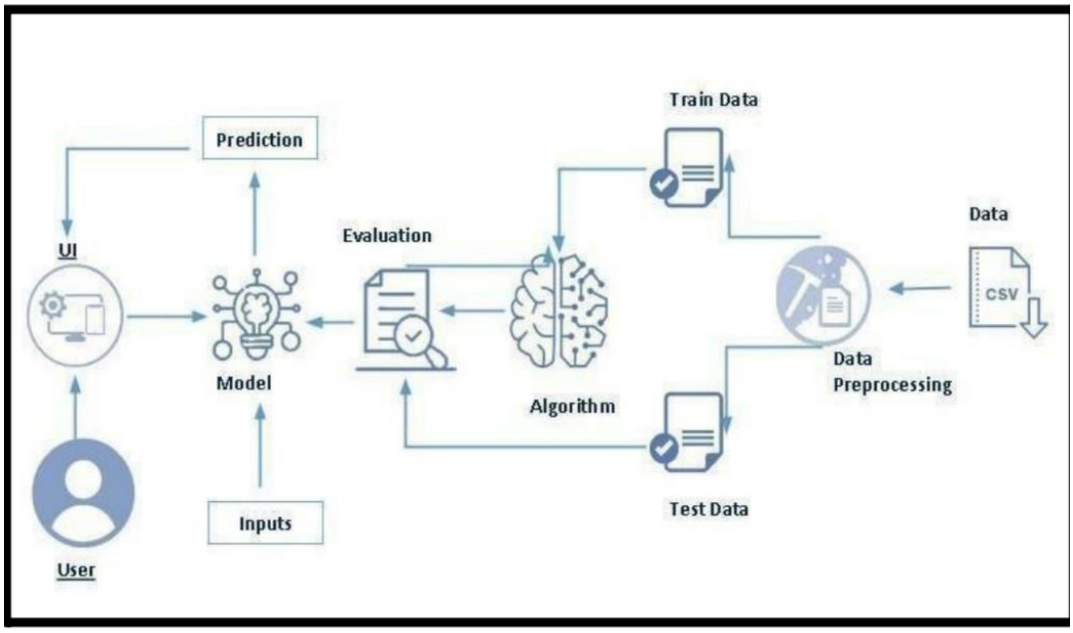
Flight Delay Prediction for aviation Industry using Machine Learning

Overview:

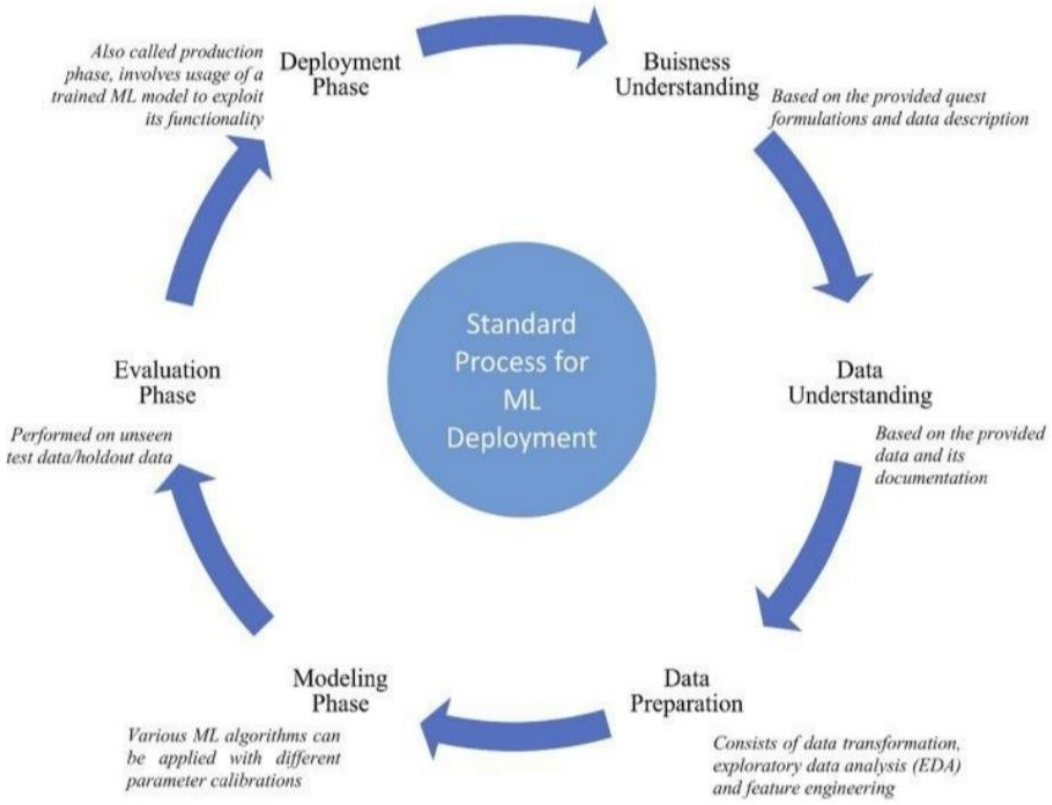
OVER the last twenty years, air travel has been increasingly preferred among travelers, mainly because of its speed and in some cases comfort. This has led to phenomenal growth in air traffic and on the ground. An increase in air traffic growth has also resulted in massive levels of aircraft delays on 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.

Technical Architecture:



Technical about the Project:

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. Furthermore, 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



A Project Description:

To predict flight delays using machine learning, you will need to collect and process a large amount of data on past flight delays. This data should include information such as the flight's departure and arrival times, the airline, the

aircraft type, and the weather conditions at the departure and arrival airports. Once you have collected and cleaned the data, you can use a variety of machine learning techniques such as regression, decision trees, or neural networks to train a model that can predict flight delays based on this data. It is important to note that flight delay prediction is a highly complex task and requires a lot of data. The literature suggests that ML models, specifically decision tree, ANN and random forest models, have been used to predict flight delays with varying degrees of accuracy. Commonly used features include historical flight data, weather conditions, and airport operations. It also shows that a combination of data mining techniques can be used to identify the factors that contribute to flight delays.

Project Flow:

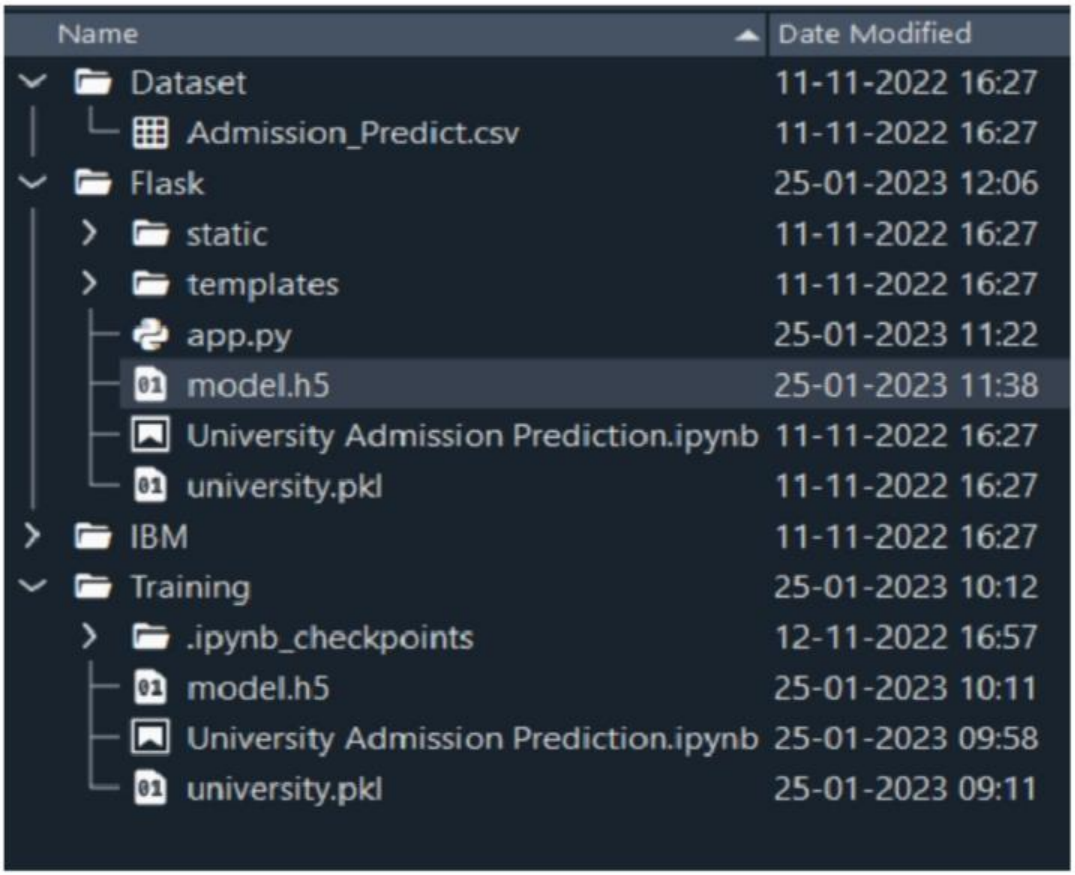
- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UI
- To accomplish this, we have to complete all the activities listed below,
- Define Problem / Problem Understanding
- Specify 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 & Hyper parameter Tuning
- Testing model with multiple evaluation metrics
- Comparing model accuracy before & after applying hyper parameter 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

Project Structure:

Create the Project folder which contains files as shown below



- ❖ We are building a flask application which needs HTML pages stored in the templates
- ❖ folder and a python script app.py for scripting.
- ❖ flight.pkl is our saved model. Further we will use this model for flask integration.
- ❖ Training folder contains a model training file.

Purpose:

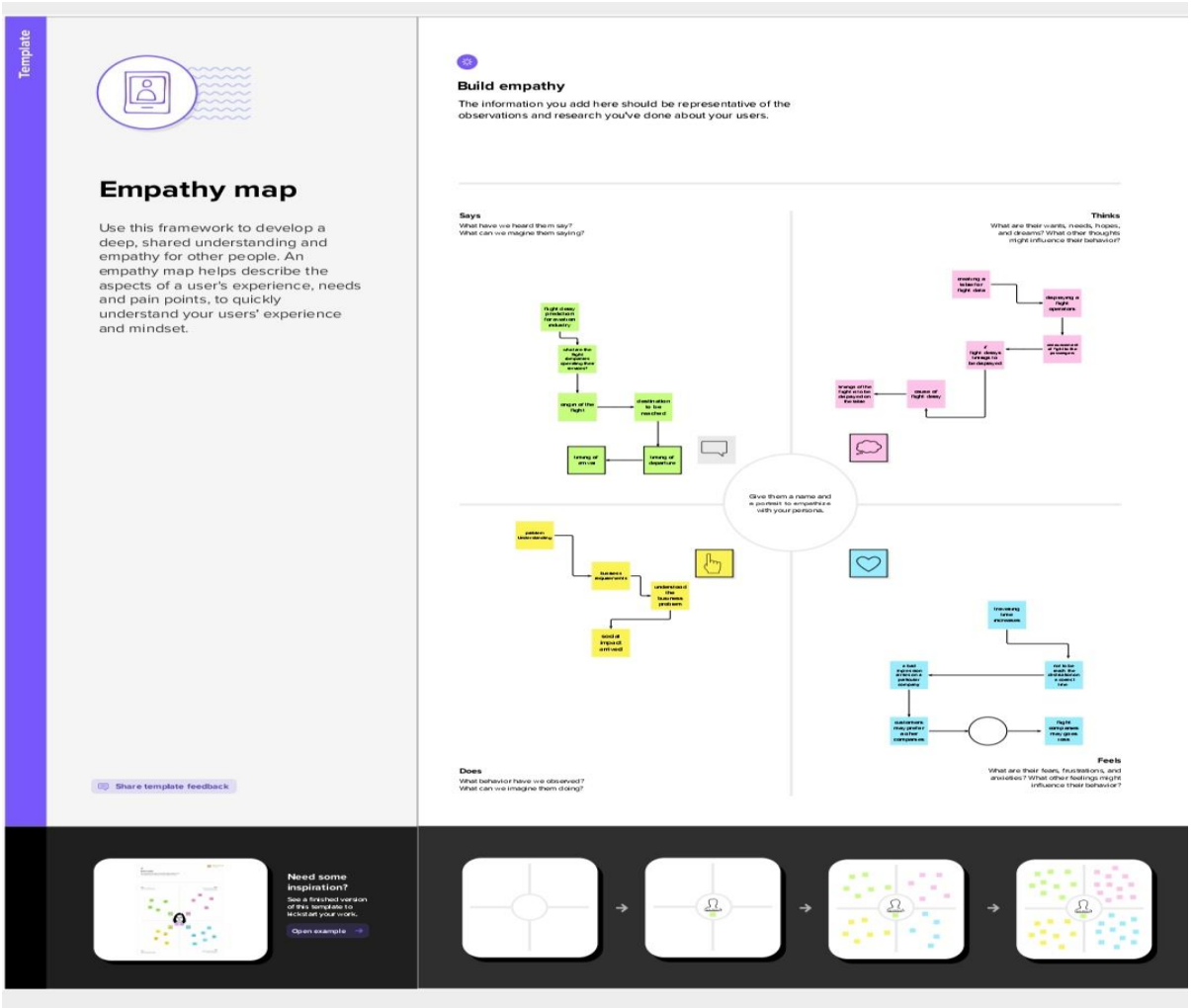
Machine Learning has an additional benefit of processing large chunks of data that is sometimes tiresome for men to do and eventually lead to a failure in Making the right decision. It is easily adaptable to new and complex data. After Processing the data, It is capable of analyzing any flaws or errors.

These also Help in creating effective plans of Actions for improvement. There is a co-Relation between inputs and outputs in the process of decision-making. These points are extremely useful for ventures that work mainly around risk management.

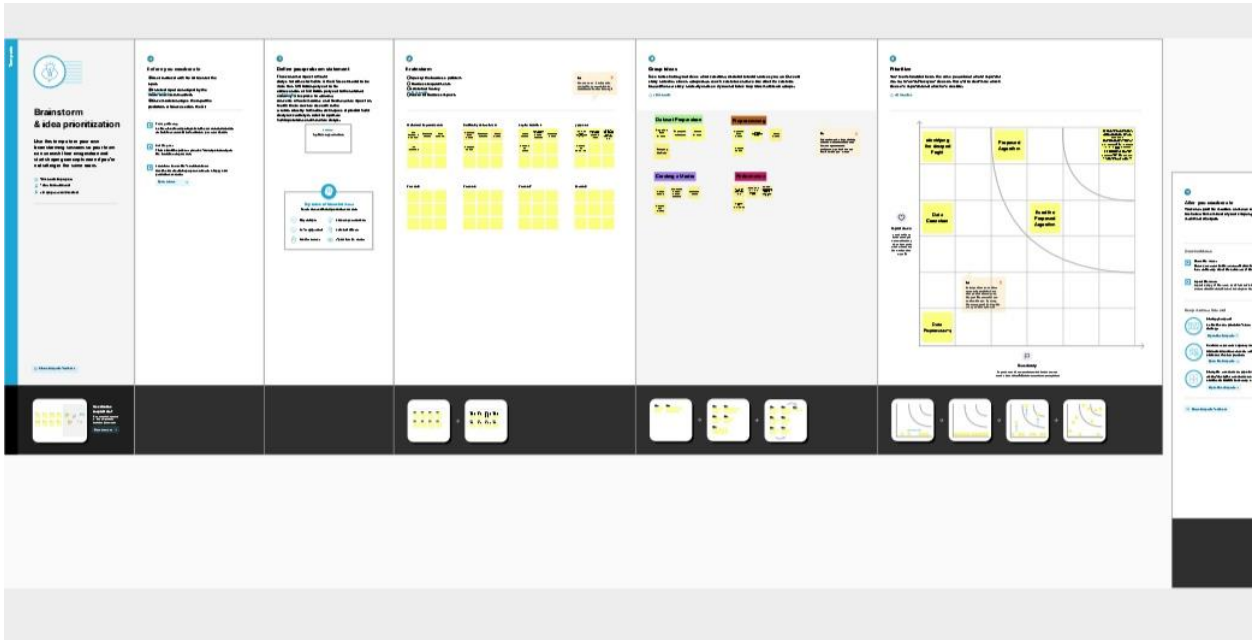
PROBLEM DEFINITION AND DESIGN THINKING

Machine learning has become an increasingly popular tool in recent years, given Its ability to automatically detect patterns in data and make predictions about Future events. This can be extremely useful for making decisions in a wide range Of domains, of flight delay prediction**…**

EMPATHY MAP:



IDEATION AND BRAINSTORM MAP:



RESULT :

ADVANTAGE AND DISADVANTAGE:

Advantage:

- **Providing better information**: Since machine learning technology can sift through extremely large amounts of data, it is able to also provide better information to decision makers.

- **Automating the process**: In many industries, it is simply not possible for human beings to make optimal decisions all of the time. This is especially true in industries where the data is constantly changing, such as financial markets. In these cases, machine learning algorithms can be used to automatically make decisions as trends change and evolve.

- **Improving accuracy**: By identifying patterns in data that humans may not be able to see, machine learning can drastically improve the accuracy of its predictions. It can also create models that simulate different decision scenarios and help identify the best course of action. And as new data becomes available, machine learning can be used to constantly update and refine decision models.

Disadvantage:

- **Data Acquisition**

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

- **Time and Resources**

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

- **Interpretation of Results**

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

- **High error-susceptibility**

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

APPLICATION

- **Decisions in business operations**

Machine Learning algorithms come to the rescue in areas built on a constant flow of heterogeneous data, whether it is several financial reports, payrolls, procurement, the analysis of employee productivity, or predicting further churn rates.

Overall, AI, in terms of inner business processes, is able to leverage business intelligence and make a company data-driven in many aspects, including decision making.

- **Complex problem-solving**

The potential of AI in decision making is robust, but you can solve multilayer and complex problems, too. Artificial Intelligence here gathers tons of different data and conducts an interdisciplinary study. Eventually, there's a way to leverage anything from product development stages to digital marketing approaches of product promotion. Also, it's a way to optimize various types of predictions and risk management. For example, you can predict and optimize pricing with the help of AI tools.

- **Strategic changes**

AI allows better planning of production, managing all restrictions, reducing shortcomings in operations, and improving manufacturing.

It also helps to anticipate and adequately plan product customization, enhance postponement processes, and maintain efficiency with high levels of customer satisfaction.

- **Customer-related decisions**

AI can be valuable for customer service management, personalized customer communication, evaluation of customer behavior, predicting consumer trends and patterns. Artificial intelligence enables automatic recognition and profiling of potential customers.

- **Performance assessment**

Firstly, it relates to people's performance evaluation and afterward decisions. AI is capable of minimizing human errors and making employee performance data more transparent.

AI can also recommend online courses, training, and development programs to employees based on their performance history.

CONCLUSION

In this project addresses machine learning models to predict the chance of flight delay. This will assist customer to know in advance if they have a chance to get another flight. The machine learning models included are multiple linear regression, random forest, Multiple Linear Regression with Backward Elimination and random forest regression with backward elimination. Experiments show that the Linear Regression model surpasses other models.

Our aim would be to predict the "Chance for flight delay" based on the different parameters that are provided in the dataset. We will achieve this aim by using the Linear Regression model. Based on the data that we have, we will split out data into training and testing sets. The Training set will have features and labels on which our model would be trained. The label here is the "Chance for flight delay". If you think from a no-technical standpoint then label is basically the output that we want and features are the parameters that drive us towards the output. Once our model is trained, we will use the trained model and run it on the test set and predict the output. Then we will compare the predicted results with the actual results that we have to see how our model performed. This whole process of training the model using features and known labels and later testing it to predict the output is called Supervised Learning.

FUTURE SCOPE

With the help of machine learning services like SDKs and APIs, developers are able to include and hone the intelligent capabilities into their applications.

APPENDIX:

Source Code:

```
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,RandomForestClassi
fier
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, confusio
n_matrix,f1_score

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

dataset.head()
```

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	UNIQUE_CARRIER	TAIL_NUM	FL_NUM	OR
0	2016	1	1	1	5	DL	N836DN	1399	
1	2016	1	1	1	5	DL	N964DN	1476	
2	2016	1	1	1	5	DL	N813DN	1597	
3	2016	1	1	1	5	DL	N587NW	1768	
4	2016	1	1	1	5	DL	N836DN	1823	

5 rows × 26 columns

```
dataset.info()
```

```
<class 'pandas.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
```

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

```
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_ELAPSED_TIME       0
ACTUAL_ELAPSED_TIME  188
DISTANCE               0
dtype: int64
```

```
dataset=dataset[["FL_NUM","MONTH","DAY_OF_MONTH","DAY_OF_WEEK","ORIGIN","DEST","CRS_ARR_TIME","DEP_DEL15","ARR_DEL15"]]
```

```
dataset.isnull().sum()
```

```
FL_NUM            0
MONTH             0
DAY_OF_MONTH      0
DAY_OF_WEEK       0
ORIGIN            0
DEST              0
CRS_ARR_TIME      0
DEP_DEL15       107
ARR_DEL15       188
dtype: int64
```

```
dataset [dataset.isnull().any(axis=1)].head(10)
```

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15
177	2834	1	9	6	MSP	SEA	852	0.0	NaN
179	86	1	10	7	MSP	DTW	1632	NaN	NaN
184	557	1	10	7	MSP	DTW	912	0.0	NaN
210	1096	1	10	7	DTW	MSP	1303	NaN	NaN
478	1542	1	22	5	SEA	JFK	723	NaN	NaN
481	1795	1	22	5	ATL	JFK	2014	NaN	NaN
491	2312	1	22	5	MSP	JFK	2149	NaN	NaN
499	423	1	23	6	JFK	ATL	1600	NaN	NaN
500	425	1	23	6	JFK	ATL	1827	NaN	NaN
501	427	1	23	6	JFK	SEA	1053	NaN	NaN

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

x

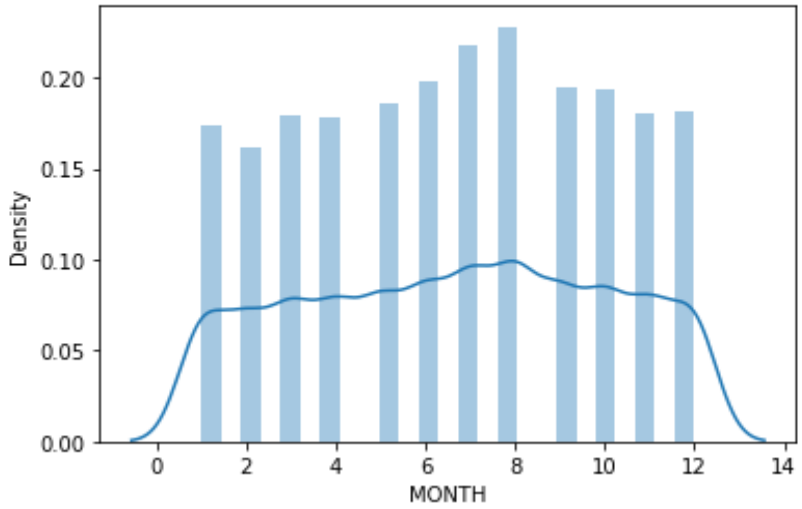
```
array([[1.399e+03, 1.000e+00, 1.000e+00, ..., 0.000e+00, 0.000e+00,
        1.000e+00],
       [1.476e+03, 1.000e+00, 1.000e+00, ..., 0.000e+00, 0.000e+00,
        0.000e+00],
       [1.597e+03, 1.000e+00, 1.000e+00, ..., 0.000e+00, 0.000e+00,
        1.000e+00],
       ...,
       [1.823e+03, 1.200e+01, 3.000e+01, ..., 0.000e+00, 0.000e+00,
        0.000e+00],
       [1.901e+03, 1.200e+01, 3.000e+01, ..., 0.000e+00, 0.000e+00,
        1.000e+00],
       [2.005e+03, 1.200e+01, 3.000e+01, ..., 0.000e+00, 0.000e+00,
        1.000e+00]])
```

```
dataset.describe()
```

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	CRS_ARR_TIME	DEP_
count	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11124.0
mean	1334.325617	6.628973	15.790758	3.960199	15.067314	0.1
std	811.875227	3.354678	8.782056	1.995257	5.023534	0.3
min	7.000000	1.000000	1.000000	1.000000	0.000000	0.0
25%	624.000000	4.000000	8.000000	2.000000	11.000000	0.0
50%	1267.000000	7.000000	16.000000	4.000000	15.000000	0.0
75%	2032.000000	9.000000	23.000000	6.000000	19.000000	0.0
max	2853.000000	12.000000	31.000000	7.000000	23.000000	1.0

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

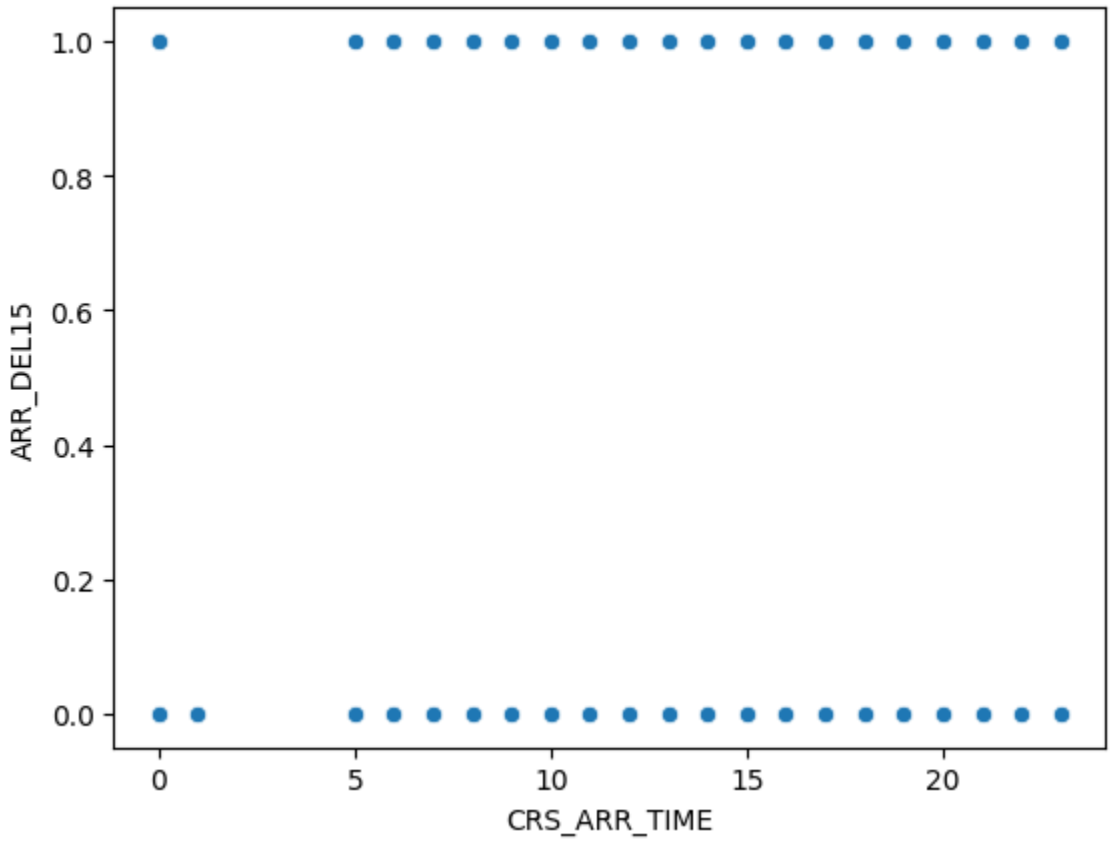
```
sns.distplot(dataset.MONTH)
<Axes: xlabel='MONTH', ylabel='Density'>
```



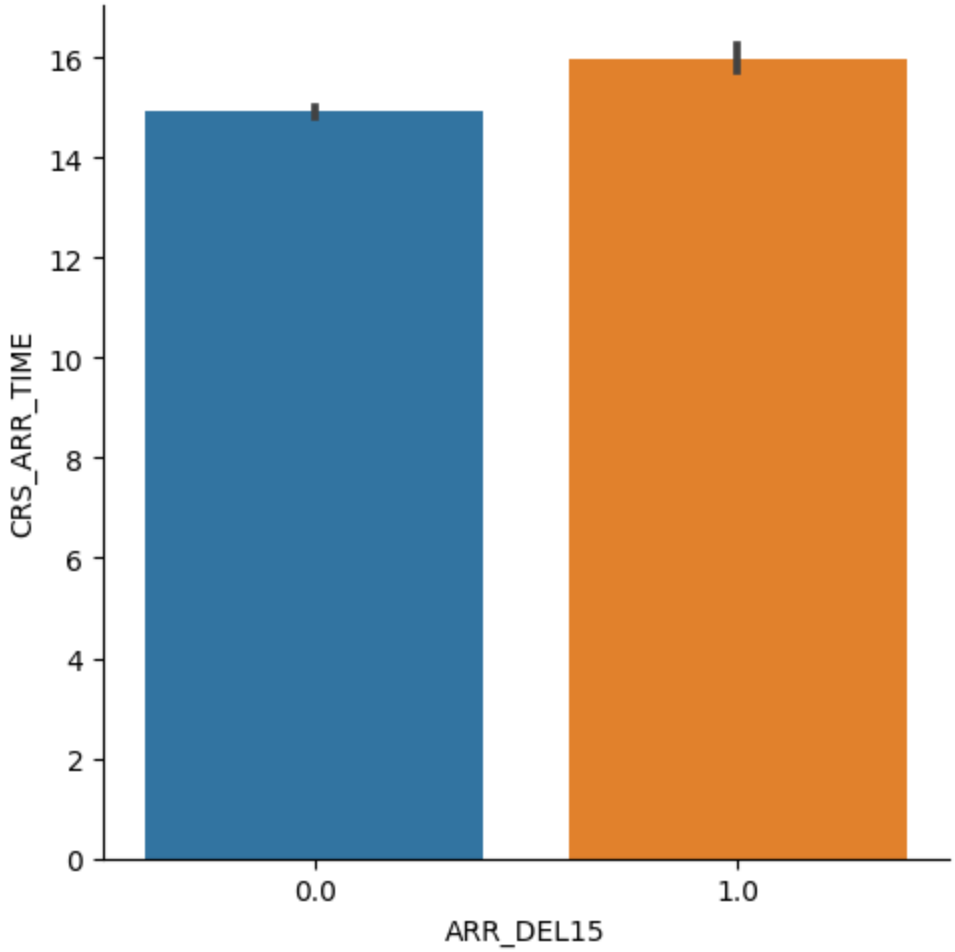
```
plt.scatter
```

```
sns.scatterplot(x='CRS_ARR_TIME',y='ARR_DEL15',data=dataset)
```

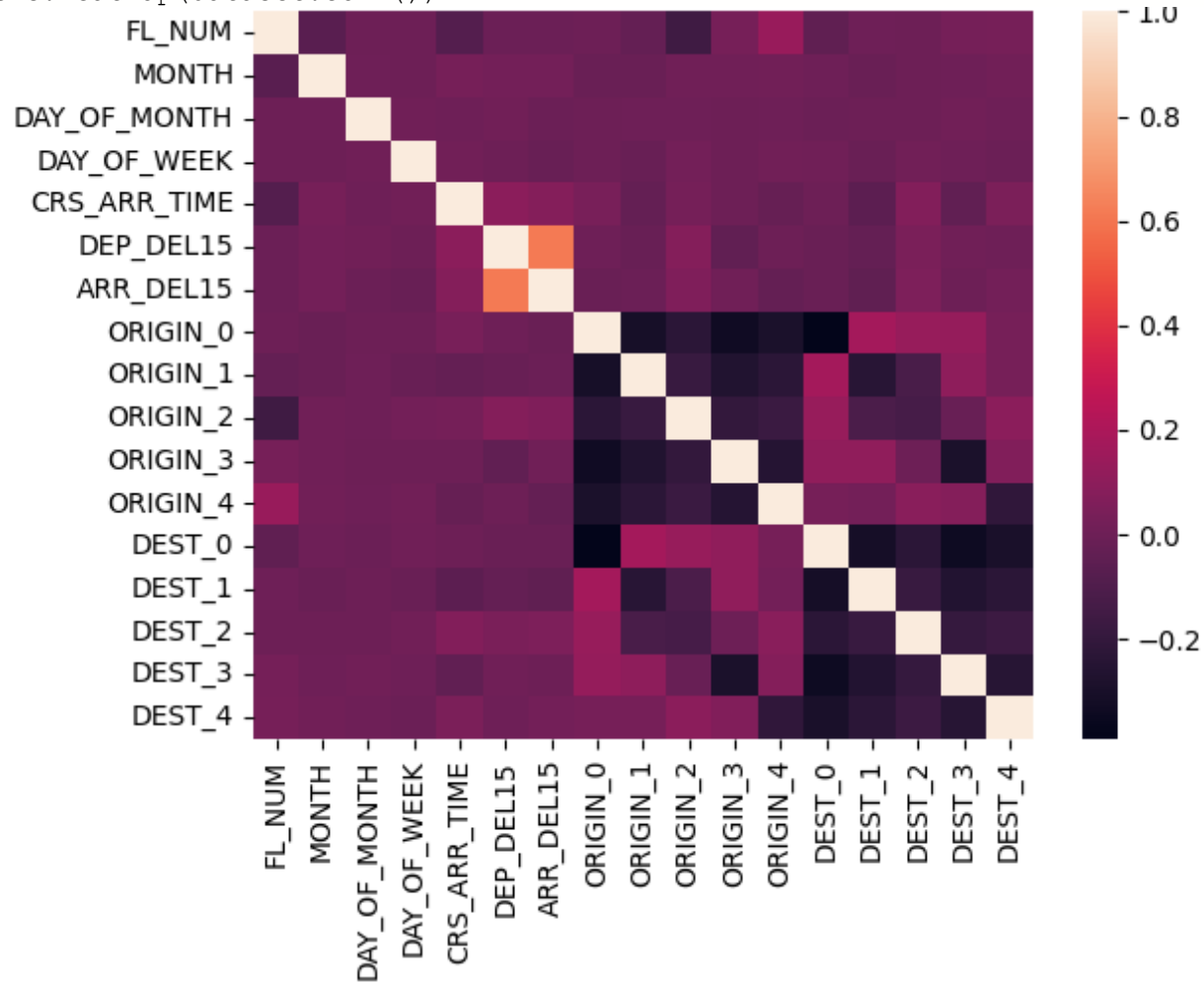
```
<Axes: xlabel='CRS_ARR_TIME', ylabel='ARR_DEL15'>
```



```
sns.catplot(x="ARR_DEL15",y="CRS_ARR_TIME",kind='bar',data=dataset)
<seaborn.axisgrid.FacetGrid at 0x7fda743a1100>
```



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




```
classification.compile(optimizer='adam',loss='binary_crossentropy',metrics
=['accuracy'])
classification.fit(x_train,y_train,batch_size=4,validation_split=0.2,epoch
s=100)
```

```
Epoch 1/100
1797/1797 [==============================] - 4s 2ms/step - loss: 0.4345 - accuracy: 0.8016 - val_loss: 0.4120 - val_accuracy: 0.8125
Epoch 2/100
1797/1797 [==============================] - 3s 2ms/step - loss: 0.4233 - accuracy: 0.8026 - val_loss: 0.4131 - val_accuracy: 0.8125
Epoch 3/100
1797/1797 [==============================] - 4s 2ms/step - loss: 0.4208 - accuracy: 0.8026 - val_loss: 0.4110 - val_accuracy: 0.8125
Epoch 4/100
1797/1797 [==============================] - 3s 2ms/step - loss: 0.4200 - accuracy: 0.8026 - val_loss: 0.4131 - val_accuracy: 0.8125
Epoch 5/100
1797/1797 [==============================] - 4s 2ms/step - loss: 0.4186 - accuracy: 0.8026 - val_loss: 0.4162 - val_accuracy: 0.8125
Epoch 6/100
1797/1797 [==============================] - 4s 2ms/step - loss: 0.4164 - accuracy: 0.8026 - val_loss: 0.4052 - val_accuracy: 0.8125
Epoch 7/100
```

```
y_pred
 array([[0.40808588],
        [0.14142916],
        [0.        ],
        ...,
        [0.5173107 ],
        [0.        ],
        [0.9536835 ]], dtype=float32)
```

```
from 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_weigh
ted', 'roc_auc']
    target_names = ['no delay', 'delay']
    for name, model in models:
      kfold=model_selection.KFold(n_splits=5, shuffle=True, random_state=9
0210)
      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=target_nam
es))
      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)
```

```
                precision    recall  f1-score   support

      no delay       0.85      0.94      0.89      1802
         delay       0.58      0.34      0.43       445

      accuracy                           0.82      2247
     macro avg       0.71      0.64      0.66      2247
  weighted avg       0.80      0.82      0.80      2247
```

DecisionTree

```
                precision    recall  f1-score   support

      no delay       0.99      0.99      0.99      1802
         delay       0.96      0.95      0.96       445

      accuracy                           0.98      2247
     macro avg       0.97      0.97      0.97      2247
  weighted avg       0.98      0.98      0.98      2247
```

```
                precision    recall  f1-score   support

      no delay       0.80      1.00      0.89      1802
         delay       1.00      0.00      0.01       445

      accuracy                           0.80      2247
     macro avg       0.90      0.50      0.45      2247
  weighted avg       0.84      0.80      0.72      2247
```

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

```
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]


}
```

```
rf = RandomForestClassifier()

RCV.fit(x_train,y_train)


y_predict_rf=RCV.predict(x_test)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.0s remaining:    0.0s
building tree 1 of 120
building tree 2 of 120
building tree 3 of 120
building tree 4 of 120
building tree 5 of 120
building tree 6 of 120
building tree 7 of 120
building tree 8 of 120
building tree 9 of 120
building tree 10 of 120
building tree 11 of 120
building tree 12 of 120
building tree 13 of 120
building tree 14 of 120
building tree 15 of 120
```

```
y_predict_rf=RCV.predict(x_test)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done  30 out of  30 | elapsed:    0.0s finished
```

```
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
import pickle
model = pickle.load(open('flight1.pkl','rb'))
app=Flask (__name__)
def home ():
    return render_template("index.html")
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,origin5=0,0,0,0,1
    if(origin == "dtw"):
        origin1, origin2, origin3, origin4, or1gin5=1,0,0,0,0
    if (origin=="jfk"):
```

```
        origin1, origin2, origin3, origin4,origin5=0,0,1,0,0
    if(origin == "sea"):
        origin1, origin2, origin3, origin4, or1gin5=0,1,0,0,0
    if(origin == "alt"):
        origin1, origin2, origin3, origin4, or1gin5=0,0,0,1,0
    destination=request.form['destination']
    if(destination=="msp"):
        destination1, destination2, destination3, destination4, destination5=0,0,0,0,1
    if(destination=="dtw"):
        destination1, destination2, destination3, destination4, destination5=1,0,0,0,0
    if(destination == "jfk"):
        destination1, destination2, destination3, destination4, destination5= 0,0,1,0,0
    if(destination == "sea"):
        destination1, destination2, destination3, destination4, destination5 = 0,1,0,0,0
    if(destination=="alt"):
        destination1, destination2, destination3, destination4, destination5 = 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, origin, origin1, origin3, origin4,origin5, destination,
destination1, destination2, destination3,destination4,destination5]]
    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("Intex1.html", showcase=ans)
if __name__=='__main__':
    app.run(debug=True)
```