

PREDICT FLIGHT DELAYS USING SUPERVISED MACHINE LEARNING TECHNIQUE

A Project Report

Submitted by

SAIPRASAAD K- 312317104146
ROHITH VIGNESHWAR D –312317104143

in partial fulfillment for the award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



St. JOSEPH'S COLLEGE OF ENGINEERING

(An Autonomous Institution)

St. Joseph's Group of Institution

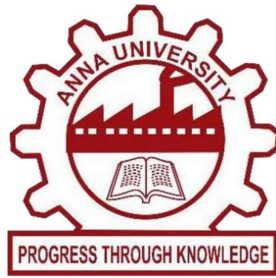
Jeppiaar Educational Trust

OMR, Chennai 600 119

ANNA UNIVERSITY: CHENNAI

April 2021

ANNA UNIVERSITY: CHENNAI 600 025



BONAFIDE CERTIFICATE

*Certified that this project report on **PREDICT FLIGHT DELAYS USING SUPERVISED MACHINE LEARNING TECHNIQUE** is the bonafide work of **SAIPRASAAD K (312317104146)** and **ROHITH VIGNESHWAR D (312317104143)** who carried out the project under my supervision during **the academic year 2020 - 2021**.*

Signature

Mr. S. Vinu M.E.,(Ph.D)

Assistant Professor

Department of Computer Science and
Engineering

St. Joseph's College of Engineering

OMR, Chennai- 600119.

Signature

Dr. A. Chandrasekar M.E., Ph.D

Professor and Head

Department of Computer Science and
Engineering

St. Joseph's College of Engineering

OMR, Chennai- 600119

CERTIFICATE OF EVALUATION

COLLEGE NAME : St. Joseph's College of Engineering,
OMR. Chennai - 600 119.

BRANCH : B.E. - Computer Science and Engineering

SEMESTER : VIII

Sl. No	Name of the Student	Title of The Project	Name of the Supervisor with Designation
1.	Saiprasaad K (312317104146)	Predict Flight Delays Using Supervised Machine Learning Technique	Mr. S. Vinu M.E.,(Ph.D) Assistant Professor
2.	Rohith Vigneshwar D (312317104143)		

The report of the project work submitted by the above students in partial fulfillment for the award of the Degree of Bachelor of Engineering in Computer Science and Engineering at Anna University is confirmed to be report of the work done by the above students and then evaluated.

Submitted to Project and Viva Examination held on_____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

At the outset, We would like to express our sincere gratitude to our beloved **Dr. Babu Manoharan M.A., M.B.A., Ph.D** *Chairman, St. Joseph's Group of Institutions* for his constant guidance and support to the student community and the Society.

We would like to express our hearty thanks to our respected **Managing Director Mrs. S. Jessie Priya M.Com.** for her kind encouragement and blessings.

We wish to express our sincere thanks to the **Director Mr. B. Shashi Sekar, M.Sc.** for providing ample facilities in the institution.

We express sincere gratitude to our beloved **Principal Dr.Vaddi Seshagiri Rao M.E., M.B.A., Ph.D., F.I.E.** for his inspirational ideas during the course of the project.

We express sincere gratitude to our beloved **Dean (Research) Dr. Parvathavarthini M.Sc., M.B.A., M.E., Ph.D** for her inspirational ideas during the course of the project.

We wish to express our sincere thanks to **Dr. A. Chandrasekar M.E., Ph.D. Head of the Department, Department of Computer Science and Engineering, St. Joseph's College of Engineering** for his guidance and assistance in solving the various intricacies involved in the project.

We would like to acknowledge our profound gratitude to our supervisor **Mr. S. Vinu** for his expert guidance and connoisseur suggestion to carry out the study successfully.

Finally, we thank the **Faculty Members** and our **Family**, who helped and encouraged us constantly to complete the project successfully.

ABSTRACT

The primary goal of this project is to predict airline delays caused by various factors. Flight delays lead to negative impacts, mainly economical for commuters, airline industries and airport authorities. The growth of the aviation sector has made flight delays more common across the world. They cause inconvenience to the travelers and incur monetary losses to the airlines. We analyzed the various factors responsible for flight delays and applied machine learning models such as Random Forest, XGBoost, KNN, Decision Tree to predict whether a given flight would be delayed or not. Also with certain features we can predict how far the delay is going to be using some regression techniques like Random Forest Regression and Decision Tree Regression. We also added a recommendation feature in which given a source and destination, we would list flights which are recommended to travel. Also we can know the percentage of Delay and Not delayed of a particular journey by entering Source , Destination and the name of Airlines.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	
	LIST OF FIGURES	
	LIST OF ABBREVIATIONS	
1	INTRODUCTION	
	1.1 PROBLEM IDENTIFICATION	
	1.2 PROJECT OBJECTIVE	
	1.3 SIGNIFICANCE OF WORK	
	1.4 EXISTING SYSTEM	
	1.4.1 Disadvantage Of Existing System	
	1.5 PROPOSED SYSTEM	
	1.5.1 Advantage Of Proposed System	
2	LITERATURE SURVEY	
3	SYSTEM REQUIREMENTS	
	3.1 SOFTWARE REQUIREMENTS	
	3.2 HARDWARE REQUIREMENTS	
4	SYSTEM DESIGN	
5	MODULE DESCRIPTION	
	5.1 EXPLORING AND ANALYSING DATA	
	5.1.1 Importing Dataset	
	5.1.2 Data Preprocessing	
	5.2 FEATURE SELECTION AND LABEL	

ENCODING

5.3 PREDICTING FLIGHT DELAYS

5.3.1 Predict Whether The Flight Will Get Delayed Or Not

5.3.2 Prediction Of Delay In Minutes

5.3.3 Ratio Of Delay To Not – Delayed

5.4 RANKING AND RECOMMENDATION OF FLIGHTS

5.5 DEPLOYING USING DJANGO

6 CONCLUSION AND FUTURE ENHANCEMENTS

6.1 CONCLUSION

6.2 FUTURE ENHANCEMENTS

APPENDIX 1

APPENDIX 2

REFERENCES

LIST OF FIGURES

Figure No.	Title	Page No
3.1	Machine Learning Diagram	
4.1	Architecture Diagram	
4.2	Flowchart Diagram	
3.1	Machine Learning Diagram	
4.1	Architecture Diagram	
4.2	Flowchart Diagram	

LIST OF TABLES

Table No.	Title	Page No
1.1	Existing system vs proposed system	4
5.2	Recommendations of Airlines	25

LIST OF ABBREVIATIONS

ACRONYM	EXPANSION
CSV	Comma-Separated Values
HTML	HyperText Markup Language
IDE	Integrated Development Environment
ML	Machine Learning
XGB	Extreme Gradient Boosting
RF	Random Forest
KNN	K Nearest Neighbours
DT	Decision Tree
CF	Confusion Matrix
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Square Error
R ²	R - Squared
UI	User Interface

CHAPTER 1

INTRODUCTION

1.1 PROBLEM IDENTIFICATION

In present day scenario, Time is money. Flight delays end up hurting airports, passengers and airlines. Being able to predict how much delay a flight incurs will save passengers their precious time as well as hardships caused due to flight delays or in worse cases cancellations. The problem we trying to solve is to accurately predict flight delays when we have certain features of the flight with us, like airlines who operate them, distance they have to cover, origin airport, target airport, departure times and so on. Being able to accurately predict flight delays can help the passengers know what delays they should be ready to face depending on where they fly from and the airlines they choose to fly. This can enable them to take a buffer, so they do not end up missing connecting flights or meetings. The goal of the project will be to do in depth analysis of the data and play with the input features to see how the prediction accuracy changes. The development of prediction models that perform accurately is difficult due to the complex nature of air transport.

The airline companies incur substantial monetary losses and observe a fall in their reputation if their flights are delayed often. The unforeseen delays also have a cascading effect on various other sectors. According to a report by the Joint Economic Committee of United States Congress, the total cost of flight delays to the US economy was over \$40 billion with \$19 billion to the airlines, \$12 billion to the passengers and around \$10 billion to other industries. The delayed flights also pose certain environmental concerns. Delayed flights consumed an additional 740 million units of jet fuel and released over 7 million metric tones of additional Carbon Dioxide. Thus, the prediction of flight delays is a crucial task.

1.2 PROJECT OBJECTIVE

The main objectives of this work:

1. Given certain features, we would predict that whether a flight would be delayed or not.
2. Also we can predict, how far the delay would be in minutes.
3. We can also predict the percentage of delayed and non-delayed by giving the source, Destination and name of the Airlines.
4. Another feature was added to rank airlines and to recommend which airlines to prefer for a journey.

1.3 SIGNIFICANCE OF WORK

With the increase in the demand for air travel, effects of flight delay have been increasing. The Federal Aviation Administration (FAA) estimates that commercial aviation delays, cost airlines more than \$3 billion per year and according to BTS. Impacts of flight delay in future are likely to get worse due to an increase in the air traffic congestion, growth of commercial airlines and increase in the number of passengers per year. While flight delays are likely to persist in future due to unavoidable factors, we create a predictive algorithm to forecast flight delay.

1.4 EXISTING SYSTEM

The existing system deals with Time series forecasting of data which sees the data year wise and country wise. The existing system uses an approach of articulation point which is assumed to be having the greatest delay. And using arima modelling, the system has seen the change in the delay over years, concluding the delay of which airport is the delay most likely to occur in the country.

1.4.1 Disadvantage Of Existing System

The existing system deals with year-wise timeseries forecasting and uses the approach of clustered networks, where the delay is forecasted only by airport wise. They are harder to explain and to interpret coefficients requires stationary series with constant autocorrelation.

1.5 PROPOSED SYSTEM

In the proposed system, we use classification algorithms like KNN Classification, XGB Classifier , Random Forest Classifier and Decision Tree Classifier to predict whether the flight would be delayed or not. Out of these algorithms XGB Classifier gives best accuracy and so with the help of features like Origin ,Destination ,Airlines , Distance and the delay in departure, we can predict whether the flight arriving in the destination will have delay or not. Also regression techniques are performed to predict arrival delays of how much delay is to occur.

1.5.1 Advantage Of Proposed System

This project aims at analyzing factors responsible for flight delays and designing a machine learning model to predict them. We classify a flight as 'Delayed' or 'Not Delayed' using classification algorithm and use regression to obtain how much delay is to occur. Finally entering source and destination the flights are analyzed on their delays ,cancellation and speed upon which flights are ranked and recommended that can help customer choose best flight for the journey and help airlines to identify flaws in their organization

EXISTING SYSTEM	PROPOSED SYSTEM
Deals with Time-Series forecasting of data only by year	Deals with supervised machine learning solution involving various parameters
Uses a approach of articulation point which is assumed to be having the greatest delay.	No such approach of articulation point.
Country wise forecasting.	Origin , Destination and Airlines wise forecasting
Works only on small datasets	Works also on large datasets
Just shows the trend of delay over years.	Shows delayed or not , along with how much delay is to occur.
No recommendations of flights for journey is made.	Flights are ranked with score based on their delays and speed , recommendations are done.
Do not show the ratio of flights delayed and non delayed.	With the predicted value a pie chart is represented to know how far the flight is to get delayed.

Table 1.1 Existing Vs Proposed System

CHAPTER 2

LITERATURE REVIEW

In architecture of complex weighted networks [2], the Networked structures arise in a wide array of different contexts such as technological and transportation infrastructures, social phenomena, and biological systems. Along with a complex topological structure, real networks display a large heterogeneity in the capacity and intensity of the connections. These features, however, have mainly not been considered in past studies where links are usually represented as binary states, i.e., either present or absent. The analysis provide a better description of the hierarchies and organizational principles at the basis of the architecture of weighted networks. In this, the impact of a local hazard/fault/disturbance can easily spread out to the whole system due to domino effect, cascading effect, and/or ripple effect and eventually evolves into a large-scale disaster.

In The Traffic Flow Management Rerouting Problem in Air Traffic Control[3]: A dynamic Network Flow Approach The problem of determining how to reroute aircraft in the air traffic control system is addressed when faced with dynamically changing weather conditions. It involves mathematical programming approach that consists of several methodologies. Lagrangian Generation Algorithm is used which can quickly generate many non- integral solutions for the problem. Used randomized rounding heuristic Approach which provides a solution to air traffic congestion but fails to give a Optimum solution.

In Systemic delay propagation in the US airport network[6], the performance of an air transportation system in terms of delays is studied. It involves Characterization of flight delays using 3 methods - Distribution of the delay per flight for arrivals and departures, Distribution of departure delays separating the flights according to the season: Summer and winter, Delay distribution for flights departing from certain airports. The model offers the possibility of evaluating the effects of interventions in the system before their real

implementation. The analysis did not suit for all the possible cases as it takes into account only certain parameters.

In Light Statistical Method of Air Traffic Delays Prediction[7] , It uses a light statistical method of airplane landing delay prediction at the destination airport. Proposed method grounds on statistical data processing of previous flights to identify mean trajectory of an airplane at the pre-flight stage and continuously comparison of mean trajectory with current airplane location in order to identify a time to land in the destination airport. A simple solution of minimization problem at the en-route phase of flight needs low computation power and guarantees the required confidence band for results. Statistical methods make it practical to measure the reliability of results. Statistical data is often secondary data which means that is can be easily be misinterpreted.

In the MultiAirport Ground-Holding Problem in Air Traffic Control [10], the Congestion problems are becoming increasingly acute in many major European and American airports. Long-term approaches include construction of additional airports, improved air traffic control technologies and procedures and use of larger aircraft. Medium-term approaches include modification of the temporal pattern of aircraft flow to eliminate periods of "peak" demand. Short-term approaches have a planning horizon of 6-12 hours and include, most important ground-holding policies. In this, the impact of a local hazard/fault/disturbance can easily spread out to the whole system due to domino effect, cascading effect, and/or ripple effect and eventually evolves into a large-scale disaster. Here the project proposed a heuristic algorithm which finds a feasible solution to the integer program by rounding the optimal solution of the LP relaxation. Finally, presents extensive computational results with the goal of obtaining qualitative insights on the behavior of the problem under various combinations of the input parameter

CHAPTER 3

SYSTEM REQUIREMENTS

In this chapter we present technologies used to develop a project. And also here we explain software requirement and hardware requirements.

Machine Learning

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python. Process of training and prediction involves use of specialized algorithms. It feeds the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated into three categories. There are supervised learning, unsupervised learning and reinforcement learning. Supervised learning program is both given the input data and the corresponding labeling to learn data has to be labeled by a human being beforehand. Unsupervised learning is no labels. It is provided to the learning algorithm. This algorithm has to figure out the clustering of the input data. Finally, Reinforcement learning dynamically interacts with its environment and it receives positive or negative feedback to improve its performance.

Data scientists use many different kinds of machine learning algorithms to discover patterns in python that lead to actionable insights. At a high level, these different algorithms can be classified into two groups based on the way they “learn” about data to make predictions: supervised and unsupervised learning. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function from input variables(X) to discrete output variables(y). In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data

set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc.

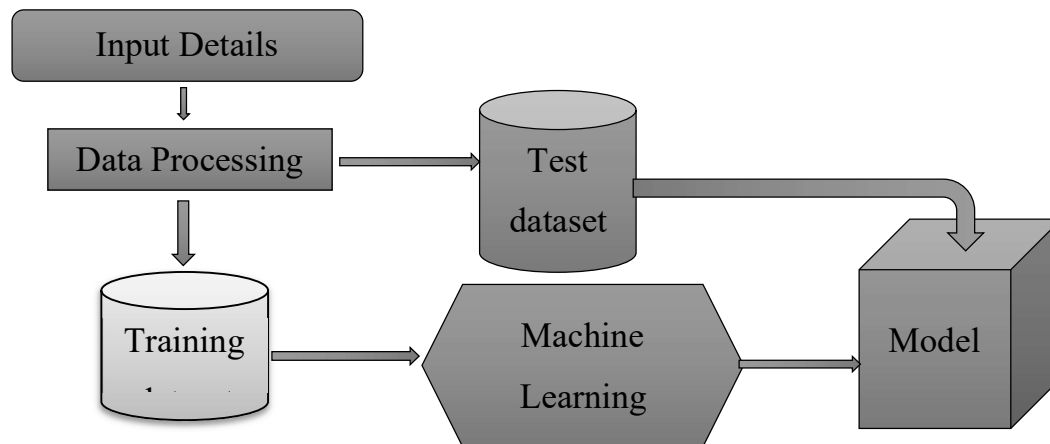


Fig 3.1 Machine Learning Diagram

3.1 HARDWARE REQUIREMENTS

- a) Processor -I3,I5,I7
- b) RAM -4GB
- c) Hard Disk -250GB

a) Processors

Generally speaking, Core i7s are better than Core i5s, which are in turn better than Core i3s. Core i7 does not have seven cores nor does Core i3 have three cores. The numbers are simply indicative of their relative processing powers.

b) RAM

If you wish to enhance the performance, as well as, the graphics output of your system, you can do so by adding 4 GB of RAM, which is more than enough to ensure that you can multitask without a hitch.

c) **HARD DISK**

A 250 GB hard disk is the minimum requirement to run the software.

3.2 SOFTWARE REQUIREMENTS

- a) Python version 3.6 and above
- b) Google colab software or Jupyter Notebook
- c) Python packages :

- ❖ Pandas
- ❖ Numpy
- ❖ Matplotlib
- ❖ Xgboost
- ❖ Sklearn
- ❖ Flask

a) **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

b) Google Colab or Jupyter Notebook

The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. In addition to displaying/editing/running notebook documents, the Jupyter Notebook App has a “Dashboard” (Notebook Dashboard), a “control panel” showing local files and allowing to open notebook documents or shutting down their kernels.

We may also use Jupyter Notebook because Jupyter supports various programming languages. Jupyter makes it easier to distribute research results, as notebooks are easy to share with others. Jupyter Notebooks can also be used as a user interface for big data frameworks, databases and computer clusters and it is open source.

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs.

To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

C) Python Libraries and packages

❖ Pandas:

Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python

programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.

❖ **Numpy:**

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided. All this is explained with the help of examples for better understanding.

❖ **Matplotlib:**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

❖ **XGBoost**

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. It is an implementation of gradient boosting machines created by Tianqi Chen, now with contributions from many developers. It belongs to a broader collection of tools under the umbrella of the Distributed Machine Learning Community or DMLC who are also the creators of the popular mxnet deep learning library.

❖ **Sklearn:**

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

❖ **Flask :**

Flask is a Python web framework built with a small core and easy-to-extend philosophy. Flask is considered more Pythonic than the Django web framework because in common situations the equivalent Flask web application is more explicit. Flask is also easy to get started with as a beginner because there is little boilerplate code for getting a simple app up and running.

CHAPTER 4

SYSTEM DESIGN

This project deals with predicting the flight delays using supervised machine learning techniques which is represented in architecture diagram in fig 4.1. The dataset which is obtained from Kaggle is first being imported and preprocessed. The processed data is cleaned to handle missing values and is visually represented in the form of graphs, charts and heatmaps. The required features are then selected , that is the data is now framed with only required values needed for prediction. The data contains some values in String format which has to be label encoded to number as only number can be used for prediction. The selected data is now fit into classification and regression algorithm for training. The trained data is then tested to get accuracy of models of both classification and regression techniques . The model with best accuracy is used for prediction. The model is further used to get recommendation to fly from a origin to destination, where airlines which is most preferable is represented as output. Also one can see the ratio of delayed to not-delayed given their origin, destination along with airlines. This flow is represented in fig 4.2

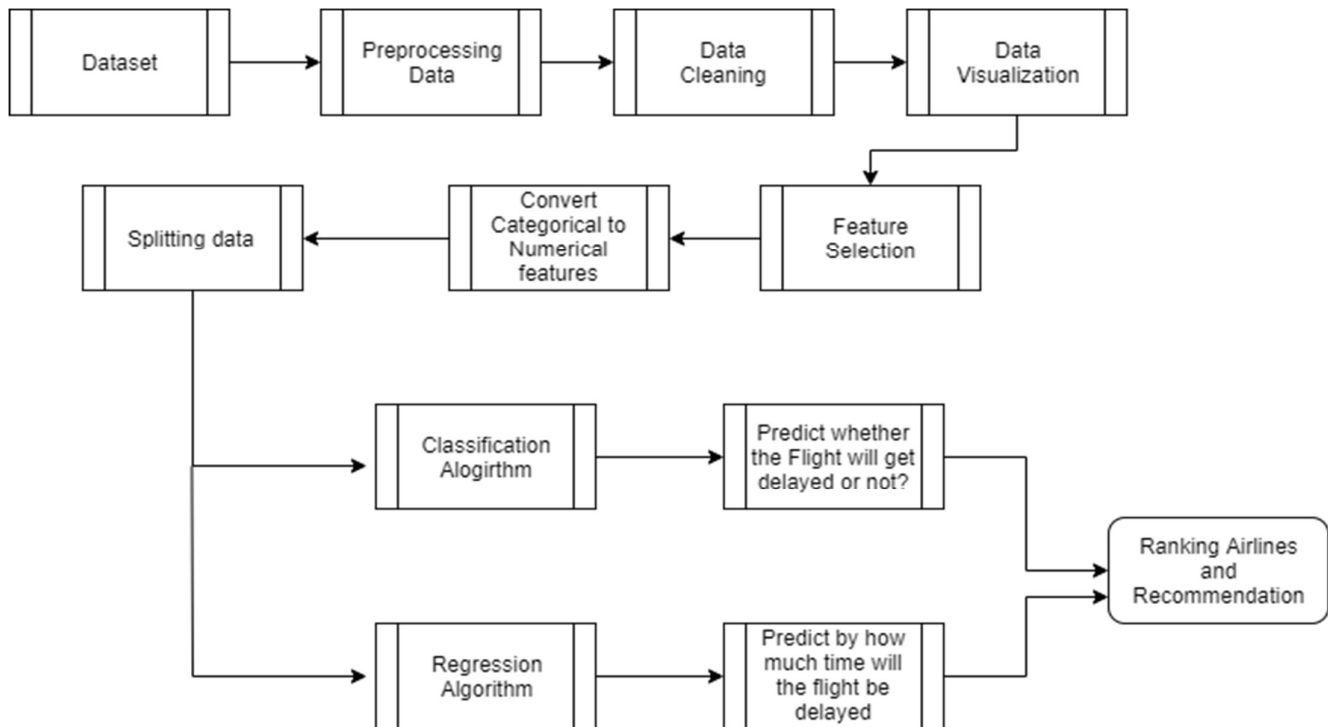


Fig 4.1 Architecture Diagram

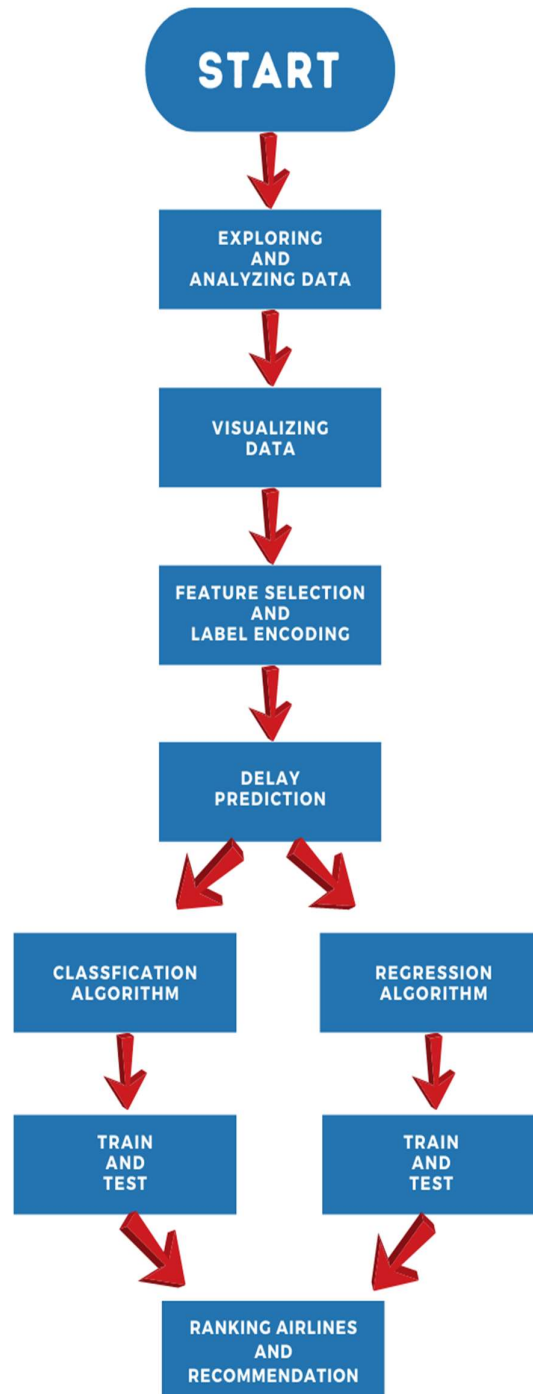


Fig 4.2 Flowchart Diagram

CHAPTER 5

MODULE DESCRIPTION

5.1 EXPLORING AND ANALYSING DATA

Data exploration is the initial step in data analysis, where users explore a large data set in an unstructured way to uncover initial patterns, characteristics, and points of interest. Data exploration can use a combination of manual methods and automated tools such as data visualizations, charts, and initial reports. The dataset contains small percentage of missing values for certain columns and these values are dropped as they make up a very small portion of the dataset. The exploring and analyzing steps occur as in fig 5.1

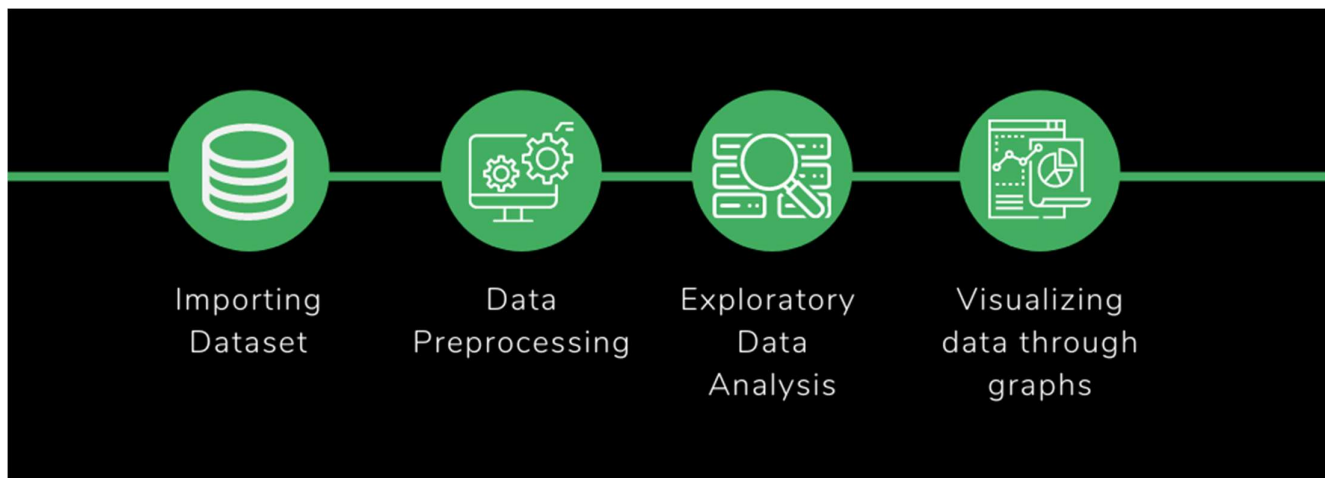


Fig 5.1 Exploring Data and Analyzing

5.1.1 Importing Dataset

The dataset was located from Kaggle, this dataset was collected from U.S department of transportation. The dataset tracks the performance of domestic flights within the united states. This dataset has information about the flowing information about the flights Year, Month, Day, Day Of Week, Airline, Flight Number , Tail Number, Origin Airport, Destination Airport, Scheduled Departure, Departure Time, Departure Delay, Taxi Out, Wheels Off, Scheduled Time, Elapsed Time, Air Time, Distance, Wheels On, Taxi In, Scheduled Arrival, Arrival Time, Arrival Delay, Diverted, Cancelled, Cancellation Reason, Air System Delay, Security Delay, Airline Delay, Late Aircraft Delay, Weather Delay.

5.1.2 Data Preprocessing

Handling missing values – The dataset contains small percentage of missing values for certain columns like Departure delay, taxi out and so on. These rows containing missing values are dropped as they make up a very small portion of the dataset

5.1.3 Exploratory Data Analysis

Exploratory data analysis, or EDA, is one of the most important things a data scientist needs to complete on any project. As the name implies, it is the deep exploration of your data, most often completed with visualizations such as bar plots, scatter plot, etc. From EDA, a scientist will uncover relationships between different variables, or possibly, important missing data. No matter where the data is sourced from, it is always important to complete in-depth EDA on your project. A Correlation matrix is a table showing the value of the correlation coefficient (Correlation coefficients are used in statistics to measure how strong a relationship is between two variables.) between sets of variables. Each attribute of the dataset is compared with the other attributes to find out the correlation coefficient. This analysis allows you to see which pairs have the highest correlation, the pairs which are highly correlated represent the same variance of the dataset thus we can further analyze them to understand which attribute among the pairs are most significant for building the model.

5.1.4 Visualizing Through Graphs

Data visualization is the technique to present the data in a pictorial or graphical format. It enables stakeholders and decision makers to analyze data visually. The data in a graphical format allows them to identify new trends and patterns easily. Several graphs are drawn to understand the orientation of the data. Fig 5.2 shows the scatter plot of airlines to no of minutes delayed. Overall proportion or percentage of airline companies in the dataset is represented as pie chart in Fig 5.3

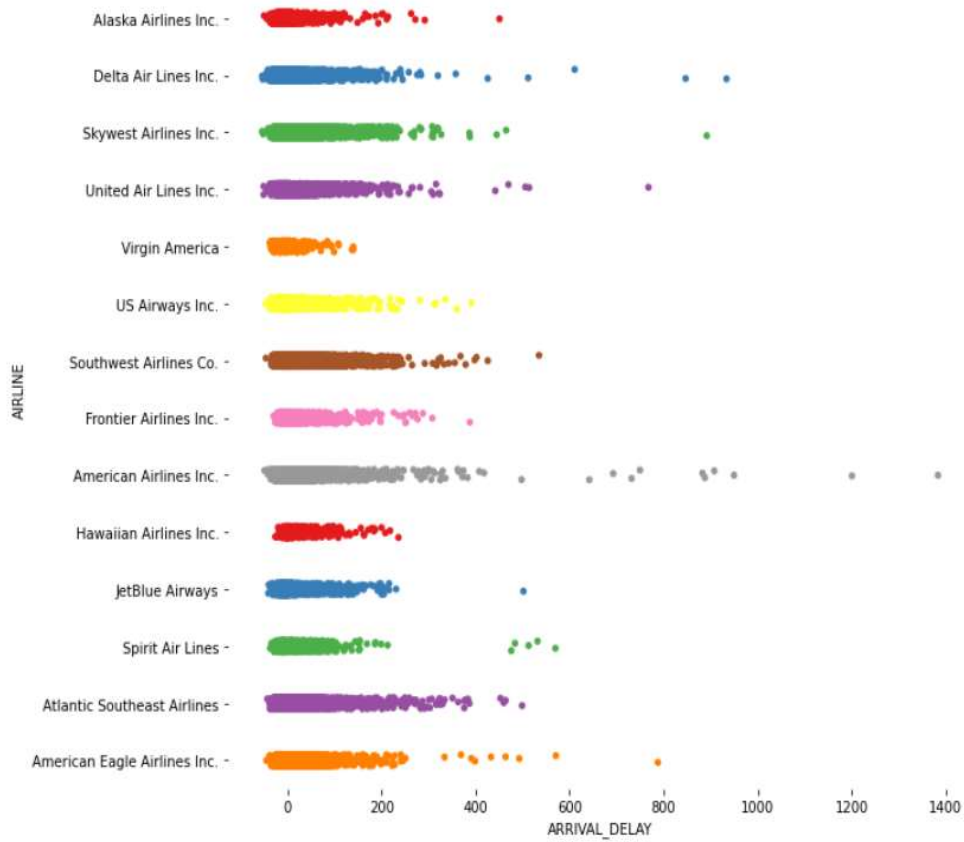


Fig 5.2 Scatter Plot

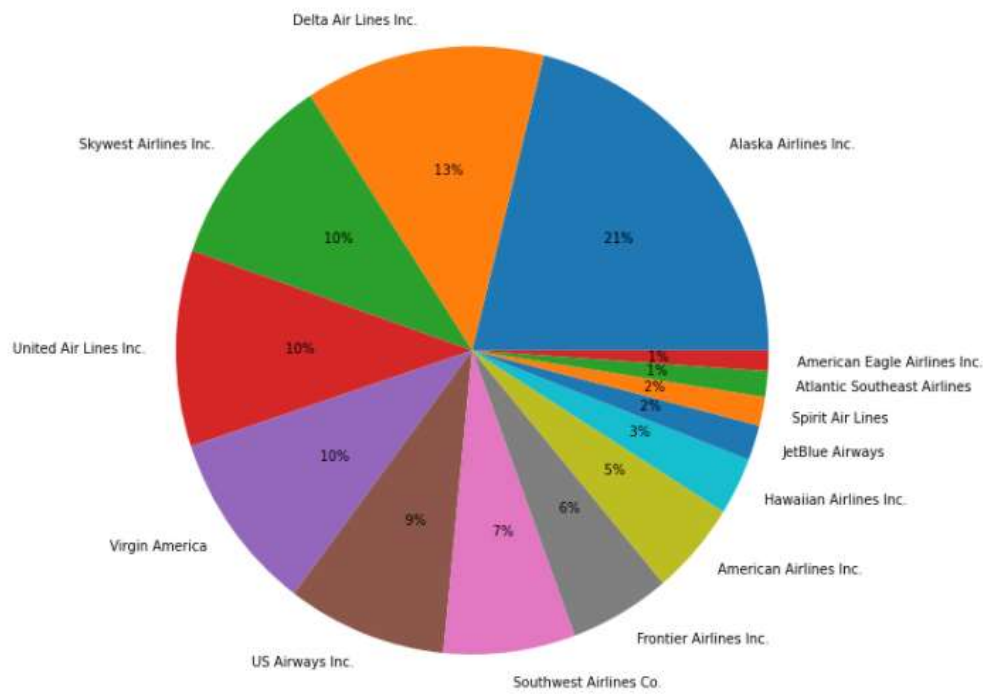


Fig 5.3 Pie chart

5.2 FEATURE SELECTION AND LABEL ENCODING

When building a machine learning model in real-life, it's almost rare that all the variables in the dataset are useful to build a model. Adding redundant variables reduces the generalization capability of the model and may also reduce the overall accuracy of a classifier. Furthermore adding more and more variables to a model increases the overall complexity of the model. The goal of feature selection in machine learning is to find the best set of features that allows one to build useful models of studied phenomena. So here certain features like Airline, Origin Airport, Destination Airport, Distance, Departure Delay, Scheduled Time, Airtime, Taxi Out are selected for prediction of flight delay. Among these features , some are categorial and some features are numerical features. Prediction can be done only when values in the data are in numbers. There are some categorial features which are Airline, Origin, Destination.

These are present as a String and not as a number so that it undergoes process called **label encoding**. Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. For Example the Airlines name which is a string is encoded in numerical format as in Fig 5.4.

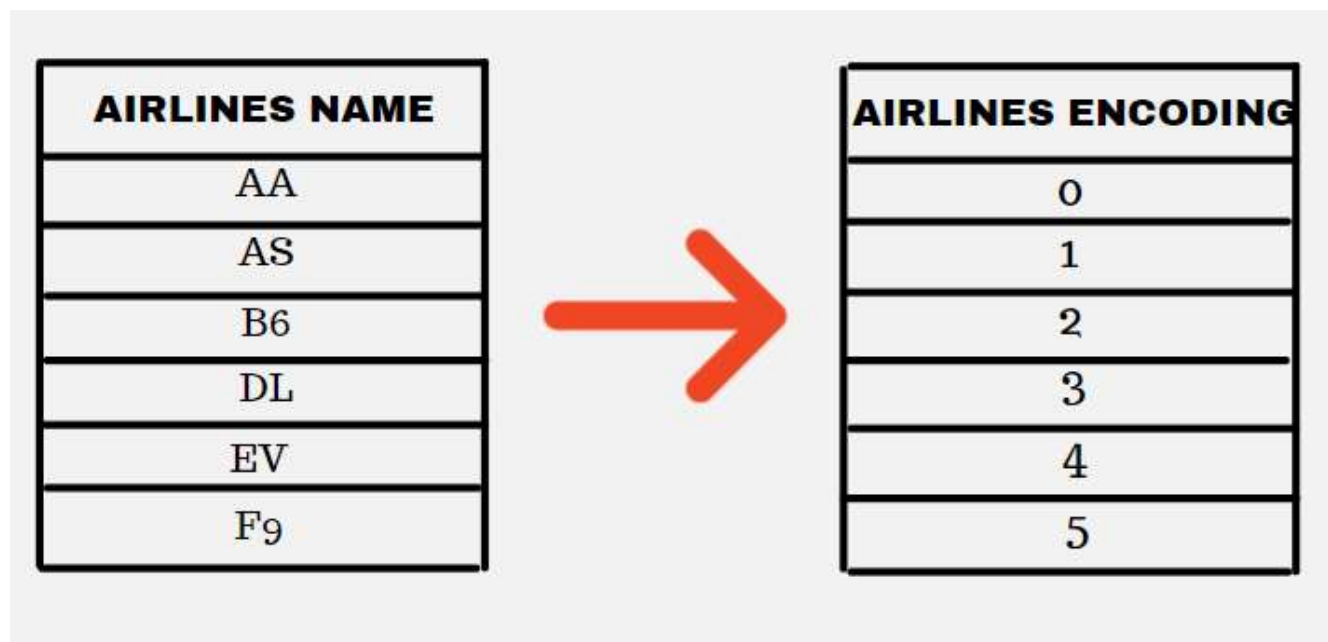


Fig 5.4 Label Encoding

5.3 PREDICTING FLIGHT DELAYS

Flight delay is predicted using 2 algorithms :

- ❖ Classification Algorithm
- ❖ Regression Algorithm

The prediction of whether the flight would get delayed or not is done using classification algorithms in supervised learning. Classification algorithms in machine learning use input training data to predict the likelihood that subsequent data will fall into one of the predetermined categories. The value of how much it gets delayed is predicted with regression technique, which is a statistical process for estimating the relationships between the dependent variables or criterion variables and one or more independent variables or predictors. Regression analysis explains the changes in criteria in relation to changes in select predictors. General steps of training and prediction is shown in Fig 5.5

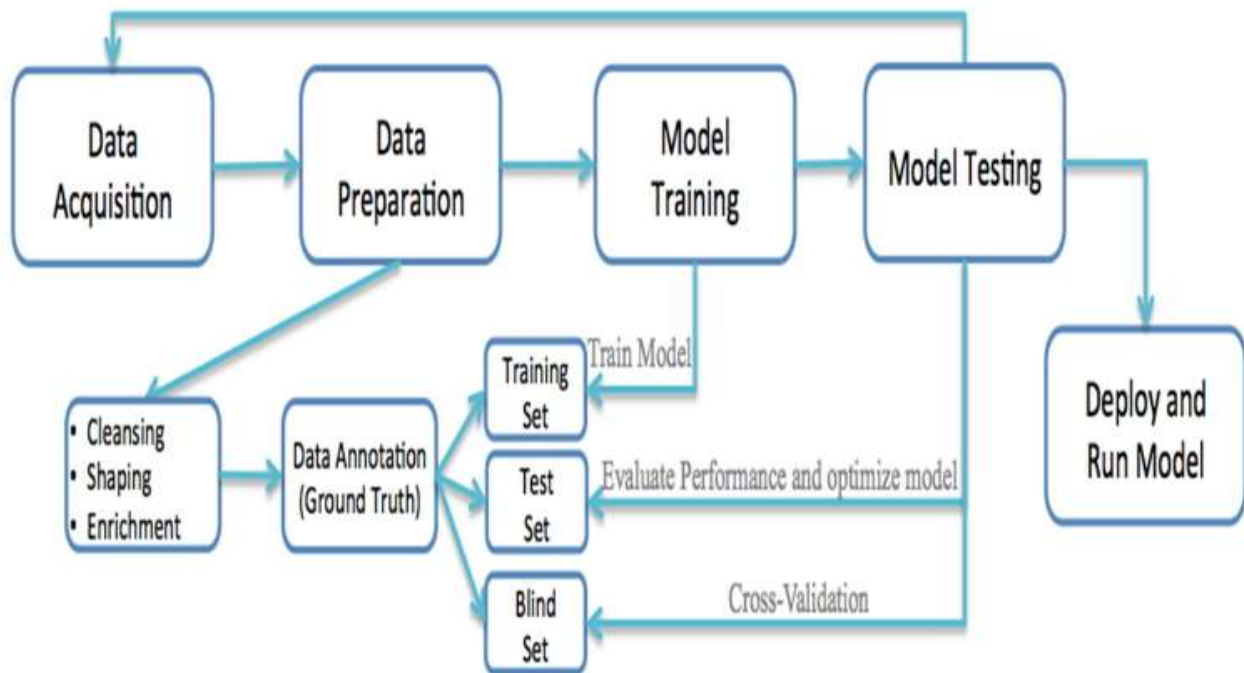


Fig 5.5 Block Diagram of the System

5.3.1 Predict Whether The Flight Will Get Delayed Or Not

For this process, classification algorithm is used to predict whether or not the flight will get delayed. Classification algorithm is generally used to classify among 2 sides of a system. So we use features – Airline, Origin_Airport, Destination_Airport, Distance, Departure_Delay, Scheduled_Time, Air_Time, Taxi_Out to predict the delay. We use 80-20 ratio for Training and Testing the model. 80 % of data is being trained and tested with 20% of the data.

Several Classification Algorithms have been trained and test which are:

❖ KNN:

K-nearest neighbors (KNN) algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems. However, it is mainly used for classification predictive problems in industry. K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. This produced a accuracy of around 81%.

❖ Decision Tree Classifier:

A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret. This produced a accuracy of around 85%.

❖ Random Forest Classifier:

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance. This produced a accuracy of around 87%.

❖ **XGBoost Classifier:**

Boosting is a sequential technique which works on the principle of an ensemble. It combines a set of weak learners and delivers improved prediction accuracy. At any instant t , the model outcomes are weighed based on the outcomes of previous instant $t-1$. XGBoost provides a wrapper class to allow models to be treated like classifiers or regressors in the scikit-learn framework. The XGBoost model for classification is called XGBClassifier. We can create and fit it to our training dataset. This produced a accuracy of around 92%.

The data is being trained and tested across 4 classification algorithms. Among this Four Classification Algorithms , XGBoost has the most accuracy of 92% so this model is used to predict the flight will get delayed or not.

5.3.2 Prediction Of Delay In Minutes

For this process, regression algorithm is used to determine how much minutes will the flight get delayed taking into account the following features: Airline, Origin_Airport, Destination_Airport, Departure_Delay, Scheduled_Time, Taxi_Out. We use 80-20 ratio for Training and Testing the model. 80 % of data is being trained and tested with 20% of the data. The Following regression techniques were used:

❖ **Decision Tree Regressor:**

We are focusing on decision tree regression only. So, decision tree regression is used for the continuous output problem. Continuous output means the output of the result is not discrete. Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. This produced a accuracy of around 90%.

❖ **Random Forest Regressor:**

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

❖ **XGBoost Regressor:**

XGBoost is a powerful approach for building supervised regression models. The validity of this statement can be inferred by knowing about its (XGBoost) objective function and base learners. XGBoost expects to have the base learners which are uniformly bad at the remainder so that when all the predictions are combined, bad predictions cancels out and better one sums up to form final good predictions.

5.3.3 Ratio Of Delay To Not – Delayed

In this part the user gives user origin, destination and airlines name, the XGBoost classification is used to predict how far the airlines for that particular origin and destination is got as output. The values like Airline, Origin_Airport, Destination_Airport, Distance, Departure_Delay, Scheduled_Time, Air_Time, Taxi_Out. Here with the origin, destination and airlines name other features are extracted from dataset predicted and then finally a pie-chart of delayed and non-delayed ratio is obtained for the user to see how far the flight will get delayed to plan accordingly.

5.4 RANKING AND RECOMMENDATION OF FLIGHTS

Users more often face a problem which is choosing the airlines for their journey. This To rank and recommend flights to the user we use certain features like flight delay, cancelled to operated ratio , speed which is calculated from airtime and distance. For a origin and destination , the airways are grouped by their company and for each airline company the score is calculated accordingly. Then the airlines are sorted according to their scores and recommendations are given to the user, based on which the user can select the airlines to book for his journey. This prediction will be helpful for giving a detailed analysis of the performance of individual airlines, airports, and then making a well-assessed decision. Moreover, apart from the assessment related to the passengers, delay prediction analysis will also help in important decision-making procedures necessary for every pivotal player in the air transportation system.

```
DESC_AIRLINE
Delta Air Lines Inc.          251.840153
American Airlines Inc.       163.534996
Spirit Air Lines              34.328611
Atlantic Southeast Airlines   6.222104
Name: SCORE, dtype: float64
```

Table 5.2 Recommendations of Airlines

In this Table 5.2 the source given by user gives origin airport as Hartsfield–Jackson Atlanta International Airport and destination airport as Dallas/Fort Worth International Airport, so for the given source and destination the score is calculated and is displayed for the users in descending order of the scores for his comfortness.

5.5 DEPLOYING USING DJANGO

Django is a high-level Python Web Development framework that encourages rapid development and clean, pragmatic design. It has been built by experienced developers, and takes care of much of the hassle of Web development. It is also free and open source. Django is a high-level Python framework that lets you build robust and scalable web applications. This is the most popular framework available in python. It follows the MVT or Model-View-Template pattern. It is closely related to other MVC frameworks like Ruby on Rails and Laravel. In the MVC framework, the view and model parts are controlled by the Controller but in Django, the tasks of a controller are handled implicitly by the framework itself. Django lets you create a number of applications under a single project. The application has all the functionalities to work independently. The app is considered as a package which you can reuse in other projects without making any major changes. It offers a flexible way of separating context and business logic - each layer has its own responsibilities. This is the greatest advantage of using Django for building web applications.

The user interface is the graphical layout of an application. The user interface, in the industrial design field of human-computer interaction, is the space where interactions between humans and machines occur. The goal of effective UI is to make the user's experience easy and intuitive, requiring minimum effort on the user's part to receive maximum desired outcome. A webpage using Django is created for user and airline transport system to get delays of flights and recommendations .In flight delay prediction page, user enters the source airport, destination airport and airlines name, so the percentage of delayed to not-delayed is returned to the user as a pie-chart to get an understanding about delays. In another page the airline transport system can get the minutes of delay of airline giving details which are needed to predict the delays. Finally in another page of the website the user gives his source and destination of the journey by which he can get recommendations of the flights which he can prefer to fly for his journey.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

6.1 CONCLUSION

This project is devoted to develop a predictive model to forecast flight delays. Data spanning of US domestic flights variables was used. Model based supervised machine learning algorithms are trained and tested, concluding that XGBoost model performs best prediction in both Classification and Regression Techniques obtaining an accuracy of nearly around 92%. Also the recommendations of flights were given for the user to get an idea of which flight to book for their journey.

6.2 FUTURE ENHANCEMENT

Although good results were obtained, there is a huge scope for future work. If weather and air traffic control information is made available we can then go on to predict arrival delay even without the inclusion of departure delay as an attribute. Also, we can progress into predicting if a flight will be delayed or cancelled based on weather factors like snow, rain, storms and so on. Although the model gives very good prediction accuracy, more variables can be considered to develop a predictive model. For example, Weather data can be extracted and used to better develop a predictive model for flight delay. The future scope of this study involves various approaches that can be used to analyze the data. Principal component analysis or transformation can be done to uncover hidden relations between variables. In addition, since the data is not exactly linear, artificial neural networks or Support vector machines can be used to analyze the effect of various variables on flight

APPENDIX I

SAMPLE CODING

DATA ANALYZING AND VISUALIZING CODE:

```
import datetime, warnings, scipy
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

flightsinfo = pd.read_csv("drive/My Drive/flights.csv")
airport = pd.read_csv('drive/My Drive/airports.csv')
airlines = pd.read_csv('drive/My Drive/airlines.csv')
flightsinfo.dtypes
airlinecompanies = airlines.set_index('IATA_CODE')['AIRLINE'].to_dict()
flightsinfo1=flightsinfo.dropna(subset
                                =
                                ["TAIL_NUMBER",'DEPARTURE_TIME','DEPARTURE_DELAY','TAXI_OUT','WHEELS_OFF','SCHEDULED_TIME',
                                'ELAPSED_TIME','AIR_TIME','WHEELS_ON','TAXI_IN','ARRIVAL_TIME','ARRIVAL_DELAY']) #dropping missing values in the dataset
flightsinfo_modified = flightsinfo1.dropna(subset
                                             =
                                             ['AIR_SYSTEM_DELAY','SECURITY_DELAY','AIRLINE_DELAY','LATE_AIRCRAFT_DELAY','WEATHER_DELAY'])
flightsinfo_modified = flightsinfo_modified.drop(['YEAR','MONTH','DAY','DAY_OF_WEEK','TAIL_NUMBER','SCHEDULED_DEPARTURE','DEPARTURE_TIME','SCHEDULED_TIME',
                                                  'SCHEDULED_ARRIVAL','ARRIVAL_TIME','DIVERTED','CANCELLED','CANCELLATION_REASON','FLIGHT_NUMBER','WHEELS_OFF','WHEELS_ON','AIR_TIME'],
                                                  axis = 1)
#Data Visualization
plt.figure(figsize=(10, 10))
```

```

axis = sns.countplot(y=Flights['Origin_city'], data = Flights,
                    order=Flights['Origin_city'].value_counts().iloc[:20].index,palette="Set2")
axis.set_yticklabels(axis.get_yticklabels())
plt.tight_layout()
plt.show()
axis = plt.subplots(figsize=(10,10))
sns.despine(bottom=True, left=True)
sns.stripplot(x="ARRIVAL_DELAY", y="AIRLINE",data = Flights,  dodge=True,
jitter=True,palette="Set1")
plt.show()
axis = plt.subplots(figsize=(18,12))

```

```

Flightscorr=pd.DataFrame(Flights[['DISTANCE','DEPARTURE_DELAY','SCHEDULED_
TIME','AIR_TIME','TAXI_IN','TAXI_OUT']])
Flightscorr['Is_Delayed'] = np.where(Flights['ARRIVAL_DELAY']<=0, 0,1)
sns.heatmap(Flightscorr.corr(),annot = True,cmap="YlGnBu")

```

```

b, t = plt.ylim() # discover the values for bottom and top
#b += 0.5 # Add 0.5 to the bottom
t -= 0.5 # Subtract 0.5 from the top
plt.ylim(b, t) # update the ylim(bottom, top) values
plt.show()
plt.figure(figsize=(10, 10))
axis = sns.countplot(x=Flights['ORIGIN_AIRPORT'], data =Flights,
                    order=Flights['ORIGIN_AIRPORT'].value_counts().iloc[:20].index)
axis.set_xticklabels(axis.get_xticklabels(), rotation=90, ha="right")
plt.tight_layout()
plt.show()
axis = plt.subplots(figsize=(10,14))
Name = Flights["AIRLINE"].unique()

```

```

size = Flights["AIRLINE"].value_counts()
plt.pie(size, labels=Name, autopct='%5.0f%%')
plt.show()

delay_type = lambda x: ((0,1)[x > 5], 2)[x > 45]
flightsinfo['DELAY_LEVEL'] = flightsinfo['DEPARTURE_DELAY'].apply(delay_type)

fig = plt.figure(1, figsize=(10,7))
ax = sns.countplot(x="AIRLINE", hue='DELAY_LEVEL', data=flightsinfo, palette=
["#00FF00", "#FFA500", "#FF0000"])

labels = ax.get_xticklabels()
ax.set_xticklabels(labels)
plt.setp(ax.get_yticklabels(), fontsize=12, weight = 'normal', rotation = 0);
plt.setp(ax.get_xticklabels(), fontsize=12, weight = 'normal', rotation = 0);
ax.xaxis.label.set_visible(False)
plt.ylabel('No. of Flights', fontsize=16, weight = 'bold', labelpad=10)

L = plt.legend()
L.get_texts()[0].set_text('on time (t < 5 min)')
L.get_texts()[1].set_text('small delay (5 < t < 45 min)')
L.get_texts()[2].set_text('large delay (t > 45 min)')
plt.show()

```

PREDICTION CODE:

```
from sklearn.externals import joblib
from sklearn.preprocessing import LabelEncoder
flightsinfo = pd.read_csv("drive/My Drive/flights.csv",nrows=50000)
airport = pd.read_csv('drive/My Drive/airports.csv')
airlines = pd.read_csv('drive/My Drive/airlines.csv')
from xgboost import XGBClassifier
classifierXGB = XGBClassifier(n_estimators=1000)
le = LabelEncoder()
flights=flightsinfo
Flights1 = flightsinfo
Flights1=flightsinfo.drop(['YEAR','MONTH','DAY','DAY_OF_WEEK','TAIL_NUMBE
R','DEPARTURE_TIME','WHEELS_OFF','WHEELS_ON','SCHEDULED_ARRIVAL','
ARRIVAL_TIME','CANCELLATION_REASON','AIR_SYSTEM_DELAY','SECURIT
Y_DELAY','AIRLINE_DELAY','LATE_AIRCRAFT_DELAY','WEATHER_DELAY'],
axis = 1)
Flights1['Is_Delayed'] = np.where(Flights1['ARRIVAL_DELAY']<=0, 0,1)
Flights2=Flights1
airlines_dict = dict(zip(airlines['IATA_CODE'],airlines['AIRLINE']))
airport_dict = dict(zip(airport['IATA_CODE'],airport['AIRPORT']))
Flights1 = Flights1.dropna(subset = ['TAXI_IN','ARRIVAL_DELAY'])
X=
Flights1.drop(['ELAPSED_TIME','DIVERTED','SCHEDULED_DEPARTURE','CANCE
LLED','FLIGHT_NUMBER','Is_Delayed','TAXI_IN'], axis = 1)
Flights1['DESC_AIRLINE'] = flightsinfo['AIRLINE'].apply(lambda x: airlines_dict[x])
Flights2['DESC_AIRLINE'] = flightsinfo['AIRLINE'].apply(lambda x: airlines_dict[x])
Flights1['Is_Delayed'] = np.where(Flights1['ARRIVAL_DELAY']<=0, 0,1)
```

```

X['AIRLINE']= le.fit_transform(X['AIRLINE'])
mapping = dict(zip(le.classes_, range(len(le.classes_))))
X['ORIGIN_AIRPORT'] = le.fit_transform(X['ORIGIN_AIRPORT'])
mapping = dict(zip(le.classes_, range(len(le.classes_))))
X['DESTINATION_AIRPORT'] = le.fit_transform(X['DESTINATION_AIRPORT'])
print(mapping['SEA'])
# X=X.drop(['ARRIVAL_DELAY'])
X = X.drop(['ARRIVAL_DELAY'],axis = 1)
y = Flights1['Is_Delayed']
classifierXGB.fit(X,y)
# print(X.columns)
joblib.dump(classifierXGB, 'drive/My Drive/xgbmodel.pkl')

```

```

from sklearn.externals import joblib
from xgboost import XGBClassifier
from sklearn.preprocessing import LabelEncoder
classifierXGB = XGBClassifier(n_estimators=1000)
def recommend(src,dest):
    flightsinfo = pd.read_csv("drive/My Drive/flights.csv",nrows=200000)
    airport = pd.read_csv('drive/My Drive/airports.csv')
    airlines = pd.read_csv('drive/My Drive/airlines.csv')
    flights=flightsinfo
    Flights1 = flightsinfo

```

```

Flights1=flightsinfo.drop(['YEAR','MONTH','DAY','DAY_OF_WEEK','TAIL_NUMBER',
'DEPARTURE_TIME','WHEELS_OFF','WHEELS_ON','SCHEDULED_ARRIVAL',
ARRIVAL_TIME','CANCELLATION_REASON','AIR_SYSTEM_DELAY','SECURITY_DELAY',
'AIRLINE_DELAY','LATE_AIRCRAFT_DELAY','WEATHER_DELAY'],
axis = 1)
Flights1['Is_Delayed'] = np.where(Flights1['ARRIVAL_DELAY']<=0, 0,1)

```

```

Flights2=Flights1
Flights1=Flights1.loc[(Flights1['ORIGIN_AIRPORT'] == src) |
(Flights1['DESTINATION_AIRPORT'] == dest)]
Flights2=Flights2.loc[(Flights2['ORIGIN_AIRPORT'] == src) &
(Flights2['DESTINATION_AIRPORT'] == dest)]
airlines_dict = dict(zip(airlines['IATA_CODE'],airlines['AIRLINE']))
airport_dict = dict(zip(airport['IATA_CODE'],airport['AIRPORT']))
Flights1 = Flights1.dropna(subset = ['TAXI_IN','ARRIVAL_DELAY'])
X=
Flights1.drop(['ELAPSED_TIME','DIVERTED','SCHEDULED_DEPARTURE','CANCE
LLED','FLIGHT_NUMBER','Is_Delayed','TAXI_IN', axis = 1)
Flights1['DESC_AIRLINE'] = flightsinfo['AIRLINE'].apply(lambda x: airlines_dict[x])
Flights2['DESC_AIRLINE'] = flightsinfo['AIRLINE'].apply(lambda x: airlines_dict[x])
Flights1['Is_Delayed'] = np.where(Flights1['ARRIVAL_DELAY']<=0, 0,1)
Flights2 = Flights2.dropna(subset = ['TAXI_IN','ARRIVAL_DELAY'])
X['AIRLINE']= le.fit_transform(X['AIRLINE'])
mapping = dict(zip(le.classes_, range(len(le.classes_))))
# print(mapping['AS'])
X['ORIGIN_AIRPORT'] = le.fit_transform(X['ORIGIN_AIRPORT'])
mapping = dict(zip(le.classes_, range(len(le.classes_))))
srcno=mapping[src]
X['DESTINATION_AIRPORT'] = le.fit_transform(X['DESTINATION_AIRPORT'])
X = X.drop(['ARRIVAL_DELAY'],axis = 1)
mapping = dict(zip(le.classes_, range(len(le.classes_))))
destno=mapping[dest]
y = Flights1['Is_Delayed']
X_test=X.loc[(X['ORIGIN_AIRPORT'] == srcno) & (X['DESTINATION_AIRPORT']
== destno)]
# return X_test
xgb_from_joblib = joblib.load('drive/My Drive/xgbmodel.pkl')

```



```

y_pred=xgb_from_joblib.predict(X_test)
Flights2['delayed']=y_pred
Flights2=Flights2.loc[(Flights2['delayed'] == 0)]
rank_airlines =
pd.DataFrame(Flights2.groupby('DESC_AIRLINE').count()['SCHEDULED_DEPARTU
RE'])

rank_airlines['CANCELLED']=Flights2.groupby('DESC_AIRLINE').sum()['CANCELLE
D']
rank_airlines['OPERATED']=rank_airlines['SCHEDULED_DEPARTURE']-
rank_airlines['CANCELLED']

rank_airlines['RATIO_OP_SCH']=rank_airlines['OPERATED']/rank_airlines['SCHEDUL
ED_DEPARTURE']
rank_airlines.drop(rank_airlines.columns[[0,1,2]],axis=1,inplace=True)
Flights2['FLIGHT_SPEED'] = 60*Flights2['DISTANCE']/Flights2['AIR_TIME']
rank_airlines['FLIGHT_SPEED'] =
Flights2.groupby('DESC_AIRLINE')['FLIGHT_SPEED'].mean()
Flights2.groupby('DESC_AIRLINE')[['ARRIVAL_DELAY','DEPARTURE_DELAY']].
mean()
rank_airlines['ARRIVAL_DELAY']=
Flights2.groupby('DESC_AIRLINE')['ARRIVAL_DELAY'].mean()
rank_airlines['ARRIVAL_DELAY']=rank_airlines['ARRIVAL_DELAY'].apply(lambda
x:x/60)
rank_airlines['FLIGHTS_VOLUME'] =
Flights2.groupby('DESC_AIRLINE')['FLIGHT_NUMBER'].count()
total = rank_airlines['FLIGHTS_VOLUME'].sum()
rank_airlines['FLIGHTS_VOLUME'] =
rank_airlines['FLIGHTS_VOLUME'].apply(lambda x:(x/float(total)))
for i in rank_airlines.columns:

```

a

=

```
rank_airlines.RATIO_OP_SCH*rank_airlines.FLIGHT_SPEED*rank_airlines.FLIGHTS  
_VOLUME
```

```
b = rank_airlines.ARRIVAL_DELAY
```

```
rank_airlines['SCORE'] = a/(1+b)
```

```
rank_airlines.sort_values(['SCORE'],ascending=False,inplace=True)
```

```
return rank_airlines['SCORE']
```

```
recommend('ANC','SEA')
```

```
def percentageofdelay(src,dest,airlinesname):
```

```
    flightsinfo = pd.read_csv("drive/My Drive/flights.csv",nrows=200000)
```

```
    airport = pd.read_csv('drive/My Drive/airports.csv')
```

```
    airlines = pd.read_csv('drive/My Drive/airlines.csv')
```

```
    flights=flightsinfo
```

```
    Flights1 = flightsinfo
```

```
Flights1=flightsinfo.drop(['YEAR','MONTH','DAY','DAY_OF_WEEK','TAIL_NUMBE  
R','DEPARTURE_TIME','WHEELS_OFF','WHEELS_ON','SCHEDULED_ARRIVAL','  
ARRIVAL_TIME','CANCELLATION_REASON','AIR_SYSTEM_DELAY','SECURIT  
Y_DELAY','AIRLINE_DELAY','LATE_AIRCRAFT_DELAY','WEATHER_DELAY'],  
axis = 1)
```

```
# Flights1['Is_Delayed'] = np.where(Flights1['ARRIVAL_DELAY']<=0, 0,1)
```

```
Flights1=Flights1.loc[(Flights1['ORIGIN_AIRPORT']==src)&  
(Flights1['DESTINATION_AIRPORT'] == dest) & (Flights1['AIRLINE'] ==  
airlinesname)]
```

```
Flights1['AIRLINE']= le.fit_transform(Flights1['AIRLINE'])
```

```
mapping = dict(zip(le.classes_, range(len(le.classes_))))
```

```
Flights1['ORIGIN_AIRPORT'] = le.fit_transform(Flights1['ORIGIN_AIRPORT'])
```

```
mapping = dict(zip(le.classes_, range(len(le.classes_))))
```

```
srcno=mapping[src]
```

```

Flights1['DESTINATION_AIRPORT']
le.fit_transform(Flights1['DESTINATION_AIRPORT'])
Flights1 = Flights1.drop(['ARRIVAL_DELAY'],axis = 1)

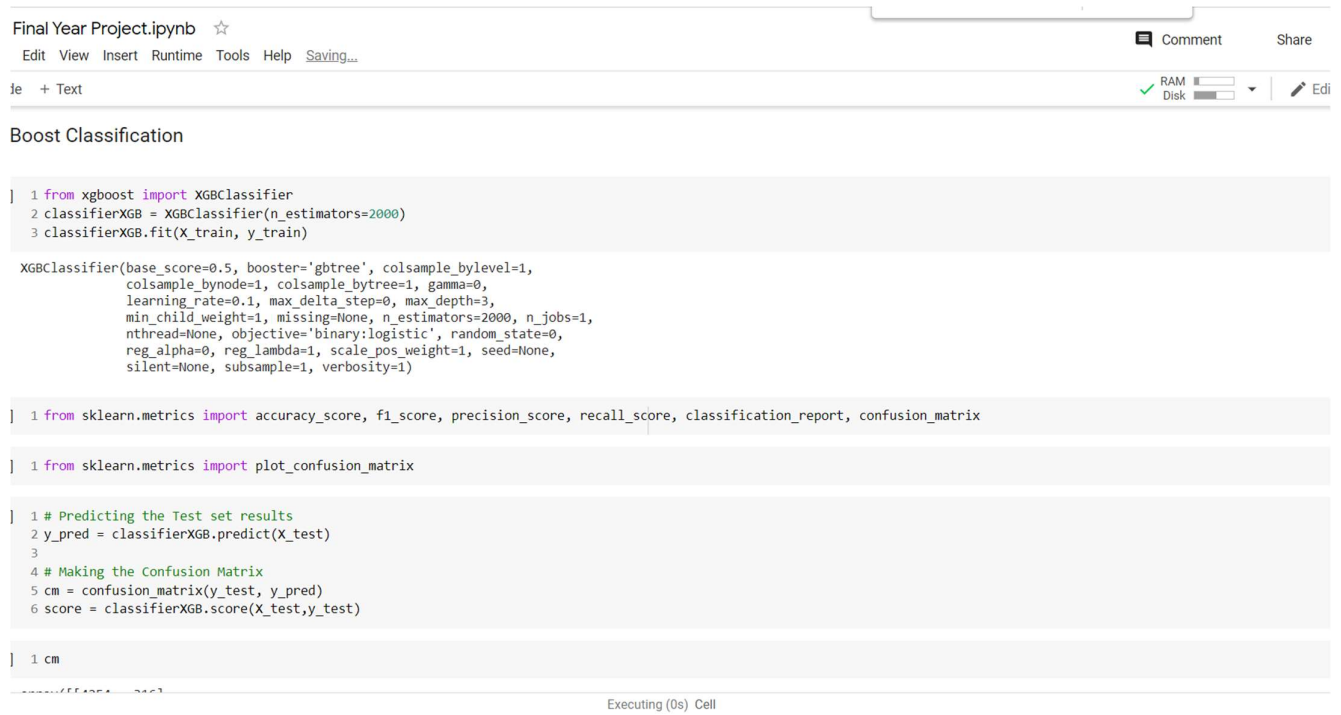
X_test=Flights1.drop(['ELAPSED_TIME','DIVERTED','SCHEDULED_DEPARTURE','
CANCELLED','FLIGHT_NUMBER','TAXI_IN'], axis = 1)
# return X_test
xgb_from_joblib = joblib.load('drive/My Drive/xgbmodel.pkl')
y_pred=xgb_from_joblib.predict(X_test)
Flights1['Is_Delayed']=y_pred
totalcount=Flights1['Is_Delayed'].count()
delayedcount=Flights1.loc[(Flights1['Is_Delayed'] == 1)]['Is_Delayed'].count()
nodelayedcount=Flights1.loc[(Flights1['Is_Delayed'] == 0)]['Is_Delayed'].count()
axis = plt.subplots(figsize=(10,14))
Name = ['Delayed','Not Delayed']
values = [(delayedcount/totalcount)*100,(nodelayedcount/totalcount)*100]
plt.pie(values,labels=Name,autopct='%5.0f%%')
plt.savefig('drive/My Drive/foo.png')
return plt.show()
percentageofdelay('LAX','BOS','AA')

```

APPENDIX II

SCREENSHOTS

XGBoost Classification Model



The screenshot shows a Jupyter Notebook interface with the title 'Final Year Project.ipynb'. The top bar includes a star icon, a menu (Edit, View, Insert, Runtime, Tools, Help), and a 'Saving...' status. On the right, there are 'Comment' and 'Share' buttons. Below the menu, the file type is 'Text' and there are RAM and Disk usage indicators. The notebook content is titled 'Boost Classification' and contains the following code:

```
1 from xgboost import XGBClassifier
2 classifierXGB = XGBClassifier(n_estimators=2000)
3 classifierXGB.fit(X_train, y_train)

XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
               colsample_bynode=1, colsample_bytree=1, gamma=0,
               learning_rate=0.1, max_delta_step=0, max_depth=3,
               min_child_weight=1, missing=None, n_estimators=2000, n_jobs=1,
               nthread=None, objective='binary:logistic', random_state=0,
               reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
               silent=None, subsample=1, verbosity=1)

1 from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, classification_report, confusion_matrix

1 from sklearn.metrics import plot_confusion_matrix

1 # Predicting the Test set results
2 y_pred = classifierXGB.predict(X_test)
3
4 # Making the Confusion Matrix
5 cm = confusion_matrix(y_test, y_pred)
6 score = classifierXGB.score(X_test, y_test)

1 cm
```

At the bottom, it shows 'Executing (0s) Cell'.

```

1 print("F1 score :", f1_score(y_test, y_pred, average="macro"))
2 print("Precision Score :", precision_score(y_test, y_pred, average="macro"))
3 print("Recall Score :", recall_score(y_test, y_pred, average="macro"))

```

```

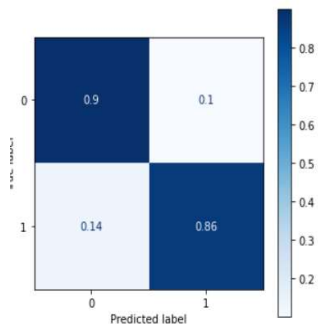
f1 score : 0.9250212841086357
precision Score : 0.9246809798463163
recall Score : 0.925542862347775

```

```

1 fig, ax = plt.subplots(figsize=(5, 5))
2 plot_confusion_matrix(classifierRF, X_test, y_test, normalize='true', cmap=plt.cm.Blues, ax=ax)
3 plt.show()

```



XGBoost Regression Model

XGBoost Regression

```

1 import xgboost as xgb
2 from sklearn.model_selection import train_test_split
3 from sklearn.metrics import mean_squared_error as MSE
4 xgb_r = xgb.XGBRegressor(objective='reg:linear',
5                           n_estimators = 1000)
6 xgb_r.fit(X_train, y_train)
7 predictedValues = xgb_r.predict(X_test)
8 print('MAE:', mean_absolute_error(y_test, predictedValues))
9 print('MSE:', mean_squared_error(y_test, predictedValues))
10 print('RMSE:', np.sqrt(mean_squared_error(y_test, predictedValues)))
11 print('R2:', r2_score(y_test, predictedValues))

```

[08:22:10] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.

MAE: 5.901617202833815
MSE: 63.80785536057175
RMSE: 7.98731540359702
R2: 0.9629014000775101

```

1 import matplotlib.pyplot as plt
2 fig, ax = plt.subplots()
3 ax.scatter(y_test, predictedValues)
4 ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
5 ax.set_xlabel('Actual')
6 ax.set_ylabel('Predicted')
7 plt.show()

```

```

1 df = pd.DataFrame({'Actual': y_test, 'Predicted': predictedValues})
2 df

```

Actual	Predicted
100	100
200	200
300	300
400	400
500	500
600	600
700	700
800	800
900	900
1000	1000
1100	1100
1200	1200
1300	1300
1400	1400

```
2nd Review.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
[151] 1 df = pd.DataFrame({'Actual': y_test, 'Predicted': predictedValues})
      2 df

      Actual Predicted
45566      -8.0    -8.540763
18460      11.0     8.928385
11968       -3.0     1.299929
11506       7.0     7.628021
39752       -8.0    -3.480436
...
2819       -9.0    -5.440205
35326       -8.0     1.638222
10945      176.0    193.697296
13067       21.0    12.561585
33514       7.0     1.258877
14602 rows x 2 columns

[152] 1 xgb_r.score(X_test, y_test)
      0.9641853094852266

1 X_val=np.array([[9,88,227,10,270,8,0,45,0]])
2 X_val
      array([[ 9.,  88., 227.,  10., 270.,   8.,  45.]])

[154] 1 y_val= fitResultotc.predict(X_val)
      2 y_val
      array([25.])

Functions

[155] 1 from sklearn.externals import Joblib
      2 from sklearn.preprocessing import LabelEncoder
      3 flightsinfo = pd.read_csv('drive/My Drive/flights.csv',nrows=50000)
      4 airport = pd.read_csv('drive/My Drive/airports.csv')
      5 airlines = pd.read_csv('drive/My Drive/airlines.csv')
      6 from xgboost import XGBClassifier
```

Home Page of Deployed Webapp



Ranking and Recommendation selection page

Ranking and Recommendation of Airlines

ORIGIN

Los Angeles

DESTINATION

Boston

Submit

Ranking and Recommendation output page

Recommendation of Flights

From : LAX

To: BOS

	Airlines	Score
0	American Airlines Inc.	217.754323
1	JetBlue Airways	201.786208
2	Virgin America	143.021080
3	Delta Air Lines Inc.	73.197517
4	United Air Lines Inc.	34.200783

Predict Delay Selection page

Home Delay Percentage Predict Delay Flight Recommendation

Predict Delay or Not

ORIGIN
Anchorage International Airport

DESTINATION
Seattle-Tacoma International Airport

AIRLINES
Alaska Airlines

DISTANCE
1448

DEPARTURE DELAY
-11.0

SCHEDULED TIME
205

AIRTIME
169.0

TAXI OUT
21.0

Submit

Predict Delay Output Page

127.0.0.1:5000/dornotresult x +

127.0.0.1:5000/dornotresult

Six Phrase - mySlate light blue html colo... 127.0.0.1 cp Login Using css page bre... Install NumPy, SciP... Log in | Codecademy Login Page Setting Up Bootstra... Other favorites

Prediction Result

Flight will not get delayed

Delay and Non Delay Ratio Input Page

[Home](#) [Delay Percentage](#) [Predict Delay](#) [Flight Recommendation](#)

Percentage Delay in Journey

ORIGIN

Atlanta

DESTINATION

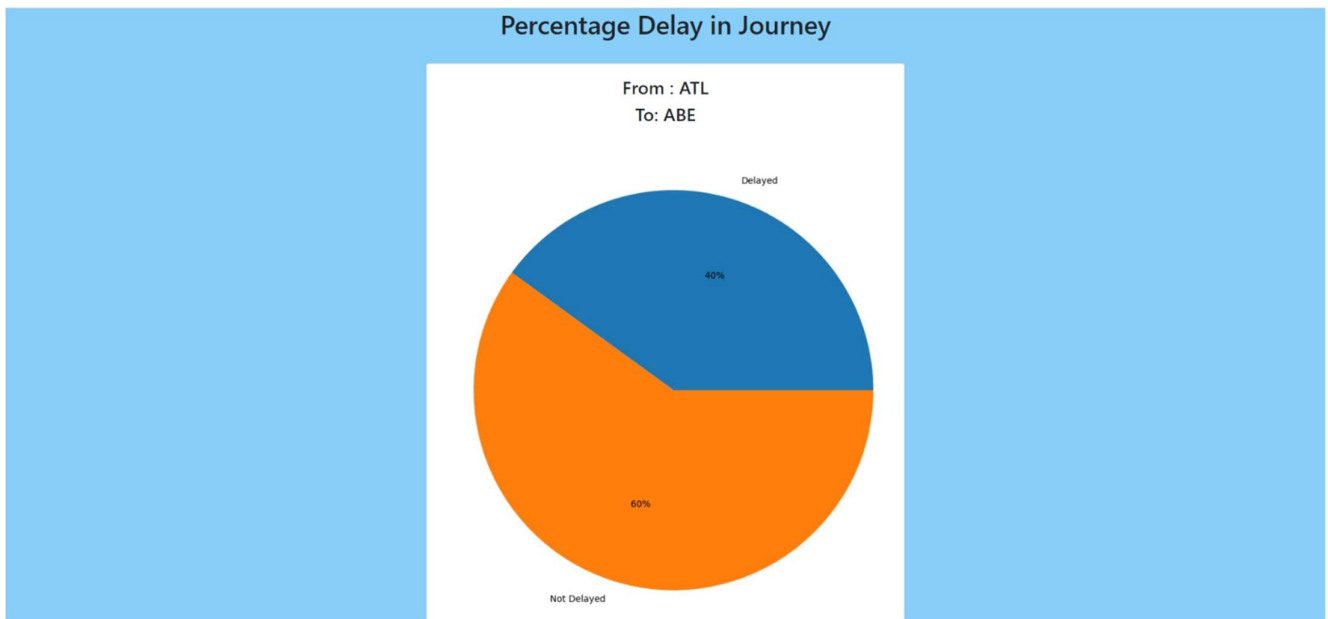
Lehigh Valley

AIRLINES

Express Jet

Submit

Delay and Non Delay Ratio Output Page



REFERENCES

1. J.E. Aronson, “A Survey Of Dynamic Network Flows,” Ann. Opns. Res. 20, 1–66
2. A. Barrat, M. Barthe’Lemy, R. Pastor-Satorras, And A. Vespignani, “The Architecture Of Complex Weighted Networks”, Pnas March 16, 2004 101 (11) 3747-3752;
3. Dimitris Bertsimas , Sarah Stock Patterson “The Traffic Flow Management Rerouting Problem In Air Traffic Control: A Dynamic Network Flow Approach”
<https://pubsonline.informs.org/doi/abs/10.1287/trsc.34.3.239.12300>
4. L. R. Ford And D. R. Fulkerson, Flows In Networks, Princeton University Press, Princeton, New Jersey, 1958.
5. L. Macdonald, “Collaborative Decision Making In Aviation,” J. Air Traffic Control 40, 12–17 (1998).
6. Pablo Fleurquin , Jose Javier Ramasco And Víctor M. Eguíluz, “ Systemic Delay Propagation In The Us Airport Network” , 10.1038/Srep01159
7. Stanislav Tarasevych; Ivan Ostroumov, “A Light Statistical Method Of Air Traffic Delays Prediction,”. 2020 Ieee 2nd International Conference On System Analysis & Intelligent Computing (Saic)
8. Shuai Li , Yuelei Xu, Mingming Zhu, Shiping Ma, And Hong Tang, “Remote Sensing Airport Detection Based On End-To-End Deep Transferable Convolutional Neural Networks”, Ieee Geoscience And Remote Sensing Letters (Volume: 16, Issue: 10, Oct. 2019)
9. P. Vranas, D. Bertsimas, And A. R. Odoni, “Dynamic Ground-Holding Policies For A Network Of Airports,” Transp. Sci. 28, 275–291 (1994b)
10. P. Vranas, D. Bertsimas, And A. R. Odoni, “The Multiairport Ground-Holding Problem In Air Traffic Control,” Opns. Res. 42, 249–261 (1994a).