy Map

2.2 Ideation & Brainstorming Map

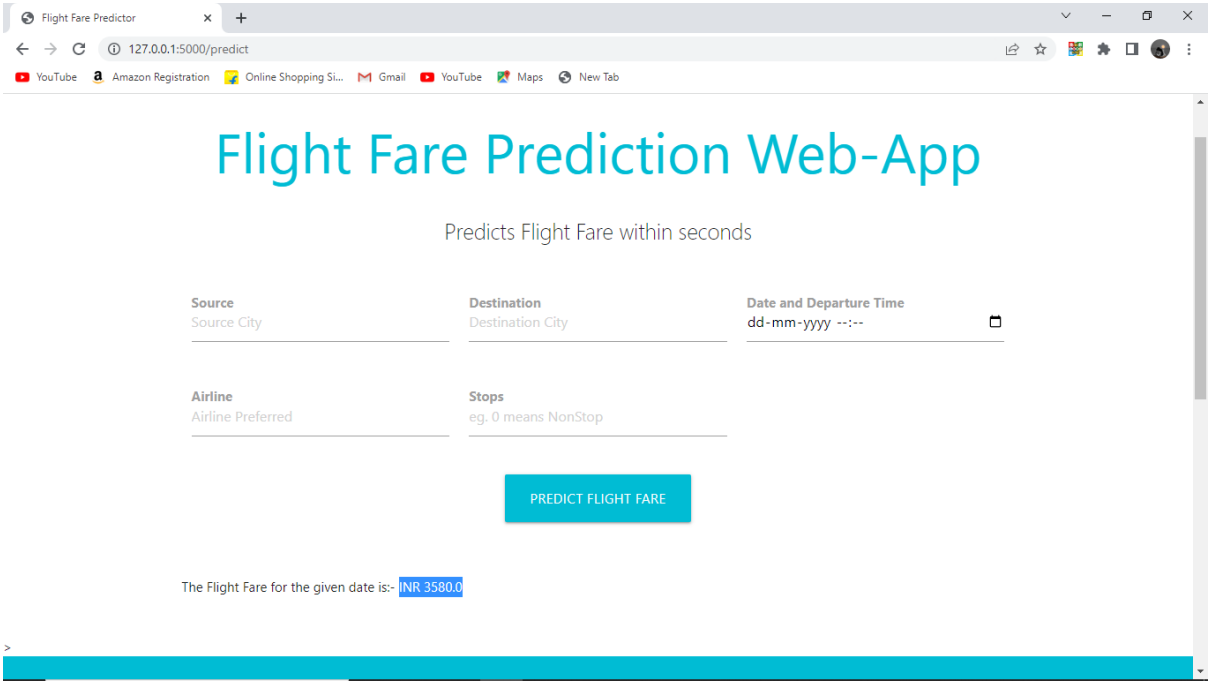


3. RESULT

Home page



Predicting page of Flight Price Prediction



4. ADVANTAGES AND DISADVANTAGES

Advantages

1. Traveller get the fare prediction handy using which it’s easy to decide the airlines.
2. Saves time in searching / deciding for airlines.

Disadvantages

1. Data will result in incorrect fare predictions.

5. APPLICATIONS

1. Make traveling easier

2. Airfare tracking

3. flight search and airfare prediction

4. Airfare tracking and hotel booking.

6. CONCLUSION

We learn that ML models can be used to predict prices based on earlier data more correctly. The presented paper reflects the dynamic change in the cost of flight tickets from which we get the information about the increase or decrease in the price as per the days, weekends, and the time of the day. With the Ml algorithm applied on various datasets, better results can be obtained for prediction. The error values that we got for Artificial Neural Network are comparatively high but for obtaining lesser values we can use evolutionary algorithms of ANN like genetic algorithms in the future.

7. FUTURE SCOPE

1. More routes can be added and the same analysis can be expanded to major airports and travel routes in india.
2. The analysis can be done by increasing the data points and increasing the historical data used. That will train the model better giving better accuracies and more savings.
3. More rules can be added in the rule-based learning based on our understanding of the industry, also incorporating the offer periods given by the airlines .
4. Developing a more user-friendly interface for various routes giving more flexibility to the users.

8. APPENDIX

A .SORCE CODE

- **app.py**

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

app = Flask(__name__)
model = pickle.load(open('model.pkl','rb'))

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

@app.route('/predict',methods=['GET','POST'])
def predict():
    ##For rendering result on HTML interface
    if request.method=='POST':
        features = [x for x in request.form.values()]
        source_dict = {'Bangalore': 0, 'Chennai': 1, 'Delhi': 2, 'Kolkata': 3,
'Mumbai': 4}
        destination_dict = {'Bangalore':0,'Cochin':1,'Delhi':2,'Kolkata':
3,'Hyderabad':4,'New Delhi':5}
        airline_dict = {'IndiGo': 3, 'Air India': 1, 'Jet Airways': 4, 'SpiceJet': 8,
'Multiple carriers': 6, 'GoAir': 2, 'Vistara': 10, 'Air Asia': 0, 'Vistara Premium
economy': 11, 'Jet Airways Business': 5, 'Multiple carriers Premium economy':
7, 'Trujet': 9}
```

```
    source_value = features[0]
    dest_value = features[1]
    date_value = features[2]
    airline_value = features[3]

    stops_value = int(features[4])   #<----------

    a= pd.Series(source_value)
    source = a.map(source_dict).values[0]   #<----------
    b= pd.Series(dest_value)
    destination = b.map(destination_dict).values[0] #<---------
    c= pd.Series(airline_value)
    airline = c.map(airline_dict).values[0]   #<----------

    day = int(pd.to_datetime(date_value, format="%Y-%m-
%dT%H:%M").day)    #<----------------
    month = int(pd.to_datetime(date_value, format="%Y-%m-
%dT%H:%M").month)  #<---------

    hour = int(pd.to_datetime(date_value, format ="%Y-%m-
%dT%H:%M").hour)
    minute = int(pd.to_datetime(date_value, format ="%Y-%m-
%dT%H:%M").minute)

    if  source==destination:
        return render_template('index.html',pred='Source and Destination City
cannot be same. Please try again! ')

    else:
```

```
        pred_features = [np.array([day,month,stops_value,hour,minute,airline,source,destination])]
        prediction = model.predict(pred_features)

        if stops_value==0:
            output = round(prediction[0],0)

        else:
            output = round(prediction[0],0)-2000

        return render_template('index.html',pred='The Flight Fare for the given date is:-  INR {}'.format(output))
    else:
        return render_template('index.html')
if __name__=='__main__':
    app.run(debug=True)
```

- **model.py**

```
import pandas as pd
```

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import time
import pickle

##Source - https://www.kaggle.com/nikhilmittal/flight-fare-prediction-mh
train=pd.read_excel('Data_Train.xlsx',engine='openpyxl')
sample = pd.read_excel('Sample_submission.xlsx',engine='openpyxl')
test = pd.read_excel('Test_set.xlsx',engine='openpyxl')
test = pd.concat([test,sample],axis=1)
df= pd.concat([train,test])

##Droping columns that does not seem practical to ask to a customer.
df.drop(labels=['Route','Arrival_Time','Duration','Additional_Info'],axis=1,in
place=True)
df.dropna(inplace=True)
df['Day']= df['Date_of_Journey'].str.split('/').str[0]
df['Month']= df['Date_of_Journey'].str.split('/').str[1]
df['Year']= df['Date_of_Journey'].str.split('/').str[2]

df['Total_Stops']=df['Total_Stops'].str.replace('non-','0 ')
df['Stops'] = df['Total_Stops'].str.split().str[0]
df['Departure_Hour'] = df['Dep_Time'].str.split(':').str[0]
df['Departure_Minute'] = df['Dep_Time'].str.split(':').str[1]

#Converting the datatype o newly created features
df['Day'] = df['Day'].astype(int)
df['Month'] = df['Month'].astype(int)
df['Year'] = df['Year'].astype(int)
df['Stops'] = df['Stops'].astype(int)
df['Departure_Hour'] = df['Departure_Hour'].astype(int)
df['Departure_Minute'] = df['Departure_Minute'].astype(int)

#Now droping the parent features since we don't need them
df.drop(['Date_of_Journey','Dep_Time','Total_Stops'],axis=1,inplace=True)
#Label encoding executed manually
source_dict = {y:x for x,y in
enumerate(df.Source.value_counts().index.sort_values())}
```

```
destination_dict = {'Banglore':0,'Cochin':1,'Delhi':2,'Kolkata':
3,'Hyderabad':4,'New Delhi':5}

from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['Airline_Encoded']= le.fit_transform(df['Airline'].values)

df3 = df[['Airline']].copy()
df3['Encoded']=df['Airline_Encoded']
df3=df3.drop_duplicates('Airline').reset_index().iloc[:,1:]
d5=df3.Airline.values
d6=df3.Encoded.values
airline_dict = dict(zip(d5,d6))

df['Source_Encoded']=df['Source'].map(source_dict)
df['Destination_Encoded']=df['Destination'].map(destination_dict)
df = df.drop(['Airline','Source','Destination'],axis=1)
#Feature Selection
from sklearn.linear_model import Lasso
from sklearn.feature_selection import SelectFromModel
from sklearn.model_selection import train_test_split

df_train = df[0:10600]
df_test = df[10600:]
X = df_train.drop(['Price'],axis=1)
y = df_train.Price
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=0)
model = SelectFromModel(Lasso(alpha=0.005,random_state=0))
model.fit(X_train,y_train)
features_selected = X_train.columns[model.get_support()]
##All features selected except Year
X_train = X_train.drop(['Year'],axis=1)
X_test = X_test.drop(['Year'],axis=1)

#Feature Normalization
import scipy.stats as stat
for x in list(X_train.columns):
    X_train[x] = stat.yeojohnson(X_train[x])[0]
```

```
for y in list(X_test.columns):
    X_test[y] = stat.yeojohnson(X_test[y])[0]

##Random forest regressor model
from sklearn.ensemble import RandomForestRegressor
reg=RandomForestRegressor()
reg.fit(X_train,y_train)

pickle.dump(reg,open('model.pkl','wb'))
model=pickle.load(open('model.pkl','rb'))
```