

8-11-2019

Enterprise Data Mining & Machine Learning Framework on Cloud Computing for Investment Platforms

Narasimharao V. Casturi
Georgia State University

Follow this and additional works at: https://scholarworks.gsu.edu/cs_diss

Recommended Citation

Casturi, Narasimharao V., "Enterprise Data Mining & Machine Learning Framework on Cloud Computing for Investment Platforms." Dissertation, Georgia State University, 2019.
https://scholarworks.gsu.edu/cs_diss/150

This Dissertation is brought to you for free and open access by the Department of Computer Science at ScholarWorks @ Georgia State University. It has been accepted for inclusion in Computer Science Dissertations by an authorized administrator of ScholarWorks @ Georgia State University. For more information, please contact scholarworks@gsu.edu.

ENTERPRISE DATA MINING & MACHINE LEARNING FRAMEWORK ON CLOUD
COMPUTING FOR INVESTMENT PLATFORMS

by

NARASIMHARAO CASTURI

Under the Direction of Rajshekhar Sunderraman PhD

ABSTRACT

Machine Learning and Data Mining are two key components in decision making systems which can provide valuable in-sights quickly into huge data set. Turning raw data into meaningful information and converting it into actionable tasks makes organizations profitable and sustain immense competition. In the past decade we saw an increase in Data Mining algorithms and tools for financial market analysis, consumer products, manufacturing, insurance industry, social networks, scientific discoveries and warehousing. With vast amount of data

available for analysis, the traditional tools and techniques are outdated for data analysis and decision support. Organizations are investing considerable amount of resources in the area of Data Mining Frameworks in order to emerge as market leaders. Machine Learning is a natural evolution of Data Mining. The existing Machine Learning techniques rely heavily on the underlying Data Mining techniques in which the Patterns Recognition is an essential component. Building an efficient Data Mining Framework is expensive and usually culminates in multi-year project for the organizations. The organization pay a heavy price for any delay or inefficient Data Mining foundation. In this research, we propose to build a cost effective and efficient Data Mining (DM) and Machine Learning (ML) Framework on cloud computing environment to solve the inherent limitations in the existing design methodologies. The elasticity of the cloud architecture solves the hardware constraint on businesses. Our research is focused on refining and enhancing the current Data Mining frameworks to build an Enterprise Data mining and Machine Learning Framework. Our initial studies and techniques produced very promising results by reducing the existing build time considerably. Our technique of dividing the DM and ML Frameworks into several individual components (5 sub components) which can be reused at several phases of the final enterprise build is efficient and saves operational costs to the organization. Effective Aggregation using selective cuboids and parallel computations using Azure Cloud Services are few of many proposed techniques in our research. Our research produced a nimble, scalable portable architecture for enterprise wide implementation of DM and ML frameworks.

INDEX WORDS: FinTech, Big Data, Data Mining, Machine Learning, Cloud Computing, Enterprise Architecture

ENTERPRISE DATA MINING & MACHINE LEARNING FRAMEWORK ON CLOUD
COMPUTING FOR INVESTMENT PLATFORMS

by

NARASIMHARAO CASTURI

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy
in the College of Arts and Sciences
Georgia State University

2019

Copyright by
Narasimharao Casturi
2019

ENTERPRISE DATA MINING & MACHINE LEARNING FRAMEWORK ON CLOUD
COMPUTING FOR INVESTMENT PLATFORMS

by

NARASIMHARAO CASTURI

Committee Chair: Raj Sunderraman

Committee: Anu Bourgeios
Yanqing Zhang
Sheldon Schiffer

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
May 2019

DEDICATION

This journey of mine wouldn't have been possible, had it not been for many of those, who trusted and cared for me. I dedicate this work to my parents (Mrs. C.V. Padmavathi and Dr. C. Krishna Rao) who continue to be my first teachers and encourage me to always pursue higher education. My academic advisor Dr. Sunderraman whom we call "Sir" is my mentor, philosopher and guide who has always been there to support and guide me through several difficult phases of my research. Without his support I don't think I could have completed this research and be where I am today! Without the support of my wife Usha and my son Shreyas, this achievement would have been a near impossible!! They shared my days, good and bad, equally without quitting on me, always encouraging me saying the end is within sight! They deserve this dedication as they are big part of my life! The sacrifices they had to do to fulfil my dream, is incredible. I am blessed to have each one of you in my life.

ACKNOWLEDGEMENTS

I would like to thank my committee members Dr. Yanqing Zhang, Dr. Anu Bourgeois, and Prof. Sheldon Schiffer for giving their precious time and serving as the review committee members. In spite of their busy schedules, they were kind enough to discuss my research and gave me very valuable feedback which helped to shape this thesis. They are always supportive and gave me time, appropriate feedback and encouraged me at every stage of this research. I thank the entire computer science department faculties and staffs who have educated me and made me eligible to reach this level.

I would like to thank my company Voya Investment Management (Voya IM) in supporting my educational aspirations. My Risk Management team right from beginning gave me the flexibility to do research and to attend, teach classes when necessary. Without the support of my management and senior leaders like Amir Sahibzada, Brad Taylor, Charles Kim and Peter-Paul Hoogbruin I won't be able to complete this quest. Their feedback and support helped me to achieve this milestone. Thanks for my team Owen Joiner and Nirmal Darak for testing my concepts which lead to very successful implementations of ideas.

I would like to thank my colleagues on Mortgage Derivatives desk Dr. Peter Guan, Dr. Derek Chen, Dr. Yingli Zhu and Nikhil Killamsetti for providing the test data sets and believing in me that I can provide a Big Data Data Mining framework and a FinTech solution for better investment decisions. Along with my desk members, I would like to thank Stephen Easton from compliance team in going through my research papers and suggesting necessary edits.

I would like to thank my Voya IM technology teams who helped me to get access to various tools to accomplish and complete the research. Special thanks to Ray Rascher, Lori Kennedy, David Gunter for their help in setting up an environment and providing resource help when I needed for my research. I acknowledge and thank everyone who are current or ex-Voya IM employees who supported and helped me in my research.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	v
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xiv
CHAPTER 1 INTRODUCTION	1
1.1 Data Mining	1
1.2 Machine Learning	5
1.3 Problem Statement	6
<i>1.3.1 Current Problem</i>	<i>6</i>
<i>1.3.2 Data cleaning cost</i>	<i>8</i>
<i>1.3.3 Technical resource cost</i>	<i>8</i>
<i>1.3.4 Fixed cost</i>	<i>8</i>
<i>1.3.5 Expansion cost</i>	<i>8</i>
<i>1.3.6 Machine learning cost</i>	<i>9</i>
<i>1.3.7 Problem Identification Research Focus</i>	<i>9</i>
1.4 Contribution to the solution - Proposed Research	10
1.5 Research Methods	12
1.6 Data Sets & Sources	14
1.7 Technology & Conceptual Design	14
<i>1.7.1 Proposed Conceptual Design</i>	<i>14</i>
CHAPTER 2 AGGREGATION ENGINE	16
2.1 Introduction	16

2.2	Background	18
2.3	Problem Statement and Existing Research	19
	<i>2.3.1 Problem Statement</i>	19
	<i>2.3.2 Existing Research</i>	20
	<i>2.3.3 Proposed Solution</i>	22
2.4	Implementation	24
	<i>2.4.1 Current Architecture - No Aggregation</i>	24
	<i>2.4.2 Proposed Architecture - Selective Cuboids</i>	24
2.5	Results	28
2.6	Conclusion	29
2.7	Future	29
 CHAPTER 3 BIG DATA - CLOUD COMPUTING		30
3.1	Introduction	30
3.2	Background	32
3.3	Problem Statement and Related Research	33
	<i>3.3.1 Current Problem</i>	33
3.4	Proposed Solution	35
	<i>3.4.1 Pre-Migration:</i>	36
	<i>3.4.2 Proposed Implementation Steps:</i>	37
	<i>3.4.3 Post Migration:</i>	38
3.5	Conclusion	42
3.6	Future Work	43
 CHAPTER 4 DATA MINING - PARALLEL COMPUTING		45
4.1	Introduction	45
4.2	Background	47
4.3	Problem Statement and Related Research	49
	<i>4.3.1 Problem Statement</i>	50

4.3.2	<i>Related Work</i>	50
4.3.3	<i>Proposed Solution</i>	53
4.4	Implementation	54
4.4.1	<i>Current Architecture</i>	55
4.4.2	<i>Proposed Architecture</i>	55
4.5	Results	63
4.6	Conclusions	65
4.7	Future Work	67
CHAPTER 5 REPORTING FRAMEWORK		69
5.1	Introduction	69
5.2	Background	72
5.3	Problem Statement	75
5.3.1	<i>Related Work</i>	77
5.3.2	<i>Proposed Solution</i>	80
5.3.3	<i>Current On-Premises set up</i>	82
5.3.4	<i>Cloud Implementation</i>	85
5.4	Observations and Comments	85
5.5	Future Work	87
CHAPTER 6 MACHINE LEARNING FRAMEWORK		88
6.1	Introduction	88
6.2	Background Information	90
6.3	Problem statement	91
6.4	Current Research and Proposed Solution	92
6.4.1	<i>Current Research</i>	92
6.4.2	<i>Proposed Solution</i>	94
6.5	Implementation	95
6.5.1	<i>Azure Machine Learning Studio</i>	95

6.5.2 <i>Azure ML Model Development</i>	97
6.6 Comments and Observations	103
6.7 Future Work	104
CHAPTER 7 FUTURE SCOPE	107
7.1 Introduction	107
7.2 Aggregation	107
7.2.1 <i>Front End</i>	107
7.2.2 <i>Data Retrieval and Reporting</i>	108
7.3 Cloud Computing	108
7.3.1 <i>Data Security on Cloud Environment</i>	108
7.4 Data Mining	109
7.4.1 <i>Techniques</i>	109
7.5 Reporting and Machine Learning Framework	109
7.5.1 <i>Data Access Controls</i>	109
7.5.2 <i>Guidelines</i>	109
CHAPTER 8 CONCLUSION	110
REFERENCES	114

LIST OF TABLES

Table 2.1	Rule Master	25
Table 2.2	Dimension Master	25
Table 2.3	Key Value Master	26
Table 2.4	Fact Data Table	26
Table 4.1	Calc Rule Master	57
Table 4.2	Range Master	57
Table 4.3	Distinct Value Table	58
Table 4.4	Attribute Master	58
Table 4.5	Summary Fact Data Table	59

LIST OF FIGURES

Figure 1.1	Data mining as a step in the process of knowledge discovery.[1]	3
Figure 1.2	High level problem and solution path	9
Figure 1.3	Enterprise Data Mining Architecture with Sub problems .	10
Figure 1.4	Enterprise Data Mining and Machine Learning Framework .	13
Figure 1.5	Proposed Research Methods	13
Figure 1.6	Proposed Data Mining Framework - Aggregation Engine .	14
Figure 2.1	Security Level Analytics	18
Figure 2.2	Fund Summary	19
Figure 2.3	Data Cube Visual	21
Figure 2.4	Aggregation Run Result Observations	28
Figure 3.1	MBS Pool Structure	32
Figure 3.2	Big Data - 3 Vs	34
Figure 3.3	Proposed Architecture on Cloud	36
Figure 3.4	Azure Cloud Dashboard	37
Figure 3.5	Azure SQL Linked Server Details	38
Figure 3.6	Azure SQL Dashboard layout	39
Figure 3.7	Azure SQL Client Connection Information	40
Figure 3.8	Azure SQL Activity Monitor	41
Figure 3.9	Azure SQL Migration Criteria	42
Figure 3.10	Azure SQL Migration - Implementation Time Chart	43
Figure 4.1	High-level Mortgage Backed Securities life cycle	47
Figure 4.2	Big Data categorization diagram	48
Figure 4.3	Example of a pools of loans in an MBS pass through security	48
Figure 4.4	Brief description of major services by Cloud Computing providers	52

Figure 4.5	Proposed Distributed Calculation on Data Copies (SQL Stage Tables)	53
Figure 4.6	Proposed Architecture (Parallel computational execution on Table 1..n)	56
Figure 4.7	Process Steps for an Individual Raw Factor File	61
Figure 4.8	Parallel Execution of SSIS Package with the Node Attached Calculation	62
Figure 4.9	Process Steps for an Individual Raw Factor File	62
Figure 4.10	SQL Stored Procedure with parameters	62
Figure 4.11	Various Scenarios to test to propose a best-fit for organization	64
Figure 4.12	Microsoft vCore Model to match on-premises model	66
Figure 5.1	Organizational Reporting Requirements	73
Figure 5.2	Organizational Reporting Requirements	74
Figure 5.3	Reporting and Data Mining Framework Evolution	76
Figure 5.4	Magic Quadrant for Analytics and Business Intelligence Platforms	79
Figure 5.5	Proposed Reporting Framework - Need based Model	81
Figure 5.6	On-Premises implementation model	83
Figure 5.7	On-Premises user categorization	84
Figure 5.8	Tools - Data Source Recommendation	86
Figure 6.1	Organizational ML Workflow	93
Figure 6.2	The Azure Machine Learning Studio Setup	96
Figure 6.3	The Azure ML Studio POC Experiment Set up	97
Figure 6.4	The Azure ML Studio Model Flowchart	98
Figure 6.5	Azure SQL Database as Data Source	99
Figure 6.6	Azure ML Data Split and Algorithm-1	100
Figure 6.7	Azure ML Data Split and Algorithm-2	101
Figure 6.8	Azure ML Field Selection	102

Figure 6.9	Azure ML Area Under the Curve AUC	102
Figure 6.10	Azure ML AUC Observed	103

LIST OF ABBREVIATIONS

- AI - Artificial Intelligence
- BI - Business Intelligence
- BM - Benchmark
- DM - Data Mart
- EFFDATE - Effective Date of the Report
- PORT - Portfolio
- ML - Machine Learning

CHAPTER 1

INTRODUCTION

1.1 Data Mining

Data Mining became a natural extension [1] of the Information Technology. Data Mining is extraction of meaningful information which can be used in any decision making from a set of data points collected over a time period. We can also define the Data Mining as discovery of model(s) from a huge data set(s). The growth of data[2] has exploded in the past 20 years to a point where the old techniques of processing and accessing the data to convert it to meaningful information has become a problem in and of itself. The legacy applications which render the data to the end users are phased out as their original goal was to simply just retrieve the data and present it to the user. The evolution of the database concepts and various Data Base Management Systems (DBMS) complemented the growth of Data Mining techniques. The gathering of data by itself is not much help for the growth of an organization or an institution. The general progression of deriving intelligent information from the data can be seen in 4 distinctive steps.

1. Data Collection
2. Data Exposure
3. Decision Support
4. Predictive Analysis

The first three steps are well developed but there are several inherent issues of how we source, store and retrieve the data we need for analysis purpose. On the other hand the Predictive Analysis is the one where new methods and models are emerging. By understanding the different patterns of the data, a company can enhance its ability to grow in a

competitive environment. The foundation for the Predictive Analysis comes from the ability to understand the data. The first three steps mentioned earlier give a solid base for the Predictive Analysis on which strategic decisions can be made. The implementation of the first three steps (Data Collection, Data Exposure, and Decision Support) can be expensive and can be a major hindrance to implement a successful Predictive Analysis Framework (PAF). The studies suggest [3] that many corporations spend 50% to 80% of their time and effort cleaning data to prepare it for the Predictive Analysis.

The next [2] major cost is learning the patterns of the data presented. This can be described as pattern recognition. There are various [4] algorithms currently in use to categorize and derive a pattern from the input data. Predictive Analysis heavily depends on the pattern recognition that a machine can show when presented with the data set. This type of technique is usually called Machine Learning.

The general flow of data from start to finish till it is used by the end users take a spiral step as shown in the 1.1 on page 3. The critical component for a successful Data Mining Framework (DMF) depends on several layers of intermediate steps. To have a clean and reliable DMF we start with the source data. Understanding the data in the initial stages will help in getting better results in setting up closely suited Patterns for the problem under consideration. The progression from Data to Information or Knowledge takes several iterations and this process can be very expensive if the right techniques and strategies are not deployed. Usually the failure of DMF is not due to lack of technically qualified teams but lack of direction from the senior management. Creating a DMF for an organization needs a lot of commitment from the senior management. The approach we will take is a bottoms up approach.

The challenge every organization faces is the right technique and approach to build a successful DMF. There is no single silver bullet to solve this problem. Each organization

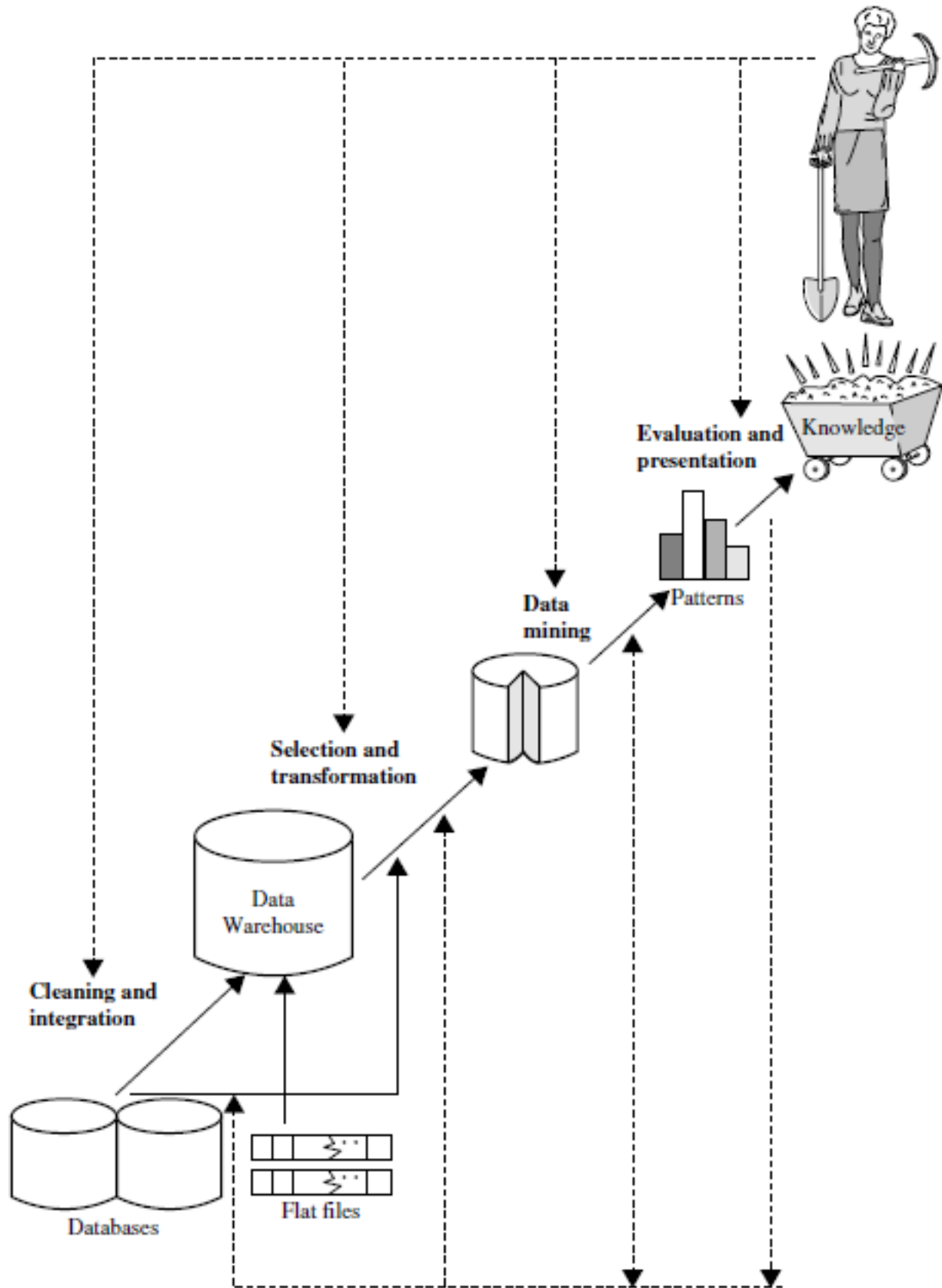


Figure (1.1) Data mining as a step in the process of knowledge discovery.[1]

is unique. There are 2 major methodologies available. One is Top Down and the other one is called Bottoms Up. The example of top down approach is that senior management may decide to get more relevant matrix for the expansion of the organization and that can trickle down to the lowest level of the department employees. In a Bottoms Up approach the change or need to have a DMF comes from the end users creating a need for a higher level DMF in an organization. Both the approaches have their pro and cons but it is critical to understand the source data in order to build any successful DMF.

There are several types of data we can use to understand various patterns. The question is what types of data can be mined to find meaningful information. The answer is we can categorize the data into 2 distinct sets. The first one is the descriptive and the second one is the predictive. The descriptive data mining reflects to characterize the properties of the data in the given target data set. The predictive data mining reflects the tasks performed in order to come up with predictive models. These 2 are different tasks and will have different paths of traverse. The end users can define the functionality that they are more interested in depending on the target data set as well as the institutional need. An organization involved in selling product line to customers would like to mine the sales data to find what types of products are popular in demography. This analysis will help the company to focus the advertisement budget in the areas where there is a higher return on their investment. The application of the data mining can be very helpful in financial industry. In financial industry the data is usually related to the market conditions and various factors which will influence the outcome of an investment. For an example if we invest in a stock of a company the predictive price of the stock can be modeled by mining historical information on the specific stock. The statistical model can be used on the target data set to find various characteristics of the information buried in the subject.

The usual Data Mining research focuses on 5 major areas.

1. Mining methodologies

2. User interface
3. Efficiency & Scalability
4. Diversity of data types [5]
5. Social impact

Data mining methodologies can be viewed as mining new information methods, mining various multidimensional data model, interdisciplinary data mining methods, pattern evaluation methods. The research is also in the areas of machine learning which is used once we identify the mining category goals. The category goals depend on the various user interests. The User Interface drives a very rich research area of how to present the various mining rules and to leverage the users knowledge into a machine learning exercise. The Machine Learning is the benefactor of this as the more users can teach the machine the pattern recognition can be rich and will be close to the real word experience. Machine Learning is a domain area where the science learns how to teach a computer system. The main area of research is to investigate how a computer program can automatically learn to recognize complex patterns and make intuitive and intelligent decisions based on the data presented.

1.2 Machine Learning

The machine learning is highly related to data mining. There are few classifications views of machine learning.

Supervised learning is a technique to enhance the learning capability of a computer system or program to able to learn from data and fact pair. The learning technique is also called classification learning. In this technique the computer program will be taught with a training data set which consists of the raw data point and the representation of the data point as an outcome Example is to interpret human voice and reproduce the same in a written text format. In this case the computer program is presented with various variations of the sound patterns and the equivalent text output. During the diction phase the computer

program will identify the word and will map it back to the learned output data set which is the text conversion of the voice. This is widely used in dictation and also in crime detection. The same training data set can be extended with various other attributes like age, stress level when a word is spoken differently under different scenario. The age as an attribute, the computer program can identify and classify the human being into various categories like age between 23 and 35 and taking under stress etc.

In **unsupervised learning** is a way to cluster the target data to learn and discovery the underlying patterns. There is no mapping of the data set to an output to be labeled by the computer program. The absence of the training data will make the computer program to use different methods of clustering the information to subject for further mining methods.

1.3 Problem Statement

1.3.1 Current Problem

During our preliminary research what we found there is a gap in data mining applications, tools and frameworks in financial industry. These tools or applications built to solve a specific problem and those tools are very expensive and propriety. The vendor solutions take several years for implementation and can lead to a huge cost to the organization. There are very few options for a financial company to utilize any open source data mining and machine learning framework. The existing research either solves a specific problem or needs very extensive rework of the existing organizational data stores.

Purpose of our research is to design and develop cost effective and an efficient enterprise framework on cloud computing environment for data mining and machine learning using a rule based aggregation engine along with parallel processing of user defined financial calculations to enable predictive analysis and knowledge discovery with a business intelligence reporting capability.

Currently we have efficient way of gathering and processing data from various data sources but in financial industry there is a big gap in open source architecture in terms of data processing, data mining and machine learning on cloud based computing architecture. As we noticed in the prior section of related research, the costs quickly add up to build a data mining tool or tools in an organization as the data sources grow. It will be very helpful to the research and industry to have an open source framework which can solve the data processing, aggregation at various user defined levels, data mining and predictive analytic and reporting based on machine learning framework.

The intended research is more focused towards the minimizing the cost to implement a Data Mining Framework (DMF) by maximizing the available tools (Existing Data Sources, Algorithms etc.) effectively with limited technical resources. The proposed research will try to address the problem of Technical Resource Pools by implementing efficient DMF by developing road maps for a User Defined Rule Based Data Mining Engine and extend the framework for Machine Learning (ML) to able to predict certain events from the data sets.

The cost to build an efficient Data Mining Framework (DMF) for a small to mid-size organization requires a lot of funding capital. The traditional Data Warehouse method can be very time-consuming due to the very nature of data warehouse development. These projects are multi-year and some can even be multi-generational. The costs can go up in acquiring the technical resources needed to build and maintain the DMF along with the various enhancements needed for the future expansion of the company product line and businesses. The major areas of hurdles we need to cross are as following.

1. Data cleaning cost
2. Technical resource cost

3. Fixed cost
4. Expansion cost
5. Machine learning cost

1.3.2 Data cleaning cost

The data used for the predictive analysis should be clean enough to get the closed solution to the problem we are trying to answer. As mentioned in my initial introduction, the cost of cleaning is the biggest item in building the Data Mining Framework.

1.3.3 Technical resource cost

Recruiting and retaining technical experts [1] to build an efficient data mining framework involves cost. The first step, which is to collect the data and to have the data in a format the end user can use, requires expensive technical resources to write and modify the raw data into the final usable format to utilize for decision making. Acquiring that resource pool and maintaining the high level of technical resources will cost the company.

1.3.4 Fixed cost

Cost can be derived from the human resources and also from the hardware / software costs. The cost for human resources can be linked to the first point and the software/hardware cost can be part two of the problem. Many companies usually go for the Data Warehouse models and that itself is a huge undertaking framework for any organization.

1.3.5 Expansion cost

The growth of a company should not depend on the system capabilities of that company. Usually, this does not happen. The growth should correlate to the system capability as the

company grows into different areas or acquires other firms. The ability to organize the new assets or products poses a big challenge to the existing system teams.

1.3.6 Machine learning cost

The biggest area of Data Mining encompasses the ability to teach a machine to identify the patterns visible in data sets and be able to predict a solution path. This involves a variety of Training Data Sets deployed during the teaching phase so the machine can effectively interpret any new data set presented to it and also deduce a meaningful result set. This result set can then be used for any kind of predictive analysis. However, that endeavor requires huge amounts of data sets, resources, and time for the analysis of the outcome, and with that comes the deviation from the real observation to the machine predicted solution.

1.3.7 Problem Identification Research Focus

The research focus is clearly not on the fixed costs or expansion costs ***but on the reusable framework development cost***, which, in other terms, would help reduce the in-house technical costs for a company. The proposed research will try to understand the impacts of the quality of the data when used for predictive analysis. The quality can impact the efficiency of Machine Learning techniques and algorithms when used in a Data Mining Framework.



Figure (1.2) High level problem and solution path

The entire research which was shown at a high level in the figure 1.2 on page 9 is divided into 5 sub components and data mining and parallel computing can be show as two rather than one component. Each component solves a specific problem in our overall research problem. By solving the sub components and bring them together we can solve the overall problem of **Efficient Enterprise Data mining and Machine Learning Framework**. At high level the components are show in the figure 1.3 on page 10.

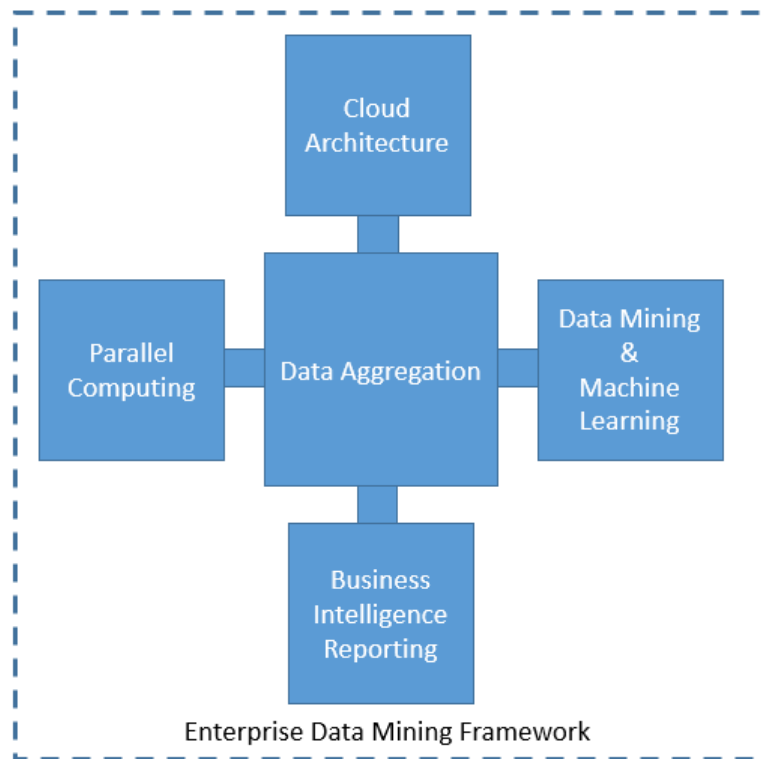


Figure (1.3) Enterprise Data Mining Architecture with Sub problems

1.4 Contribution to the solution - Proposed Research

To solve the problems mentioned above (Data cleaning, data mining, expansion and Ma-

chine Learning) we would like to propose a simpler, more cost-effective data mining model on cloud computing technologies. The data attributes can be of any class (Student, Product, Sales, Retail, Analytical or Financial). The proposed architecture should be able to handle a user-interface-based rule defining engine that can generate aggregated data, depth views, and decision trees for mining purposes. This can be done by using existing data systems the company currently has and building a framework around the existing models to improve data mining capabilities, with marginal cost to the company. The proposed research will also focus on the machine learning algorithms and build a predictive analytical framework using the industry standard cloud computing services [?] like (Microsoft Azure, Amazon Web Services etc.) **The current problem can be divided into 5 parts.**

- *The first part is to develop a rule based aggregation engine (Analytical Cubes, Data Marts) to pipeline the source data in a form that the data mining engine can use for decision making.*
- *The second step of the process is to identify a cloud computing service provider and migrate the existing solution of aggregation engine to build the base for the next generation data mining framework*
- *Next step of the process is to build a data load process to utilize the big data capabilities on cloud computing with the parallel calculation engine*
- *The fourth phase of the research problem will be to build the data mining engine itself, using pattern recognition and existing data mining algorithms which are available within the cloud computing platform*
- *The fifth step is to build a user friendly front-end to efficiently manage the aggregation and mining rules by end users*

Solving the first part of the problem RBAE (Rule Based Aggregation Engine) benefits the company tremendously as they can implement the solution with ease and prepare the

ground for the data mining applications.

My proposal is to develop a framework to solve the problem of RBAE, and also, to provide a data mining facility which can be deployed on any structured data set in the industry. During this process, the various industry techniques of Machine Learning will be analyzed and incorporated into the framework to sustain suitability and re-usability.

The process outline and steps are shown in the 1.4 on page 13. The evolution of the problem identification and solution implementation will proceed as an incremental steps rather than a big bang approach. Each step will be base for next iteration of the solution. Example the first step of building the aggregation engine will serve as the basis for the next phase of utilizing the cloud computing architecture. Once the initial solution is on cloud, that will serve as the launch pad for the next step of parallel computing. When the parallel computing solution is implemented, that will form the base for the data mining and machine learning.

The research will focus on the existing data mining techniques and algorithms which are available in the industry and in academia. These techniques will be studied and proposals will be made to enhance and extend them wherever it is needed. There is a good amount of research in the field of data mining pertaining to consumer industries, but little available pertaining to financial and educational sectors. There is a bigger need for this kind of research in financial and educational areas.

1.5 Research Methods

My research will follow the following methods/steps.

1. Feasibility study of the topic (Data Mining, Cloud computing, Aggregation Engines)
2. Literature Survey on the relevant topic of research
3. Proof of Concept using a sample data set.
4. Model development

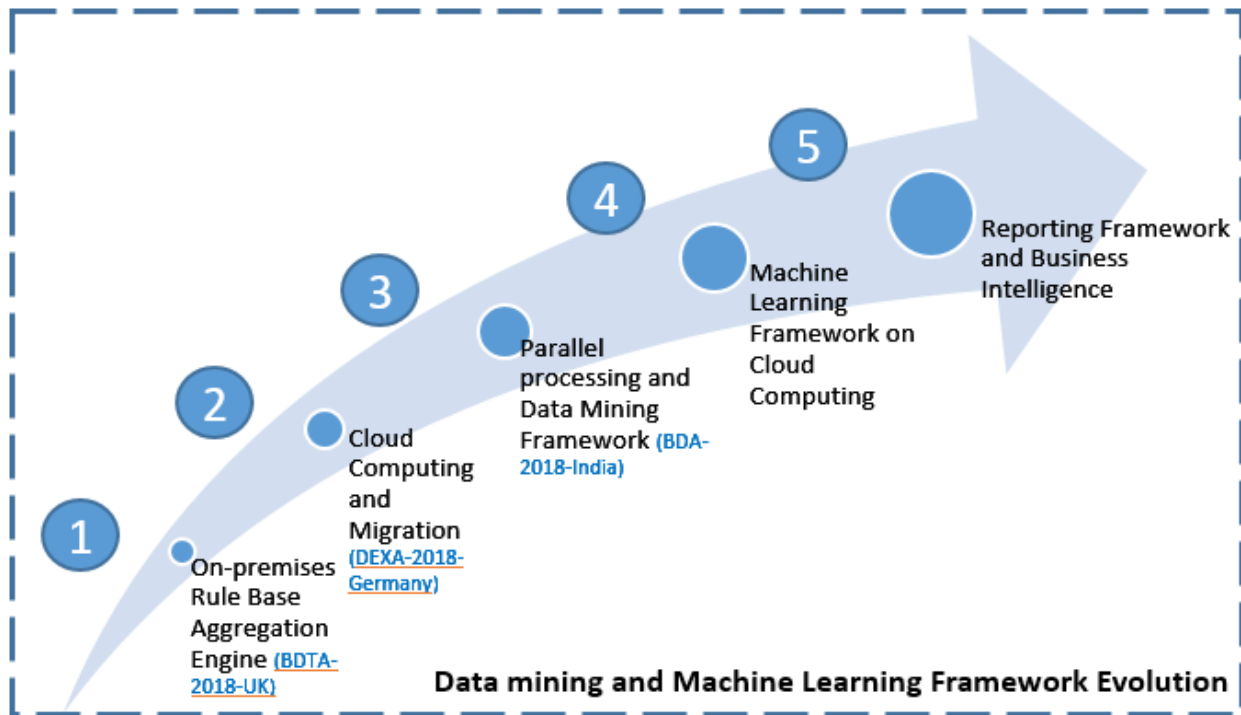


Figure (1.4) Enterprise Data Mining and Machine Learning Framework

5. Simulation of a real world experiment (Fixed Income Investments)
6. Proposal analysis and Conclusion

The research time lines were mentioned at the end of this proposal. The original research started two years back when the problem was identified. The areas of where we want to investigate to come up and identify the research problem and also to make sure there are no equivalent applications or frameworks readily available as open-source or freeware in the industry or in the academia. The process is shown in the figure 1.5 on page 13.



Figure (1.5) Proposed Research Methods

1.6 Data Sets & Sources

During my research, I will be using a financial data set which will be the main data set going through all the iteration steps from start to end. This data set is for Fixed Income bonds which consists of corporate issued debt, Mortgage bonds and US Treasury instruments. This data set will be mocked up to eliminate any sensitive institutional information. Apart from this data set I will incorporate a mocked up student data set to mimic the financial data set to evaluate the final framework.

1.7 Technology & Conceptual Design

1.7.1 Proposed Conceptual Design

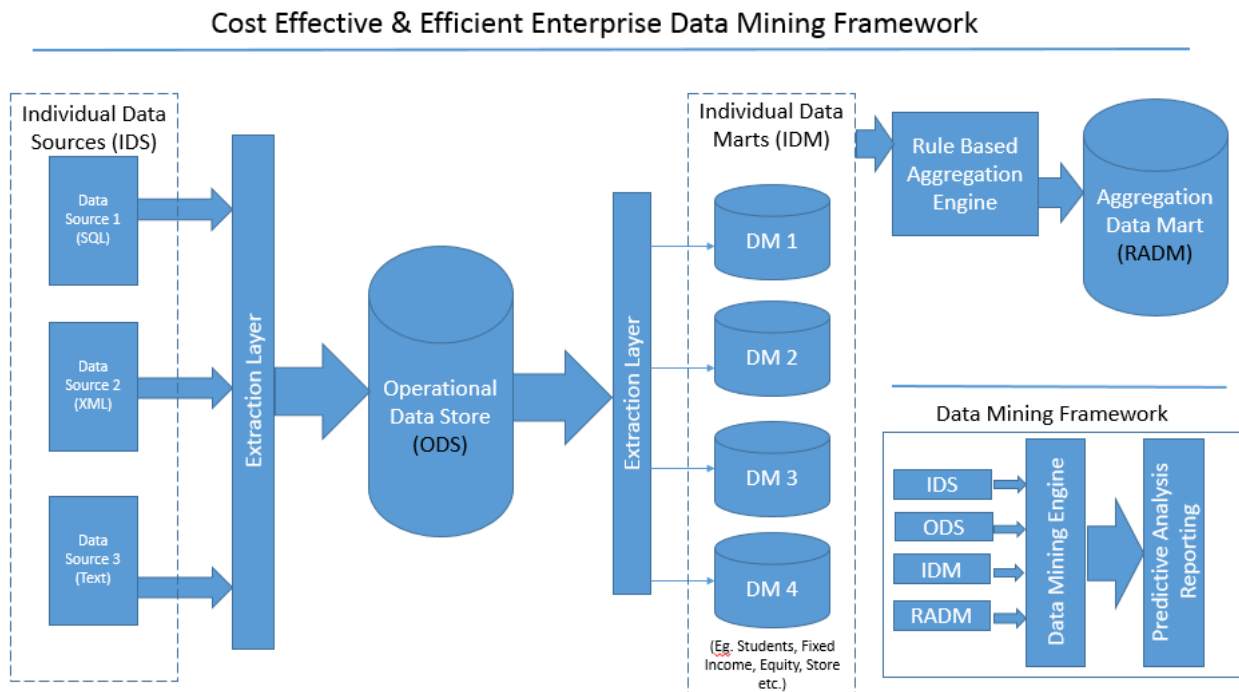


Figure (1.6) Proposed Data Mining Framework - Aggregation Engine

During my research, I will be exploring some of the problems we can come across in

building a Cube structure and the expandability of the Cube structure when a new set of aggregations are needed. Along with the aggregation frameworks, the focus will be on designing a flexible Data Mining Engine using the existing sets.

The Technology stack can consist of RDBMS, Statistical tools, and programming languages that can be used to implement a solution. The following figure shows a prototype model for the aggregation engine which will be the center for the data mining framework. The initial design and development is focused on Microsoft technology stack and will be able to replicate on any ANSI SQL to keep it platform independent.

In the coming chapters we will expand the initial high-level proposed architecture to give a better understanding of each individual sub problem.

CHAPTER 2

AGGREGATION ENGINE

2.1 Introduction

Recent developments in Big Data in financial industry has created a huge opportunity for design and development of effective aggregation (higher level) analytical measures (Fund, Portfolio, Sector, Industry etc.). Lack of these aggregated measures will jeopardize organizations ability to provide the financial services promised to clients. Vendor solutions and existing academic research (Data Cube, OLAP) can provide these aggregated measures but are expensive, time consuming and not practical to implement for a small to mid-size investment organization. Our proposed solution using rule-based architecture is cost effective, efficient and building block for Rapid Application and Decision Support Systems on Big Data. Our new approach *Selective Dimensional Cuboids* provides a simple but robust solution with flexibility for future expansion into data mining, portfolio trend analysis and cycle forecasting. The solution is easily portable to any dimensional data set. High-level (aggregated) portfolio characteristics are key measures, which will help a fund Portfolio Manager (PM) to monitor and manage an investment portfolio. These measures will also help Risk Management in avoiding surprises and protect our client investments. An Investment portfolio consists of financial assets, which can be, traded (Bonds, Stocks etc.). To manage our client portfolios effectively, the aggregated measures (Duration, Yield, Effective Credit Rating, Counterparty risk etc.) play a crucial role [6]. Our daily investment portfolio information falls under a broad definition of Big Data [2] with the 3Vs (Velocity, Volume and Variety). The price of stocks, bonds are constantly changing depending on the market conditions. Multiple funds, portfolios and Fund of Funds data at the lowest level quickly builds the volume dimension of the Big Data and the various measures which are unique to each instrument make the variety of the Big Data. The build of a data warehouse is a prerequisite

to host the various sources of the huge data set(s) we receive from various vendors at granular level. Our Data Warehouse (DW) is designed and built to provide a foundation of our Big Data as a data store and to launch our next generation Big Data Investment Application Framework (BDIAF). Currently our data set is around 5 TB and constantly growing on daily basis. The current form of the DW is flat or denormalized format to match True to the Source model. Lack of aggregated measures in the DW sets stage for our BDIAF to be built on an efficient aggregation engine on user defined dimensions at various levels. The existing vendor solutions and tools for aggregation of the financial analytics are expensive in terms of cost and implementation time and typically divert Business Resources (BR) and takes our subject matter experts (SME) away from their day to day Portfolios Management and Risk Management responsibilities.

After evaluating various potential solutions (Vendor and Academic), we propose a cost effective, easy to implement and flexible Big Data Aggregation Engine, based on user defined aggregation rule-based architecture to accomplish data aggregation, and address storage issues. Our solution not only saves capital costs and resources, but also gives a solid foundation for our next generation Data Mining and Machine Learning projects. ***The proposed solution is easy to maintain (3 Dimension tables and 1 Fact Table) and there is no additional overhead on our existing DW operations.*** Our solution provides selective dimensional cuboid aggregation framework defined by users and avoids full-materialization of all cuboids. We will attempt to implement our Aggregation Engine on our existing data warehouse.

The current chapter is organized into seven sections. Section 2 introduces the preliminary definitions and background information. Section 3 highlights the problem, presents related work done on aggregation methods in academia, and introduces our proposed solution. Section 4 presents the implementation of our solution over the current data warehouse architecture. Section 5 compares the results observed from the existing ad-hoc queries and the proposed solution. Section 6 describes our conclusions and Section 7 gives our future works including Data Mining initiatives on Big Data.

2.2 Background

An investment portfolio consists of Fixed Income Bonds [7] and/or Equity securities. Fund is made up of investment portfolios. Borrowers issue the bond with a fixed coupon, maturity and a par value. Investors buy the bond providing capital for borrower. Borrower pays a fixed interest rate (coupon) to the investors. Investors can hold the bond until maturity or can trade in secondary market. The nature of the fixed income bonds can be classified in several ways by their security attributes. A security attribute is value depending on a specific quality of the security. For example, coupon rate will distinguish between a fixed interest rate bond and floating-rate bond. In-depth financial overview is out of scope of this paper.

The aggregation measure of a fund or portfolio depends on the individual securities that fund or portfolio consists of. Example of an analytical measure is bonds Duration which is an approximate measure of sensitivity to changes in interest rates. Figure 2.1 on page 18 shows security level analytics. The aggregated analytics of a fund are calculated as weighted average by MV or any other denominator. E.g. Duration of Fund1 will be $\text{cusip}(\text{Duration} * \text{MV}) / (\text{fund MV})$. This is shown in figure 2.2 on page 19 as aggregated measure. On the same lines we can calculate OAS and any other analytical measures

Report Date	Fund	Portfolio	Cusip	Security Type	PAR (m)	MV (m)	Maturity	Coupon	Duration	OAS	Convexity
11/16/2017	Fund1	Port - A	Cusip1	Investment Grade	\$ 1,000	\$ 1,321	3/31/2019	5.5	3.67	123	1.2
11/16/2017	Fund1	Port - A	Cusip2	Agency Mortgage	\$ 1,000	\$ 1,200	6/30/2021	4.5	4.5	43	0.01
11/16/2017	Fund1	Port - B	Cusip3	High Yield	\$ 2,000	\$ 568	3/31/2019	7.5	7.5	345	0.031
11/16/2017	Fund1	Port - B	Cusip4	Asset Backed	\$ 1,000	\$ 968	8/31/2029	3	3.21	87	0.31
11/16/2017	Fund2	Port - B	Cusip5	Investment Grade	\$ 1,000	\$ 790	6/30/2025	4.5	4.1	123	0.22
11/16/2017	Fund2	Port - B	Cusip6	Investment Grade	\$ 1,000	\$ 968	1/31/2027	5	6.7	123	2.23

Figure (2.1) Security Level Analytics

Report Date	Fund	MV (m)	Duration	OAS
11/16/2017	Fund1	\$ 4,057	4.3418	122
11/16/2017	Fund2	\$ 1,758	5.5317	123

Figure (2.2) Fund Summary

2.3 Problem Statement and Existing Research

2.3.1 Problem Statement

Currently there are no pre-aggregated analytical measures available for users in our DW. The ability to aggregate effectively by various required measures in our existing DW is a challenge due to the size of the data and the number of dimensions we have. The users (Portfolio Manager, Risk Managers and Asset Managers) use on-the-fly queries to aggregate the data at various levels. For example, if we need to see a port-folio duration over a period of time then users use the following SQL Query.

```

/* SQL for a simple Duration Aggregation at Report Date and Portfolio */
SELECT      REPORT_DATE, PORTFOLIO, SUM (MV*DUR) / SUM (MV) AS DUR
FROM        dbo.TBL_POSITION
WHERE       REPORT_DATE > '01/01/2017'
GROUP BY   REPORT_DATE, PORTFOLIO

```

The current DW architecture can not be modified. Pre-computing of the data cubes for all the dimensions (Full Cube or Full Materialization) is not practical and any Selective Data Cube (Iceberg Cube or Partial Materialization) will not be possible on a threshold condition. Every cuboid is important and can not effort to create only selective Iceberg Cubes [2] from a full cube for a given dimension.

The issue is not building a data cube but the problem we want to solve is to capture the user defined aggregation rules and create only the needed cuboids. Our contribution is to enhance the selective data cubes by using very simple and easy to maintain dimension

and fact tables for any relational database architecture [8].

A simple analogy to our approach is ordering food what we need at a fast food place. If customer needs only fries and milk shake, they should be able to order them without ordering any extra food (meal package which may have other food items). Similarly, if our users request aggregation at Team, Country, Issuer they should be able to set that rule or request rather than DW aggregating all the unnecessary levels Team, Country, Team, Issuer, Country, Issuer etc.

2.3.2 Existing Research

Researchers studied summarization or aggregation extensively. There are various efficient algorithms to solve the aggregation problem by creating Data Cubes [9]. The Data Cube architecture depends on aggregation of the measure by a given dimension. A data cube allows data to be viewed in multiple dimensions. A data cube is defined by its dimensions and facts, which each dimension represents. Dimensions are the perspectives or entities respective to the organizational data structure. In our example, REPORT DATE is a dimension and PORTFOLIO is another dimension. Data Cube, in simple terms, is a multidimensional data store. The actual physical storage of such data may differ from its logical representation. Usually we think of cubes as 3-D geometric shapes, in data warehouse the data cube is n-dimensional. As shown in the figure 2.3 on page 21 is a simple representation of a 4-D Investment Portfolio Cube with cuboids. The total number of cuboids needed for 'n' dimensions are given by the formula shown in the equation 2.1. Where L_i is the number of levels associated with dimension i . One is added to L_i to include the virtual top level, all.

$$TotalCuboids = \prod_{i=1}^n (L_i + 1) \quad (2.1)$$

If the cube has 10 dimensions and each dimension has five levels including all, the total number of cuboids that can be generated is $5^{10} = 9.8 \times 10^6$. [1]. The size of the cuboid depends on the dimensions distinct values (cardinality). The Data Cube creation with the existing algorithms and techniques cannot be applied due to the expensive storage space and

computational intensive operations.

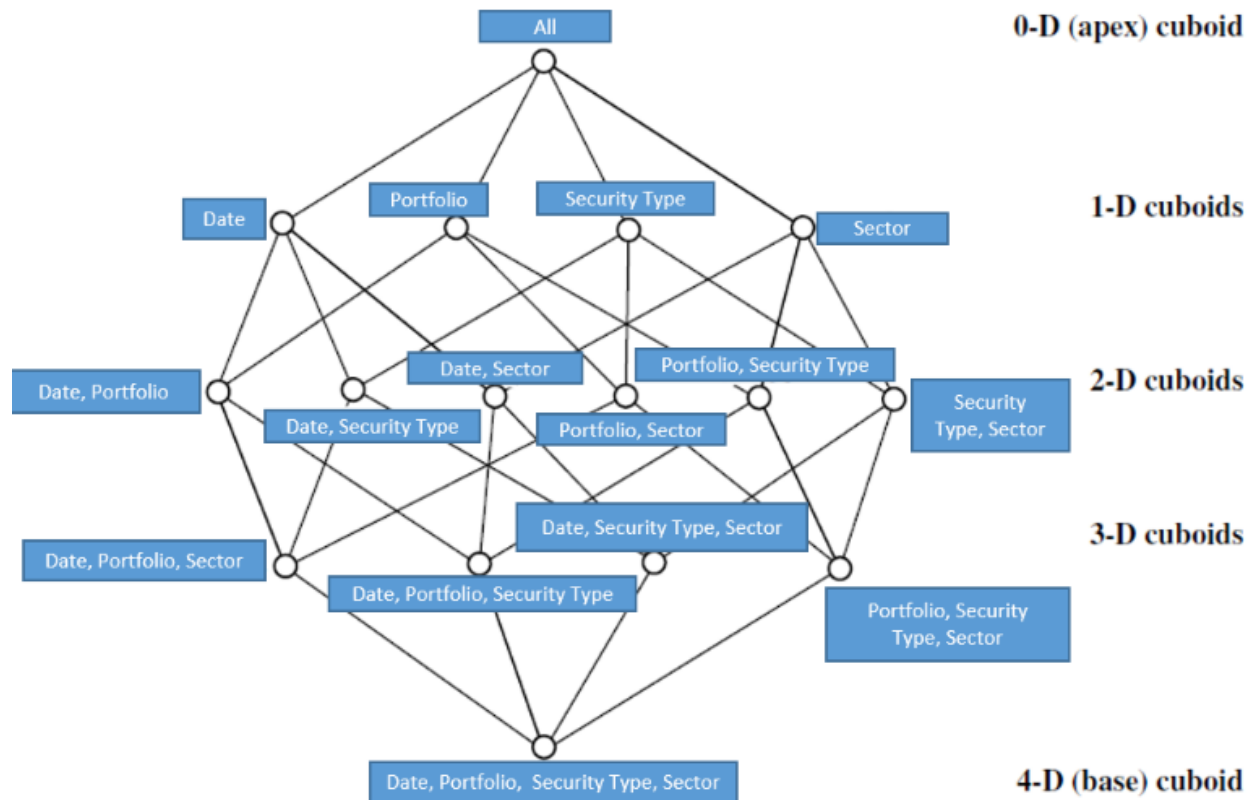


Figure (2.3) Data Cube Visual

There are several industry standard vendor tools [10], which can create aggregation sets, but they need huge investment in-terms of technology, human resources and take long time to implement. Microsofts SQL Server Analytical Services [4] is one such tool which can help in created data cubes, but implementation is very long drawn and not flexible for our usage which depends on dynamic user defined aggregation rules. The current research is mainly focused on a various automated Full (Multiway Array, Star-Cubing Aggregation) or Partial materialization depending on Apriori Algorithms [1], [2] methods but not on selective data cubing. The few selective algorithms are developed by deterministic algorithms [11] rather than user defined selection. Our contribution to the paper will be the application and implementation of user defined rule-based aggregation for any dimensional data set on conventional or on Big Data frameworks using rule-based architecture. This implementation

is portable and can be applied for other Bid Data Initiatives where aggregation is needed for further Data Mining activities. The other area we can use our approach is anomaly detection in financial data [12] which needs a lot of computations [13] which we can reduce by our method of selective aggregation. There is extensive research done on data materialization and selected computation of cuboids. We have three types of materialization of data cube based on the base cuboid. We researched three major materialization techniques widely used in data warehousing and data mining applications.

a) No materialization in which we don't compute any non base cuboid. This leads to expensive multidimensional aggregates on the fly, which is extremely slow in our case.

b) Full materialization in which we compute all cuboids. We could not implement as this is time consuming and impractical for our data set and user needs as discussed in earlier on our dimensions and cardinality.

c) Partial materialization in which users will select the cuboids, which they would like to see on a daily basis. This has its advantages as the subject matter experts are involved to pick the most frequently used data sets for their decision-making. These selective cuboids are also called subcubes. The other variation of partial materialization is shell cube. This method involves precompute the cuboids for only a small number of dimensions usually three to five of the data cube. Queries [14] on the additional combinations of dimensions can be computed on the fly.

2.3.3 Proposed Solution

We propose an implementation solution to Big Data Analytical Aggregation by providing a Rules-Based Analytical Aggregation Engine. Rules are set up by users and the aggregation engine will use them in building a dynamic SQL Aggregation rule to be materialize. The proposed solution will allow the user to set up the base cuboid only. This is super selective technique which only a subject matter expert will be able to define the rule. As part of daily DW population we initiate our Big Data Aggregation Engine (BDAE) which will extract and generate an executable SQL and results will be materialized (saved)

to a Dimensional Fact table. This is a daily process which stores the aggregated daily data. The daily aggregated data is used for historical re-reporting and will solves the long running ad-hoc user request of a cuboid query.

An aggregation rule is defined as a set of dimensions aggregated over a set of measures with a support condition. Equation (1) is the definition of a rule with a simple example. Also, a simple Relational Algebra (RA) notation in equation (2).

$$RuleR(i) = (D(1), D(2)..D(i), ..D(n))Aggregate(m_{(1)}, m_{(2)}..m_{(i)}..m_{(n)}withS_{(support)}) \quad (2.2)$$

Example: R(1) = Manager, Country, Issuer Over MV,Duration mv <> 0

SQL Interpretation :

```
SELECT  MANAGER, COUNTRY, ISSUER, SUM (MV) MV, SUM (MV*DURATION) / SUM (MV)
FROM    DBO.DWView
GROUP BY REPORT_DATE = 11/01/2017 WHERE MV <> 0
```

Our aggregation engine will interpret the rule at a higher level as a RA as shown in the equation (2) where DW is a composite view of the DW tables and , and are the RA operators for select and grouping attributes dim1dim(n) is the user set up cuboid.

$$RA = dim_1, ..dim_{(n)}(\sigma_{supportDW < dim_1, ..dim_{(n)} > \tau Wtg.Avg.(Measures)(DW)) \quad (2.3)$$

As an example if user is requesting the aggregation for Manager, Issuer, Sector level there is no need to calculate the sub cuboids Manager,Issuer,Sector or Manager, Sector,Manager, Issuer,Issuer, Sector. If there are n-dimensions, we can avoid (2n -1) cuboids by giving users the ability to set up the necessary aggregations on our Big Data. The reduction of the (2n -1) calculations is very significant in terms of computational time and storage space. The rule can be set up by users using a simple front end by selecting dimensions over measures with support condition. Our proposed and implemented aggregation engine will extract the rule and then build dynamic SQL Aggregation Rule will perform the calculation which then be materialized to the database. Our contribution to the paper is enhancing the Selective

Dimensional Selection by rule-based framework to reduce the number of calculations by $(2n - 1)$ over n -dimensions. Our method is simple to understand, implement and maintain and enables data mining opportunities on our Big Data.

2.4 Implementation

2.4.1 Current Architecture - No Aggregation

In our current DW we have a set of historical relational tables which host the Big Data in terms of five major relations (tables). The five tables are 1) Fund Data 2) Security Data 3) Holding Data 4) Analytical Data 5) Issuer Data. All the five tables can be joined by the key attributes (fund id, security alias, and issuer id). We can derive any level of aggregation using a DW View (DWV) which consists of the main five tables. Users are provided with various querying tools (BI, SQL etc.) to run ah-hoc queries but the result of these queries are not materialized on database. Same report should be rerun again once the query window or report window is closed. Most of our dimensions in our DW are not indexed which is a major hurdle in running an historical report on a non-indexed attribute. By adding more indices, it increases the DW load time very significantly. Indexing all the dimensions is not possible. Hence the long run times for the ad-hoc queries.

2.4.2 Proposed Architecture - Selective Cuboids

The proposed database schema is to extend the current DW scheme with a new set of tables (Dimensions, Fact) which will support the proposed Aggregation Engine. The proposed schema for Big Data Aggregation Engine (BDAE) will have four tables.

- Aggregation Rule Master
- Dimension Master
- Key Value Master
- Aggregation Fact Data.

Table (2.1) Rule Master

Agg ID	Rule	Levels	Support
1	(Asset Team, Country, Issuer)	3	MV<>0
2	(Portfolio, Asset Class, Barclay Sector 1)	3	PORT='A'

Table (2.2) Dimension Master

DIM ID	DIM Name	DB Table
1	Asset Team	dbo.tbl_security
2	Portfolio	dbo.tbl_holdings
3	Issuer	dbo.tbl_security
4	Sector	dbo.tbl_security

Proposed Scheme Explanation: **1) Aggregation Rule Master** will hold the user defined rules Dimension 1, Dimension 2, , Dimension (n), cuboid dimension count and support condition. For example, if user sets up a rule to aggregate by Asset Team, Country, Issuer, the Aggregation Master table will have the three dimensions and a field indicating that users is requesting the cuboid at level 3 or 3-D cuboid. Example of the rule set up is shown in the table 2.1 on page 25.

2) Dimension Master is a domain value set of all the possible dimensions or attributes which will be part of the aggregation rule. This serves as a metadata table which is used by BDAE to build the dynamic SQL. The table 2.2 on page 25 Shows few tuples of the Dimension Master.

3) Key Value Master is the Key-Value pair serving as the Index Map Table holding the distinct domain value, Key to serve as a look up table. Example of the data is shown in the table 2.3 on page 26. The Fact Table will hold the Key ID when the aggregation measure is materialized.

4) Aggregation Fact Data is designed to hold the Fact Data of the measures by the requested dimensions. This follows the highly normalized design using the key-values to store the measured data points. Currently the measures are fixed in number. This is show in the table 2.4 on page 26.

Table (2.3) Key Value Master

DIM ID	DIM Name	DIM Value
1	Asset TEAM	IG
2	Asset TEAM	HY
3	Issuer	Apple
4	(Asset Team, Country, Issuer)	IG:US:Apple

Table (2.4) Fact Data Table

RPT Date	Agg ID	Fact Key Value ID	Measure 1	Measure 2	Measure 3
1/1/2017	1	1	4.5	36,897	0.376
1/2/2017	1	2	4.501	36,761	0.376

Proposed Algorithm Proposed Algorithm and Process: The implementation is on MS SQL with stored procedures run daily after the DW load completes. BDAE runs on the current day security level (lowest level) data and stores the results. We have ability to run for any historical period if needed (new rule).

High Level Process starts with the Distinct Values (Key-Value) pair process is run before BDAE run to capture any new distinct values we may have for the current day to add to the Key-Value Master. Once Key-Value table is ready the BDAE will go through the Aggregation Rule Master (AGM) in a sequential order and will extract the Aggregation Rule and Support to generate the dynamic SQL statement. The dynamic SQL Statement will be executed on DWV. The results from the dynamic execution are stored in-memory and updated with the key-value ID and store in the Fact Table, used by users for their analysis purpose using BI/Reporting tools.

As example, for Agg. ID = 1, Rule := Asset Team, Country, Issuer in AGFT Fact {Report date, Agg Id ,Key Value ID} will hold the aggregated measures making it a 3-D Cuboid. This is shown in Table 4. The Key-Values are used extensively in report building. IG:US:Apple can be split into 3 fields and gives the next layer of Aggregation if needed. E.g. If user needs Asset Team, Country we can aggregate if from the lower level rule Asset

Team, Country, Issuer rather than re-aggregating as a separate rule. This is in a way the Bottom-Up-Construction (BUC) as ad-hoc of Iceberg Cubes from lower level cuboids. For that reason we do run the individual distinct values for each dimension before the BDAE.

With just 4 tables we were able to implement the first Big Data Aggregation on DW. Reference to the terms for algorithm: Data Warehouse (View) DW(V), Aggregation Rule Master (AGM), Dimension Master (DM).

Algorithm :

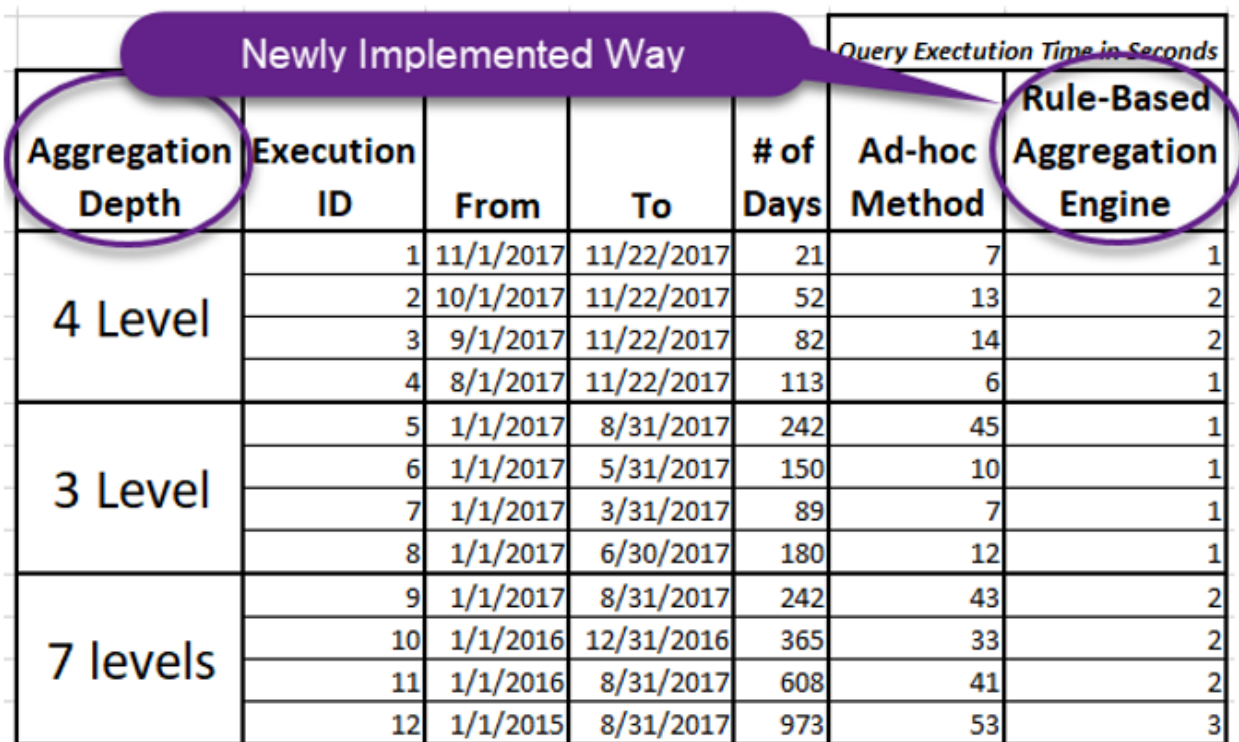
```

1: START BDAE after DW Load Complete
2: Set Reporting Date := Max(DW Holding Date)
3:   Loop : Key-Value Pair
4:     For Every {Dim Name & Rule Value} from (DM Union AGM)
5:       Generate Distinct Values {Dim Name & Rule Value} USE DW
6:       Identify New Distinct Values add them to the Key Value Master
7:       Next {Dim Name & Rule Value}
8:     End Loop : Key-Value Pair
9: Loop : Aggregation Calculation
10:   For Every Aggregation ID i (Rule , Where Condition) from AGM
11:     i(rule) := Extract ({Dim1, Dim2 .Dim(n)})
12:     i(Support):= Extract({Support})
13:   i(SQL) := Build Dynamic SQL from i(rule) and i(support)
14:   Execute (i(SQL)) Result into to In-Memory block
15:   Update Key-Values for {Dim values} in Result (In-Memory)
16:   Delete FACT Table for the Reporting Date and Rule ID
17:   INSERT to FACT Table with the aggregated measures for Rule ID
18:   Next i (Aggregation ID)
19: End Loop : Aggregation Calculation
20:END BDAE

```

2.5 Results

Even though pre-aggregated reports are faster than ad-hoc, we want to show a high level comparison of the running times for both models (ad-hoc New). As there is no aggregated data in our current DW, users run ad-hoc queries (time series) and generate reports. For a comparison purpose, the reports are selected based on increasing complexity of aggregation. Example, aggregation depth 3, 4 and 7 Level (cuboid) reports are selected to compare between ad-hoc method and our new proposed method. The reports are run for various levels of aggregation with varying time periods (Daily, Monthly, Quarterly, and Yearly) with an Execution ID to identify the result set. The table shown in the figure 2.4 on page 28 shows the observed execution times for both implementations (Ad-hoc vs. Rule-Based).



Newly Implemented Way					<i>Query Execution Time in Seconds</i>	
Aggregation Depth	Execution ID	From	To	# of Days	Ad-hoc Method	Rule-Based Aggregation Engine
4 Level	1	11/1/2017	11/22/2017	21	7	1
	2	10/1/2017	11/22/2017	52	13	2
	3	9/1/2017	11/22/2017	82	14	2
	4	8/1/2017	11/22/2017	113	6	1
3 Level	5	1/1/2017	8/31/2017	242	45	1
	6	1/1/2017	5/31/2017	150	10	1
	7	1/1/2017	3/31/2017	89	7	1
	8	1/1/2017	6/30/2017	180	12	1
7 levels	9	1/1/2017	8/31/2017	242	43	2
	10	1/1/2016	12/31/2016	365	33	2
	11	1/1/2016	8/31/2017	608	41	2
	12	1/1/2015	8/31/2017	973	53	3

Figure (2.4) Aggregation Run Result Observations

2.6 Conclusion

As expected, the new method (pre-aggregated) has a superior performance compared to ad-hoc approach. Due to non-indexed dimensions, DW queries are very slow over a longer time periods (execution IDs (5,9,12)). Newly implemented process showed performance increase by an average factor of 15 at all levels of aggregation compared to ad-hoc method. We are able to show the clear case for a flexible aggregation on DW rather than ad-hoc approach. The performance gain in new framework is due to the creation of the base cuboid for the user set up rule and run only that instance and materialize them to table and using the key-value pair for index. By adding 4 tables we are able to implement Big Data Aggregation Engine and avoid $(2n-1)$ cuboid calculations. Our contribution of the paper is in capturing user defined aggregation rules and utilize them for summarized measures which enhances the data cubes techniques and avoids unnecessary cuboid calculations.

2.7 Future

Our solution can bring rapid application development in our investment organization. We are extending our architecture for Data Mining to find trend and identify market events and alert our Risk and Portfolio Managers. By extending academic research on data aggregation, we built a simple, cost effective and portable solution which has potential to be the key building blocks in our Data Mining [2][11] and Machine Learning financial applications [2] using Big Data [2]. We are working towards a Big Data Financial Calculation Framework for Cloud computing enabling application development on Big Data and Cloud Computing Environment. We are piloting a project to migrate our on-premises aggregation engine and reporting framework to Microsoft Azure cloud computing. This will give us flexibility to grow without any hardware costs to maintain our on-premises data servers.

In the next chapter we will discuss how to migrate an on-premises SQL solution provided in this chapter on to Azure Cloud Environment.

CHAPTER 3

BIG DATA - CLOUD COMPUTING

3.1 Introduction

The 2008 Financial Crisis which created a global financial market meltdown is mainly due to badly structured mortgage loans with poor or subpar credit quality and lack of proper tools to measure portfolio risks by the lenders. Even though several problems led to this crisis, we looked at this from a Big Data. Had the infrastructure and analytical analysis tools were present to the lenders, they would have found the various early warning signs on these mortgage loans and could have better prepared for the crisis. Aftermath of the crisis, all the big financial institutions took a fresh look and embarked onto build various tools and frameworks to address this Big Data in their portfolios with data driven analysis. The 3Vs (Velocity, Volume and Variety) of the Big Data in our Mortgage Loan Analysis System challenges our traditional approach in collecting, processing and presenting the individual and aggregated loan level data in a meaningful format to facilitate our portfolio managers in decision making. The traditional methods are implemented on a standalone on-premises SQL server. The methods we presented in this paper of *Capture, Transform, Calculate and Visualize (CTCV)* Framework creates the foundation of migrating from traditional standalone database architecture (on-premises) to Cloud Computing environment using Script Based Implementation. The methods we present are simple but effective and saves resources in terms of Hardware, Software and on-going maintenance costs. CTCV implementation takes a phased approach rather than a big bang model. Our implementation helps the Big Data Management to be part of organizational IT tool kit. This saves hard dollars and brings us in line with the overall firm strategic vision of moving to Cloud Computing for Investment Management Services.

In any investment portfolio management, diversification of the portfolio holdings is

critical to achieve client expected returns by minimizing the downside risk in the portfolio. Diversification can be within a specific sector or across sectors or different types of fixed income bonds. One such investment strategy is to have exposure to the mortgage bonds in the portfolio along with other investment instruments [1]. The total Mortgage Debt Outstanding for 2017Q3 is estimated to be around \$14.7 trillion in U.S. The structure of these mortgage securities are called pools and they consists of mortgage loans taken by individuals. The raw data (factor data) is released on a monthly basis by various Mortgage Agencies like Ginnie Mae, Freddie Mac and Fannie Mae. The individual financial institutions either use third party vendor provided solutions or build their own data gathering applications to collate this information. The data set is huge and is grows from month over month as the loans and pools tend to increase as time progresses. For smaller institutions the cost in implementing third party vendor for Big Data solutions are expensive and cant justify the costs. The other problem is presenting this information in a useful and flexible manner for the portfolio managers. We were able to solve our Big Data problem by breaking it into smaller manageable phases and build a modular based frame work to save organizational costs and reduce maintenance and other infrastructure overheads. Our approach (CTCV Framework) gave our portfolio managers the ability to analysis huge amount of data in a very short period compared to the legacy EXCEL based application. The EXCEL model was severely limited to the amount of information they can analyze in a given day due to the technical limitations of processing Big Data in EXCEL.

The current chapter is organized into eight sections. Section 2 introduces the preliminary definitions and background information. Section 3 highlights the problem, presents related work done on migration of traditional database models to cloud based models in academia, and introduces our proposed solution. In Section 4, we present the solution and in section 5 shows the implementation. Section 6 captures the results and 7 is conclusion with 8 laying the foundation for our future work.

3.2 Background

The Investment Portfolio consists of Fixed Income Bonds [6] and or Equity securities. In the day and age of analytics, the key analytical indicators [15] of these investment assets drive lot of portfolio decisions. Calculating and picking up trends in these analytical measures is a challenge when we need to go through several millions of mortgage loans. Further the Bonds can be classified into corporate and mortgage or asset backed bonds. For our discussion, we will focus on the mortgage backed securities. The MBS (Mortgage Backed Securities) can be further classified into two broad categories by their defining attributes. The MBS security is backed by a pool of securities which can be residential-mortgage backed or commercial-mortgage backed. The figure 3.1 on page 32 is a very high level of how an investor or an institution can invest into a Mortgage Backed Security (MBS) depending on their risk appetite. Furthermore, the MBS which are sold by Investment banks are divided into various tranches depending on the risk profile of the individual loans. These mortgage backed securities are very complex and the details are out of scope of this paper.

Loan	Outstanding mortgage balance	Weight in pool	Mortgage rate	Months remaining
1	\$125,000	22.12%	7.50%	275
2	\$85,000	15.04%	7.20%	260
3	\$175,000	30.97%	7.00%	290
4	\$110,000	19.47%	7.80%	285
5	\$70,000	12.39%	6.90%	270
Total	\$565,000	100.00%	7.28%	279

Figure (3.1) MBS Pool Structure

The Weighted Average Coupon (WAC) and Weighted Average Maturity (WAM) for the above mortgage pool is calculated with the equation shown as 3.1 on page 32.

$$WAC = 0.2212(7.5\%) + 0.1504(7.2\%) + 0.3097(7.0\%) + 0.1947(7.8\%) + 0.1239(6.90\%) = 7.28\% \quad (3.1)$$

$$WAM = 0.2212*(275) + 0.1504*(260) + 0.3097*(290) + 0.1947*(285) + 0.1239*(270) = 279 \quad (3.2)$$

WAM is rounded to the nearest months show in the equation 3.2 on page 33.

3.3 Problem Statement and Related Research

3.3.1 Current Problem

The problem of having a flexible and scalable mortgage analytics from the huge data set by using EXCEL as calculation tool is very challenging and prone to three major issues. The EXCEL solution Extract Load and Transform (ETL) the raw agency mortgage files is not sustainable or practical and is very inefficient for decision making. Each file can contain millions of tuples or rows. The second issue of EXCEL as a calculation tool limits the ability of flexibility for any new metric calculations which are frequently needed on either ad-hoc basis or on a permanent basis. The modifications of EXCEL macros to accommodate any new calculations is difficult in our current version of EXCEL due to the complexity of code and lack of any version control or testing environment, opening a huge Operational Risk for the organization. The third constraint is the storage of historical data for trend analysis or data mining. The issues we have in our EXCEL version can be categorized into three main problem segments. 1) ETL is only possible for one deal at a time and is time consuming. 2) User defined calculations are not possible. 3) Trend and Data Mining capability is not available. Solving these three issues will increase the productivity of our investment teams giving more time for analysis rather than working on data collection and code manipulation. To address the above mentioned EXCEL solution issues, our research focused on the existing academic and industry research on these issues of huge data stores and ETL tools to address our problem (1) and flexible calculation frameworks and data mining tools to address (2,3) issues. There is a lot of academic research in terms of huge data store/ETL [16] and data mining tools.

The area we found is challenging is the flexible user defined calculations and running it on a distributed and on elastic data lakes and migration of SQL databases to cloud environment [10]. The mortgage pools are made up of individual loans. The data size of these pools can be viewed as Big Data. The main characteristics of Big Data is defined by the main three pillars by which we categorize the underlying data set. They are called 3Vs. [16].

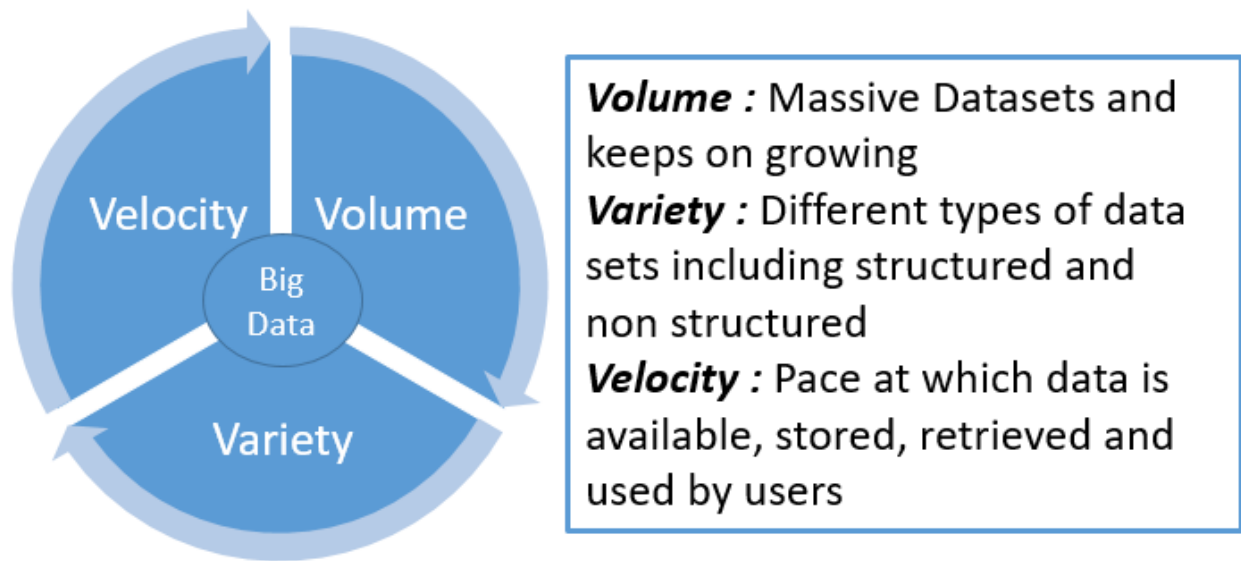


Figure (3.2) Big Data - 3 Vs

The Fig. 2 shows 3Vs of Big Data in our study are Volume (Terabytes of data) in raw format with the Variety (Mixed data values) and Velocity (Changing daily) which makes it a best candidate for our study and implement our method of Capture, Transform, Calculate and Visualize (CTCV) to build a framework which can be leveraged for other investment data related decision systems.

In academia as well as in industry there is a major research work carried out in terms of supporting large files coming from various source systems. During 2002 and 2003 Google came up with their proprietary file system Google File System (GFS) [4] which led the way to several other research groups to come up with their architecture to support large file systems. Google also published their MapReduce [5] programming model which can process terabytes

of data on thousands of machines. MapReduce program was distributed over large clusters of commodity machines. During this time, Apache open-source developed Hadoop architecture and called it as Hadoop Distributed File Systems (HDFS) and introduced their MapReduce programming model. The MapReduce architecture uses a map function specified by users that processes key-value pair to generate a set of intermediate key-value pairs, and a reduce function that merges all the intermediate values with the intermediate key [4]. Even though this is out of scope for our current paper, we are laying the architectural foundation for future work on processing our large data sets of mortgage pool information on a Cloud Computing environment using these distributed techniques. We keep our implementation open for our future needs.

3.4 Proposed Solution

After going through the pros and cons of Cloud Computing and the flexibility of the services provided by various Cloud vendors, we decided on using Microsoft Azure Cloud as our platform. We propose the solution in two phases. **Phase I:** is building a standalone SQL DB solution to implement ETL and flexible calculation engine to calculate various analytical measures on our Big Data. This is not our focus of our current paper. The Phase II which is migration of the phase I solution is our focus.

The proposed Phase II or CTCV solution at high level, will take the existing installation of standalone in-house or on-premises solution and port it to Cloud Computing platform with minimal disruption to our business processes. The figure 3.3 on page 36 gives a high level process flow of the events of our proposed implementation of CTCV. Our preferred method is to take the SQL DB Script to build and implement the Cloud Instance rather than restoring the standalone database backup. Our approach of Script Based will have initial benefit of going through the objects where there will be changes needed to fit into the Cloud Computing framework. This shortens the implementation time and produces a cleaner and leaner version of the database schema. Implementation Our proposed solution of migration of our Phase I solution (Standalone SQL Database) to Cloud Computing environment is done in two parts.

Phase II – Migration of Standalone SQL solution to Cloud Computing

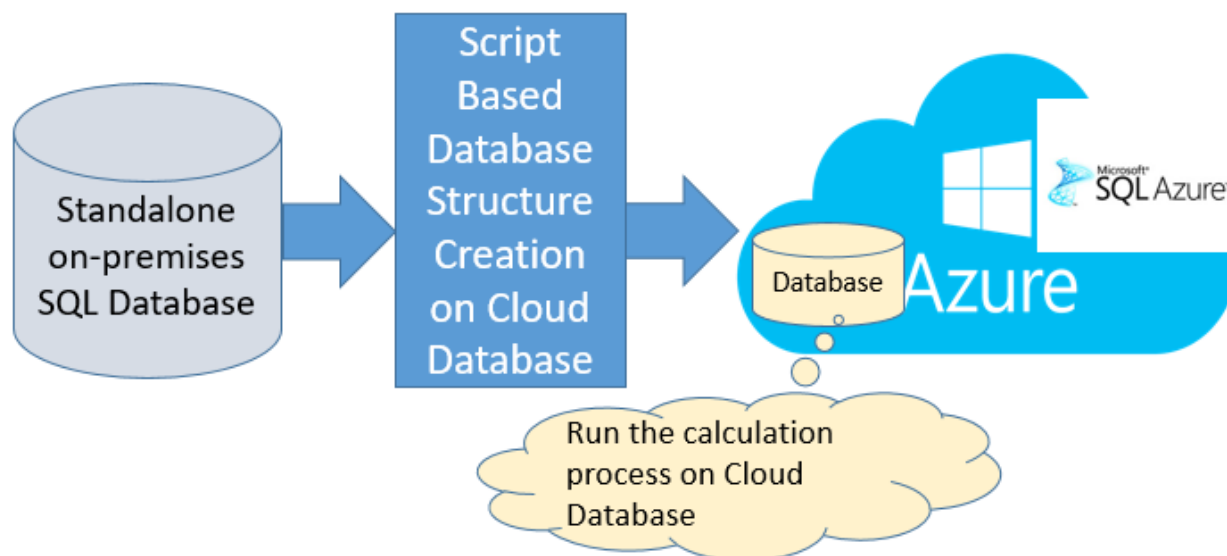


Figure (3.3) Proposed Architecture on Cloud

Pre-migration and Post-migration. These two will set stage for a better and stable migration process.

3.4.1 Pre-Migration:

The first step is to verify the subscription of the Microsoft Azure Account with the institutional license team and if not we need to obtain the needed subscription. Usually depending on the usage the organization purchases the subscription level. Once we have the proper license set up, check the access to the Azure portal (<http://portal.azure.com>). In the figure 3.4 on page 37 We show the general layout of the Azure Portal which can be maintained by the subscription we have

Our implementation is on a Windows Azure MSDN Visual Studio Premium which gives us the ability to build elastic database servers with a 250GB space. We are using this space to prove the proposed architecture solution will work and can be implemented to bigger database needs. Our test data is for a sub set of Mortgage Agency Pools. The data set we have is for Agency FN and currently we have 12 raw files with 14 million total rows.

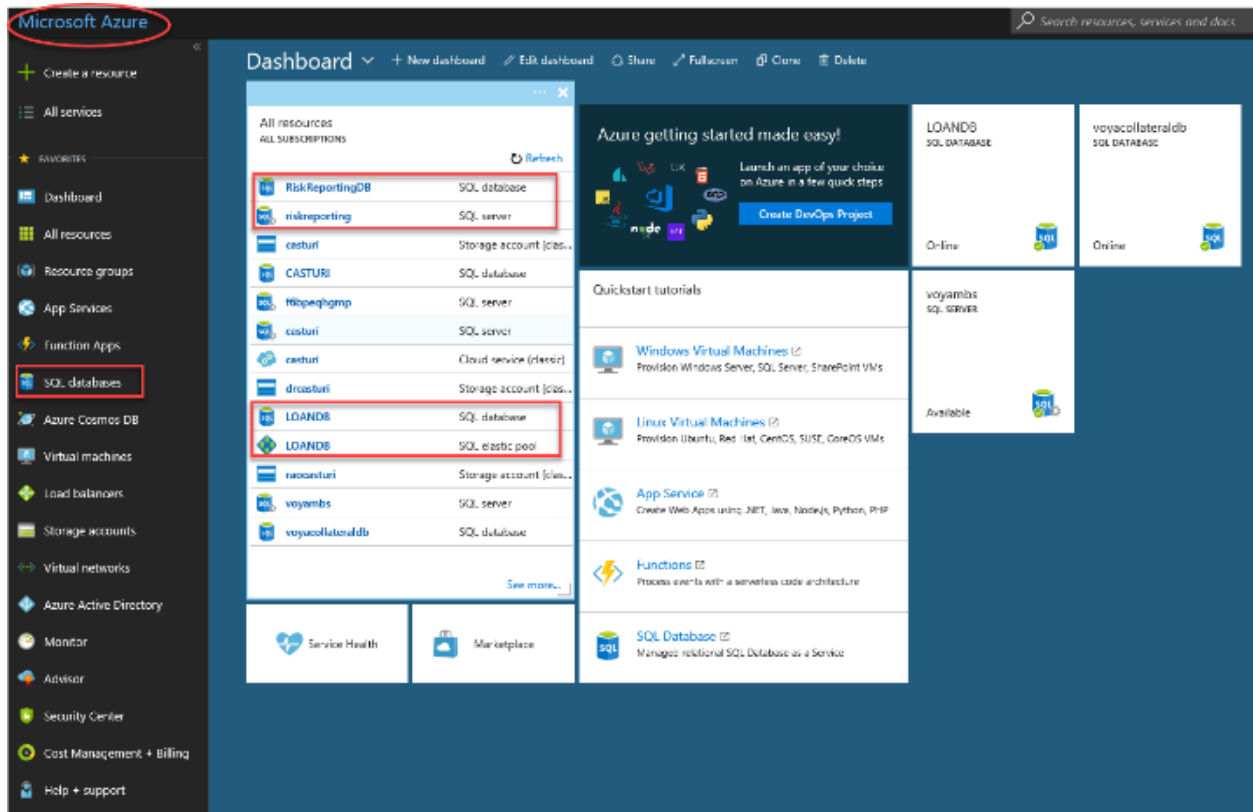


Figure (3.4) Azure Cloud Dashboard

3.4.2 Proposed Implementation Steps:

To implement our solution of Script Based, we first took the current implementation database schema script. These DDL scripts are saved as Object Categories. E.g. For Tables, we called as AZURE Table Script and for the Store Procedures are called as AZURE Procedures. It is up to the individual team to decide the naming convention. This step gives the implementation team a change to modify any non-cloud scripts or rework the non-cloud scripts before implementing on the Cloud instance. One example of script modification is removing any cross-database-queries. Next step is to create a proper database server and database on the Azure Cloud. This can be done with simple steps of by going through the Microsoft Documentation [17]. Once we have the Azure Cloud SQL Server and Database, the next step is to run the DDL scripts which we saved as Object Category Scripts our current implementation. If the SQL DDLs are standard, then we should not see any issues.

We did encounter few issues which are rectified during the migration. The Azure SQL code will not let us reference the Database.Owner.Object. The (database or Schema name) is not permitted in the Azure syntax of referencing the database objects. This will pose an issue with migration if we have linked servers or queries going across multiple databases. The newer version of Azure SQL has a fix.

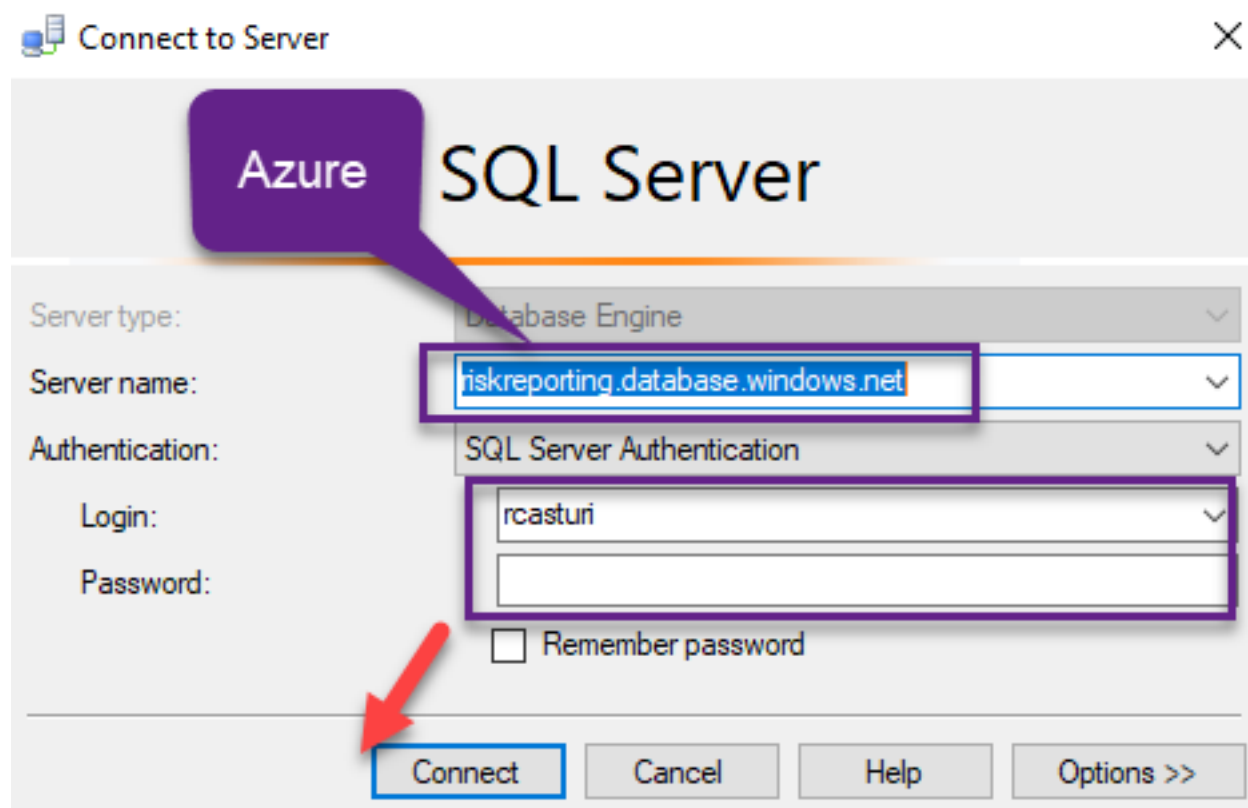


Figure (3.5) Azure SQL Linked Server Details

Create an External Data Source uses the architecture in the Azure SQL Database under the External Resources and use that in DML when writing the standard query referencing the remote SQL table of SQL Server instance in Azure SQL installation.

3.4.3 Post Migration:

Till this point of implementation, we used the Azure portal. Now it is time for connecting to the Azure Cloud via other client tools. Usually in the organization, the data

base professional (Database Administrators, Developers, Designers) use several client tools to connect to the SQL Database instances to do their regular tasks. The Azure SQL is no different and we would like to show the easy way to connect to the already created Azure Cloud SQL database. For our project we used the Microsoft SQL Server Management Studio (MSSQLMS) as our client tool to connect to the Azure Database instance. While creation of the Azure SQL Server Azure will create a unique server name to reference the instance. In the figure 3.6 on page 39 shows the Azure SQL Server and also the Azure SQL Database with the names we provided for referencing them through our implementation. This is a big step as creating the database on Azure depending on the subscription level and we want to make sure we have enough service level contract in place to proceed.

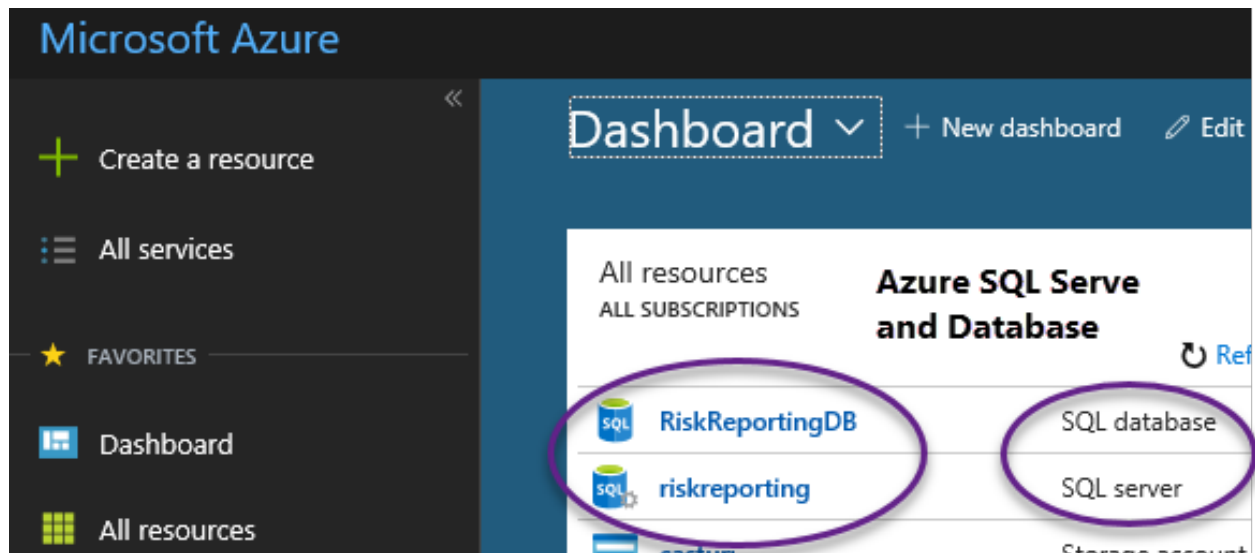


Figure (3.6) Azure SQL Dashboard layout

Once the database schema is set up with the needed external sources, we then populate the mapping table data needed but importing the data from files or current standalone SQL Database. For our instance we named our SQL Server as riskreporting and the instance can be referenced as riskreporting.windows.net.

Our Azure Database instance is RiskReportDB. The figure reffigpart2:sqlclientinfo on page 40 shows the connection to the Azure SQL Server and the database on the server to which we have access. We set up 2 resource pools one with elastic (voyacollateraldb) and one

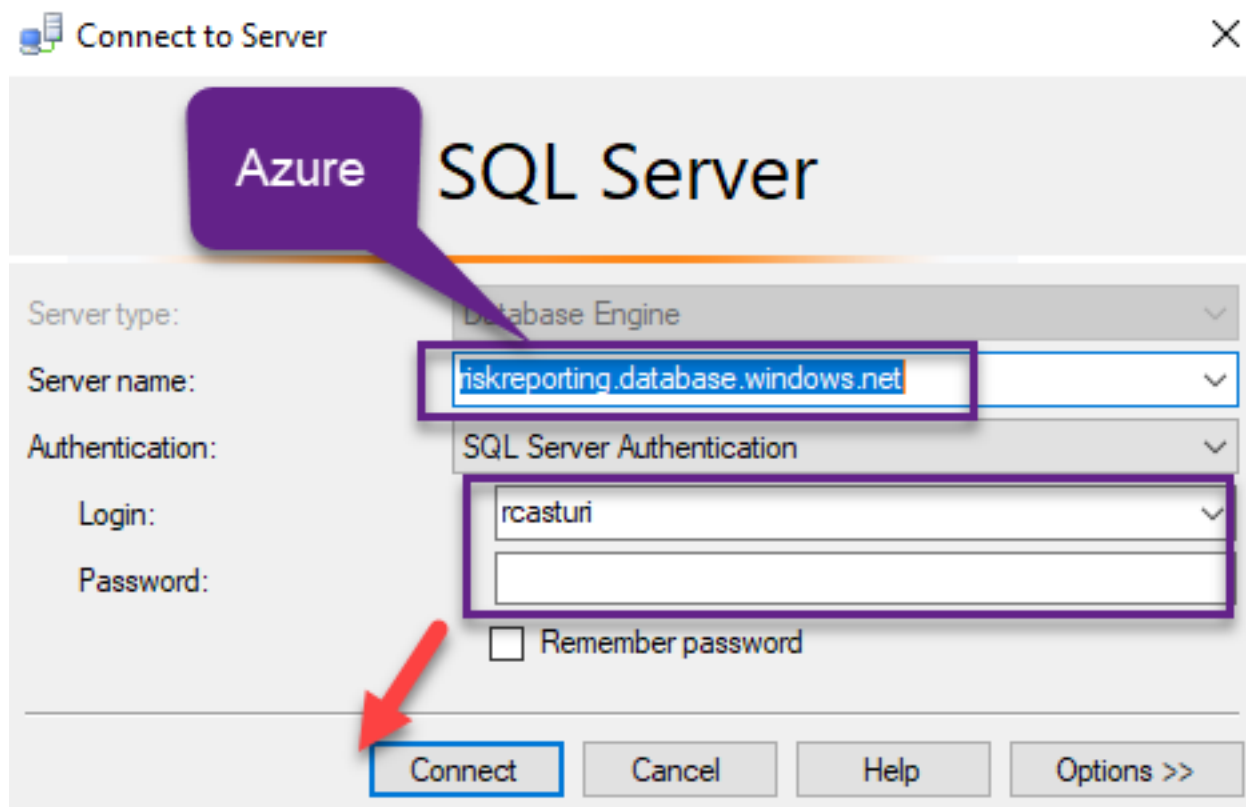


Figure (3.7) Azure SQL Client Connection Information

without elastic database (RiskReportDB). This set up is to verify the elastic nature of the database which we need for our implementation to test the multiple month storage which solved our historical nature of the data store.

Results We ran our user defined Analytical Aggregation Rules on the newly set up Azure SQL Cloud Computing environment. There were no errors. The final calculations were verified with the standalone SQL Database implementation which is our Phase I production. There are no deviations or errors in our newly implemented Azure SQL implementation (Phase II). The subscription we currently have is a very close match to the on-premises so we did not see the any significant improvement in performance. As we noted the higher subscription for more DTUs the better the performance. This is an advantage of utilizing Azure Cloud Computing services as we can increase our DTU subscription depending on our necessity. These are conservative estimates to give us ample room for migration and re-work if needed. This is show in the figure 3.8 on page 41.

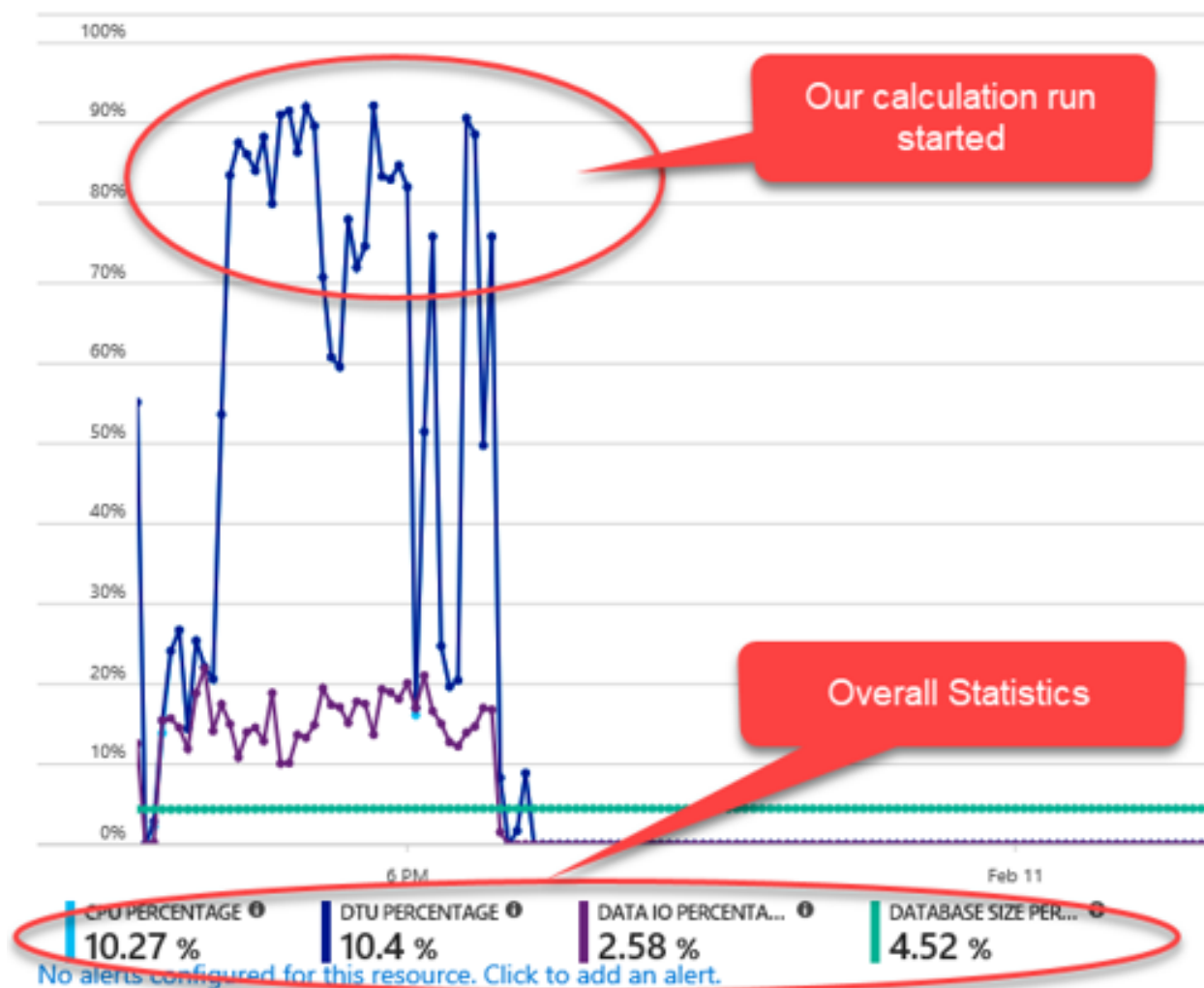


Figure (3.8) Azure SQL Activity Monitor

Another observation we did depending on the rating category by services without any business disruption. Here we applied ratings for several categories which are important for our organization. We recorded the platform (On-premises, Azure) suitable for our organization for future needs and plotted the rating of each category and assigned an appropriate rating between 0 and 5. The results of the rating based scale will be discussed in our next section of conclusion as part of the comparison of the on-premises and Azure Cloud Computing Architecture. In the figure 3.9 on page 42 shows the plot by category and rating for each category.

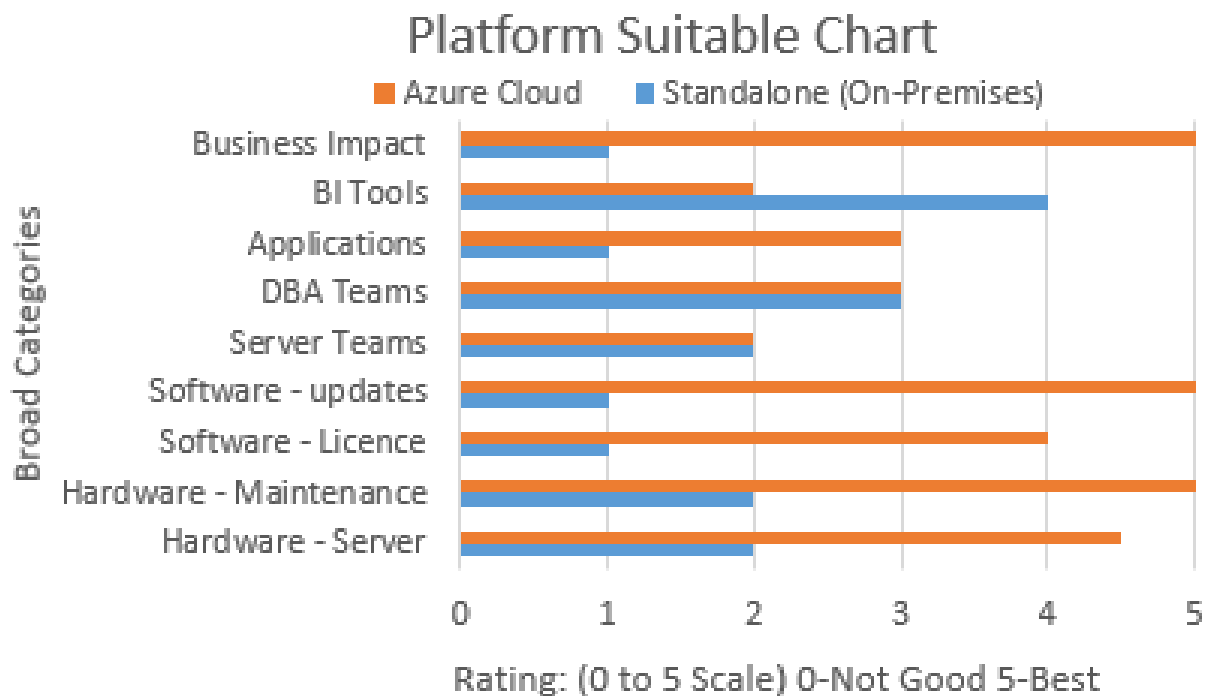


Figure (3.9) Azure SQL Migration Criteria

3.5 Conclusion

The results of migration from a standalone or on-premises SQL Database to Cloud was a successful and this is due to the preparation of the pre-implementation preparation we took before actually implementing the DDLs on Azure SQL platform. The initial estimates of our migration was very conservative giving us enough room to implement any work around for migration scripts to Azure SQL database. We plotted the initial estimated time for each task and the actual time taken for the task to implement. The graph in figure 3.10 on page 43 shows the solid bars are estimated and the line showing actual time we took to implement. The shorter times for actual implementation can be largely contributed to the preparation and project planning going through various scenarios and studying several case studies which are available in industry and also on Microsofts knowledge base.

This actually helped a lot in preparing the perfect scripts to upload and run on Azure SQL Database. Usually the rework is in making the scripts run without errors and that we were able to accomplish early in our implementation phase. We called this as pre-

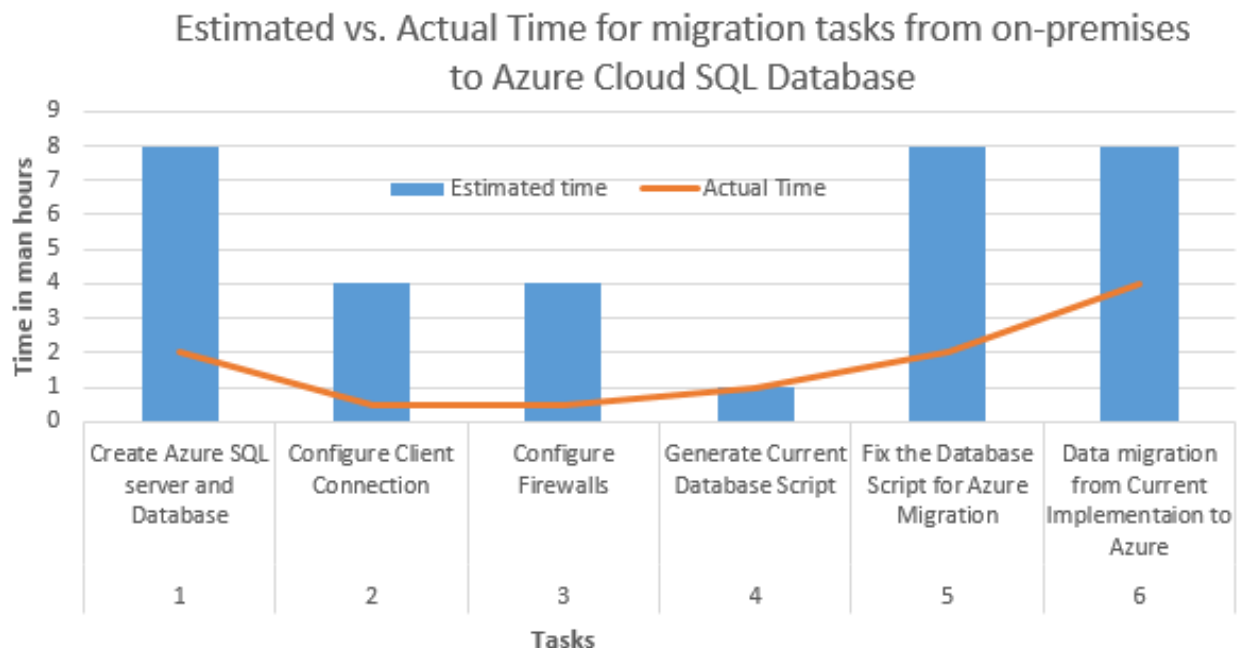


Figure (3.10) Azure SQL Migration - Implementation Time Chart

implementation task.

The Infrastructure as a Service (IaaS) [18] architecture promised by Microsoft is actually worked well for us. Due to the planning and execution the implementation by script based approach, we were able to achieve building the foundation of our migration tool kit to manage Big Data Management. The growth of asset under management AUM depends on the flexible and scalable enterprise architecture and we believe Microsoft Azure Cloud Computing platform solutions will enable to travel on a growth trajectory. The contribution of this paper is not just for any investment company but can apply to any data driven business in industry or in academia.

3.6 Future Work

As we mentioned earlier in the paper, this project implementation is a multiphase implementation designing each phase to be the launch pad for the next phase. From EXCEL version to the Standalone SQL Solution itself is a huge project which included flexible user calculations. During that design phase we did keep our next phase of implementing the

standalone solution on Cloud Computing environment as that technology is evolving and we do have need to embrace that technology as the hardware, software cost is getting higher for a standalone SQL Solutions. With the implementation of the migration project, the future is to build the analytical trending models on the Cloud Computing platform to give our portfolio manager the cutting edge technology for the day to day investment decisions.

Our implementation added value to our organization and paved the path for other teams to migrate from standalone database instances to go in to next stage of Cloud Computing. Our paper and implementation successfully demonstrated that script based implementation is worthwhile undertaking in migrating several of our on-premises databases to Azure Cloud Computing environment. The Microsoft tools [19] we mentioned and proposed will help any small to mid-sized company to migrate their existing on-premises databases to cloud based architecture using the simple procedures mentioned in our paper. The methods are be applied to any database driven applications in any organization. The solution is flexible and scalable and in future we are trying to make is a standard migration tool for our organization. The tool kit we proposed which includes the script based implementation framework along with the set-up of the client and development tools will make any migration seamless and very little downtime for an organization.

The next chapter we discuss the parallel computation of our on-premises aggregation engine and the modifications to incorporate the complete user defined calculations for mortgage characteristics to set the stepping stone for the data mining FinTech application.

CHAPTER 4

DATA MINING - PARALLEL COMPUTING

4.1 Introduction

Even though the recent technological innovations in cloud computing, distributed database architecture grew to a point where they can now address Big Data process [19], still most of the companies are struggling to implement their solutions on cloud computing environment. This is mainly due to lack of proper case studies and application frameworks available on a public research domain. Most of the implementations are vendor provided, high cost, consulting type and long drawn projects which consume valuable Business and IT resources for an organization. In this paper we propose a simple to implement (parallel data load and any aggregation calculation framework), easy to maintain and flexible architecture framework which can be adapted as a tool for small to mid-size investment organization in implementing a Distributed Cloud Computing Architecture. Our framework is an extension of traditional Distributed Database Design with the horizontal partition of the relations to parallelize the computation on Azure SQL instance and materialize the aggregated results with SQL Views for users, Business Intelligence (BI) Reporting, Data Mining and Knowledge Discovery applications. The solution is implemented on Azure SQL Cloud Computing platform to build the financial calculation framework. The raw data (factor data) for Mortgage pools [1][2] is released on a monthly basis by Mortgage Agencies like Ginnie Mae (GN), Freddie Mac (FH) and Fannie Mae (FN). The individual financial institutions either use third party vendor provided solutions or build their own data gathering applications to collate this information. The data set is around 2 TB every month and grows from month over month as the loans and pools tend to increase as time progresses. Currently we are only targeting a small portion of the mortgage pool universe as the current architecture cant even handle bigger data sets in our existing calculation framework. With our proposed new architecture,

we can target full universe of mortgage pool data or factor data. For smaller institutions the cost in implementing third party vendor for Big Data solutions are expensive and cant justify the costs. After going through various cost benefit scenarios, we in our organization decided, to build a flexible framework which not only capture the raw mortgage factor data in timely manner but also build our user defined calculation analytical engine to aggregate factor data to support future Portfolios Analytical Decision Systems (PADS) [3] with the help of latest Microsoft Cloud Computing Technology (Azure). Our approach and successful implementation of our proposed solution gave our portfolio managers the ability to analysis huge amount of data in a very short period compared to the legacy EXCEL based application. The EXCEL model was severely limited to the amount of information we can analyze in a given day. This is due to the technical limitations of processing Big Data in EXCEL. Excel can be effective as an analysis tool for small amount of data but when it comes to Big Data analysis, it is severely limited. Subsequent paragraphs, however, are indented.

The contribution of our paper is a flexible calculation framework implemented successfully on AZURE SQL Cloud Computing [18] Environment by **Distributing the raw Big Data among multiple SQL relations (tables) and build a parallel execution method based on Node based architecture** harnessing the computational power of Azure SQL Cloud based implementation which can save huge costs to the organization in terms of hardware and software. The architecture on Azure is expandable to meet the organization needs when it comes to computational power. The pay as you use operation model for a long time in service sectors and that concept is now introduced and implemented successfully with Microsoft Azure Cloud Computing. We can scale up or down on our computational power depending on our needs.

Our contribution and research in creating a node-based architecture on Azure SQL Database [20] can be used by any organization as a case study to implement their data driven financial calculation applications which deal with large or Big Data and utilize industry strength cloud platforms (Microsoft, ORACLE, AMAZON etc.).

The current chapter is organized into seven sections. Section 2 introduces the prelim-

nary definitions and background information. Section 3 highlights the problem, presents related work done on migration of traditional database models to cloud based models in academia, and introduces our proposed solution. In Section 4, we present the implementation of our solution over the current mortgage database. Section 5 compares the benefits from existing architecture to the newly proposed framework. Section 6 is our conclusion and Section 7 gives our future works including Data Mining initiatives and Hadoop and MapReduce project initiatives.

4.2 Background

An Investment Portfolio can consist of Fixed Income Bonds [15] and or Equity securities. In the day and age of big data, the key analytical indicators [?] of these investment assets drive lot of portfolio decisions. Calculating and picking up trends in these analytical measures is a challenge when we need to go through millions of mortgage loans. For our discussion, we will focus on the mortgage backed securities. The Mortgage Backed Securities (MBS) can be further classified into two broad categories by their defining attributes (Agency and Non-Agency mortgage-backed). The pool of residential mortgages can be thousands of individual loans put together. Individual or institution investors can buy this MBS. The Fig. 1 is a very high level of how an investor or an institution can invest into a MBS depending on their risk appetite.



Figure (4.1) High-level Mortgage Backed Securities life cycle
MBS security life cycle

The mortgage pools are made up of individual loans. The data size of these pools can

be viewed as Big Data. The main characteristics of Big Data is defined by the main three pillars by which we categorize the underlying data set. They are called 3Vs. [21].

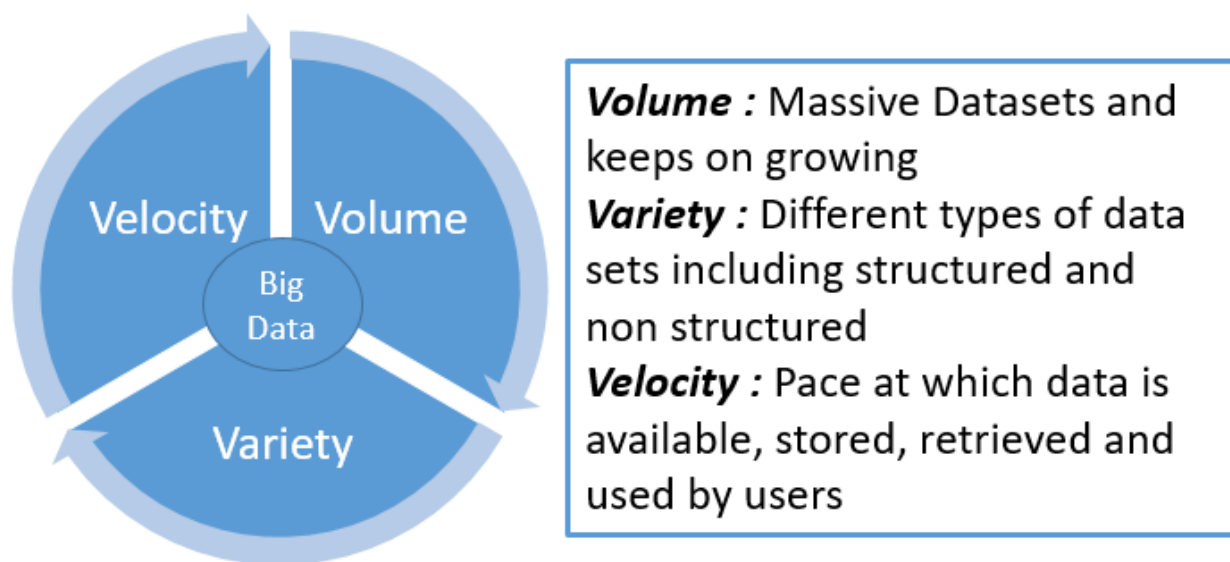


Figure (4.2) Big Data categorization diagram
 Big Data categorization diagram

The figure 4.2 on page 48 shows 3Vs of Big Data in our study are Volume (Terabytes of data) in raw format with the Variety (mixed data values) and Velocity (Changing daily) which makes it a best candidate for our study and implement our flexible calculation framework which can be leveraged for other investment data related decision systems.

Loan	Outstanding mortgage balance	Weight in pool	Mortgage rate	Months remaining
1	\$125,000	22.12%	7.50%	275
2	\$85,000	15.04%	7.20%	260
3	\$175,000	30.97%	7.00%	290
4	\$110,000	19.47%	7.80%	285
5	\$70,000	12.39%	6.90%	270
Total	\$565,000	100.00%	7.28%	279

Figure (4.3) Example of a pools of loans in an MBS pass through security
 Example of a pools of loans in an MBS pass through security

The details of MBS [15] security or the structure is out of scope for this paper discussion.

To make it simple if we take the loan 1 from figure 4.3 on page 48 we can see what is outstanding on the mortgage as a balance and what is the rate at which the loan is being serviced and the how many months are remaining on the loan. Along with it also give the % weight of the loan in a given pool where it belongs. This is key for an investor when they are looking to invest in this pool of securities. From these basic measures we compute more complicated measures to use as indicators for investing.

There are several calculations done on various dimensions (range bound). E.g. Showing the WAC over 0-2, 2-5, 5-7, 7-10 and above 10. Another example of a measure is Conditional Prepayment Rate (CPR). CPR is a sub calculation on Single Month Mortality Rate (SMMR) which is calculated for each month of the mortgage and use that SMMR to calculate the CPR for a specific month. The equation (2) shows the calculation for one-month CPR. SMM is calculated with as shown in equation (1). Some of the mortgage calculations are recursive in nature.

$$SMM(t) = \text{Prepayment in month } (t) / \{ \text{beginning mortgage balance for month } (t) \text{ scheduled principal payment in month } (t) \} \quad (4.1)$$

$$CPR(1) = 1 - (1 - SMM(1))^{12} \quad (4.2)$$

Prior to our new implementation, the solution was based on EXCEL with VBA code for calculation various fixed aggregate values.

4.3 Problem Statement and Related Research

4.3.1 Problem Statement

Excel is a tool for analysis for a limited data set but not for a Big Data analytical analysis. There are three major issues we face in our current EXCEL based tool. Extract Transform and Load (ETL) [7] the raw agency mortgage files are not sustainable and is not possible as each file can contain millions of tuples. The second issue of EXCEL as a calculation tool limits the ability of flexibility for any new metric calculations which are frequently needed on either ad-hoc basis or on a permanent basis. Due to the nature of links and look ups and no control on editing in the existing EXCEL solution, it opens a huge operational risk to the organization. The third constraint is the Storage of historical data [19] for trend analysis and data mining. Excel cant handle a 2 TB data set or aggregated data [14] of 5 to 10 million rows in EXCEL for one month depending on number of calculations needed and performed. For Data Mining we do need historical data set to see patterns and further extend it to AI and Machine Learning algorithms, our current solution cant be used.

Summarizing the issues, we have three main problem segments. 1) ETL is only possible for one deal at a time and is time consuming. 2) User defined calculations are not easily possible. 3) Trend and Data Mining [20] capability is not available. Solving these three issues will increase the productivity of our investment teams giving more time for analysis rather than working on data collection and code manipulation. Our contribution to the research community is a perfect use case for a financial calculation framework using the Cloud Computing solution for our Big Data problem. Our Mortgage Factor Raw data can be viewed as Big Data and a good candidate for our Capture, Transform, Calculate and Visualize (CTCV) framework which can be leveraged for other investment data related decision systems.

4.3.2 Related Work

There is lot of academic research and vendor products available in the domain space ETL in terms of processing multiple files efficiently. There are several architectural frameworks which can be implemented on our problem. In academia as well as in industry there is

major research work carried out in terms of supporting large files coming from various source systems. During 2002 and 2003 Google came up with their proprietary file system Google File System (GFS) [22] which led the way to several other research groups to come up with their architecture to support large file systems. Google also published their MapReduce [13] programming model which can process terabytes of data on thousands of machines. MapReduce program was distributed over large clusters of commodity machines. During this time, Apache open-source developed Hadoop architecture and called it as Hadoop Distributed File Systems (HDFS) and introduced their MapReduce programming model. The MapReduce architecture uses a map function specified by users that processes key-value pair to generate a set of intermediate key-value pairs, and a reduce function that merges all the intermediate values with the intermediate key [11]. Even though our implementation is not on Hadoop platform we are researching the possibility of our implementation of raw factor file load via Hadoop-MapReduce architecture.

As part of the research for our project, we came across several Distributed Processes (DP) [19] [23] or Distributed Computing (DC) methods and papers. DP and DC are inter-linked and to some extent distributed processing is already implemented in our modern computer architecture. For example, in single-processor based computing systems, the central processing unit (CPU) and the input and output devices and operations (I/O) are separated and overlapped where it is possible. Parallel computing is further defined by Flynn's Taxonomy of Parallel Architectures by data and instruction set execution. Single-Instruction, Single-Data (SISD), Multiple-Instruction, Single-Data (MISD), Single-Instruction, Multiple-Data (SIMD) and Multiple-Instruction, Multiple-Data (MIMD). Each one of the four (SISD, MISD, SIMD, MIMD) are studied extensively and implementations are very widely used in research as well as in industry. There are various ETL tools adapted and incorporated all four types of architectural designs. Example Microsoft SQL Server Integration Tool (SSIS) can handle multiple data files in parallel by using the For-Loop Container [7]. The design is to create Parallel Task and Child Task and copy the Parallel Task to achieve the parallelism by copying the child task inside the Parallel Task Container. Bootstrapped to SSIS package

the SQL Procedure we came up to calculate our user defined calculations is a perfect solution for the parallel execution of the code.

We evaluated several industry products in the Big Data and Cloud Computing space including Oracle Cloud Platform, Microsoft Cloud Technology (AZURE) [10] and Amazon Web Services (AWS). The best suited for our needs is Microsoft plat-form minimizing the business disruption. Fig. 4 gives us a high-level view of various products the vendors offer in the Cloud Computing Space.

Cloud Product	Short Description and services
SaaS: Software as a Service	Centrally hosted and managed for the end users. Scalable to meet customer demand. E.g.. Like any utility company
PaaS: Platform as a Service	Vendor provides platform and Customer provides applications. E.g. Virtual Machines, SQL Serves , Web Servers etc.
IaaS: Infrastructure as a Service	Vendor provides all the support and maintenance of server farms, running virtualization software.

Figure (4.4) Brief description of major services by Cloud Computing providers
Brief description of major services by Cloud Computing providers

The gaps we found in our research literature is how to implement a user defined Cloud Computing Architecture for a Financial Calculations. This is not done or not available to research community as to design and build such kind of applications needs a lot of financial acumen along with a very strong Database Design and Development knowledge. The vendor products available are very expensive and are focused to a very narrow needs of business. Some example of such vendors is in MBS space are KDSGlobal, MDS etc. We could not find any relevant research material on this topic so we decided to research and find a solution which can be ported easily to other financial calculation not just restricting to our Mortgage Factor Aggregation (MFA). The complex calculations should be handled by a flexible user defined rules to accomplish the MFA.

4.3.3 Proposed Solution

Our approach is to use the Azure SQL Database Server to host our raw factor data for mortgage pools (Big Data) and run distributed user defined calculations on the raw data files sourced to SQL stage tables in parallel and materialize the results for BI Tools. The actual data load can be accomplished in parallel using Microsoft SQL Server Integration Services (SSIS) Looping techniques. This is not the focus for our paper as parallel loading is done many a times using many vendor tools like Microsoft SSIS and Hadoop - Map Reduce Techniques. The challenge for us is to design a database schema to accommodate our goal of providing easy, flexible and cost-effective user defined calculation engine on a Cloud Platform using distributed data-base design patterns [24] [25]. The focus of this paper is on the second part of the overall problem statement of parallel execution of user defined calculations on Azure SQL Cloud Computing environment. At high level the architecture of the proposed solution is shown in Fig. 5. The idea was tried several times and implemented

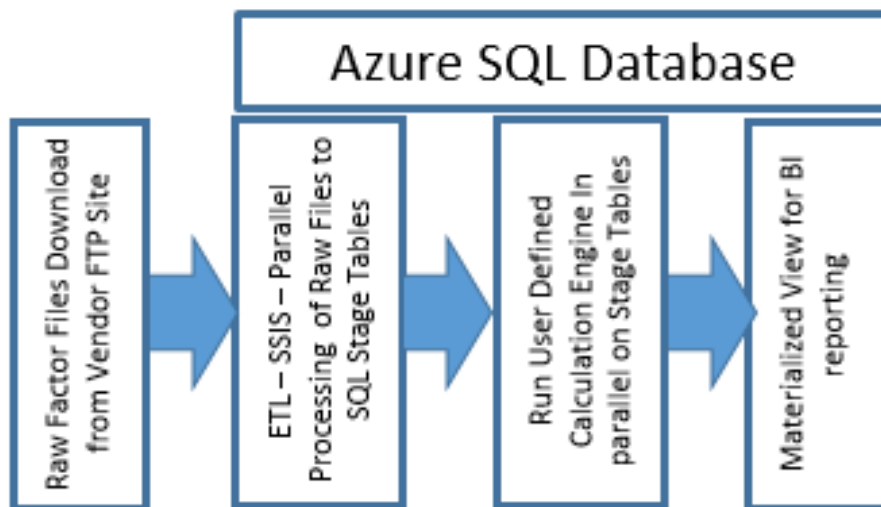


Figure (4.5) Proposed Distributed Calculation on Data Copies (SQL Stage Tables)
Proposed Distributed Calculation on Data Copies (SQL Stage Tables)

The proposed architecture is a flexible data base design using the standard Relational Schema Architecture on Microsoft SQL Server Database. The execution of the calculation engine runs in parallel as multiple instances with different data set. The actual creation of

the Azure SQL Database is out of scope for this paper. The architecture in layman terms is like a single cash counter at a store checkout verses multiple cash counters. The customers with the products they bought can be treated as our raw factor files and each cash counter is a node designed to calculate the total amount to be charged per each customer and updates one main register with the customers payment. This is concept of parallel processing is not a new concept but adapting to financial calculations on cloud computing environment gives re-search community to use it in a public domain. Technical details are discussed in the implementation section.

4.4 Implementation

Our proposed implementation is on Microsoft AZURE SQL Server with a S3 standard (100 DTUs). For this implementation we created a non-elastic database pool and with 250 GB database. The actual creation of the AZURE SQL Server Database is out of scope for this paper but can be referenced to prior paper by the same authors [21]. Along with the Azure implementation, we also implemented the same solution on on-premises SQL database on Windows server with 8 CPUs and 20 GB RAM with SQL 2012 database.

The on-premises solution serves two purposes. The on-premises solution serves as a benchmark to our Azure and serial processes of Big Data and secondly the on-premises solution gives us an opportunity to propose an institutional solution for Azure applications for future investment frameworks and paves path for Global Access of our applications.

Our IT department was able to procure a user license for our research on Azure Cloud Platform providing the tools needed (Visual Studio for SSIS and Microsoft SQL Management Studio Client) in order to connect and test our proposed concept of user defined calculation database schema [22]. The key it to utilize the database design and ability of Azure Cloud database architecture to drive

4.4.1 Current Architecture

The current architecture can only process one factor file a given time. There is a Data Load Form which will ask the user to input the deal number and then the EXCEL Macro (VBA Code) will be executed to get the data from raw factor files from the Vendor Data Source (VDS) depending on the user data license.

Even though we can modify the EXCEL Solution code to take in multiple deals at a time, still the raw factor file sourcing, parsing and loading will be an issue in EXCEL sheet. Some of these deals will have many loans (in millions) which EXCEL sheet cant handle. The EXCEL application is built to extract the data, transform the data and also calculate the user defined calculations one deal at a time. Modification of the VBA Code is very manually. The EXCEL version served as a business use case for us to build more robust and enterprise solution using the standard tools and the utilize new cloud computing technology platform.

4.4.2 Proposed Architecture

The proposed architecture is based on AZURE SQL Server [26] with RDBM [9] Architecture. The core architecture is based on the main assumption that user defined calculation rules can be dynamic. Users can come up with another set of new calculations which are critical to the business and the system should able to handle without any code rewrites. If there is a specific calculation which our calculation engine cant handle, we can incorporate it easily in our main calculation engine by extending the existing calculation classification category. The calculation engine is based on calculation classification category as a primary driver for extendable code. To have a dynamic calculation and reporting framework, we designed a flexible database schema, giving the users the ability to edit the core calculation rules. A calculation rule can be defined and saved in a SQL Relation as following in terms of relational algebra shown in the next couple of lines shown as $C(i)$

$C(i) = R(\text{reportdate}, \text{collateralgroupname}, \text{agencytype}) (\text{re-portdate}, \text{collateralgroupname}, \text{agencytype})$
 $SUM \text{ obal}, SUM \text{ olnsz} (StageTable R)$ (3) $C(i)$ is the user defined calculation rule is the selection of the attributes from the Relation (Stage Table R) is the aggregate function to be

used on the set group by attributes

The calculation C can be set up by the user and that we capture as an individual calculation. The database schema [17] [18] consists of 5 major relations (SQL tables). They are 1) User Calculation Rule Master also called Calculation Field Master 2) Range Definition Master 3) Distinct Values Data Set (Key-Value Pair) 4) Result Materialization and 5) Raw Factor Data. With few supporting relations we were able to build a dynamic user defined and easy to implement architecture which is portable to any other data sets. We enhanced the calculation framework to parallelize by adding a Node identifier when a calculation is initiated. this is shown in the figure 4.6 on page 56. The Raw Factor Files are sourced through the ETL process to the SQL Stage Tables. Once the raw factor file is loaded the SSIS Package will initiate Master SQL Procedure (MSP) with a Node ID and SourceTableName (STN) as parameters. MSP Will work on the STN going through all the user defined calculations and save the results to Results Relation and marking the file process complete in the Process Log Table. The MSP is written in anticipation of platform independent by using standard SQL code.

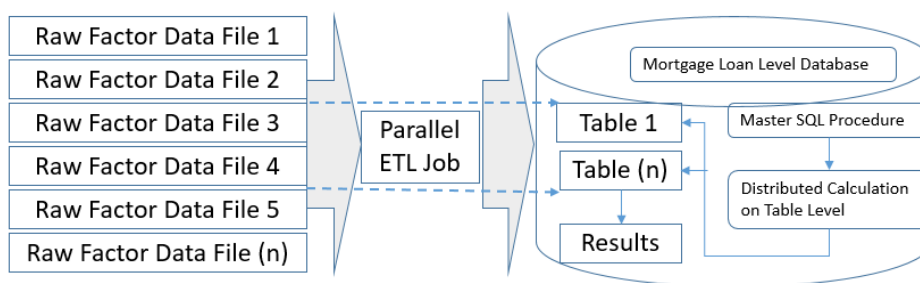


Figure (4.6) Proposed Architecture (Parallel computational execution on Table 1..n)
Proposed Architecture (Parallel computational execution on Table 1..n)

SQL Table Description: Calculation Field Master (CFM) will hold the calculations to be done on different attributes. Example of a user set up is shown in the table 4.1 on page 57. This table has 16 attributes and to save space we are showing only few attributes. The Field Name denotes the actual Calculation to be done and rest of the fields will give more details about the calculation. Where there are weighted average or any specific calculations

using a denominator and numerator, we can specify that in the CFM table. There are fields available for custom numerator and denominator.

Table (4.1) Calc Rule Master

Calc ID	FieldName	Calculation Type	Condition	Custom Calculation	Range ID
1	CBal	SUM	AND Olszie<>0	SUM(CBAL)/SUM(SBAL1)	NULL
2	FICO	NULL	NULL	NULL	1
3	AGE	DYNAMIC RANGE	NULL	NULL	NULL

Range Master (RM): The RANGE calculation [27] is used to set up an aggregation over a range. Example is FICO where the data should be aggregated by a specific bucket using the Range Values. There are two types of Range Values. One is a static range value like FICO and other type is dynamic. For a dynamic range the calculation engine will determine the range (min, max) with in each deal during the execution and the calculation engine will generate the ranges for the given attribute into four buckets.

Table (4.2) Range Master

Range ID	Field ID	Range Description	Min Value	Max Value
1	4	FICO >0 AND FICO <= 500	0	500
1	4	FICO >500 AND FICO <= 600	500	600
1	4	FICO >600 AND FICO <= 700	0	700
1	4	FICO >700 AND FICO <= 800	0	800

A sample user defined calculation is to find the FICO Score by the shown in the sample FICO Range Set up in the RANGE MASTER Table shown in the table 4.2 on page 57. The CalcId = 4 from the Field Master has a relationship with the FieldId in Range Master table giving the calculation engine the needed information to group by the field ranges defined. The Type RANGE in the Field Master table give the calculation further information to fetch the ranges for the given field. This is all done dynamically during the actual execution of the calculation.

Distinct Values (DV) or Key-Value: Holds all the distinct domain values for the attributes which we use for the calculation. Example for Servicer attribute, the do-main

values could be any qualified servicer to service the loan. The key value for the servicer 1 ST STATE BANK is currently 12 and for AUBURNBANK is 34. The KeyID will be used to save the final results. This Key-Value table is called Distinct Value Table (DVT) for easy reference. The table is show as 4.3 on page 58.This is the Distinct Values table.

Table (4.3) Distinct Value Table

Key ID	Attribute	Value
9	Collateral Group Name	2001-28-65
10	Loan Prefix	GN
11	Loan Prefix	FN
12	Loan Prefix	FH
13	State	GA
23142	FIELD DISPLAY	AtlInsurance _S ATA _C BRs

Attribute Master: This determines what domain values are needed to build the Key-Value table which is used by the calculation. This table has 3 fields Attribute Id, Attribute Name, Attribute active flag which can be used to suppress if we dont need the specific attribute in future. All the tables are designed with Relational Database Design architecture. The table is show in 4.4 on page 58. With our new proposed architecture, the Vendor File Extract is built on Microsoft SQL Integration Services (SSIS) which is an industry standard ETL tool. Individual file is parsed to an individual table (Stage table) by providing the source data copy for each calculation node. Node ID give the details in a supporting table which file is passed as variable to the SQL Stored Procedure. Process Flag N Indicates that the file is not assigned any Node or not ready for calculation yet.

Table (4.4) Attribute Master

Attribute ID	Attribute Name	Attribute Description	Attribute Active Flag
1	ColGroupName	Collateral Group Name	Y
2	LoanPrefix	Loan Prefix	Y
3	State	Stage	N
4	Servicer	Service Name	Y

Once the source raw file is available the entry in the Master Node Table, the entry

will be updated with the Source Table Name and the Node ID. Example if a source file (remic_FN_aa.txt) is available for the node 1 then the entry will look like the first row in the Fig. 10. With the source file name added to the table.

Summary Data: The summary table holds the final calculations done on the raw files and materialized the final aggregated results. The sample table layout is shown in the Fig. 11. The Result table is designed as Key-Value table following standard STAR [9] Schema. The Summary Data table was designed for the maximum flexibility and adaptability. The L1, L2, L3 and L4 will define the grouping for the Field ID (translated to the actual field using the Key-Value Pair). Date Key Id will bring back the data for which the calculation is been done. Like ASOF Data of the raw file. The N1 defines the values for that grouping level. We designed four numeric value fields in Summary Data table to accommodate all the calculation needs. We dont need any more than four numeric values to capture any MBS calculation. There is no need to modify this table design for any future expansions in storing any kind of summarized data with our design. The view created for the users and for BI applications bring the actual data values for reporting needs.

Table (4.5) Summary Fact Data Table

Fact ID	Date Key ID	Field ID	L1	L2	L3	L4	N1
1	1	1	5	10	2045	0	16547.98
1	1	1	6	10	2045	0	890147.03
1	1	1	7	10	2045	0	9812.17
857	3	3	231	10	2045	0	467512.71

The interpretation of the above Fact Table if we convert them to actual values will be as for Field Cbal (Field ID) the SUM value will be 1634223.2 (N1) for the collateral group name (L1) 2013-67-FN , Agency as FN (L2). L3 and L4 are used for Top values, CBRs, CPRs and CRRs. The L3 and L4 are multipurpose fields but values are functionally dependent on the calculation filed master calculation field set up. Range Master which was explained in details in the earlier text, is a very flexible and used to create the user defined ranges over any valid attributes. If a calculation has a range bound value then the range id will be saved

in L4 of the Fact Table. Users set up ranges on FICO score shown in Table 2 but users can also create another set of ranges for the same attribute. In the traditional calculation frameworks these ranges or any range bound values are created as additional attributes in the base table and used in the calculation (aggregation functions). This creates unnecessary columns. By our methods of dynamically generating these ranges at the calculation time, reduces the storage needed to store these as a new attributes and also give the ability to build new ranges without any DDLs (ALTER Table) to extend the existing schema. These tables are Big Data Tables. The other issue of extending with columns will be either character values or defined Key-Value pairs. If one wants to do it as VARCHAR data type for these new attributes (ranges) then better way is to use a Key-Value pair for the range value and convert to an integer based vector to use as the domain value for the attribute. The range can be defined with a CASE statement as shown in following code snippet for each tuple. This statement is generated dynamically by the calculation engine and saves an extra attribute to be created in the source raw factor table.

```

CASE
    WHEN (FICO > 0 AND FICO < 500) THEN 1
    WHEN (FICO >= 500 AND FICO < 650) THEN 2
    WHEN (FICO >= 650 AND FICO < 700) THEN 3
    WHEN (FICO >= 700) THEN 4
END AS FICO_BUCKET

```

The grouping by FICO_BUCKET is easy and also saves a lot of disk space in the base table when billions tuples of raw factor tables. There are several Range bound calculations and our method of dynamically generating ranges and use CASE to calculate is better by saving I/O reads as physical attributes. In the sample SQL {FICO_BUCKET} is replace by the CASE Statement at the execution time of the calculation engine as following SQL Statement to calculate the Sum of Original Balance (OBAL).

```

SELECT ASOF_DATE, LOAN_ID, {FICO_BUCKET}, SUM(OBAL) AS OBAL

```

```
FROM dbo.LOAN_LEVEL_DATA
GROUP BY ASOF_DATE, LOAN_ID, FICO_BUCKET
```

New Proposed Algorithm The new proposed algorithm is encompassed into two SQL Stored Procedures (SP). The two steps (Distinct Values, Aggregation) are done in memory to minimize the IO operation. The high-level process flow in figure 4.9 on page 62 and algorithm shown in the figure 4.7 on page 61.

(Start)

1. SSIS package is initiated once Factor Raw (**FR**) Files are available on Vendor FTP Site.
2. *For Each FR File (Run in parallel)*
3. Load *FR* Data to a SQL Table (Predefined SQL Table Name)
4. Generate Key-Value pairs which are new from the *FR* data
5. Generate Dynamic Range Value (key-value) Pairs for requested fields on the *FR* data
6. Insert the Calculation Request in Adhoc Date table with **Node ID** and **Source Table Name**
7. Materialize the calculation results
8. Mark the Adhoc Date Table entry as complete for the given source *FR* file.
9. *Next File*
10. Complete the Loop and process

(End)

Figure (4.7) Process Steps for an Individual Raw Factor File
High Level Algorithm for Overall process

The main algorithm at higher level 10 steps are shown in the Fig. 13. Start to End, the in between steps are performed on each Factor Raw Files (FR). The calculation engine is activated by providing an identifiable Node ID and a Source File Name on which the calculations are to be performed.

The proposed SQL Stored procedure with parameters is shown in the following figure 4.10 on page 62.

The first parameter is the source file table name (Stage Table) and the next parameter is the node id. The procedure will keep track of the activities on the node to report the completion to the parent SSIS package to mark its path is complete which means all the calculations for one individual file are complete. One raw factor file contains more than

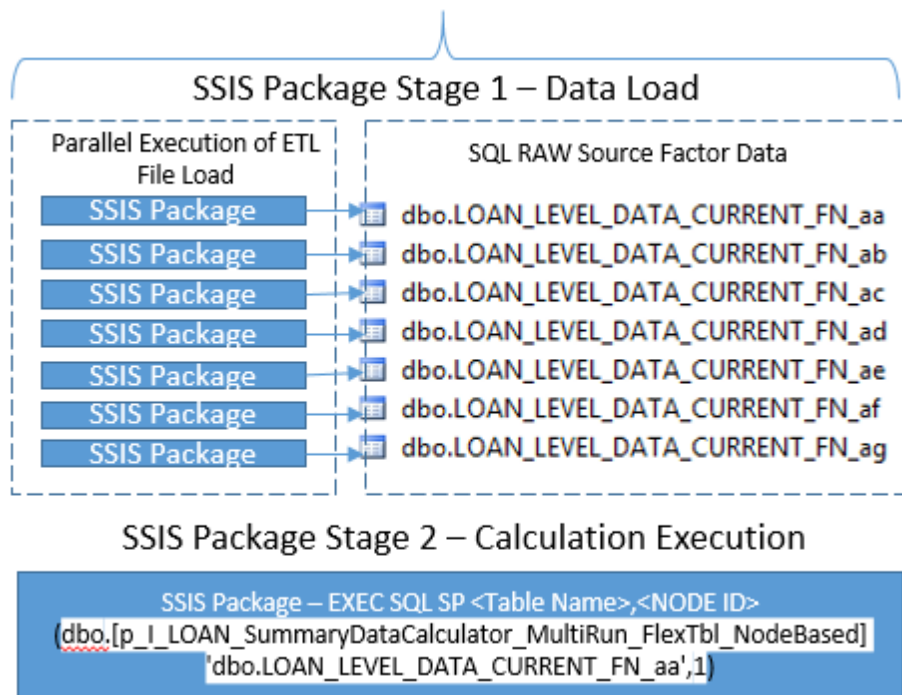


Figure (4.8) Parallel Execution of SSIS Package with the Node Attached Calculation
Parallel Execution of SSIS Package with the Node Attached Calculation



Figure (4.9) Process Steps for an Individual Raw Factor File
Process Steps for an Individual Raw Factor File

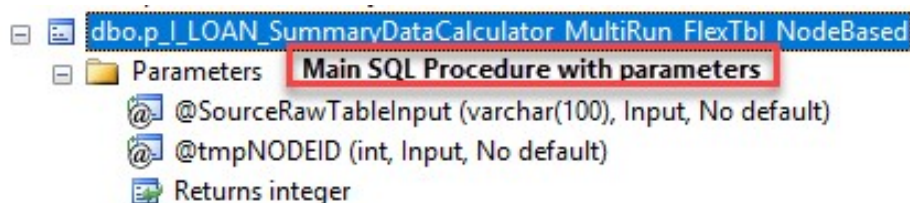


Figure (4.10) SQL Stored Procedure with parameters
SQL Stored Procedure with parameters

one deal. One deal can contain many individual loans. A deal is not split across multiple raw factor files. The one deal or loan information is contained in one source file and all the calculations are valid on that file to give a complete picture of for that Loan. The Raw

Data has 73 attributes for each file and there are 97 different calculations involved for each loan. The SQL Stored Procedure is build for flexibility to expand or add any new Calculation Types which are not currently available in the exiting code. The stored procedure **LOANData Calulator MultiRun FlexTbl NodeBased** is the main procedure which goes through each row in the Calculation Master to pick up the Calculation Type and generates a complex dynamic SQL based on all the different attributes from Calculation Master Table and executes the calculation on the stage data source. In-memory SQL tables are used to speed up the calculations and dropped once the calculation is performed and results are materialized. Our method of having a table driven flexible and easy to implement solution on distributed data source is better in terms of development and maintenance compared to the the SSIS package code for the calculations. The reasons why we used the SQL Stored Procedure for calculation implementation with distributed stage tables having the local data copy, over SSIS package implementation was discussed in our conclusion section.

4.5 Results

Our implementation is a big step up from existing application based on EXCEL version. As we see the EXCEL version cant handle some of the larger files as these files are contain millions of rows which EXCEL cant handle nor build for such kind of application or data processing. Our way of with simple and open architecture design and using the best in class (SSIS, SQL SPs) enabled the complete process to be done for approximately under an hour for 2,440 loans having 14,895,727 rows with total of 7 files. For a comparison purpose we did benchmark our process with a benchmark process (sequential process) implemented on a standalone on-premises SQL database before migrating that to Azure SQL Database in cloud. The timing on the same data set but with processing all the files into one huge table and then running the user defined calculation on that one table. The benchmark implementation is enhanced to have INDEX keys on the single table.

The benchmark implementation (Sequential Standalone on-premises SQL implementation) took approximately 3 hours to run calculations on the 14MM tuples (ID # 1 in figure

ID	Scenario Type	INDEX KEYS				Calculation				
		Architecture	Available	# of Keys	Time (Minutes)	# of Calculatoins	Time (minutes)	Total Rows	Type	Total Time (minutes)
1	Factor Raw Files load to a single SQL Table	On-Premises	NA	NA	NA	97	171	14,895,727	Single Instance	171
2	Factor Raw Files load to a single SQL Table	On-Premises	Yes	17	21	97	28	14,895,727	Single Instance	45
3	Factor Raw Files load to a Separate SQL Stage Table	On-Premises	NA	NA	NA	97	4.5	14,895,727	Parallel Multiple Instances	4.5
4	Factor Raw Files load to a Separate SQL Stage Table	On-Premises	Yes	17	12	97	4.5	14,895,727	Parallel Multiple Instances	16.5
5	Factor Raw Files load to a Separate SQL Stage Table	Azure Cloud SQL DB	No	NA	NA	97	51	14,895,727	Parallel Multiple Instances	51

Figure (4.11) Various Scenarios to test to propose a best-fit for organization
 Various Scenarios to test to propose a best-fit for organization

4.11 on page 64). In total we marked around 171 minutes for the benchmark process to complete. We then compared to our new process which the longest calculation path took 6 minutes which has 3.2 mm tuples and the table has NO INDEX Keys. The average completion time of 4.5 minutes which shows in the ID # 3 (parallel multiple instances of calculation) in the Fig. 16. Without any INDEX KEYS still the overall process took less time than the one we created for benchmark process. The benchmark process is an intermediate step of replacing the original EXCEL Solution. The ID #5 in the results show in the figure 4.11 on page 64 shows our Azure implementation with No Index Keys still outperform the single file single calculation instance with a speed up of approximately 58 Azure implementation did show a longer execution time than the on-premises parallel (ID3) process. This is due to the contract and service we have on Azure which is not totally comparable to the on-premises environment. Even though Cloud computing process is slower than the on-premises process, makes our case stronger to migrate to Azure due to the ability to grow the memory and the CPUs in the cloud. Our point is over a period of time, we do need more computational power and with cloud we can achieve that without any overhead on our existing infrastructure. Over the original EXCEL Process, converted to a standalone on-premises process (benchmark) our process is superior in all categories (ETL, Flexible new Calculations, History and BI Tool). The time saving is due to the distributed nature of the calculation engine

which can be run in parallel over multiple source files.

For the results it is very clear that ID #3 which is our preferred implementation outperforms any other implementation. The Azure implementation ID 5 can outperform our preferred implementation of on-premises distributed implementation if we had gone to a higher service contract. Having higher DTU on Azure Cloud we can conclude that our implementation of distributed framework can outperform on-premises data architecture and also save money by not having hardware on-site.

4.6 Conclusions

As expected, the new method has a superior performance compared to ad-hoc approach on EXCEL based VBA. The EXCEL cant even be a solution in an enterprise environment. This can be a serious problem from an operational risk. The implementation of the benchmark process on standalone on-premises SQL server gave us the foundation and helped us in evaluating our new cloud computing solution for MBS factor calculations. By migrating to a Microsoft Azure Cloud Environment, we were able to solve the expandability and flexibility for future data analysis needs. Even though the cloud computing did not outperform the parallel and distributed solution we had as benchmark over 12 CPU, 45 GB RAM, it gave us a very good understanding of the future needs for Azure Cloud DB requirements. If we can match our license to the existing on-premises we can achieve the same speedup and also can save computational

costs. Example to match our current needs we would go to the next level of Azure Subscription will cost around \$4.056 per one-hour computational need which at our current parallel timing, we will spend less than \$100 per monthly calculations. Our standard license of S3 on Azure which gives a limited number of CPUs and RAM memory. With more resources (DTU and CPUs) we can achieve a higher speedup using Azure. For an Azure implementation, all we need is a higher service contract which provides us more computational power. With more computational power we can bring down the final calculation time to near 4 minutes to match the on-premises 12 CPU 45 GB RAM simulation. Also,

our implementation shows the solution on SQL Server and using the Cloud architecture to address any future space related issues can be solved easily by expanding more vCores, CPUs. Shown in Fig. 17. The implementation is to spread our calculations on distributed files (tables) and running the calculations in parallel on the Azure SQL Cloud Environment. As a proof of concept any investment company can utilize this study to build their next generation investment platform using our simple framework of distributed computing and achieve a tremendous performance improvement over any legacy applications.

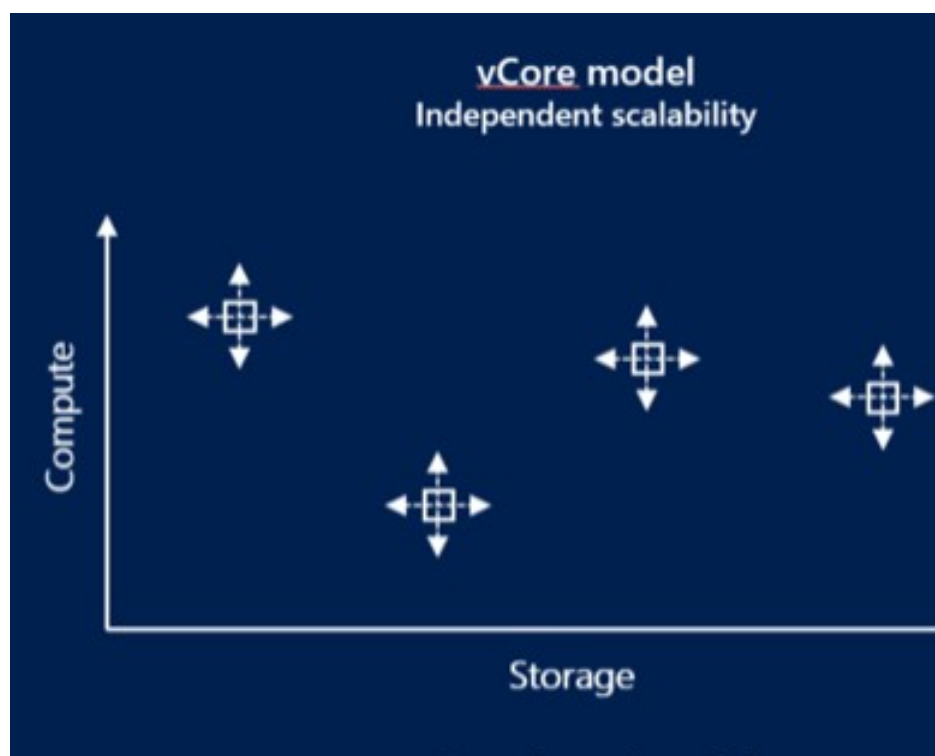


Figure (4.12) Microsoft vCore Model to match on-premises model
Microsoft vCore Model to match on-premises model

There are very strong reasons why we implemented our user defined calculation through a SQL Stored procedure rather than a SSIS ETL package. The first reason a) SSIS is efficient when the calculation is at a tuple level. It complicates when it has to bring in a block of tuples and to run a calculation. On the other hand Stored Procedure can be done at tuple as well as a set of tuples. Second reason b) In SSIS it is not possible to build a dynamic range unless all the tuples are available. MIN and MAX cant be determined to build range values

with partial tuples. SQL procedure can handle this as the complete data set is available in the stage tables in form of Raw Factor Data and can be done for all loans efficiently by grouping and breaking the tuples between MIN and MAX at an individual loan in one pass. Third c) the portability of the SSIS is limited as it needs Microsoft Developer tools. Whereas our SQL Stored procedure is portable to any standard SQL instance running with ANSI Standard SQL. This is major win as we can port our code to any environment either on premises

SQL instance or Cloud Computing SQL instance. The fourth d) Maintaining a SQL Stored procedure is simpler than an SSIS and the skill set required to enhance a SQL Stored procedure is much widely available than SSIS coding. These are our main reasons we decided to implement our solution on SQL Stored procedure driven calculation engine. In future we are going to implement the ETL load with Hadoop and MapReduce to have the portability for ETL also.

Even though the on-premises implementation showed better results, the disadvantage of this is an overhead of hardware and software costs on our organization to maintain this solution. We recommend the Azure solution with a higher subscription to gain the performance speedup and reduce over all technology costs in maintenance hardware and software on-premises.

4.7 Future Work

We are extending our architecture for Data Mining to find trend and identify market events and alert our Risk and Portfolio Managers. This we are calling it as Early Warning Analytical Trigger System (EWATS). Data aggregation is a key building block in our Data Mining [8] and Machine Learning financial applications [20]. The biggest work we are undertaking is to implement the data load process using the Big Data ETL tools using Hadoop and MapReduce. By migrating the existing SSIS processes over an open source platform will make our solution portable to any cloud environment. We are in processing of scoping the project and outline the needed research components to formulate a widely used open

source solutions for other researches to be able to leverage from our case study. Even though we wanted to try and use the Massively Parallel Processing (MPP), due to time constraints we were not taking advantage of MPP. This is slated as future development requirement to give us more computational power by implementing over Hadoop [16] and MapReduce. Our implementation did give the foundation for future Distributed Cloud Computing Framework for Financial Applications. We are attempting a larger data set with approximately 450MM tuples and 36 files.

Microsoft Azure Cloud Platform also provides Massive Parallel Processing (MPP) processing of files and computational node architecture. This is another area we can invest and use the Azure Data Warehouse architecture to see if we can leverage Microsoft cloud technologies to further build our Distributed Financial Cloud Computing Environment.

We are reaching the vCore model of Azure Cloud computing shown in the figure 4.12 on page 66 to scale our performance and to match and outperform the existing on-premises model. This will cost initial investment in terms of resources but our paper and our case study, proof of concept (POC) that our Cloud Computing framework will work makes it very convening to our senior management to embark on expanding our computational capabilities.

CHAPTER 5

REPORTING FRAMEWORK

5.1 Introduction

End user reporting in an organization is a key component in a FinTech application roll-out. Most of the implementations in several technical solutions, usually focus on the computational models, data storage and development tools of business requirements. We found gaps in capturing the end user reporting needs of an organization. End user reporting consists of day-to-day management reporting, standard regulatory reporting and various visualization dashboards which will serve various management levels in an organization. Business to be nimble for change, they need tools which can adapt quickly for the continuously changing nature of business to sustain in the competitive world. Some of the reporting needs may not need an official book of records data but other reporting requires a strong data governance dependent reporting to satisfy various regulatory and auditing requirements. Lack of standard reporting tools across the organization with well defined reporting governance structure creates a big gap in institutional reporting needs and cultivates mushroom growth of various reporting tools across the organization. The growth of individual reporting tools will pose a huge enterprise architectural problem to the technology groups and also from a data governance perspective which can lead into auditing and regulatory breakdown which can lead to business interruptions and head line risk for an organization. This risk (operational risk) of not able to standardize reporting frameworks at enterprise has a downside risk of not able to attract new clients and potentially lose the existing business. Lack of enterprise reporting frameworks cause organizations struggle to be competitive in their domain of business. In our paper we propose an easy to maintain and well controlled reporting framework model which we call **Enterprise Hybrid Business Intelligence Model (HBIM)** which can work both for business needs as well as fits into

the broader enterprise technology architecture to support the business in terms of growth and retaining and existing client base. Our proposed architecture looks into various business needs and suggests a hybrid reporting architecture consisting of best in class industry strength tools which can be supported by technology team reducing the operational risk footprint of an investment organization. A little of this

In real world it is highly unlikely that every organization can institute a single tool to solve an institutional problem. For some problems organizations will decide to implement a single solution rather than multiple application solutions and in other instances there may not be possible or not efficient to impose a single solution. For example trading systems which an organization would like to consolidate on to a single trading platform but for data storage and database applications the organization may deploy more than one database solution depending on the need and necessity to address the problem. For multiple databases, some times the vendor solution may work optimally on a specific database compared to an organizational preferred database. In that scenario it may deploy multi-platform solution rather than one. The same approach will be taken when it comes to reporting tools.

In an organization reporting happens at various levels. Each level of employee group or functional roles are assigned specific reporting roles. An investment organization [28] is no different in terms of reporting needs compared to any organization. There may be several regulatory filing are required being a financial company which manages several clients funds to help them in making better financial decisions. Investment organization can manage funds on behalf of individual clients or can manage at an institutional level. Example for an institutional client money is managing retirement funds of the clients employees. An investment company can be hired by a corporation to manage their employee benefits and one of the benefit plan can be employee retirement plan. When employees contribute to their 401k fund the company can outsource the management of the fund money to an outside investment management company to invest to make good on the corporations obligations to employees when it is time for disbursing the funds. Liability management is critical depending on the cash flows. There is a lot of reporting usually needed between the fund

manager and the fund owner on a agreed upon time frame. Reporting can be classified into several categories. The main categories are shown below and the list is a demonstrative than a comprehensive. we can broadly divide the reporting into two major categories. 1. Standard Reporting (Internal and External) 2. Early Warning Reporting. Standard reporting includes all the regulatory filings, reports needed by the clients on a fixed frequency or adhoc basis, portfolio and risk management reports to manage the funds, daily settlement and cash flow reports for operations to support the day-to-day operations of the fund. The following are few standard report categories which a business needs and expects to be available on day one of the fund launch.

1. Regulatory Reporting
2. Risk Reporting
3. Financial Reporting
4. Client Reporting
5. Compliance Reporting

Apart from the standard reports the organization sets up some guard rails to prevent any failure which can incur in losses and can hurt the reputation of the fund manager. These reports are called trigger reports or early warning reports. These reports are more dynamic in nature and are driven by the day to day transactions done in the fund portfolios. These trigger reports and early warning reports can give indicators to the investors, risk managers and compliance teams a prior warning on set metrics to adjust and if possible to avoid any breaks in the mandate agreements which a client and fund managers agree upon to manage their funds. These mandates are legal contracts which all parties to be binding during the period of the agreement. For an investment management firm to lose a mandate is like losing a client and can some times lead into reputations risk.

When a Financial Technology (FinTech) solution is been proposed usually the reporting part becomes a secondary need unless business users identify and the project sponsor pushes

the need for the reporting component as part of the solution. When a project is rolled out without proper reporting requirements collected, the users struggle and invest a lot of time to migrate the existing reports generated from legacy applications or stranded without any proper reporting mechanism with the newly implemented FinTech solution. In our paper we will address the inherent drawbacks on implementing a FinTech solution without proper reporting framework and especially when the industry is moving towards the cloud based architecture, the reporting functionality and frameworks are critical for the success of the organization.

This chapter is organized into six sections. The background section gives a brief background of the industry reporting needs and layout the problem for the next section which we discuss the problem statement. In current problem and related work we will highlight the existing reporting tools and compare their strengths and weakness and set the foundation for our next section which is proposed solution for enterprise business intelligence reporting framework. In the Implementation section we will show the actual implementation of our proposed solution. In the observation and comments section, we will compare our baseline reporting set up with the newly proposed and implemented solution. Comments section will capture our overall enterprise reporting framework and discuss the lessons learned during our journey in implementing an industry strength enterprise business intelligence solution on cloud computing environment. The Future Work section will layout the gaps which can be fill in for later research and implementation.

5.2 Background

Reporting is a key component of any business [29] which make the internal processes transparent to the outside world. With the regulatory filing by a corporation, public can learn about the company and the financial strength of the corporation. This is critical for investors to know when they are investing into the company bonds or stocks. With out the regulatory filing it will be difficult to any investor to understand the company structure and risks associated with the company. These regulatory filing are very important and they take

a lot of time and resources for a company.

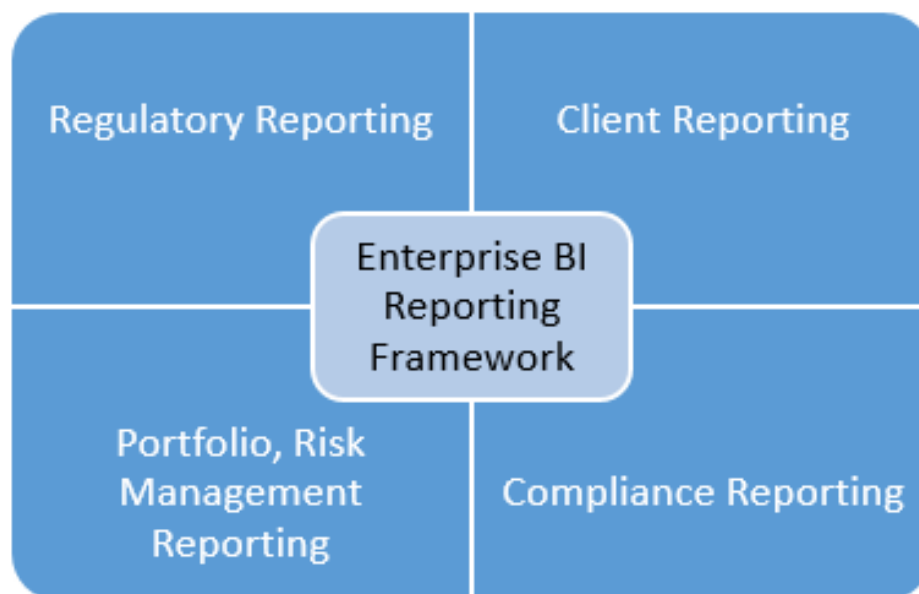


Figure (5.1) Organizational Reporting Requirements
Reporting components needed for an organization

Usually the regulatory filings are on a quarterly and yearly basis to various government entities. As shown in the figure 5.1 on page 73 we can see our approach should be a complete reporting solution sitting in the middle and able to service all the constituents of the organization. For simplification, we did not add all constituents like operations and infrastructure and IT reporting but they are part of overall reporting framework. As we mentioned in our introduction, there are several reports which are standard and there are several reports which are meant to help as early warning. Apart from the early warning reports, organizations would like to have a dashboards to show the current status of the investments. These dashboards are critical for senior management for decision making. One example can be *"Daily Investment Risk Dashboard for Fund : ABC"*. If the FinTech solution is proposed to implement a new trading system, or a data warehouse for investment platforms, the dashboard design [30], delivery and maintenance should be thought through. This should be a part of deliverable else end users will be putting in lot of effort in building such dashboard to from the data set or from vendor applications which sometimes

are not very flexible to create custom reports or dashboards. One of such example can be shown in the figure below.

The screenshot displays a Bloomberg Ticker interface with several data sections:

- Header:** Includes fields for 'Bloomberg Ticker', 'Company Name', 'Country Code', 'Sector', and 'Industry'.
- Summary Table:** A table with columns for 'Metric', 'Value', and 'Unit'. It lists various financial metrics such as 'Current Balance', 'Original Balance', 'Yield to Maturity', etc.
- Historical Information:** A table with columns for 'Date', 'Metric', and 'Value'. It shows historical data for various metrics over time.
- Main Data Table:** A large table with multiple columns, including 'Field', 'Value', 'Unit', and 'Description'. It provides detailed data for various fields related to the mortgage pool.

Figure (5.2) Organizational Reporting Requirements Reporting components needed for an organization

As we see in the figure 5.2 on page 74 we can see there are several components needed to build this dashboard which is used by the investors to see various metrics on a mortgage pool of assets. To preserve the deal data the ticker field and the fields relevant to identify the ticker are masked. If the FinTech implementation is not including the details or the requirements to address the need of the business uses, the implementation won't be successful and organization needs to invest large resources after the initial implementation of the initial project.

Sometimes the project implementation teams will ignore or overlook this reporting need and end out with several individual tools to address an overall problem which can become a home grown tools and reports without a proper control structure or support model.

To address this we propose a **Enterprise Hybrid Business Intelligence Model (HBIM)** where several reporting tools can co-exist and can help the business at various levels of reporting. This model we deploy on Azure cloud so we can efficiently utilize other tools Microsoft Azure [31] can offer. As an organization if we can't standardize the reporting framework it will lead to a massive rework when the groups or organizations migrate from

one system to another system which may not support the tools on which the legacy reporting modules. It is critical at the initial phase of the project to identify the tools needed rather than at the end of the projects. In Investment Management world there are so many variations of the reports which may need an intensive review of the requirements as well as a road map to accomplish them with a planned and sustainable manner.

The other wrinkle is to come up with an overall reporting framework to support all the constituents of the organization. With this background we will articulate our problem statement and will propose a suitable BI reporting framework on cloud computing environment.

5.3 Problem Statement

As we see in the figure 5.2 on page 74 we can see there are several components needed to build this dashboard which is used by the investors to see various metrics on a mortgage pool of assets. To preserve the deal data the ticker field and the fields relevant to identify the ticker are masked. If the FinTech implementation is not including the details or the requirements to address the need of the business uses, the implementation won't be successful and organization needs to invest large resources after the initial implementation of the initial project.

Sometimes the project implementation teams will ignore or overlook this reporting need and end out with several individual tools to address an overall problem which can become a home grown tools and reports without a proper control structure or support model.

To address this we propose a *Enterprise Hybrid Business Intelligence Model (HBIM)* where several reporting tools can co-exist and can help the business at various levels of reporting. This model we deploy on Azure cloud so we can efficiently utilize other tools Microsoft Azure [31] can offer. As an organization if we can't standardize the reporting framework it will lead to a massive rework when the groups or organizations migrate from one system to another system which may not support the tools on which the legacy reporting modules. It is critical at the initial phase of the project to identify the tools needed rather than at the end of the projects. In Investment Management world there are so many

variations of the reports which may need an intensive review of the requirements as well as a road map to accomplish them with a planned and sustainable manner.

The other wrinkle is to come up with an overall reporting framework to support all the constituents of the organization. With this background we will articulate our problem statement and will propose a suitable BI reporting framework on cloud computing environment.

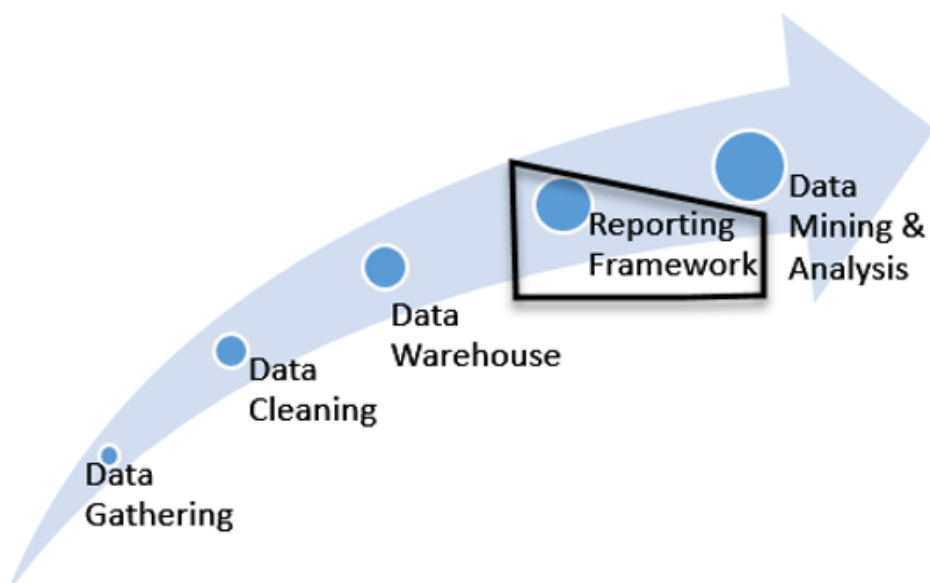


Figure (5.3) Reporting and Data Mining Framework Evolution
Reporting and Data Mining Framework Evolution

As shown in the figure 5.3 on page 76 we usually start the data warehouse build out from the data gathering stage which involves various data sources. The data cleaning is another step in the process of building a good data warehouse. Once Data warehouse is build usually we will have data marts to support individual groups or roles. End users will be using the data from data warehouse or (data marts not shown in the figure) using various reporting tools. Some examples of the current reporting tools in or environment are EXCEL, SSRS, WEBFocus, Tableau, Microsoft Access, Power BI [32] MATLAB, .NET Framework [33] etc.

Lack of a standard [34] reporting framework which can be used by business and is well supported by IT architecture and development teams causes the confusion within business users and they tend to use what they are familiar with rather than the best tool to solve

their reporting needs. With this approach we are adding new tools without proper vetting and building a standard approach for enterprise reporting framework. As we go with this paradigm, we will soon hit a road block where business would like to use a tool they are comfortable but can't fit into the overall enterprise architecture which needs to be supported by IT and other teams. This leads **inefficient allocation of resources, data breaches, longer time to resolve issues, database design to fit the tools which will eventually become too big to manage and maintain.**

We want to address these problems and bring a proper framework which will pave a path and gives a clear road map to the organization in terms of reporting needs. The reporting needs as mentioned in earlier sections it not just standard reports generated but also include dashboards, data mining applications and other decision systems used in financial management of clients funds.

5.3.1 Related Work

So far as we mentioned there are several tools deployed in our business environment. These tools evolved over a period of time. There are various vendors who are big players in reporting space compete to win the market share. The big players in current market conditions are shown in the following research conducted by Gartner Inc. Information Dashboard Design [35] is an art. This needs a strong organizational reporting framework.

The other useful and very informative study was done by Gartner Inc.[36] The research plotted the BI tools on five different categories to come up with what they call as a magic quadrant on BI tools. All BI tools are not created equal!. Gartner picked up 5 use cases across 15 critical capabilities of Analytics and BI platforms. The heat map shown in the figure 5.4 on page 79 shows us the score across the top 20 vendors included in the study. We can notice as we go on across the spectrum we can see the 5 use cases they picked up. The following are for the Strategic alignment of the organization which should be taken into consideration when rolling out an organizational BI tool on Big Data and cognitive computing [37]. Apart from the high level criteria they categorized critical categories into 5.

- Agile Centralized BI Provisioning
- Decentralized Analytics
- Governed Data Discovery
- OEM or Embedded BI
- Extranet Deployment

The critical categories are 1. Infrastructure 2. Data Management 3. Analysis and Content Creation 4. Share Findings 5. Overall. For example under Data Management they considered Scalability and Model Complexity as one of the critical criteria to be evaluated. Similarly there are in total 15 critical items on which each vendor tool was evaluated.

The figure shown in the 5.4 on page 79 gives us an idea of the top 20 market players. Each of the products/services has been evaluated on the critical capabilities on a scale of 1 to 5; a score of 1 = Poor (most or all defined requirements are not achieved), while 5 = Outstanding (significantly exceeds requirements).

From the figure 5.4 on page 79 we can see the leaders and challengers along with the visionaries and niche player in the BI market. Within the 5 use cases the 15 critical capabilities which we are not going to go through but with our current needs we can use this report as an input to our own vendor selection process.

Apart from Gartner research there are several research papers which looked into the reporting frameworks and they are more geared towards dynamic dashboard design [38] and building a development platform for businesses. One of them is SQL Server Reporting Services (SSRS) [39] which is a Microsoft Product integrated onto MS SQL Server to develop industry strength reporting packages. Apart from the SSRS another product which plays in the niche space is Information Builders (IBI) WebFOCUS [40] too. The architecture of IBI serves at various modes like single-tier to multi-tier. After going through a good research work and vendor product literature, we propose a ideal solution and a road map for an enterprise



Figure (5.4) Magic Quadrant for Analytics and Business Intelligence Platforms
 Magic Quadrant for Analytics and Business Intelligence Platforms

reporting framework. Most of the implementations as we see will have a developer, user, administrator profiles to manage and develop and deliver the reporting content.

5.3.2 Proposed Solution

After going through extensive inventory and user interviews, we found that in our organization there is a lot of ad-hoc reporting as well as standard reporting. The proposed solution is a hybrid solution which can be used on cloud computing environment as part of our overall enterprise architecture. Users can use the existing tools which they are comfortable but IT, Architecture and Business Teams which are familiar with the latest BI tools will do a proof-of-concept with new new tools like Power BI to enhance the existing reporting tools. the follow diagram will guide the uses to pick what tools are best for their build out. This is not available in our current environment. This will give us the ability to come up with a better support models to business needs and also this road map will be a part of any FinTech projects the organization will take on. Each architecture has a defied way they operate. Each model at high level will have 3 major components. 1. A server component 2. Developer Component 3. User viewer component. They can be called differently in each tool. The server component will be the core to maintain the reports repository, schedules, metadata, business rules and any hardware setting for the installation. The Developer component provides the needed tools to build the actual reports. The User Viewer provided the presentation layer. The User Viewer Layer [41] helps the end users to run the reports and export to different formats (PDF, EXCEL, PowerPoint etc.). We are taking advantage of these key 3 components of each BI Tool and proposing a road map to the organisation to come up with a flexible model to implement our idea of an Enterprise Reporting Framework which can be deployed and can work seamlessly on Cloud Computing Environment. Which is our end goal as the business is going towards Cloud Computing environment.

The new proposed model will educate users on why to use the tools mentioned in the figure 5.5 on page 81 showing the right tool for the right report. A little bit on the WebFOCUS reporting object here as we are proposing this option for ad-hoc and User Run reports as this model will be maintained and developed by a central development team who will be in charge of the metadata management. **Reporting Object (RO) in this context is a metadata view defined with business rules embedded.** This is carefully developed

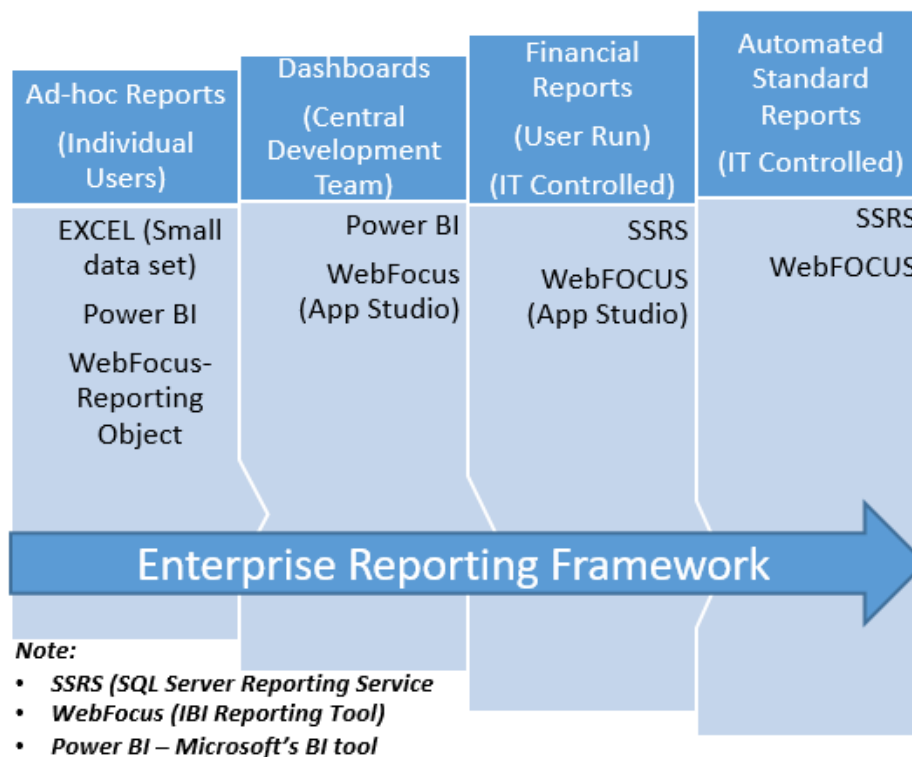


Figure (5.5) Proposed Reporting Framework - Need based Model
Proposed Reporting Framework Dependent of Data usage

in consultation with the business users who are subject matter experts on the domain data coming with business rules to expose the data as selectable data fields for end users. End users who are using this RO can then build their own reports based on the metadata defined by the RO. The advantage of this model is, we define the business rules only once and only one place and the organization can use them across without any ambiguity in their reporting. To have this model work, we need to have a centralized data governance team overseeing the business rule set up and approvals. The model defines a set of rules when to use which tool as shown in the high level figure 5.5 on page 81. Power BI [42] is a very powerful visualization tool which can be leveraged by the organization for any ad-hoc, presentation based reporting. Microsoft Power BI [43] has lot of features and also getting enhanced continuously to compete in the BI Space.

The financial and client reporting which depends on a controlled and signed off data should be coming from IT Development group which will work hand-in-hand with the busi-

ness to develop them and put in a production environment where users can run them but can't edit. This model is already in place and we are proposing it to be formalized and build a dedicated team to support the organization on these initiatives. The key here is the resources to be deployed to build the controlled reports in order to work this model. If we can't source the centralized team with sufficient resources, we proposed to designate an individual or individuals in each business team to be able to build these controlled reports and IT Development teams can migrate to a production environment. This still serves the purposed of the controlled reports which users can run but can't change unless they approach the teams who developed them in the first place. If there is a need to enhance or edit these reports, the owners will edit and will send them to production after testing and signing off on the report in test. For our implementation of enterprise solution we tested the concept on two different environments. The first one is on-premises where our data sources are all on hardware which we maintain and located on our location. The other environment is data sources on Cloud for which we used Microsoft Azure Cloud Environment [44]. This second model in fact consists of the BI Tool (Power BI) as a service SaaS (Software as Service) which is the model we would to see as our overall organizational strategy to deploy our Reporting framework. The advantage of setting up the on-premises is gives us the benchmark when we migrate our services to cloud computing environment.

5.3.3 Current On-Premises set up

In our current set up we used all the tools with on site set up. EXCEL is used to gather data from onside SQL Servers to build the existing reports. The set up is not new as we have legacy reports with ODBC connections from EXCEL to SQL Servers to pull information via VBA (Visual Basic for Applications) which act as data collection program and build upon the data set retrieved from the SQL Server. This we want to test when the SQL on-premises server moves to Cloud and run the same report. We also identified some reports which are build EXCEL which should be done via a industry standard reporting tool like SSRS, WebFOCUS [45] or Power BI to see how we can migrate those standard reporting tools as

a use case study to cloud computing environment. The cloud computing environment [31] here is AZURE Cloud Services provided by Microsoft. The sample implementation is shown in the figure 5.6 on page 83 which gives us an idea that all the data sources are either picked up for the top layer reporting tools no matter whether they fit the need or not.

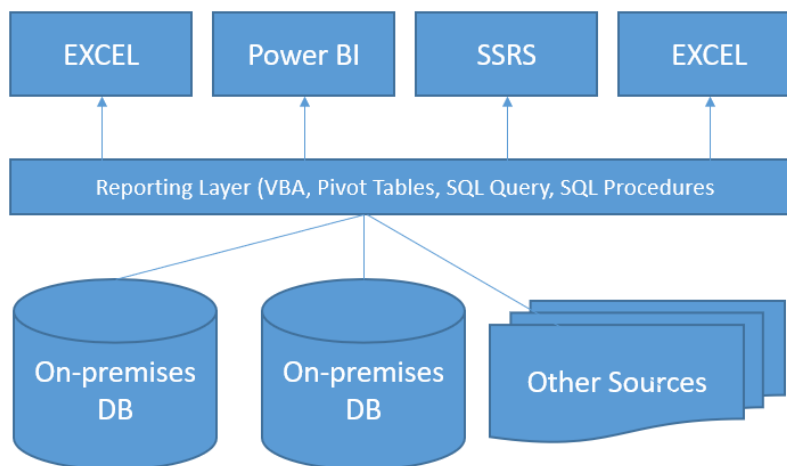


Figure (5.6) On-Premises implementation model
On-Premises implementation model

For on-premises implementation there is no special use case but we used as-is-scenario how users are currently using the reporting tools. In the figure 5.6 on page 83 is an example users using EXCEL and an SP which brings millions of rows is still considered a valid reports even though the performance is not good. Ideally the EXCEL is not supposed to be used as data store but to display aggregated values or use the data in the form of EXCEL Pivot tables. We will demonstrate the same set up but on Azure using Power BI as a tool to integrate into EXCEL but not bring back the actual rows but just bringing the needed data set via the power pivot add on. The other set up in the on-premises model is users use several files to build data driven applications. In the current environment we implemented the tools apart from the standard EXCEL tool. Some of the users are given SSRS tools with access to the on-premises SQL data base access. The users who are given SSRS tools are more advanced then others and also the teams are equipped with the needed SQL and SSRS development skill set. There is another group of users for whom we provided

WebFOCUS developer tool kit to be able to connect to the same on-premises SQL Data sources. Both groups are given the needed tools and support where needed by our IT and Infrastructure teams. SSRS Server install was done one dedicated server with servers as the gateway for all SSRS Reports. WebFOCUS was given a dedicated windows server to have the server components and scheduler components. The Environments are replicated as TEST and PRODUCTION to create an enterprise environment with proper controls to mitigate operational risk and also able to be audit proof for internal auditors as well as external auditors.

The implementation is show in the figure **5.7 on page 84** which make it easy to understand our user classification by the tools they use in-order to accomplish their reporting needs. The users are encouraged to build and develop their reports as per their skill set and with the tools they are familiar and provided by the shown Venn Diagram. During this implementation we noted the any issues our users faced and investigated if they are related to the tool or the combination of the tool and data.

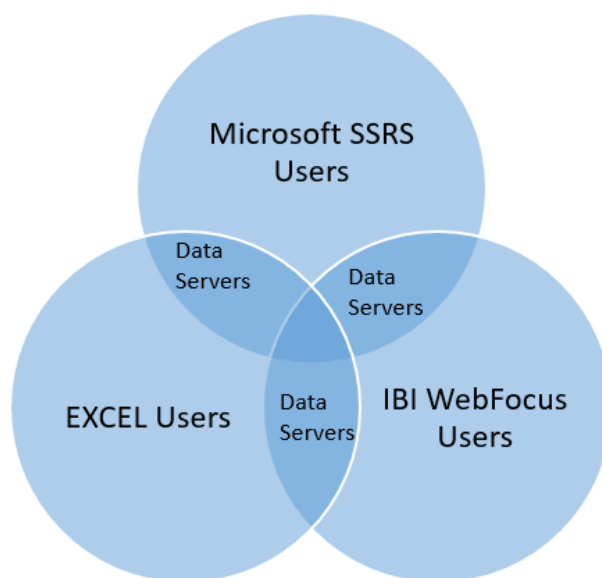


Figure (5.7) On-Premises user categorization
On-Premises user categorization

5.3.4 *Cloud Implementation*

In the implementation of the cloud, we migrated the **Data Servers** shown in the figure **5.7 on page 84** to Azure Cloud. With our Cloud license, we were able to set up an enterprise access with our Active Directory accounts set up. The process is still in progress but we are successful in proof-of-concept of Database migration to Cloud Computing environment. Users were able to access the same data sources via cloud using the same tools. This is made transparent to the users so they are not aware the location of their source data. To make the SSRS and WebFOCUS work, we need to make sure we have the same connectivity to the cloud based data bases [?] via the native connectivity controls. In case of SSRS and WebFocus we can use the native SQL Driver to connect to the database. These connections will work as the data transporters. In case of EXCEL we created the ODBC connections to connect to the database. In few instances VBA was used to generate dynamic SQL and gather the data from the Azure Cloud Computing environment.

The set up is similar to the on-premises with users able to use the tools they are familiar but their data is now on cloud. Our Hybrid Model of tools we tested and noted the user experience and determined the which tool will be the better choice to fit for their reporting needs.

For the cloud implementation we did sign up for Microsoft Azure Account [17] and migrated the databases we need on to cloud with test data. Each user for Power BI was provided a test license for 2 month period for testing. All the other tools where we have licences are already used on the on-premises implementation.

5.4 **Observations and Comments**

With the two sets of implementations (On-premises and Cloud) we noted our experience with our newly proposed enterprise hybrid model which is shown in the figure 5.5 on page 81 which fits perfectly to solve our problem. We were able to demonstrate the model of having multiple reporting tools are not going to impact or over load the our tool space but actually

brings an awareness among users what tools to use and how they can improve the productivity by using the right tools for the right job. Reporting is a complex activity. The various results and comments are tabulated in to the table shown below.

Tool	Data Source	Comments
EXCEL	Another EXCEL Sheet(s)	<ul style="list-style-type: none"> • Easy to use as users are very hands on EXCEL tool • Not IT Controlled • Quick and Ad-Hoc reporting only
EXCEL	SQL Data (on-premises & Cloud)	<ul style="list-style-type: none"> • Create ODBC to connect • Easy to pull the data • Slow for heavy data sets • Heavy calculations should be done on server
SSRS	EXCEL , CSV Files	<ul style="list-style-type: none"> • Not idea as there is meta data set up needed • Can be used but takes lot of time to build reports • Not to encourage • Not controlled or supported by IT
SSRS	SQL Data (on-premises & Cloud)	<ul style="list-style-type: none"> • Create data connection. Easy to maintain at server level • Integrates very well with SQL Server • Ability to create SQL Stored procedures and out put the needed data set • Can create various data sets in a single report. • IT Controlled and should be encouraged for Standard Reports. • Scheduling functionality available with various output format.
WebFOCUS	EXCEL , CSV Files	<ul style="list-style-type: none"> • Meta Data (master file) set up needed • Easy to use but not suggested as maintenance will be tough • IT Controlled but loses the ability by end users to modify the layout of the source file one the metadata is set up.
WebFOCUS	SQL Data (on-premises & Cloud)	<ul style="list-style-type: none"> • Create a data source connection like SSRS on the sever level • Easy to use and lot of functionality for controlling the reports • IT Controlled and suggested for Standard Reports and Dashboards development • User portals can be created and published on SharePoint of Webservice • Extensive scheduling capabilities. Easy for administration
Power BI	EXCEL, SQL Data (on-premises & Cloud)	<ul style="list-style-type: none"> • Very powerful visualization tool • Impressive user friendly layout. Need good designer if data size is huge • Low cost licensing and Reporting sharing functionality • Idea for end user ad-hoc reporting and quick analytical reports. • Good integration into EXCEL.

Figure (5.8) Tools - Data Source Recommendation
Tools - Data Source Recommendation

Even though the analysis posted is at very high level we are confident that the model we are proposed *Enterprise Hybrid Business Intelligence Model (HBIM)* works well with in our organization. The ability to use various tools in the our ecosystem is perfectly viable without any overload on the exiting infrastructure. The key is to identify the full extent of the user reporting needs upfront and provide the categorized tool deployment to achieve the goal in any FinTech implementation.

5.5 Future Work

Even though we did this as proof-of-concept (POC) there are several areas we can improve and enhance our reporting framework road map. One area is to enforce a proper data governance framework when users are using BI tools. One example is the Power BI use as a tool to share report with other users. For this to work well we need to integrate into our Office 365 framework and also give users the professional license. Even though the cost is minimal for the standard desktop version to professional version, the control of the data flow needs to be understood. If the users can't access the data via the gateway there should be proper triggers to indicate a possible violation of the information. Sometimes this can lead into a compliance violation if an individual can access data he or she is not supposed to be able to view. The other area of future work can be done on setting up separate development teams to support the business. This model costs resources but in our view this will work well as the organization will build the knowledge centers for various reporting tools. This will give the company a big competitive edge. This has a potential upside but there is an initial cost for this model to work well within an organization. This **Central Reporting Team** build should have executive sponsorship and the composition of the team should be both business and technical skill personal. It won't be successful if it consists of only one type of resource. This is an area we would like to extent and come up with another POC to see which model can be optimal for our organization. The other area of interest is cognitive computing [46] which can lead to many more application areas.

CHAPTER 6

MACHINE LEARNING FRAMEWORK

6.1 Introduction

Financial companies came a long way to build massive data driven FinTech applications incorporating data mining capabilities via on-premises databases which now are in process of migrating to cloud computing environment. Moving to cloud, solved the computational power and space constraints organizations are bogged for a long time are now turning their focus on to intelligent decision making using Machine Learning to predict the best possible investment strategies or trade ideas which can make profit to the clients. If machines can discover patterns in the trade data, market data indicators and suggest the optimal or best trade(s), suggest portfolio balancing, suggest correction in the portfolios, trigger early warning on a set rules will certainly help investment managers. In past decade industry started focusing on artificial intelligence to solve some of our problems by teaching the computers to think and act like humans. By teaching the computer to think like a trader or a portfolio manager enable the organizations with more innovative products and make the money manager marketable and gain client base or retain the existing clients. Even though there is good research on machine learning it is more into engineering and industrial areas but not much into financial domain. Our research and proposal of an Enterprise Architecture for Machine Learning will help financial organization to set up a their own Machine Learning framework with easy using Microsoft Azure Machine Learning Studio. We believe having our research published in public domain and in academic forums will advance the Machine Learning and Artificial Intelligence implementations in financial domain. Our proposal will be a solution on Azure using the Microsoft Machine Learning Studio which is easy to implement and well documented with good support model and scalability for future development.

Decision making in financial industry largely depends on the data analysis and intuition. Any decision a portfolio or a risk manager takes depends on the individual experience and the data analysis the person can perform to come up with a meaningful and explainable decision for the problem they are trying to solve. The problem can be a complex optimization of a client portfolio re-balancing with certain constraints or an analysis of an individual company future profitability so they can invest to reap the positive returns on their investments. During this process the individuals go through a set of steps with the data at hand and decide a logical path which ends with a solution. nowadays with the amount of information Big Data and the complexities of identifying useful information from a pile of market data, the task of going through and churning the data to come up with meaningful information became a Herculean task. The idea of using computers to mine the data is not new. This has been for quite some time and with the database research and advancements in database management systems the processing of the raw data to a data-sets where we can mine became a needed infrastructure for any financial institution.

The need to **source, store and retrieve** the data for financial intuitions led to several Financial Technology (FinTech) Solutions. The evolution started with database systems and led to data warehousing techniques with several ground breaking research in processing the raw data, transformation and effectively aggregating with cubes and online analytical processing (OLAP) techniques. These techniques are very matured and is part of any data mining framework. We are in an era where just storing and retrieving of the data is no more cutting edge. To be an effective investment manager it is very important to access and analyze the financial and market data into **actionable information** in a timely manner. This requires that the organization should able to invest in building FinTech applications which can support the portfolio managers, traders and risk managers to get to the financial data, models and other need information made available. When it comes to models, now we are in need of machines to learn what we do and assist in the investment process. We wish it is easy to teach a machine which can access the data faster than a human to have the ability to understand and to see various patterns we find to mirror the same approach

and come with results which we do on a daily basis will be the cutting edge and competitive advantage for an organization.

In our paper we propose a Machine Learning Framework [47] which can be implemented as an enterprise framework and build FinTech applications which can help the financial managers to assist in their decision making process.

This paper is organized into seven sections including introduction. The Background section gives a brief information on the financial industry and machine learning techniques which are needed to utilize to harness the power of the super computers of modern era. In **Problem Statement section** we will highlight the current problem we are trying to solve and overcome. In the **current research and proposed solution section** we develop our proposed solution and discuss our enterprise architecture for machine learning and discuss the academic research done to identify the best of the class algorithms that fit our needs and the infrastructure architecture which can serve as base platform for future development needs. In **Section five, we implement the proposed solution** to see how the solution addresses our problem mentioned in section three. In the **comments and observation section** discusses about our experience and observations which can help next generation research. In the last section we identify the **future work** which can be taken by us or other researchers to advance the machine learning frameworks which can be made available for public consumption.

6.2 Background Information

Machine Learning (ML) [48] is a tool to help humans to make data exploration and decision making easier than manually going through millions and millions of data points to visualize a pattern. Past behaviour is an indicator for the future behaviour of an individual or an entity. Our entity here can be anything ranging from a human being or to an inanimate object like an elevator or a stock issued by a corporation or a debt issued by a company to float capital to fund their operations or expansion projects. For example if a company is not doing well in the past several quarters, that is an indicator for future deterioration of the

company and eventually going out of business. In traditional decision making model, the expert in the that domain will go through the historical company data and come up with a narrative to other to take decisions on the analysis put together. This is called data analysis or in new world called as Data Science (DS) as this depends on a set of mathematical and statistical rules which an individual apply to the data to come up with the narrative. Now a days each company is working on the data science project and infrastructure which we call as Predictive Analytics [20] and this field is growing and becoming an essential component in the financial analysis or investment process work flows. Another example is to predict the credit quality of an issuer over time and come up with a credit score or rating for that issuer. Industry recognized credit ratings agencies and individual companies take the ratings on an issuer seriously as that can impact the profits or returns they make when they invest into those companies where the ratings is a bit part of the financial strength of the company.

In the world where we are now with Big Data, going through any analytical predictions by a human manually will take several days or weeks. The time to come up with a narrative will be too late for investors as the opportunity will be lost and the information becomes stale or now available for other investors in the market. The efficient markets theory indicates the arbitrage of the returns will converge as the times pass. This is key in making the early investments with the ability to quickly analyse the data and come up with an investment strategy before other can come up with the same narrative. Deep learning [49]

6.3 Problem statement

Currently organizations are struggling to come up with an enterprise architecture for Machine Learning Framework which can be deployed quickly on cloud computing environment to give an edge for the business users in their daily investment decisions. The current problem is defined in two folds. First one, is have resource group in the organization who can understand the Cloud Architecture for Machine Learning (ML) for predictive analytics and the second aspect is to have the ability to scale the enterprise architectural solution to the business users without going through a massive project initiation. Summarizing the

problem into two major areas as listed below.

- Cost of the resources and hardware procurement
- Deploying the architecture to business users

If we can solve the above 2 problems we can give a competitive edge to our users. This paper focuses on these two aspects of the problem in defining a solution.

The focus of this paper is not to go into the theory and techniques of the machine learning but to layout a enterprise architecture framework which can be used by an organization when it comes to ML FinTech development.

6.4 Current Research and Proposed Solution

6.4.1 Current Research

There are several definitions of Machine Learning (ML) and one we will pick is by Tom Mitchell [50]. According to Tom "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experiences E." If we take an example of playing checkers where E is the experience of playing many games of checkers , T is the task of playing checkers and P is the probability [51] that the program will win the net game.

Any machine learning problem can be classified into two broad categories. The categories are called **Supervised learning** [52] and the other one is called **Unsupervised learning**. The focus of this paper is not to go into the theory and techniques of the machine learning [53] but to layout a enterprise architecture framework which can be used by an organization when it comes to ML FinTech development. There are several book on ML [49] which are very useful in understanding the basic concepts of ML.

Machine learning [54] is not new but it is quickly evolving and becoming as essential part of any decision making process. Artificial Intelligence [55] in machines is an extension of ML. Here is a quick workflow of a Azure Cloud [18] Computing Environment for a Machine

Learning (ML) work flow. There are several research papers and implementations using Azure ML components [56]. The general industry proven work flow starts with the data gathering. This is a very key in any implementation as the problem of identifying the data and setting up governance rules to clean and make it available for model development is very costly if not done properly.

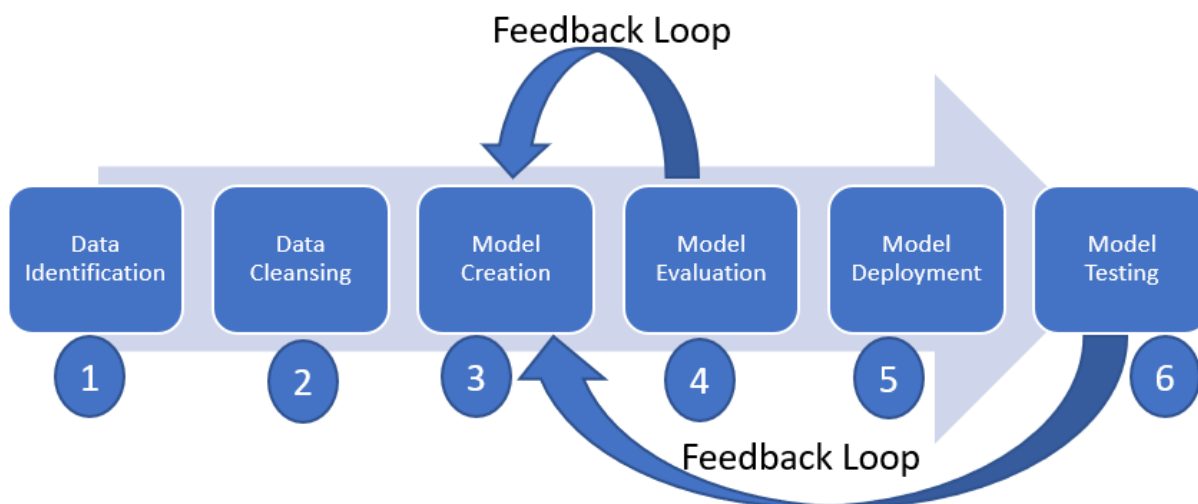


Figure (6.1) Organizational ML Workflow
Organizational workflow

- **Data and Cleansing** [9] Collect the needed data for the model development. After cleaning the data, Analyse and identify and separate test and training data. This is used for creating Azure Machine Learning predictive models.
- **Model Selection** Using machine learning algorithms to create new models that are capable of making predictions based on inferences about the data sets created and designated as training data set. We can create a new model or can use the existing models.
- **Model Evaluation** Examine the accuracy of new predictive models [57] [58] based on ability to predict the correct outcome, when both the input and output values are

known in advance. Accuracy is measured in terms of confidence factor approaching the whole number one.

- **Feedback Loop** and evaluate the model Compare, contrast, and combine alternate predictive models to find the right combination(s) that can consistently produce the most accurate results.
- **Deploy of the model** the model Expose the new predictive model as a scalable cloud web service, one that is easily accessible over the Internet by any web browser or mobile client.
- **Test** and use the model Implement the new predictive model in a test or production application scenario. Feedback is added for continuous model improvements and for accuracy improvement. Feedback is so important as the model is as good as the human writing them. Settling for a lesser quality of model will lead into several consequences. The other way to put the testing is **tuning** which is constant feedback from the users or the model creator to achieve high level of confidence from the model. In statistical terms it is the confidence level of with which the model can predict the positive result for which it is designed.

6.4.2 Proposed Solution

As mentioned in our problem statement, the paper is more focused on proposing a simple and easy solution to set up an enterprise ML framework on cloud computing environment. During our research and gap analysis phase, we found Microsoft Azure Cloud services offer better solutions which fit to our organizational goals than than vendors. The goals are set to maximize the resource optimization (hardware as well as software), growth and expandability with minimal impact to business operations in implementing the ML solution. Our proposal is not to focus on the actual ML model but focus and refine the existing ML Frameworks to give the business the transparency needed in-terms of hardware, software components used along with a strong governance. The biggest problem in the industry using ML as their

decision making tool is the ability to explain the results shown by the machine. Some times the results are not intuitive and it is hard to explain the outcome with the learning algorithms. Especially when we use the unsupervised or deep learning techniques. So to overcome we propose a clear structure of **audit proof model with strong governance structure**. The idea of audit proof model is to document each and every step of the proposed ML solution which can be a combination of ML algorithms (Classification, Clustering etc.) and rational why those models are used before we can implement them. This will be reviewed by an enterprise data governance team who are part of business who understands the business and acts as subject mater experts. The second proposal is a governance structure which will protect the business from any key person risk and keep the development, testing and production models isolated to prevent any unwanted and adverse ML model propagation.

In the first step we propose a simple and easy to use Microsoft Azure Cloud Machine Learning Framework demonstrating the implementation but not a full model development. The model development a problem solution is out of scope of this paper and is part of another paper which explores the full life cycle of a ML implementation from start to end with Data Science approach. Second step is to come up with a governance structure which can help in developing the proper guardrails around the ML models and framework which can help in explaining the results and stand internal and external audits.

6.5 Implementation

6.5.1 Azure Machine Learning Studio

In this section we will show the actual implementation of our framework with a simple example. The implementation is decided to evaluate the solution for our enterprise framework and as part of ***Proof of Concept (POC)***. For our implementation, we used our existing license provided by our organisation and created a **credit spend limit** ML [18] (MSML) [16] account. This account will limit our ability to run heavy CPU intensive executions as well as limited size of data store. **For our current experiment and POC**

this set up is more than sufficient. This solves the first stage problem of hardware and required software. This is done after going through other cloud providers compared to Microsoft. That analysis and results are out-of-scope for this paper.

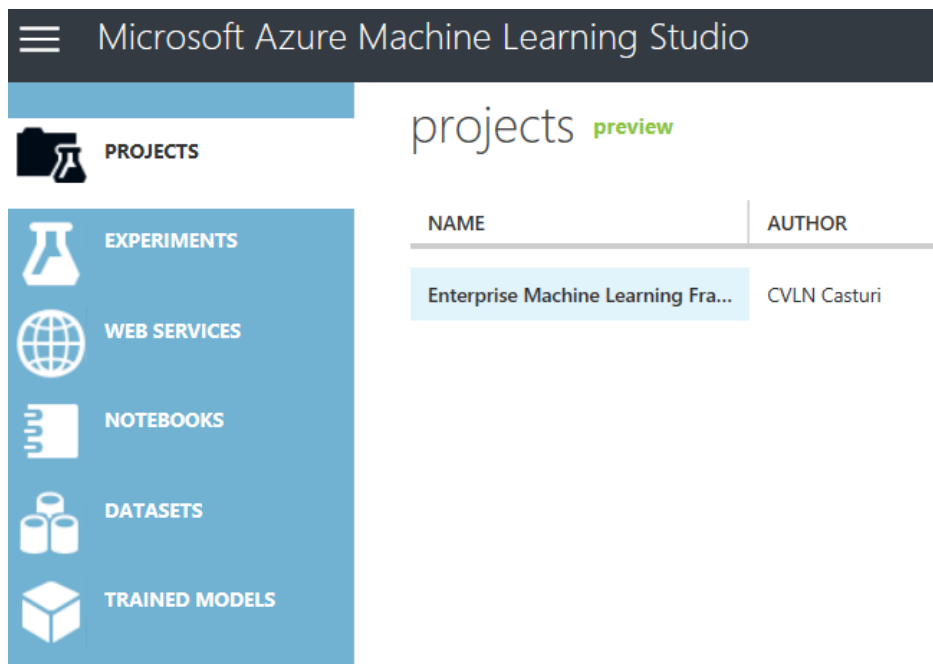


Figure (6.2) The Azure Machine Learning Studio Setup

For our implementation, we used our existing license provided by our organisation and created a credit spend limit ML account. This account will limit our ability to run heavy CPU intensive executions as well as limited size of data store. For our current experiment and POC this set up is more than sufficient.

Data Preparation: For testing our POC we used a sample data set which has 50,000 mortgage loans [59] which can be categorized as default and non-default loans. The set is divided into two subsets as 80% training data and 20% as test data. The data source is created as a excel file and uploaded to the Azure Cloud computing environment. This file will be used as source to build our model and to demonstrate the workings of the proposed ML framework on Azure Cloud Computing environment [60].

Azure ML Configuration setup is done with the use of Microsoft Azure ML Studio. The choice of MSML give several quick wins during the POC. As discussed in the problem

statement, the implementation is divided into two parts. The first part is to set up the hardware or cloud computing set up to be utilized by the ML models. The Azure Cloud set up depending on the organizational licensing terms. For our POC we obtained a license which has the ability to use Azure Cloud computing environment and can deploy the ML models provided by Microsoft ML framework. The ability to use our internally developed ML models can be deployed in the MSML environment. Usually the internal models are developed either in R or in Python [61] language by our data scientists and those models can be deployed on cloud. This is one of the criteria we wanted to explore as part of our POC. Ability to incorporate internally developed models which will be proprietary models to the organization. As shown in the figure 6.3 on page 97 gives the ability to start with a

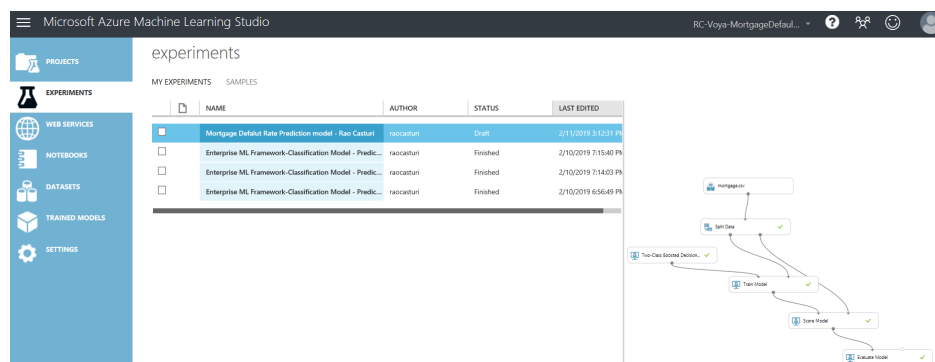


Figure (6.3) The Azure ML Studio POC Experiment Set up

new project. Under project we can develop several experiments. In our POC we developed 3 experiments under one project to find out the functionality offered by MSML Studio.

6.5.2 Azure ML Model Development

In this section we will develop the model for the experiment started in the subsection 6.5.1 and extend the project with adding data source and also the model.

The flow chart show in the 6.4 on page 98 is a fully developed Azure Categorization Model used as part of the ML studio. We will work through the various steps which we discussed in the research section 6.4.1. The data cleaning step is done outside the Azure environment and the data is uploaded as user data set. Once data set is uploaded to the

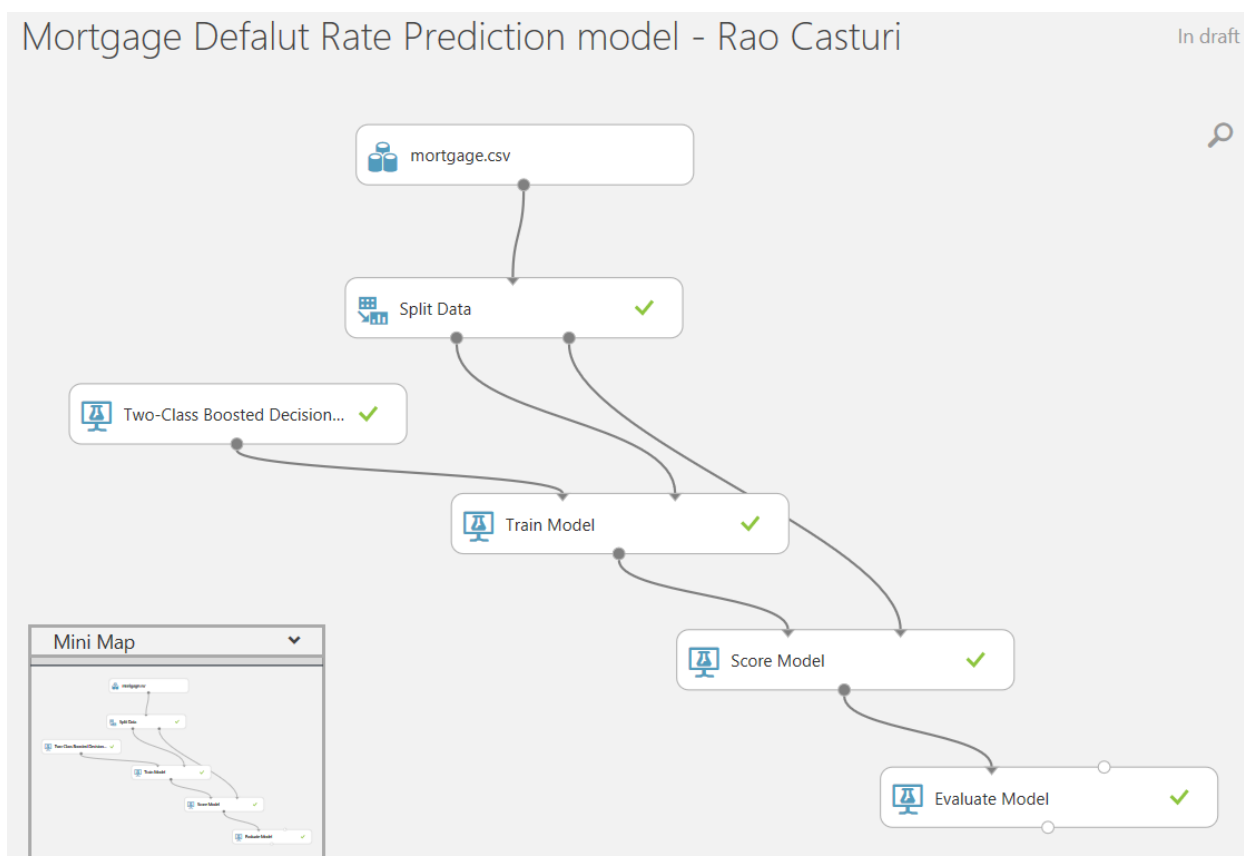


Figure (6.4) The Azure ML Studio Model Flowchart

Azure ML environment, the next step is to start an experiment. The data set can be any type of data. We tested with a flat file as well as our Azure SQL data base on cloud which we migrated earlier as part of our Azure SQL Data Migration POC [62] . To import data from SQL database, we need to specify the SQL Server name and write the Data SQL as port of extract. There are several advantages we see in using SQL data directly from Azure SQL database. We will discuss this in our comments section. Once created a project we can now create an experiment and add the model. Experiment creation is to add a blank experiment to the project by clicking the '+' symbol at the bottom of the screen. The details are listed more detailed in Microsoft documentation.

The Azure SQL data source is show in the figure 6.5 on page 99 which shows the connection string to the server and the SQL statement to extract the needed data. If the data stays constant during the experiment, we can use the cache check box to indicate the

data is static and use the cached data for experiment. The Azure ML canvas is shown as part of the flowchart which is developed during our experiment. The first step is to select the data source. we can either use the data file we upload or we can use the SQL data source, Once the data source is added next step is to add the data split module. The ML Studio provides several modules to work with ranging from data input to model development.

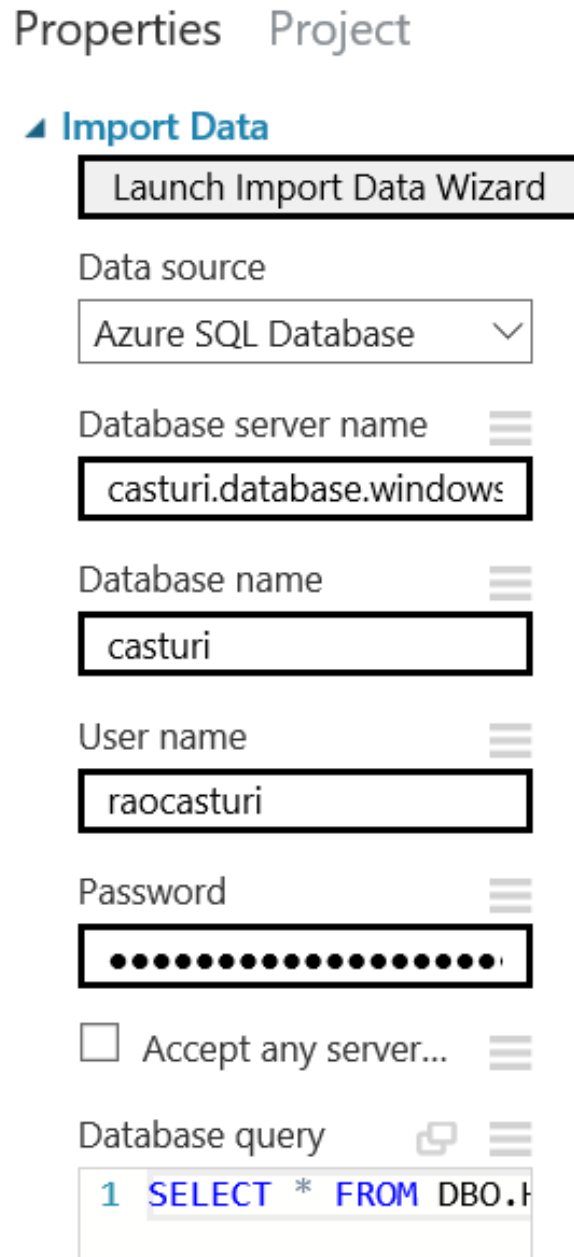


Figure (6.5) Azure SQL Database as Data Source

The data split module will help us in dividing the source data into training and test data for the model. The split module is added to the canvas and settings are set to have either 80% or 70% training and the 20% or 30% as test data. This depends on the users and the size of the data set. If we have large data set increasing the training data % will improve the model accuracy. The next step is to train the model with an algorithm. The selection of a ML algorithm depends on various factors and the expected output. Example if the outcome is a binary value like *Default or Non-Default* then using a classification algorithm [1] makes the prediction more accurate. This step involves a lot of data science and for the scope of this paper and for the POC we will use a simplest classification algorithm.

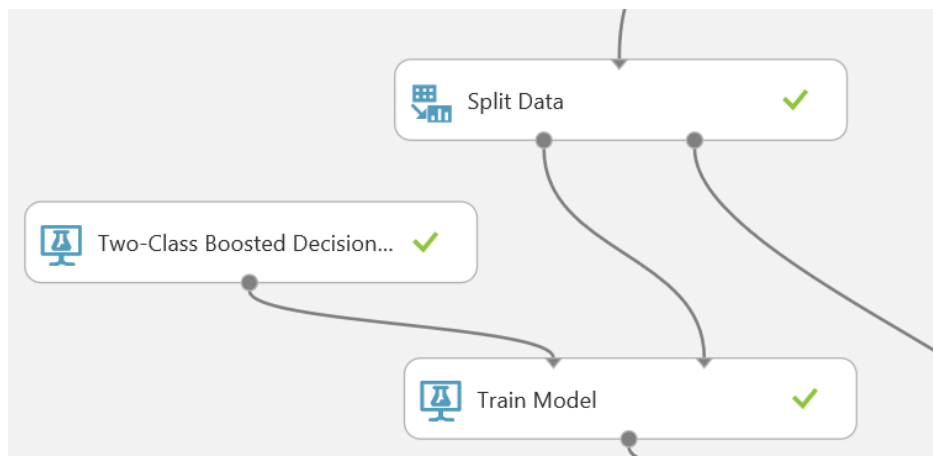


Figure (6.6) Azure ML Data Split and Algorithm-1

The flow for ML evolution is shown in the figure 6.6 on page 100 shows the selection of the algorithm and the way to implement and train the data. The Train the model module will take the training data set and uses the ML algorithm in our case the Two-Class Boosted Decision Tree.

When executed, the model will go through the data set and will learn to come with the outcome which in this case will be a binary outcome. During the training process we will select the column(s) which we would like to see the statistical results to build a fine tuned ML model. The training model gives us the ability to pick desired field for the ML analysis. The next step in building a ML model is to score the model. This is shown in

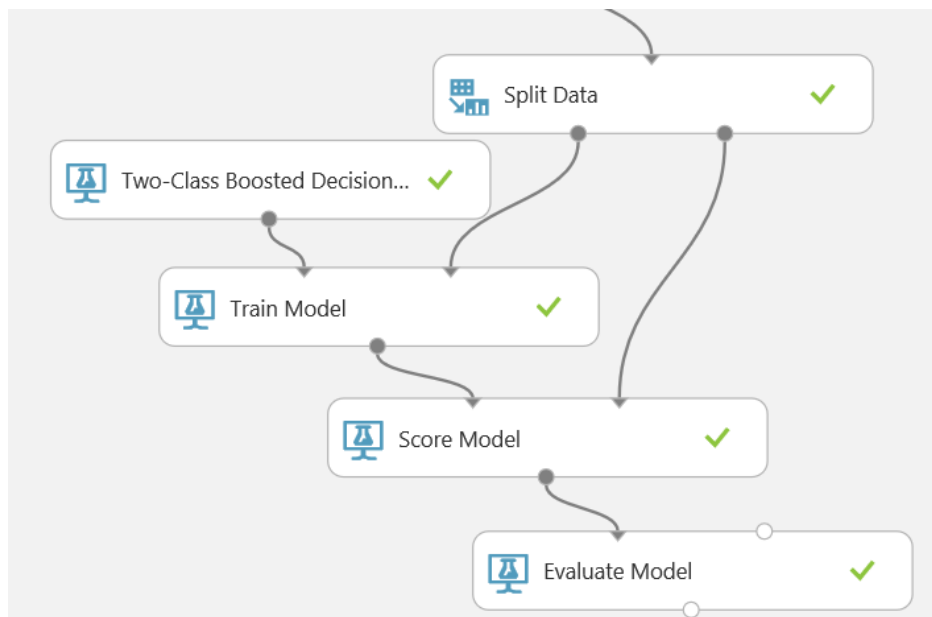


Figure (6.7) Azure ML Data Split and Algorithm-2

the figure 6.7 on the page. The Score module will take the training data as well as the test data set kept aside (20%) in the data split and scores the output. In the figure 6.8 on page 102 shows the LTV_time as the field we are using to analyze and train the data using the classification algorithm. At every stage we can actually see the data output with various statistics visually. When we highlight the module and right click we have the options to select from to show the data statistics. Now as we set the score module and satisfied with the output results the final step is to run through the evaluator module. The evaluator is the last step in building the model and getting ready to run through the actual real data for predictive analysis. Evaluation of the model is very crucial as this will evaluate the model build Evaluates a scored classification or regression model [63] with standard metrics. This will generate the results which can be viewed with the standard graphs to see how accurate our model compared to the standard mathematical metrics. The example is shown in the figure 6.9 on page 102. At every stage we can run and observe the output results. This completes the build and evaluation of our model using Azure ML framework using the ML Studio. In the next section we will go over some of our observations and comment on the process and results.

Train Model

Label column

Selected columns:

Column names: LTV_time

Launch column selector

Figure (6.8) Azure ML Field Selection

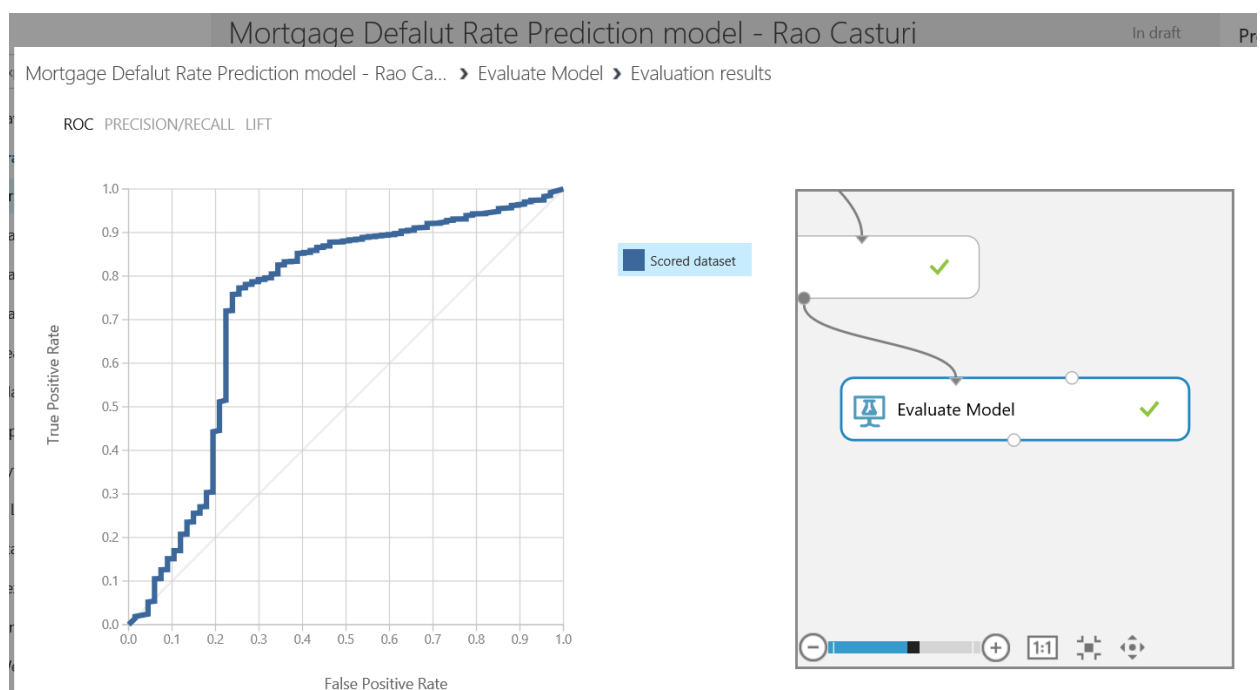


Figure (6.9) Azure ML Area Under the Curve AUC

6.6 Comments and Observations

The most important thing in Machine Learning is to evaluate the performance of the model and tune it till the model converges to the real world observations. During the process of building a real model close to real-world, there are few metrics we can observe. One of them is AUC-ROC curve (Area Under the Receiver Operating Characteristics). For any classification models performance this AUC-ROC or sometimes called AUCROC [59] curve is one of the most important evaluation metrics. The idea is to observe the AUCROC curve and determine the performance of the ML model. In our generic observation the threshold is set to .5 and the curve is observed which as an example is shown in the figure 6.9 on page 102. ROC is a probability curve and AUC represents degree or measure of separability. The observation of LTV_time which is Loan to Value at the time of observation with the threshold 0.5 is compared to the model. In our observation the model performed relatively good showing an value of AUC 0.736 which is shown in the figure in 6.10 on page. AUC is 0.7, it means there is 70% chance that model will be able to distinguish between positive class and negative class. 103.

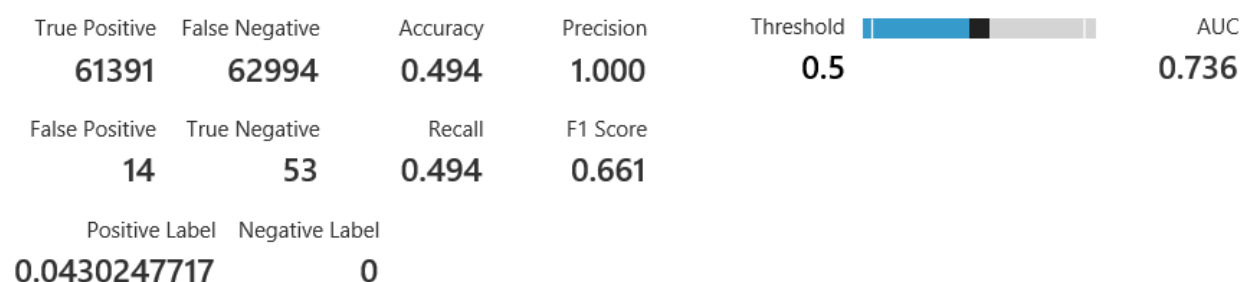


Figure (6.10) Azure ML AUC Observed

The general rule is the higher AUC [59] the higher accuracy of the model prediction power. AUC-ROC measure is important because it will give a sense of the True Predicted Positives versus Negative Positives. We want to minimize the false or negative positives and increase the true positives. In our observation and our conclusion is the model can be

improved and if needed data cleaning can be performed and rerun the learning algorithm [64]. For the scope of our paper, the idea is to implement a standard enterprise architecture of ML using Microsoft Azure and utilizing the ML Studio. Our implementation of ML framework on Microsoft Azure using the Microsoft ML Studio clearly gives a solution for our problem of not having an enterprise ML framework utilizing the Azure Cloud Computing environment. Our implementation can be a solution path for up coming several ML projects in an investment organization.

Even though there are several minor issues, the knowledge we now build in our organization can be levered across multiple groups who are waiting for road map or the Enterprise ML Framework. The implementation overall is very smooth except few hurdle we would like to look into further before we can roll out an Enterprise ML Framework. Data Sharing is one concern we would like to address before rolling out to wider audience.

6.7 Future Work

Even though we were able to solve and propose a simple and elegant solution for our Enterprise ML Framework, there are several things we would like to address as part of our continuous development efforts. The first thing we need to address on a bigger scale the access to the Microsoft ML studio to users. **Cost of having access MS ML should be evaluated before we can make it a standard platform for our ML and predictive analytics.** On the other hand developing an in-house expertise in MS ML is another path we need to address. This has cost associated and can have several strategies to solve this. One strategy is to have a **centralized ML team** which can serve overall organization and provide solutions for any ML projects. The other strategy is to build the subject matter experts (SME) with in the department which would like to incorporate the ML models into their business line. The first method (Centralized) will need initial staff to hire or transfer from different departments and that can have a serious impact on business and day to day operations. On the other hand the centralized ML Team can bring quick value add to the ML projects in the organization as the team is equipped with knowledge and tools needed

to implement a ML solution. The other advantage for the centralized team will bring a standard development methodology and documented processes. As part of future growth, predictive Analytics can be Incorporated into the business decisions and create an enterprise application framework or Predictive Analytics [65] and data science group.

In a **decentralized ML framework** where resources are located with in the business unit or department where the ML and predictive analytics are implemented will need to get up to speed on the infrastructure. This can impact the productivity of the analytical teams which can in turn can show a drag overall performance of the team or department. The advantage for a decentralized ML team is the domain knowledge. As they are masters of their domain and data, it will be easy for the users to validate the results from models and to fine tune the model to get an acceptable AUCROC. This with the centralized ML team can take longer as they may not be aware of the data and are not closer to the expected results. Deciding the ML team structure is a key strategic initiative for a successful implementation of the ML Framework. This is not part our POC but will be a future project for any organization to see what will work well for them and which model can server best for teams with minimal down time for business.

The other area we would like to invest more time is in ***data sharing*** and data access outside the groups who develop the ML models and generate the output and who can view or access the raw data used or the model evaluated data for any kind of FinTech applications this is very important to decide before implementing any enterprise level framework. Our paper didn't address or took this as part of our POC. This area of investigation and research will blend into data security on Azure Cloud and also data governance frameworks. This is another area we would like to extend our research.

The another future research area we didn't include in our paper is the ***Model Governance***. A good model governance [66] can save a lot of operational reputation risk. As we see the ML models are always trying to increase the accuracy and during that process Data Scientist can assume parameter values. As the model evolves and gets to a point where it can be put to real world application test,that model should be locked down with some

change control governance around it. If the model is in perennial development and this is used for any business decisions, that can breach internal auditing rules. Any financial models typically will have some kind of change control governance. This is an area we can extend the study and bring that as a part of Enterprise roll out of ML framework. Some times the business models can be done outside Microsoft ML framework like Python, R or another statistical language. Not able to direct or help the business in setting up proper guardrails of governance will impact the business. In our POC we didn't use any external scripts like Python or R to implement the ML Model. Implementing third party scripts can be another area to research and come up with best practices for scripts which are not part of the core Microsoft Machine Learning Studio.

Some of the future proposed items are already addressed by creating different environments to promote the MSML framework. Overall what we proposed in this paper of an Enterprise ML Framework will fill well with any model and data governance framework. The Proposed architecture is independent of the governance but will strengthen the enterprise ML architecture framework with a working implementation. Organizations can put in more guardrails on their governance structure depending on their needs. Some can have a open architecture to promote code in a agile development cycle and others can have a more waterfall framework to promote their Machine Learning models to facilitate growth of their organization.

Overall what we proposed in this paper of an Enterprise ML Framework will fit well with any model and data governance framework. The Proposed architecture is independent of the governance but will strengthen the enterprise ML architecture framework with a working implementation. The extension for this research can go into Artificial Intelligence [67] which can help further machine learning techniques.

CHAPTER 7

FUTURE SCOPE

7.1 Introduction

So far we demonstrated several steps in accomplishing a streamlined Enterprise solution and a framework which can be implemented in a small to mid size financial organization building their capability in Data Mining and Machine Learning for decision making. In the current proposal we demonstrated the evolution of data aggregation to machine learning with a hybrid reporting framework on cloud computing environment. Even though there is a study progression through different stages, each stage leaves a rich scope for expansion and future scope at each step. In this chapter, we will consolidate all the future development needs and bring together for future scope of this thesis to take it to next level.

7.2 Aggregation

7.2.1 Front End

During the stage of Aggregation we mentioned there are a few things we could have incorporated in bringing the Data Aggregation to next level. Due to time constraints, the scope of the research was scoped to build a user front end at a later phase. The selective cuboid technique currently depends on the rule based architecture which is controlled by the user with entries into SQL Tables. The front end for this rule master is currently outside the scope of a front end. We propose an easy to use with various on screen selection front end to facilitate the setting up of the ability for users to select what level of aggregation they would like to set up in utilizing our proposed architecture.

7.2.2 Data Retrieval and Reporting

During our initial discussion design we proposed a reporting framework to be used by end user for an easy retrieval of the data once the aggregation rule is set up and run. The rule is run on a daily basis till it is turned off or made inactive. Using a Business Intelligence tool, we can enhance the reporting on the aggregated data in a format where users can easily see the trends and mine further when they see any observable trends in their data or report. This enhancement can be made with simple reporting tool discussed in part 5 of this proposal.

7.3 Cloud Computing

7.3.1 Data Security on Cloud Environment

During the initial proposal, we didn't scope the security of data and access [68] when the migration from on-premises [62] to cloud computing environment. This is not due to the ability but due to the time constraints and also as the scope of the project dictated to show the move as a proof of concept. During the part 2 we were able to demonstrate the migration from on-premises to cloud was achieve with simple script based architecture which bi-pass the network authentication and and kept the security and authentication as an enhancement for next phase or as a sub research project.

The other area of interest is to run a distributed processing using multiple nodes on Spark [56]. Apache - Spark fits into the Big-Data Eco-system for parallel data processing. We in our research used Microsoft Azure SQL Cloud but there is a huge opportunity for future researchers to work on the Spark-SQL and implement the parallel design discussed in the research. This can provide an alternative implementing of Azure Cloud Environment using the open source Apache - Spark.

7.4 Data Mining

7.4.1 Techniques

In the current research few techniques are used to mine the data. The usage of these algorithms are not accessible to the end users. This is done to prevent scope creep of the implementation and to avoid unwanted delays and business interruptions. Exposing the methodology of the algorithm or the decision path will help the users to trust program decision rather than using as black box systems. The other area of research the future researchers can take is to implement or use what provides Microsoft Azure Cloud Platform also provides Massive Parallel Processing (MPP) processing of files and computational node architecture. This is another area we can invest and use the Azure Data Warehouse architecture to see if we can leverage Microsoft cloud technologies to further build our Distributed Financial Cloud Computing Environment.

7.5 Reporting and Machine Learning Framework

7.5.1 Data Access Controls

In this research we did not work on data controls for a reporting framework. This itself can be an area which is very important in-terms of Data Governance. This area is widely growing and industry is always looking for a better model to manage their data access.

7.5.2 Guidelines

Good governance on machine learning algorithms is a key when it comes to financial institutions. There is a need to disclose or able to audit the algorithms. In our current research we didn't touch or investigate or set up an governance structure to implement the ML framework. This can be a research area to come up with industry best practices and publish some guidelines to user community.

CHAPTER 8

CONCLUSION

This research is mainly focused on consolidation and improving the existing Data mining and Machine Learning frameworks in terms of providing a road map for an organization for their data driven analysis. Even though this research didn't touch directly at data sciences but indirectly builds a path way for a smooth transition from manual analysis to an automated and robust framework for preparing data along with exposing the processed results in terms of hybrid reporting framework. The initial part of the research is focused on building efficient data aggregations techniques and data migration to cloud computing environment and the later part of the research is focused on the data mining and machine learning techniques. The Research leaves room for various avenues of expansion to strengthen the proposed frameworks.

The proposed implementations of sub problems are built as modules and can be used as needed. This gives the flexibility to choose what an end user needs rather than the whole package if there is no need to implement the entire end to end solution. This is an added advantage of the approach and this is well thought out and discussed at various levels and fits into object oriented framework design. This is an extension of the object oriented software design. In general we use several design patterns to build software platforms and our build of using rules based approach for aggregation and data mining follows the similar path of object oriented design. In the problem definition section we defined the problem as *"Purpose of our research is to design and develop cost effective and an efficient enterprise framework on cloud computing environment for data mining and machine learning using a rule based aggregation engine along with parallel processing of user defined financial calculations to enable predictive analysis and knowledge discovery with a business intelligence reporting capability."* After

the completion of all the sub problems, we can say the were able to solve the problem by creating a rule based methodology to aggregate and also to execute parallel runs on cloud computing environment with less to non down time the business.

During the research phase, initially the idea is to build a Data mining and Machine Learning frameworks with limited capability of reporting. Once we started the research on data mining we found a new problem which is data aggregation. Every data Mining framework depends on a quality data. If the data is in its raw form that is not normalized in terms of data base structure, it makes it hard to predict or to observe any actionable patterns. So in order to solve the initial problem, we designed an efficient aggregation technique with **selective cuboids**. With this technique we were able to speed up and give users ability to set up rules to aggregate raw data or transaction level data to a meaningful information on which they can see trends and act upon. This has proved very effective and the gave flexibility for any sort of aggregation with minimal or no code changes. This architecture is portable to any transnational data with relational database characteristics. This is the foundation to the next layer which is data mining [1] with pattern recognition. While working on the next step, we encountered few hurdle as data storage and expandability of our architecture. Our Aggregation Engine (AE) which is developed on a SQL Engine can be ported to cloud computing to solve the data storage problem. The selective cuboid method avoids several levels of aggregation and will only calculated the specific level of aggregation requested by user. This saves enormous computational requirements and storage requirements. With only specific level of aggregation needed, we only need to save one cuboid compared to several higher aggregation cuboids.

To solve the next problem of storage and computing power, we researched into cloud computing technologies. During this phase, to reduce migration time and cost from on-premises databases to cloud computing databases we used Microsoft Azure as our proof of concept migration architecture. There are several ways to migrate an on-premises SQL Databases to Microsoft Azure SQL Databases. We used a simpler technique to migrate using **“Scripted Based Migration”** then a total back up migration. This saved us several

days and heavy technical team involvement as this technique can be used by anyone with script knowledge on SQL databases. The data population is simple as we only target the required rules objects (tables and procedures). The raw data process won't be changed except the destination which will be the Azure SQL database rather than the on-premises. The Aggregation Engine execution was tested on the Azure SQL Database to see if there are any issues. The run was successful and there is no code change required from migrating the Aggregation Engine from on-premises to cloud computing environment.

Once we moved our SQL Database from on-premises environment, we focused on the data mining on what we aggregated (mortgage data). Before that we worked on speeding up the data load of our source data and also running the Aggregation Engine in a **parallel computing architecture**. For this step of we designed a rule based architecture which is an extension of the initial Aggregation Engine. We researched several parallel computing methodologies and came up with a SQL based approach which saved a lot of hardware costs and time of implementation. The Aggregation Engine was extended with couple of parameters to make it parallel execution procedure which can easily controlled and managed through a SQL table architecture. During this phase we also touched on Data Mining concepts which gives us the trends in the time series data we used for testing our concepts.

We also researched on reporting frameworks which can bring value to the business. Reporting framework we proposed is a **hybrid reporting framework** and that works well in any organization. There are several requirements which we took into consideration before proposing an consolidated reporting framework. Our approach satisfies all the set criteria and easy to implement.

The last problem we want to solve is to propose a working framework for Machine Learning (ML). For this we researched various ML architectures which are available and came up with a simple and easy to use architecture on Microsoft Azure. Microsoft offers various ML models and it can help reduce the development time for an organization. Using the ML framework we laid-out a road map to incorporate various Data Science tools which come with Microsoft Data Science tool kit.

With solving various sub problems we were able to accomplish and propose and implement an enterprise framework for Data Mining and Machine Learning for an investment platform. Our proposed research can be extended and used for any organization where there is a need for an end to end solution of data aggregation to artificial intelligence.

There is ample opportunity to extend the current research and implementation to make it as a FinTech Software As Service (FinTechSaS).

REFERENCES

- [1] M. K. Jiawei Han, *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, 2006.
- [2] M. A. H. C. J. P. Ian H. Witten, Eibe Frank, *Data Mining, Practical Machine Learning Tools and Techniques*, fourth edition ed. Springer, 1998.
- [3] J. Pei and J. Han, “Can we push more constraints into frequent pattern mining?” *KDD*, 2000.
- [4] F. Fabozzi, *The Hand book of Fixed Income Securities*, ser. Seventh Edition.
- [5] Weber, Nina., “Big data: Can we prevent the next financial crisis?” *Economics Finance Society (EFS)*. Kings College London.
- [6] F. Fabozzi, *Fixed Income Analysis, CFA Institute Investment Series*, ser. CFA Institute Investment Series. John Wiley & Sons, 2001.
- [7] J. P. J. Han, M. Kamber, *Data Mining Concepts and Techniques*, ser. Third Edition, 2012.
- [8] S. James, *An Introduction to Data Analysis using Aggregation Functions in R*. Springer, 2016.
- [9] D. Loshin, *The Practitioner’s Guide to Data Quality Improvement*. Morgan Kaufmann, 2011.
- [10] Microsoft, *Microsoft SQL Server 2012: Analysis Services Multidimensional Modeling*. Microsoft e-Publication, 2012.
- [11] P. G. A. N. J. R. R. S. S. Agarwal S. Agrawal, R. Deshpande, “On the computation of multidimensional aggregates,” *International Conference on Very Large Databases*, pp. 506–521, 1996.

- [12] Ahmed, Mohiuddin., Mahmood, Abdun Naser., Islam, Md. Rafiqul, “A survey of anomaly detection technique in financial domain,” *ACM-Future Generation Computer Systems*, 2016.
- [13] M. M. D. X. S. K. Tok Wee-Hyong, Parida Rakesh, *Microsoft SQL Server 2012 Integration Services*. Microsoft e-Publication, 2012.
- [14] A. S. Arik Friedman, Ran Wolff, “Building cubes with mapreduce,” *DOLAP’11, Glasgow Scotland UK.*, pp. 17–24, October 28, 2011.
- [15] D. D. D. T. L. G. Eaton, Chris. and P. Zikopoulos, *Understanding Big Data Analytics for Enterprise Class Hadoop and Streaming Data*, 2012.
- [16] D. C. . Associates, *Introduction to Azure Machine Learning A Guide for Technical Professionals*. Chappell Associate, 2015.
- [17] E. D. B. Leonard G. Lobel, *Step by step Microsoft Azure SQL Databases*. Washington: Microsoft Press, 2014.
- [18] J. Barnes, *Azure Machine Learning*. Microsoft Press, 2015.
- [19] B. Burns, *Designing Distributed Systems Patterns and Paradigms for Scalable, Reliable Services*. O’Reilly, 2018.
- [20] W. H. T. a. Roger Barga, Valentine Fontama, *Predictive Analytics with Microsoft Azure Machine Learning*. Apress, 2015.
- [21] J. Savill, *Mastering Microsoft Azure Infrastructure Services*. Indianapolis: John Wiley Sons, Inc, 2015.
- [22] N. Gemayel, “Analyzing google file system and hadoop distributed file system,” *Research Journal Of Information Technology*, September 15 2016.
- [23] M. A. Documentation, *Overview of Azure Data Lake Store*. Microsoft e-Publication, 2012.

- [24] T. Y. L. Wesley W. Chu, *Foundations and Advances in Data Mining*. Springer-Verlag, 2005.
- [25] C. B. Thomas Connolly, *Database Systems, A Practical Approach to Design, Implementation, and Management*. PEARSON, 2015.
- [26] J. Krishnaswamy, *Microsoft SQL Azure Enterprise Application Development*. Spinger, 2011.
- [27] Esling, P., Agon, Carlos., “Time series data mining,” *ACM Computing Surveys*, 2012.
- [28] F. Fabozzi, *Fixed Income Analysis*. New York: John Wiley and Sons, 2007.
- [29] E. M. Jyh-Shing Roger Jang, Chuen-Tsai Sun, *Neuro-Fuzzy and Soft Computational Approach to Learning and Machine Intelligence*. Prentice Hall, 1997.
- [30] B. Powell, *Microsoft Power BI Cookbook*. Mumbai: Packt Publishing, 2007.
- [31] R. C. Rajshekhar Sunderraman, “Script based migration toolkit for cloud computing architecture in building scalable investment platforms,” in *Database and Expert Systems Applications - DEXA 2018 Workshops*. DEXA 2018, September 2018, pp. 46–64.
- [32] M. R. Alberto Ferrari, *Introduction to Power BI*. Washington: Microsoft Press, 2016.
- [33] Microsoft, *Microsoft Azure Documentation*. USA: Microsoft Press, 2015.
- [34] A. Zimmermann, “The data problem in data mining,” *SIGKDD Explorations*, vol. 16, Issue 2, pp. 38–45, 2014.
- [35] S. Few, *Information Dashboard Design*. USA: O’Reilly., 2006.
- [36] C. H. R. S. C. I. A. W. Analysts James Richardson, Joao Tapadinhas, *Magic Quadrant for Analytics and Business Intelligence Platforms*. USA: Gartner Inc., 2018.
- [37] A. B. Judith s. Hurwitz, Harcia Kaufman, *Cognitive Computing and Big Data Analytics*. Indianapolis: John Wiley Sons, Inc, 2015.

- [38] C. H. R. S. C. I. A. W. Analysts James Richardson, Joao Tapadinhas, *Magic Quadrant for Analytics and Business Intelligence Platforms*. USA: Gartner Inc., 2018.
- [39] R. M. B. Paul Turley, *Microsoft SQL Server Reporting Services Recipes*. USA: Wiley-Publishing, Inc., 2010.
- [40] I. Builders, *WebFOCUS 8.1 Technical Documentation*. USA: Information Builders, 2018.
- [41] S. Few, *Information Dashboard Design*. USA: O'Reilly., 2006.
- [42] B. Powell, *Microsoft Power BI Cookbook*. Mumbai: Packt Publishing, 2007.
- [43] M. R. Alberto Ferrari, *Introduction to Power BI*. Washington: Microsoft Press, 2016.
- [44] J. Savill, *Mastering Microsoft Azure Infrastructure Services*. Indianapolis: John Wiley Sons, Inc, 2015.
- [45] I. Builders, *WebFOCUS 8.1 Technical Documentation*. USA: Information Builders, 2018.
- [46] A. B. Judith s. Hurwitz, Harcia Kaufman, *Cognitive Computing and Big Data Analytics*. Indianapolis: John Wiley Sons, Inc, 2015.
- [47] S. B.-D. Shai Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.
- [48] P. N. Stuart J. Russell, *Artificial A Modern Approach*. PEARSON, 2016.
- [49] S. Skansi, *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*, ser. Undergraduate Topics in Computer Science. Springer International Publishing, 2018.
- [50] T. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

- [51] M. A. Maloof, *Machine Learning and Data Mining for Computer Security: Methods and Applications*. Springer, 2009.
- [52] S. J. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*. Pearson, 2016.
- [53] U. J, *Python for Probability, Statistics, and Machine Learning*. Springer, 2018.
- [54] V. K. C. D. S. e. Yves Kodratoff (auth.), Georgios Paliouras, *Machine Learning and Its Applications: Advanced Lectures*, 1st ed. Springer-Verlag Berlin Heidelberg, 2001.
- [55] M. Azure, *AI In Action Technical case studies*. Microsoft, 2015.
- [56] R. W. Holden Karau, *High Performance Sparc*. O'Reilly, 2017.
- [57] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Springer, 2006.
- [58] A. M. Elahe zibanezhad, Daryush Foroghi, "Applying decision tree to predict bankruptcy," *IEEE*, pp. 165–169, 2011.
- [59] H. S. Brat Baesens, Daniel Rosch, *Credit Risk Analytics Measurement Techniques, Applications, and Examples in SAS*. Wiley, 2016.
- [60] J. Savill, *Mastering Microsoft Azure Infrastructure Services*. Sybex, 2015.
- [61] S. F. Elston, *Data Science in the cloud with Microsoft Azure Machine Learning and Python*. O'Reilly, 2016.
- [62] R. C. Rajshekhar Sunderraman, *Script Based Migration Toolkit for Cloud Computing Architecture in Building Scalable Investment Platforms*. DEXA Workshop 2018: 46-64, 2018.
- [63] D. A. D. John O. Rawling, Sastry G. Pantulu, *Applied Regression Analysis: A Research Tool*. Springer, 1998.
- [64] A. E. S.P. Rahayu, S.W. Purnami, "Applying kernel logistic regression in data mining to classify credit risk," *IEEE*, 2008.

- [65] P. C. Samet Ayhan, Johnathan Pesce, “Predictive analytics with surveillance big data,” *ACM SIGSPATIAL BIGSPATIAL*, pp. 81–90, 2012.
- [66] W. C. Elm, “User evaluation methodology framework in big data environments,” *1st. Workshop on Human-Centered Big Data Research*, 2014.
- [67] C. Z. Ming Xue, “A study and application on machine learning of artificial intelligence,” *2009 international Joint Conference on Artificial Intelligence*, 2009.
- [68] O. R. Alberto Abell, Jaume Ferrarons, “Providing k-anonymity in data mining,” *VLDB*, pp. 789–804, 2007.