**FLIGHT DELAY PREDICTION (A comparative Analysis) USING VARIOUS ML ALGORITHMS BASED ON ARRIVAL TIME, ARRIVAL DELAY AND VARIOUS OTHER FEATURES**

**A PROJECT REPORT**

*Submitted by*

PRIYANSHU PANDEY

**ABSTRACT**

A flight is claimed to be delayed when it arrives 15 or more minutes later than its scheduled time. Flight delays may cause a lot of inconvenience to passengers. It could make them late to their scheduled events or miss a connecting flight, thus leading to anger and frustration. Also, passengers may not always be entitled to compensation when a delay occurs. Airlines claim that few of the many reasons leading to most flight delays or cancellations are airline glitches, weather conditions, maintenance problems with the aircraft, congestion in air traffic, late arrival of the aircraft to be used for the flight from a previous flight and security issues. Growth in the aviation industry has resulted in air-traffic congestion causing flight delays. Flight delays not only have economic impact but also various harmful environmental effects. Air-traffic management is becoming increasingly challenging. The aim of our research work is to predict the delay of flights due to various factors using machine learning and deep learning so as to minimize losses and increase customer satisfaction. This Machine Learning model could be integrated with Airlines systems for the use of staff and customers and it could also rank Airlines and flights based on delays.

**ACKNOWLEDGEMENT**

I wish to express our deep gratitude to the below-mentioned people who have helped and supported us in the completion of this project. We would like to thank our instructors of the course.

We would like to extend our heartfelt gratitude toall my fellow mates of this course, for their valuable technical guidance, support, and encouragement rendered every stage of the course completion.

Several other people contributed immensely to the project. We sincerely acknowledge all their efforts which were indispensable to our project.

**PRIYANSHU PANDEY**

TABLE OF CONTENTS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CHAPTER** | | | **TITLE** | **PAGE NO.** |
|  | ABSTRACT | |  | iv |
|  | LIST OF TABLES | |  | xvi |
|  | LIST OF FIGURES | |  | xviii |
|  | LIST OF SYMBOLS, ABBREVIATIONS | | | xxvii |
| 1. | INTRODUCTION | | |  |
|  | 1.1 | MOTIVATION FOR THE PROJECT | | 1 |
|  | 1.2 | RESEARCH METHODOLOGY | | 4 |
|  | 1.3 | ABOUT PROJECT | | 5 |
|  |  | 1.3.1 | PROBLEM STATEMENT | 5 |
| 2. | LITERATURE REVIEW | | |  |
|  | 2.1 | FLIGHT DELAY PREDICTION SYSTEM USING MACHINE LEARNING | | 6 |
|  | 2.2 | AIRLINE DELAY PREDICTION USING MACHINE LEARNING AND DEEP LEARNING. | | 9 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 2.3 | FLIGHT ARRIVAL DELAY PREDICTION USING MACHINE LEARNING AND GRADIENT BOOSTING CLASSIFIER | | 11 |
|  | 2.4 | FLIGHT DELAY PREDICTION USING DATA MINING AND BIG DATA APPROACH | | 12 |
| 3. | SYSTEM ANALYSIS | | |  |
|  | 3.1 | EXISTING SYSTEM | | 13 |
|  | 3.2 | PROPOSED SYSTEM | | 13 |
|  | 3.3 | OBJECTIVES | | 14 |
|  | 3.4 | REQUIREMENT ANALYSIS | |  |
|  |  | 3.4.1 | HARDWARE REQUIREMENTS | 15 |
|  |  | 3.4.2 | SOFTWARE REQUIREMENTS | 15 |
| 4. | MODULES | | |  |
|  | 4.1 | DATA GATHERING AND PREPROCESSING | | 16 |
|  | 4.2 | DATA SPLITTING, WRANGLING, DISTRIBUTION AND VISUALIZATION | | 16 |
|  | 4.3 | HANDLING IMBALANCED DATA | | 19 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 4.4 | FEATURE SELECTION | | 21 |
|  | 4.5 | ML MODEL IMPLEMENTATION | |  |
|  |  | 4.5.1 | NAÏVE BAYES | 26 |
|  |  | 4.5.2 | SUPPORT VECTOR MACHINE | 28 |
|  |  | 4.5.3 | CLASSIFICATION AND REGRESSION TREES | 32 |
|  |  | 4.5.4 | RANDOM FOREST | 33 |
|  |  | 4.5.5 | RF CLASSIFIER | 35 |
|  |  | 4.5.6 | DECISION TREE | 35 |
|  |  | 4.5.7 | BAGGED DECISION TREE | 37 |
|  |  | 4.5.8 | LIGHT GBM (LGBM): | 37 |
|  |  | 4.5.9 | XGB CLASSIFIER | 39 |
| 5. | SYSTEM DESIGN | | |  |
|  | 5.1 | ARCHITECTURE | | 41 |
|  | 5.2 | UML DIAGRAM | | 42 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | 5.2.1 | SEQUENCE DIAGRAM |  |
|  | 5.3 | DATASET | | 43 |
|  | 5.4 | PROCEDURE | | 44 |
| 6 | TESTING | | | 46 |
| 7 | RESULTS | | | 49 |
| 8 | CONCLUSION | | | 52 |
| 9 | FUTURE ENHANCEMENTS | | | 53 |
|  | PAPER PUBLICATION STATUS | | | 53 |
|  | REFERENCES | | | 54 |
|  | APPENDIX | | | 56 |
|  | PLAGIARISM REPORTS | | | 65 |

**LIST OF TABLES**

7.1 Results table 46

LIST OF FIGURES

* + 1. A box plot 17
    2. A missingno matrix of a clean dataset 18
    3. Distribution plot for DEPARTURE\_DELAY column 18
    4. Balanced dataset display using histogram 20
    5. Dataset Heatmap 21

4.4.1 Implementation of Recursive Feature Elimination 25

* + 1. Bayes Theorem mathematical formula 27
    2. Classifier Plane Diagram 29
    3. Kernel Machine 30
    4. Classifier Model 31
    5. CART algorithm flow 33
    6. Random Forest Working Diagram 34
    7. A decision tree 36
    8. LGBM leaf wise tree growth 38
    9. XGBoost Level-wise tree growth 40
  1. Flow Chart Diagram 41
  2. Sequence Diagram 42
  3. Dataset 44
  4. Results as calculated 46
  5. Bar-Graph representation of Results 48

**ABBREVIATIONS**

**ML** Machine Learning

**API** Application Programming Interface

**CSV** Comma Separated Values **SVM** Support Vector Machine **XGB** eXtreme Gradient Boosting

**CART** Classification and Regression Tree

**KNN** K Nearest Neighbor

**RF** Random Forest

**CHAPTER 1**

**INTRODUCTION**

The aviation industry around the globe incurs huge losses due to various factors, Airline Delay can be considered as one of these factors. Airline delays tend to be onerous for every entity involved like airports, airlines and passengers. Flight delay is inevitable and it plays a vital role in each profits and loss of the airline agencies. Associate correct estimation of flight delay is important and of nice use for airlines as a result of the results area unit usually applied to increase client satisfaction and incomes of the airline agencies. As travel through air incorporates a vital role within the economy of agencies and airports, it’s necessary for them to extend the standard of their services. One in every of the foremost necessary trendy life challenges of airports and airlines area unit flight delays. Additionally to the current, delay in flights create passengers involved regarding varied matters and this in itself causes additional expenses for the agency and also the airdrome. Reasons that may be command answerable for the explanation for such flight delays may be named as: security, weather, shortage of components, technical and heavier-than-air craft instrumentation problems further as flight crew delays.

**1.1 MOTIVATION FOR THE PROJECT**

Delay in a very flight is inevitable, that not solely incorporates a heap of negative economic effects on passengers however on the agencies and airports further normally. Moreover, a delay on the wing will cause varied environmental issues through multiplied fuel consumption and emission of loads of harmful waste material gases. Additionally, the delay affects the trade, as a result of merchandise transport is extremely captivated with client trust, which may increase or decrease the price tag sales, so on time flight ends up in client confidence and multiplied variety of bookings. For an equivalent reason, flight prediction may be thought-about to predict a helpful call and operation for agencies and airports, and additionally an honest traveler system for the relative satisfaction of the client.

According to abundance and also the diversity of reasons that cause flight delays, there is a tendency to encounter a huge quantity of information that makes it not possible to be processed through previous ways of information analysis like classification, or the choice trees and age recent machine learning based mostly ways so as to method the info of this volume. In fact, the results obtained through the older ways don't seem to be correct, as a result of characteristics of older intelligent systems are designed by humans and frequently were customized. Also, folks seldom understand some options and frequently neglect most of the others out because of their sheer volume. On the opposite hand, in older learning processes, because the range of classes obtainable for classification augmented, the amount of issue additionally augmented and also the extraction of necessary and effective options became comparatively not possible.

Due to the on top of mentioned discrepancies, there are a unit several researches on modeling and predicting flight delays, wherever most of them are attempting to predict the delay through extracting necessary characteristics and most connected options that may facilitate in determinative somewhat precise outputs. However, most of the projected ways don't seem to be thought of correct enough owing to the large volume information, dependencies and extreme range of parameters.

There are several studies during this field. For instance, older Regression ways are accustomed reason delay propagation. For this model, the destination delay is very passionate about arrival of flights and also the effective factors include: day, time, flying field capability and a few factors are associated with rider masses. This model employs factors associated with the flying field sort, airplanes sort, date, time, flight path, and flight frequency for network coaching still as non-linear and linear for dataset analysis. Because it is troublesome to interpret neural network parameters, the assorted factor’s behavior and most significantly, the verification of these factors on the wing is extraordinarily troublesome. Moreover, older intelligent algorithms typically use shadow learning models to resolve conditions with huge information techniques in sophisticated classifications. However, results of this analysis area unit terribly totally different in comparison to the perfect conditions. Though the model styles will have a decent or dangerous scenario, the response is very passionate about the expertise and this procedure needs an excessive amount of time. Therefore, ancient simulation and modelling techniques don't seem to be appropriate or maybe economical for determination such issues. There are a unit numerous in progress subjects of study that work on determination this downside and this model additionally has tried to use trendy techniques and ways to assist predict higher ends up in this field.

One of the latest trendy ways in determination such extended and complex issues amid large information or voluminous dataset that has been thought of by several scientists’ area unit deep neural networks. The look of this learning technology is galvanized from human neural networks and may be a branch of machine learning and assortment of algorithms that attempt to model high-level abstract contents through application learning in many layers and levels. Therefore, this subject allows the deep learning neural networks to method a large information set in an exceedingly sophisticated data classification. Moreover, this structure is adequate for extracting a number of the characteristics, in order that the educational is capable enough to extract most range of potential characteristics. The stratified network structure and its capability of computation for every information scale has crystal rectifier to progressing application of techniques. These networks have differing kinds of computation techniques and might be applied counting on that technique is most fitted for the involved downside. Some recent studies depict works and techniques for determination issues by using the continual Deep Neural Network and their results have a high accuracy on the wing delay predictions.

Precise and meticulous prediction of Airline delay mistreatment the factors that play prodigious roles with the assistance of many algorithms and mathematical calculations are going to be the key to reduce the losses and additionally to extend the client satisfaction.

The project was initiated by reviewing previous works associated with our topic and explained these works very well within the “Literature review” section of Chapter a pair of. A whole description of the analysis method and additionally the restrictions of these processes were studied so as to be told from them and apply our understanding on this model to predict results with most potential accuracy.

This project works on representing a model supported machine learning and its algorithms that think about the effective factors accountable for the delay of flights. During this model, many formulas are utilized to provide a comparative study with relation to the accuracy of every algorithm. Machine Learning, attended with its numerous accuracy predicting algorithms, is one amongst the latest ways utilized in determination issues with high levels of complexness and large amounts of knowledge. Moreover, machine learning is additionally capable of mechanically extracting the vital options from large volumes of knowledge. To analyze the pattern of flight delays throughout numerous years thanks to numerous reasons, such ways are applied on flights’ knowledge obtained from kaggle’s 2015 forrader knowledgeset that contains unclean data i.e. null values, redundancies and unrequired data so as to form a clean dataset with applicable values and needed data, knowledge pre-processing and have choice ways square measure applied. The preciseness, accuracy, recall and F-measure of the varied algorithms is measured to match the results. The results additionally show that accuracy of the projected model in prediction flight delays supported the input file and elite options.

**1.2 RESEARCH METHODOLOGY**

The flight delay prediction system, when built, goes through these various steps in order. A flow chart diagram is attached below to help visualize the same, Figure 1.1:

* Collect the data from the airline company website
* Pre- process the data properly to remove any null or unnecessary data
* Perform data visualization, to know the data distribution in the dataset, and to remove any kind of null data that may be present.
* Perform data balancing if required.
* Once the data has been cleaned and visualized, comes the feature selection process. Before this step, it is important everything about our dataset, i.e the kind of data, be it object, float or anything else. Feature selection method is applied to pull out the features in that dataset that would give the best results for our project.
* Once the important features have been selected, split your data into training and testing groups.
* Apply the different machine learning algorithms on these training and testing datasets to view the results.
* Capture the results to compare the accuracies of different machine learning algorithms.

The dataset is collected from the airline website, after which it is pre-processed before being used for the project. Once the processed data is gathered, a feature selection algorithm (Recursive Feature Elimination) is applied to get the features from the original dataset that would give the best results for our project. After that data is split into testing and training datasets before applying the various machine learning algorithms to the dataset. Finally the different algorithms are compared, based on various factors like Accuracy, f1-score etc., to check which algorithm gave the best results for the taken dataset.

**1.3 ABOUT PROJECT**

The aviation industry around the globe incurs huge losses due to various factors, one of these factors is Airline Delay. Airline delay tends to be onerous for every entity involved i.e. airports, airlines and passengers. Precise and meticulous prediction of Airline delay using the factors which play prodigious roles will be the key to minimize the losses and increase customer satisfaction. In the paper, several machine learning and deep learning algorithms have been employed to produce a comparative study with respect to the accuracy of each algorithm.

**1.3.1 PROBLEM STATEMENT**

Some of the main goals pertaining to this project can be:-

* Predict the delay of flights due to various reasons.
* Increase customer satisfaction while minimizing losses due to flight delays.
* To create a highly accurate model for flight delay prediction using ML and Deep Learning algorithms.
* Reduction of magnitude of Air Traffic Congestion that may add to the delay of flights.
* To determine how different combinations of feature selection affect the accuracy of the model.

Our objective is to analyze and predict flight departure delays for a sample of flights and optimize different models to get best possible predictions for the same.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 FLIGHT DELAY PREDICTION SYSTEM USING MACHINE LEARNING**

This paper is based on a flight delay prediction system which focuses mainly on predicting the delay of a flight based on the weather situation during that particular time. Essentially supervised learning is used to teach or train the machine exploitation data which is well tagged which means some of the data from the dataset is already labeled with the correct answer. After that, the machine is fed a new set of examples so supervised learning algorithmic rule analyses the coaching knowledge on the new set of examples and produces a correct outcome from tagged data. Using a supervised machine learning approach, the labeled data gives it authenticity according to its learning. The Naïve Bayes model is one of the algorithms which is used for the real time prediction. It then returns the predicted weather data using an AP (application programming interface) and passes the data into the algorithm. The attributes considered for calculations and taken by the AP can be taken as one or all of the follows: weather, temperature, humidity, rain in mm, visibility and month number. The supervised machine learning is based on having a set of correct labeled data from which the algorithm analyses the data and makes its prediction. A CSV file is used for storing that said data. Other methodologies that can be used include Feed forward network, Neural Network, Random Forest, decision trees, logistic regression, Regression Tree, etc.

This paper focuses on flight delay predictions caused by the weather. Though weather are the foremost reasons for flight delay, different unexampled events like major calamities, natural or synthetic also can cause major delays in any flight that don't seem to be thought-about for this method [1].

Considering the recording equipment characteristic of the SVM rule, a sensitivity analysis was performed to assess the connection between the dependent and also the instructive variables. The feature choice of this paper lined varied aspects in regard to flying field

ground operation, delay, demand-capacity, weather information, and flow management characteristics. The variable impact analysis revealed that factors such as ground delay program, pushback delay, taxi-out delay and demand-capacity imbalance with probabilities of 0.506, 0.478, 0.339, and 0.338, respectively, are significantly associated with the flight departure delay in New York City airport. These findings provide insight for better understanding of causes of the flight delays and helped in analyzing the various flight delay patterns.

The scope of this study is limited to departure delays for the primary New York airports: EWR, JFK, and LGA and hence the models and practices researched about in this paper may not be applicable to other airports in the USA, let alone other countries. In this research paper just the impact of the variables on a high delay level or of a higher ranking is presented. So, the variables that have a rather lower level significance ranking in terms of causing flight delays were not considered in this paper and were discarded [2].

In this paper, a model for estimating flight departure delay distributions caused by air traffic congestion prediction models is developed. Identification and study of the major factors influencing flight departure delays and developing a strategic departure delay prediction model has taken place. This model employed non-parametric methods for daily as well as the seasonal trends. In addition, the said model used a mixture distribution to estimate the residual errors. In order to overcome problems with the local optima in this mixture distribution, the paper focuses on developing a global optimization version of the Expectation Maximization algorithm, borrowing ideas from the Genetic Algorithms.

The long analysis objective of the model could be a fairly complete overhaul of the present mechanism for predicting the airspace congestion that causes flight delays. However, the model also can be wont to predict flight delays thanks to different variables and in regions apart from capital of Colorado International flying field however solely up to a restricted extent which may be thought-about as a disadvantage [3].

In this paper, the author checked out the various Machine Learning techniques/ algorithms to do to predict if a flight are delayed or not before it's even declared on the departure boards. The paper doesn't specialize in achieving the best potential accuracy as a result of in step with the author constant may be simply achieved by simply adding a list of features/ classes which will bias the model in terms of prophetic power. Thus this data was checked out as a section of Associate in Nursing alpha information Analysis (EDA), however was taken out of the most models implying that the paper principally focuses on delays caused by incoming and outgoing delays. This could be explained by the very fact that if a plane leaves late from a precise location, chances are high that that it'll even be incoming late at its destination. However, this may not invariably be true since airlines, sometimes, try and account for such delays by reducing their time period. Thanks to constant, the predictions ar created before the delays are declared on the departure boards and before a traveler boards the plane. Methodologies employed in this paper embody adenosine deaminase boost, random forest, bagged trees, random forest with bootstrap category weight, and deep neural networks.

The dataset collected for this paper is limited to the US Domestic flights and hence the prediction of flight delays can only be obtained for the same. It also does not consider other variables apart from departure/ arrival delays that may be the reason behind the delay of flights. The passengers cannot get to know about the delays well advanced in time which makes this model less desirable [4].

This study was transformed into an application in order to predict the chances of a flight being delayed for a specified interval of time. In this study, the time slot for which a flight may be delayed, and the chances of the flight being delayed for that duration was studied, and the results for the same were observed. Different software’s like AWS and spark were used for this application. The problem was defined as a multiclass classification problem for this paper. Techniques like AWS and LightGBM service like Sagemaker were used for the working of this application [5].

This study was done in order to categorize the different ways that a flight delay prediction study can be done. It was said that the flight delay prediction can be broken down into two main categories which are delay propagation and root delay as well as cancellation. The authors also formulated the required data types and data management for this study. Five main methods were proposed for this modelling which can be classified under the following names: statistical analysis, probabilistic model, network representation, operations research, and machine learning [6].

**2.2 AIRLINE DELAY PREDICTION USING MACHINE LEARNING AND DEEP LEARNING**

In this paper, a prediction model was used to classify the flight delays influenced by bleak weather order. A model is constructed on the datasets obtained from both the flight delays and the weather data sets and a sampling technique is applied to balance that data. Machine Learning is used to improve models with voluminous amounts of dataset like flight dataset and weather dataset. Algorithms such as Ada Boost, KN neighbors are applied to construct models that can predict whether the flight will be delayed or not. Extra tree classifier, Decision tree and Random forest algorithms were applied on the balanced data to predict flight delays in order to obtain a better accuracy. Flight data and weather data are merged together and fed in the model which can then accomplish a binary classification to predict the flight delaysi. Accuracy can furthermore be increased/ improved by using deep neural networks. This paper focuses on the flight delays caused by bleak weather conditions and their predictions [7].

The researchers of this paper used numerous machine learning and deep learning ways like K-means, Support vector Machine, Random Forest and etc., and regarded numerous factors that will cause a delay on the wing, might or not it's because of climate or NAS delay. The output of the study was that Random forest was the algorithmic program that provides the most effective results for the given dataset. This study will facilitate to predict the delay of flights with efficiency, serving to each the purchasers and also the airline trade by creating correct predictions. It additionally helps in crucial the assorted reasons behind the reason behind the delay of a flight. [8].

The study during this paper was created mistreatment the information Mining and Machine Learning techniques to predict the delays for native airlines by the name of yank Airline and for the highest five busiest airports. Techniques like Gradient Boosting Classifier and Hyper-parameter calibration were accustomed build predictions during this study. It involves applications of additional advanced pre-processing itechniques, isampling algorithms iand iMachine Learning, iDeep iLearning Hybrid Models ituned iwith Grid irummage around for iachieving higher model iperformance and better accuracies [9].

This analysis foremost has extracted necessary characteristics so used the extracted data for each neural networks and deep belief networks through impulsive samples to coach the modeli. The model utilizes reminder and Resilient Back Optimized Propagation so as to point out that the Resilient back propagations area unit faster than the rear propagation and as a result the model coaching and consequently has been increased . [10].

This study analyzes the high-dimensional data from Beijing International Airport and presents a practical flight delay prediction model based on data from the same. Using a multifactor approach, a deep belief network method is deployed to detect the inner patterns of the flight delays. Support vector regression is a method that is embedded in the developed model to perform a supervised fine-tuning within the presented predictive architecture. The proposed method has proven to be highly capable of handling the challenges of large datasets and capturing the key factors responsible for the delays. This ultimately enables the connected airports to collectively alleviate delay propagation within their network through collaborative efforts (e.g., delay prediction synchronization) [11].

This study researched the Deep Learning Approach using Artificial Neural Network (ANN) and also introduced a new type of multilevel input layer ANN which is interpretable. Machine Learning is used to improve models with bulk amounts of datasets like flight dataset and weather dataset. Ada Boost and KN neighbor’s algorithms are applied to construct models that can predict whether the flight will be delayed or not. Extra tree classifier, Decision tree and Random forest algorithms are applied on the balanced data to predict the flight delays in order to obtain a result with better accuracy. Flight data and weather data are merged together and fed in the model which can then accomplish a binary classification to predict the flight delays. Accuracy can furthermore be improved by using deep neural networks in the same model [12].

**2.3 FLIGHT** **ARRIVAL** **DELAY** **PREDICTION** i**USING** **MACHINE LEARNING AND GRADIENT BOOSTING CLASSIFIER**

Chakrabarty, Navoneeli, et al projected a Machine Learning Model victimisation Gradient Boosting Classifier for predicting arrival delays of flight of the yank Airlines covering the five busiest airports within the USA. The paper does not focus on achieving the highest accuracy possible because according to the authors, the same can easily be achieved by just adding a series of featuresi/ categories that will bias the model in terms of its predictive power. So, this information was looked at as a part of Exploratory Data Analysis (EDA), but was taken out of the main models implying that the paper mainly focuses on delays caused by “departure delays” and “arrival delays” and other features were given lesser priority [13].

This study was done by employing the LightGBM technique of machine learning. A very important point discussed in this paper was the very fact that there is very less data on the reasons for the delay of a flight due to weather as there is very less meteorological data about it. Flight delay predictions are mainly focused on airport-based data only. It can also be considered as a future scope of this study [14].

**2.4 FLIGHT DELAY PREDICTION USING DATA MINING AND BIG DATA APPROACH**

A Big Data Approach by analyzing and mining the flight information AS well as its corresponding weather conditions using parallel algorithms was implemented as MapReduce programs were executed on the Cloud Platform for weather induced flight delay predictionsi. The results show a high accuracy in predicting the delays in flights above a given thresholdi. For instancei, with a delay threshold of 60 minutes we can achieve an accuracy of 85.7% and a recall of 86.9% on the delayed flights [15].

**CHAPTER 3**

**SYSTEM ANALYSIS**

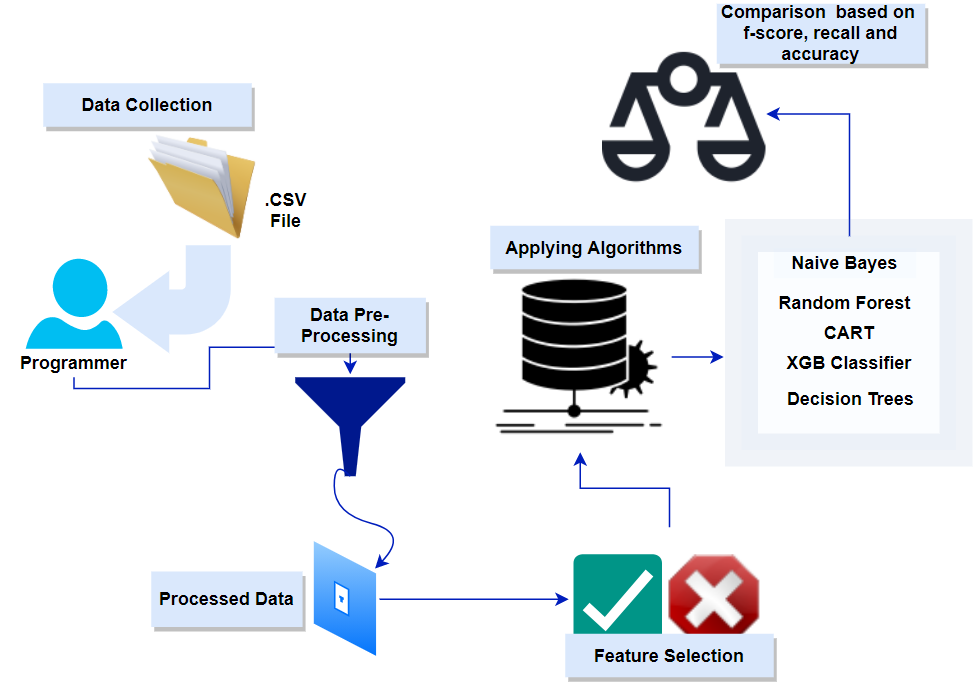
**3.1 EXISTING SYSTEM**

The current system present with us is as follows:

* Flight delays are inevitable and play a vital role in the benefits and losses of aircraft.
* Accurate measurement of the aircraft delays is important for airline industry as these results can be used to increase the fulfilment of customer needs and revenue for the airlines.
* There has been a lot of research on modeling and approach towards prediction of flight delays, with maximum of the works trying to predict delays by pulling out the important features and closely bonded features.
* However, most of the research works done were not up to the mark in case of having a good accuracy due to the lack of large volumes data, dependence and excess number of parameters.
* Also, the airlines have their own delay predictions system. The customers get to know that their flight has been delayed once they arrive at the airport.

**3.2 PROPOSED SYSTEM**

In this project, procurement of dataset is the first and most important step towards preparing a model. The data used here is acquired from the Kaggle’s 2015 airline delay dataset. Then comes the Data Pre-processing phase, where the data wrangling is performed to change the data into desired format. Cleaning of data i.e., to remove the tuples with null values, checking for redundancies, dropping irrelevant details, etc. is carried out in this phase. The processed data is then used for extracting features (using the feature extraction algorithms) which are relevant and needed for the training and testing phase. The training and testing phase is carried out in each of the machine learning algorithms used. Then Voting takes place where the values like f-score, recall, precision and accuracy of all the machine learning algorithms are compared. The model with the best accuracy is chosen to be the ideal ML model for predicting flight delays.



**3.3 OBJECTIVES**

Incorporating the idea of Machine Learning so as to give effective flight delay predictions for the customers and the Airline Industry.

1. To have one single flight delay prediction technique that can be made handy not only for passengers, but for every company in the aviation industry, as they are codependent.
2. To reduce flight delays as the financial losses incurred by the industry, flight delays also portray a negative reputation of the airlines, and decreases their reliability.
3. To address a variety of environmental concerns, such as rising fuel consumption and carbon emissions.
4. To have an analysis that predicts delays based on the previously available data, based on their on-time results and delays over time, the peak hours of delay are shown.
5. To provide a model that every aviation authorities could use as a template for their own advantage, including the Context of India.
6. To construct a framework that can be used as an efficient method or even a practical example when studying delay analysis using real dataset provided.

**3.4 REQUIREMENT ANALYSIS**

**3.4.1 HARDWARE REQUIRED**

* A high-end working system to run the model
* WIF or Internet connection

**3.4.2 SOFTWARE REQUIRED**

* Anaconda
* Python
* Jupiter Notebook

**CHAPTER 4**

**MODULES**

**4.1 DATA GATHERING AND PREPROCESSING**

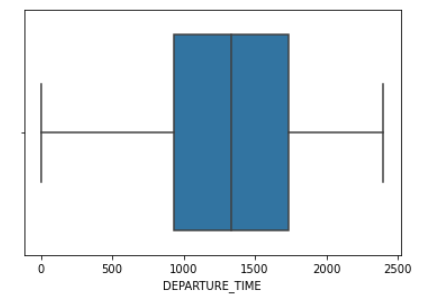
It’s the very first step that needs to be performed while doing a project. A numerous amount of data is available on the internet nowadays on various topics. People have worked to collect large amounts of data and tried to make it available for the use of others in one single place. One such site is the Kaggle data warehouse. Our dataset is retrieved from the Kaggle site itself. The said data is a flight delay dataset, from the year 2015. The dataset contains various columns like Year, Date, and Flight number, Delay time and many other columns which we call features.

Once the dataset has been chosen that fulfils our requirement, the dataset was downloaded in the form of .CSV file and imported into our Jupiter notebook. Due to limitations on the processing power, only 20,000 tuples randomly from the dataset. Next, the data was cleaned and all the Null values were removed. After this step, the clean data is achieved with which the the project is proceeded.

**4.2 DATA SPLITTING, WRANGLING, DISTRIBUTION AND VISUALIZATION**

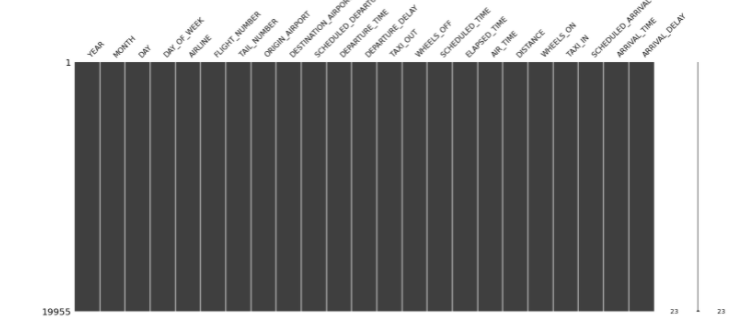
After achieving clean data after preprocessing, the data is converted in such a way that suits our needs. Firstly, the object type data were converted into float as it was the requirement that needed to be satisfied for the feature selection process.

After that, different types of plots, like box plot, distribution plot or missingno matrix are drawn, to check the distribution of data in the datasets. Here, the box plot is taken from the seaborn package of python, and it depicts the quantitative data spread in such a manner in which it aids in the comparison of unambiguous input parameters. Except for the points that are calculated to be outliers using a method that is a function of the inter-quartile scale, the box represents the dataset's quartiles, while its whiskers reflect the rest of the distribution. The below figure 4.2.1 represents a box plot for the variable “DEPARTURE\_TIME”:



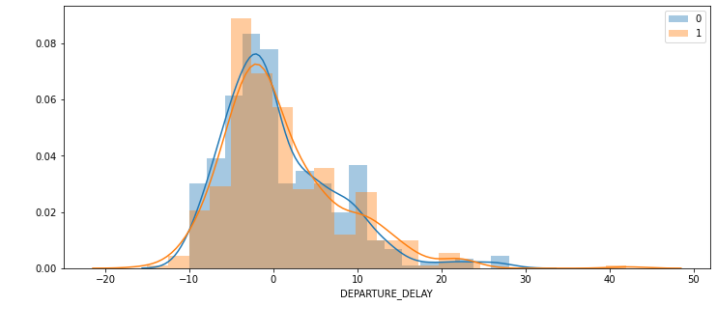
**Figure 4.2.1:** A box plot

In case of a missing no matrix, it represents the amount of missing data in the database for each particular column. It helps in visualizing the dataset at a glance. It gives the objective to remove all the clutter from the dataset before moving ahead in the project. The below figure 4.2.2 represents the missingno matrix for our dataset once all the null values were removed:



**Figure 4.2.2:** A missingno matrix of a clean dataset

The distribution plot is used basically for a univariant set of observations and visualizes it with the help of a histogram. It comes with lines, or curves to show the data distribution in the column. The figure 4.2.3 below depicts the distribution plot for the column “DEPARTURE\_DELAY”:

**Figure 4.2.3:** Distribution plot for DEPARTURE\_DELAY column

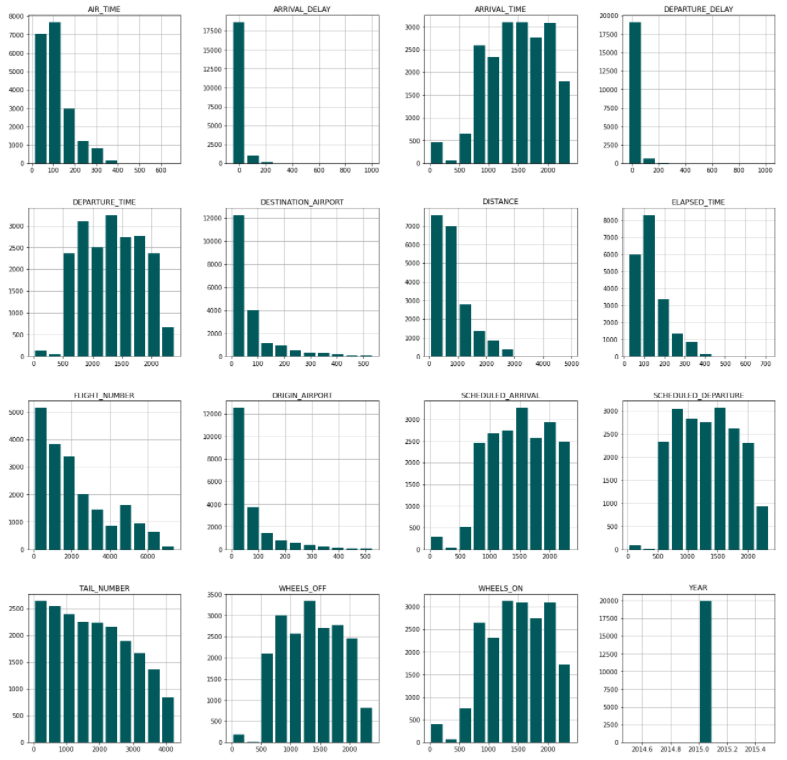
This module plays a major role in the project as it sets up the ground on which the project is further built.

**4.3 HANDLING IMBALANCED DATA**

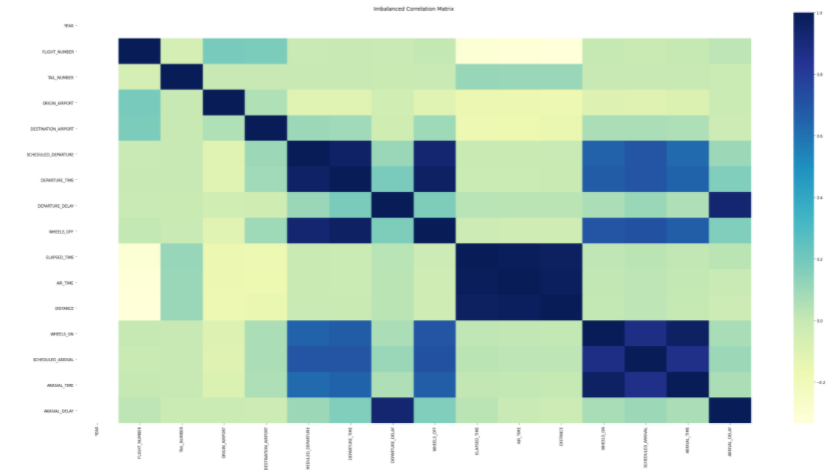
With all the data within the preferred data types, there still can be one problem that would hinder the project. Sometimes, datasets do not have the right kind of data distribution. For example, if in our dataset, the majority of the flights get delayed then, may get a very high result, or vice versa is true. So, this data imbalance can totally give different results than expected. This is where data balancing comes into play. A balanced data gives much better and reliable results. In this case, the data distribution is great and the dataset is balanced enough for use. In the below figure 4.3.1, histogram plot is used to show the data distribution of different features in the dataset. The diagram depicts the distribution of data in each of the columns of the dataset. The figures help understand the dataset well.

A histogram plot shows the frequency distribution of the data in a given column of the dataset. It is a graph that depicts the number of observations within each given interval. It's a precise approach for displaying numerical data distribution graphically. To make a histogram, start by making a bin of ranges, then divide the entire range of data into a predefined sequence, and finally calculate the values that fall into each interval. Bins are described as non-overlapping, consecutive intervals of variables.

A heatmap was also plotted to visualize the data distribution. A heat map is a visual representation of relevant data, graphically. Using colors to visualize the value of the matrix. A colored map is achieved as output that describes the data distribution in the dataset. The figure 4.3.2 displays a heatmap that shows the data distribution of our dataset.



**Figure 4.3.1:** Balanced dataset display using histogram



**Figure 4.3.2:** Dataset Heatmap

**4.4 FEATURE SELECTION**

Once the data balancing part is over, another important part of the project begins, i.e. the feature selection process. This module contains information about what exactly is feature selection, what is it important, and what are the different feature selection processes.

When constructing a machine learning model in the real world, it's almost unheard of for all of the variables in the dataset to be useful. Introducing typically reached variables will reduce the model's segmentation accuracy as well as the classifier's overall accuracy. Furthermore, as even more attributes are added to a model, the model's complexity grows. According to ‘Occam's Razor's Law of Parsimony,' the explanation with the fewest features is the better explanation throughout every situation. As a result, feature selection becomes an essential part of developing machine learning algorithms. The aim of feature selection approaches in machine learning is to find the best collection of features for building effective method of the phenomenon under investigation. In machine learning, feature selection strategies can be narrowly divided into the following categories:

* **Supervised Techniques**: These are methodologies for labelled data that can often be used to classify relevant features to improve the performance of prediction models such as regression and classification.
* **Unsupervised Techniques:** These are approaches that can be used on data that has not been labelled.

These methods can be classified as follows from a genus standpoint:

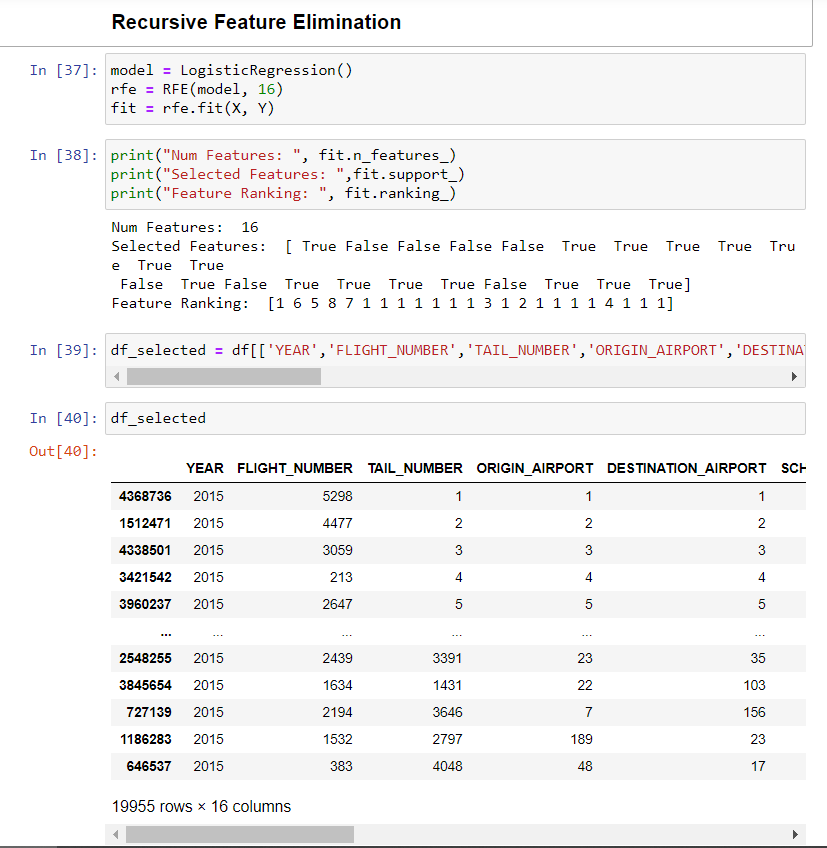
**A. Filter methods**: In the Filter techniques, only the features' inherent characteristics which are chosen instead of intra- and inter validating the results, it was calculated using univariate statistics. This method of function collection is a lot simpler and smoother to implement and take computationally less time than the approaches of the wrapper. When coping with large amounts of data, these filter methods are computationally much cheaper to use. Some of the tests to figure out the performance of these techniques can be classified as:

1. **Chi-square Test:** When categorical characteristics are found in a dataset, the Chi-square test is used. By calculating the Chi-square among the feature and the chosen target variable, the optimal number is picked of features with the highest Chi-square values. When using the chi-square test to test the relationship between different attributes in the dataset and the decision feature, the following conditions must be met: the parameters must be categorical, tested separately, and quantities must have a predicted variance greater than 5.
2. **Fishers Score:** Considered to be among the most commonly used tools for function selection for supervised learning algorithms. The algorithm which will be used in this model reverts the vector ranks in decreasing order depending on the fisher's ranking. The variables can then be chosen according to the situation.
3. **Correlation Coefficient:** A calculation of the linear association between multiple variables is known as correlation. It can inferred that one attribute from the others present using correlation. The good variables are known to be strongly correlated with the objective, which is how the correlation functions with feature extraction. In addition, variables must be associated with the objective but unrelated with one another. As a result, if two attributes are associated, the model only needs one of them, as the other adds no additional value.
4. **Variance Threshold:** A basic baseline solution for function selection is the variability limit. It eliminates all functionality whose deviation is below a certain threshold. As a result, by implication, all nil variability functions are excluded, which are features which have the similar value for all samples. It is presumed that factors with such a greater heterogeneity provides more valuable knowledge than those with lower variance, but one of the disadvantages of the filter approach is that we don't take into consideration the relationship between feature & output variable.
5. **Mean absolute difference (MAD):** The absolute deviation from the dataset's mean value is computed using the mean absolute difference (MAD). The inclusion of the square in the deviation and MAD measurements is the primary distinction. The MAD is a scale variation, much like the variance. This implies that as the MAD importance rises, the value of discriminatory force rises as well.
6. **Dispersion Ratio**

**B. Wrapper methods**: Wrappers need a way to look at the space of all potential function subsets and evaluate their consistency by studying and comparing a classifier for certain subsets. The procedure for selecting important features in this type uses different ML algorithms so as to fit in according to the chosen dataset. In this method, an evaluation criterion is set, and a greedy search approach is used to evaluate all the possible combination of features. When compared to filter methods, the wrapper methods have better feature selection accuracy. Some of the tests to figure out the performance of these techniques are as follows:

1. **Forward Feature Selection**: It is a repetitive process in which is begun with the highest scoring function and work our way up to the desired value. Then new variables are chosen continuously until the one that produces the best results when combined with the first is found. This procedure is repeated until the previously specified criteria is fulfilled.
2. **Backward Feature Elimination:** This strategy operates in the same way as forward function selection does, but in the opposite direction. It begins with all the available features and create a model from there. As a result, the model variable with the best measurement results calculates meaning. It keeps on going until we've met the requirements we're looking for.
3. **Exhaustive Feature Selection:** This is one of the most reliable feature collection techniques we've seen so far. Each feature subset in the dataset goes through Brute-force evaluation. Therefore, in this method, every possible feature combination is tried out, and the best performing combination is selected.
4. **Recursive feature elimination:** The aim of the recursive feature elimination (RFE) technique is to pick factors by iteratively examining fewer sets of features and assigning weights to the features using an independent estimation method. The estimator is first centered upon the early collection of features, and the value of each feature is then determined using either a coefficient index or a weighted linear attribute.
5. **C. Embedded methods**: Through incorporating connections between features while retaining fair computational costs, these approaches combine the advantages of both the wrapper and filter methods. Engrained strategies are recursive as in context that they pay attention to every phase of the project testing process and selectively excavates the features that add most to the development for such variation.
6. **LASSO Regularization:** Regularization is the process of applying a constraint to the varying factors of a ML algorithm in order to minimize the model's independence and prevent computational burden. The penalty is imposed over the parameters that multiply each of the predictor variables in a linear model regularization. Lasso or L1 does have the ability to be able to compress any of the parameters to 0, which distinguishes it from other regularization strategies. As a result, the function can be deleted from the model.
7. **Random Forest Importance:** Random Forest is a variant of the Bagging Algorithm that combines a set of decision trees. The random forests' tree-based techniques are typically ranked by how much they increase the uniqueness of a node, or by how much they reduce imperfection (Gin impurity) across all trees. The nodes with the highest reduction in imperfection are now at the beginning of the trees, whereas the nodes with the lowest decrease in impurity are at the top. We may construct a branch with the most significant features by trimming these trees beyond a certain node.
8. **D. Hybrid methods**: Combination of two or more of the above-mentioned methods.

This project involves the implementation of Recursive Feature Elimination technique to get the important features for our project. On applying the technique on our dataset, 16 important features were attained of 23 features that were already present in the dataset earlier. The figure 4.4.1 below shows the results of the feature elimination process.



**Figure 4.4.1:** Implementation of Recursive Feature Elimination

**4.5 ML MODEL IMPLEMENTATION**

Once the feature selection process has been completed, a temporary dataset is made to store the data with those features. The dataset is then divided in the ratio of learning to test results i.e., 4:1 (training:80%, and testing 20%). Various Machine Learning algorithms are applied and start recording the results. The algorithms were as follows:

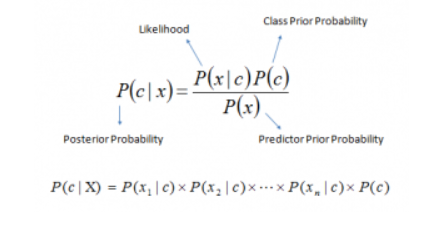
**4.5.1 Naive Bayes**

The Naïve Bayes algorithm is a classification technique based on the Bayes theorem, which says that the presence of one feature in a dataset is independent of the presence of the other feature. In simple words, the Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature (not necessarily of the same class), thus making it better than the other algorithms.

For example, apple can be considered as a fruit if it has the following characteristics like, it is red in colour, round in shape, and about 3 inches in diameter. Even, in the case where these features depend on one another or upon the existence of the other features in order to differentiate it as an apple, all the features have their own significance and importance to the probability of this fruit being an apple and that is why it is known as ‘Naive’.

Naive Bayes model can be built easily, and is especially useful when the dataset is huge. It reduces the complexity of the problem to a large extent. Along with the simplicity, one striking point about Naive Bayes is that, it also is the most advanced classification approaches used in most Machine Learning techniques are considered to underperform Bayes.

Bayes theorem uses the below mentioned formula determine the likelihood of the outcome i.e. P(c|x) from P(c), P(x) and P(x|c). The mathematical formula is given in the figure 4.5.1 below:



**Figure 4.5.1:** Bayes Theorem mathematical formula

**Pros:**

* It is simple and quick to estimate the sample data sets' class. In a cross-class prediction scheme, it also works well.
* When the presumption of freedom is valid, a Naive Bayes classifier outperforms other frameworks such as logistic regression, and it needs fewer training samples.
* In comparison to numerical input variables, it works better for categorical input variables (s). A regular distribution is considered for quantitative variables like for example bell curve, which is a strong assumption.

**Cons:**

* If an attribute in the test data set has a segment that was not part of the training data set, the approach will give a 0 (zero) likelihood and would be unable to foresee anything. This is commonly referred to as the "Null Frequency." The smoothing technique called as the Laplace estimation is one of the most basic.
* Nave Bayes, on the other hand, has been rated as a weak appraiser, but its estimates can be taken with a pinch of salt.
* In addition to the shortcomings of Naive Bayes, one of them might be the presumption of significant predictors. In fact, obtaining a collection of totally autonomous determinants is practically impossible.

**4.5.2 Support Vector Machine (SVM):**

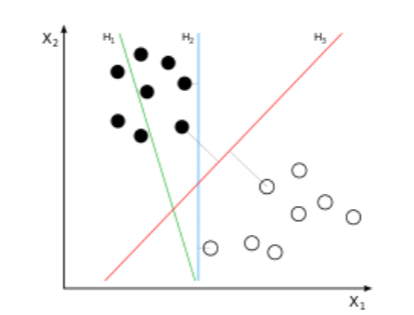
**4.5.2.1 Introduction**

In machine learning, support vector machines (SVM) are a sorted supervised learning model with related learning algorithms that dissect information utilized for classification and regression investigation. Given a lot of preparing models, a SVM preparing model assembles a model that doles out new guides to one class or the other one, and makes it a non-probabilistic paired linear classifier. A SVM architecture is a portrayal of the models as focused in space, mapped with the goal that the instances of the different classifications are partitioned by a space that is as wide as could reasonably be expected. New models are then mapped into that equivalent space and anticipated to have a place with a classification dependent on the side of the hole on which they fall.

Notwithstanding performing linear classification, SVMs can effectively play out a nonlinear classification utilizing what is known as the kernel stunt, certainly mapping their contributions to high-dimensional element spaces.

At the point when information is unlabeled, supervised learning can’t, and an unsupervised learning approach is required, which endeavors to discover common clustering of the information to gatherings, and afterward, maps new information to these framed gatherings. The support-vector clustering algorithm, made by Hava Siegelamann and Vladimir Vapnik, applies the measurements of support vectors created in the support vector machines algorithm, to order unlabeled information and is one of the most generally utilized clustering in mechanical applications.

**4.5.2.2 Classifying data**

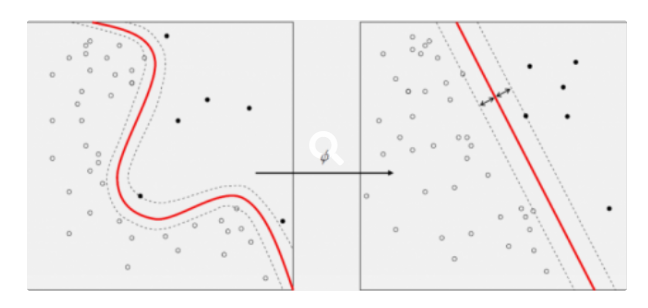
In machine learning, classifying knowledge is a common task. Consider that you have a set of information targets, each of which is assigned to one of two classes, and the goal is to determine which class the other data point would be assigned to. By virtue of Support-Vector Machines, a data is aimed and viewed vector with n dimensions (a summary of n numbers), and need to know if those distributions can be separated using a (n-1) - dimensional hyperplane. This one is referred to as a linear classifier. There are also several hyperplanes that can be used to group the results. The hyper - plane that tackles the largest portion, or edge, between the two groups has a fair choice as to be the best hyperplane.

**Figure 4.5.2:** Classifier Plane Diagram

In order to intensify the delay projection to the closest point of information on either side, subspace is selected. If such a hyperplane is present, it may be referred to as the most serious edge hyperplane and it is referred to as the largest edge classification by the linear classifier it reflects; or, otherwise, the best solid classification.

**4.5.2.3 Definition**

A vector holder constructs a hyperplane or a series of hyperplane systems in a large or never-ending dimension that can be used for classification, regression or for various tasks such as detection of anomalies. The hyperplane, which has the best part in preparing knowledge behind every classes (a proved edge), is instinctively boiled out to create an incredible package and if anything, else fails the more popular the border, the less the classification’s theory error. Even if the first problem can be represented in a restricted dimension, it always occurs that the sets to be separated in that space cannot be linearly distinguished. Therefore, the first minimal dimensional space was proposed to be transformed into a much higher dimension space, which would possibly simplify the deduction in that void. For a sensitive computer charge, the mappings used in SVM's plans are aimed at ensuring that the speck effects of sets of vectors are efficiently presented in relation to the factors in the initial space, as far as the kernel function chosen to address the problem is characterized. In a higher dimensional space, the hyperplanes are defined as a series of points whose vector product is constant in that space, where a number of those vectors are orthogonal vectors that are a hyperplane defining (and therefore minimal).

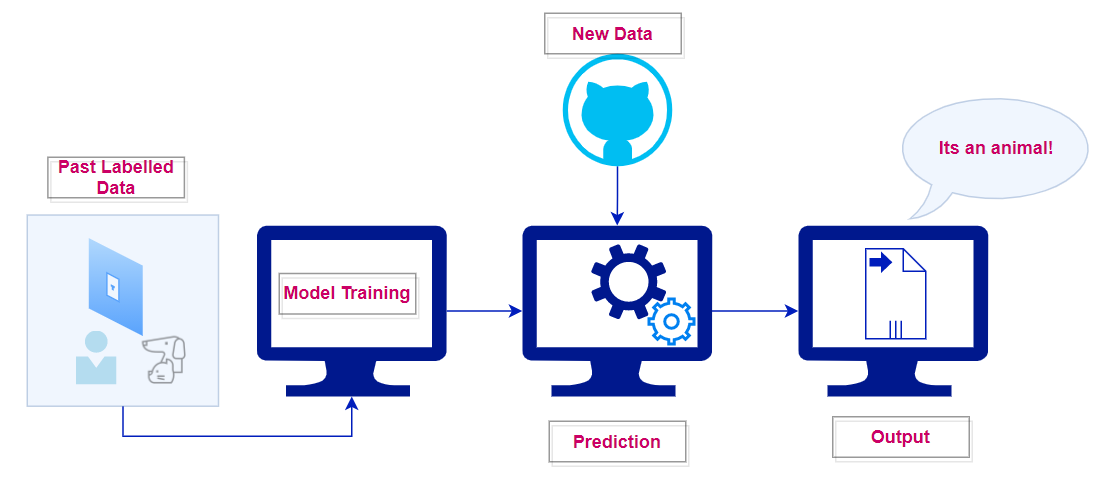


**Figure 4.5.3:** Kernel Machine

The hyperplanes can be described by vectors the display would be linear variations of the vector image parameters in the database. The locations in the future, which are projected into the hyperplane, are determined by the relationship with this feature space option. Know that if the amount of proximity of test data points to the corresponding data base point gets smaller and moves farther away from each word in certain steps. Via the sum of the above channels the relative proximity of each point of the data points from one or another of the sets to be biased against can be measured. Notice that as a product of a far more Dynamic differentiation between said nodes that are not convex inside the original space, the set of defined dots in any hyperplane can be very complicated.

**4.5.2.4 Example**

SVM can be comprehended with the model that has been utilized in the KNN classifier. An unusual feline that additionally has a few highlights of mutts, so in the event that a model is needed that can precisely distinguish whether it's a cat or dog, so such a model can be made by utilizing SVM calculation. We will initially prepare a model with lots of pictures of cats and dogs so it can find out about various highlights of cats and dogs, and afterwards we test it and this odd animal. So as a support vector makes a choice between these two information (cat or dog) and picks outrageous cases (support vectors), it will see the extraordinary instance of cat and dog. Based on the support vectors it will arrange it as a cat.



**Figure 4.5.4:** Classifier Model

**4.5.3** **Classification and Regression Trees (CART):**

**4.5.3.1 Introduction**

CART is mostly referred to as a decision tree. It acts as a base for many other tree based algorithms like bagged decision trees, R.F. and boosted decision trees. This model can be represented as a binary tree. It works on the binary tree concept of data structures and there is nothing fancy about it. Each and every root node depicts a single input variable (X). The leaf nodes are output (Y), used to make the prediction.

**4.5.3.2 Classification**

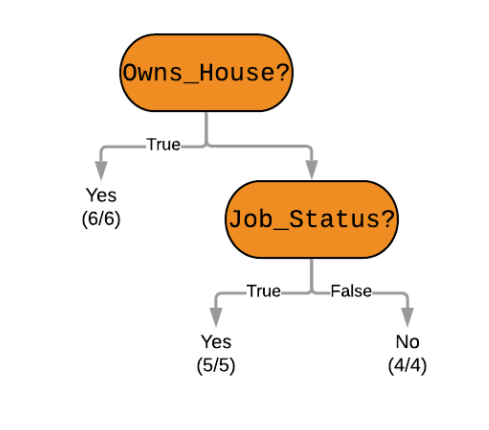
In this, a target function is found out which will enables to forecast values of a discrete class attribute. This is how it is found out in which class does a labelled training set falls in.

The basic flow of a CART algorithm is as follows:

1. Start with the root node with all training instances.
2. An attribute is selected on the basis of splitting criteria.
3. Instances are partitioned according to selected attributes recursively.
4. The partitioning stops when:

* No examples are left
* All the examples of a given node belongs to the same class
* No remaining attributes available for partitioning any further - majority class in the leaf.

The figure below depicts how a CART algorithm works:



**Figure 4.5.5:** CART algorithm flow

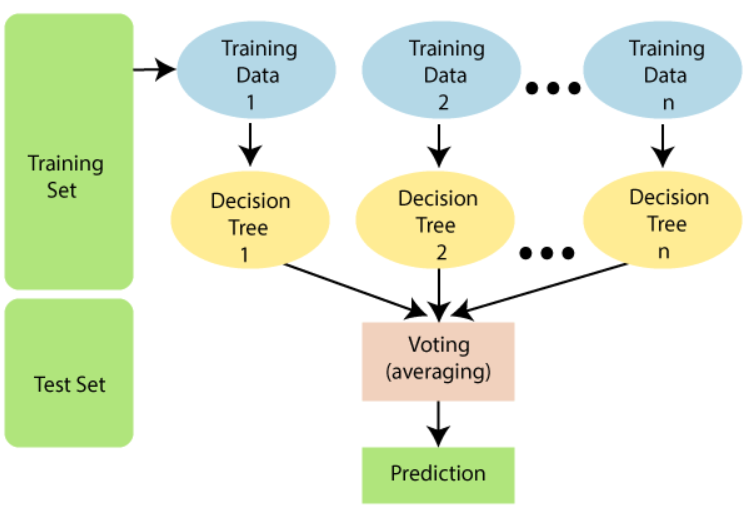
All in all, CART is a powerful algorithm which is also relatively easy to explain compared to other Machine Learning approaches. It does not require much computing power, hence allowing us to build models faster.

**4.5.4 Random Forest**

**4.5.4.1 Introduction**

Random Forest is a very popular ML algorithmic that belongs to the other supervised learning algorithms. It is useful for both classification and regression problems. The concept behind this algorithm is of ensemble learning, which is a process of solving complex problems and improving the performance of the model.

Random Forest is a classifier that contains multiple decision trees on various subsets of the given dataset, and takes the average to improve the predictive accuracy of the model. The greater the number of decision trees, the better the result is. The below figure shows the working of a Random Forest Algorithm:



**Figure 4.5.6:** Random Forest Working Diagram

As random forest combines multiple decision trees, it is sometimes possible that not all trees give out correct output values. Although, together, all trees predict the correct output. Assumptions to achieve good Random Forest Classifier results are:

* Some actual values in the feature variable of the dataset should be there so that the classifier can give out an educated result rather than a guessed one.
* The predictions from each tree must have very low correlations.

Advantages:

* R.F. is capable of performing both Classification and Regression tasks.
* It is capable of handling large datasets with high dimensionality.
* It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantage:

* Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

**4.5.5 RF Classifier:**

**4.5.5.1 Introduction**

Random Forest Classifier is a subset of Random Forest as it is the classification part only of the random forest algorithm.

Some points on why we should use random forest algorithm are:

* The algorithm takes less time to train w.r.t to other algorithms.
* Output result prediction accuracy is very high for this algorithm, and even for large datasets, it runs efficiently.
* In cases of large missing data, the algorithm maintains the accuracy very well.

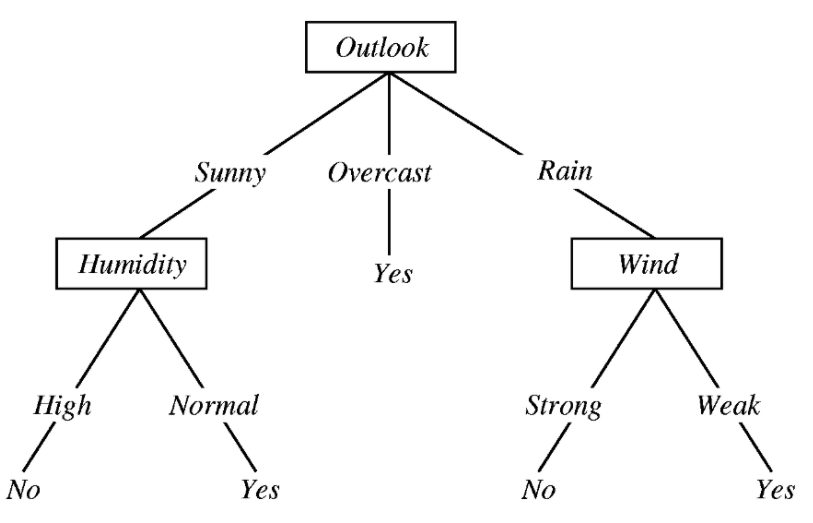
Steps involved in the working of a Random Forest algorithm:

1. Choose any random r data from the dataset.
2. Decision trees are built based on the selected data.
3. Decide number N for decision trees that you would like to create.
4. Repeat 1 and 2.
5. For fresh data, determine the predictions of each decision tree and give the value to the tree with the maximum votes.

**4.5.6 Decision tree:**

**4.5.6.1 Introduction**

Decision trees are a supervised type of learning algorithms in which the data is continuously split according to certain parameters. A decision tree can be best explained with the help of two terms, decision nodes and leaf nodes, the leaf nodes act as the final outcomes, whereas the decision nodes are where the data is split. These trees are mainly binary trees. A decision tree can be imagined to be like a figure below:



**Figure 4.5.7:** A decision tree

**4.5.6.2 Approach**

While making a decision tree, at each node of the tree, various sorts of inquiries are posed. In light of the inquiries posed, it figures out the data acquire relating to it. Data acquire is utilized to choose which highlight can be chosen to part on at each progression while building the tree.

**4.5.7 Bagged Decision Tree:**

**4.5.7.1 Introduction**

Bagged Decision Tree is a part of the decision tree family. Basically, there are two techniques to perform ensemble decision trees, one of them is a bagged decision tree. Ensemble methods are ones which combine several decision trees to produce better predictive models, rather than just using one decision tree.

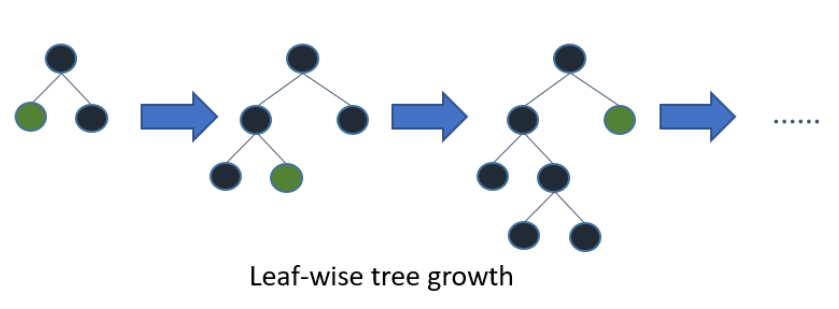
Here, a bagged decision tree is used because our goal is to reduce the variance of a decision tree. The idea here is to create several subsets of data from training samples chosen randomly with replacement. Now, these collections of subset data are used to train their decision trees. The Random Forest algorithm is just an extension of the bagged decision tree. The bagged decision tree uses the bootstrap method, which is a powerful statistical method for estimating a quantity from a dataset.

When a Bagged Decision Tree is used, there is less concern about the individual trees overfitting the training data. The number of decision trees can be increased and keep bagging until the accuracy stops increasing. The final prediction of the ensemble model will be given by calculating the average of all predictions from the individual decision trees.

**4.5.8. Light GBM (LGBM):**

**4.5.8.1 INTRODUCTION**

LGBM is a quick, distributed and high-performing gradient boosting framework based on decision trees algorithm, used in classifications, rankings and other ML tasks. It splits the trees with the best fits on contrary to the other algorithms which splits the trees depth wise. Hence, in LGBM, leaf-wise calculation diminishes more loss than the level-wise calculation, and along these lines brings about much better exactness which can infrequently be accomplished with other boosting algorithms. Also, it's surprisingly faster, hence the word “Light”. The figure below shows how the decision tree is made in LGBM.



**Figure 4.5.8:** LGBM leaf wise tree growth

Advantages:

* Faster training speed and higher accuracy
* Uses less memory
* Higher accuracy than other booster methods
* Compatibility with huge datasets
* Support of parallel learnings

**4.5.9. XGB Classifier (Extreme Gradient Boosting):**

**4.5.9.1 INTRODUCTION**

It’s said that if you are having a hard time with predictive modelling, use XGBoost. It acts as the final weapon for data scientists. It’s a highly complex but sophisticated model, so strong that it can deal with all sorts of indiscretions in the data. Though developing a model is easy but to improve a model using the same is difficult as this algorithm uses multiple parameters. Parameter tuning is must to improve the model. XGBoost is a progressive implementation of GB Algo. XGBoost algorithm has many advantages, like:

1. **Regularization:**

* Unlike XGBoost, standard GBM implementations have no regularization. Hence it reduces overfitting.

1. **Parallel Processing:::**

* Compared to GBM, XGBoost is much faster.
* Hadoop is supported by XGBoost.

1. **High Flexibility::::**

* XGBoost allows its user to define the evaluation criteria and also lets the user decide custom optimization objectives.
* This gives the user immense power as he can do whatever he wants.

1. **Handling missing values::::**

* There is a default routine that handles values which are missing in XGBoost.
* XGBoost always tries different ways to encounter missing values on every single node, and takes different paths in the next run.

1. **Tree Pruning:**

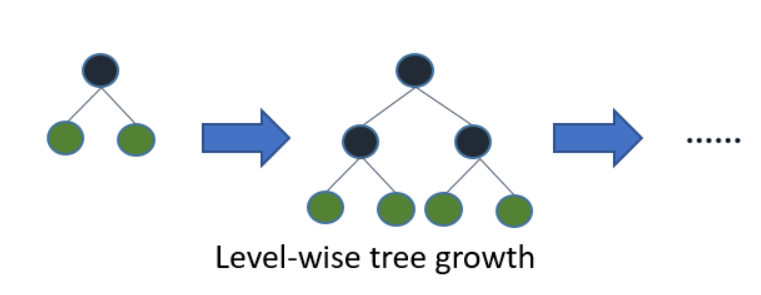
* GBM implements the greedy algorithm as it stops splitting a node if a negative loss is encountered.
* XGBoost on the other hand makes separates to the max\_depth determined and afterward begins pruning the tree in reverse and eliminating parts past which there is no positive gain.

1. **Built-in**zzz **Cross-Validation**zzz**:**

* Cross validation at each iteration is made possible to the user when applying XGBoost so it is much easier to achieve an optimum no. of boost recursions in one run.
* Unlike XGBoostzz, in GBM, only a grid search can be done, and only a limited amount of value can test be performed on.

1. **Continuation of current model**

The XGBoost Decision tree progress can be seen from the image below. When compared with LGBM, you can also spot the difference between the two algorithms when the figures are compared.

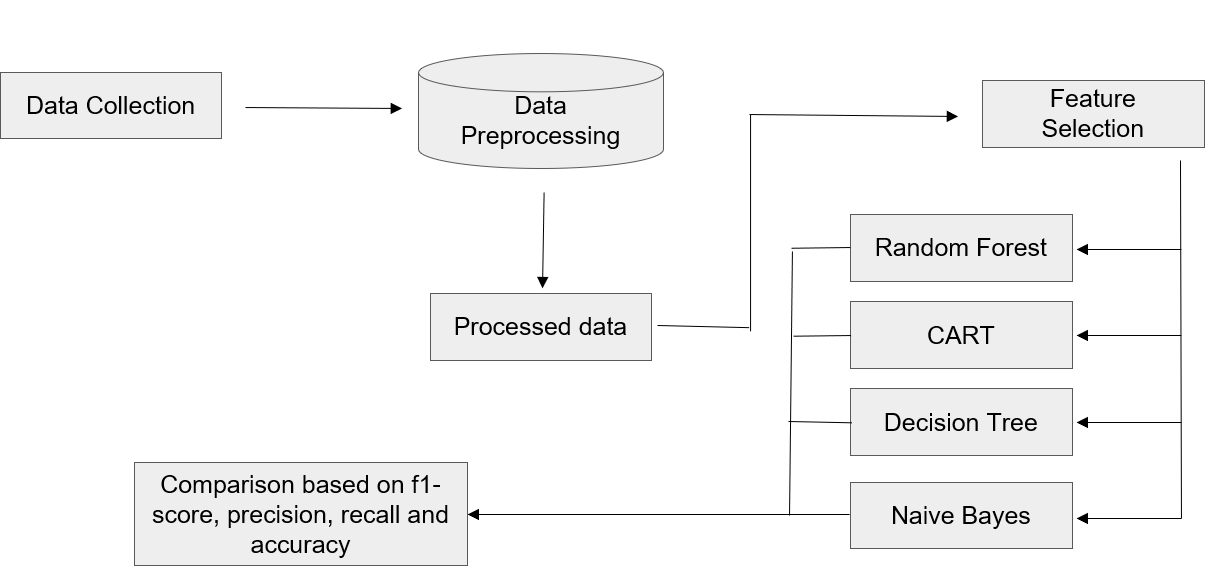


**Figure 4.5.9:** XGBoost Level-wise tree growth

**CHAPTER 5**

**SYSTEM DESIGN**

**5.1 ARCHITECTURE**

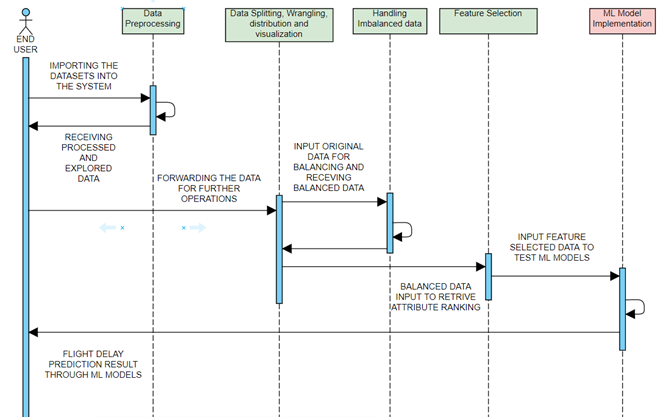
****

**Figure 5.1:** Flow Chart Diagram

The above figure depicts the various components that make up the architecture of the Flight Delay Prediction model. The various components are- data collection, data processing, wrangling and visualization, feature selection, application of ML models, and comparing the models using accuracy, f1 score, precision and recall. The data is collected from the Kaggle cite and downloaded as an excel sheet. The excel sheet is then downloaded in the Jupiter notebook. Then the data is processed, data wrangling is performed and the data distribution is visualized. The data then goes through the feature selection process. After this, the data splitting takes place after which different ML algorithms are used over the data and the results are noted and the best algorithm is decided.

**5.2 UML DIAGRAM**

**5.2.1 SEQUENCE DIAGRAM**

****

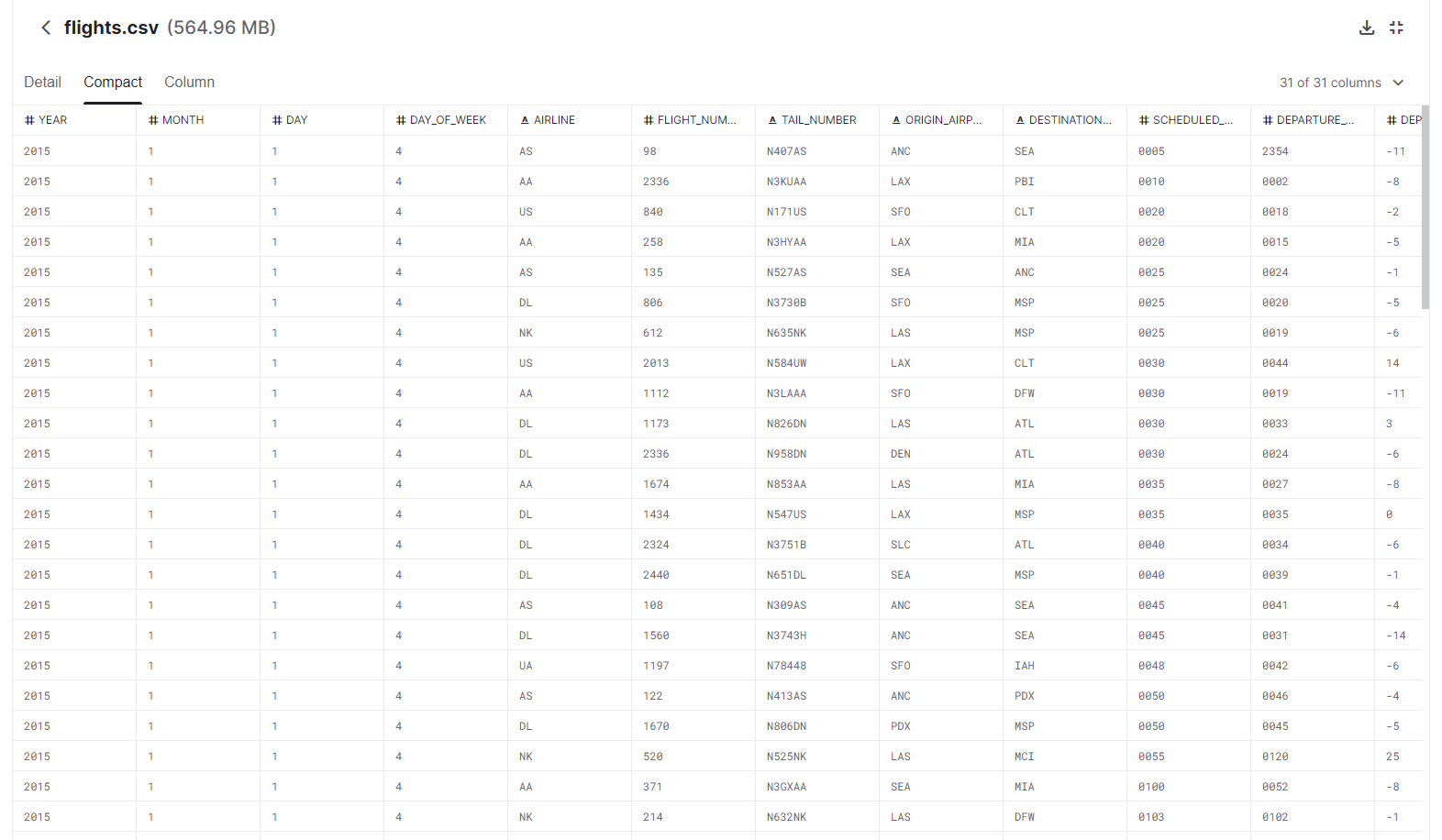
**Figure 5.2:** Sequence Diagram

The above diagram shows the sequence of operations involved in the Flight Delay Prediction model. It comprises an end user and has the following components- data processing, data splitting, wrangling, distribution and visualization, handling imbalance data, feature selection and ML model implementation. Firstly, the end user imports the dataset into the system, after which data processing takes place. The end user then receives processed and explored data. The end user then forwards the data for further operations like data splitting, wrangling, distribution and visualization, after which the data balancing stage takes place. The balanced data is then sent back to the data splitting phase. From there, the data is made to go through the feature selections process where the important features in the dataset are kept and the rest are removed. This data is then passed onto the stage where the ML algorithms are applied on the datasets. The results are then returned to the end user, so that he can figure out which model gave him the best results.

**5.3 DATASET**

The dataset is basically trains the ML model. It also performs all test related work on the machine learning model. The bigger, better, more refined the dataset is the more accurate the model predictions become. Thus, the dataset plays a very huge role in the model outcome. The dataset should not have any consistencies like blank data or data out of range. The **31 parameters** are: YEARSS, MONTHzz, DAYzzz, DAYz\_OFz\_WEEKz, AIRLINEzz, FLIGHTzz\_NUMBERz, TAILz\_NUMBERzz, ORIGINzz\_AIRPORTzzzz, DESTINATIONzz\_AIRPORTzzz, AIR\_TIMEzz, TAXI\_OUTzzz, SCHEDULEDzz\_DEPARTUREzz, DEPARTUREzzz\_TIMEzz, DEPARTUREzzz\_DELAYzzz, WHEELSzzz\_OFFzzz, SCHEDULEDzz\_TIMEzz, ELAPSEDzzz\_TIMEzz, DISTANCEzzz, WHEELSzzz\_ONzz, TAXI\_INzzz, SCHEDULEDzzzz\_zzARRIVAL, zzzARRIVALzz\_TIME, ARRIVALzz\_DELAYzz, zzDIVERTEDzz, WEATHER\_DELAYzzz, CANCELLEDzzzz, CANCELLATION\_REASONzzz, zzAIR\_SYSTEM\_DELAYzzz, zzSECURITY\_DELAYzzz, zzzAIRLINE\_DELAYzzz, zzLATE\_AIRCRAFT\_DELAYzzz.

Out of these, only **16** important features were selected after data processing and feature selection methods were applied over the dataset. Those were: YEARSS, FLIGHTzz\_NUMBERz, TAILz\_NUMBERzz, ORIGINzz\_AIRPORTzzzz, DESTINATIONzz\_AIRPORTzzz, SCHEDULEDzz\_DEPARTUREzz, DEPARTUREzzz\_TIMEzz, DEPARTUREzzz\_DELAYzzz, WHEELSzzz\_OFFzzz, ELAPSEDzzz\_TIMEzz, DISTANCEzzz, WHEELSzzz\_ONzz, TAXI\_INzzz, SCHEDULEDzzzz\_zzARRIVAL, zzzARRIVALzz\_TIME, ARRIVALzz\_DELAYzz. After training the model on the following dataset it predicts the condition of the person using the values it gets from the sensors.



**Figure 5.3:** Dataset

**5.4 PROCEDURE**

* The very first step of any project is to search and download the dataset that suits the purpose of the project.
* The next step is to import the downloaded data into the Jupiter notebook.
* Now, remove all the columns that are not required for the project from the dataset, and also remove all the missing values that are present in each column until the number of missing values for each column come to 0.
* Once the previous step is completed, we start visualizing the data distribution using different techniques like box plot, missingno graph, heatmap, histograms and etc.
* After the data visualization step, it's now time to perform the feature selection over the dataset. There are various methods to the same. In our case, we have used the Recursive Feature Elimination method.
* Now that the previous step has been implemented, we are left with 16 important features in our dataset, which are important for our project.
* The next step is to split the dataset into training and testing halves.
* After the previous stage is over, we start applying the different machine learning models over the dataset.
* Finally, we record the accuracy of the different machine learning models, and compare them to vote out the best ML model among the ones we used.

##### **CHAPTER 6**

##### **TESTING**

* 1. **UNIT TESTING:**

It is the way toward testing every single module created by the designing team. The whole setup is divided into numerous bundles which comprise of little units of code. It improves the general structure of the modules and refactors the code whenever essential. These modules are tested autonomously independent of other modules. This test also checks for repetition. If there is any redundancy, the copy of the records is deleted. It checks for run time blunder too and checks if the blunder was in an individual page or method. Preferred standpoint of performing unit testing is its capacity to check every module exclusively which is supportive in finding the littlest of littlest mistakes. Since unit testing is done at an in all respects early stage the expense of testing is negligible when contrasted with other testing. Modules which are well enormous for unit testing can be accessed utilizing integration testing. Most Machine Learning algorithm projects use unit testing.

* 1. **INTEGRATION TESTING:**

This is subsequent stage after unit testing is performed. Once, every module is tried autonomously and is clear of mistakes, these individual modules are consolidated together and tried in general. The fundamental explanation behind playing out this test is to check for issues when every one of the units are joined. Such tests are experienced in big machine learning projects where there are multiple members, coding different modules. Once all the members are done, an integration testing is performed after combining the modules together. There are diverse manners by which these units can be coordinated. They are:

1. Top-down Integration: Top-down mix joins and tests every one of the modules start to finish. However, one inconvenience of this testing is that it needs a lot of time.
2. Bottom-up Integration: The base up methodology is the other way around of the previous discussed approach. Significant modules are tried last which can make issue amid combination.
3. Big-bang Integration: In this type of testing every one of the functionalities are incorporated and tried at the same time. This methodology is subject to the quantity of modules present. Lesser the modules progressively, the more successful it is.
4. Hybrid Integration: A combination of all the above discussed methodologies.

**6.3 SYSTEM TESTING:**

System testing is the subsequent stage after coordination testing. In this procedure, the entire project is tried for issues and mistakes. They are of two kinds:

1. Black-box Testing
2. White-box Testing

To explain this testing, let’s consider the case of assembling a ball point pen. The top, the ink cartridge, the body, the tail is created independently and tried independently (unit testing). Whenever, at least two modules are prepared, they are consolidated and Integration Testing is finished. At the point when the total pen is collected, system Testing is finished. It this the entire system as a single element.

* Black-box Testing: It is a testing method which is completed by the analyzers. This product can be tried without knowing the inside structure of the program. Programing knowledge isn’t expected to this type of testing procedure. Its fundamental desire is to check for the activity that is performed by the system. It is less tedious. Black box testing is generally called functional test or external testing. It isn’t best for algorithm testing. It very well may be tried on preeminent dimensions of testing like acceptance testing.
* White-box Testing: It is a testing technique which is done by s/w engineers. The usefulness of the program must be known to the developers. Programming learning is an unquestionable requirement to perform White-box Testing. It is generally called inside testing or basic testing. Its principal point is to check program code, circles, conditions, branches and how framework is performing. It tends to be tried on more elevated amounts of testing like acknowledgement testing.

**6.4 REGRESSION TESTING:**

This is a standout amongst the most significant sort of testing with regards to the correct advancement of a product. We can likewise consider it as one significant advance in the software development life cycle (SDLC). Each product has a particular sort of functionalities which should be refreshed without fail. This is typically done to guarantee its security in all stages. Along these lines, for this to be guaranteed, these functionalities need to be refreshed with new bit of code without fail. In this manner, so as to guarantee that the new code doesn’t influence the new usefulness, relapse testing is completed. This is normally done by specialists or programming engineers who have profound comprehension of the product activities in and out.

**6.5 SMOKE TESTING**

This is additionally one angle to ensure that the usefulness is simply working fine independent of the new code that is added to change it. A standout amongst the most significant motivation to play out this type of testing is to expel one of those lines of code that isn’t required any longer and make sure that they are not to influence the usefulness of the model. It covers the greater part of the critical elements of the programming, however do not dissect them in detail. The outcome of this test is used to pick whether to proceed with the further testing. If the soke test passes, continue with further testing.

**CHAPTER 7**

**RESULTS**

In this project, as there were several ML algorithms implemented, the following results were attained after accurate code compilation. These results were later tabulated for better understanding, and to decide the best Model of the all. The figure below shows some of the results obtained.



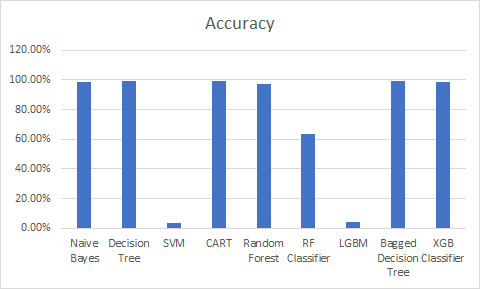
**Figure 7.1:** Results as calculated

The results have also been tabulated as we can see from the table below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | F1-score | Recall | Precision | Accuracy |
| Naive Bayes | 0.7096 | 0.7225 | 0.7056 | 98.65% |
| Decision Tree | 0.7213 | 0.7237 | 0.7209 | 98.90% |
| SVM | 0.0016 | 0.0056 | 0.0012 | 3.73% |
| CART | 0.7678 | 0.7714 | 0.7659 | 99.15% |
| Random Forest | 0.6367 | 0.6474 | 0.6455 | 97.24% |
| RF Classifier | 0.1458 | 0.1577 | 0.1493 | 63.76% |
| LGBM | 0.0020 | 0.0085 | 0.002 | 3.96% |
| Bagged Decision Tree | 0.7272 | 0.7346 | 0.7238 | 98.85% |
| XGB Classifier | 0.7052 | 0.7006 | 0.7173 | 98.50% |

**Table 7.1:** Results table

To represent the results, we used bar graphs. Its helps to visualize the data well as attached below:



**Figure 7.2:** Bar-Graph representation of Results

The bar graph plotted shows the ML algorithm vs. their accuracy. This can help visualize the results in a much better way.

* The experimental investigation has showed that the best results are yielded by using the tree based models. CART (99.15%), Decision Trees (98.90).
* Naïve Bayes and XGB Classifier have also shown significantly high accuracy. All of the algorithms give an accuracy score of more than 98%.
* The implemented supervised ML model indicates that the best way to predict flight delays are by using tree based ML algorithms.

**CHAPTER 8**

**CONCLUSION**

In this project, the use of Machine Learning to predict flight delay has been summarized. Even though machine learning is currently being used in all the sectors known, there is always room for improvement and research. The early detection of flight delay is useful for the customers to a greater extent as it helps them save large amounts of money, and their valuable time. Machine Learning can help largely in this regard. ML based flight delay predictions can warn the passengers of the changes of a flight getting delayed before the passenger even books tickets. Even after he does, ML based flight delay predictions can be used to inform the users about a flight delay before they reach the airport. With a proper implementation of ML based flight delay prediction model, even the airline companies can cut their losses by a much larger extent. This may also help portray a positive reputation of the companies. This may also help reduce sustainability issues, and reduce fuel consumption and gas emissions. This project may also help other researchers the study delay pattern of flights, depending on actual real time data provided in a much better way.

The possibility of a brilliant flight delay prediction system by incorporating machine learning is a novel commitment in the field of data science and it will lessen delay related problems for both the aviation industry and their customers. It is often seen the how the customers have to go through a lot of inconvenience in terms of experiencing any kind of flight delays. When this happens, the airlines are badly reviewed and the number of customers tend to decrease in the future. And thus, the customer-airline relationship in interrelated based on how good or bad of an experience is provided to them. By creating such a model where these delays can be predicted beforehand, it can be concluded that most predictions were fairly accurate and the precision can help in reducing various airlines related problems (as mentioned before). Hence, with the help of a system like this a plethora of inconveniences related to flight delays can be removed and worked upon for further enhancement of the model.

**CHAPTER 9**

**FUTURE ENHANCEMENTS**

As it can be observed that we used different ML algorithms to compare their accuracy to give the best results but we did face a path block. Due to the lack of enhanced computing hardware, we had to restrict our project with 20000 tuples. Well with the increase in computational power of computers, future researchers can increase the number of tuples, and work on a much larger dataset than we have. This may certainly increase the delay prediction accuracy to a greater extent. We can also try to implement deep learning in much better ways to predict flight delays. As we found in our research work that researchers who have been implementing deep learning into flight delay prediction have not been getting good results. This is something that could be worked on in the future. Also, now that we have many researchers implementing ML in flight delay prediction, it's now time that we work on implementing ML with an AP or web page so that people can actually use the application and really benefit from it. It would be better if we have a model where the machine can predict flight delays while the customer is booking his tickets so that he knows from beforehand whether to book the ticket or not. This may cut money losses to a great extent.

**PAPER PUBLICATION STATUS**

The research paper created has been sent to Materials Today published by ELSEVIER. Currently awaiting approval.

**REFERENCES**

1. Kim, Y. J., Choi, S., Briceno, S., & Mavris, D. (2016). A deep learning approach to flight delay prediction. 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC).
2. Chakrabarty, N. (2019). A data mining approach to flight arrival delay prediction for American Airlines. 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON).
3. George, M., Denton, E., & Zwiggelaar, R. (2018). Mammogram breast density classification using mean-elliptical local binary patterns. 14th International Workshop on Breast Imaging (IWBI 2018).
4. Thiagarajan, B., Srinivasan, L., Sharma, A. V., Sreekanthan, D., & Vijayaraghavan, V. (2017). A machine learning approach for prediction of on-time performance of flights. 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC).
5. Yogita Borse, Dhruvin Jain, Shreyash Sharma, & Viral Vora, Aakash Zaveri. (2020). Flight delay prediction system. International Journal of Engineering Research and, V9(03).
6. Esmaeilzadeh, E., & Mokhtarimousavi, S. (2020). Machine learning approach for flight departure delay prediction and analysis. Transportation Research Record: Journal of the Transportation Research Board, 2674(8), 145-159.
7. Tu, Yufeng and Ball, Michael O. and Jank, Wolfgang. (2005). Estimating Flight Departure Delay Distributions - A Statistical Approach with Long-Term Trend and Short-Term Pattern. Robert H. Smith School Research (RHS 06-034).
8. 8. K., & Rao, K. B. (2018). Machine learning approach to predict flight delays. International Journal of Computer Sciences and Engineering, 6(10), 231-234.
9. 9. Natarajan, V., Meenakshisundaram, S., Balasubramanian, G., & Sinha, S. (2018). A novel approach: Airline delay prediction using machine learning. 2018 International Conference on Computational Science and Computational Intelligence (CSCI).
10. 10. Carvalho, L., Sternberg, A., Maia Gonçalves, L., Beatriz Cruz, A., Soares, J. A., Brandão, D., Carvalho, D., & Ogasawara, E. (2020). On the relevance of data science for flight delay research: A systematic review. Transport Reviews, 1-30.
11. 11. Chakrabarty, N. (2019). A data mining approach to flight arrival delay prediction for American Airlines. 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON).
12. 12. Manna, S., Biswas, S., Kundu, R., Rakshit, S., Gupta, P., & Barman, S. (2017). A statistical approach to predict flight delay using gradient boosted decision tree. 2017 International Conference on Computational Intelligence in Data Science(ICCIDS).
13. 13. Natarajan, V., Meenakshisundaram, S., Balasubramanian, G., & Sinha, S. (2018). A novel approach: Airline delay prediction using machine learning. 2018 International Conference on Computational Science and Computational Intelligence (CSCI).
14. 14. Yazdi, M. F., Kamel, S. R., Chabok, S. J., & Kheirabadi, M. (2020). Flight delay prediction based on deep learning and Levenberg-Marquart algorithm. Journal of Big Data, 7(1).
15. 15. Yu, B., Guo, Z., Asian, S., Wang, H., & Chen, G. (2019). Flight delay prediction for commercial air transport: A deep learning approach. Transportation Research Part E: Logistics and Transportation Review, 125, 203-221.

**APPENDIX**

**CODE**

**DATA GATHERING AND PREPROCESSING**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import time

df\_use = pd.read\_csv('C:/Users/PRIYANSHU/MJ/flights.csv')

print(df\_use.shape)

df = df\_use.sample(n=20000).copy()

del df\_use

print(df.shape)

df.head()

missing = df.isnull().sum()

missing

df.drop('CANCELLATION\_REASON', axis='columns', inplace = True)

df.drop(['AIR\_SYSTEM\_DELAY','SECURITY\_DELAY','AIRLINE\_DELAY','LATE\_AIRCRAFT\_DELAY','WEATHER\_DELAY'], axis= 1, inplace = True)

df['FLIGHT\_NUMBER'].value\_counts()

df['DEPARTURE\_TIME'].fillna(value=df['DEPARTURE\_TIME'].mean(), inplace=True)

df['DEPARTURE\_TIME']=df['DEPARTURE\_TIME'].astype(int)

df['DEPARTURE\_DELAY'].fillna(value=df['DEPARTURE\_DELAY'].mean(), inplace=True)

df['TAXI\_OUT'].fillna(value=df['TAXI\_OUT'].mean(), inplace=True)

df['WHEELS\_OFF'].fillna(value=df['WHEELS\_OFF'].mean(), inplace=True)

df['WHEELS\_ON'].fillna(value=df['WHEELS\_ON'].mean(), inplace=True)

df['ELAPSED\_TIME'].fillna(value=df['ELAPSED\_TIME'].mean(), inplace=True)

df['AIR\_TIME'].fillna(value=df['AIR\_TIME'].mean(), inplace=True)

df['TAXI\_IN'].fillna(value=df['TAXI\_IN'].mean(), inplace=True)

df['ARRIVAL\_TIME'].fillna(value=df['ARRIVAL\_TIME'].mean(), inplace=True)

df['ARRIVAL\_DELAY'].fillna(value=df['ARRIVAL\_DELAY'].mean(), inplace=True)

import warnings

warnings.filterwarnings('ignore')

df['DEPARTURE\_DELAY']=df['DEPARTURE\_DELAY'].astype(int)

df['TAXI\_OUT']=df['TAXI\_OUT'].astype(int)

df['WHEELS\_OFF']=df['WHEELS\_OFF'].astype(int)

df['WHEELS\_ON']=df['WHEELS\_ON'].astype(int)

df['ELAPSED\_TIME']=df['ELAPSED\_TIME'].astype(int)

df['AIR\_TIME']=df['AIR\_TIME'].astype(int)

df['TAXI\_IN']=df['TAXI\_IN'].astype(int)

df['ARRIVAL\_TIME']=df['ARRIVAL\_TIME'].astype(int)

df['ARRIVAL\_DELAY']=df['ARRIVAL\_DELAY'].astype(int)

df.drop("DIVERTED", axis=1, inplace = True)

df.drop("CANCELLED",axis=1, inplace = True)

df['AIRLINE'] = pd.factorize(df.AIRLINE)[0] + 1

print (df['AIRLINE'])

df['TAIL\_NUMBER'] = pd.factorize(df.TAIL\_NUMBER)[0] + 1

df['ORIGIN\_AIRPORT'] = pd.factorize(df.ORIGIN\_AIRPORT)[0] + 1

df['DESTINATION\_AIRPORT'] = pd.factorize(df.DESTINATION\_AIRPORT)[0] + 1

**DATA VISUALIZATION**

import seaborn as sns

sns.boxplot(x=df['DEPARTURE\_TIME'])

import missingno as msno

msno.matrix(df)

num\_bins = 15

df.hist(bins=num\_bins, figsize=(20,15))

plt.savefig("histogram\_plots")

plt.show()

plt.figure(figsize=(12,5))

sns.distplot(df["DEPARTURE\_DELAY"][df["ARRIVAL\_DELAY"]==0])

sns.distplot(df["DEPARTURE\_DELAY"][df["ARRIVAL\_DELAY"]==1])

plt.legend(['0','1'])

plt.show()

**FEATURE SELECTION**

from pandas import read\_csv

from sklearn.feature\_selection import RFE

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import ExtraTreesClassifier

from sklearn.feature\_selection import RFE

from sklearn.linear\_model import LogisticRegression

array = df.values

X = array[:,0:23]

Y = array[:,22]

model = ExtraTreesClassifier()

model.fit(X, Y)

print(model.feature\_importances\_)

model = LogisticRegression()

rfe = RFE(model, 16)

fit = rfe.fit(X, Y)

print("Num Features: ", fit.n\_features\_)

print("Selected Features: ",fit.support\_)

print("Feature Ranking: ", fit.ranking\_)

df\_selected = df[['YEAR','FLIGHT\_NUMBER','TAIL\_NUMBER','ORIGIN\_AIRPORT','DESTINATION\_AIRPORT','SCHEDULED\_DEPARTURE','DEPARTURE\_TIME','DEPARTURE\_DELAY','WHEELS\_OFF','ELAPSED\_TIME','AIR\_TIME','DISTANCE','WHEELS\_ON','SCHEDULED\_ARRIVAL','ARRIVAL\_TIME','ARRIVAL\_DELAY']].copy()

df\_selected

from sklearn.model\_selection import KFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.linear\_model import LogisticRegression

kfold = KFold(n\_splits=10, random\_state=7)

model = LogisticRegression()

scoring = 'accuracy'

results = cross\_val\_score(model, X, Y, cv=kfold, scoring=scoring)

print("Accuracy: ",results.mean())

**DATA SPLIT**

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

test\_size = 0.20

seed = 7

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=test\_size, random\_state=seed)

model = LogisticRegression()

model.fit(X\_train, Y\_train)

result = model.score(X\_test, Y\_test)

print("Accuracy:",result)

**ML MODELS IMPLEMENTATION**

**NAIVE BAYES:**

from sklearn.metrics import accuracy\_score, f1\_score, precision\_score, recall\_score, classification\_report, confusion\_matrix

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score

model = GaussianNB()

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**SVM:**

from sklearn.svm import SVC

model = SVC()

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**CART:**

from sklearn.model\_selection import KFold

from sklearn.model\_selection import cross\_val\_score

from sklearn.tree import DecisionTreeClassifier

Kfold = KFold(n\_splits=10, random\_state=7)

model = DecisionTreeClassifier()

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**RANDOM FOREST:**

from sklearn.ensemble import RandomForestClassifier

num\_trees = 100

max\_features = 16

kfold = KFold(n\_splits=10, random\_state=7)

model = RandomForestClassifier(n\_estimators=num\_trees, max\_features=max\_features)

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**RF CLASSIFIER:**

from sklearn.model\_selection import cross\_val\_score

from sklearn.ensemble import RandomForestClassifier

rfc =RandomForestClassifier(n\_estimators=500, n\_jobs=-1, random\_state=7, min\_samples\_leaf=5)

rfc.fit(X\_train, Y\_train)

y\_pred = rfc.predict(X\_test)

from sklearn.metrics import confusion\_matrix

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

from sklearn.metrics import accuracy\_score, f1\_score, precision\_score, recall\_score, classification\_report, confusion\_matrix

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**LGBM:**

from lightgbm import LGBMClassifier

lgbmc = LGBMClassifier(random\_state=7, n\_estimators=100, colsample\_bytree=0.5, max\_depth=2, learning\_rate=0.1, boosting\_type='gbdt')

lgbmc.fit(X\_train, Y\_train)

y\_pred = lgbmc.predict(X\_test)

from sklearn.metrics import confusion\_matrix

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**BAGGED DECISION TREE:**

from sklearn.ensemble import BaggingClassifier

Seed = 10

kfold = KFold(n\_splits=10, random\_state=seed)

cart = DecisionTreeClassifier()

num\_trees = 100

model = BaggingClassifier(base\_estimator=cart, n\_estimators=num\_trees, random\_state=seed)

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy = %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**DECISION TREE:**

from sklearn import tree

model = tree.DecisionTreeClassifier()

model.fit(X\_train, Y\_train)

a = accuracy\_score(Y\_test, model.predict(X\_test))

print("Accuracy = %.2f%%" % (a \* 100.0))

y\_pred = model.predict(X\_test)

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))

**XGB CLASSIFIER:**

from numpy import loadtxt

from xgboost import XGBClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

model = XGBClassifier()

model.fit(X\_train, Y\_train)

y\_pred = model.predict(X\_test)

predictions = [round(value) for value in y\_pred]

accuracy = accuracy\_score(Y\_test, predictions)

print("Accuracy: %.2f%%" % (accuracy \* 100.0))

print("F1 Score = ", f1\_score(Y\_test, y\_pred, average="macro"))

print("Precision Score = ", precision\_score(Y\_test, y\_pred, average="macro"))

print("Recall Score = ",recall\_score(Y\_test, y\_pred, average="macro"))