FLIGHT DELAY PREDICTION USING MACHINE LEARNING ALGORITHMS

Jeshmitha. G [1] , Samanvitha. M [1], Bindu Sri Sai. U [1] , Mahesh. J [1], Mrs. Lakshmi Kalyani. N [1]

Department of CSE, 1 VNRVJIET

[1] jeshgunuganti@gmail.com

[1] msamanvitha99@gmail.com

[1] uppalapatibindusrisai@gmail.com

[1] maheshjimidi2000@gmail.com

[1] lakshmikalyani_n@vnrvjiet.in

ABSTRACT

Prior Prediction of Flight delays is necessary for both airlines and passengers because flight delays cause not only huge economic loss but also cause airlines to lose their reputation and passengers to lose their valuable time. In this work, we aim to utilize available data sources to predict the arrival delay of a scheduled flight at the destination airport. The primary goal of the model proposed in this paper is to predict airline delays using supervised machine learning algorithms. US domestic flight data, and the weather data from July 2019 to December 2019 were extracted and are used to train the model. XGBoost and linear regression algorithms were implemented to build models that can predict delays of individual flights. Then, the performance of each algorithm was analyzed. Then, in the prediction step, flight data and weather data were gathered and fed into the model. Using this data, the XGBoost trained model performed a binary classification to predict whether a scheduled flight would be delayed or on-time, and the linear regression model predicts the delay time of the flight.

I INTRODUCTION

With the air travel increasing rapidly there is a serious problem of flight delays for both airlines and passengers. Passengers not only lose their time but also their trust in airlines. This will result in a huge economic loss to the airline companies and Airlines lose their reputation as well. Thus, proper monitoring and prediction of flight delays are very important. Airlines usually increase the time between gate departure and gate arrival time to anticipate potential delays due to weather conditions or airspace congestion. So, In our study, we mainly focus on departure delay time, distance, and weather parameters to model the flight arrival delays. Prediction of flight is significantly important for airlines and travellers because flight delays cause not only tremendous economic loss but also potential security risks.

According to Nextor, domestic flight delays cost the US company a sum of $31.2 billion of which $8.2billion directly affects the airlines, $16.2billion affects the passengers, $2.2billion for the lost demand due to flight delays,$4.0billion in forgone GDP. It has also been noted that among all the complaints received from the passengers 33% of the complaints are related to the flight delay.

The primary goal of the model proposed in this paper consists of two phases. The first phase involves predicting if there is any delay in the arrival of an individual flight using a supervised classification algorithm and if yes, the second phase involves predicting approximate delay time in minutes using a supervised regression algorithm.

A more accurate arrival delay prediction model can provide passengers at the same level of certainty without capriciously increasing the waiting time. Considering all the parameters that are the cause for the delay we found that weather affects the delay to a great extent and hence used it as a factor to predict the delay of the flight.

Hence we are using the US dataset of airports and weather data of the airports at the same time. Both the datasets are collected from different online sources. In the first step, we pre-processed the data to make our data smooth and then joined both flight and weather data into a single final data set. Then we split the data into training and testing sets and then the predictive model was built using machine learning methods. The algorithm that was used here was XGBoost classification algorithm as its speed of execution and model performance are very good. Our Classification algorithm predicts if there is any arrival delay or not. For knowing the delay time we pass this data into a Linear Regression algorithm which then predicts by how much time the flight will get delayed.

II STATISTICS

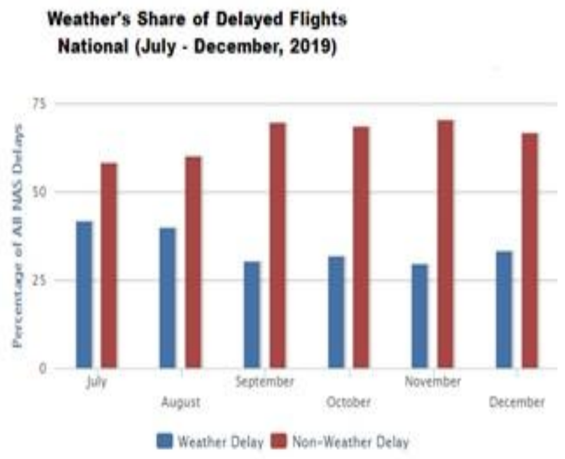


Fig 1 Weather share of delayed flights

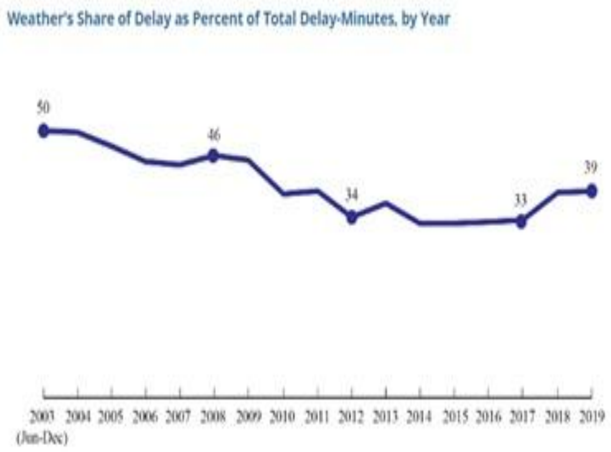
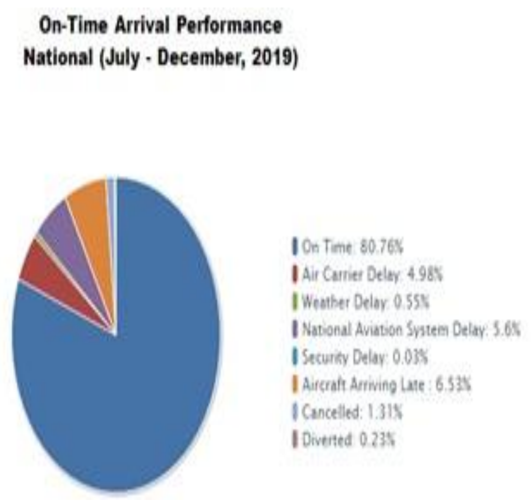


Fig 2 Weathers share of delay time as percentage



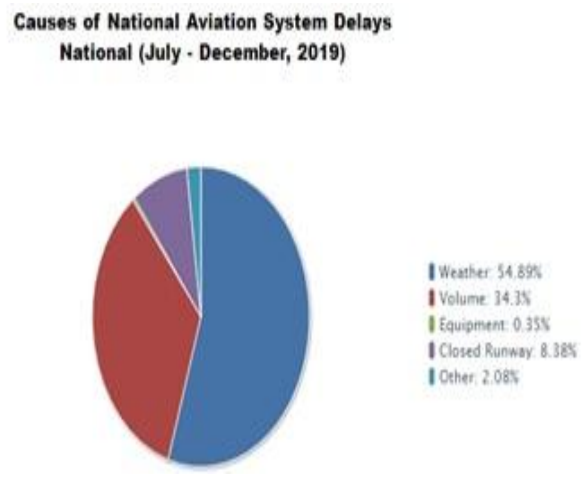


Fig 3 On-Time Arrival Performance **Fig 4 Causes of NAS delays**

III LITERATURE SURVEY

Prediction of Airline Delays based on Machine Learning Algorithms[1]

Alok Dand, Khawaja Saeed, Bayram Yildirim, In this research aimed at examining classification of flights into on-time and delayed based on various features using different machine learning algorithms the dataset was divided into 80:20 ratio for training and testing and was balanced using SMOTE method then the efficiency of the algorithms was evaluated based on accuracy score, roc-auc score, and confusion matrix. The accuracy results of Random Forest, Decision Tree, K-Nearest Neighbor, Naive Bayes, Local Outlier Factor are 80.9%,81.6%,74.8%,62.9%,51.24%.

Flight delay prediction for commercial air transport: A deep learning approach[2]

Zhen Guo and Sobhan Sean Asian analyzing high-dimensional data from Beijing International Airport presented a flight delay prediction model. A belief network technique was employed to understand the hidden patterns of flight delays. Support vector regression was embedded in the developed model to perform a supervised fine-tuning within the presented predictive architecture. The overall performance of the proposed DBN-SVR model was also compared with three methods, namely k-nearest neighbors (k-NN), support vector machine (SVM), and linear regression (LR). Influential factors, such as air route situation and crowdedness degree of airports were introduced and examined through a multifactor approach. The observed values of mean absolute error(MAE) and root mean absolute error (RMSE) for the DBN-SVR model are 8.41, 12.04 respectively.

Flight Delay Prediction using Binary Classification[3]

R Musaddi, A Jaiswal, J Pooja, M Girdonia, MS_Minu proposed a work that aims at analysing flight delays using binary classification with score higher than 0.85 and obtained

ROC curve that determines the ability of the binary classifier by using grid search on random forest model. Data sets from the month of January from US department of transportation has been taken and using label binarizer , the categorical features were converted into sparse matrices. Then , Binary classifier was built using random forest algorithm. Also ,various histograms to compare airlines and their delays with respect to weeks and days

Predictive Modeling of Aircraft Flight Delay[4]

Anish M. Kalliguddi and Aera K. Leboulluec developed a predictive model to forecast flight delays. The models that are based on multiple linear regression, decision trees, and random forest algorithms are created and tested. This concluded that the Random forest model outperforms the other two models based on the evaluation criteria. The splitting variables or the significant variable are found to be late aircraft delay, Carrier delay, weather delay and NAS delay. Although the model gives very good prediction accuracy, more variables can be considered to develop a predictive model. The prediction error was found to be 153.94 and the Root Mean Squared Error (RMSE) for the Random forest tree (RFT)model is 12.5 minutes.

Predicting flight delay based on multiple linear regression[5]

Yi Ding, In this study, proposes a method to predict the estimation of arrival delay using multiple linear regression algorithm. The dataset contains the data around 100,000 records, for each record they use a secondary data source to enrich it with information about airplanes and historical weather statistics. After predicting whether a flight will be delayed or not, using multiple linear regression, they compare their model with Naive-Bayer and C4.5 approach. Results of the Naive-Bayes algorithm show us that the classifier performance is better in predicting non-delayed flights than delayed ones. F-score on predicting on-time flights is 0.75, while that for delays is only 0.57. The C4.5 performs slightly worse than Naive-Bayes in predicting delays with F-score equal to 0.48. Based on the results, the accuracy of the proposed model using multiple linear regression is 80%, which is better when compared with Naive-Bayes and C4.5 approaches.

Flight Arrival Delay Prediction Using Gradient Boosting Classifier[6]

Navoneel Chakrabarty, Tuhin Kundu, Sudipta Dandapat, Apurba Sarkar and Dipak Kumar Kole proposed work to predict arrival delay of flights using 4 supervised machine learning algorithms. the accuracy scores of support vector machine, random forest, k-nearest neighbour,gradient boosting algorithms are 78.2%,78.7%,77.9%,79.7%.The results suggest that the gradient boosting classifier performed better when compared to other algorithms with a testing accuracy of 79.7%,auc-roc score of 0.54, and having minimum false negatives in its confusion matrix.

Chained Predictions of Flight Delay Using Machine Learning[7]

Jun Chen and Meng Li proposed a new machine learning-based air traffic delay prediction model that does multi-label random forest classification and approximates a model for delay propagation. Departure delay and late-arriving aircraft delay are considered to be the most important features for delay prediction. To utilize these two features, a delay propagation model was built as a connection to develop a chained delay prediction model. The arrival delay and departure delay prediction modules are based on the Random forest model trained with selected features. The comparison is done among Arrival Delay all features, Arrival Delay optimal features, Departure Delay all features, Departure Delay optimal features with Relaxed Accuracy as 0.87498, 0.92669, 0.88244, 0.90354 and Accuracy as ,0.85915, 0.86727, 0.82684 ,0.83050 respectively.

A Methodology for Predicting Aggregate Flight Departure Delays in Airports Based on Supervised Learning[8]

Bojia Ye , Bo Liu, Yong Tian, and Lili Wan proposed a new methodology for predicting flight delays in airports by using supervised learning. The proposed model was enabled by four types of airport-related aggregate,including time, flight plan, delay, and local weather characteristics. Used models like LightGBM, ExtraTR, SVM, and SVM considering local meteorological data to calculate the mean square error (MSE) and mean absolute error (MAE). Among all, the LightGBM model provides the best result, giving 0.8653 accuracy.

IV DATASET DESCRIPTION

4.1 Flight Dataset:

The US Bureau of Transport Statistics provides data on all US domestic flights. We can download the data from a specific time period, make selections from the Filter Year and Filter Period drop-down lists and also select specific data fields from the table, including scheduled and actual departure, arrival and takeoff times, origin, destination, date, distance, carrier, etc...

4.2 Weather Dataset:

The Local Climatological Data from NOAA provides weather data for stations and locations within the United States and its territories. We can download the data by specifying the state or territory, location, along with the time period. Climatic values in the dataset include hourly, daily, and monthly measurements of temperature, dew point, humidity, winds, sky condition, weather type, atmospheric pressure, and more.

4.3 Final Dataset:

Joining the flight and weather datasets to form final dataset:

Joining both (flight and weather) datasets presents a significant challenge because weather observation times are not the same as flight departure times and both weather and flight datasets have different formats as they are from different organizations. Now, we must join both the datasets into one dataset based on place, date, and time. For instance, if we have a flight departing at Los angles airport on 1st July 2019, at 1 am, we need weather observations at the same place, date, and time. So, we calculated average of weather observations for each hour at each station and then performed an inner join on both datasets over the unique code field that was created by us to uniquely identify flight and weather data at each hour on a particular day at a given place. i. e., code used to join two datasets was: PLACE.YEAR.MONTH.DAY.HOUR.

Features (X-values)considered in building the model:

S. No.	Attribute	Attribute type	Description
1.	DEP_DEL_15 (0 (on-time) or 1(delay))	Boolean	**Departure Delay** Indicator, 15 Minutes or More (1=Yes)
2.	DEP_DELAY	Numeric	Difference in minutes between scheduled and actual **departure time**. Early departures show negative numbers.
3.	DISTANCE	Numeric	Distance between origin and destination airports in miles
4.	Altimeter setting	Numeric	Atmospheric pressure reduced to sea level using temperature profile of the “standard” atmosphere in inches of Mercury (in Hg).
5.	Dew point temperature	Numeric	It is the temperature to which air must be cooled .The dew point in relation to the temperature gives the pilots information about the humidity

6.	Dry bulb temperature	Numeric	It is the temperature of air measured by a thermometer freely exposed to the air, but shielded from radiation and moisture.
7.	Relative humidity	Numeric	Relative humidity is a measure of how close the air is to reaching saturation.
8.	Station pressure	Numeric	This is the pressure that is observed at a specific elevation and is the true barometric pressure of a location.
9.	Visibility	Numeric	The horizontal distance an object can be seen and identified given in whole miles.
10.	Wet bulb temperature	Numeric	It is an apparent measurement used to estimate the most accurate level of heat stress in direct sunlight.
11.	Sea level temperature	Numeric	It is the water temperature close to the surface of sea.

Table 1 Features(X-values) of Dataset

Y-values to be predicted:

S. No	**Attribute**	**Attribute type**	**Description**
1.	ARR_DELAY_15(0(on-time) or 1(delay))	Boolean	**Arrival Delay** Indicator, 15 Minutes or More (1=Yes)
2.	ARR_DELAY	Numeric	Difference in minutes between scheduled and actual **arrival time**. Early arrival show negative numbers.

Table 2 Values to be predicted

V ALGORITHMS

5.1 XGBoost:

XGBoost is a decision-tree-based ensemble machine learning algorithm that used a gradient boosting framework. It can be used to solve regression, classification, ranking, and user-defined prediction problems.

Features of XGBoost algorithm:

- Regularized boosting (prevents over fitting)
- Parallel processing
- Can cross-validate at each iteration
    - Enables early stopping, finding optimal number of iterations
- Incremental training
- Can plug in your own optimization objectives
- Tree pruning
    - Generally results in deeper, but optimized trees

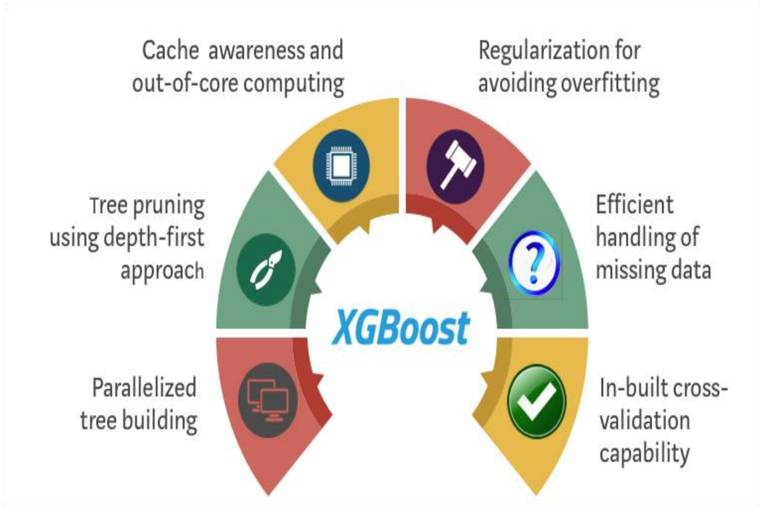

Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Fig 5 XGBoost

5.2 Linear Regression:

Linear regression is one of the most popular machine learning algorithms for predicting values given a set of values. Linear regression is a linear method used to model the relationship between independent and dependent variables. Its simplicity and ease of implementation are its advantages. The least-square method can be used to create a line that best fits the data. Some applications of linear regression can be found in machine learning, Business domain, forecasting sales, economics and in places where estimation is required.

Parameters for a Linear Regression model :

- **fit_intercept** is a Boolean (True by default) that decides whether to calculate the intercept b_0 (True) or consider it equal to zero (False).
- **normalize** is a Boolean (False by default) that decides whether to normalize the input variables (True) or not (False).
- **copy_X** is a Boolean (True by default) that decides whether to copy (True) or overwrite the input variables (False).
- **n_jobs** is an integer or None (default) and represents the number of jobs used in parallel computation. None usually means one job and -1 to use all processors.

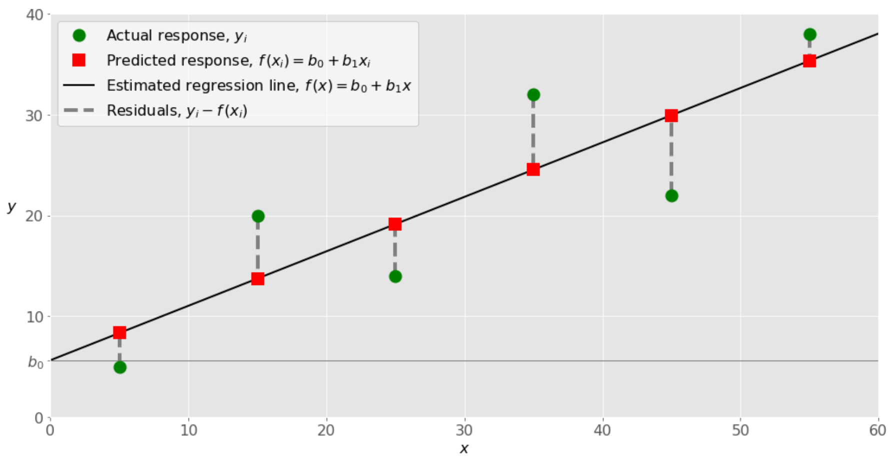


Fig 6 Linear Regression

VI TOOLS AND LANGUAGE

In this study, we used the Jupyter notebook as a tool and python 3.7 as a programming language and python libraries namely numpy, pandas, matplotlib, seaborn, sklearn.

VII IMPLEMENTATION

7.1 Implementation Flow Charts:

Flow Chart for Data Pre-processing:

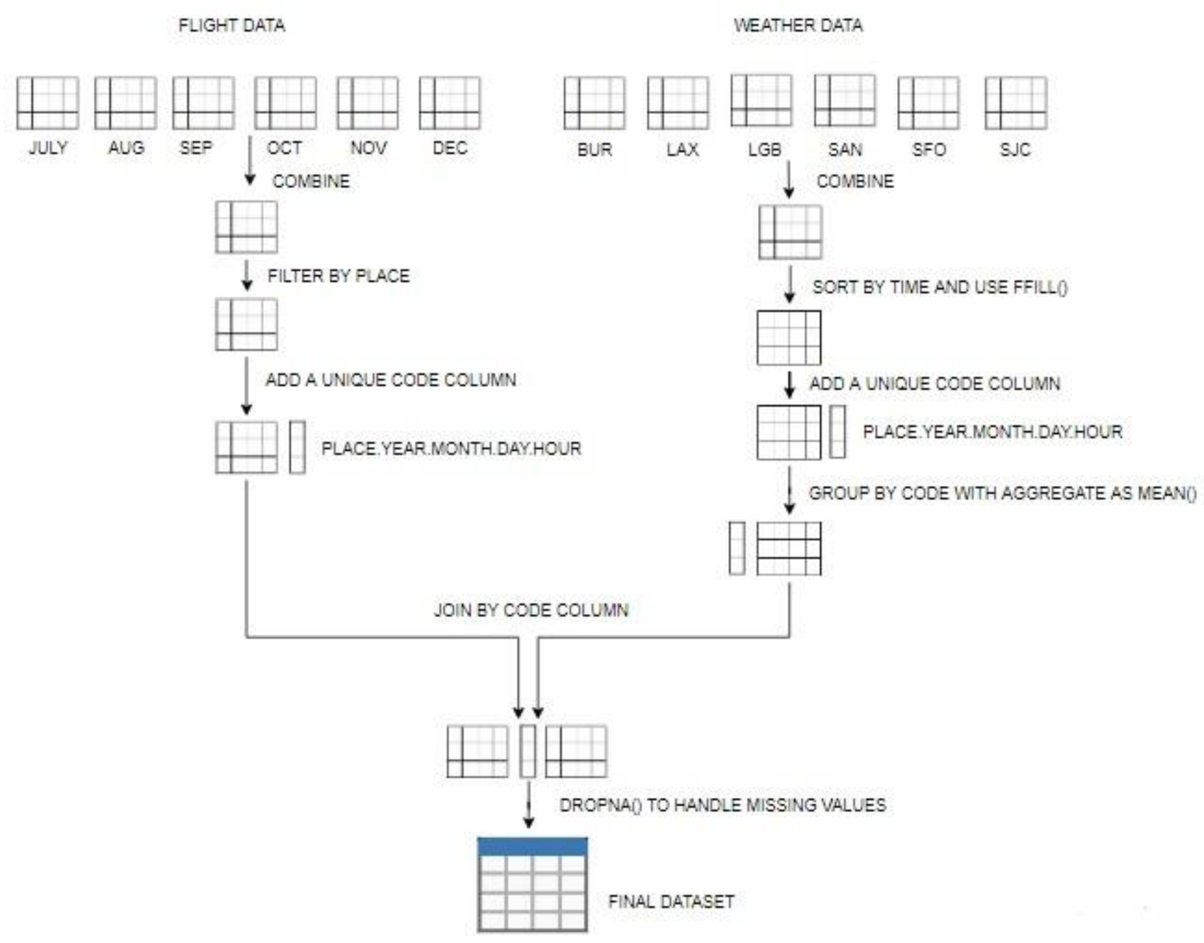


Fig 7 Flow chart for Data Pre-processing

Flow Chart for Modelling:

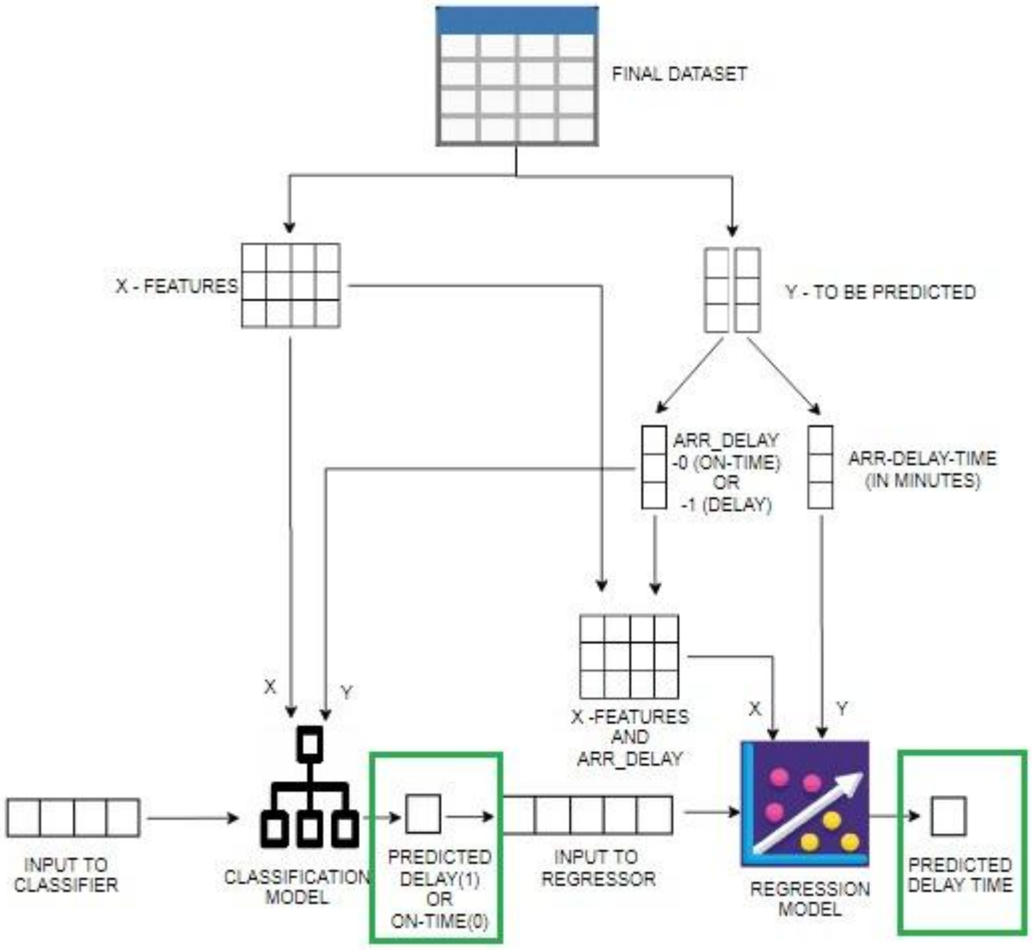
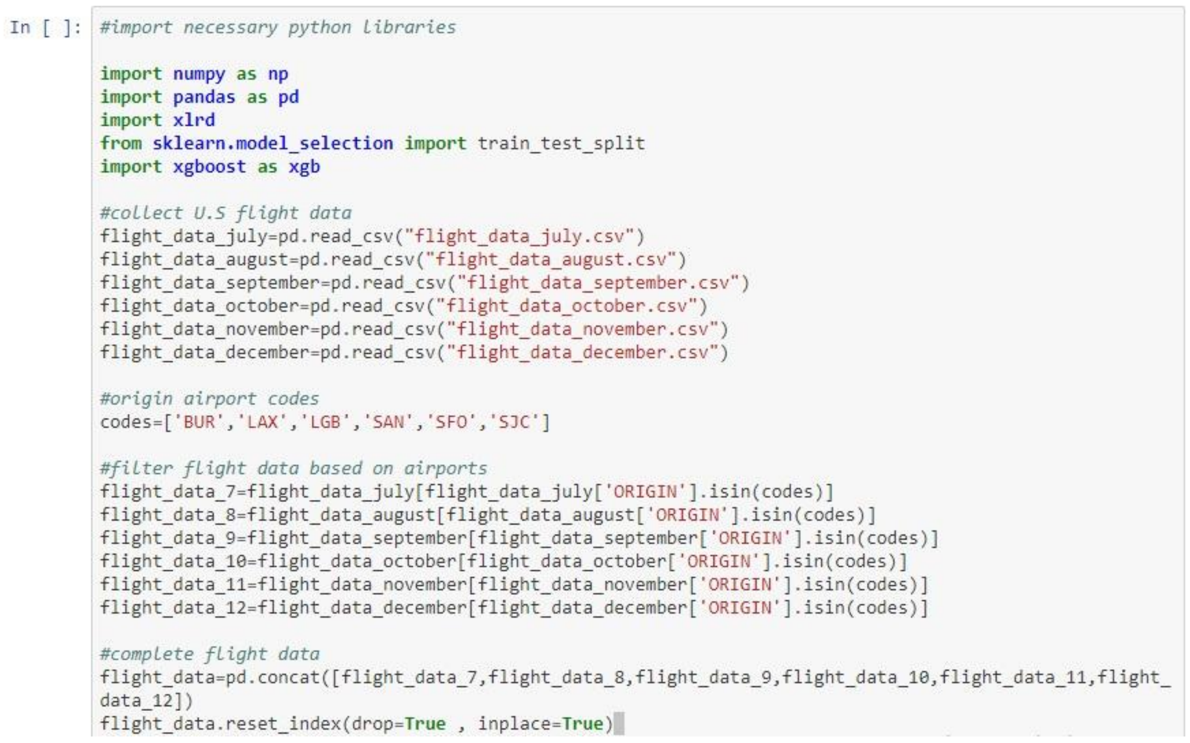
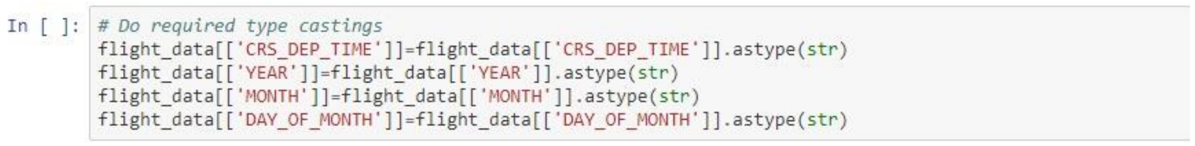


Fig 8 Flow chart for Modelling

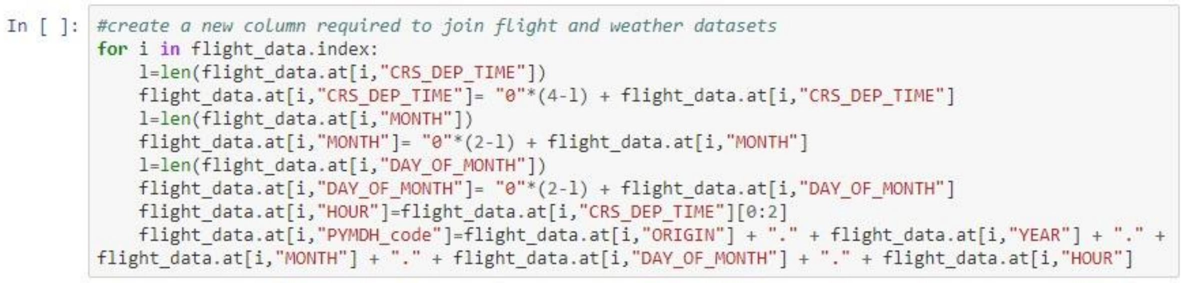
7.2 Code



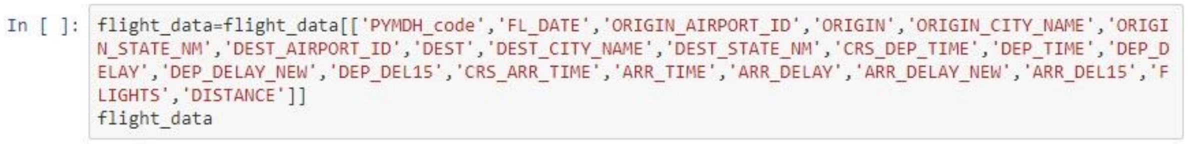
```
In [ ]: #import necessary python libraries

        import numpy as np
        import pandas as pd
        import xlrd
        from sklearn.model_selection import train_test_split
        import xgboost as xgb

        #collect U.S flight data
        flight_data_july=pd.read_csv("flight_data_july.csv")
        flight_data_august=pd.read_csv("flight_data_august.csv")
        flight_data_september=pd.read_csv("flight_data_september.csv")
        flight_data_october=pd.read_csv("flight_data_october.csv")
        flight_data_november=pd.read_csv("flight_data_november.csv")
        flight_data_december=pd.read_csv("flight_data_december.csv")

        #origin airport codes
        codes=['BUR','LAX','LGB','SAN','SFO','SJC']

        #filter flight data based on airports
        flight_data_7=flight_data_july[flight_data_july['ORIGIN'].isin(codes)]
        flight_data_8=flight_data_august[flight_data_august['ORIGIN'].isin(codes)]
        flight_data_9=flight_data_september[flight_data_september['ORIGIN'].isin(codes)]
        flight_data_10=flight_data_october[flight_data_october['ORIGIN'].isin(codes)]
        flight_data_11=flight_data_november[flight_data_november['ORIGIN'].isin(codes)]
        flight_data_12=flight_data_december[flight_data_december['ORIGIN'].isin(codes)]

        #complete flight data
        flight_data=pd.concat([flight_data_7,flight_data_8,flight_data_9,flight_data_10,flight_data_11,flight_
        data_12])
        flight_data.reset_index(drop=True , inplace=True)
```



```
In [ ]: # Do required type castings
        flight_data[['CRS_DEP_TIME']]=flight_data[['CRS_DEP_TIME']].astype(str)
        flight_data[['YEAR']]=flight_data[['YEAR']].astype(str)
        flight_data[['MONTH']]=flight_data[['MONTH']].astype(str)
        flight_data[['DAY_OF_MONTH']]=flight_data[['DAY_OF_MONTH']].astype(str)
```



```
In [ ]: #create a new column required to join flight and weather datasets
        for i in flight_data.index:
            l=len(flight_data.at[i,"CRS_DEP_TIME"])
            flight_data.at[i,"CRS_DEP_TIME"]= "0"*(4-l) + flight_data.at[i,"CRS_DEP_TIME"]
            l=len(flight_data.at[i,"MONTH"])
            flight_data.at[i,"MONTH"]= "0"*(2-l) + flight_data.at[i,"MONTH"]
            l=len(flight_data.at[i,"DAY_OF_MONTH"])
            flight_data.at[i,"DAY_OF_MONTH"]= "0"*(2-l) + flight_data.at[i,"DAY_OF_MONTH"]
            flight_data.at[i,"HOUR"]=flight_data.at[i,"CRS_DEP_TIME"][0:2]
            flight_data.at[i,"PYMDH_code"]=flight_data.at[i,"ORIGIN"] + "." + flight_data.at[i,"YEAR"] + "." +
        flight_data.at[i,"MONTH"] + "." + flight_data.at[i,"DAY_OF_MONTH"] + "." + flight_data.at[i,"HOUR"]
```



```
In [ ]: flight_data=flight_data[['PYMDH_code','FL_DATE','ORIGIN_AIRPORT_ID','ORIGIN','ORIGIN_CITY_NAME','ORIGI
        N_STATE_NM','DEST_AIRPORT_ID','DEST','DEST_CITY_NAME','DEST_STATE_NM','CRS_DEP_TIME','DEP_TIME','DEP_D
        ELAY','DEP_DELAY_NEW','DEP_DEL15','CRS_ARR_TIME','ARR_TIME','ARR_DELAY','ARR_DELAY_NEW','ARR_DEL15','F
        LIGHTS','DISTANCE']]
        flight_data
```

In [ ]:
```
#weather dataset
weather_data=pd.read_excel("Weather_Data.xlsx","weather_data")
#do required type castings
weather_data["STATION"]=weather_data["STATION"].astype(str)

#find station names with given station id's
station_name={'72288023152':'BUR','72290023188':'SAN','72295023174':'LAX','72297023129':'LGB','7249452
3293':'SJC','99999923272':'SFO'}
weather_data["station_name"]=weather_data["STATION"].map(station_name)
```

In [ ]:
```
#Create a new column required to join flight and weather datasets
for i in weather_data.index:
    weather_data.at[i,"MINUTE"]=weather_data.at[i,"DATE"][14:16]
    weather_data.at[i,"PYMDH_code"]= weather_data.at[i,"station_name"]+"."+weather_data.at[i,"DATE"]
[0:4]+"."+weather_data.at[i,"DATE"][5:7]+"."+weather_data.at[i,"DATE"][8:10]+"."+weather_data.at[i,"DA
TE"][11:13]
```

In [ ]:
```
#Handling missing values in weather dataset

weather_data.sort_values(by=['PYMDH_code','MINUTE'],inplace= True)
weather_data.reset_index(inplace=True,drop=True)
weather_data=weather_data[['PYMDH_code','STATION','station_name','DATE','MINUTE','HourlyAltimeterSetti
ng','HourlyWetBulbTemperature','HourlyDewPointTemperature','HourlyDryBulbTemperature','HourlyRelativeH
umidity','HourlyStationPressure','HourlyVisibility','HourlySeaLevelPressure']]
weather_data.ffill(axis=0,inplace=True)

weather_data=weather_data.groupby('PYMDH_code').agg({'HourlyAltimeterSetting':np.mean,'HourlyDewPointT
emperature': np.mean,'HourlyDryBulbTemperature': np.mean,'HourlyRelativeHumidity': np.mean,'HourlyStat
ionPressure': np.mean,'HourlyVisibility': np.mean,'HourlyWetBulbTemperature': np.mean,'HourlySeaLevelP
ressure': np.mean})
```

In [ ]:
```
#inner join wethaer and flight datasets over unique code column
final_data=pd.merge(flight_data,weather_data,on='PYMDH_code')
```

In [ ]:
```
#handling missing values in final dataset
final_data.dropna(inplace=True)
final_data=final_data.sample(frac=1,axis=0)
final_data.reset_index(drop=True,inplace=True)
```

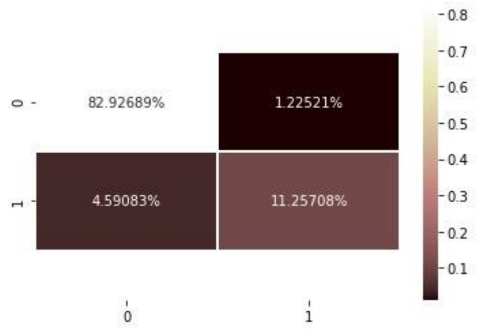
In [ ]:
```
#do required type castings in final dataset
final_data['DEP_DEL15']=final_data['DEP_DEL15'].astype(bool)
final_data['ARR_DEL15']=final_data['ARR_DEL15'].astype(bool)

#splitting the dataset for training and testing

X=final_data[['DEP_DELAY','DEP_DEL15','DISTANCE','HourlyAltimeterSetting','HourlyDewPointTemperatur
e','HourlyDryBulbTemperature','HourlyRelativeHumidity','HourlyStationPressure','HourlyVisibility','Hou
rlyWetBulbTemperature','HourlySeaLevelPressure' ]]
Y=final_data[['ARR_DEL15','ARR_DELAY']]
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.60, random_state=42)
X_train.reset_index(drop=True,inplace=True)
X_test.reset_index(drop=True,inplace=True)
Y_train.reset_index(drop=True,inplace=True)
Y_test.reset_index(drop=True,inplace=True)
CY_train=Y_train[['ARR_DEL15']]
CY_test=Y_test[['ARR_DEL15']]
RY_train=Y_train[['ARR_DELAY']]
RY_test=Y_test[['ARR_DELAY']]
```

```
In [ ]: #Training the model for classification task using xgboost algorithm

        boost =  xg_b.XGBClassifier(
         learning_rate = .1,
         n_estimators=459,
         max_depth=5,
         min_child_weight=1,
         gamma=0,
         subsample=0.8,
         colsample_bytree=0.8,
         objective= 'binary:logistic',
         nthread=4,
         scale_pos_weight=1,
         seed=27).fit(X_train, CY_train)
        Y_pred=boost.predict(X_test)

        Y_pred=pd.DataFrame(Y_pred)
```

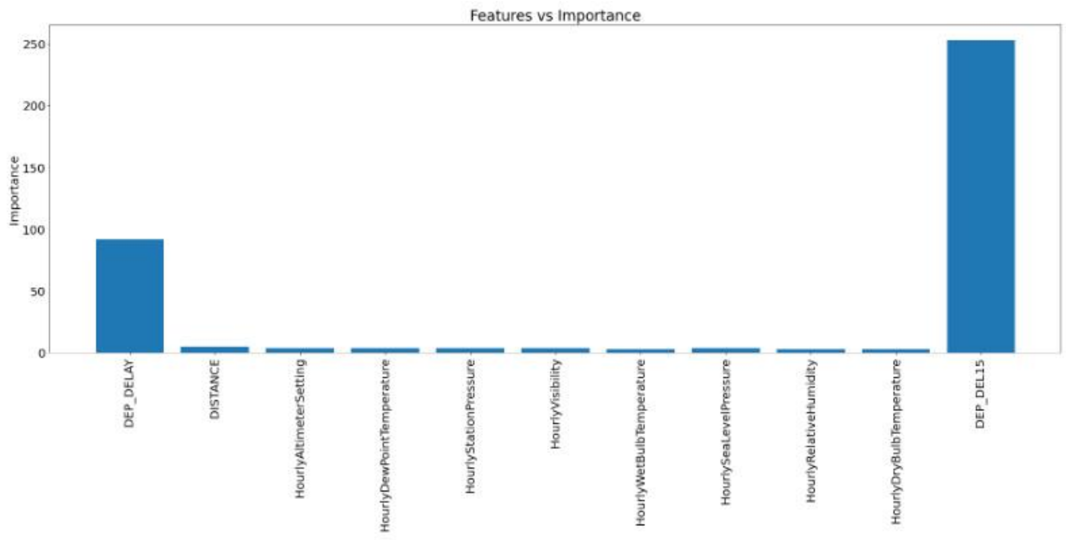
```
Performance metrics for classfication model
---------------------------------------
confusion matrix:
[[105925   1565]
 [  5864  14379]]
accuracy score: 0.9418396185793805
precision: 0.9018439538384345
recall: 0.71031961665761
f1_score: 0.7947052809019814
auc score: 0.8478800613756001
classification report:
              precision    recall  f1-score   support

       False       0.95      0.99      0.97    107490
        True       0.90      0.71      0.79     20243

    accuracy                           0.94    127733
   macro avg       0.92      0.85      0.88    127733
weighted avg       0.94      0.94      0.94    127733
```

```
In [ ]: import matplotlib.pylab as plt
        import seaborn as sns
        ax=sns.heatmap(c_m/np.sum(c_m),annot=True ,linewidth=.5,
                    fmt='.5%', cmap='pink')
        bottom, top = ax.get_ylim()
        ax.set_ylim(bottom + 0.5, top - 0.5)
```

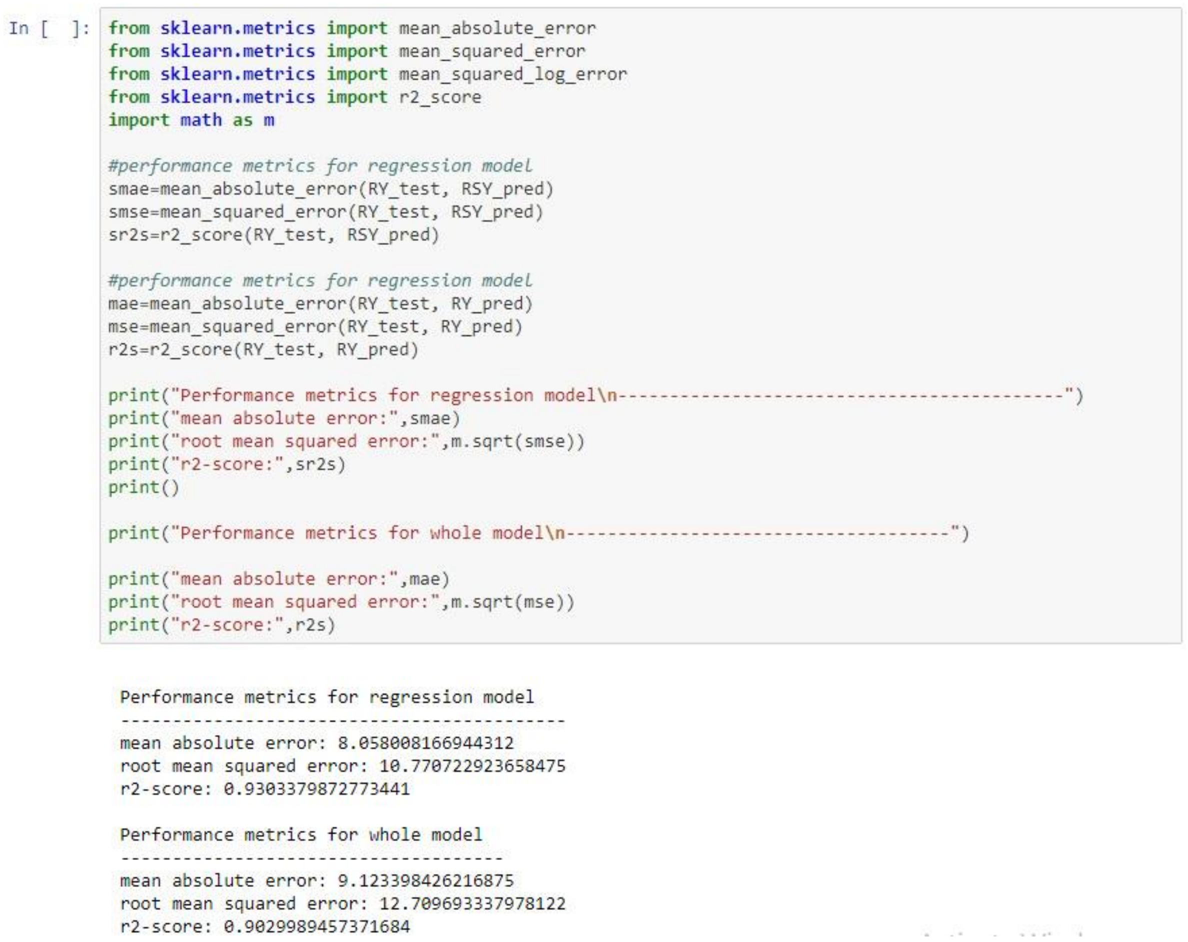
```
Out[ ]: (2.5, -0.5)
```



```
In [ ]: d=boost.get_booster().get_score(importance_type= 'gain')
        l=d.items()
        x=[]
        y=[]
        for i in l:
            x.append(i[0])
            y.append(i[1])

        from matplotlib import pyplot as plt
        x_pos = np.arange(len(x))
        plt.rcParams['font.size']=20
        plt.figure(figsize=(30,10))
        plt.bar(x_pos, y , align='center', alpha=1)
        plt.xticks(x_pos, x,rotation=90)
        plt.ylabel('Importance')
        plt.title('Features vs Importance')

        plt.show()
```



```
In [ ]: RX_train=pd.concat([X_train,CY_train],axis=1)
        RX_test=pd.concat([X_test,Y_pred],axis=1)
        RSX_test=pd.concat([X_test,CY_test],axis=1)

        RX_test.rename(columns={0:'ARR_DEL15'},inplace=True)

        #Training the model for regression task
        from sklearn.linear_model import LinearRegression

        model=LinearRegression()
        model.fit(RX_train,RY_train)

        RY_pred=model.predict(RX_test)
        RSY_pred=model.predict(RSX_test)
```



```
In [ ]: from sklearn.metrics import mean_absolute_error
        from sklearn.metrics import mean_squared_error
        from sklearn.metrics import mean_squared_log_error
        from sklearn.metrics import r2_score
        import math as m

        #performance metrics for regression model
        smae=mean_absolute_error(RY_test, RSY_pred)
        smse=mean_squared_error(RY_test, RSY_pred)
        sr2s=r2_score(RY_test, RSY_pred)

        #performance metrics for regression model
        mae=mean_absolute_error(RY_test, RY_pred)
        mse=mean_squared_error(RY_test, RY_pred)
        r2s=r2_score(RY_test, RY_pred)

        print("Performance metrics for regression model\n-------------------------------------------")
        print("mean absolute error:",smae)
        print("root mean squared error:",m.sqrt(smse))
        print("r2-score:",sr2s)
        print()

        print("Performance metrics for whole model\n------------------------------------")

        print("mean absolute error:",mae)
        print("root mean squared error:",m.sqrt(mse))
        print("r2-score:",r2s)
```

```
Performance metrics for regression model
-------------------------------------------
mean absolute error: 8.058008166944312
root mean squared error: 10.770722923658475
r2-score: 0.9303379872773441

Performance metrics for whole model
------------------------------------
mean absolute error: 9.123398426216875
root mean squared error: 12.709693337978122
r2-score: 0.9029989457371684
```

VIII RESULTS

8.1 Results of classification model:

Accuracy score :	94.1%
Precision :	90.1%
Recall :	71.0%
F1 score :	79.4%
AUC score :	84.7%

Confusion matrix:

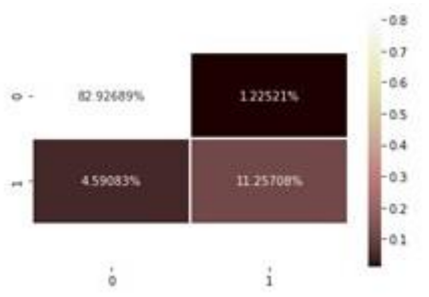


Fig 9 Confusion matrix

8.2 Results of regression model:

Mean absolute error	:	8.05 minutes
Root mean square error	:	10.77 minutes
R2 score	:	0.93

8.3 Results of overall model:

Mean absolute error	:	9.12 minutes
Root mean square error	:	12.70 minutes
R2 score	:	0.90

IX CONCLUSION

This project aims to predict the flight's delay along with the estimation of delay time in minutes using machine learning algorithms namely Decision Tree Algorithm (XGBoost) and Linear regression. Data set of both flight data and weather data will be taken to compare with the given inputs and validate them by applying classification and Regression concepts of Machine Learning. We have also done feature extraction, handling missing values using appropriate methods, sampling to handle imbalanced data and also tuning the hyper parameters with which we were able to achieve better accuracy.

REFERENCES

[1] Dand, Alok, Khawaja Saeed, and Bayram Yildirim. "Prediction of Airline Delays based on Machine Learning Algorithms." (2019).

[2] Yu, Bin, et al. "Flight delay prediction for commercial air transport: A deep learning approach." *Transportation Research Part E: Logistics and Transportation Review* 125 (2019): 203-221.

[3] Musaddi, Roshni, et al. "Flight Delay Prediction using Binary Classification."

[4] Kalliguddi, Anish M., and Aera K. Leboulluec. "Predictive modeling of aircraft flight delay." *Universal Journal of Management* 5.10 (2017): 485-491.

[5] Ding, Yi. "Predicting flight delay based on multiple linear regression." *IOP Conference Series: Earth and Environmental Science*. Vol. 81. No. 1. IOP Publishing, 2017.

[6] Chakrabarty, Navoneel, et al. "Flight Arrival Delay Prediction Using Gradient Boosting Classifier." *Emerging Technologies in Data Mining and Information Security*. Springer, Singapore, 2019. 651-659.

[7] Chen, Jun, and Meng Li. "Chained predictions of flight delay using machine learning." *AIAA Scitech 2019 Forum*. 2019.

[8] Ye, Bojia, et al. "A Methodology for Predicting Aggregate Flight Departure Delays in Airports Based on Supervised Learning." *Sustainability* 12.7 (2020): 2749.