

A DECISION TREE CLASSIFIER APPROACH TO ANALYZE THE AIRLINE DATAS

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering

by

Name: NOMI GANDRETI

(Reg. no. 37110220)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119

MARCH - 2021



SATHYABAMA



INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI- 600119

www.sathyabama.ac.in

DEPARTEMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **NOMI GANDRETI (37110220)** who carried out the project entitled —**A DECISION TREE CLASSIFIER APPROACH TO ANALYZE THE AIRLINE DATA**—S II under my supervision from **August 2020** to **March 2021**.

Internal Guide

Name: E. BRUMANCIA, M.E., (Ph.D).

HEAD OF DEPARTEMENT

Dr. S. VIGNESHWARI, M.E., Ph.D.

Dr. L. Lakshmanan, M.E., Ph.D.

Submitted for Viva Voice Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **NOMI GANDRETI (37110220)** hereby declare that the Project Report entitled —**A
DECISION TREE CLASSIFIER APPROACH TO ANALYZE THE AIRLINE DATA**”S II
done by me under the guidance of **E. BRUMANCIA, M.E., (Ph.D).**,

Dr.S.VIGNESHWARI, M.E., Ph.D., and Dr.L.Lakshmanan, M.E., Ph.D. at Sathyabama
Institute of Science and Technology is submitted in partial fulfillment of the requirements
for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

NAME OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.S.Vigneshwari M.E., Ph.D.,** and **Dr.L.Lakshmanan M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **E. BRUMANCIA M.E., (Ph.D).,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The purpose of this study is to analyze the US domestic flights dataset using pysparkdataframes. It aims to predict which flight/flight carrier is most likely to be cancelled or delayed. Using Pyspark, a big data technology tool, it can therefore work with RDDs/Dataframes/Datasets. This analysis will help us find out what are the top airlines with the maximum delays/cancellations in the US. Also, it will investigate whether there are any factors that cause flight delays/cancellations. Further, it predicts if a flight will get cancelled or not. In order to build the predictive system Machine learning models were incorporated, where it was modeled as a binary classification problem with categorical variables, logistic regression, decision tree classifier, random forest and gradient boosted trees. So, the main aim and purpose of this study is to predict the chances of a flight being cancelled. There are many factors that influence it.. This study considers such different factors like the weather condition, delay in flights etc. It considers all such factors and predict the chances of a flight to be cancelled. This will help the customer save his money as they will not be provided with a refund for the cancelled flight. This will help the customer save their money and time. After comparing all the algorithms it gives 63% of accuracy.

TABLE OF CONTENTS

CHAPTER No.	TITL E	PAGE No.
	ABSTRCT	V
	LIST OF FIGURES	Viii
1	INTRODUCTION	1
2	LITERATURE SURVEY	3
3	AIM AND SCOPE	6
	3.1 AIM OF THE PROJECT	6
	3.2 SCOPE AND OBJECTIVES	6
	3.3 EXISTING SYSTEM	6
	3.4 PROPOSED SYSTEM	6
	3.5 ADVANTAGES OF PROPOSED SYSTEM	7
4	METHODOLOGY AND SYSTEM IMPLEMENTATION	8
	4.1 SYSTEM ARCHITECTURE	8
	4.2 SYSTEM REQUIREMENTS	8
	4.2.1 HARDWARE REQUIREMENTS	8
	4.2.2 SOFTWARE REQUIREMENTS	9
	4.3 REQUIREMENT ANAYSIS	9
	4.3.1 FUNCTIONAL REQUIREMENTS	9

	4.3.2 NON-FUNCTIONAL REQUIREMENTS	11
	4.4 SOFTWARE ENVIRONMENT	11
	4.5 MODULES	21
	4.6 SYSTEM DESIGN AND TESTING PLAN	23
	4.7 SYSTEM STUDY	24
	4.8 DATA FLOW DIAGRAM	26
	4.9 UML DIAGRAMS	29
	4.10 SEQUENCE DIAGRAM	31
	4.11 SYSTEM TESTING	32
	4.11.1 TYPES OF TESTS	32
5	RESULTS AND DISCUSSION	36
6	CONCLUSION AND FUTURE WORK	42
	REFERENCES	43
	APPENDICES	44
	A. SOURCE CODE	44

LIST OF FIGURES

FIGURE No.	TITLE	PAGE No.
4.1	ARCHITECTURE DIAGRAM	8
4.2	LEVEL - 1 DIAGRAM	27
4.3	LEVEL – 2 DIAGRAM	27
4.4	LEVEL – 3 DIAGRAM	28
4.5	LEVEL – 4 DIAGRAM	28
4.6	USE CASE DIAGRAM	30
4.7	SEQUENCE DIAGRAM	31
5.1	DATASET	36
5.2	BALANCED DATASET	37
5.3	IMBALANCED DATASET	37
5.4	TICKET FARE PREDICTION	38
5.5	DELAY TIME	39
5.6	CANCELLATION FACTORS	39
5.7	PIE-CHART OF CANCELLATION FACTORS	40
5.8	CANCELLATION PREDICTION	40
5.9	CANCELLATION PROBABILITY	41

CHAPTER 1

INTRODUCTION

People love to travel across the world. And the first transportation medium they get reminded of, is the airlines. Airlines have been a booming trend due to its advancement. And due to this there has been a massive increase of the passengers travelling through the airlines. They plan their trips, or meetings in advance as they may not get a ticket in the last minute. But what if the flight gets cancelled when you are about to leave to the airport? So, the main aim and purpose of this study is to predict the chances of a flight being cancelled. There are many factors that influence it.. This study considers such different factors like the weather condition, delay in flights etc. It considers all such factors and predict the chances of a flight to be cancelled. This will help the customer save his money as they will not be provided with a refund for the cancelled flight. This will help the customer save their money and time.

Data for flights are generated regularly in large volumes. This is to analyses all that data. Using some machine learning we will try to predict based on the factors. The study needs a large dataset in order to achieve this. So, a dataset was taken from Kaggle on NBC from 2009 to 2015, included 7 CSV files with data and a size of 7 GB. Airlines subject to delay information, location details (origins and locations) and cancellation (tagged as a new reason code) It also contained technical details such as the time it was empty and the aircraft on the grounds. In my approach and Apache Spark, especially when using PySpark EMR cluster support. But this can be the power of the use of it is, it is divided falls, for many orders of magnitude better than in the knowledge of the structure of memory struggled with a lot of laborious literature. As an analysis engine that allows us to parallelize an important role in information processing, clusters are distributed., transformed, and were loaded. The datasets

from the Kaggle website are loaded in the Jupyter notebook. This performed exploratory analysis and model prediction.

The dataset was taken from Kaggle, comprised 7 CSV files containing data from 2009 to 2015, and was about 7GB in size. It contained information about airlines, delay information, location details (origins and destinations), and cancellation (reasons labelled as cancellation codes). It also had certain technical details such as the time that the plane was stationary on ground.

CHAPTER 2

LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

In the article —**Data analytics development of FDR (Flight Data Recorder) data for airline maintenance operations**|| by Chang-Hun Lee, Hyo-Sang Shin, AntoniosTsourdos and Zakwan Skaf in 2017 proposed a data analytics development to detect unusual patterns of flights from a vast amounts of FDR (flight data recorder) data for supporting airline maintenance operations. A fundamental rationale behind this development is that if there are potential issues on mechanical parts of an

aircraft during a flight, evidences for these issues are most likely included in the FDR data.

Therefore, the data analysis of FDR data enables to detect the potential issues in the aircraft before they occur.

To this end, in a data pre-processing step, a data filtering, a data sampling, and a data transformation are sequentially performed. And then, in this analysis, all time series data in the FDR are classified into three types: a continuous signal, a discrete signal, and a warning signal. For each type of signal, a high-dimensional vector by arranging the time series data is chosen as features.

In the feature section process, a correlation analysis, a correlation relaxation, and a dimension reduction are sequentially conducted. Finally, a type of k-nearest neighbor approach is applied to automatically identify the FDR data in which the unusual flight patterns are recorded from a large amount of FDR data. The proposed method is tested with using a realistic FDR data from the NASA's open database.

In another proposed model named —Big Data Analytics on Aviation Social Media from 2016, the authors Sien Chen, Yinghua Huang and Wengqiang Huang used sentiment analysis to study the case of China Southern Airlines on Sina Weibo, and illustrating Weibo users' attitudes toward China Southern Airlines.

This study also discusses the practical implications for airlines to manage their social media platforms. With the combination of a passenger's social media value and passenger's other offline behavior information, we can establish a comprehensive profile for travelers.

In the year 2019, ZudhaAuliaRachman and Arviansysh studied to quantitatively evaluate the operational efficiency in the airline scheduling and execution process by implementing big data analytics approach. Parameters for calculation are obtained from prior studies. These parameters are calculated using Data Envelopment Analysis (DEA) method to get efficiency scores for each operation process every month. Finally, they argue that the data analytics approach is beneficial to be implemented in airlines and find a decreasing trend in the efficiency score of the sample airline during 2017-2018.

The study concludes that agile development in Data Analysis is an important area in the present world of increasing data. There are many open challenges and gaps. In particular this analysis is beneficial not only for the industry but also for a common man.

CHAPTER 3

AIM AND SCOPE

3.1 AIM OF THE PROJECT:

The main aim of this project is to predict the cancellation and delay of the US airlines. By this project can know the probability of cancellation of the flight as we are not going to get the refund if the flight is cancelled because of weather conditions or any airline traffic problems. So we can book the flight that have less cancellation probability.

3.2 SCOPE AND OBJECTIVES:

The project give the accuracy of the cancel and delay of the US airlines. It analysis the cancellation of flights with the help of the past few years of airline data.

3.3 EXISTING SYSTEM

In Existing system, an airport has huge amount of data related to number of flights, data and time of arrival and dispatch, flight routes, number of airports operating in each country, list of active airlines in each country. The problem they faced till now it's, they have ability to analyze limited data from databases.

3.4 PROPOSED SYSTEM

This project is to analyze data regarding domestic flights to predict flight cancellations in the US. Our project will analyze the cancellation including delay time, number of passengers, airline carrier, scheduled and actual departure and arrival times, and reason of delay. Using these values, plan to future calculate delays

caused by the airport, get the count of the flights delayed with respect to the total number of flights etc.

Our project also aim to calculate the average airline delay time, which airline has the most number of delays, at what time of the day do we see more number of delays in flights. This project also checks if there are any seasonal delays i.e if delays are mostly during the holiday season. Spark's MLlib are used to create powerful machine learning models alongside the DataFrame syntax and spark

3.5 ADVANTAGES OF PROPOSED SYSTEM

- By using airline data analysis we can save time of users
- Cost Savings

CHAPTER 4

METHODOLOGY AND SYSTEM IMPLEMENTATION

4.1 SYSTEM ARCHITECTURE

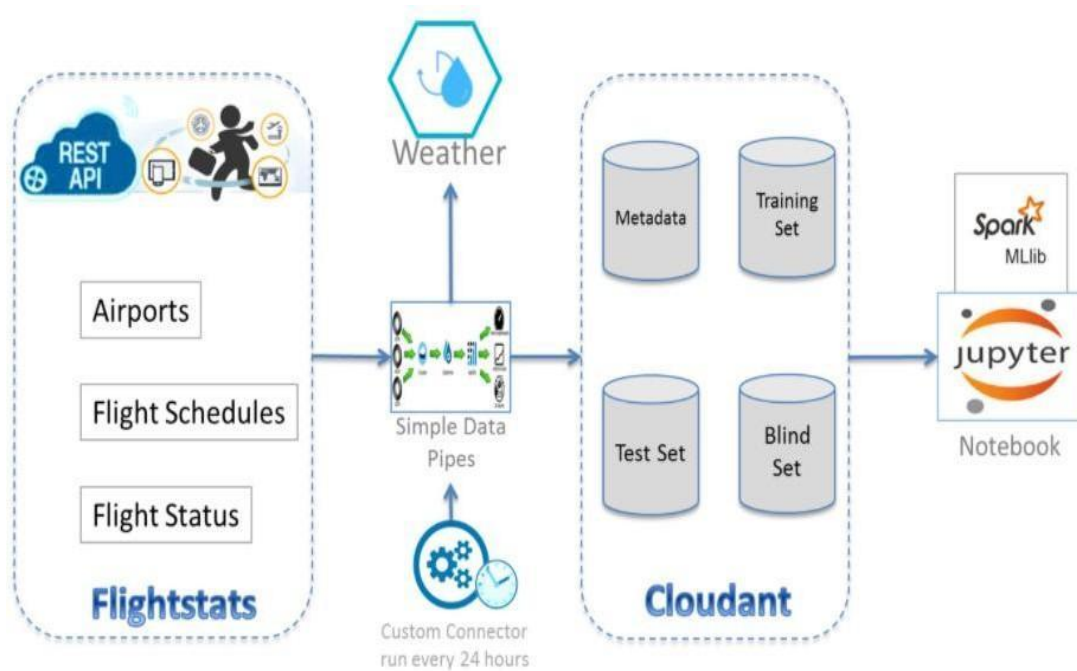


Fig 4.1 Architecture Diagram

4.2 SYSTEM REQUIREMENTS

4.2.1 HARDWARE REQUIREMENTS

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Monitor : 15" LED
- Input Devices : Keyboard, Mouse
- Ram : 4 GB.

4.2.2 SOFTWARE REQUIREMENTS

- Operating system: Windows 7/10.
- Coding Language :Python

4.3 REQUIREMENT ANALYSIS:

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

4.3.1 FUNCTIONAL REQUIREMENTS:

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

- **Usability**

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

- **Robustness**

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

- **Security**

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

- **Reliability**

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

- **Compatibility**

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

- **Flexibility**

The flexibility of the project is provided in such a way that it has the ability to run on different environments being executed by different users.

- **Safety**

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

4.3.2 NON- FUNCTIONAL REQUIREMENTS:

- **Portability**

It is the usability of the same software in different environments. The project can be run in any operating system.

- **Performance**

These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

- **Accuracy**

The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

- **Maintainability**

Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyse, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

4.4 SOFTWARE

ENVIRONMENT Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.

- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.

- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python <https://www.python.org>.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

```
$ python
```

```
Python2.4.3(#1,Nov112010,13:34:43)
```

```
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>>
```

Type the following text at the Python prompt and press the Enter –

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!")**; However in Python version 2.4.3, this produces the following result –

```
Hello, Python!
```

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file –

```
print"Hello, Python!"
```

Let's assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

```
$ python test.py
```

This produces the following result –

Hello, Python!

Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods –

Sr.No	Methods & Description
1	GET Sends data in unencrypted form to the server. Most common method.
2	HEAD Same as GET, but without response body
3	POST Used to send HTML form data to server. Data received by POST method is not cached by server.

4	PUT Replaces all current representations of the target resource with the uploaded content.
5	DELETE Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

```
<html>

<body>

<form action="http://localhost:5000/login" method="post">

<p>Enter Name:</p>

<p><input type="text" name="nm"/></p>

<p><input type="submit" value="submit"/></p>

</form>

</body>

</html>
```

Now enter the following script in Python shell.

```

from flask import Flask, redirect, url_for, request

app = Flask(__name__)

@app.route('/success/<name>')

def success(name):

    return 'welcome %s' % name

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

def login():

    if request.method == 'POST':

        user = request.form['nm']

        return redirect(url_for('success', name = user))

    else:

        user = request.args.get('nm')

        return redirect(url_for('success', name =

        user))

if __name__ == '__main__':

    app.run(debug = True)

```

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.



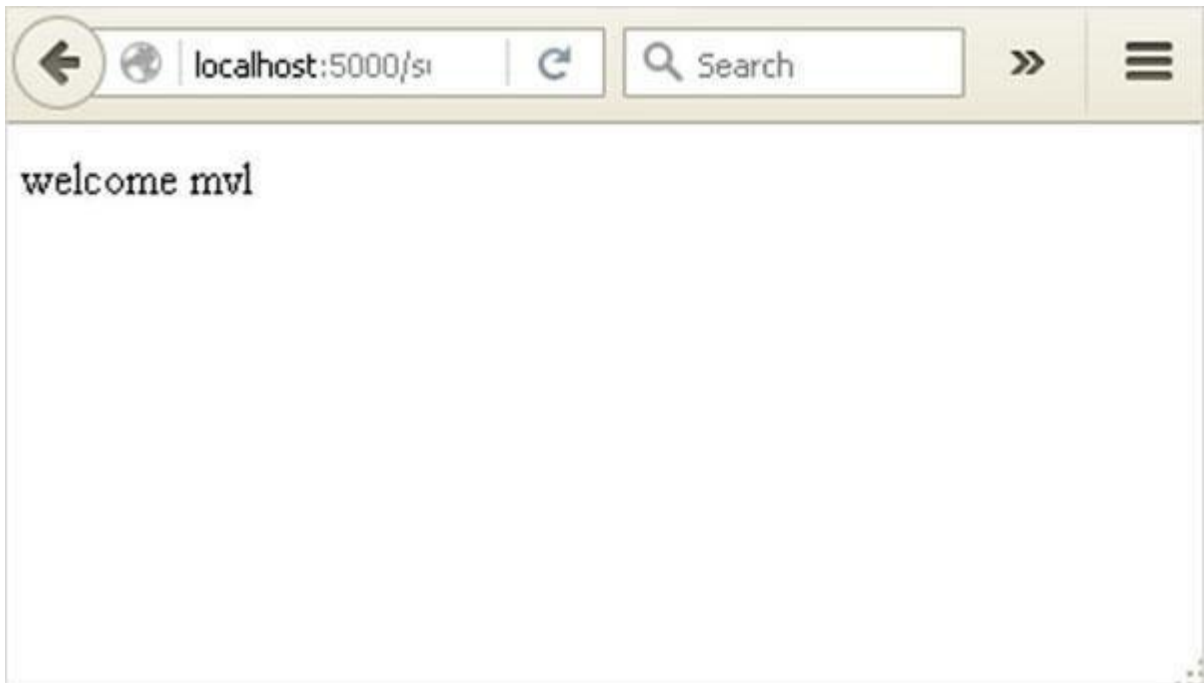
Enter Name:

Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of `_nm` parameter obtained from the form data is obtained by –

```
user = request.form['_nm']
```

It is passed to „**/success**“ URL as variable part. The browser displays a **welcome** message in the window.



Change the method parameter to „**GET**“ in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of `__nm'` parameter is now obtained by –

```
User = request.args.get('__nm')
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to `__nm'` parameter is passed on to `__/_success'` URL as before.

4.5 MODULES

- Data Profiling
- Data Wrangling
- Data Cleaning and Preparation
- Statistical Data Analysis
- Model Prediction
- RandomForest model

DATA PROFILING

To get a quick glance at the data, the 2009 csv files are loaded into a Resilient Distributed Dataset (RDD) which is Spark's representation of a dataset that is distributed across RAM, in the memory of a cluster of many machines. Through this Spark session, I found that there was about 27 unnamed variables and multiple null values. After combining the dataset (2009 to 2015 data), I created 3 major categorical CSV files namely: flight-cancellation, flight-delays, and flight-profile. This was essential to debug.

Data Cleaning and Preparation

Originally, there was 28 variables to work with. After removing the unnamed columns, the remaining 19 were checked for null values (as shown below). In addition, only columns with relevant data about the flight profile, delay, and cancellation information were kept. With time sensitive data, it was hard to recreate the null values with either mean or median of the data, so I had to drop these records which boils down to over 6+ million records to still work with.

After cleaning out the date column, it contained only the yearly information which was used in the analysis.

Statistical Data Analysis

- From the profile sub-dataset, there were 61,556,964 flight profile records including 7605 US domestic flights in total, around 380 unique origins and 378 unique destinations. This information was populated in temporary tables inside the Spark session to query specifics.
- From the delay sub-dataset, there were negative values in the departure delay column indicating that around 3,1204,918 flights were actually ahead of their scheduled time; that is roughly 50% of the dataset. This means that 50% of the time flights were delayed.
- Within the cancellation sub-dataset, the cancel column is categorical data represented by either a 0 or a 1 (not cancelled or cancelled respectively).

MODEL PREDICTION

To predict whether the flight would be cancelled or not, the dataset was modeled as a binary classification problem with categorical variables.

1. Prepare the data for machine learning: use StringIndexer, OneHotEncoder, and VectorAssembler to transform our features
2. Split the data into a 70/30 test and train ratio

3. Apply models: logistic regression, decision tree classifier, random forest and gradient boosted trees
4. Compare all the models' accuracy for predicting cancellations.

Decision Tree Classifiers

Decision Tree Classifiers can handle categorical variables well and are able to capture non-linearity. With the aid of the `pyspark.ml.classification` library, the `DecisionTreeClassifier` was imported. This was essential to show the probability of each prediction and evaluate the model.

4.6 SYSTEM DESIGN AND TESTING

PLAN INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

What data should be given as input?

How the data should be arranged or coded?

The dialog to guide the operating personnel in providing input.

Methods for preparing input validations and steps to follow when error occur.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the Future.

- Signal important events, opportunities, problems, or warnings.

- Trigger an action.

- Confirm an action.

4.7 SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4.8 DATA FLOW DIAGRAMS

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



Fig 4.2 Level – 1 Diagram

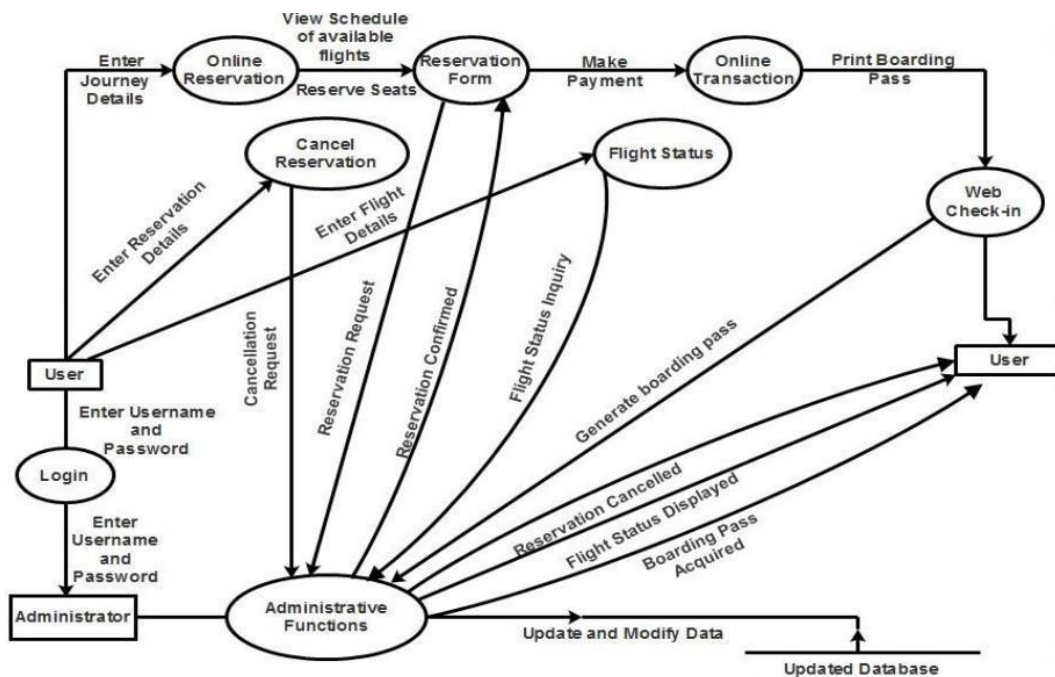


Fig 4.3 Level – 2 Diagram

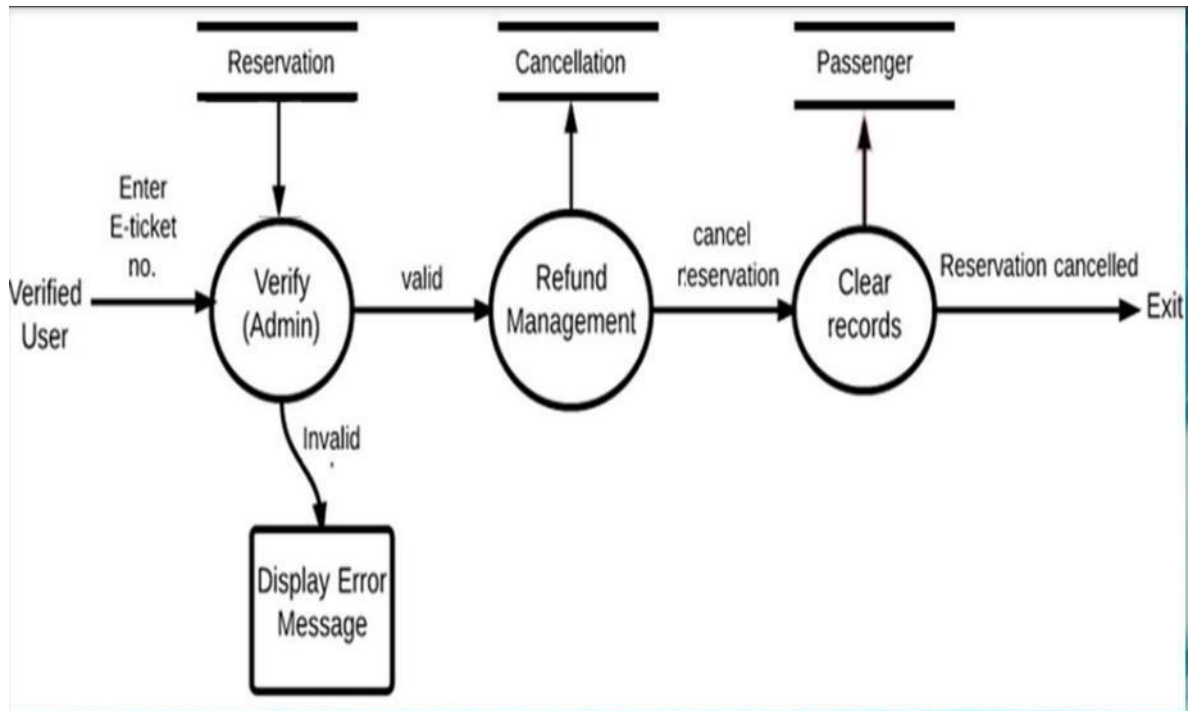


Fig 4.4 LEVEL- 3 Diagram

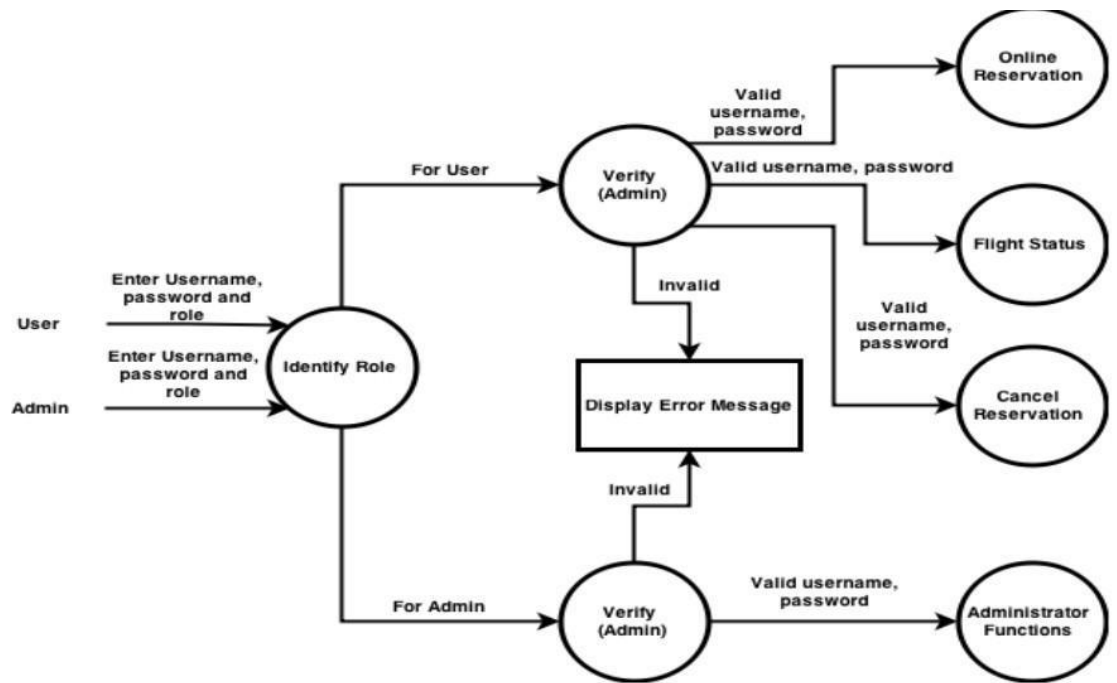


Fig 4.5 Level – 4 Diagram

4.9 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

Use case Diagram

There are three actors in our project first one is the airline data analyst, second is airline and the third one is the user. The role of the analyst is to connect to the airline and then create an API which give the access to the to extract the data from airline. After getting access from airline using API .we can extract the airline data. Afterwards we will put the data into a excel table and insert it into HDFS after which the analysis one the particular topic. The analyst will receive the output by which the client will use the particular data.

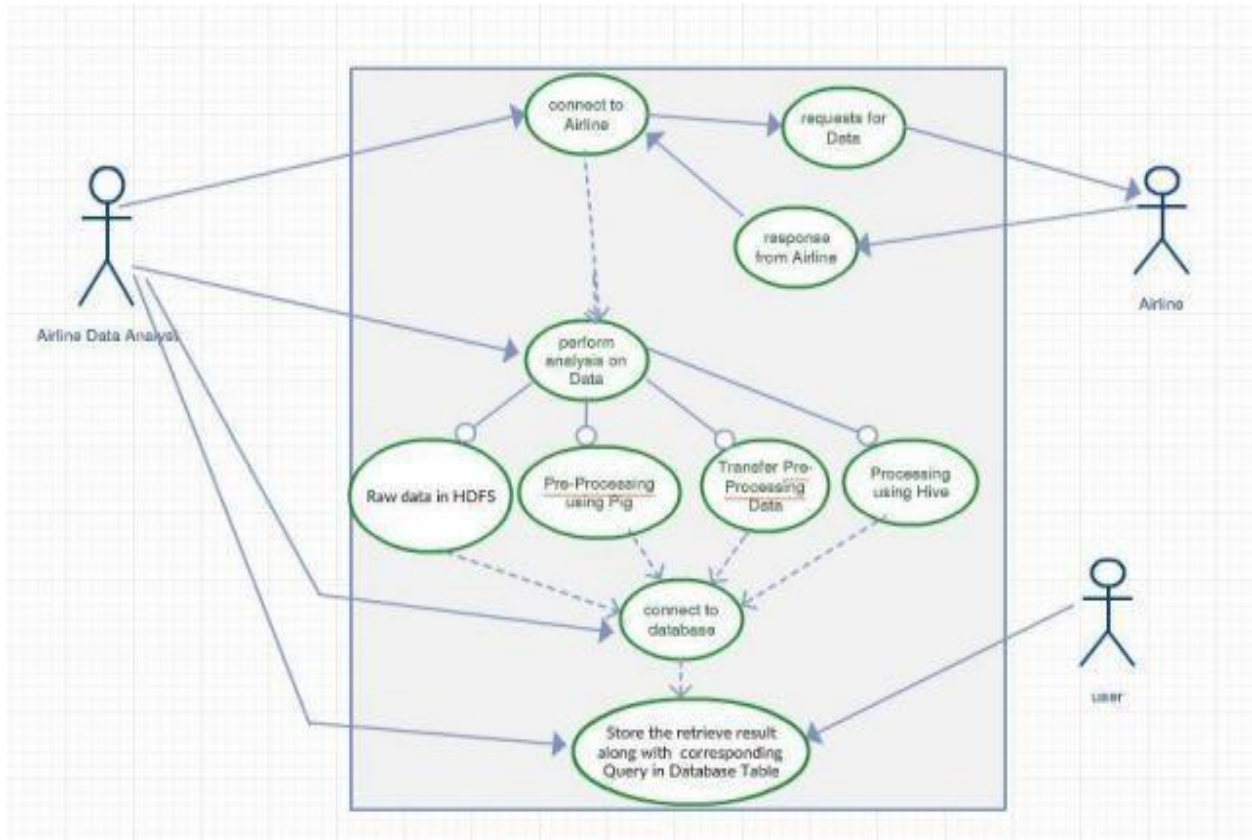


Fig 4.6 Use case Diagram

4.10 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. There are four objects in our project which are airline data analyst, system interface, airline and client. first of all the process starts by the analyst by creating an API for airline data . the analyst request for the excess of airline data and then the access grated by the airline. Now the role of airline is done here, after we will program the raw data into HDFS and then insert into the excel table than it will show the extract view of the data. Now we are ready to fire the command and then we get the particular data as our output and provide to the clients.

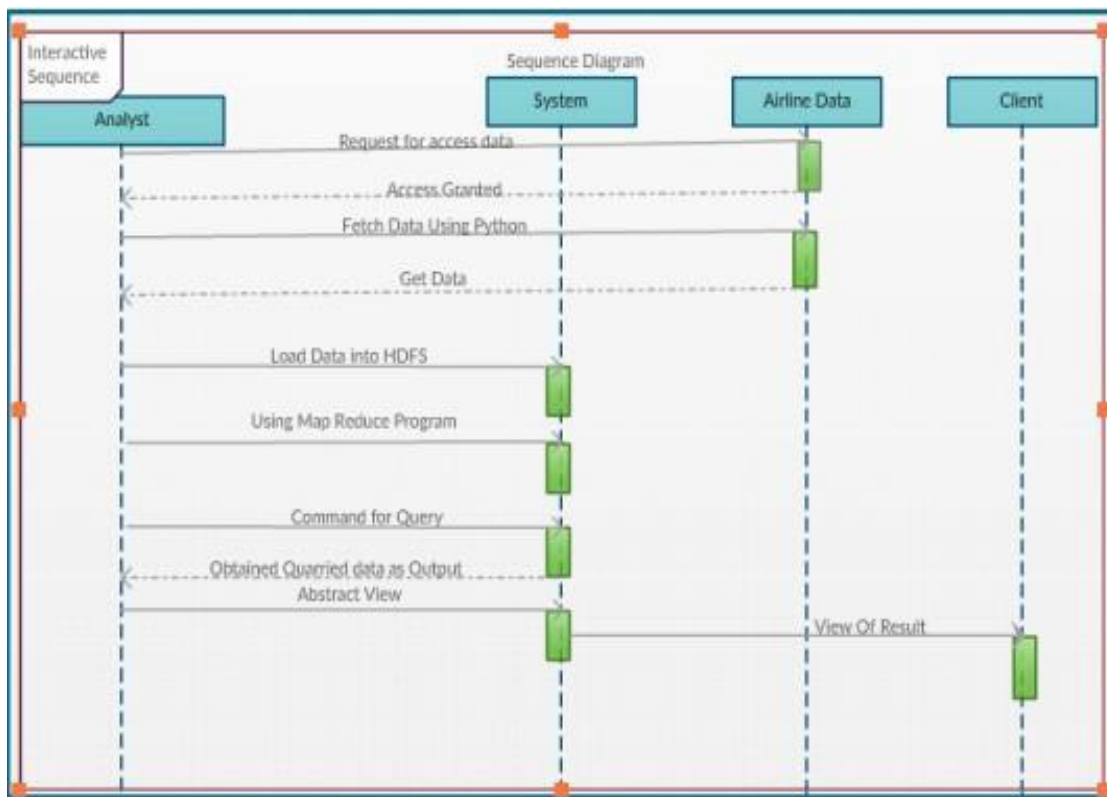


Fig 4.7 Sequence Diagram

4.11 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

4.11.1 TYPES OF TESTS:

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration

testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test.

System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing:

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot —see into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach:

Field testing will be performed manually and functional tests will be written in detail.

Test objectives:

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested:

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing:

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 5

RESULTS AND DISCUSSION

The data is extracted from kaggle website and is uploaded to the jupyter notebook.

Year	Month	Day of Month	Day of Week	Carrier	Origin Airport	Destination Airport	CRS Dep Time	Dep Delay	Dep Del15	CRS Arr Time	Arr Delay	Arr Del15	Cancelled
2013	4	19	5	DL	11433	13303	837	-3	0	1138	1	0	0
2013	4	19	5	DL	14869	12478	1705	0	0	2336	-8	0	0
2013	4	19	5	DL	14057	14869	600	-4	0	851	-15	0	0
2013	4	19	5	DL	15016	11433	1630	28	1	1903	24	1	0
2013	4	19	5	DL	11193	12892	1615	-6	0	1805	-11	0	0
2013	4	19	5	DL	10397	15016	1726	-1	0	1818	-19	0	0
2013	4	19	5	DL	15016	10397	1900	0	0	2133	-1	0	0
2013	4	19	5	DL	10397	14869	2145	15	1	2356	24	1	0
2013	4	19	5	DL	10397	10423	2157	33	1	2333	34	1	0
2013	4	19	5	DL	11278	10397	1900	323	1	2055	322	1	0
2013	4	19	5	DL	14107	13487	1540	-7	0	2043	-13	0	0
2013	4	19	5	DL	11433	11298	835	22	1	1035	41	1	0
2013	4	19	5	DL	11298	11433	1115	40	1	1450	20	1	0
2013	4	19	5	DL	11433	12892	1935	-2	0	2140	-7	0	0
2013	4	19	5	DL	10397	12451	1625	71	1	1738	75	1	0
2013	4	19	5	DL	12451	10397	1830	75	1	1955	57	1	0
2013	4	19	5	DL	12953	10397	1000	-1	0	1234	10	0	0
2013	4	19	5	DL	11433	12953	725	-3	0	918	-10	0	0
2013	4	19	5	DL	10397	14771	1725	31	1	1953	38	1	0
2013	4	19	5	DL	13204	10397	2030	8	0	2201	25	1	0
2013	4	19	5	DL	13487	12889	655	-3	0	827	-2	0	0
2013	4	19	5	DL	12889	13487	909	7	0	1415	16	1	0
2013	4	19	5	DL	10397	14027	1150	0	0	1337	19	1	0
2013	4	19	5	DL	14027	10397	1430	13	0	1623	25	1	0
2013	4	19	5	DL	13244	10397	1835	4	0	2054	13	0	0
2013	4	19	5	DL	12953	11697	1950	12	0	2311	-8	0	0
2013	4	19	5	DL	10397	12892	1345	15	1	1549	7	0	0
2013	4	19	5	DL	14869	14747	832	-7	0	940	-6	0	0
2013	4	19	5	DL	14747	14869	1020	15	1	1315	18	1	0
2013	4	19	5	DL	10397	10423	1100	0	0	1235	56	1	0
2013	4	19	5	DL	10433	10397	1335	56	1	1634	49	1	0

Fig 5.1 Dataset

The data is later passed on to different modules like Data profiling, Data wrangling, data cleaning & preparation, statistical data analysis and model prediction. The data set is further broken down and reduced for better analysis of data

Then this datasets are undergone through algorithms for analysis.

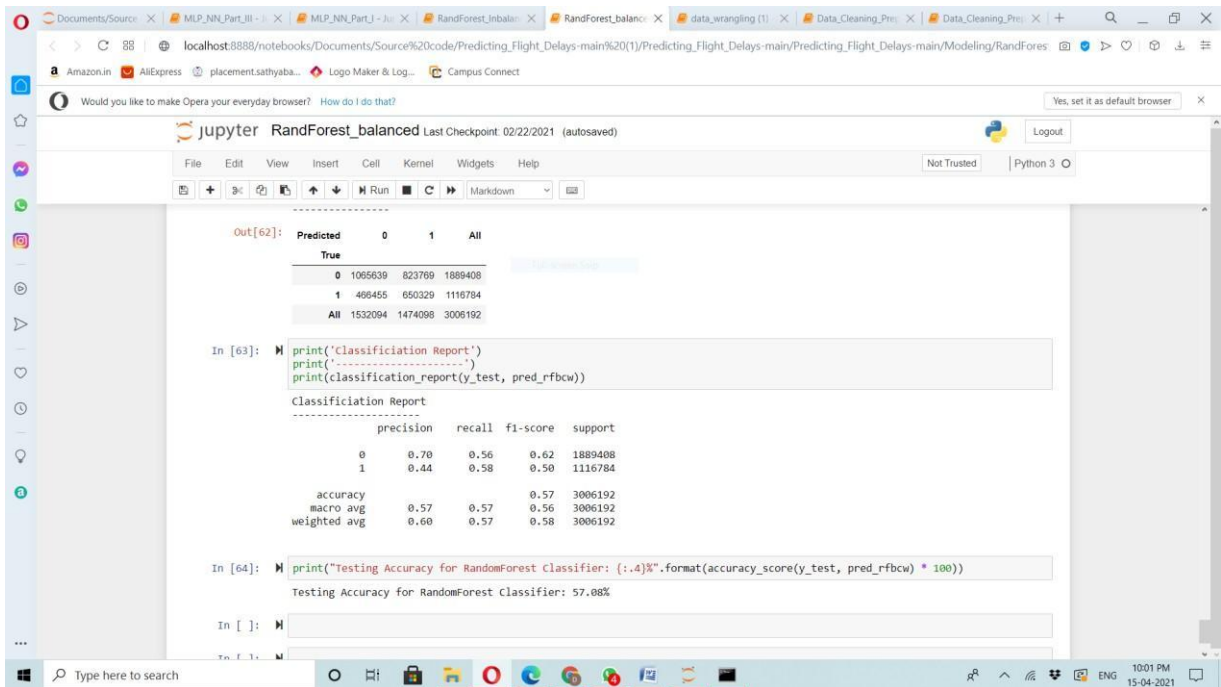


Fig 5.2 Balanced Dataset

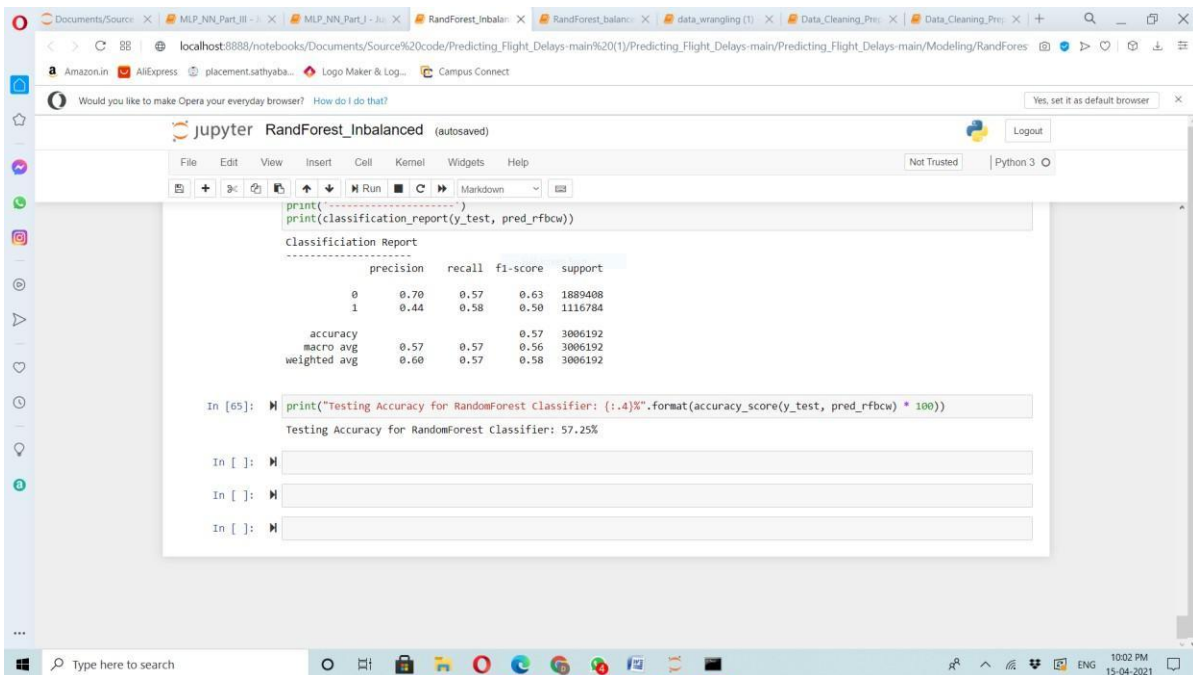
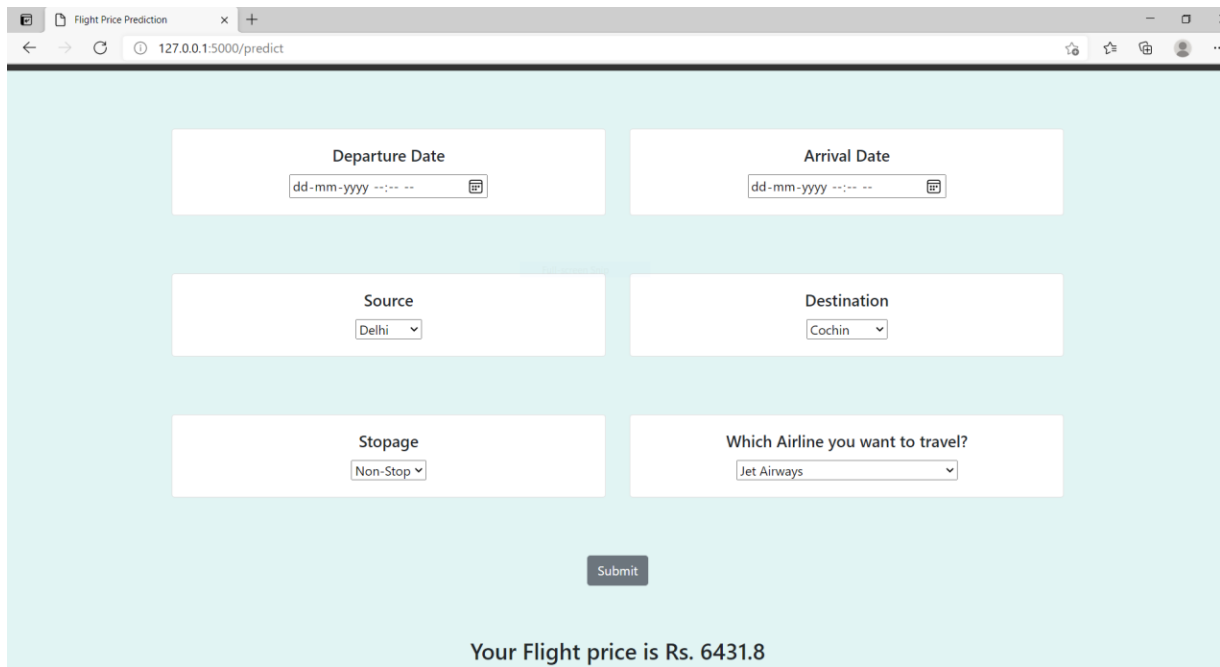


Fig 5.3 Imbalanced Dataset

Here we are using random forest algorithm to get the accuracy for both balanced and imbalanced dataset.

The accuracy for the balanced dataset is 57.08%

And the accuracy for the imbalanced dataset is 57.25%



The screenshot shows a web browser window with the title "Flight Price Prediction" and the URL "127.0.0.1:5000/predict". The interface is a light blue form with six input fields arranged in a 3x2 grid. The first row contains "Departure Date" and "Arrival Date", both with date pickers showing "dd-mm-yyyy --:-- --". The second row contains "Source" (a dropdown menu with "Delhi" selected) and "Destination" (a dropdown menu with "Cochin" selected). The third row contains "Stopage" (a dropdown menu with "Non-Stop" selected) and "Which Airline you want to travel?" (a dropdown menu with "Jet Airways" selected). Below the form is a "Submit" button. At the bottom of the page, the text "Your Flight price is Rs. 6431.8" is displayed.

Fig 5.4 Ticket Fare Prediction

Also the ticket fare is also predicted where in you have to fill in the basic details of your arrival and departure dates along with source and destination. Also select the airline you wish to travel and press submit. And thus you can predict the ticket fare.

```
In [0]: from pyspark.ml import Pipeline
from pyspark.ml.feature import StringIndexer
indexers = [StringIndexer(inputCol=column, outputCol=column+"_index").fit(delay_df) for column in categoricalColumns]
pipeline = Pipeline(stages=indexers)
delay2 = pipeline.fit(delay_df).transform(delay_df)
delay2.show(10)
```

date	flight_identifier	dep_delay	date_index	flight_identifier_index
2009	XE	-5	1.0	10.0
2009	XE	-4	1.0	10.0
2009	XE	-7	1.0	10.0
2009	XE	-5	1.0	10.0
2009	XE	25	1.0	10.0
2009	XE	-6	1.0	10.0
2009	XE	-1	1.0	10.0
2009	XE	-2	1.0	10.0
2009	XE	-4	1.0	10.0
2009	XE	-1	1.0	10.0

only showing top 10 rows

```
In [0]: #from pyspark.ml.feature import OneHotEncoder, StringIndexer
#indexers_ON = [OneHotEncoder(inputCol=column, outputCol=column+"_Vec") for column in filter(lambda x: x.endswith('_index'),
#pipeline = Pipeline(stages=indexers_ON)
#delay2 = pipeline.fit(delay2).transform(delay2)
#delay2.show(10)
```

Fig 5.5 Delay Time

```
In [0]: cancellation_df = spark.read.format('csv').load('s3://flightanalysis/cleaned-dataset/flight-cancellation/cancellation.csv',
cancellation_df.na.drop().show())
```

VBox()

FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', width='50%'),...

date	flight_number	flight_identifier	cancel	reason
2009	7104	YV	1	A
2009	7329	YV	1	A
2009	7065	YV	1	A
2009	2984	YV	1	B
2009	2823	YV	1	B
2009	7344	YV	1	A
2009	2798	YV	1	B
2009	2939	YV	1	B
2009	4537	00	1	A
2009	4537	00	1	A
2009	4785	00	1	B
2009	4794	00	1	B
2009	4810	00	1	B
2009	5424	00	1	A
2009	5466	00	1	B
2009	5510	00	1	B
2009	5544	00	1	A
2009	5555	00	1	B
2009	5557	00	1	B
2009	5568	00	1	B

only showing top 20 rows

```
In [0]: cancellation_df.filter(cancellation_df.reason == 'A').show()
```

Fig 5.6 Cancellation Factors

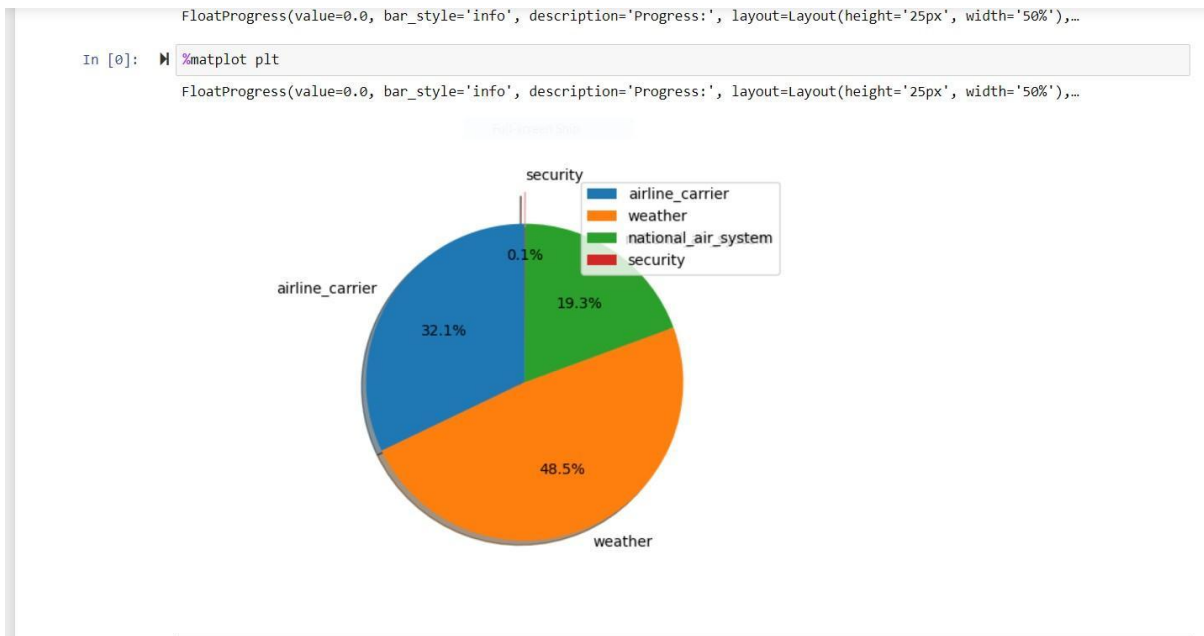


Fig 5.7 Pie-chart of Cancellation Factors

This is the result where the result of cancellation are predicted where A implies flight cancelled due to the airline carriers. Similarly B indicates cancellation due to the weather conditions, C due to the National Air System and finally D due to security reasons

```
In [11]: cancel_df.show(5)
```

label	features	date	flight_number	flight_identifier	cancel
0.0	(7566, [2, 560, 7555...]	2009	1212	XE	0
0.0	(7566, [2, 1076, 755...]	2009	1236	XE	0
0.0	(7566, [2, 1494, 755...]	2009	2002	XE	0
0.0	(7566, [2, 1827, 755...]	2009	2010	XE	0
0.0	(7566, [2, 1629, 755...]	2009	2029	XE	0

only showing top 5 rows

```
In [12]: train, test = cancel_df.randomSplit([0.7, 0.3], seed = 2018)
```

```
In [13]: from pyspark.ml.classification import LogisticRegression
lr = LogisticRegression(featuresCol = 'features', labelCol = 'label')
lrModel = lr.fit(train)
```

Fig 5.8 Cancellation Prediction

```
In [17]: > import matplotlib.pyplot as plt
import numpy as np

predictions = lrModel.transform(test)

predictions.select('date',
'flight_number',
'flight_identifier', 'prediction', 'probability').show(10)
```

date	flight_number	flight_identifier	prediction	probability
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...
2009	16	WN	0.0	[0.99108781155542...

only showing top 10 rows

```
In [18]: > from pyspark.ml.evaluation import BinaryClassificationEvaluator
evaluator = BinaryClassificationEvaluator()
print('Test Area Under ROC', evaluator.evaluate(predictions))

Test Area Under ROC 0.6379556663210726
```

Fig 5.9 Cancellation Probability

Finally the accuracy is compared, and the result shows the prediction of flight cancellation. If the flight is cancelled it indicates as '1' else it indicates with '0'

After comparing all the accuracies of the algorithms an average of 63% accuracy is obtained.

CHAPTER 6

CONCLUSION AND FUTURE WORK

The project is to identify if the airline cancels at a rate of 63% accuracy. GBT is the best thing to do in this case. Data is likely to be cut. . From the data, the most likely flight to be cancelled was ExpressJet Airlines and the least likely was Delta Airlines. However, airline identity was not a meaningful predictor of delay time. This suggests that delay of flight is likely to be for reasons other than the airline, such as weather conditions or airport. The dataset is brought down, neglecting the null values. To predict whether the flight would be cancelled or not, the dataset was modelled as a binary classification problem.

Later, machine learning models such as decision tree classifier and random forest were applied. All the model's accuracy were compared for predicting calculations. This project can be enhanced in future, where we can make its approach to a user more friendly. It can further be developed into a mobile application or a website and give access publicly.

REFERENCES

- [1] Bureau of Transportation Statistics. (2016). Airline On-Time Performance and Causes of Flight Delays. Retrieved from <https://catalog.data.gov/dataset/airline-on-time-performance-and-causes-of-flight-delays-on-time-data>
- [2] Deshpande, V., & Arikan, M. (2011). The Impact of Airline Flight Schedules on Flight Delays. *Manufacturing & Service Operations Management*, 14, 423-440. Retrieved from <https://pubsonline.informs.org/doi/10.1287/msom.1120.0379>
- [3] Mu, Y. (2019, August). Airline Delay and Cancellation Data, 2009 - 2018. Retrieved April 2020 from <https://www.kaggle.com/yuanyuwendymu/airline-delay-and-cancellation-data-2009-2018/data>
- [4] Chakrabarty, Navoneel, et al. ||Flight Arrival Delay Prediction Using Gradient Boosting Classifier.|| Emerging Technologies in Data Mining and Information Security. Springer, Singapore, 2019. 651-659. Retrieved from https://www.researchgate.net/publication/327389509_Flight_Arrival_Delay_Prediction_Using_Gradient_Boosting_Classifier
- [5] Yi Ding ||Predicting flight delay based on multiple linear regression||, IOP Conference Series: Earth and Environmental Science. Retrieved from <https://iopscience.iop.org/article/10.1088/1755-1315/81/1/012198>
- [6] Belcastro, L. &Marozzo, Fabrizio & Talia, Domenico & Trunfio, Paolo. (2016). Using Scalable Data Mining for Predicting Flight Delays. *ACM Transactions on Intelligent Systems and Technology*. 8. 10.1145/2888402. Retrieved from <https://dl.acm.org/doi/10.1145/2888402>
- [7] Li, S. —Machine Learning with PySpark and MLlib — Solving a Binary Classification Problem||. Towards Data Science. Retrieved from <https://towardsdatascience.com/machine-learning-with-pyspark-and-mllib-solving-a->

binary-classification-problem-96396065d2aa

APPENDICES

A.SOURCE CODE:

```
import findspark

findspark.init()

import pyspark

from pyspark.sql.session import SparkSession

sc = pyspark.SparkContext(appName="myAppName2").getOrCreate()

spark = SparkSession(sc)

from pyspark.sql.types import *

from pyspark.sql import functions as F

from pyspark.sql.functions import isnan, when, count, col

import pandas as pd

cancel_schema = StructType([StructField('date', StringType(), True),

StructField('flight_number', StringType(), True),

StructField('flight_identifier', StringType(), True),

StructField('cancel', StringType(), True),

StructField('reason', StringType(), True)])

cancel_df = spark.read.format('csv').option("mode",

"DROPMALFORMED").load('flight-cancellation.csv', header='true',

schema=cancel_schema)

cancel_df = cancel_df.drop('reason')

cancel_df = cancel_df.dropna()

cancel_df = cancel_df.sample(.1)
```

```
cancel_df.printSchema()

cancel_df.count()

cols = ['date', 'flight_number',
        'flight_identifier',
        'cancel']

import findspark findspark.init()


import pyspark

from pyspark.sql.session import SparkSession

sc = pyspark.SparkContext(appName="myAppName2").getOrCreate()

spark = SparkSession(sc)

from pyspark.sql.types import *

from pyspark.sql import functions as F

from pyspark.sql.functions import isnan, when, count, col

import pandas as pd

delay_schema = StructType([StructField('date', StringType(), True),
                             StructField('flight_identifier', StringType(), True),
                             StructField('plan_time', StringType(), True), StructField('actual_time',
                             StringType(), True), StructField('dep_delay', IntegerType(), True)])

delay_df = spark.read.format('csv').load('flight-delays.csv', header='true',
schema=delay_schema)
```

```

delay_df = delay_df.drop('actual_time')

delay_df = delay_df.drop('plan_time')

delay_df = delay_df.sample(.1) delay_df

= delay_df.dropna()

delay_df.printSchema()

delay_df.show(10)

from pyspark.ml.feature import OneHotEncoderEstimator, StringIndexer,
VectorAssembler

categoricalColumns = ["date", "flight_identifier"]

#indexers = [StringIndexer(inputCol=column, outputCol=column + "_index").fit(delay_df)
for column in ["date", "flight_number", "flight_identifier", "delay", "reason"]]

from pyspark.ml import Pipeline

from pyspark.ml.feature import StringIndexer

indexers = [StringIndexer(inputCol=column, outputCol=column+"_index").fit(delay_df) for
column in categoricalColumns]

pipeline = Pipeline(stages=indexers)

delay2 = pipeline.fit(delay_df).transform(delay_df)

delay2.show(10)

trainingSummary = lrModel.summary roc

= trainingSummary.roc.toPandas()

plt.plot(roc['FPR'],roc['TPR'])

plt.ylabel('False Positive Rate')

```

```
plt.xlabel('True Positive Rate')
```

```

plt.title('ROC Curve') plt.show()

print('Training set area under ROC: ' + str(trainingSummary.areaUnderROC))

from pyspark.ml.classification import RandomForestClassifier

rf = RandomForestClassifier(featuresCol = 'features', labelCol = 'label')

rfModel = rf.fit(train)

predictions = rfModel.transform(test)

predictions.select('date', 'flight_number',

'flight_identfier', 'prediction', 'probability').show(10)

cancellation_df.filter(cancellation_df.reason == 'A').show(2)

print("Cancellation due to Airline/Carrier ", cancellation_df.filter(cancellation_df.reason
== 'A').count())

cancellation_df.filter(cancellation_df.reason == 'B').show(2)

print("Cancellation due to Weather ", cancellation_df.filter(cancellation_df.reason
== 'B').count())

cancellation_df.filter(cancellation_df.reason == 'C').show(2)

print("Cancellation due to National Air System ",
cancellation_df.filter(cancellation_df.reason == 'C').count())

cancellation_df.filter(cancellation_df.reason == 'D').show(2)

print("Cancellation due to Security ", cancellation_df.filter(cancellation_df.reason
== 'D').count())

```