FLIGHT DELAY PREDICTION FOR AVIATION INDUSTRY USING MACHINE LEARNING

TEAM MEMBER'S NAME : Priya.V
Priyadharshini.M
Sathya.M
Nivetha.G

CLASS : B.SC Computer Science

TABLE OF INDEX

1. INTRODUCTION

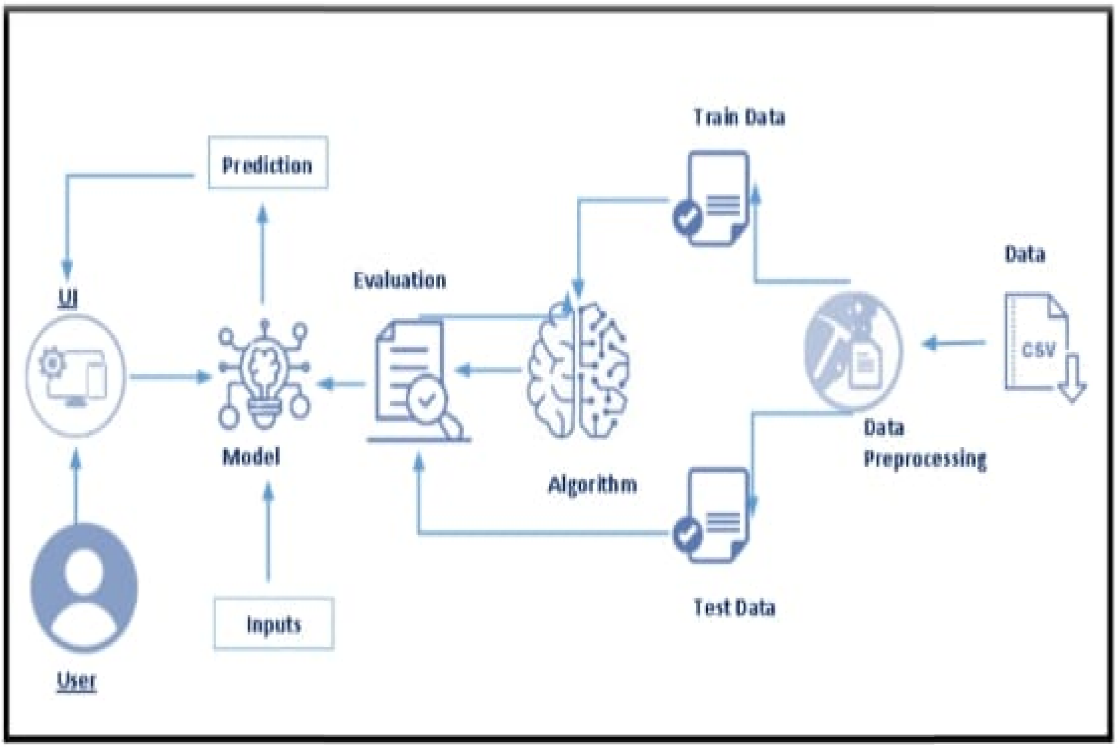
FLIGHT DELAY PREDICTION WITH MACHINE LEARNING:

Over the last twenty years, air travel has been increasinglypreferred among travelers, mainly because of its speed and insome cases comfort. This has led to phenomenal growth in airtraffic and on the ground. An increase in air traffic growth has alsoresulted in massive levels of aircraft delays on the ground and inthe air.

These delays are responsible for large economic andenvironmental losses. According to, taxi-out operations areresponsible for 4,000 tons of hydrocarbons, 8,000 tons of nitrogenoxides and 45,000 tons of carbon monoxide emissions in theUnited States in 2007. Moreover, the economic impact of flightdelays for domestic flights in the US is estimated to be more than$19 Billion per year to the airlines and over $41 Billion per year tothe national economy In response to growing concerns of fuelemissions and their negative impact on health, there is activeresearch in the aviation industry for finding techniques to predictflight delays accurately in order to optimize flight operations andminimize delays.

Using a machine learning model, we can predict flight arrivaldelays. The input to our algorithm is rows of feature vector likedeparture date, departure delay, distance between the twoairports, scheduled arrival time etc. We then use decision treeclassifier to predict if the flight arrival will be delayed or not. Aflight is delayed when difference between scheduled and actualarrival times is greater than 15 minutes

TECHNICAL AIRCHITECTURE:



1.1. OVERVIEW

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated
- Once model analyses the input the prediction is show
  Cased on the UI

To accomplish this, we have to complete all the activities listed below,

- Define Problem / Problem Understanding
  - Specify the business problem
  - Business requirements
  - Literature Survey
  - Social or Business Impact.
- Data Collection & Preparation
  - Collect the dataset
  - Data Preparation
- Exploratory Data Analysis
  - Descriptive statistical
  - Visual Analysis
- Model Building
  - Training the model in multiple algorithms
  - Testing the model

- Performance Testing & Hyperparameter Tuning

  - Testing model with multiple evaluation metrics
  - Comparing model accuracy before & after applying hyperparameter tuning

- Model Deployment

  - Save the best model
  - Integrate with Web Framework

- Project Demonstration & Documentation

  - Record explanation Video for project end to end solution
  - Project Documentation-Step by step project development procedure

1.2. PURPOSE

There are various reasons for flight delays. Sometimes its things the airlines can control, such as scheduling and staffing, and sometimes it’s due to random events, such as weather.

2. PROBLEM DEFINITION &DESIGN THINKING

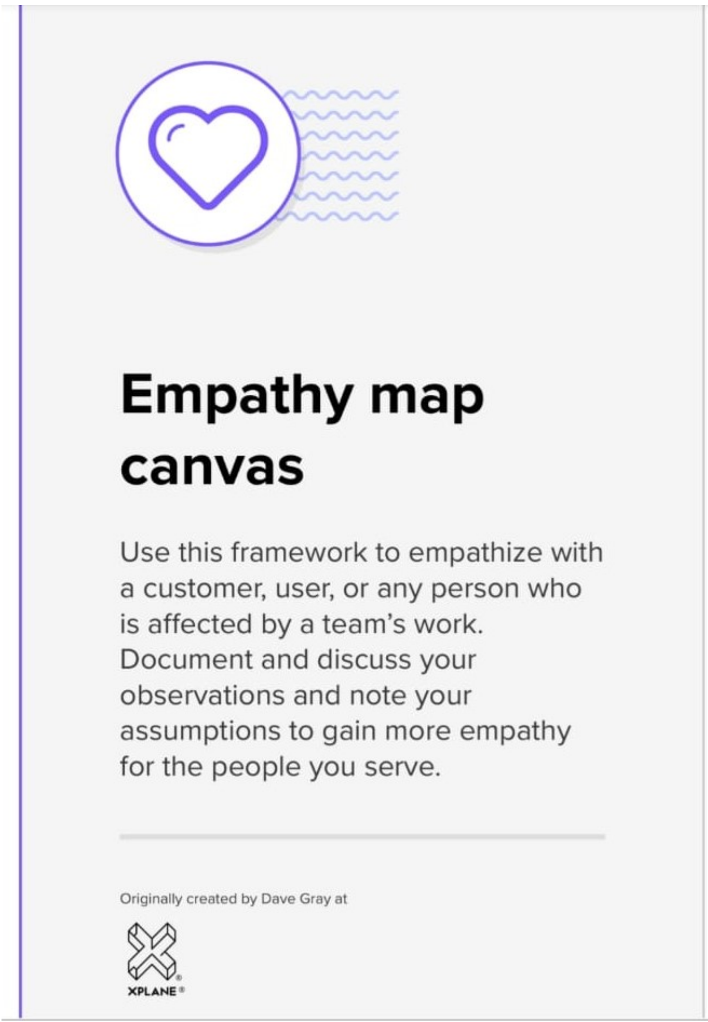
Over the last twenty years, air travel has been increasingly Preferred among travelers, mainly because of its speed and in some cases comfort. This has led to phenomenal growth in air traffic and on the ground. An increase in air traffic growth has also resulted in massive levels of aircraft delays on the ground and in the air. These delays are responsible for large economic and environmental losses. According to, taxi-out operations are responsible for 4,000 tons of hydrocarbons, 8,000 tons of nitrogen oxides and 45,000 tons of carbon monoxide emissions in the United States in 2007. Moreover, the economic impact of flight delays for domestic flights in the US is estimated to be more than $19 Billion per year to the airlines and over $41 Billion per year to the national economy In response to growing concerns of fuel emissions and their negative impact on health, there is active research in the aviation industry for finding techniques to predict flight delays accurately in order to optimize flight operations and minimize delays.

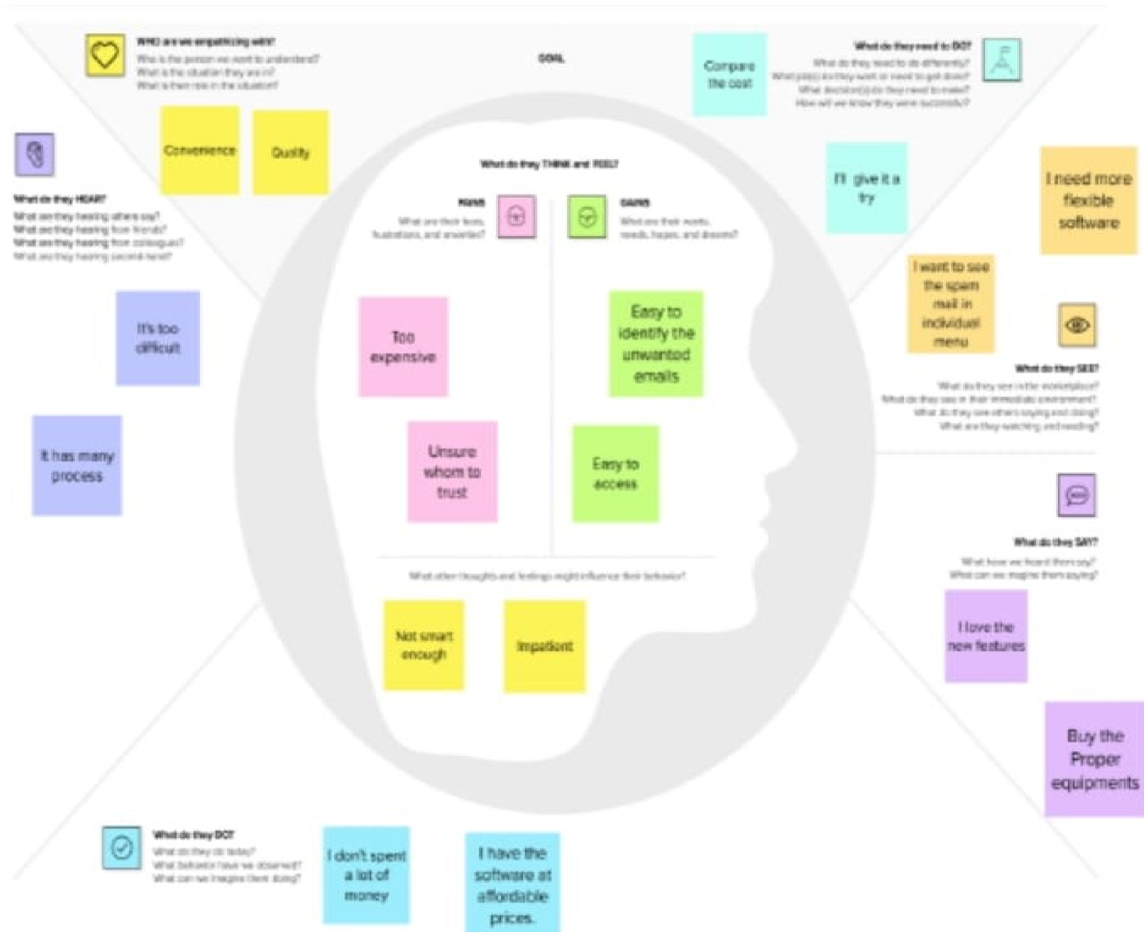
Using a machine learning model, we can predict flight arrival delays. The input to our algorithm is rows of feature vector like departure date, departure delay, distance between the two airports, scheduled arrival time etc. We then use decision tree classifier to predict if the flight arrival will be delayed or not. A flight is delayed when difference between scheduled and actual arrival times is greater than 15 minutes.

This is the initial step in building a machine learning model which aims to understand the need for it in the organization. The machine learning development process can be resource intensive, so clear objectives should be agreed and set at the start. Clearly define the problem that a model needs to solve and what success looks like. A deployed model will bring much more value if it’s fully aligned with the objectives of the organization. Before the project begins, there are key elements that need to be explored and planned

2.1 EMPATHY MAP

In the ideation phase we have empathized as our client Flight Delay Prediction with machine learning and we have acquired the details which are represented in the Empathy Map given below

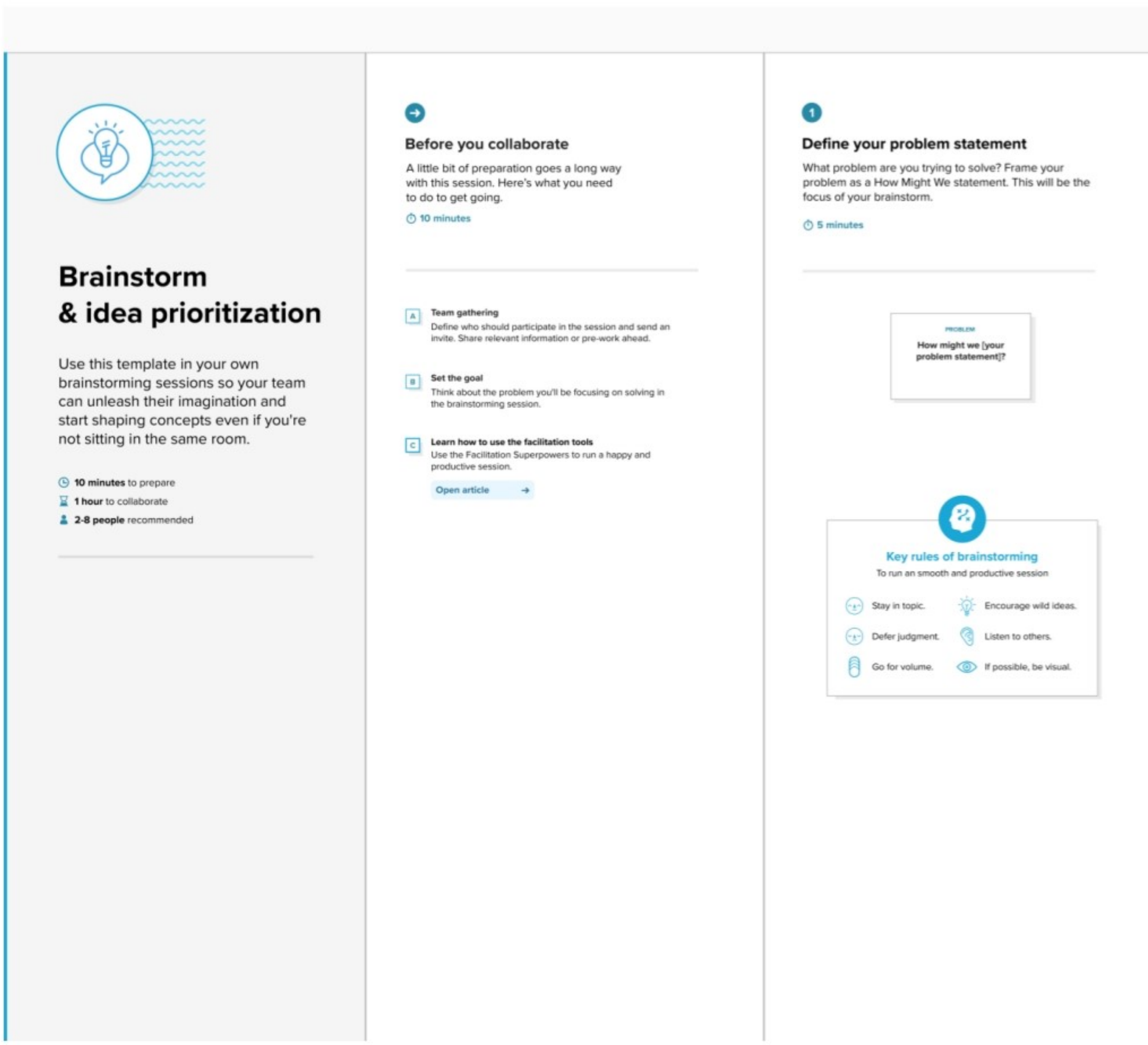



GOAL
Compare the cost
Convenience
Quality
What do they THINK and FEEL?
I'll give it a try
I need more flexible software
I want to see the spam mail in individual menu
It's too difficult
Too expensive
Easy to identify the unwanted emails
It has many process
Unsure whom to trust
Easy to access
Not smart enough
Impatient
I love the new features
Buy the Proper equipments
I don't spent a lot of money
I have the software at affordable prices.

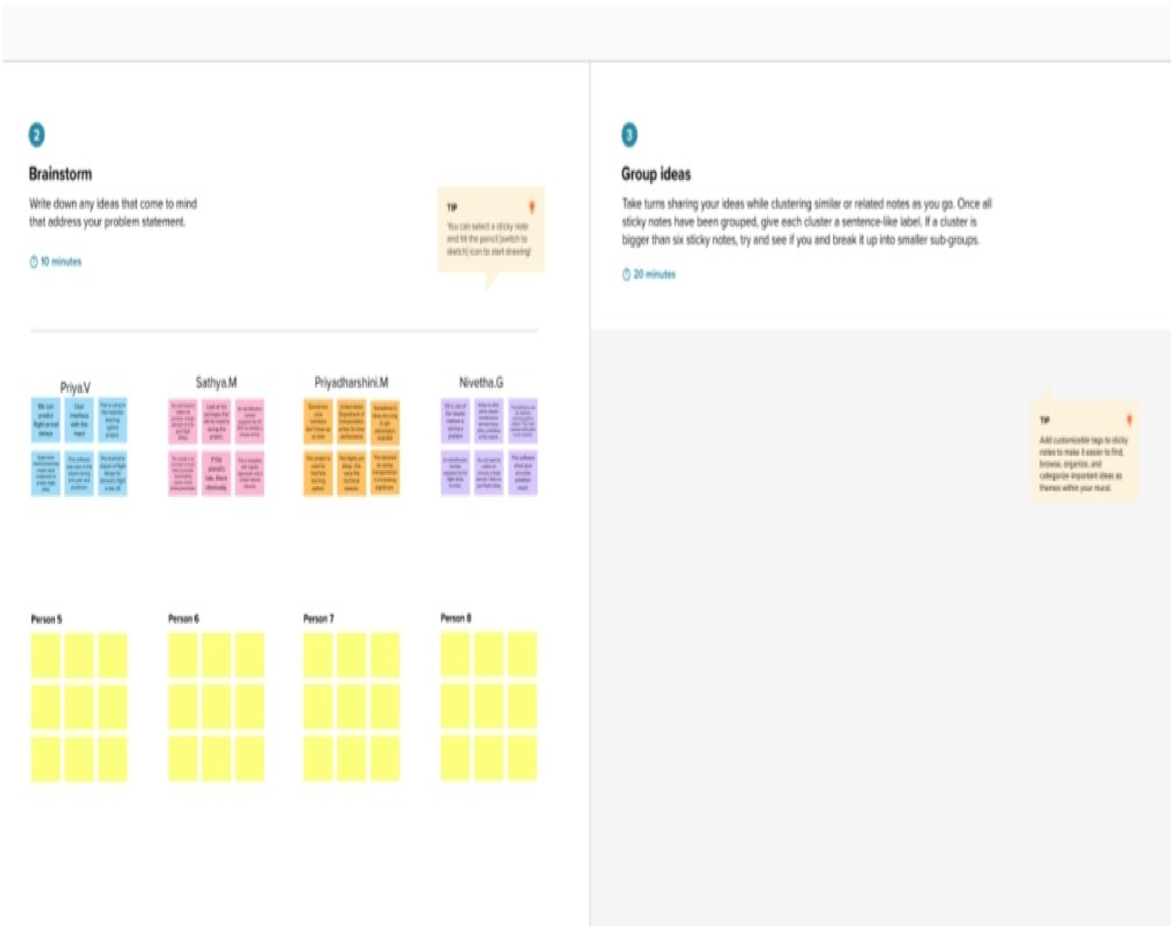
2.2 IDEATION &BRAINSTORMING MAP

Under this activity our team members have gathered and discussed various ideas to solve our project problem. Each member contributed 6 to 10 ideas after gathering all ides we have assessed the impact and feasibility of each point. Finally, we have assigned the priority for each point base on the impact value.

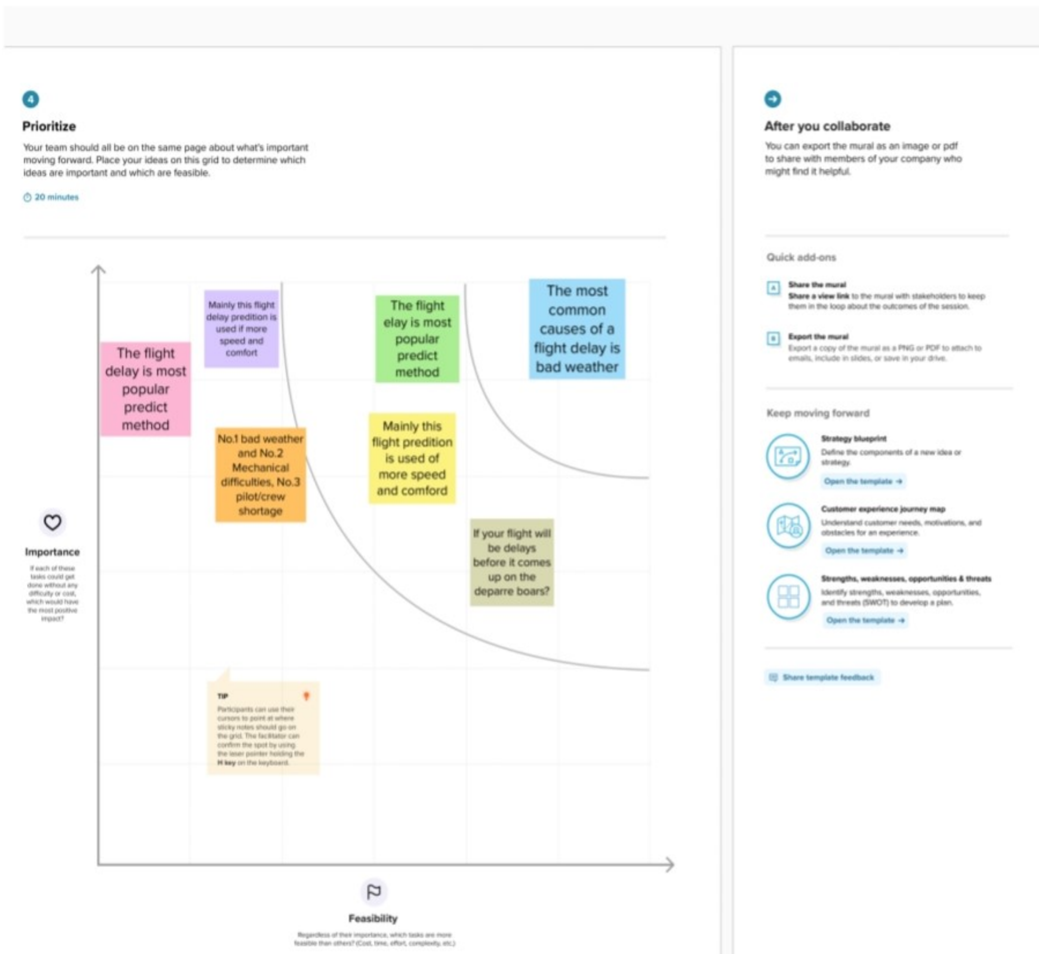
STEP 1: Team Gathering, collaboration and Select the Problem



STEP 2: Brainstorm, Idea Listing and Grouping



STEP 3: Idea Prioritization



3. RESULT

Read the datasets

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	UNIQUE_CARRIER	TAIL_NUM	FL_NUM	ORIGIN_AIRPORT_ID	ORIGIN	...
0	2016	1	1	1	5	DL	N836DN	1399	10397	ATL	...
1	2016	1	1	1	5	DL	N964DN	1476	11433	DTW	...
2	2016	1	1	1	5	DL	N813DN	1597	10397	ATL	...
3	2016	1	1	1	5	DL	N587NW	1768	14747	SEA	...
4	2016	1	1	1	5	DL	N836DN	1823	14747	SEA	...

5 rows × 26 columns

CRS_ARR_TIME	ARR_TIME	ARR_DELAY	ARR_DEL15	CANCELLED	DIVERTED	CRS_ELAPSED_TIME	ACTUAL_ELAPSED_TIME	DISTANCE	Unnamed: 25
2143	2102.0	-41.0	0.0	0.0	0.0	338.0	295.0	2182.0	NaN
1435	1439.0	4.0	0.0	0.0	0.0	110.0	115.0	528.0	NaN
1215	1142.0	-33.0	0.0	0.0	0.0	335.0	300.0	2182.0	NaN
1335	1345.0	10.0	0.0	0.0	0.0	196.0	205.0	1399.0	NaN
607	615.0	8.0	0.0	0.0	0.0	247.0	259.0	1927.0	NaN

Handling missing values

```
<class'panas.core.frame.DataFrame'>
RangeIndex: 11231 entries, 0 to 11230
Data columns (total 26 columns):
```

```
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   YEAR                  11231 non-null   int64
 1   QUARTER               11231 non-null   int64
 2   MONTH                 11231 non-null   int64
 3   DAY_OF_MONTH          11231 non-null   int64
 4   DAY_OF_WEEK           11231 non-null   int64
 5   UNIQUE_CARRIER        11231 non-null   object
 6   TAIL_NUM              11231 non-null   object
 7   FL_NUM                11231 non-null   int64
 8   ORIGIN_AIRPORT_ID     11231 non-null   int64
 9   ORIGIN                11231 non-null   object
 10  DEST_AIRPORT_ID       11231 non-null   int64
 11  DEST                  11231 non-null   object
 12  CRS_DEP_TIME          11231 non-null   int64
 13  DEP_TIME              11124 non-null   float64
 14  DEP_DELAY             11124 non-null   float64
 15  DEP_DEL15             11124 non-null   float64
 16  CRS_ARR_TIME          11231 non-null   int64
 17  ARR_TIME              11116 non-null   float64
 18  ARR_DELAY             11043 non-null   float64
 19  ARR_DEL15             11043 non-null   float64
 20  CANCELLED             11231 non-null   float64
 21  DIVERTED              11231 non-null   float64
 22  CRS_ELAPSED_TIME      11231 non-null   float64
 23  ACTUAL_ELAPSED_TIME   11043 non-null   float64
 24  DISTANCE              11231 non-null   float64
 25  Unnamed: 25           0 non-null       float64
dtypes: float64(12), int64(10), object(4)
memory usage: 2.2+ MB
```

```
YEAR                    0
QUARTER                 0
MONTH                   0
DAY_OF_MONTH            0
DAY_OF_WEEK             0
UNIQUE_CARRIER          0
TAIL_NUM                0
FL_NUM                  0
ORIGIN_AIRPORT_ID       0
ORIGIN                  0
DEST_AIRPORT_ID         0
DEST                    0
CRS_DEP_TIME            0
DEP_TIME              107
DEP_DELAY             107
DEP_DEL15             107
CRS_ARR_TIME            0
ARR_TIME              115
ARR_DELAY             188
ARR_DEL15             188
CANCELLED               0
DIVERTED                0
CRS_ELAPSEDTIM          0
ACTUAL_ELAPSD_TIME    188
DISTANCE                0
dtype: int64
```

Exploratory Data Analysis

Descriptive statistical

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15
0	1399	1	1	5	0	4	21	0.0	0.0
1	1476	1	1	5	1	3	14	0.0	0.0
2	1597	1	1	5	0	4	12	0.0	0.0
3	1768	1	1	5	4	3	13	0.0	0.0
4	1823	1	1	5	4	1	6	0.0	0.0

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15	ORIGIN_0	ORIGIN_1	ORIGIN_2	ORIGIN_3
0	1399	1	1	5	21	0.0	0.0	1	0	0	0
1	1476	1	1	5	14	0.0	0.0	0	1	0	0
2	1597	1	1	5	12	0.0	0.0	1	0	0	0
3	1768	1	1	5	13	0.0	0.0	0	0	0	0
4	1823	1	1	5	6	0.0	0.0	0	0	0	0

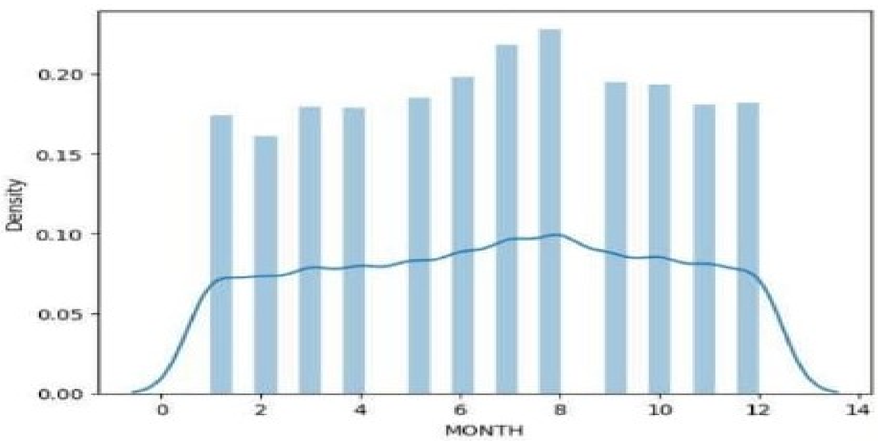
ORIGIN_0	ORIGIN_1	ORIGIN_2	ORIGIN_3	ORIGIN_4	DEST_0	DEST_1	DEST_2	DEST_3	DEST_4
1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	1	0	0	0

Visual Analysis

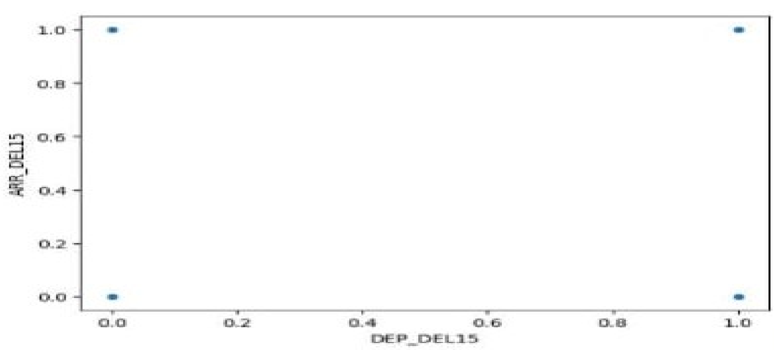
```
<ipython-input-23-43f5c122a6ef>:1: UserWarning:

`distplot` is a deprecated function and will be
removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a
figure-level function with
similar flexibility) or `histplot` (an axes-level
function for histograms).
```

For a guide to updating your code to use the new functions, please see https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

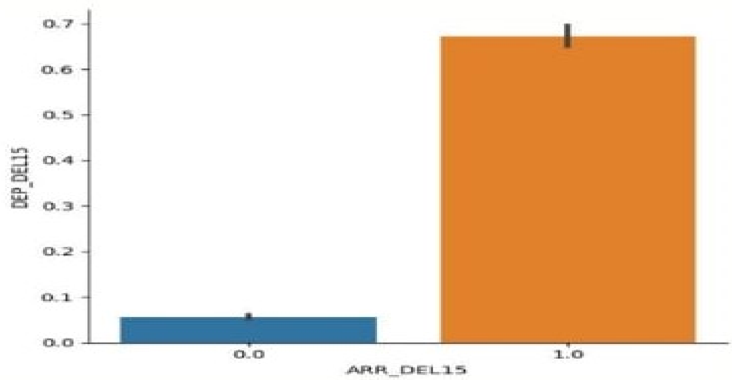
```
sns.distplot(dataset.MONTH)
<Axes: xlabel='MONTH', ylabel='Density'>
```



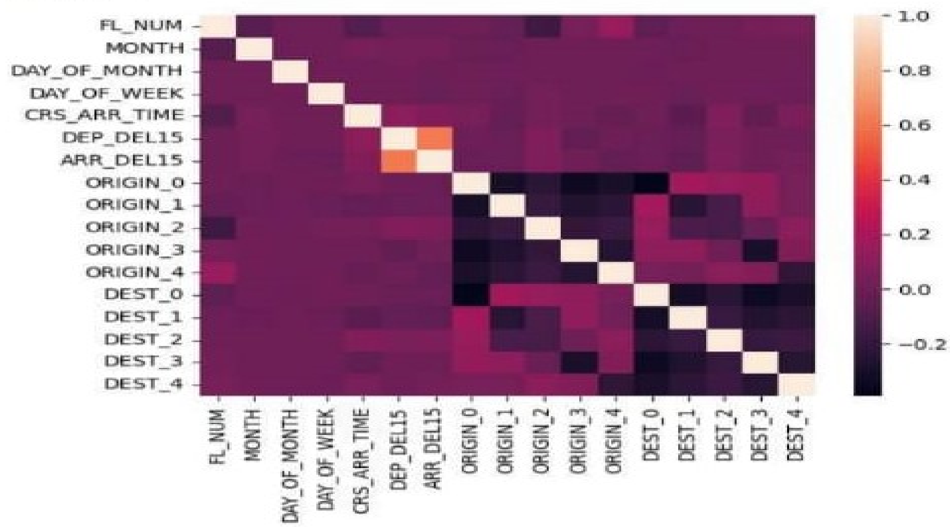

```
<Axes: xlabel='DEP_DEL15', ylabel='ARR_DEL15'>
```



```
<seaborn.axisgrid.FacetGrid at 0x7fa8785f7190>
```




```
<Axes: >
```



MODEL BULDING

Decision Tree Model

```
☑ DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

Random Forest Model

```
<ipython-input-40-b87bb2ba9825>:1:
DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
rfc.fit(x_train,y_train)
```

```
☑ RandomForestClassifier
RandomForestClassifier(criterion='entropy',
n_estimators=10)
```

ANN Model

```
Epoch 1/100
1797/1797 [==============================] - 7s 3ms/step - loss: 0.4355 -
Epoch 99/100
1797/1797 [==============================] - 5s 3ms/step - loss: 0.0964 -
accuracy: 0.9578 - val_loss: 0.5974 - val_accuracy: 0.8765
Epoch 100/100
1797/1797 [==============================] - 4s 2ms/step - loss: 0.1136 -
accuracy: 0.9539 - val_loss: 0.4560 - val_accuracy: 0.8742
<keras.callbacks.History at 0x7fa8205aceb0>
```

Integrate with web Frame Work

Building HTML pages

```
<!DOCTYPE html>

<html>

<head>

<title> Flight Delay Prediction</title>

</head>

<body background="flight image.jpg" style="background-repeat:no-repeat; background-size:100% 100%" text='black'>

<h1>

<b>

<i>

        Flight Delay Prediction

</i>

</b>

</h1>

<h2> Enter the details to check whether Flight Delay or not!</h2>

<h4>

<form action="/getdata" method="post">

<table>

<tr>

<td>year<td>:   <input type='text' name='year' placeholder='Enter YEAR' Enter Numerical part required='required'/><br>
```

```
</tr>

<tr>

<td>quarter<td>:   <input type='text' name='quarter' placeholder='Enter QUARTER' Enter 0 for Male 1 for Female required='required' /><br>

</tr>

<tr>

<td>month<td>:   <input type='text' name='month' placeholder='Enter 0 for no 1 for yes' required='required'/><br>

</tr>

<tr>

<td>dayofmonth<td>:   <input type='text' name='dayofmonth' placeholder='mcg/L' required='required' /><br>

</tr>

<tr>

<td>dayofweek<td>:   <input type='text' name='dayofweek' placeholder='Enter 0 for no 1 for yes' required='required' /><br>

</tr>

<tr>

<td>uniquecarrier<td>:   <input type='text' name='uniquecarrier' placeholder='UNIQUE_CARRIER' required='required' /><br>

</tr>

<tr>

<td>tailnum<td>:   <input type='text' name='tailnum' placeholder='TAIL_NUM' required='required' /><br>

</tr>

<tr>
```

```
<td>flnum<td>:   <input type='text' name='flnum' placeholder='FL_NUM'
required='required'/><br>

</tr>

<tr>

<td>originairport<td>:   <input type='text' name='originairport'
placeholder='ORIGIN_AIRPORT' required='required' /><br>

</tr>

<tr>

<td>origin<td>:   <input type='text' name='origin' placeholder='ORIGIN'
required='required' /><br>

</tr>

<tr>

<td>destairport<td>:   <input type='text' name='destairport'
placeholder='DEST_AIRPORT' required='required' /><br>

</tr>

<tr>

<td>dest<td>:   <input type='text' name='dest' placeholder='DEST'
required='required' /><br>

</tr>

<tr>

<td><br><br>            &n
bsp  

<button type="submit" class="btnbtn-primary btn-block btn-large"> Predict </button>

</td>

</tr>

</table>
```

</form>

</h4

<h2>

{{ prediction_text }}

</h2>

</body>

</html>

Building Python Code

```
import flask

from flask import Flask, render_template, request

import pickle

import numpy as np

import sklearn


app = Flask(__name__)


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



@app.route('/')

def home():

    return render_template('index.html')



@app.route('/getdata', methods=['POST'])

def pred():

    year = request.form['YEAR']

    print(year)

    quarter = request.form['QUARTER']

print(year, quarter)

    month= request.form['MONTH']
```

```
print(year, quarter, month)
dayofmonth = request.form['DAY_OF_MONTH']
print(year, quarter, month, dayofmonth)
dayofweek = request.form['DAY_OF_WEEK']
print(year, quarter, month, dayofmonth, dayofweek)
uniquecarrier = request.form['UNIQUE_CARRIER']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier)
tailnum = request.form['TAIL_NUM']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum)
flnum = request.form['FL_NUM']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum, flnum)
originairport = request.form['ORIGIN_AIRPORT']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum,
flnum, originairport)
  origin = request.form['ORIGIN']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum,
flnum, originairport, origin)
destairport = request.form['DEST_AIRPORT']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum,
flnum, originairport, origin, destairport)
dest = request.form['DEST']
print(year, quarter, month, dayofmonth, dayofweek, uniquecarrier, tailnum,
flnum, originairport, origin, destairport, dest)
```

```
inp_features = [[np.log(float(year)), int(quarter), int(month), np.log(float(dayofmonth)),
int(dayofweek),
                int(uniquecarrier),
                int(tailnum),
                int(flnum), int(originairport), int(origin), int(destairport),
                np.log(float(dest))]]
    print(inp_features)
    prediction = model.predict(inp_features)
    print(type(prediction))
    t = prediction[0]
    print(t)
    if t > 0.5:
prediction_text = 'Chance of delay'
    else:
prediction_text = 'No chance of delay'
    print(prediction_text)
    return render_template('prediction.html', prediction_results=prediction_text)



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

Run the Web Application

4. ADVANTAGES & DISADVANTAGES

Advantages:

- Fastest mode of Transport.
- Cheaper infrastructure than rail or road.
- Air travel is strategically significant.
- Free from geographic barriers like hills and lakes.
- Useful during natural disasters.

Disadvantages:

- High Operational Costs.
- Air transport accidents are usually fatal.
- Huge Initial Investments.
- Difficult to carry very large cargo.
- Affected by bad weather.

5. APPLICATIONS

- Airlines can determine efficient routes with minimum delay possibility.
- Opt for secondary airports for particular routes between cities.
- This model can help passengers to plan layover at particular airport.

EXAMPLE:

- SEA-LGA instead of SEA-JFK since SEA-JFK flight are more likely to be delayed

6. CONCLUSION

Over the last twenty years, air travel has been increasingly preferred Among travelers, mainly because of its speed and in some case 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.

We compare decision tree classifier with logistic regression and a simple neural network for various figures of merit. Finally, it will be Integrated to web based application we have develop the machine learning model using python programming language on the reportsare show above.

7. FUTURE SCOPE

- Further supportive study is required to correlate all the problem.
- Scope and method for getting most accurate result.
- Although weather conditions are the major reasons events such as major calamites natural or man-mode can cause major delay in flight.
- The model accuracy can be increased by taking into the account variables like weather conditions and airline employee efficiency.

8. APPENDIX

SOURCE CODE:

Importing the libraries:

```
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier,RandomForestCl
assifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RandomizedSearchCV
import imblearn
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,classification_report,confu
sion_matrix,f1_score
```

Read the Dataset:

```
dataset=pd.read_csv("/content/flightdata.csv")
dataset.head()
```

Handling missing values:

```
dataset.info()


dataset = dataset.drop('Unnamed: 25', axis=1)
dataset.isnull().sum()

dataset = dataset[["FL_NUM","MONTH","DAY_OF_MONTH","DAY_OF_WEEK","ORIG
IN","DEST","CRS_ARR_TIME","DEP_DEL15","ARR_DEL15"]]
dataset.isnull().sum()

dataset[dataset.isnull().any(axis=1)].head(10)
dataset['DEP_DEL15'].mode.()
dataset = dataset.fillna({'ARR_DEL15':1})
```

```
dataset = dataset.fillna({"DEP_DEL15":0})
dataset.iloc[177:185]
```

Handling categorical values:

```
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.head(5)

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


from sklearn.preprocessing import OneHotEncoder
oh = OneHotEncoder()
z=oh.fit_transform(x[:,4:5]).toarray()
t=oh.fit_transform(x[:,5:6]).toarray()


z

t


x=np.delete(x,[4,5],axis=1)
```

Exploratory Data Analysis

Descriptive statistical

```
dataset.describe()
```

Univariate analysis:

```
sns.distplot(dataset.MONTH)
```

Bivarite analysis:

```
sns.scatterplot(x='DEP_DEL15',y='ARR_DEL15',data=dataset)
sns.catplot(x="ARR_DEL15",y="DEP_DEL15",kind='bar',data=dataset)
```

Multivariate analysis:

```
sns.heatmap(dataset.corr())
```

Splitting data into train and test:

```
x = dataset.iloc[:,0:8].values
y = dataset.iloc[:,8:9].values

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_sta
te=0)

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_sta
te=0)

x_test.shape

x_train.shape

y_test.shape

y_train.shape
```

Scaling the Data:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

Model Building:

Training the model in multiple algorithms

Decision tree model:

```
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)
```

Random forest model:

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=10,criterion='entropy')

rfc.fit(x_train,y_train)

y_predict = rfc.predict(x_test)
```

ANN model:

```
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

classification = Sequential()
classification.add(Dense(30,activation='relu'))
classification.add(Dense(128,activation='relu'))
classification.add(Dense(64,activation='relu'))
classification.add(Dense(32,activation='relu'))
classification.add(Dense(1,activation='sigmoid'))

classification.compile(optimizer='adam',loss='binary_crossentropy',met
rics=['accuracy'])

classification.fit(x_trainc,y_train,batch_size=4,validation_split=0.2,
epochs=100)
```

Test the model:

```
y_pred = classifier.predict([[129,99,1,0,0,1,0,1]])
print(y_pred)
(y_pred)

y_pred = rfc.predict([[129,99,1,0,0,1,0,1]])
print(y_pred)
(y_pred)

classification.save('flight.h5')

y_pred = classification.predict(x_test)

y_pred

y_pred = (y_pred > 0.5)
y_pred

def predict_exit(sample_value):
  sample_value = np.array(sample_value)
  sample_value = sample_value.reshape(1, -1)
  sample_value = sc.transform(sample_value)
  return classifier.predict(sample_value)

test=classification.predict([[1,1,121.000000,36.0,0,0,1,0]])
if test==1:
  print('Prediction:Chance of delay')
else:
  print('Prediction:No chance of delay.')
```

Testing model with multiple evaluation metrics

Compare the model:

```
crom sklearn import model_selection

from sklearn.neural_network import MLPClassifier

dfs=[]
models = [
          ('RF',RandomForestClassifier()),
          ('DecisionTree',DecisionTreeClassifier()),
          ('ANN',MLPClassifier())
        ]
```

```
results = []
    names = []
    scoring = ['accuracy','precision_weighted','recall_weighted','f1_w
eighted','roc_auc']
    target_names = ['no delay', 'delay']
    for name, model in models:
        kfold = model_selection.KFold(n_splits=5, shuffle=True, random
_state=90210)
        cv_results = model_selection.cross_validate(model, x_train, y_
train, cv=kfold, scoring=scoring)
        clf = model.fit(x_train,y_train)
        y_pred = clf.predict(x_test)
        print(name)
        print(classification_report(y_test, y_pred, target_names=targe
t_names))
        results.append(cv_results)
        names.append(name)
        this_df = pd.DataFrame(cv_results)
        this_df['model'] = name
        dfs.append(this_df)
final = pd.concat(dfs, ignore_index=True)



from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_predict)
cm

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



from sklearn.metrics import accuracy_score,classification_report
score = accuracy_score(y_pred,y_test)
print('The accuracy for ANN model is: {}%'.format(score*100))



from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

comparing model accuracy before &after applying hyperparmeter tuning:

```
parameters = {
              'n_estimators' :[1,20,30,55,68,74,90,120,115],
              'criterion':['gini','entropy'],
              'max_features':["auto", "sqrt", "log2"],
          'max_depth' :[2,5,8,10], 'verbose' :[1,2,3,4,6,8,9,10]
}

RCV = RandomizedSearchCV(estimator=rfc,param_distributions=parameters,
cv=10,n_iter=4)

RCV.fit(x_train,y_train)

RCV.fit(x_train,y_train)

bt_params

bt_score

model = RandomForestClassifier(verbose= 10, n_estimators= 120, max_fea
tures='log2',max_depth= 10,criterion= 'entropy')
RCV.fit(x_train,y_train)

y_predict_rfc = RCV.predict(x_test)
```