A Project Report

Improving Predictive Data Analytics in line with Airline Industry

Project Team:

1. Mr. Getinet Yilma(PI)
2. Mr. Dileep Kumar G.
3. Mr. Mohammed Kemal
4. Mr. Girma Debele
5. Dr. Vuda Sreenivasa Rao
6. Mr. Ejigu Tefera
7. Mr. Kifle Derese

አዳማ ሳይንስ እና ቴክኖሎጂ ዩኒቨርሲቲ
Adama Science & Technology University

School of Electrical Engineering & Computing
Computer Science & Engineering Program

Abstract

Air Transportation is one of the toughest and most dynamic industries in the world. The airline industry is the one where a few high value customers are more significant than many low value ones. Naturally, the winners are those who can do predictive analytics on the data and provide a personalized flying experience. There exists a tough competition among air travel companies especially on their fare pricing, Cargo price, passenger's time, safety, security and other related travel issues. Travel companies are applying multitude of techniques to retain customers to their carriers by using business intelligence, CRM, revenues management, financial analysis, and social media and so on. Big data analytics is the processing of complex data sets to discover hidden patterns, get useful correlations, extract market trends, and uncover customer preferences, product or services ratings and guiding principles of business information. Big data analytics can benefit airlines for effective marketing of services, better revenue, better customer service, improved operational efficiency, competitive advantages over rival organizations and other business benefits. Fare processing involves extracting and consolidating information from external sources such as Agent systems, External pricing systems, and internal pricing systems such as Revenue Accounting, Forecasting, Inventory and Yield Management systems. This data itself will run into tens of terabytes. Newer data sources such as Social Media conversations about pricing decisions & competitors, and un-structured data in enterprise like customer service e-mail data, call center logs etc. will push the data volumes even further. Analyzing the customer's data from various systems including Loyalty, CRM, Sales & Marketing etc. and data from partners systems along with data from social media based on customer social profile can help airlines to create deeper personalization preference of customers and also understand their current social status and preferences. The research study proposed Big Data Architecture leveraging Apache Hadoop and Apache Spark stack provides ability to extract, aggregate, load and process large volumes of data in a distributed manner which in turn reduce the complexity and overall turn-around time in processing large volumes of data. The built predictive model predict the fare price based on the historical big data after finding high value customers.

Keywords*: Airline, Analytics, SparkR, DataFrame, Machine Learning, Predictive Model.*

Acknowledgements

We would like to express deep sense of gratitude to Research and Technology Transfer Office, Adama Science and Technology University for funding this research project, and School of Electrical Engineering and Computing for their continuous support and encouragement by giving suggestions and lifting us wherever required thought the research project.

We truly thank and acknowledge Computer Science and Engineering program for dedicating a lab for Data Science, Special Interest Group.

Finally, we thank all who are involved internally and externally without whom the research project could not able to aim high.

Contents

List of Figures

List of Tables

Revision History

Version	Date	Remarks
1.0	10-APRIL-17	Initial Draft
1.1	15-APRIL-17	Revision
2.0	28-APRIL-17	Final Document

Glossary

Abbreviation	**Explanation**
HDFS	Hadoop Distributed File System
MR	Map Reduce
RAM	Random Access Memory
RDD	Resilient Distributed Dataset
VM	Virtual Machine
YARN	Yet Another Resource Negotiator
NoSQL	Not Only SQL
BI	Business Intelligence
GLM	Generalized Linear Model

Chapter 1

Introduction

1.1 Background and Motivation

Air Transportation is one of the toughest and most dynamic industries in the world. The Business constantly troubled by factors; like oil prices, thin profit margin, environmental concerns, employee anxieties, customers trust, travel delay and other problems. It is truly a world of survival of fittest business. Understanding the end consumer needs and discovering the competitive and attractive price to consumer is always a challenge. Airlines strive to achieve key differentiators like providing a personalized experience, enhance the brand, predictive intelligent systems for identifying profit and loss etc. In the airline industry, a few high-value customers can generate much more revenue than a number of low value customers, which highlights the importance of CRM systems [1, 2].

There exists a tough competition among air travel companies especially on their fare pricing, Cargo price, passenger's time, safety, security and other related travel issues. Travel companies are applying multitude of techniques to retain customers to their carriers by using business intelligence, CRM, revenues management, financial analysis, and social media and so on [2].

Big data analytics is the processing of complex data sets to discover hidden patterns, get useful correlations, extract market trends, and uncover customer preferences, product or services ratings and guiding principles of business information. Big data analytics can benefit companies for effective marketing of services, better revenue, better customer service, improved operational efficiency, competitive advantages over rival organizations and other business benefits [3].

Big data analytics is used to make informed business decisions by analyzing complex datasets that traditional business intelligence application couldn't handle. Big data source includes web server logs and Internet clickstream data, social media content and social media activity reports, customer emails and customer survey responses, phone call detail records and machine generated data captured by sensors connected to the Internet of Things. All the above data sources can be categorized Semi-structured and unstructured data. These data sources may not fit well in traditional databases storage. In addition, traditional Data processing tools may not be able to handle the processing demands of big datasets as huge volume of data frequently updated with high speed from heterogeneous sources. As a result, organizations shifted to collect, process and analyze big datasets using Hadoop and related tools such as YARN, Map Reduce, Spark, Hive, Pig and NoSQL databases. These technologies form the core of an open source software framework that supports the processing of large and diverse data sets on cheap commodity hardware across clustered systems [4]. Hadoop clusters and NoSQL handle task of capturing, cleaning, loading, and processing large dataset on demand/in real time using commodity hardware. Hadoop employs map reduce programming model for distributing (mapping) a task across the clusters of node

(reducing) [5].

By switching over to big data based solution, it is estimated that the processing times can be cut down from days to matter of hours or even minutesThomas13. The airline industry operates in high risk domain vulnerable to a large number of factors, and this agility will be a vital tool in maximizing revenues. Big data based solutions for personalization helps in understanding and deriving granular personalization parameters and understand customer's social status and needs based on their interaction in social media systems. The potential benefits for an airline could include:

- Increased revenue & practice
- Better fare management
- Efficient mechanisms to handle fare changes
- Faster decision making and reduced time-to-market for key fares
- Introduction of competitive fares and reactive fare response
- Increased efficiencies to airline in managing fare strategies for different scale channel
- Deeper Understanding of customers and personalization.

1.2 Problem Statement

Airlines classify fare into main classes of service(Economy, Business, First Class etc.), and each of these are subdivided into booking classes. The number of booking classes depends on factors like aircraft type, sector of the flight, fly date etc. The objective here is to have a greater level of control over the type of fares sold. When making a fare change, the fare re-priced, which means recalculating each fare, taking into consideration factors like:

- Flown data/Passenger booking information
- Forecasted data-load factor information
- Inventory
- Currency exchange rate at different point of sale.

Information about these factors will be distributed across various databases. Naturally, an attempt to process data would run into all the above mentioned problems. And these data sources may not be sufficient to discover a price aligned to customer's point of view.

Fare processing involves extracting and consolidating information from external sources such as Agent systems, External Pricing Systems, and Internal Systems such as Revenue Accounting, Forecasting, Inventory and Yield Management Systems. This data itself will run into tens of terabytes. Newer data sources such as Social Media conversations about pricing decisions and competitors, and unstructured data in enterprise like customer service e-mail data, call center logs etc. will push the data volume even further. In this context, the high-cost data-warehousing solutions are confronted with the following challenges.

- Expensive hardware and software: costs grows with data size
- Analytics process needs to be 100% customizable, and not governed by the confines of SQL.

- Analytics process should not be extended by the size of data; terabytes should be proceesed in minutes rather than days or hours.
- Data loss is unacceptable, the solution requires high availability and failover.
- Learning curve should not be steep.

Personalization and Fare processing involve large data sets. While the current system leverage conventional enterprise information, new data sources such as social media data, web logs, click streams data, call center logs, central statistical documents and competitor pricing would have to be considered for better personalization requirements and airline service pricing.

Smart businesses know that the real business value lies in retaining customers. The airline industry encompasses a huge amount and heterogeneous and complex datasets, and it is difficult for many airlines and airports to manage and process the amount of data they receive, but such data could be used to revolutionize the passenger experience. Here the question is - Is massive dataset collected and analyzed efficiently? As the Data-driven marketing can provide insights from data in real-time so there can be consistent understanding of airline service user's behavior, increase operational efficiencies, create competitive advantage, and efficient service marketing. The proposed study will find high value customers and suggests the fare pricing solutions focusing on customer retention solutions for airlines in particular.

1.3 Objectives of the Study

The main objective of the proposed study is to improve predictive analytics of airline industry

The specific objectives of the proposed study are:

- Review literature on existing predictive big data analytical methods
- Design a predictive model for airline industry
- Develop a prototype for finding high value customers and suggesting the fair pricing solutions using big data analytics tools in airline industry.
- Visualizing Analyzed Data
- Test and evaluate the performance of the proposed prototype.

1.4 Research Questions

The main research questions posed as part of this research are expressed as follows:

- How predictive analytics improves Airline's Business?
- Which factors influence fare?

1.5 Scope of the Problem

The predictive analytics has just been done on structured data.

1.6 Limitations of the Problem

The study proposes a Big Data Architecture to extract data from external sources such as Agent systems, External pricing systems and social media. But the researchers could not get such type of data; availing data in Ethiopia is very hard job. So, the study just collected the open data for analysis.

1.7 Organization of Document

The remainder of the document is organized as follows.

- The Chapter 2 presents a comprehensive description of related literature.
- Recent work with distributed computing when applying the Hadoop framework for scalability and availability is also referenced.
- In Chapter 3, there is a discussion on research methodology.
- Chapter 4 demonstrates the design and implementation of the proposed framework.
- Chapter 5 gives the results obtained and tests performed.
- Finally, Chapter 6 concludes with a summary of contributions and discuss future work.

Chapter 2

Literature Review

2.1 Big Data

The term 'Big Data' has recently been applied to datasets that grow so large that they become awkward to work with using traditional database management systems. They are data sets whose size is beyond the ability of commonly used software tools and storage systems to capture, store, manage, as well as process the data within a tolerable elapsed time [6]. Big data sizes are constantly increasing, currently ranging from a few dozen terabytes (TB) to many petabytes (PB) of data in a single data set. Consequently, some of the difficulties related to big data include capture, storage, search, sharing, analytics, and visualizing. Today, enterprises are exploring large volumes of highly detailed data so as to discover facts they didn't know before [3].

2.1.1 Characteristics of Big Data

Big data is data whose scale, distribution, diversity, and/or timeliness require the use of new technical architectures, analytics, and tools in order to enable insights that unlock new sources of business value. Three main features characterize big data: volume, variety, and velocity, or the three V's. The volume of the data is its size, and how enormous it is. Velocity refers to the rate with which data is changing, or how often it is created. Finally, variety includes the different formats and types of data, as well as the different kinds of uses and ways of analyzing the data [7].

Data volume is the primary attribute of big data. Big data can be quantified by size in TBs or PBs, as well as even the number of records, transactions, tables, or files. Additionally, one of the things that make big data really big is that it's coming from a greater variety of sources than ever before, including logs, clickstreams, and social media. Using these sources for analytics means that common structured data is now joined by unstructured data, such as text and human language, and semi-structured data, such as eXtensible Markup Language (XML) or Rich Site Summary (RSS) feeds. There is also data, which is hard to categorize since it comes from audio, video, and other devices. Furthermore, multi-dimensional data can be drawn from a data warehouse to add historic context to big data. Thus, with big data, variety is just as big as volume. Moreover, big data can be described by its velocity or speed. This is basically the frequency of data generation or the frequency of data delivery. The leading edge of big data is streaming data, which is collected in real-time from the websites [7]. Some researchers and organizations have discussed the addition of a fourth V, or veracity. Veracity focuses on the quality of the data. This characterizes big data quality as good, bad, or undefined due to data inconsistency, incompleteness, ambiguity, latency, deception, and approximations [8].

2.2 Big Data Analytics

Nowadays, people don't just want to collect data, they want to understand the meaning and importance of data, and use it to aid them in making decisions. Data Analytics is the process of applying algorithms in order to analyze sets of data and extract useful and unknown patterns, relationships, and information [9]. Furthermore, data analytics are used to extract previously unknown, useful, valid and hidden patterns and information from large data sets, as well as to detect important relationships among the stored variables. Therefore, analytics have had a significant impact on research and technologies, since decision makers have become more and more interested in learning from previous data, thus gaining competitive advantage [10].

2.3 Big Data Analytics Tools

2.3.1 Big Data Storage and Management

One of the first things organizations have to manage when dealing with big data, is where and how this data will be stored once it is acquired. The traditional methods of structured data storage and retrieval include relational databases, data marts, and data warehouses. The data is uploaded to the storage from operational data stores using Extract, Transform, Load (ETL), or Extract, Load, Transform (ELT), tools which extract the data from outside sources, transform the data to fit operational needs, and finally load the data into the database or data warehouse. Thus, the data is cleaned, transformed, and catalogued before being made available for data mining and online analytical functions [11].

However, the big data environment calls for Magnetic, Agile, Deep (MAD) analysis skills, which differ from the aspects of a traditional Enterprise Data Warehouse (EDW) environment. First of all, traditional EDW approaches discourage the incorporation of new data sources until they are cleansed and integrated. Due to the ubiquity of data nowadays, big data environments need to be magnetic, thus attracting all the data sources, regardless of the data quality [12]. Furthermore, given the growing numbers of data sources, as well as the sophistication of the data analyses, big data storage should allow analysts to easily produce and adapt data rapidly. This requires an agile database, whose logical and physical contents can adapt in sync with rapid data evolution [13]. Finally, since current data analyses use complex statistical methods, and analysts need to be able to study enormous datasets by drilling up and down, a big data repository also needs to be deep, and serve as a sophisticated algorithmic runtime engine [12].

Accordingly, several solutions, ranging from distributed systems and Massive Parallel Processing (MPP) databases for providing high query performance and platform scalability, to non-relational or in-memory databases, have been used for big data.

Non-relational databases, such as Not Only SQL (NoSQL), were developed for storing and managing unstructured, or non-relational, data. NoSQL databases aim for massive scaling, data model flexibility, and simplified application development and deployment. Contrary to relational databases, NoSQL databases separate data management and data storage. Such databases rather focus on the high-performance scalable data storage, and allow data management tasks to be written in the application layer instead of having it written in databases specific languages [11]. On the other hand, in-memory databases manage the data in server memory, thus eliminating disk input/output (I/O) and enabling real-time responses from the database. Instead of using mechanical disk drives, it is possible to store the primary database in silicon-based main memory. This results in orders of magnitude of improvement in the performance, and allows entirely new applications to be developed [14]. Furthermore, in-memory databases are now being used for advanced analytics on big data, especially to speed the access to and scoring of analytic models for analysis. This provides scalability for big data, and speed for discovery analytics [3].

Alternatively, Hadoop is a framework for performing big data analytics which provides reliability, scalability, and manageability by providing an implementation for the MapReduce paradigm, which is discussed in the following section, as well as gluing the storage and analytics together. Hadoop consists of two main components: the HDFS for the big data storage, and MapReduce for big data analytics [13]. The HDFS storage function provides a redundant and reliable distributed file system, which is optimized for large files, where a single file is split into blocks and distributed across cluster nodes. Additionally, the data is protected among the nodes by a replication mechanism, which ensures availability and reliability despite any node failures [11]. There are two types of HDFS nodes: the Data Nodes and the Name Nodes. Data is stored in replicated file blocks across the multiple Data Nodes, and the Name Node acts as a regulator between the client and the Data Node, directing the client to the particular Data Node which contains the requested data [11].

2.3.2 Big Data Analytic Processing

After the big data storage, comes the analytic processing. According to [14], there are four critical requirements for big data processing. The first requirement is fast data loading. Since the disk and network traffic interferes with the query executions during data loading, it is necessary to reduce the data loading time. The second requirement is fast query processing. In order to satisfy the requirements of heavy workloads and real-time requests, many queries are response-time critical. Thus, the data placement structure must be capable of retaining high query processing speeds as the amounts of queries rapidly increase. Additionally, the third requirement for big data processing is the highly efficient utilization of storage space. Since the rapid growth in user activities can demand scalable storage capacity and computing power, limited disk space necessitates that data storage be well managed during processing, and issues on how to store the data so that space utilization is maximized be addressed. Finally, the fourth requirement is the strong adaptivity to highly dynamic workload patterns. As big data sets are analyzed by different applications and users, for different purposes, and in various ways, the underlying system should be highly adaptive to unexpected dynamics in data processing, and not specific to certain workload patterns [15].

Map Reduce is a parallel programming model, inspired by the 'Map' and 'Reduce' of functional languages, which is suitable for big data processing. It is the core of Hadoop, and performs the data processing and analytics functions [16]. According to EMC, the MapReduce paradigm is based on adding more computers or resources, rather than increasing the power or storage capacity of a single computer; in other words, scaling out rather than scaling up [7]. The fundamental idea of MapReduce is breaking a task down into stages and executing the stages in parallel in order to reduce the time needed to complete the task [16].

The first phase of the MapReduce job is to map input values to a set of key/value pairs as output. The 'Map' function accordingly partitions large computational tasks into smaller tasks, and assigns them to the appropriate key/value pairs [16]. Thus, unstructured data, such as text, can be mapped to a structured key/value pair, where, for example, the key could be the word in the text and the value is the number of occurrences of the word. This output is then the input to the 'Reduce' function [7]. Reduce then performs the collection and combination of this output, by combining all values which share the same key value, to provide the final result of the computational task [16].

The MapReduce function within Hadoop depends on two different nodes: the Job Tracker and the Task Tracker nodes. The Job Tracker nodes are the ones which are responsible for distributing the mapper and reducer functions to the available Task Trackers, as well as monitoring the results [7]. The MapReduce job starts by the Job-Tracker assigning a portion of an input file on the HDFS to a map task, running on a node. On the other hand, the Task Tracker nodes

actually run the jobs and communicate results back to the Job Tracker. That communication between nodes is often through files and directories in HDFS, so inter-node communication is minimized [16].

Hadoop is a MAD system, thus making it popular for big data analytics by loading data files into distributed file system, running parallel MapReduce computations on the data. Hadoop gets its magnetism and agility from the fact that data is loaded into Hadoop simply by copying files into the distributed file system, and MapReduce interprets the data at processing time rather than loading time [13]. Thus, it is capable of attracting all data sources, as well as adapting its engines to any evolutions that may occur in such big data sources [16]. After big data is stored, managed and processed, decision makers need to extract useful insights by performing big data analysis.

2.4 Big Data Analytics Methods

Along with some of the most common advanced data analytics methods, such as association rules, clustering, classification and decision trees, and regression some additional analyses have become common with big data.

For example, social media has recently become important for social networking and content sharing. Yet, the content that is generated from social media websites is enormous and remains largely unexploited. However, social media analytics can be used to analyze such data and extract useful information and predictions [7]. Social media analytics is based on developing and evaluating informatics frameworks and tools in order to collect, monitor, summarize, analyze, as well as visualize social media data. Furthermore, social media analytics facilitates understanding the reactions and conversations between people in online communities, as well as extracting useful patterns and intelligence from their interactions, in addition to what they share on social media websites [17].

On the other hand, Social Network Analysis (SNA) focuses on the relationships among social entities, as well as the patterns and implications of such relationships [18]. An SNA maps and measures both formal and informal relationships in order to comprehend what facilitates the flow of knowledge between interacting parties, such as who knows who, and who shares what knowledge or information with who and using what [19]. However, SNA differs from social media analysis, in that SNA tries to capture the social relationships and patterns between networks of people. On the other hand, social media analysis aims to analyze what social media users are saying in order to uncover useful patterns, information about the users, and sentiments. This is traditionally done using text mining or sentiment analysis.

On the other hand, text mining is used to analyze a document or set of documents in order to understand the content within and the meaning of the information contained. Text mining has become very important nowadays since most of the information stored, not including audio, video, and images, consists of text. While data mining deals with structured data, text presents special characteristics which basically follow a non-relational form [20].

Moreover, sentiment analysis, or opinion mining, is becoming more and more important as online opinion data, such as blogs, product reviews, forums, and social data from social media sites like Twitter and Facebook, grow tremendously. Sentiment analysis focuses on analyzing and understanding emotions from subjective text patterns, and is enabled through text mining. It identifies opinions and attitudes of individuals towards certain topics, and is useful in classifying viewpoints as positive or negative. Sentiment analysis uses natural language processing and text analytics in order to identify and extract information by finding words that are indicative of a sentiment, as well as relationships between words, so that sentiments can be accurately identified [21].

Table 2.5.1: Classification of Analytic Models

Predictive Models	Descriptive Models	Decision Models
Finds causality,relationships and, patterns between, explanatory variables, and dependent,variables. Focus is on, specific variables.	Finds clusters of data, elements with, similar characteristics. Focus is on as many,variables as possible.	Find optimal and most,certain outcome for a, specific decision. Focus is on a specific decision.
Examples: customer preference, fraud, credit,worthiness, system, failure	Examples: customer, segmentation based, on socio-demographic, characteristics, life,cycle, profitability,„product preferences	Examples: critical path, network planning, scheduling, resource, optimization,„simulation, stochastic,modeling

Finally, from the strongest potential growths among big data analytics options is Advanced Data Visualization (ADV) and visual discovery [22]. Presenting information so that people can consume it effectively is a key challenge that needs to be met, in order for decision makers to be able to properly analyze data in a way to lead to concrete actions [23].

ADV has emerged as a powerful technique to discover knowledge from data. ADV combines data analysis methods with interactive visualization to enable comprehensive data exploration. It is a data driven exploratory approach that fits well in situations where analysts have little knowledge about the data [24]. With the generation of more and more data of high volume and complexity, an increasing demand has arisen for ADV solutions from many application domains [18]. Additionally, such visualization analyses take advantage of human perceptual and reasoning abilities, which enables them to thoroughly analyze data at both the overview and the detailed levels. Along with the size and complexity of big data, intuitive visual representation and interaction is needed to facilitate the analyst's perception and reasoning [24].

ADV can enable faster analysis, better decision making, and more effective presentation and comprehension of results by providing interactive statistical graphics and a point-and-click interface. Furthermore, ADV is a natural fit for big data since it can scale its visualizations to represent thousands or millions of data points, unlike standard pie, bar, and line charts. Moreover, it can handle diverse data types, as well as present analytic data structures that aren't easily flattened onto a computer screen, such as hierarchies and neural nets. Additionally, most ADV tools and functions can support interfaces to all the leading data sources, thus enabling business analysts to explore data widely across a variety of sources in search of the right analytics dataset, usually in real-time [25].

2.5 Predictive Analytics

Predictive analytics is the branch of data mining concerned with the prediction of future probabilities and trends [26] and predict the risk, segmentation, propensity and associations whereas optimization handles tradeoffs and constraints [27]. With predictive analytics, organizations can achieve higher revenues and growth by improving key metrics. In predictive analytics, to deal with different parameters at different stages of business framework, varieties of models are to be used.

In predictive modeling, data is collected, a statistical model is formulated, predictions are made, and the model is validated (or revised) as additional data become available from other sources. The methodology followed here is shown in figure 2.5.1.

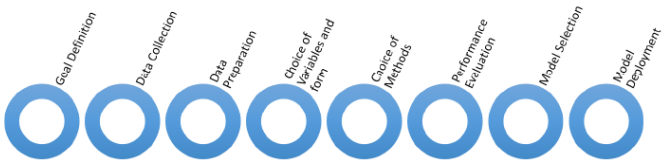

Figure 2.5.1: Steps in Data Modelling Process

Figure 2.5.2: Predictive Analytics Process

Building predictive models is an iterative process in which a model is created from an initial hypothesis and then refined until it produces a valuable business outcome or discarded in favor of another model with more potential [28].

Predictive modeling is based on one or more data instances for which we want to predict the value of a target variable. Data-driven predictive modeling generally induces a model from training data, for which the value of the target (the label) is known. These instances typically are described by a vector of features from which the predictions will be made.

Naive Bayes classifier can be used to learn models and to make predictions. Naive Bayes has been shown to have surprisingly good performance on sparse datasets. A very attractive property of 'Naive' Bayes is that it is extremely fast to run on a large, sparse dataset when it is formulated well. The main speedup stems from the fact that 'Naive' Bayes completely ignores interfeature dependencies by assuming that the within-class covariances of the features are zero. Combining this 'Naive' assumption with Baye's rule for conditional probability produces the following formulation:

There are a several variants of Naive Bayes in common use, which differ in their calculation of the conditional probabilities. Each of these variants assumes a different event model, which postulates how the features from the data were generated. The two most popular choices are the multivariate and the multinomial event models. The main idea behind the multinomial event model is that each input sample results from a series of independent draws from the collection of all features. The main advantage of this model is that it requires only computing over those features that have a count that is nonzero. For a sparse dataset, this results in a huge reduction in run time since all the zeros can be ignored. The multinomial model makes the generative assumption that the features are drawn with replacement, which is not well specified for most binary feature data (although the multinomial model is used commonly in practice for binary

data). The multinomial model further makes the implicit assumption that the number of draws in the generative model is independent of the class, which roughly implies in our setting that each instance contains on average the same number of active features (behaviors) per class, which might not be true for the datasets that we are considering. Therefore, it is helpful to consider the multivariate model as well, even though the traditional formulation is not efficient for massive datasets.

2.6 Related Work

[29] present a structural framework for deriving value from business analytics. Extracting value from data requires aligning strategy and desirable behaviors to business performance management in conjunction with analytic tasks and capabilities. The authors then introduced three special articles that provide in-depth insights regarding how business analytics is being employed in the management of healthcare, accounting, and supply chains. Business analytics is a revolution that is impossible to miss. At its core, business analytics is about leveraging value from data. Instead of being referred to as the 'sludge of the information age' data has recently been deemed 'the new oil'. While data can be employed for purposes such as detecting new opportunities, identifying market niches, and developing new products and services, it is also notoriously amorphous and hard to extract value from.

[30] describe a novel analytics system that enables query processing and predictive analytics over streams of big aviation data. As part of an Internal Research and Development project, Boeing Research and Technology (BRT) Advanced Air Traffic Management (AATM) built a system that makes predictions based upon descriptive patterns of massive aviation data. Boeing AATM has been receiving live Aircraft Situation Display to Industry (ASDI) data and archiving it for over two years. At the present time, there is not an easy mechanism to perform analytics on the data. The incoming ASDI data is large, compressed, and requires correlation with other flight data before it can be analyzed. The service exposes this data once it has been uncompressed, correlated, and stored in a data warehouse for further analysis using a variety of descriptive, predictive, and possibly prescriptive analytics tools. The service is being built partially in response to requests from Boeing Commercial Aviation (BCA) for analysis of capacity and flow in the US National Airspace System (NAS). The service utilizes a custom tool developed by Embry Riddle Aeronautical University (ERAU) that correlates the raw ASDI feed, IBM Warehouse with DB2 for data management, WebSphere Message Broker for real-time message brokering, SPSS Modeler for statistical analysis, and Cognos BI for front-end business intelligence (BI) visualization tools. This paper describes a scalable service architecture, implementation and value it adds to the aviation domain.

Tulinda [31] identifies key aviation data sets for operational analytics, presents a methodology for application of big-data analysis methods to operational problems, and offers examples of analytical solutions using an integrated aviation data warehouse. Big-data analysis methods have revolutionized how both government and commercial researchers can analyze massive aviation databases that were previously too cumbersome, inconsistent or irregular to drive high-quality output. Traditional data-mining methods are effective on uniform data sets such as flight tracking data or weather. Integrating heterogeneous data sets introduces complexity in data standardization, normalization, and scalability. The variability of underlying data warehouse can be leveraged using virtualized cloud infrastructure for scalability to identify trends and create actionable information. The applications for big-data analysis in airspace system performance and safety optimization have high potential because of the availability and diversity of airspace related data. Analytical applications to quantitatively review airspace performance, operational efficiency and aviation safety require a broad data set. Individual information sets such as radar tracking data or weather reports provide slices of relevant data, but do not provide the required

context, perspective and detail on their own to create actionable knowledge. These data sets are published by diverse sources and do not have the standardization, uniformity or defect controls required for simple integration and analysis. At a minimum, aviation big-data research requires the fusion of airline, aircraft, flight, radar, crew, and weather data in a uniform taxonomy, organized so that queries can be automated by flight, by fleet, or across the airspace system. With the increasingly widespread collection and processing of 'big data' there is natural interest in using these data assets to improve decision making. One of the best understood ways to use data to improve decision making is via predictive analytics. An important, open question is: to what extent do larger data actually lead to better predictive models? In this article author empirically demonstrate that when predictive models are built from sparse, fine-grained data-such as data on low-level human behavior; continue to see marginal increases in predictive performance even to very large scale. The empirical results are based on data drawn from nine different predictive modeling applications, from book reviews to banking transactions [32].

The study [33] provides a clear illustration that larger data indeed can be more valuable assets for predictive analytics. This implies that institutions with larger data assets-plus the skill to take advantage of them-potentially can obtain substantial competitive advantage over institutions without such access or skill. Moreover, the results suggest that it is worthwhile for companies with access to such fine-grained data, in the context of a key predictive task, to gather both more data instances and more possible data features. As an additional contribution, the authors introduced an implementation of the multivariate Berznoulli Naive Bayes algorithm that can scale to massive, sparse data.

Previous works have been done regarding airfare prediction using machine learning. Etzioni et al. have performed a pilot study on 12000 price observations over a 41-day period. Their multistrategy data mining algorithm - Hamlet generated a predictive model that could potentially save substantial amount of money which consumers pay on airline tickets [34]. Groves et al. proposed a model to predict expected minimum price of all flights on a particular route. The model was also used to predict price with different target properties such as prediction from a specific flight, non-stop only flight and etc [35]. Rama-Murthy built a model to predict the airfare price with specific focus on how different factors influence the price of airline tickets [36]. Papadakis studied how prices of airline tickets change overtime by extracting several factors that potentially affect the price fluctuation and finding out their correlation [37].

Ruixuan et al. built up models to predict the airline ticket price. The input to models are the factors that may influence the price, such as the weekday of departure and the number of stops in the itinerary and applied linear regression, Naive Bayes, Softmax regression, and Support Vector Machine (SVM) to predict the corresponding price. This study concluded that one possible way to increase the accuracy can be combining different models after carefully studying their own performance on each individual bin and adding more features will increase the accuracy of our models [38].

Chapter 3

Methodology

3.1 Data Source

The data has been collected from wide range of sources. Likewise, there are a variety of techniques applied when we gathered the primary and secondary data. Listed below are some of the most common data collection techniques we used for collecting data: Interviews, Questionnaires, Observations and Documents analysis.

In order to experiment this research, collecting and structuring the data is used for analysis which is one of the tasks that needs close attention in the process of analyzing the data. In order to collect necessary research data, we used the following data collection methods:

3.1.1 Primary data collection

- Interview: interviews are used with the selected interviewer from the organization to know their attitude toward the proposed system. Therefore, we make an interview with the Ethiopian Airlines and some private booking agents.
- Observation: we conducted a strict observation in all the manual activities of the airlines.

3.1.2 Secondary data collection

To have a detailed knowledge and understanding of the current system, we referred previous work related to the proposed system that could help us as an input and facilitate the development of the new system. Different source books, Internet sources, Conferences, magazines etc. ere analyzed and technical knowledge are reviewed. During document analysis, we gave a special consideration to those documents which can bring more features to our system.

3.1.2.1 U.S. Domestic Airline Market

The United States of America (U.S.) has a highly advanced air transportation infrastructure. There are roughly 19,000 airports, more than 70 percent of which have paved runways, and 376 with regularly scheduled airline service in terms of passengers [39]. Seventeen of the world's thirty busiest airports in 2016 were in the U.S., including the world's busiest, Hartsfield-Jackson Atlanta International . Due to the large distances between major cities in the U.S, air transportation is the preferred method of travel for trips over 300 miles. For cities that are closer together, such as Boston and New York City, or Philadelphia and Washington D.C., air travel accounts for a minority of intercity traffic. Within the U.S. domestic market there are around 2,780 airport-pairs connected with scheduled services. Around 2,140 routes are served by just a single

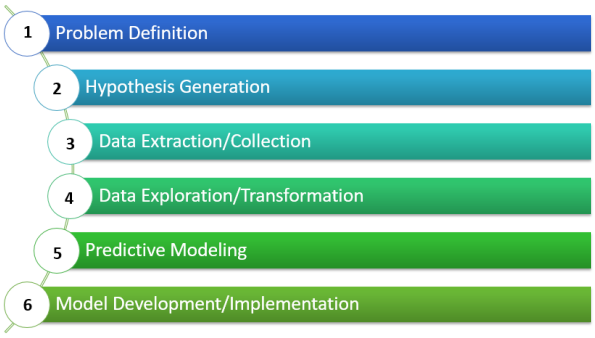

Figure 3.2.1: Process Model

carrier, 470 routes have two carriers with strong competition, 139 have three and 24 have four airlines competing for customers. Only 5 routes remain with five or more airlines competing for market share [40].

3.1.2.2 Origin and Destination Survey (DB1B)

The Airline Origin and Destination Survey (DB1B) is a 10% random sample of airline passenger tickets, which is freely available from the Bureau of Transportation Statistics. It consists of three tables in three independent files: Ticket, Coupon, and Market.

The data has been grouped by itinerary number (Itinerary ID) and each of these files provides different information about the itinerary. Each itinerary can be considered as a collection of passengers making trips with the same characteristics, such as origin and destination, fare or number of flight legs with the same airline. Those itineraries are reported in the way the customers bought the ticket, and each leg of the trip will appear as a different coupon.

DB1B Ticket: This file contains characteristics such as reporting carrier, itinerary fare, number of passengers, originating airport, a roundtrip indicator, and miles flown for each itinerary.

DB1B Coupon: This file contains the itineraries broken down into coupons, and provides information for each coupon within each itinerary, such as the operating carrier, origin and destination airports, number of passengers, fare class, coupon type, trip break indicator, and distance.

DB1B Market: This database contains directional market characteristics, such as the reporting carrier, origin and destination airport, market fare, number of market coupons, market miles flown, and carrier change indicators.

3.2 Process Model

Predictive Analytics includes techniques that use models created from past data to predict the future or determine the impact of one variable on another. Model building is a well-defined process for building predictive models. Our model building life cycle can be divided in 6 stages as shown in figure 3.2.1:

3.3 Data Preprocessing

We considered two tables; DB1B Ticket and DB1B Coupon to fulfill our research objectives.

3.3.1 Parsing Ticket

The ticket data is available at the Bureau of Transportation Statistics website. Data is provided in tables in the DB1B database: DB1BCoupon and DB1BTicket. We reduce the Ticket table to two columns only; Itinerary ID and its corresponding fare. The other information can be found in Coupon.

3.3.2 Parsing and filtering Coupon

The collected data represents a 10% random sample of airline passenger itineraries for the year 2016. Each year is divided into 4 quarters and each DB1BCoupon quarter consists of 36 variables, and approximately 54 million lines. The total size of the raw database that we treated is 9.35 GB.

To filter the data, we used the SparkR programming language. In particular, we used RStudio IDE for writing programming code, which offers a comfortable interface for better work with the code.

The database filter was designed to keep all those itineraries that we considered could contribute crucial information for a correct modeling of fares.

Firstly, the DB1B Coupon is filtered by applying certains restrictions so that the remaining itineraries satisfy :

- The 14 main airlines in U.S market were included: Southwest, American, Continental, Delta, Northwest, Skywest, United, US Airways, American Eagle, AirTran Airways, Express Jet, Jetblue, Alaska Airlines, Endeavor Air.
- Only itineraries within the 48 states of the continental U.S.
- No code-sharing. So the ticketing carrier and operating carrier must be the same on all legs.
- Direct route (2 coupons) or indirect routes restricted to just 4 coupons to include hub effects.
- Restrict the number of breaks to 2 i.e. a clearly-defined origin and destination.
- Roundtrips only i.e those itineraries with return to the origin airport.
- The fareclass will be limited to economy class (restricted and unrestricted).

After this step, the coupons belonging to the same itinerary are merged into just one line of data that will contain the main information of each trip.

The DB1BTicket also has to be filtered in order to avoid redundancy with the Coupon file. So the only information that has been considered important for this study and has been kept after the filter is the fare.

After this the DB1BCoupon is merged with DB1BTicket using the itinerary ID as coupling element. During this step another filter is applied based on fare levels to exclude unusual prices i.e we retain those fares that are higher than 150 USD, to avoid frequent flyer tickets (that only pay airport fees), and lower than 10,000 USD (not inflation adjusted). This last number was considered to be an extreme fare for an economy roundtrip within the U.S.

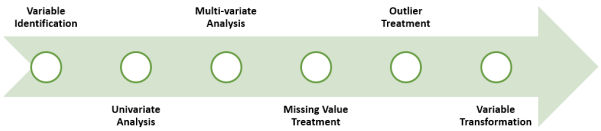

Figure 3.4.1: Data Exploration Process

3.4 Data Exploration

The data exploration stage helps to understand and explore the explicit and hidden trends in data, data cleaning and data transformation.

3.4.1 Univariate Analysis

At this stage, variables are explored one by one. Method to perform uni-variate analysis will depend on whether the variable type is categorical or continuous.

Continuous Variables: In case of continuous variables, we need to understand the central tendency and spread of the variable.

Categorical Variables: For categorical variables, we will use frequency table to understand distribution of each category. We can also read as percentage of values under each category. It can be measured using two metrics, 'Count' and 'Count%' against each category. Bar chart can be used as visualization.

3.4.2 Bi-variate Analysis

Bi-variate Analysis finds out the relationship between two variables. We can perform bi-variate analysis for any combination of categorical and continuous variables. The combination can be: Categorical & Categorical, Categorical & Continuous and Continuous & Continuous. Different methods are used to tackle these combinations during analysis process.

3.4.3 Missing Value Treatment

Missing data in the training data set can reduce the power fit of a model or can lead to a biased model because we have not analyzed the behavior and relationship with other variables correctly. It can lead to wrong prediction or classification. The following are the methods to treat missing values:

- Deletion
- Mean/Mode/Median Imputation
- Prediction model
- KNN Imputation

3.4.4 Outlier Treatment

Outlier is a commonly used terminology by analysts and data scientists as it needs close attention else it can result in wildly wrong estimations. Simply speaking, outlier is an observation that appears far away and diverges from an overall pattern in a sample. Outliers can drastically change the results of the data analysis and statistical modeling. There are numerous unfavorable impacts of outliers in the data set:

- It increases the error variance and reduces the power of statistical tests
- If the outliers are non-randomly distributed, they can decrease normality
- They can bias or influence estimates that may be of substantive interest
- They can also impact the basic assumption of Regression, ANOVA and other statistical model assumptions.

Most commonly used method to detect outliers is visualization. We use various visualization methods, like Box-plot, Histogram, Scatter Plot.

Most of the ways to deal with outliers are similar to the methods of missing values like deleting observations, transforming them, binning them, treat them as a separate group, imputing values and other statistical methods.

3.4.5 Feature Engineering

Feature engineering is the science (and art) of extracting more information from existing data. Feature engineering itself can be divided in 2 steps:

- Variable transformation.
- Variable / Feature creation.

These two techniques are vital in data exploration and have a remarkable impact on the power of prediction.

In data modeling, transformation refers to the replacement of a variable by a function. For instance, replacing a variable x by the square / cube root or logarithm x is a transformation. In other words, transformation is a process that changes the distribution or relationship of a variable with others. There are various methods used to transform variables. Some of them include square root, cube root, logarithmic, binning, reciprocal and many others.

Feature / Variable creation is a process to generate a new variables / features based on existing variable(s). For example, say, we have date(dd-mm-yy) as an input variable in a data set. We can generate new variables like day, month, year, week, weekday that may have better relationship with target variable. This step is used to highlight the hidden relationship in a variable. The following are few techniques to create new features.

- Creating derived variables
- Creating dummy variables

3.5 Model Building

The steps of predictive modeling:

Predictive Analytics employs supervised learning. Common Algorithms for regression and classification are:

- Linear Regression
- Logistics Regression

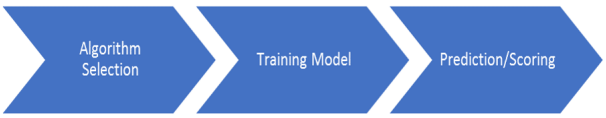

Figure 3.5.1: Model Building

- Decision Tree
- Random Forest
- KNN
- Bagging and Boosting etc.

Prediction or scoring is a process to estimate/predict dependent variable of test data set by applying model rules.

3.6 Model Deployment

Model deployment is a process to serve model estimation to users.

Chapter 4

Design and Implementation

4.1 Personalization and Smart Pricing

The airline industry is the one where a few high value customers are more significant than many low value ones. Naturally, the winners are those who can do predictive analytics on the data and provide a personalized flying experience. A wealth of customer behavioral data can be gleaned from websites - like routes and class preferences, frequency of flying, baggage, dining information, geo-location etc. to name a few. This type of analytics requires accumulation of data from multiple sources, frequent processing of high volume data, flexibility and agility in the processing logic. These are plain areas for traditional data warehouse solutions and these get compounded as the data volume grows.

Fare processing involves extracting and consolidating information from external sources such as Agent systems, External pricing systems, and internal pricing systems such as Revenue Accounting, Forecasting, Inventory and Yield Management systems. This data itself will run into tens of terabytes. Newer data sources such as Social Media conversations about pricing decisions & competitors, and un-structured data in enterprise like customer service e-mail data, call center logs etc. will push the data volumes even further. In this context, the high-cost data warehousing solutions are not feasible. The research study proposes Big Data Architecture, shown in figure 4.1.1 to confront above contexts and challenges.

Analyzing the customer's data from various systems including Loyalty, CRM, Sales & Marketing etc. and data from partners systems along with along with data from social media based on customer social profile can help airlines to create deeper personalization preference of customers and also understand their current social status and preferences.

The proposed Big Data Architecture leveraging Apache Hadoop and Apache Spark stack provides ability to extract, aggregate, load and process large volumes of data in a distributed manner which in turn reduce the complexity and overall turn-around time in processing large volumes of data.

The proposed solution is aimed at primarily addressing the following challenges:

- Ability to manage and pre-process large voles of enterprise data, partners data, and logs data and transform these data into meaningful data for analytics(Analysis, Preprocessing for Fare calculation and personalization services)
- Run analytics on unstructured data from social and other sources to derive newer dimension such as sentiment, root causes from customer interactions and user created contents.

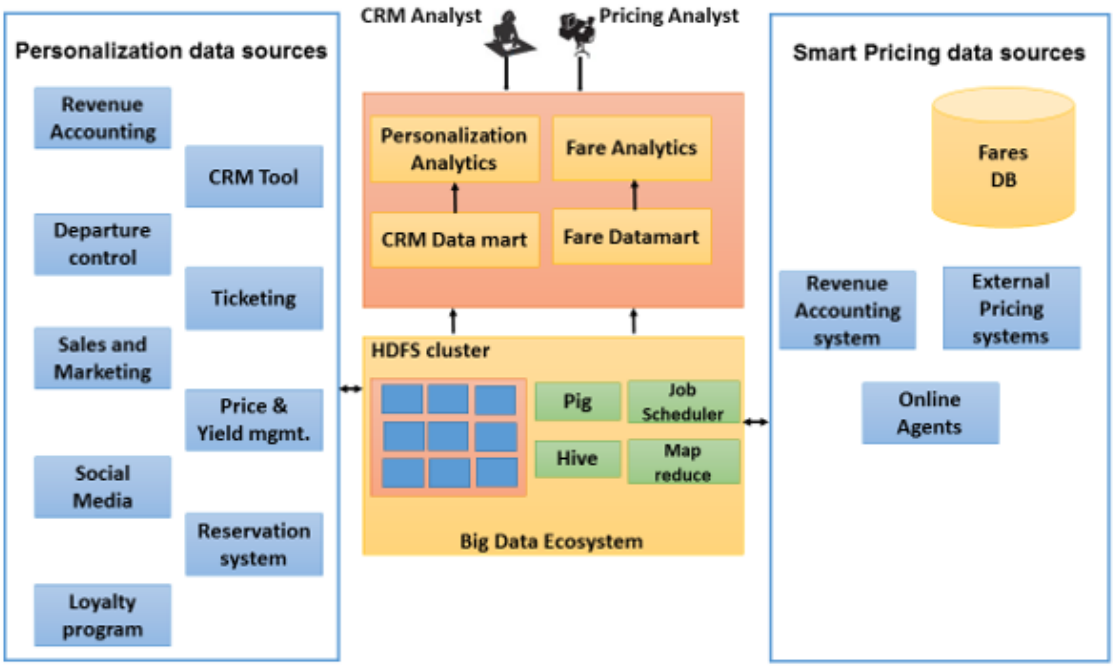

Figure 4.1.1: Proposed Architecture: Personalization and Smart Pricing

Data extracts from various feeder systems are copied to the Big Data file systems(HDFS). The Analytics platform will trigger the processing of this data, while responding to information needs for various analytics scenarios. The Analytics platform contains the machine learning algorithms tailored to the business process, to extract or mine insights out of the raw data.

4.2 Smart Pricing Preferences - Choice Game

The study suggests the following strategies that enables different airline passengers to choose between different competitive schemes based on attributes in figure 4.2.1.

4.3 Apache Spark Cluster in DSSIG

4.3.1 Apache Spark

Apache Spark [41] is a fast cluster computing framework which is used for processing, querying and analyzing Big data. It is based on In-memory computation, which is a big advantage of Apache Spark over several other big data Frameworks. Apache Spark is open source and one of the most famous Big data framework. It can run tasks up to 100 times faster, when it utilizes the in-memory computations and 10 times faster when it uses disk than traditional map-reduce tasks.

Apache Spark was originally created at University of California, Berkeley's AMPLab in 2009. The Spark code base was later donated to the Apache Software Foundation. Subsequently, it was open sourced in 2010. Spark is mostly written in Scala language. It has some code written in Java, Python and R. Apache Spark provides several APIs for programmers which include Java, Scala, R and Python.

Variable	Levels	Definition
Price	1	P + 20% added trip scheduled within 3-5 days from booking
	2	P normal price
	3	P – 20% discounted trip scheduled before 15 days
Ticketing schedule	Single Trip Round Trip	P P-10%
Passenger Types	Times Critical Not Time Critical	+20% -20%
Penalty for changes in the ticket	First Class 0 First Class 1 First Class 2 Economy 0 Economy 1 Economy 2	30% 10% 0% 100% 50% 30%
Food	First Class 0 First Class 1 First Class 2 Economy 0 Economy 1 Economy 2	Cold sandwiches + drink Hot meal + drink À la carte No food Cold sandwiches + drink Hot meal + drink

Figure 4.2.1: Airline Strategies

Scala programming language is used to implement Spark [41]. It is a statically typed and high-level programming language running over the Java Virtual Machine (JVM) exposing functional programming constructs. The Scala interpreter can be modified such that Spark can be used interactively for defining RDDs, functions, variables etc. Spark is perhaps one of the first frameworks to allow an efficient general purpose programming language to define interactive processes on large datasets.

Spark introduces a data storage and processing abstraction called Resilient Distributed Datasets (RDDs). An RDD is a read-only distribution of objects that are partitioned across a set of machines and can be rebuilt if a partition is lost [42]. The RDDs are an abstraction for distributed shared memory [43]. However, there are differences between RDDs and DSMs. First, RDDs provides a much restricted programming model but to enable data regeneration in the event of node failure. DSM system achieve fault tolerance by check-pointing, while Spark regenerates lost partitions by using a lineage information from the RDD [44]. This means only the lost partitions are recomputed in parallel on other nodes, hence there is no need to revert to a previous checkpoint and risk losing computation progress. Also, there is no overhead if a node fails. Second, similar to MapReduce Spark pushes the computation to where data resides, instead of providing access to globally shared address space.

Resilient distributed datasets are an important abstraction in Spark that allow read-only collection of objects capable of rebuilding lost partitions across clusters. These RDDs can be reused in multiple parallel operations through memory caching. RDDs use lineage information about the lost partition in the rebuilding process.

Spark performs well in these cases, where Hadoop users have reported deficiency with MapReduce:

- **Iterative jobs**: Gradient-Descent is an excellent example of an algorithm that is repeatedly applied to the same dataset to optimize a parameter. While it is easy to represent each

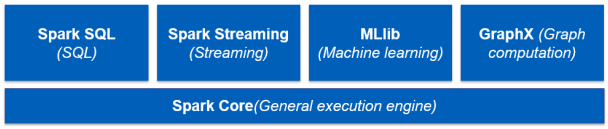

Figure 4.3.1: Spark Architecture Overview

iteration as a MapReduce job, the data from each iteration has to be loaded from the disk, incurring a significant performance penalty.

- **Interactive analytics**: Interfaces like Pig [42] and Hive [45] are commonly used running SQL queries on large datasets using Hadoop. Ideally this dataset is loaded into memory and queried repeatedly, but with Hadoop every query is executed a MapReduce job that incurs significant latency from disk read.

4.3.2 Spark Architecture

The following are the components of Apache Spark [46], shown in figure 4.3.1.

- **Spark Core**: Spark Core is the underlying general execution engine for spark platform that all other functionality is built upon. It provides In-Memory computing and referencing datasets in external storage systems.
- **Spark SQL**: Spark SQL is a component on top of Spark Core that introduces a new data abstraction called SchemaRDD, which provides support for structured and semi-structured data.
- **Spark Streaming**: Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics.
- It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.
- **MLlib (Machine Learning Library)**: MLlib is a distributed machine learning framework above Spark because of the distributed memory-based Spark architecture. Spark MLlib is 9x as fast as the Hadoop disk-based version of Apache Mahout.
- **GraphX**: GraphX is a distributed graph-processing framework on top of Spark. It provides an API for expressing graph computation that can model the user-defined graphs by using Pregel abstraction API. It also provides an optimized runtime for this abstraction.

4.3.3 Key terms used in Apache Spark

The following are the terms [46] used with Apache Spark, shown in figure 4.3.2.

- **Spark Context**: It holds a connection with Spark cluster manager. All Spark applications run as independent set of processes, coordinated by a SparkContext in a program.
- **Driver**: A driver is incharge of the process of running the main() function of an application and creating the SparkContext.
- **Worker**: A worker, on the other hand, is any node that can run program in the cluster. If a process is launched for an application, then this application acquires executors at worker node.

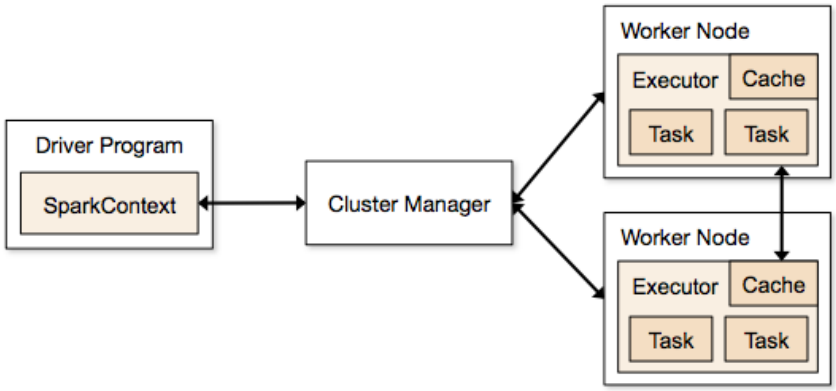

Figure 4.3.2: Apache Spark Terms

- **Cluster Manager**: Cluster manager allocates resources to each application in driver program. There are three types of cluster managers supported by Apache Spark - Standalone, Mesos and YARN. We can install any cluster manager, each has its own unique advantages depending upon the goal. They all are different in terms of scheduling, security and monitoring. Once SparkContext connects to the cluster manager, it acquires executors on a cluster node, these executors are worker nodes on cluster which work independently on each tasks and interact with each other.

4.3.4 Cluster Specification

Hadoop and Spark are designed to run on commodity hardware. That means that you are not tied to expensive, proprietary offerings from a single vendor; rather, you can choose standardized, commonly available hardware from any of a large range of vendors to build your cluster.

"Commodity" does not mean "low-end". Low-end machines often have cheap components, which have higher failure rates than more expensive (but still commodity class) machines. When you are operating tens, hundreds, or thousands of machines, cheap components turn out to be a false economy, as the higher failure rate incurs a greater maintenance cost. On the other hand, large database class machines are not recommended either, since they don't score well on the price/performance curve. And even though you would need fewer of them to build a cluster of comparable performance to one built of mid-range commodity hardware, when one did fail it would have a bigger impact on the cluster, since a larger proportion of the cluster hardware would be unavailable.

Hardware specifications rapidly become obsolete, but for the sake of experimentation, our choice of machine for running a Hadoop DataNode and TaskTracker, Workers have the specifications as shown illustrated in Table 4.3.1. While the hardware specification for cluster will assuredly be different, Hadoop and Spark are designed to use multiple cores and disks, so it will be able to take full advantage of more powerful hardware.

The bulk of Hadoop is written in Java, and can therefore run on any platform with a JVM, although there are enough parts that harbor Linux assumptions (the control scripts, for example) to make it unwise to run on a non-Linux platform in production.

The bulk of Spark is written in Scala language and it has some code written in Java, Python and R. It can therefore run on any platform with a JVM. Apache Spark provides several APIs for programmers which include Java, Scala, R and Python.

Table 4.3.1: Spark Cluster Specifications

Parameter	Value
The number of Nodes: Cores in Use: Availed Memory: Availed HDFS:	10 (1 Master, 9 Workers/Slaves) 72 24.9 GB 6.4 TB
Master Machine	CPU 2.25 Ghz RAM - 4 GB HDD 1 TB OS - Ubuntu 16.04 LTS
Slave Machine	CPU 2.25 Ghz RAM - 4 GB HDD 1 TB OS - Ubuntu 16.04 LTS
Ethernet Switch	16 Port 10/100 Mbp
Software Tools	Hadoop 2.7, Spark 2.1.0, Scala 2.12.1, OpenJDK 1.8 Hive 2.0.0, HBase 1.1.4 Flume 1.6.0, Anaconda3 4.3.0 R 3.3.2, RStudio 1.0.136
Language Support	Scala, Java, Python, R,& MapReduce

4.3.5 Cluster Setup and Installation

There are various ways to install and configure Apache Spark. This section describes how to do it from scratch using the Apache Spark distribution, and will give the background to cover the things you need to think about when setting up Spark. Alternatively, if you would like to use RPMs or Debian packages for managing your Spark installation, then you can start with Cloudera's Distribution.

To ease the burden of installing and maintaining the same software on each node, it is normal to use an automated installation method like Red Hat Linux's Kickstart or Debian's Fully Automatic Installation. These tools allow you to automate the operating system installation by recording the answers to questions that are asked during the installation process (such as the disk partition layout), as well as which packages to install. Crucially, they also provide hooks to run scripts at the end of the process, which are invaluable for doing final system tweaks and customization that is not covered by the standard installer.

The *Appendix A* describe the customizations that are needed to setup and run Spark.

4.3.6 10 Nodes Spark Cluster

We had setup Apache Spark cluster on on top of Apache Hadoop Cluster in Data Science Lab @B508 R11. The cluster architecture diagram is shown in figure 4.3.3.

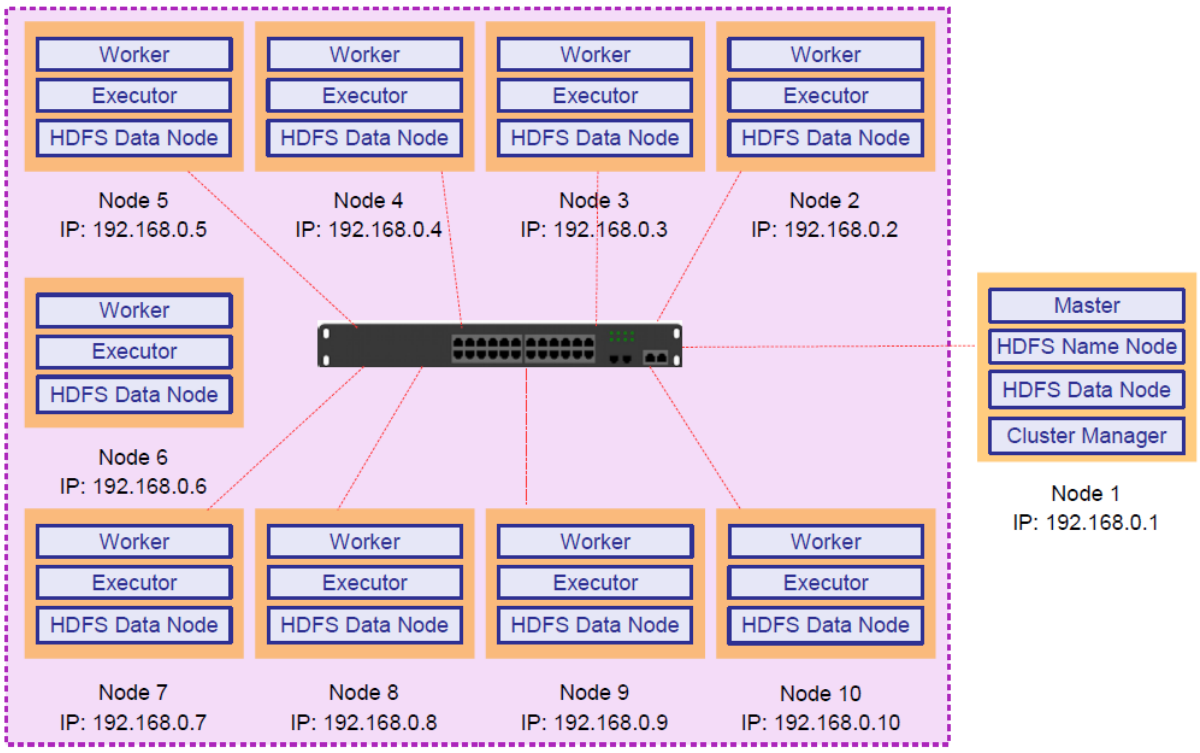

Figure 4.3.3: 10 Nodes Apache Spark Cluster in DSSIG Lab

4.4 Building Model

After construction of the dataset and preprocessing, the modeling method was determined and the model was implemented using SparkR API.

R is a popular tool for statistics and data analysis. It has rich visualization capabilities and a large collection of libraries that have been developed and maintained by the R developer community. One drawback to R is that it's designed to run on in-memory data, which makes it unsuitable for large datasets. Spark is a distributed engine for processing many Terabytes of data. It is a versatile tool with capabilities for data processing, SQL analysis, streaming and machine learning.

SparkR combines the benefits of Spark and R by allowing Spark jobs to be called from within R. This allows the analyst to leverage Spark's ability to build aggregate statistics over large, multi-Terabyte datasets and then bring the smaller aggregated data back into R for visualization and analysis.

In statistics, the generalized linear model (GLM) is a flexible generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value.

Linear regression is mainly used to establish a relationship between dependent variable and independent variable. It helps in estimating the impact of independent variable over a dependent variable. The coefficients explain the impact of changes in an independent variable on the dependent variable.

The goal is to model the fare on the historical data. So, given certain characteristics including itinerary fare, customer type, distance, number of competitors and which carriers are flying there

etc. In order to achieve this, a linear model can be used:

$$y = X\beta + u \tag{4.4.1}$$

The dependent variable $y(n \times 1)$ is a vector of fares. The matrix $X(n)$ contains the values of the k explanatory variables, and $\beta(k \times 1)$ is a vector containing the weights for each of these variables. How much and in what direction does each variable influence the fare? u is an $(n \times 1)$ error term.

The model will allow us to give the fare y as a function of the explanatory variables X. In order to estimate the coefficients β, and to keep the model "simple" enough, five "classical" assumptions are applied.

- In reality, the true model for y is linear in X.
- The error u is assumed to be zero on average, i.e. $E(u) = 0$.
- There are no completely redundant variables and hence no multicollinearity, i.e. rank $rk(x) = k$.
- There are spherical errors: $var(u) = \sigma^2 I$.
- The error u is normally distributed, $u \approx N$.

From the generalized linear model, the relation between changes in one of the explanatory variables and the change in fare can easily be found.

The GLM supports the 6 types of link functions based on the specific type and usage of response variable, as shown in below figure.

Family	Link Function	When to use it:
• gaussian (link = "**identity**")	μ	homogeneous and normal residuals, real numbers, may be negative
• poisson (link = "**log**" *or* "identity" *or* "sqrt")	ln(μ)	variance ~ mean, non-normal residuals, count data (>0, integers) (some packages also allow related negative binomial)
• Gamma (link = "**inverse**" *or* "identity" *or* "log")	μ^{-1}	variance ~ mean, even more skew in residuals, real numbers (>0)
• inverse.gaussian (link = "**inverse**" *or* "identity" *or* "log")	μ^{-2}	variance ~ mean, even more skew in residuals, real numbers (>0)
• binomial (link = "**logit**")	ln(p/(1-p))	binary response predicted by quantitative variables; logistic regression
• quasi, quasibinomial, quasipoisson	several	can address overdispersion not in above

The following SparkR API code snippet build GLM model for fare processing:

```
model <- spark.glm(Fare ~., train_data, family = "gaussian")
summary(model)
```

Chapter 5

Results and Discussion

5.1 Apache Spark Cluster GUI

As mentioned in section 4.3 of chapter 04, we have developed a Apache spark cluster of 10 nodes(1 master, 9 workers) on top of Apache Hadoop cluster in Data Science SIG. We are able to avail 24.9 GB RAM, shown in figure 5.1.1

After developing a cluster, we run our R script using RStudio GUI in distributed mode on the cluster.

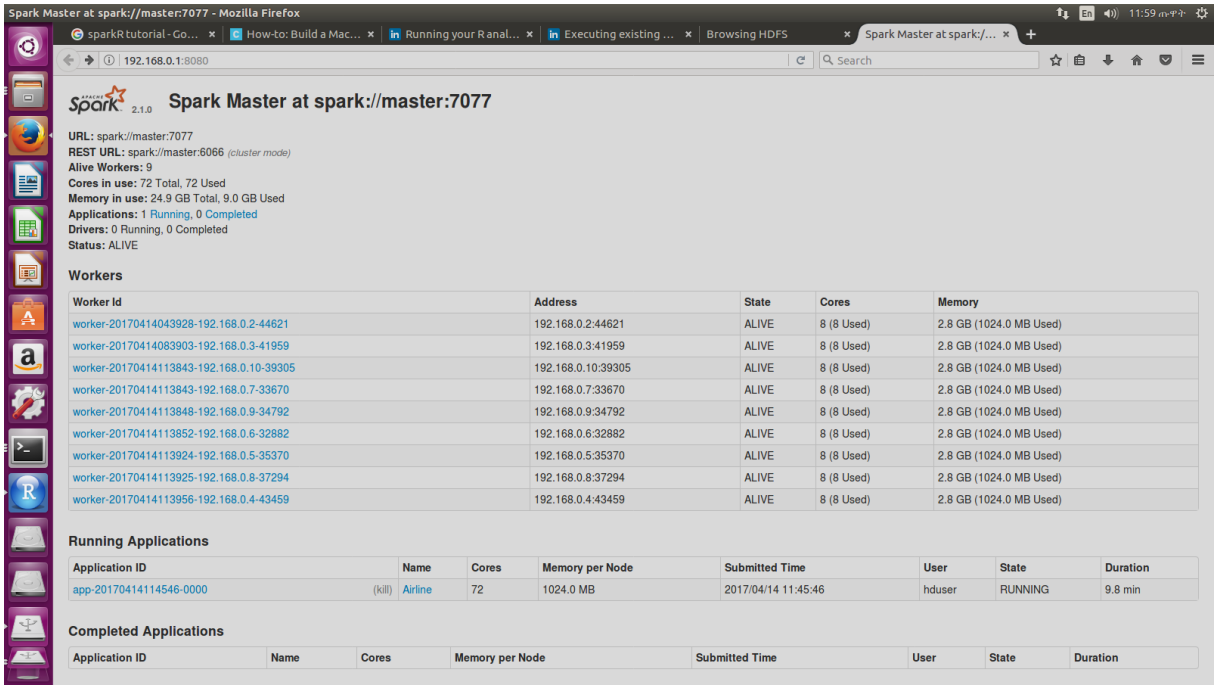

Figure 5.1.1: Apache Spark GUI

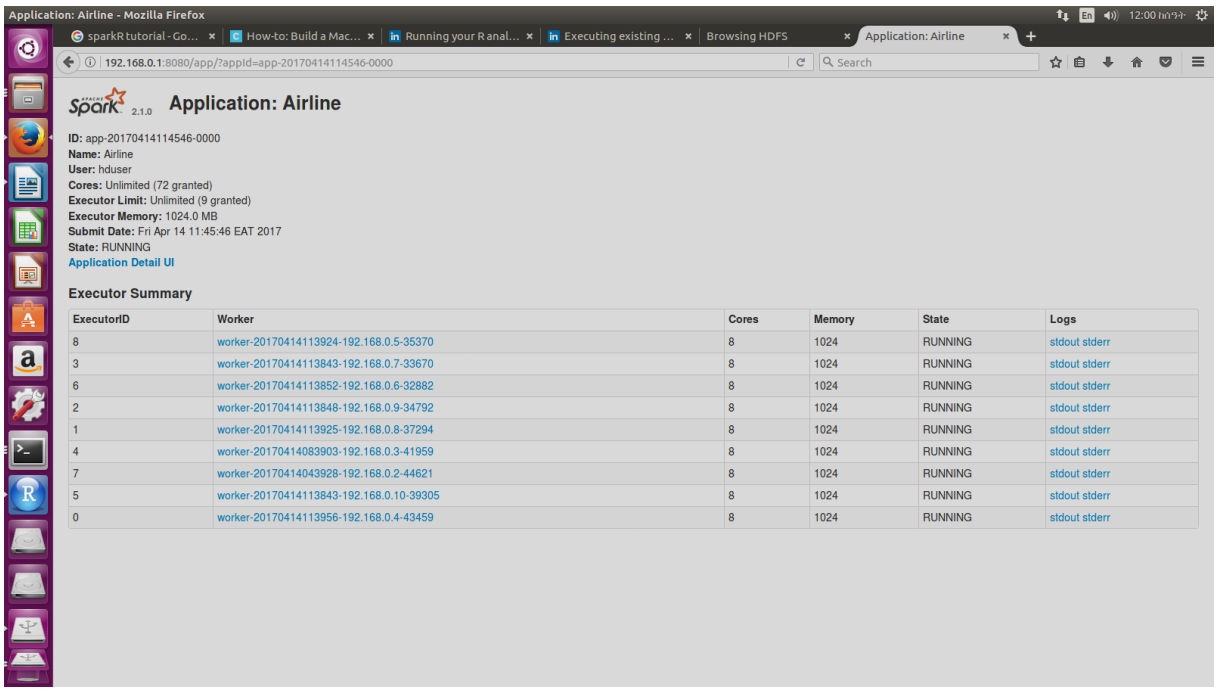

Figure 5.1.2: Airline Application Running

5.2 Airline Data Exploration

The research study has been done on Ticket and Coupon data taken from Bureau of Transportation Statistics for the year 2016 quarter 1 to quarter 4. The Table 5.2.1 gives summary of collected data.

Table 5.2.1: Data Summary

S.No.	File	Size	# of Observations	# of Variables
1	Ticket_2016_Q1	461 MB	3,490,008	26
2	Ticket_2016_Q2	461 MB	3,961,077	26
3	Ticket_2016_Q3	519 MB	3,928,376	26
4	Ticket_2016_Q4	519 MB	3,924,694	26
5	Coupon_2016_Q1	1.68 GB	9,081,945	37
6	Coupon_2016_Q2	1.91 GB	10,289,457	37
7	Coupon_2016_Q3	1.89 GB	10,177,039	37
8	Coupon_2016_Q4	1.87 GB	10,080,158	37
		9.35 GB	**54,932,754**	**63**

```
     ItinID            Coupons            Year         Quarter      Origin        OriginAirportID
 Min.   :2.016e+05   Min.   : 1.000   Min.   :2016   Min.   :1   LAX     : 115944   Min.   :10135
 1st Qu.:2.016e+11   1st Qu.: 2.000   1st Qu.:2016   1st Qu.:1   ORD     : 114065   1st Qu.:11298
 Median :2.016e+11   Median : 2.000   Median :2016   Median :1   BOS     :  90284   Median :12892
 Mean   :1.651e+11   Mean   : 2.602   Mean   :2016   Mean   :1   SFO     :  87110   Mean   :12795
 3rd Qu.:2.016e+11   3rd Qu.: 4.000   3rd Qu.:2016   3rd Qu.:1   DEN     :  84150   3rd Qu.:14107
 Max.   :2.016e+11   Max.   :13.000   Max.   :2016   Max.   :1   EWR     :  81850   Max.   :16218
                                                                 (Other):2916605
 OriginAirportSeqID OriginCityMarketID OriginCountry OriginStateFips  OriginState       OriginStateName
 Min.   :1013503    Min.   :30009      US:3490008    Min.   : 1.00   CA      : 415925   California: 415925
 1st Qu.:1129804    1st Qu.:30781                    1st Qu.:12.00   FL      : 297148   Florida   : 297148
 Median :1289203    Median :31703                    Median :27.00   TX      : 285651   Texas     : 285651
 Mean   :1279498    Mean   :31921                    Mean   :27.83   NY      : 218189   New York  : 218189
 3rd Qu.:1410702    3rd Qu.:32600                    3rd Qu.:42.00   IL      : 151859   Illinois  : 151859
 Max.   :1621801    Max.   :36133                    Max.   :78.00   VA      : 141583   Virginia  : 141583
                                                                     (Other):1979653   (Other)   :1979653
   OriginWac       RoundTrip          OnLine         DollarCred      FarePerMile         RPCarrier
 Min.   : 1.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :  0.0000   AA      :624137
 1st Qu.:33.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:  0.1121   WN      :617831
 Median :44.00   Median :1.0000   Median :1.0000   Median :1.0000   Median :  0.1860   DL      :554498
 Mean   :53.46   Mean   :0.6388   Mean   :0.6546   Mean   :0.9945   Mean   :  0.2701   UA      :375740
 3rd Qu.:81.00   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:  0.3245   OO      :193236
 Max.   :93.00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :965.0980   EV      :151742
                                                                                       (Other):972824
   Passengers        ItinFare          BulkFare            Distance     DistanceGroup      MilesFlown
 Min.   :  1.000   Min.   :     0.0   Min.   :0.0000000   Min.   :   11   Min.   : 1.000   Min.   :   11
 1st Qu.:  1.000   1st Qu.:   225.0   1st Qu.:0.0000000   1st Qu.: 1138   1st Qu.: 3.000   1st Qu.: 1134
 Median :  1.000   Median :   368.0   Median :0.0000000   Median : 1923   Median : 4.000   Median : 1914
 Mean   :  2.074   Mean   :   428.2   Mean   :0.0002181   Mean   : 2255   Mean   : 5.005   Mean   : 2248
 3rd Qu.:  1.000   3rd Qu.:   550.0   3rd Qu.:0.0000000   3rd Qu.: 2896   3rd Qu.: 6.000   3rd Qu.: 2888
 Max.   :654.000   Max.   :850300.0   Max.   :1.0000000   Max.   :18862   Max.   :25.000   Max.   :18862

 ItinGeoType       X
 Min.   :1.000   Mode:logical
 1st Qu.:2.000   NA's:3490008
 Median :2.000
 Mean   :1.934
 3rd Qu.:2.000
 Max.   :2.000
```

Figure 5.2.1: Summary - Year 2016 Q1

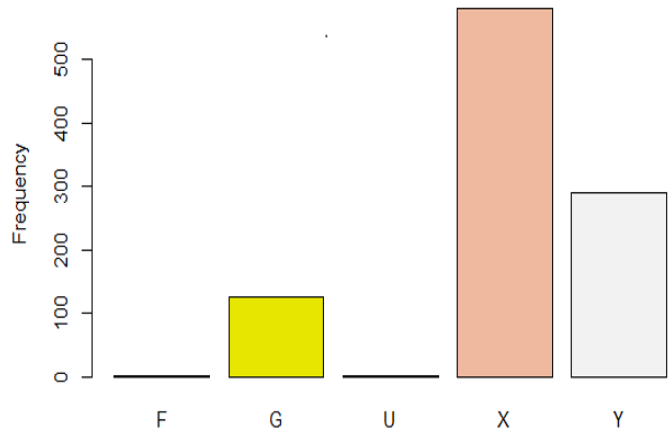

Figure 5.2.2: Histogram of Fare Classes frequency in Year 2016 Q1

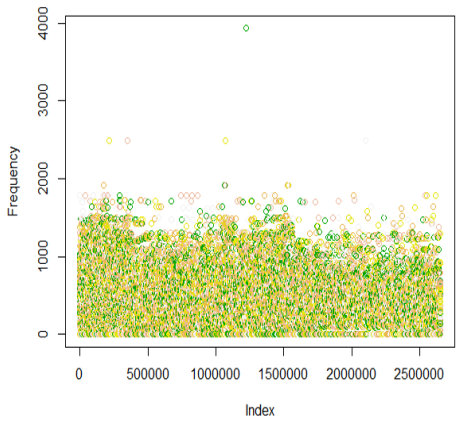

Figure 5.2.3: Histogram of Itinerary Fare frequency in Year 2016 Q1

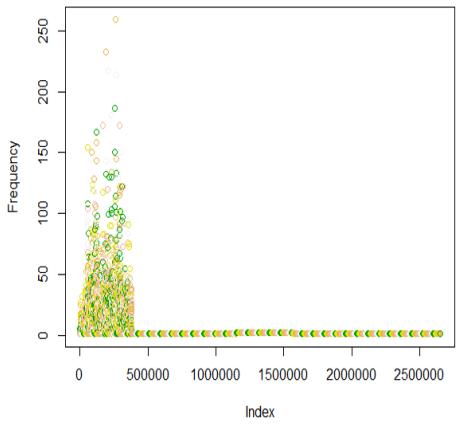

Figure 5.2.4: Histogram of Passenger's frequency in Year 2016 Q1

5.3 Personalization

In order to find high-value customers, customers were segmented by adding one more column 'HVC' to filtered and merged trained data as shown in figure 5.3.1. The column values are either 1 or 0. The value 1 indicates high value customer to whom the proposed pricing policy would be applied. The value 0 indicates normal customer to whom no policy would be applied(normal fare).

5.4 Building Prediction Model and Evaluation

The merged data is partitioned into 60% training and 40% testing data. Then the researchers built two models: generalized linear model and linear model as the data is numerical.

5.4.1 Generalized Linear Model

Generalized linear models unify various statistical models such as linear and logistic regression through the specification of a model family and link function. In R, such models can be fitted by passing an R model formula, family, and training dataset to the glm() function. Spark 1.5 or later one extends glm() to operate over Spark DataFrames, which are distributed data collections managed by Spark.

	ItinID	ItinFare	FarePerMile	MilesFlown	Origin	Dest	OpCarrier	Passengers	Distance	FareClass	HVC
1	2016134	303	0.2612	1160	DTW	ABE	EV	1	425	X	0
2	2016170	456	0.3476	1312	DTW	ABE	9E	1	425	X	0
3	20161113	640	0.2508	2552	ATL	ABE	DL	1	692	X	1
4	20161305	688	0.2628	2618	ATL	ALB	DL	1	853	X	1
5	20161312	248	0.0813	3050	ATL	ALB	DL	1	853	X	1
6	20161367	640	0.1323	4838	ATL	ALB	DL	1	853	X	1
7	20161399	1225	0.2458	4983	MSP	ALB	9E	1	980	X	1
8	20161451	688	0.3426	2008	MSP	ALB	9E	1	980	X	1
9	20161658	311	0.1138	2732	MSP	ALB	9E	1	980	X	1
10	20161672	456	0.1335	3416	ATL	ALB	DL	1	853	X	1
11	201611491	456	0.2750	1658	DTW	ATW	DL	1	296	X	0
12	201611627	303	0.1237	2450	ATL	ATW	DL	1	765	X	1
13	201611783	280	0.0933	3002	JFK	AUS	DL	1	1521	X	1
14	201611835	578	0.1876	3081	ATL	AUS	DL	1	813	X	1
15	201611958	280	0.0782	3580	MSP	AUS	9E	1	1042	X	1
16	201612018	1272	0.4659	2730	DTW	AUS	9E	1	1149	X	1
17	201612156	738	0.7124	1036	DTW	AZO	9E	1	113	X	0
18	201612332	738	0.2068	3568	DTW	AZO	EV	1	113	X	0
19	201612357	738	0.6010	1228	DTW	AZO	OO	1	113	X	0

Showing 1 to 20 of 14,355 entries

Figure 5.3.1: High Value Customer's Data Frame

```
summary(model.glm)

Call:
glm(formula = ItinFare~ItinID+HVC+Distance+Passengers+FarePerMile+MilesFlown)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1023.11    -48.24     -3.60     53.43    796.32

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.334e+00  3.418e+01   0.098 0.922332
ItinID      -7.314e-11  4.061e-11  -1.801 0.072256 .
HVC          1.378e+01  7.396e+00   1.863 0.062957 .
Distance     7.618e-03  9.499e-03   0.802 0.422884
Passengers  -3.793e+00  5.210e+00  -0.728 0.466838
FarePerMile  8.761e+02  3.660e+01  23.937  < 2e-16 ***
MilesFlown   6.569e-02  3.319e-03  19.791  < 2e-16 ***

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 14786.6)

    Null deviance: 344974676  on 8613  degrees of freedom
Residual deviance: 127268236  on 8607  degrees of freedom
```

```
AIC: 107162

Number of Fisher Scoring iterations: 2
```

The result shows the important coefficients from historical data which effects the fare prediction are HVC, ItinFare, FarePerMile, MilesFlown.

The modeling can also provide us with a predictive equation. For every actual fare, the predicted fare is computed by the model using the route-carrier characteristics.

The following code fragment gives the first 10 predictions of the test dataset.

```
> pred1 <- predict(model.glm, newdata = sampleTest,type="response")
> data.frame(prediction,sampleTest\$ItinFare)[1:10,]

       pred1 sampleTest.ItinFare
3   533.7108                640
4   552.5947                688
5   339.9974                248
6   590.2400                640
7   760.9730               1225
14  493.9610                578
16  841.3986               1272
17 1012.5919                838
18  552.9216                738
22  394.7742                303
```

5.4.2 Linear Model

```
model.lm <- lm(ItinFare~ItinID+HVC+Distance+Passengers+FarePerMile+MilesFlown,
              data = sampleTrain)
summary(model.lm)

Call:
lm(formula = ItinFare ~ ItinID + HVC + Distance + Passengers +
    FarePerMile + MilesFlown, data = sampleTrain)

Residuals:
     Min       1Q   Median       3Q      Max
-1023.11   -48.24    -3.60    53.43   796.32

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.334e+00  3.418e+01   0.098 0.922332
ItinID      -7.314e-11  4.061e-11  -1.801 0.072256 .
HVC          1.378e+01  7.396e+00   1.863 0.062957 .
Distance     7.618e-03  9.499e-03   0.802 0.422884
Passengers  -3.793e+00  5.210e+00  -0.728 0.466838
FarePerMile  8.761e+02  3.660e+01  23.937  < 2e-16 ***
MilesFlown   6.569e-02  3.319e-03  19.791  < 2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 52.42 on 556 degrees of freedom
Multiple R-squared:  0.8375,Adjusted R-squared:  0.7709
F-statistic: 12.57 on 228 and 556 DF,  p-value: < 2.2e-16

The following code fragment gives the first 10 predictions of the test dataset.

\begin{verbatim}
> pred2 <- predict(model.lm, newdata = sampleTest,type="response")
> data.frame(prediction,sampleTest\$ItinFare)[1:10,]

       pred2 sampleTest.ItinFare
3   533.7108                640
4   552.5947                688
5   339.9974                248
6   590.2400                640
7   760.9730               1225
14  493.9610                578
16  841.3986               1272
17 1012.5919                838
18  552.9216                738
22  394.7742                303
```

R-squared is a statistical measure of how close the data are to the fitted regression line. In general, the higher the R-squared, the better the model fits your data. The Linear Regression model results R-squared value 77.09%.

5.5 Discussion

This research work suggests improvement required in terms of customer's personalization and fare prediction factors while answering the research questions mentioned in section 1.4 of chapter 01 as follows:

- **RQ1: How Predictive Analytics improves Airline's Business?**
  Currently airlines are using limited data sources for personalizing customers and predicting prices. Operational databases such as CRM, Sales and other airline related RDBMS are traditional data sources that are limited. Big data sources such as social media data, web logs, click streams, operational RDBMS, emails, and business related documents applied by predictive analytics methods are key to provide hidden insights for airlines.

  The data acquisition, cleaning, loading, preprocessing and analytics require very expensive hardware and computationally scalable software services. Big Data Analytics tools and techniques have the capability to handle complex data within minimum time frame, increase business decision quality supporting cheap commodity hardware. Among the Data analytics techniques, predictive data analytics is the one which increases the confidence of business in segmenting and understanding customers, making profit, reducing loss, solving uncertainty, pointing critical decisions, and so on.

By switching over to big data based solution, it is estimated that the fare processing times can be cut down from days to matter of hours or even minutes. Our experimentation results have showed that the data loaded in a stand alone machine with 2.2 GHz processing power and 8 GB memory took more than 40 minutes to give expected response. The researchers were also noted shortage of memory error while performing data analysis. When the same task was given to our 10 nodes commodity spark cluster, model building and prediction results were seen within 9 minutes.

- **RQ2: Which factors influence fare?**
  The research work used Air Ticket and Coupon two datasets to predict fare and segment customers as high value or normal. The researchers used multitude of criteria to segment customers and apply pricing policy to high value customers. The results showed that Itinerary Fare, the Boolean value indicating high value customer or not, few origin and destination places, fare class, number of passengers are the most important coefficients which influence the response variable Itinerary Fare by the prediction model.

Chapter 6

Conclusion and Future Scope

Our goal was to study and look for improvement in predictive analytics which could predict fare from the heterogeneous historical Big Data. This was done by taking freely available data from the U.S. Department of Transportation and performing an econometric analysis. The research study proposed Big Data Architecture for personalization of customers and smart pricing. By switching over to big data based solution, it is estimated that the fare processing times can be cut down from days to matter of hours or even minutes. The airline industry operates in high risk domain vulnerable to a large number of factors, and this agility will be a vital tool in maximizing revenues. Big data based solutions for personalization helps in understanding and deriving granular personalization parameters and understand customer customer's social status and needs based on their interaction in social media systems.

However, limited by the current data source that we have, we are unable to extract more information. In the future, more features, such as the available seat, the departure time of a day, and whether the departure day is a holiday or not, can be added to the model to improve the performance of the predicting model.

The study expects and recommend the following for the better of airlines industry:

- Extracting, processing, and analyzing un-structured data from external sources such as Agent systems, External pricing systems and social media, which would help airlines to create deeper personalization preference of customers and also understand their current social status and preferences.
- Inventing new predictive and prescriptive analytics for airlines.
- Identifying or driving granular customer's personalization parameters to reach their needs.

Bibliography

[1] Allen Booz and Hamilton, "Airline Analytics".

[2] Balaji Jagannathan, "Empower, Collaborate, Reimagine: three key ingredients for successful airlines Customer Experience Management", TCS, 2014.

[3] P. Russom, "Big Data Analytics", Tech. Rep., In: TDWI Best Practices Report, 2011.

[4] Thomas H. Davenport and Jill Dyche, "Big Data in Big Companies", 2013.

[5] Tom White, *Hadoop: The definitive guide*, O'Reilly Media, 2012.

[6] W.R. Kubick, "Big Data, Information and Meaning", In: Clinical Trial Insights, 2012.

[7] EMC, "Data Science and Big Data Analytics", In: EMC Education Services, 2012.

[8] TechAmerica, "Demystifying Big Data: A Practical Guide to Transforming the Business of Government", 1-40, In: TechAmerica Reports, 2012.

[9] M.N. Adams, "Perspectives on Data Mining", *International Journal of Market Research*, vol. 52, no. 1, pp. 11–19, 2010.

[10] Kusiak A. Song, Z., "Optimizing Product Configurations with a Data Mining Approach", *International Journal of Production Research*, vol. 47, no. 2, pp. 1733–1751, 2009.

[11] K. Bakshi, "Considerations for Big Data: Architecture and Approaches", *In: Proceedings of the IEEE Aerospace Conference*, pp. 1–7, 2012.

[12] Dolan B. Dunlap M. Hellerstein J.M. Welton C. Cohen, J., "MAD Skills: New Analysis Practices for Big Data", *Proceedings of the ACM VLDB Endowment*, vol. 2, no. 2, pp. 1481–1492, 2009.

[13] Lim H. Luo G. Borisov N. Dong L. Cetin F.B. Babu S. Herodotou, H., "Starfish: A Self-tuning System for Big Data Analytics", *In Proceedings of the Conference on Innovative Data Systems Research*, pp. 261–272, 2011.

[14] Zeier A. Plattner, H., "In-Memory Data Management: An Inflection Point for Enterprise Applications", *Springer*.

[15] Lee R. Huai Y. Shao Z. Jain N. Zhang X. Xu Z. He, Y., "RCFile: A Fast and Space efficient Data Placement Structure in MapReduce-based Warehouse Systems", *In IEEE International Conference on Data Engineering (ICDE)*, pp. 1199–1208, 2011.

[16] Song I. Davis K.C. Cuzzocrea, A., "Analytics over Large-Scale Multidimensional Data: The Big Data Revolution!", *In: Proceedings of the ACM International Workshop on Data Warehousing and OLAP*, pp. 101–104, 2011.

[17] Huberman B.A. Asur, S., "Predicting the Future with Social Media", *ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 1, pp. 492–499, 2010.

[18] Hsinchun C. Lusch R. Li S.H. Zeng, D., "Social Media Analytics and Intelligence", *IEEE Intelligent Systems*, vol. 25, no. 6, pp. 13–16, 2010.

[19] Gijsbers G. Van der Valk, T., "The Use of Social Network Analysis in Innovation Studies: Mapping Actors and Technologies", *Innovation: Management, Policy Practice*, vol. 12, no. 1, pp. 5–17, 2010.

[20] O. Serrat, "Social Network Analysis", *Knowledge Network Solutions*, vol. 28, pp. 1–4, 2009.

[21] Martin-Bautista M.J. Blanco I. Torre C. Sanchez, D., "Text Knowledge Mining: An Alternative to Text Data Mining", *IEEE International Conference on Data Mining Workshops*, pp. 664–672, 2008.

[22] Devi-K.N. Bhaskaran V.M. Mouthami, K., "Sentiment Analysis and Classification Based on Textual Reviews", *International Conference on Information Communication and Embedded Systems (ICICES)*, pp. 271–276, 2013.

[23] Chui-M. Brown B. Bughin J. Dobbs R. Roxburgh C. Byers A.H. Manyika, J., "Big Data: The Next Frontier for Innovation, Competition, and Productivity", 1-156, McKinsey Global Institute Reports, 2011.

[24] Wei-J. Sundaresan N. Ma K.L. Shen, Z., "Visual Analysis of Massive Web Session Data", *Large Data Analysis and Visualization (LDAV)*, pp. 65–72, 2012.

[25] Stoffel-A. Behrisch M. Mittelstadt S. Schreck T. Pompl R. Weber S. Last H. Keim D. Zhang, L., "Visual Analytics for the Big Data EraA Comparative Review of State-of-the-Art Commercial Systems", *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 173–182, 2012.

[26] Friedman Hastie T, Tibshirani R, "The elements of statistical learning: data mining, inference, and prediction", *Springer*, Oct. 2001.

[27] Otto Koppius Galit Shmueli, "Predictive vs. Explanatory Modeling in IS Research", pp. 10–19, 2001.

[28] Hanumanth Sistla Sastry and M. S. Prasad Babu, "Cluster Analysis of Material Stock Data of Enterprises", *International Journal of Computer Information Systems (IJCIS)*, vol. 6, no. 6, pp. 8–19, 2013.

[29] Sachchidanand Singh and Nirmala Singh, "Big Data Analytics", *International Conference on Communication, Information Computing Technology (ICCICT)*, Oct. 2012.

[30] Frank Acito and Vijay Khatri, "Business analytics: Why now and what next?", *Elsevier*, pp. 565–570, 2014.

[31] Tulinda Larsen, "Cross-Platform Aviation Analytics Using Big-Data Methods", *Integrated Communications Navigation and Surveillance (ICNS) Conference*, April 23-25 2013.

[32] Paul Comitz David Sweet Steve Bliesner Samet Ayhan, Johnathan Pesce and Gary Gerberick, "Predictive Analytics with Aviation Big Data", *Integrated Communications Navigation and Surveillance (ICNS) Conference*, April 23-25 2013.

[33] David Martens Enric Junque de Fortuny and Foster Provost, "Predictive Modelling with Big Data: Is Bigger Really Better?", *Integrated Communications Navigation and Surveillance (ICNS) Conference*, April 2013.

[34] Tuchinda R. Knoblock C. A. Yates A. Etzioni, O., "To buy or not to buy: mining airfare data to minimize ticket purchase price", *In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 119–128, 2003.

[35] Gini M. Groves, W., "A regression model for predicting optimal purchase timing for airline tickets", Tech. Rep., University of Minnesota, Minneapolis, MN, 2011.

[36] K. Rama-Murthy, "Modeling of United States Airline Fares–Using the Official Airline Guide (OAG) and Airline Origin and Destination Survey (DB1B)".

[37] M. Papadakis, *Predicting Airfare Prices*, 2014.

[38] Shenli Yuan Ruixuan Ren, Yunzhe Yang, "Prediction of Airline Ticket Price".

[39] "Bureau of transportation Statistics. Number of u.s. airports".

[40] "CNN. A day in the life of the worlds busiest airport.".

[41] Michael J Franklin Scott Shenker Matei Zaharia, Mosharaf Chowdhury and Ion Stoica, "Spark: cluster computing with working sets", *In USENIX conference on Hot topics in cloud computing*, pp. 10–16, 2010.

[42] U. Srivastava R. Kumar C. Olston, B. Reed and A. Tomkins, "Pig latin: a not-so-foreign language for data processing", *In SIGMOD 08*, 2008.

[43] B. Nitzberg and V. Lo, "Distributed shared memory: a survey of issues and algorithms", *Computer*, vol. 24, no. 8, August 1991.

[44] R. Bose and J. Frew, "Lineage retrieval for scientific data processing: a survey", *ACM Computing Surveys*, vol. 37, no. 128, 2005.

[45] "Apache Hive", http://hadoop.apache.org/hive.

[46] "Apache Spark Document", http://spark.apache.org/docs/1.3.0/cluster-overview.html.

Appendix A

Multi Node Apache Spark Installation

```
Step 1: Install Java
        Java is the main prerequisite for Apache Spark. Install latest java.
        \$ sudo apt-get update
        \$ sudo apt-get install default-jdk
        Then, verify the existence of java in your system using the command 'java - version'.
        \$ java -version

Step 2. Set the Host Name as 'master' in /etc/hostname file

Step 3. Set the Known Hosts in /etc/hosts file
        \$ sudo gedit /etc/hosts

       192.168.0.1 master

        Restart the system \& Login

Step 4 Create a dedicated user account for hadoop
       \$ sudo addgroup hadoop
       \$ sudo adduser --ingroup hadoop hduser
       \$ sudo usermod -a -G sudo hduser
       \$ su hduser

        Restart the system \& Login to hduser

Step 5. Configure ssh

     5.1. Generate private and public key pair at terminal using

         \$ sudo apt-get install ssh
         \$ ssh-keygen

     5.2. To enable ssh to the local machine

         \$ cat \$HOME/.ssh/id_rsa.pub >>\$HOME/.ssh/authorized_keys
         \$ ssh master

Step 6. Disable IPV6 by including the following lines in /etc/sysctl.conf file
        \$ sudo gedit /etc/sysctl.conf

         net.ipv6.conf.all.disable_ipv6 = 1
         net.ipv6.conf.default.disable_ipv6 = 1
         net.ipv6.conf.lo.disable_ipv6 = 1
```

```
       Reboot the machine to make the changes and logon to hduser

Step 7. To find the java path
        \$sudo update-alternatives --config javac

Step 8. Install Scala
        \$sudo apt-get install scala
        or
        Download Scala and extract
        \$cd /usr/local
        \$sudo tar xvzf \$HOME/Downloads/scala-2.12.1.tar.gz
        \$sudo chmod 777 scala-2.12.1

Step 9. Download and Install Spark

        http://spark.apache.org/downloads.html

        Install Spark:
          \$cd /usr/local
          \$sudo tar xvzf \$HOME/Downloads/spark-2.1.0-bin-hadoop2.7.tar
          \$sudo chmod 777 spark-2.1.0-bin-hadoop2.7

Step 10. Set the environment variables.
         \$sudo gedit \$HOME/.bashrc

         Include the following lines in the \$HOME/.bashrc file

         # Set JAVA home directory
         export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

         # Set Scala-related environment variables
         export SCALA_HOME=/usr/local/scala-2.12.1

         # Set Spark-related environment variables
         export SPARK_HOME=/usr/local/spark-2.1.0-bin-hadoop2.7

         #Add Spark bin/ directory to PATH
         export PATH=\$PATH:\$SCALA_HOME/bin:\$SPARK_HOME/bin

Step 11. Set environment variables.
         \$sudo gedit /etc/profile


         Include the following lines /etc/profile file

         # Insert JAVA_HOME
           JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

         #--in PATH variable just append at the end of the line PATH=\$PATH:\$JAVA_HOME/bin:\$SCALA_

         #--Append SPARK_HOME at end of the export statement
         export PATH JAVA_HOME SCALA_HOME SPARK_HOME

Step 12. Run the .bashrc& profile files from the \$ prompt for updating the changes
         \$source \$HOME/.bashrc
         \$source /etc/profile
```

```
Step 13. Check Spark Installation
         \$ echo \$SPARK_HOME
         \$ bin/spark -version

Step 14. Configure Spark

         Edit configuration file spark-env.sh (in \$SPARK_HOME/conf/) and
 set following parameters: (Create a copy of template of spark-env.sh and rename it)

         \$ sudo cd \$SPARK_HOME
         \$ cd conf
         \$ sudo cp spark-env.sh.template spark-env.sh
         \$ sudo gedit spark-env.sh

          export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
          export SPARK_WORKER_CORES=8

Step 15. Add Slaves
         Create configuration file slaves (in \$SPARK_HOME/conf/) and add following entries:
        \$ sudo gedit slaves
           slave1
           slave2

Step 16: Install Spark on all slaves

         16.1 Setup Pre-requisites on all the slaves:

              Run following steps on all the slaves (or worker nodes):

              16.1.1 Add Entries in hosts file
              16.1.2 Install Java 7
              16.1.3 Install Scala
              16.1.4 Setup environment variables
              16.2 Copy setups from master to all the slaves
                 16.2.1 Create tar-ball of configured setup:
                        \$ tar czf spark.tar.gz spark-2.1.0-bin-hadoop2.7
                           NOTE: Run this command on Master
                 16.2.2 Copy the configured tar-ball on all the slaves
                        \$ scp spark.tar.gz slave01:~
                           NOTE: Run this command on Master
              16.3 Un-tar configured spark setup on all the slaves
                  \$ cd /usr/local
                  \$ sudo tar xzf /home/hduser/spark.tar.gz
                    NOTE: Run this command on all the slaves

Step 17. Start Spark Cluster
         17.1 Start Spark Services
              \$ sbin/start-all.sh
              Note: Run this command on Master
         17.2 Check whether services have been started
              17.2.1 Check daemons on Master
              \$ jps
                 Master
              17.2.2 Check daemons on Slaves
                 \$ jps
                    Worker
```

```
Step 18. Spark Web UI
         18.1 Spark Master UI
               Browse the Spark UI for information about the cluster resources (like CPU, memory),
details of worker nodes, running application, segments, etc.

http://MASTER-IP:8080/

        18.2 Spark application UI
             http://MASTER-IP:4040/

Step 19. Stop the Cluster

        Once all the applications have finished, you can stop the spark services (master and slaves
running on the cluster

        \$ sbin/stop-all.sh
           Note: Run this command on Master
```

Appendix B

Sample SparkR Code

```
#First, load the necessary packages
library(dplyr)
library(knitr)
library(caret)
library(ggplot2)
library(kernlab)
library(randomForest)
library(rpart)
library(SparkR)

#Set spark environment
if (nchar(Sys.getenv("SPARK_HOME")) < 1) {
  Sys.setenv(SPARK_HOME = "/usr/local/spark-2.1.0-bin-hadoop2.7")
}
library(SparkR, lib.loc = c(file.path(Sys.getenv("SPARK_HOME"), "R", "lib")))
sparkR.session(master = "spark://master:7077", sparkConfig = list(spark.driver.memory = "2g"))

#Create a spark context
sc <- sparkR.init(master = "spark://master:7077", appName="Airline")

#Create a sql context
sqlContext <- sparkRSQL.init(sc)

# read ticket observations
Ticket <- read.df(sqlContext, "hdfs://master:9000/input/Ticket_2016_1.csv",
        source = "com.databricks.spark.csv")
dim(Ticket)
summary(Ticket)
head(Ticket)

#craeting Sql table
registerTempTable(Ticket, "Tickettb")

# retain ticket while selecting ItinID and ItinFare
RetainTicket <- sql(sqlContext, "SELECT _c0, _c19 FROM Tickettb WHERE _c19>150 AND _c13=1")
dim(RetainTicket)
head(RetainTicket)

# read coupon observations
Coupon <- read.df(sqlContext, "hdfs://master:9000/input/Coupon_2016_1.csv",
          source = "com.databricks.spark.csv")
#dim(Coupon)
#head(Coupon)
```

```
#craeting Sql table
registerTempTable(Coupon, "Coupontb")

# retain coupon observation while selecting 'ItinID','Origin','Dest','
               OpCarrier','Passengers','TkCarrier','Distance', 'FareClass'
RetainCoupon <- sql(sqlContext, "SELECT _c0,_c9,_c18,_c26,_c27,_c29,_c30, _c31
         FROM Coupontb WHERE _c27==_c28")
dim(RetainCoupon)
#head(RetainCoupon)

train_data <- merge(RetainTicket,RetainCoupon)
dim(train_data)

#train_data <- merge(RetainTicket, RetainCoupon, by="ItinID", all=FALSE)
#train = merge(RetainTicket, RetainCoupon, RetainTicket$ItinID==RetainCoupon$ItinID)
dim(train)
head(train)
summary(train)

columns(train_data)
train_data\$Fare <- ifelse(train_data\$`_c31` > 500,1,0)
head(train_data)
str(train_data)


#gaussianDF <- train_data
#gaussianGLM <- spark.glm(gaussianDF\$Fare~.,gaussianDF,  family = "gaussian")
#summary(gaussianGLM)

model <- glm(Fare ~., train_data, family = "gaussian")
summary(model)

#predictions <- predict(model, testdata)
```