OPTIMIZING FLIGHT BOOKING DECISIONS THOUGHT MACHINE LEARNING PRICE PREDICTIOND

1. Introduction

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.

1.1 Overview

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 Duraction, Source, Destination, Arrival and Depaeture. Features are taken from chosen dataset and in the price wherein the airline price ticket costs vary overtime.

1.2 Purpose

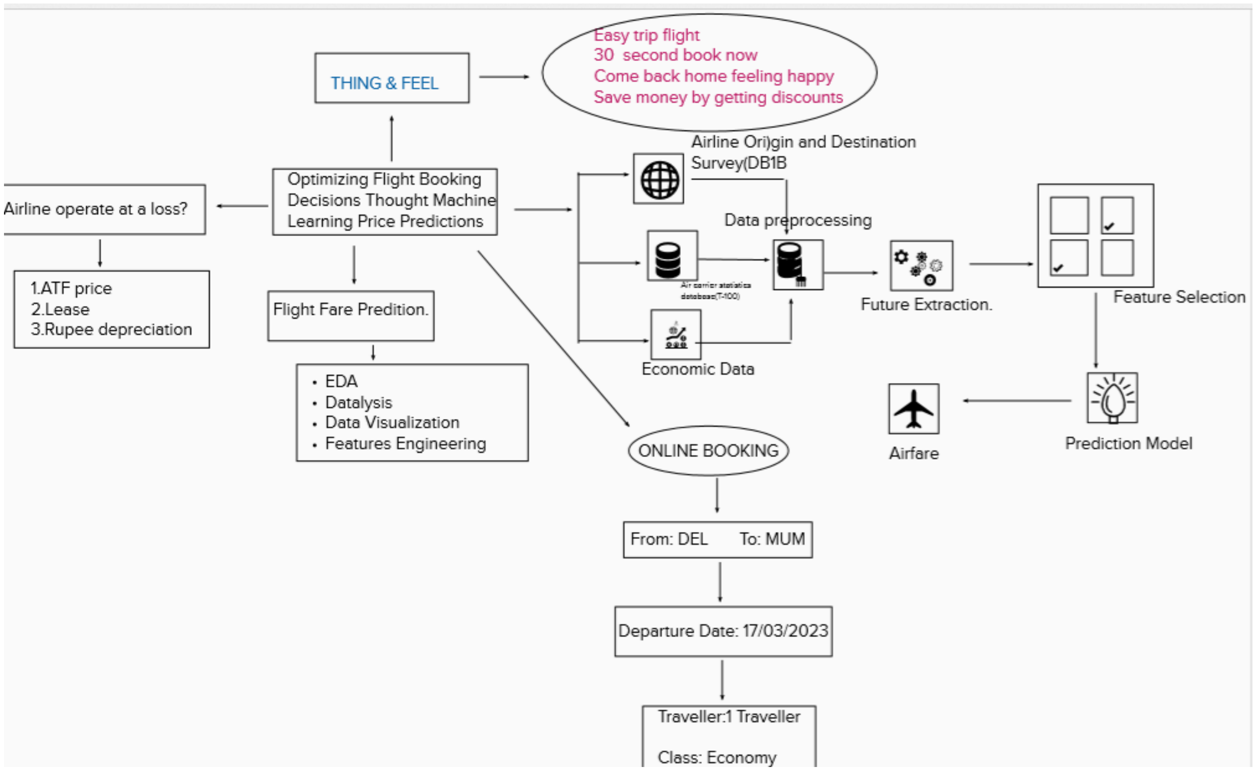
Flight booking can be a daunting task, especially when it comesto finding the best doal.

However, with the help of machine learning price predictions , travelers can optimize their booking decisions and save money.

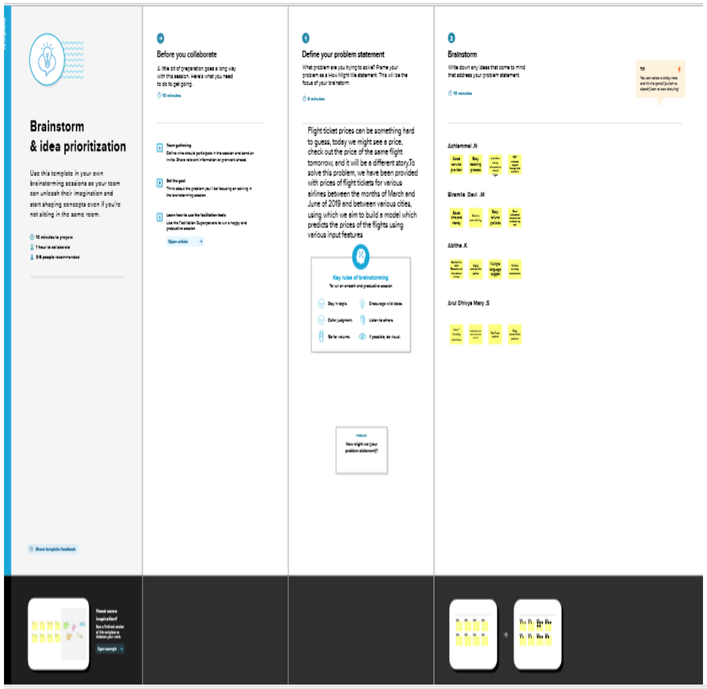
Machine learning algorithms analyze historical flight date to predict future prices, allowing travelers to make informed decisions about when and where to book their flights.

2. Problem definition & Design Thinking

2.1Empathy map



2.2 Ideation & Brainstorming Map



Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Before you collaborate

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.

Flight ticket prices can be something hard to guess, today we might see a price, check out the price of the same flight tomorrow, and it will be a different story.To solve this problem, we have been provided with prices of flight tickets for various airlines between the months of March and June of 2019 and between various cities, using which we aim to build a model which predicts the prices of the flights using various input features

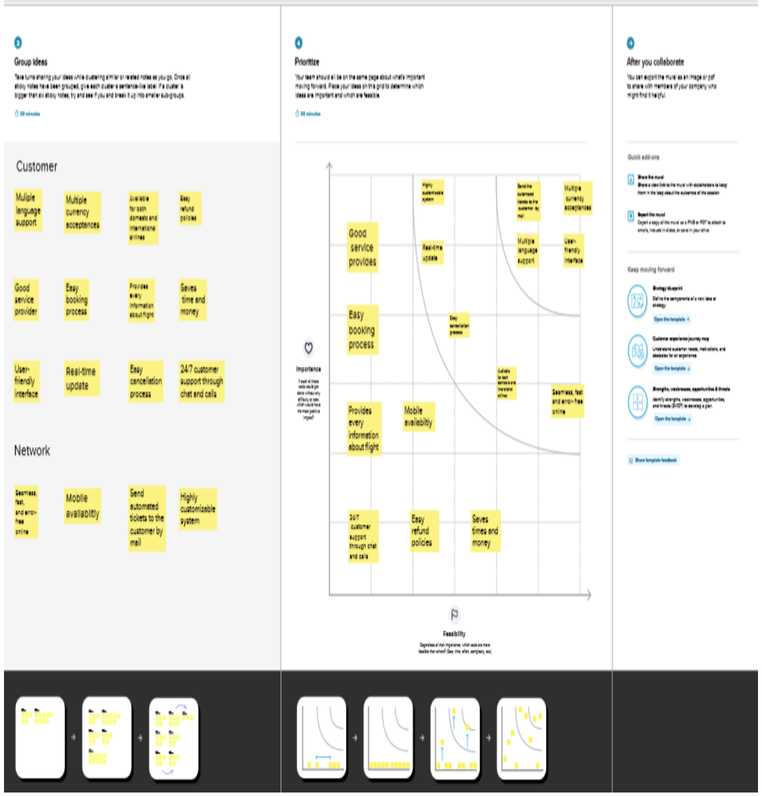
Key rules of brainstorming

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.

Brainstorm

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



Group Ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, 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.

Customer

Muliple language support | Multiple currency acceptances | Available for both domestic and international airlines | Easy refund policies

Good service provider | Easy booking process | Provides every information about flight | Saves time and money

User-friendly interface | Real-time update | Easy cancellation process | 24/7 customer support through chat and calls

Network

Seamless, fast, and error-free online | Mobile availablitly | Send automated tickets to the customer by mail | Highly customizable system

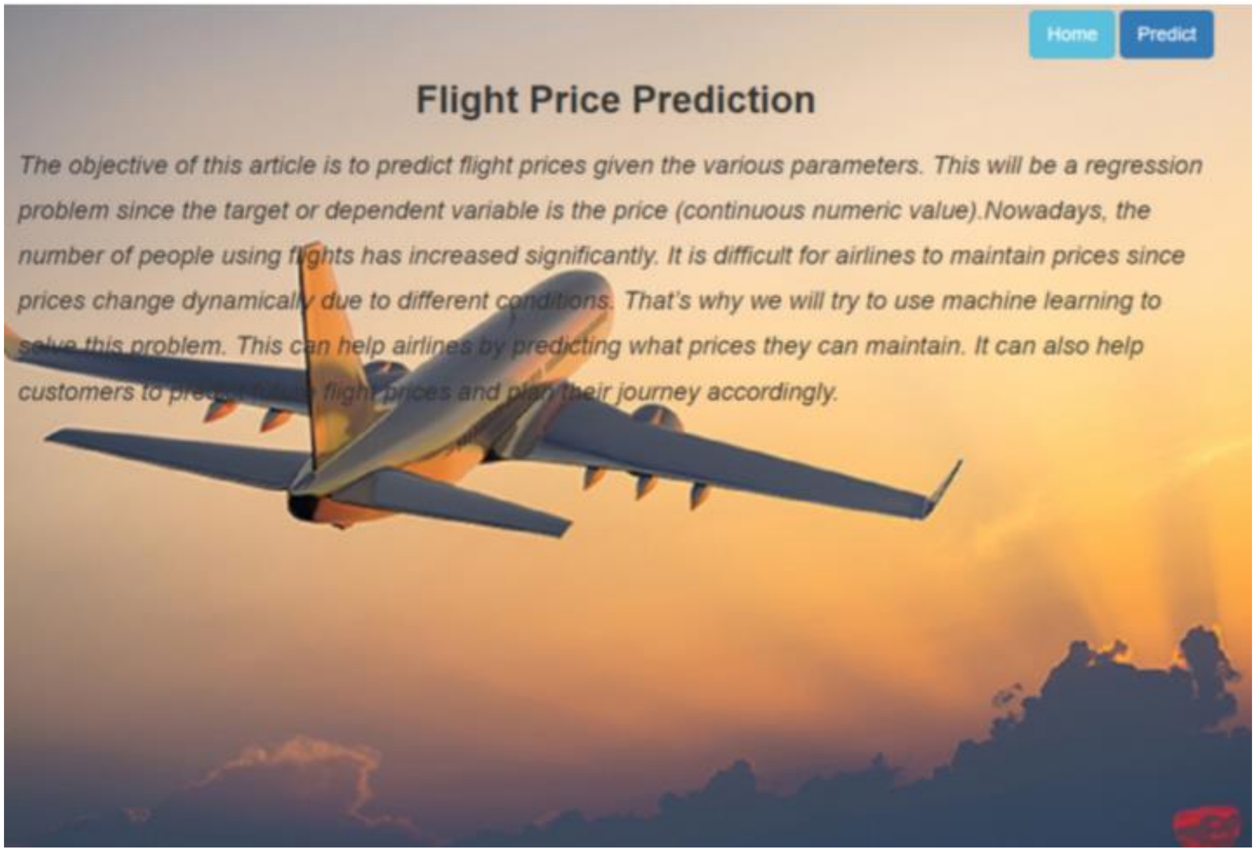
Prioritize

Importance

Feasibility

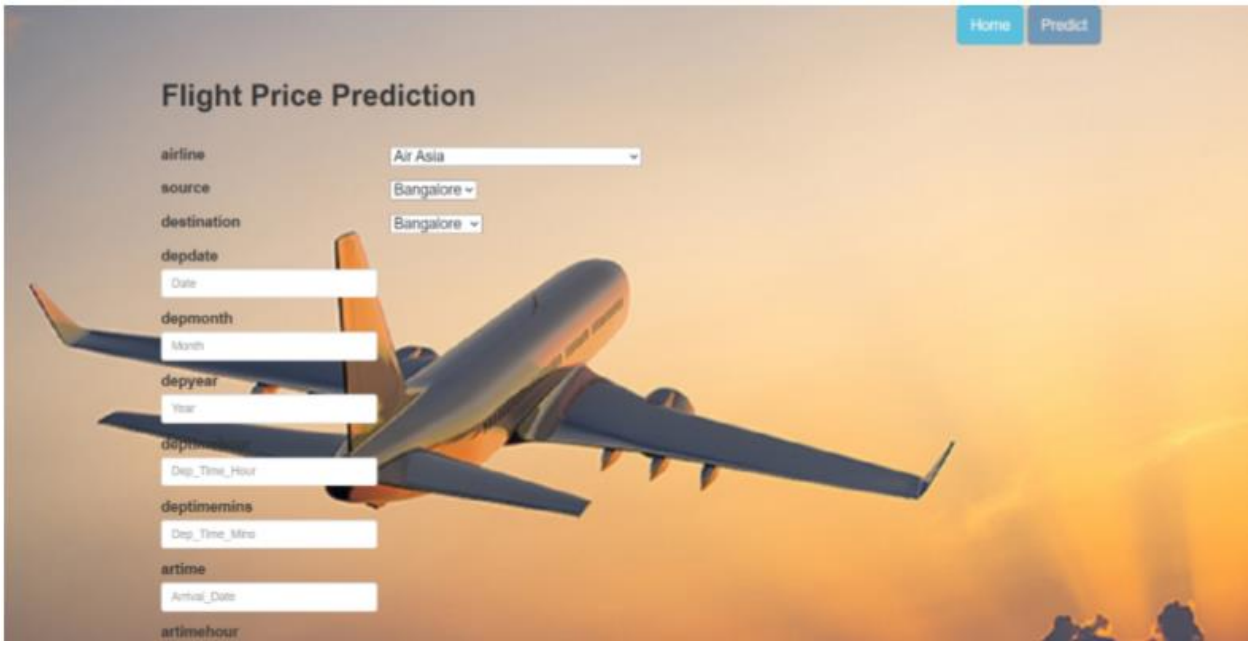
After you collaborate

3.RESULT



Flight Price Prediction

The objective of this article is to predict flight prices given the various parameters. This will be a regression problem since the target or dependent variable is the price (continuous numeric value).Nowadays, the number of people using flights has increased significantly. It is difficult for airlines to maintain prices since prices change dynamically due to different conditions. That's why we will try to use machine learning to solve this problem. This can help airlines by predicting what prices they can maintain. It can also help customers to predict future flight prices and plan their journey accordingly.



Flight Price Prediction

airline Air Asia

source Bangalore

destination Bangalore

depdate Date

depmonth Month

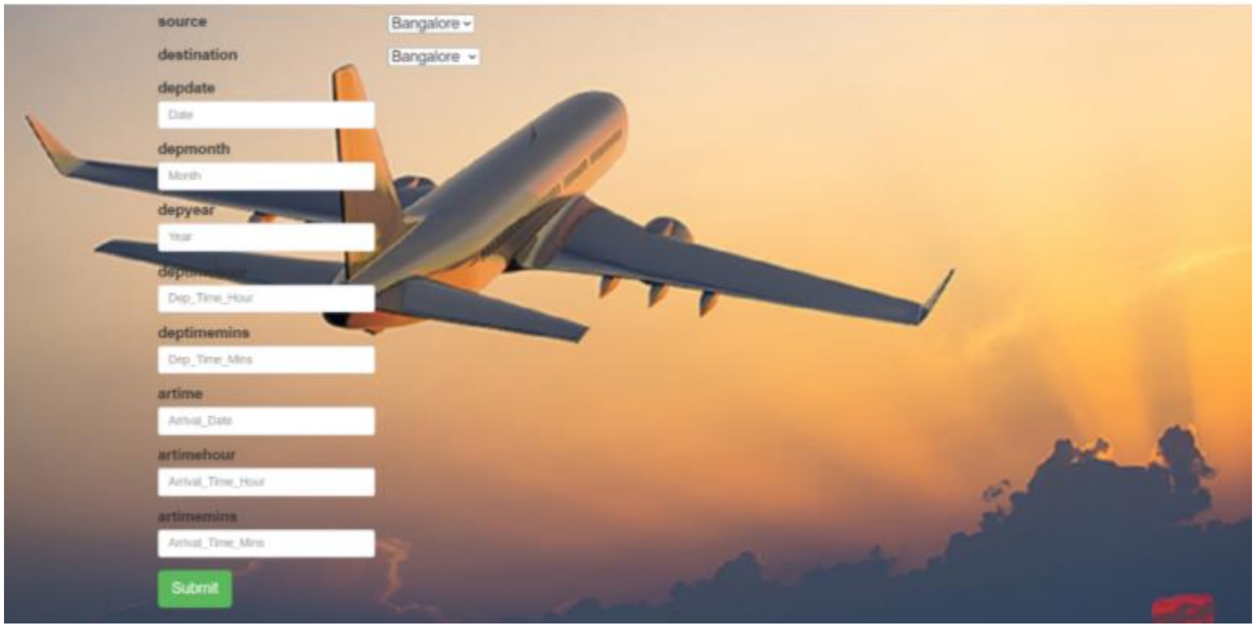
depyear Year

deptimehour Dep_Time_Hour

deptimemins Dep_Time_Mins

artime Arrival_Date

artimehour



source Bangalore

destination Bangalore

depdate

Date

depmonth

Month

depyear

Year

dep[illegible]

Dep_Time_Hour

deptimemins

Dep_Time_Mins

artime

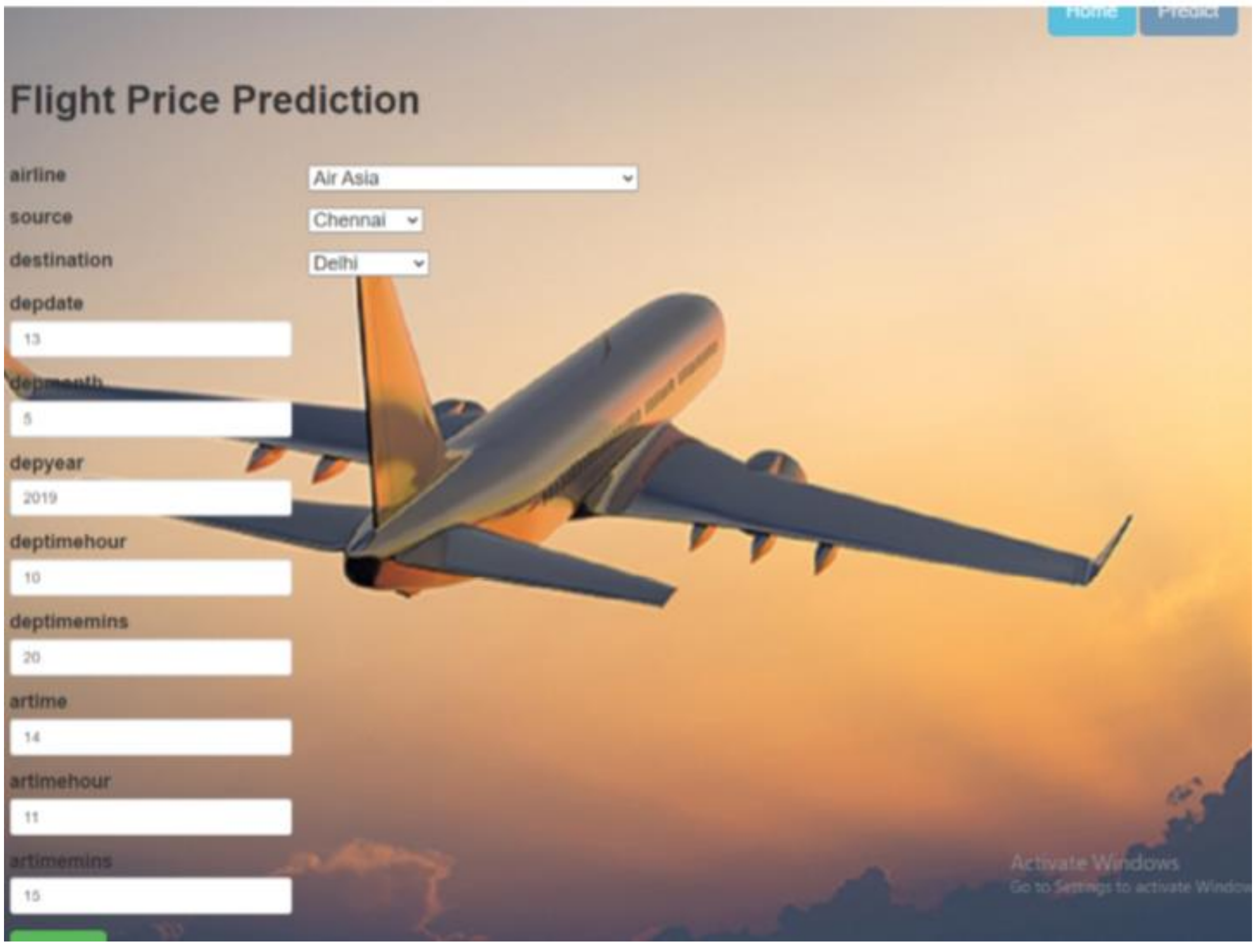
Arrival_Date

artimehour

Arrival_Time_Hour

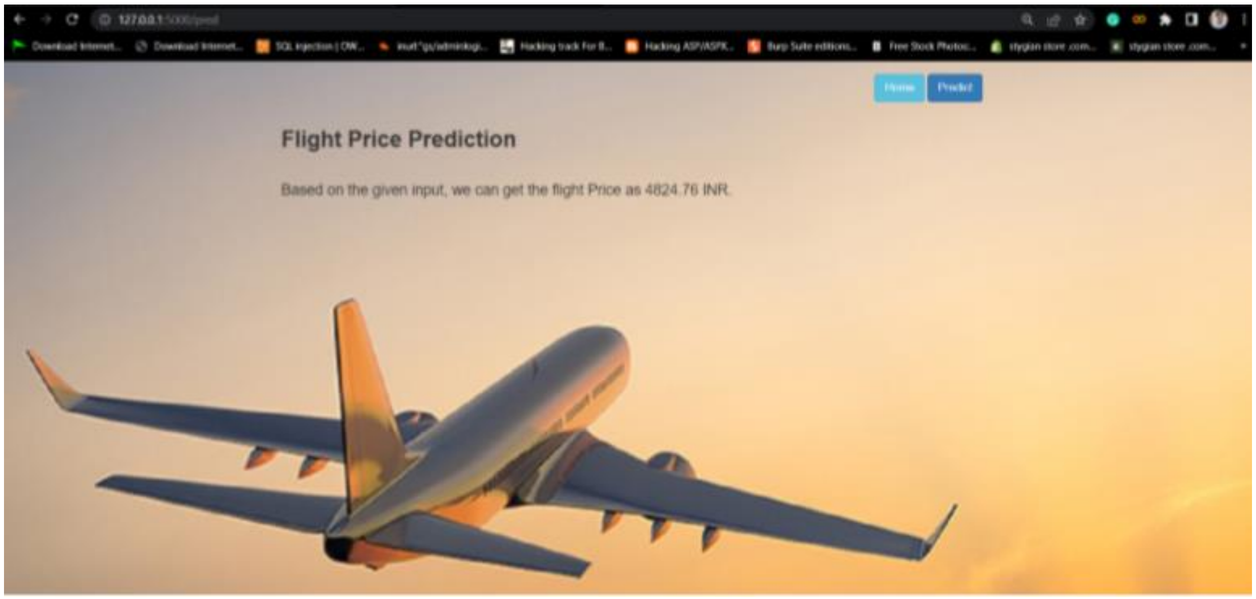
artimemins

Arrival_Time_Mins



Flight Price Prediction

airline	Air Asia
source	Chennai
destination	Delhi
depdate	13
depmonth	5
depyear	2019
deptimehour	10
deptimemins	20
artime	14
artimehour	11
artimemins	15



Flight Price Prediction

Based on the given input, we can get the flight Price as 4824.76 INR.

4. ADVANTAGES & DISADVANTAGES

ADVANTAGES

* One of the main advantages of using machine learning for flight price predictions is accuracy.
* These algorithms take into account various factors that affect flight prices, such as time of year, day of the week, and even weather conditions.
* Another advantage is convenience.
* Instead of spending hours searching for the best deals, travelers can simply input their desired travel dates and destinations into a machine learning algorithm and receive accurate predictions in seconds.

DISADVANTAGES

* While machine learning algorithms can provides accurate predictions, they are not foolproof.

*Flight prices can fluctuate rapidly due to unforeseen events such as natural disasters or political unrest.

*Additionally, some travelers may prefer to book their flight based on personal preferences rather than solely relying on machine learning predictions.

*For example, they may want to choose a specific airline or airport even if it means paying a higher price.

5. APPLICATION

Well, every customer looks for his ease, time-saving, and best deals at less expenditure. Let’s see what more a customer wants in his flight booking app.

- Users want their searching for jobs easier through the app. They want the entire detail of their flight in front of their eyes just by entering the source and destination.
- Easy and simple online payment.
- Get the best deal and that too at minimum spending.
- Catch up with direct flights and at the expected time.
- Easy cancellation with a quick refund.

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

* As technology advances, machine learning algorithms will become even more sophisticated in predicting flight prices.

* This will allow travelers to make even more informed decisions about when and where to book their flights.

* Further more, machine learning algorithms can also be used to personalize flight recommendations based on individual preferences and past travel history.

* This will provide a more personalized and seamless travel experience for passengers.

8. APPENDIX

A. Source code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import
RandomForestClassifier,GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import f1_score
from sklearn.metrics import
classification_report,confusion_matrix
import warnings
import pickle
from scipy import stats
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')


data=pd.read_csv("/content/Data_Train_1.csv")
data.head()

for i in Category:
    print(i,data[i].unique()


data.Date_of_journey=data.Date_of_Journey.str.split('/')
data.Date_of_Journey


data['Date']=data.Date_of_Journey.str[0]
data['Month']=data.Date_of_Journey.str[1]
data['Year']=data.Date_of_Journey.str[2]


data.Total_Stops.unique()


data.Route=data.Route.str.split('')
data.Route


data['City1']=data.Route.str[0]
data['City2']=data.Route.str[1]
data['City3']=data.Route.str[2]
data['City4']=data.Route.str[3]
data['City5']=data.Route.str[4]
data['City6']=data.Route.str[5]


data.Dep_Time=data.Dep_Time.str.split(':')

data['Dep_Time_Hour']=data.Dep_Time.str[0]
data['Dep_Time_Mins']=data.Dep_Time.str[1]

data.Arrival_Time=data.Arrival_Time.str.split('')
```

```
data['Arrival_date']=data.Arrival_Time.str[1]
data['Time_of_Arrival']=data.Arrival_Time.str[0]

data['Time_of_Arrival']=data.Time_of_Arrival.str.split(':')

data['Arrival_Time_Hour']=data.Time_of_Arrival.str[0]
data['Arrival_Time_Mins']=data.Time_of_Arrival.str[1]

data.Duration=data.Duration.str.split('')


data['Travel_Hours']=data.Duration.str[0]
data['Travel_Hours']=data['Travel_Hours'].str.split('h')
data['Travel_Hours']=data['Travel_Hours'].str[0]
data.Travel_Hours=data.Travel_Hours
data['Travel_Mins']=data.Duration.str[1]
data.Travel_Mins=data.Travel_Mins.str.split('m')
data.Travel_Mins=data.Travel_Mins.str[0]

data.Total_Stops.replace('non_stop',0,inplace=True)
data.Total_stops=data.Total_Stops.str.split('')
data.Total_Stops=data.Total_Stops.str[0]


data.Additional_Info.unique()


data.Additional_Info.replace('No Info','No info',inplace=True)

data.isnull().sum()
data.drop(['City4','City5','City6'],axis=1,inplace=True)

data.drop(['Date_of_Journey','Route','Dep_Time','Arrival_Time'
,'Duration'],axis=1,inplace=True)
data.drop(['Time_of_Arrival'],axis=1,inplace=True)


data.isnull().sum()


data['City3'].fillna('None',inplace=True)

data['Arrival_date'].fillna(data['Date'],inplace=True)

data['Travel_Mins'].fillna(0,inplace=True)

data.info()

data.Date=data.Date.astype('int64')
data.Month=data.Month.astype('int64')
data.Year=data.Year.astype('astype')
data.Dep_Time_Hour=data.Dep_Time_Hour.astype('astype')
data.Dep_Time_Hour=data.Dep_Time_Hour.astype('int64')
data.Dep_Time_Mins=data.Dep_Time_Mins.astype('int64')
data.Arrival_date=data.Arrival_date.astype("int64")
data.Arrival_Time_Hour=data.Arrival_Time_Hour.astype('int64
')
data.Arrival_Time_Mins=data.Arrival_Time_Mins.astype('int6
4')
```

```
data.Travel_Mins=data.Travel_Mins.astype('int64')


data[data['Travel_Hours']=='5m']

data.drop(index=6474,inplace=True,axis=0)

data.Travel_Hours=data.Travel_Hours.astype('int64')

categorical=['Airline','Source','Destination','Additional_Info','Cit
y1']
numerical=['Total_Stops','Date','Month','Year','Dep_Time_Hour'
,'Dep_Time_Mins','Arrival_Time_Hour','Arrival_Time_Mins','T
ravel_Hours','Travel_Mins']

from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()

data.Airline=le.fit_transform(data.Airline)
data.Source=le.fit_transform(data.Source)
data.Destination=le.fit_transform(data.Destination)
data.Total_Stops=le.fit_transform(data.Total_Stops)
data.City1   =le.fit_transform(data.City1   )
data.City2=le.fit_transform(data.City2)
data.City3=le.fit_transform(data.City3)
data.Additional_Info=le.fit_transform(data.Additional_Info)
data.head()

data.head()
```

```
data=data[['Airline','Source','Destination','Date','Month','Year','D
ep_Time_Mins','Arrival_date','Arrival_Time']]

data.head()

data.describe()

import seaborn as sns
c=1
plt.figure(figsize=(20,45))

for i in Categorical:
    plt.subplot(6,3,c)
    sns.countplot(data[i])
    plt.xtricks(rotation=90)
    plt.tight_layout(pad=3.0)
plt.show()

plt.figure(figsize=(15,8))
sns.distplot(data.Price)

sns.heatmap(data.corr(),annot=True)

import seaborn as sns
sns.boxplot(data['Price'])


y=data['Price']
x=data.drop(columns=['Price'],axis=1)
```

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()

x_Scaled=ss.fit_transform(x)

x_scaled=pd.DataFrame(x_scaled,columns=x.columns)
x_scaled.head()

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,ra
ndom_state=42)


x_train.head()

from sklearn.ensemble import
RandomForestRegressor,GradientBoostingRegressor,AdaBoost
Regressor
rfr=RandomForestRegressor()
gb=GradientBoostingRegressor()
ad=AdaBoostRegressor()

from sklearn.metrics import
r2_score,mean_absolute_error,mean_squared_error
for i in [rfr,gb,ad]:
    i.fit(x_train,y_train)
    y_pred=i.predict(x_test)
    test_score=r2_score(y_test,y_pred)
    train_score=r2_score(y_train,i.predict(x_train))
    if abs(train_score-test_score)<=0.2:
      print(i)
      print("R2 score is",r2_score(y_test,y_pred))
      print("R2 for train data",r2_score(y_train,i.predit(x_train)))
      print("Mean Absolute Error
is",mean_absolute_error(y_pred,y_test))
      print("Mean Squared Error
is",mean_squared_error(y_pred,y_test))
      print("Root Mean Squared Error
is",(mean_squared_error(y_pred,y_test,squared=False)))


from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import
r2_score,mean_absolute_error,mean_squared_error
Knn=KNeighborsRegressor()
svr=SVR()
dt=DecisionTreeRegressor()
for i in [Knn,svr,dt]:
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  test_score=r2_score(y_test,y_pred)
  train_score=r2_score(y_train,i.predict(x_train))
  if abs(train_score-test_score)<=0.1:
     print(i)
     print("R2 score is",r2_score(y_test,y_pred))
     print("R2 for train data",r2_score(y_train,i.predit(x_train)))
     print("Mean Absolute Error
is",mean_absolute_error(y_test,y_pred))
```

```
    print("Mean Squared Error
is",mean_squared_error(y_test,y_pred))
    print("Root Mean Squared Error
is",(mean_squared_error(y_test,y_pred,squared=False)))



from sklearn.model_selection import cross_val_score
for i in range(2,5):
    cv=cross_val_score(rfr,x,y,CV=i)
    print(rfr,cv.mean())



from sklearn.model_selection import RandomizedSearchCV

param_grid={'n_estimators':[10,30,5070,100],'max_depth':[Non
e,1,2,3],'max_features':['auto','sqrt']}
rfr=RandomForestRegressor()
rf_res=RandomizedSearchCV(estimator=rfr,param_distribution
s=param_grid,cv=3,verbose=2,n_jobs=-1)
rf_res.fit(x_train)


gb=GradientBoostingRegressor()
gb_res=RandomizedSearchCV(estimator=gb,param_distribution
s=param_grid,cv=33,verbose=2,n_jobs=-1,)

gb_res.fit(x_train,y_train)
```

```
rfr=RandomForestRegressor(n_estimators=10,max_features='sq
rt',max_depth=None)
rfr.fit(x_train,y_train)
y_train_pred=rfr.predict(x_train)
y_test_pred=rfr.predict(x_test)
print("train accuracy",r2_score(y_train_pred,y_train))
print("test accuracy",r2_score(y_test_pred,y_test))

Knn=KNeigborsRegressor(n_neighbors=2,algorithm='auto',met
ric_params=None,n_jobs=-1)
Knn.fit(x_train,y_train)
y_train_pred=Knn.predict(x_train)
y_test_pred=Knn.predict(x_test)
print("train accuracy",r2_score(y_train_pred,y_train))
print("test accuracy",r2_score(y_test_pred,y_test))


rfr=RandomForestRegressor(n_estimators=10,max_features='sq
rt',max_depth=None)
rfr.fit(x_train,y_train)
y_train_pred=rfr.predict(x_train)
y_test_pred=rfr.predict(x_test)
print("train accuracy",r2_score(y_train_pred,y_train))
print("test accuracy",r2_score(y_test_pred,y_test))


Price_list=pd.DataFrame({'Price':PriceS})
```

Price_list

```
import pickle
pickle.dump(rfr,open('model1.pk1','wb'))
```

```
import pickle
pickle.dump(rfr,open('model1.pk1','wb'))
```

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

model=Pickle.load(open(r"model1.pkl",'rb'))



def home1():
    return render_template('predict.html')




def predict():
    x=[[int(x) for x in request.form.values()]]
```

```
print(x)

x=np.array(x)
print(x.shape)

print(x)
pred=model1.predict(x)
print(pred)
return render_template('submit.html',prediction_test=pred)
```