Simplified menu-driven data analysis tool with macro-like automation

Submitted in fulfilment
of the requirements for the degree of

Master of Science

of Rhodes University

Luntha Kazembe

Grahamstown, South Africa
April 2022

Abstract

This study seeks to improve the data analysis process for individuals and small businesses with limited resources by developing a simplified data analysis software tool that allows users to carry out data analysis effectively and efficiently. Design considerations were identified to address limitations common in such environments, these included making the tool easy-to-use, requiring only a basic understanding of the data analysis process, designing the tool in manner that minimises computing resource requirements and user interaction and implementing it using Python which is open-source, effective and efficient in processing data.

We develop a prototype simplified data analysis tool as a proof-of-concept. The tool has two components, namely, core elements which provide functionality for the data analysis process including data collection, transformations, analysis and visualizations, and automation and performance enhancements to improve the data analysis process. The automation enhancements consist of the record and playback macro feature while the performance enhancements include multiprocessing and multi-threading abilities. The data analysis software was developed to analyse various alpha-numeric data formats by using a variety of statistical and mathematical techniques.

The record and playback macro feature enhances the data analysis process by saving users time and computing resources when analysing large volumes of data or carrying out repetitive data analysis tasks. The feature has two components namely, the record component that is used to record data analysis steps and the playback component used to execute recorded steps. The simplified data analysis tool has parallelization designed and implemented which allows users to carry out two or more analysis tasks at a time, this improves productivity as users can do other tasks while the tool is processing data using recorded steps in the background.

The tool was created and subsequently tested using common analysis scenarios applied to network data, log data and stock data. Results show that decision-making requirements such as accurate information, can be satisfied using this analysis tool. Based on the functionality implemented, similar analysis functionality to that provided by Microsoft Excel is available, but in a simplified manner. Moreover, a more sophisticated macro functionality is provided for the execution of repetitive tasks using the recording feature. Overall, the study found that the simplified data analysis tool is functional, usable, scalable, efficient and can carry out multiple analysis tasks simultaneously.

Acknowledgements

To God be the Glory, I would not have been able to complete this work without him. I would like to thank my dear wife, Lucy, from the begging she has supported my work. My daughters, Sekai and Addiah whom in there own ways motivated me.

This work is a product of the tireless and invaluable support and supervision of my supervisor, Prof. Karen Bradshaw. Every week making sure we were making progress until the completion of this work. Special thanks to my brothers and sisters and friends far and near from the support through out this journey. Thankful to Stones Dalitso Chindipha, John Gillam, Sara Williams and others too numerous to mention for all you have done.

I would like to thank the Computer Science Department at Rhodes University for providing a great environment and support for me to be able to carry out my work. Finally, I am grateful to Beit Trust, Rhodes University and Malawi Revenue Authority who coordinated for the financial support for this research.

Contents

List of Figures

List of Tables

Listings

List of Algorithms

Chapter 1

Introduction

1.1 Context of the Research

Information is valuable: Knowledge of different aspects of life in an environment is important for society. This is the same for different organizations that put in effort and resources to achieve a set of goals. An organization gathers resources and then uses the resources to create value. For example, an organization buys a product, hires people to sell the product and obtains money from the sales of the product to grow and survive. The organization needs to know about its profitability so that it can identify how to improve and grow the business. To obtain this information, the organization needs to look at the facts, i.e. the data generated from its operations and analyse the data to obtain information that can inform the organization on the state of its affairs."The increasing data analysis requirements of almost all application domains have created a crucial need for designing and building a new generation of data science tools that can efficiently and effectively analyse massive amounts of data in order to elicit worthy information, detect interesting insights and discover meaningful patterns and knowledge" (Elshawi *et al.*, 2018, p.4). Thus, organizations need to be able to transform data into evidence-based decisions and informed actions (Wang *et al.*, 2018b).

The Oxford Advanced Learner's Dictionary[1] defines data as known facts or things used as the basis for inference. Data is the plural of datum and data can be found in all aspects of the world around us; for example, when buying goods and services, recording weather records for a day, voting etc. and therefore, data can be seen as the essential raw material of any kind of human activity (Cuesta, 2013). Data are raw resources in the process of acquiring information and knowledge for making informed decisions. Data

[1]https://www.oxfordlearnersdictionaries.com/definition/english/data?q=data

need to be processed appropriately to produce valuable output, i.e. information. Value is derived from data through a process called data analysis. During data analysis the data are organized for later use in methods that help to explain the past and predict the future (Cuesta, 2013). A typical data analysis process has four sub-processes, namely, data collection, data preparation, data analysis and data visualization. Data analysis is used to understand past events and also predict future events by making inferences from a specific context to a more general one (Donoho, 2017). According to Zohuri and Moghaddam (2017), analysis of data is a set of processes that include inspecting, cleaning, transforming, and modelling data with the goal of discovering useful information, suggesting conclusions, and supporting decision-making. Data analysis is used in most, if not all disciplines as data are a critical raw resource. With such a broad scope, there is a wide range of data analysis techniques and approaches available for addressing various data characteristics.

The challenge of improving and growing constantly faced by organizations is a very real one, due to a number of different issues, one of which is failing to access information from data generated by the organization's business processes. Not all organizations have the capacity, both in resources and skills, to carry out data analysis. Another common problem that then arises is the choice of which tool to use based on an organization's information needs and capabilities. Microsoft Excel is a common and widely used tool for basic to advanced data analysis. It offers a range of functionality that is adequate in a general sense, however, as with most software, it has some limitations such as training required and resource limitations amongst others.

1.2 Research Statement

This research aims to design and validate a simplified generic data analysis tool with macro enhancements for use in a resource limited environment. The research consists of two parts: the first is the design and development of a simplified generic data analysis tool (in short, SDA tool) with common analytical measures and techniques in order to meet various requirements of data analysis across a range of industries. The second part involves the design and development of a record and playback macro feature to be integrated into the SDA tool to make the data analysis process more efficient and effective through both macro enhancements and multi-tasking for a general data analysis approach suitable for individuals and small businesses.

The macro enhancements are designed to eliminate some time consuming tasks in the data analysis process primarily by recording analysis steps on a subset of a dataset (referred to

as modelling) and thereafter executing the recorded steps using a complete dataset (that is, parametrizing a created model). Through multiprocessing, separate analysis tasks can be run concurrently, thus allowing users to minimize the time they spend working on data. Further enhancements are available through multi-threading on input/output (IO) heavy tasks in a data analysis process, while the menu-driven approach allows for users with no technical skills and only basic analysis knowledge to run an analysis task. Moreover, the open-source nature thereof makes the SDA tool suitable for small organizations and individuals.

Therefore, the main objective of this research is to develop a simplified generic data analysis tool, with a range of data analysis techniques and functionality that can be manually applied to a small subset of data of various types and then rerun on the full dataset using the playback enhancement to save time and computing resources.

1.3 Research Objectives

The sub-objectives that need to be achieved to satisfy the primary research objective are the following:

i. To ascertain whether a simplified data analysis tool for data analysis across a variety of different data types is feasible.

ii. To determine to what extent a macro-like record and playback macro feature could be used to enhance the tool's usability by running batch mode analysis, that is, executing previously recorded commands without user interaction on a larger dataset.

iii. To determine whether parallelization can enhance the performance of the data analysis process.

1.4 Approach

The first phase involved understanding the data analysis process and various analytical tools and techniques available as well as their feasibility and applicability to data analysis in general using the available literature. For example, data analysis tools help organizations implement real-time or non-real-time monitoring of events, processes, endpoints, network traffic etc., consolidate and coordinate diverse event data from application and network logs, and perform analysis to better understand their overall and specific states,

all of which allow for informed decision making. Grouped together, these functions help professionals in different industries assess how business processes are affected in different contexts and develop solutions or improvements from information gained after appropriate data analysis.

The second phase involved assessing weaknesses and risks in the data analysis process. Through experimentation with common tools available in various industries, common techniques and areas where there is potential for enhancement could be identified. This phase identified enhancements in terms of time and resource utilization.

The third phase of the research was to develop a prototype data analysis tool that made use of a variety of available datasets for data analysis to address some of the weaknesses and risks identified in the second phase by using various techniques and tools (scripts, parsers, etc.) These datasets were used with the aim of creating a generic design for the prototype data analysis tool. The expertise gained in analysing the various data was applied to specific network packet, log and stock datasets to show that the developed tool was widely applicable across a broad spectrum of users with small tweaks and changes to conform to specific environments.

Finally, from the information gathered, enhancements to the prototype data analysis tool were implemented. These included a macro-like record and playback macro feature as well as parallelization to improve the overall efficiency and effectiveness of the SDA tool.

1.5 Limitations of the Research

This research focuses on a generic approach to data analysis using three case studies with different data types and contexts to produce a proof of concept. This research only considered alpha-numeric data and the scenarios were based on common analysis tasks carried out in various industries. Thus, only a selection of common analysis processes and data types are explored. Data analysis techniques involving machine learning and other artificial intelligence techniques have not been considered in this prototype SDA tool.

1.6 Structure of Thesis

The remainder of this thesis comprises the following chapters:

Literature Review (Chapter 2) - discusses data, data analysis, data science, relevant data analysis tools and applications, data analysis opportunities and challenges and related work.

High Level Methodology and Research Design (Chapter 3) - outlines the research approach and methodology, the prototype SDA tool design, the record and playback macro feature, other enhancements, design and implementation considerations and introduces the experimental datasets used in this research.

Prototype SDA Tool Implementation (Chapter 4) - provides a detailed overview of the implementation of the SDA tool and its enhancements.

A Case Study using Network Data (Chapter 5) - discusses network data and documents the data analysis process through the analysis scenarios carried out.

A Case Study using Log Data (Chapter 6) - discusses log data and documents the data analysis process through the analysis scenarios carried out.

A Case Study using Apple Stock Data (Chapter 7) - discusses stock data and documents the data analysis process through the analysis scenarios carried out.

Subjective Analysis of the Software Tool (Chapter 8) - presents the results based on the indicators outlined and discusses the results.

Conclusion (Chapter 9) - discusses the thesis findings with concluding remarks, as well as mentioning future work that could be carried out.

Chapter 2

Literature Review

In exploring the generality of data analysis and enhancements of tools and techniques that can derive value from data, this chapter discusses relevant literature on data, data analysis, data science, existing tools and techniques, and related work that has been done in this area. Some data analysis enhancements have been implemented and there are areas that are continually discovered and improved as the environment in which data exists is dynamic, with reference to technologies used and industries that adopt data analysis. In Section 2.5, existing tools and techniques, data analysis challenges and opportunities are discussed that helped identify potential areas of improvement. This literature review provides a solid foundation to the research focussed on the main research objective outlined in Chapter 1.

2.1 Definitions

In addition to the definitions of data, data analysis and data analytics in Chapter 1, Table 2.1 outlines additional definitions and expands the earlier definitions as a foundation for the discussion of the literature. Some of these definitions will be elaborated on in the sections that follow. However, others like "Data Science" are not applicable to this research, but have been given in the table to help differentiate between various similar terms in the field.

2.2 Data

The main resource for this research is data from which value can be derived; how this value can be derived forms the foundation of this research. Data are facts about anything

Table 2.1: Terms in used Data Science (Cao, 2017a)

Term	Description
Advanced analytics	Refers to theories, technologies, tools, and processes that enable an in-depth understanding and discovery of actionable insights in big data, which cannot be achieved by traditional data analysis and processing theories, technologies, tools, and processes.
Big data	Refers to data that are too large and/or complex to be effectively and/or efficiently handled by traditional data-related theories, technologies, and tools.
Data analysis	Refers to the processing of data by traditional (e.g., classic statistical, mathematical, or logical) theories, technologies, and tools for obtaining useful information and for practical purposes.
Data analytics	Refers to the theories, technologies, tools, and processes that enable an in-depth understanding and discovery of actionable insight into data. Data analytics consists of descriptive analytics, predictive analytics, and prescriptive analytics.
Data science	Is the science of data.
Data scientist	Refers to those people whose roles very much center on data.
Descriptive analytics	Refers to the type of data analytics that typically uses statistics to describe the data used to gain information, or for other useful purposes.
Predictive analytics	Refers to the type of data analytics that makes predictions about unknown future events and discloses the reasons behind them, typically by advanced analytics.
Prescriptive analytics	Refers to the type of data analytics that optimizes indications and recommends actions for smart decision-making.
Explicit analytics	Focuses on descriptive analytics typically by reporting, descriptive analysis, alerting, and forecasting.
Implicit analytics	Focuses on deep analytics, typically by predictive modelling, optimization, prescriptive analytics, and actionable knowledge delivery.
Deep analytics	Refers to data analytics that can acquire an in-depth understanding of why and how things have happened, are happening, or will happen, which cannot be addressed by descriptive analytics.

in the world, physical or logical (Cuesta, 2013).

2.2.1 Types of Data

Data exists in two types, categorical or numerical (Cuesta, 2013). Figure 2.1 illustrates the categories of data. Numerical data can be measured and is also referred to as quantitative data. Numerical data is further divided into two categories, namely discrete and continuous data. Discrete data can only be distinct values, for example, the number of coin tosses. In continuous data, values can be any decimal number, not limited to any decimal place but usually rounded off, for example, speed of a vehicle.

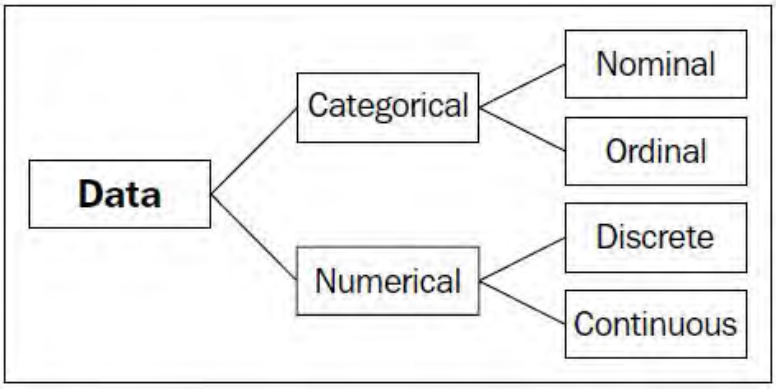


Figure 2.1: Categories of data taken from Cuesta (2013)

Categorical data on the other hand describes characteristics and can be sorted into groups or categories. It does not have mathematical meaning and is also known as qualitative data. It is further split into two categories, namely nominal and ordinal data. Nominal data have no defined ordering, for example gender is male or female. Ordinal data have a defined ordering, for example, age groups, such as young, adult, elder.

Data exists in three forms, structured, semi-structured and unstructured. Structured data is data that has a defined structure; it can be sorted or arranged in a set manner, semi-structured data can partially be sorted, while unstructured data consists as a variety of data formats (Wang *et al.*, 2018b). About 80% of the data used by organizations is unstructured (Bāliņa *et al.*, 2016). Traditionally data is stored in a fixed data format for which traditional storage systems have structured schemas (Abgaz *et al.*, 2018). With the advances in data technology, there has been similar developments in big data; big data consist of large growing data sets with heterogeneous formats, structured, semi-structured and unstructured (Oussous *et al.*, 2018). Big data is characterised by (a combination of) what is known as the 5Vs, namely volume, value, veracity velocity and variety (Oussous *et al.*, 2018; Wang *et al.*, 2018b). Volume represents large volumes of digital data, value denotes that there is hidden value in the data, veracity considers whether data is clean, consistent and complete, velocity denotes that data are generated at a high speed continuously, while variety denotes that the data is varied, with different structures and formats. These characteristics bring about challenges in data analysis.

Many types of data can be used for analysis but each requires a unique method of analysis. Audio, images, video and text can be read in various formats that exist, the common approach is then to have analysis tools convert the data to a working format that the tool uses to reduce errors and overheads during processing. The data collection space is rapidly expanding with many new data sources and formats becoming available (Power, 2016).

There is a variety of users that use data analysis tools; these users range from various educational and professional backgrounds with varying degrees of computer knowledge and skills, i.e. from technical to basic computer users. There are users that use a computer frequently in their day-to-day duties to achieve various organizational objectives and in varying organization positions or capacities. Some users use data analysis for reporting purposes and some use it as their main job, to analyse all organizational data and uncover or discover important information that other members of the organization can use to improve operations and solve problems.

2.2.2 Data, Information and Knowledge

Data goes through the process of analysis to produce information, and information has value which when applied in certain contexts or situations, becomes knowledge. Thus, information has meaning and is used to make informed decisions. Data, information and knowledge are context dependent and will usually be created and used in a single context, however, data, information and knowledge can cut across contexts and be used in related contexts (Donoho, 2017). An example of the data to knowledge flow is the following: if data on the sales of a particular product in a retail store is to be collected and analysed, information such as how frequent the product sales are and at what the times of day the sales occur, would be valuable for making decisions on what quantities of the product should be in stock and the specific times of the day to have adequate stock. Ideally, decisions to be made from the information can be based on experience or in this case knowledge, which is the understanding of information and an application of it in appropriate contexts for value creation. Figure 2.2 illustrates the direction of flow and the increase in volume as the data moves towards knowledge. Sato and Huang (2015) state that data without interpretation have no value and can only attain value when they are assessed and interpreted using existing knowledge. Information is the core element in forming knowledge and knowledge is essential to the processing of data and information, highlighting an interdependency that is important to understand when dealing with data (Donoho, 2017).

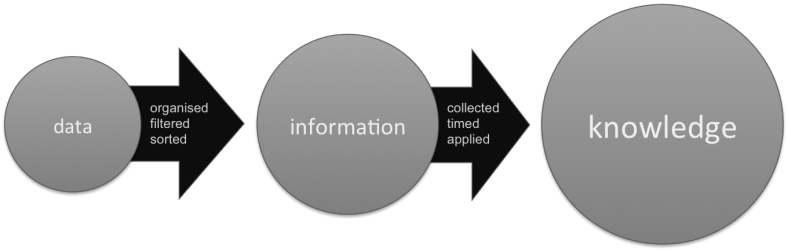


Figure 2.2: Flow of data to knowledge taken from Donoho (2017)

However, it is important to have some knowledge and understanding about the data to be analysed before the analysis begins in order to efficiently and effectively deduce information from the data (Duan *et al.*, 2020).

2.3 Data Analysis

Adding to the definition by Cuesta (2013) given in Chapter 1, Cuesta further states that data analysis is a detailed examination of data using statistical and mathematical measures with the aim of producing information and/or knowledge from data.

Optimization of business processes in organizations is increasingly becoming an important objective to improve an organization's sustainable competitive advantage (Zhang *et al.*, 2017a) and the potential of data analysis to provide insights for enhanced decision making is attracting much interest from all industry sectors (Sivarajah *et al.*, 2017). Data analysis has been widely adopted in various industries such as health care, retail, banking, financial markets, insurance, manufacturing, public transportation and many others, with the aim of obtaining valuable information from operational data that is generated and other relevant secondary data. Bāliņa *et al.* (2016) suggest that over 90% of the data in the world has been created in the last eight years, further illustrating the importance of making use of data. Data analysis tools must filter and analyse millions of events per second across a wide variety of data sources, including traditional data sources, such as log or audit files, and emerging sources such as images, social data, sensors and email. There are many steps and contexts in which data analysis is applied, with important decisions required and challenges to address at each step (Jagadish, 2015).

It is important to collect as much useful data as possible to supply data analysis tools with the raw data needed to detect patterns and provide insights. So, before deploying an analysis measure, it should be understood how such a solution can fit within an organization and the gaps it can help fill. Data analysis can add a layer of confidence to decision making, and can help organizations become competitive in different industries. In order to facilitate data analysis, a variety of tools and techniques are needed to carry out the data analysis processes, amongst which are analytical programming languages that allow analysts to customize analytical procedures and applications for focused solutions. However, there is also significant demand for analytical tools to be more interactive and provide data analysis without any low level knowledge of information technology (Bāliņa *et al.*, 2016).

Having knowledge of what occurred and why it occurred are no longer adequate; organizations now need to know what is happening currently, what is most likely to happen

and the measures to put in place for optimal results (LaValle *et al.*, 2011). Data helps answer questions by providing information with a summary of data. Data will answer a large number of questions using simple to complex statistical measures. Various statistical measures will summarise data differently and will also answer different types of questions. Power (2016) suggests that there be a rational analysis of data, analysis and data must be relevant and the analyst must be knowledgeable with relevant analytical methods and tools. According to Maxwell *et al.* (2018), data analysis requires a basic understanding of statistics and software development skills for minimum requirements. Data analysis techniques enable researchers to get better answers to existing questions, by increasing scope of data there is more precise effect size and confidence intervals are higher, with higher scope and granularity of data it is possible to examine new questions from existing or new theories (George *et al.*, 2016).

Data analysis is applied in most, if not all, industries with the purpose of gaining information from data to aid decision making. The appearance of new competitors and technologies is leading to increased competition and, with it, businesses have developed more efficient and effective ways of gaining competitive advantage by anticipating customer intentions (García *et al.*, 2017). Such anticipation can be the result of the correct application of information-based knowledge extraction from data through analysis (García *et al.*, 2017).

2.3.1 Data Analysis Methods

There are two types of data analysis methods: quantitative and qualitative methods. Each method has its own application depending on the data to be analysed and sometimes a combination of the two methods is used. Qualitative and quantitative methods are different but can complement each other because of the different strengths and logic required when trying to answer different questions (Richmond, 2006).

Qualitative Data Analysis

Qualitative data analysis deals with data represented in a verbal or narrative format, i.e. text, transcripts from video recorded conversations, photographs, and so on (Richmond, 2006). For data analysis to be carried out, data have to be coded and then themes are generated and combined with theoretical frameworks to develop conclusions due to the nature of qualitative data (Brown, 2019). Qualitative data analysis provides rich explanations and descriptions of processes within contexts, i.e. from the data, "to gain a better understanding of phenomenon through the experiences of those who have directly

experienced the phenomenon, recognizing the value of participants' unique viewpoints that can only be fully understood within the context of their experience" (Castleberry and Nolen, 2018, p.807). The descriptive character of qualitative approaches enables development of complex, holistic views in a natural environment, through an approach that can be outlined in five steps: compiling, disassembling, reassembling, interpreting, and concluding (Castleberry and Nolen, 2018). A qualitative method can adopt various research designs, assumptions, sampling, data collection and analysis methods, where the analysis is characterised by the open-ended nature of data.

Quantitative Data Analysis

Quantitative data analysis deals with analysis of numerical data by applying statistical, mathematical and computational techniques. Quantitative data analysis is also referred to as statistical analysis which uses numbers to summarise experiences or characteristics in a dataset by describing the data using descriptive statistics or inferring from the data using inferential statistics (Callaghan and Bee, 2018). Descriptive statistics organize and describe a dataset while inferential statistics allow for generalisation beyond a dataset and can be used to compare two or more datasets. Similar to qualitative method, quantitative method can adopt various research designs, assumptions, sampling, data collection and analysis methods, however, the analysis is characterised by the close-ended nature of data.

2.3.2 Data Analysis Approaches

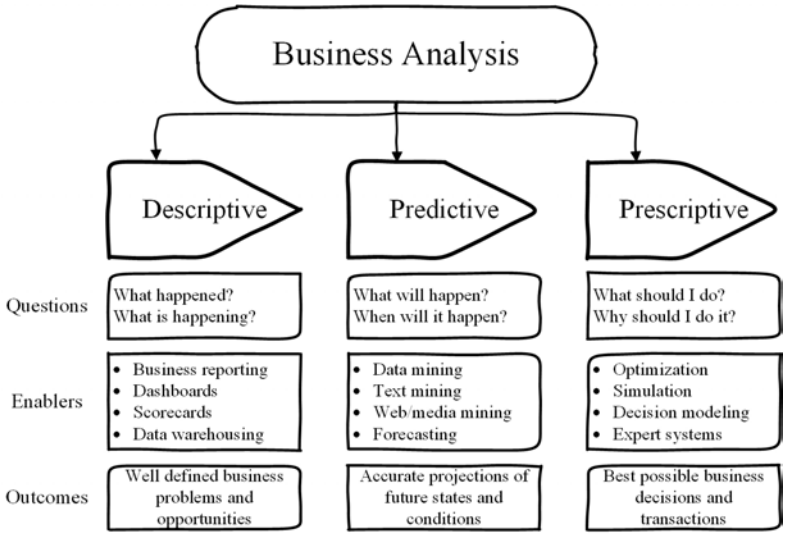


Figure 2.3: General data analysis taxonomy, taken from Delen and Demirkan (2013)

Data analysis for business analysis can be approached in three main ways namely descriptive, predictive and prescriptive forms (Sun *et al.*, 2018) or a combination of these. Delen and Demirkan (2013) elaborate on the above categories with a taxonomy of data analysis simplified in Figure 2.3.

Descriptive data analysis, also referred to as business reporting, uses data to answer the question of "what happened and/or what is happening?". It covers basic and frequent business reporting, customized reporting as well as dynamic/interactive reporting. with the main output being the identification of business opportunities and problems (Delen and Demirkan, 2013). Predictive data analysis uses mathematical and statistical techniques to discover explanatory and predictive patterns (e.g. trends and associations) representing the inherent relationships between data inputs and outputs. In essence, it answers the question of "what will happen and/or why will it happen?". Enablers of predictive analysis include data mining, text mining, Web/media mining and statistical time-series forecasting. The main outcome of predictive modelling is an accurate projection of the future occurrences and the reasoning as to why it will occur (Delen and Demirkan, 2013). Prescriptive data analysis uses mathematical and statistical algorithms to determine a set of high-value alternative courses of actions or decisions given a complex set of objectives, requirements, and constraints, with the goal of improving business performance. These algorithms may rely solely on data or expert knowledge, or a combination of both. Enablers of prescriptive analysis include optimization modelling, simulation modelling, multi-criteria decision modelling, expert systems and group support systems. The main outcome of prescriptive modelling is either the best course of action for a given situation, or a rich set of information and expert opinions provided to a decision maker that could lead to the best possible course of action (Delen and Demirkan, 2013).

2.3.3 Data Analysis Process

Cuesta (2013) defines the data analysis process to include the problem, data preparation, data exploration, predictive modelling and visualization of results.

The problem includes high-level questions about the data and context based on the approach taken as stated in Section 2.3.2. This is done in order to develop and understand objectives and data analysis requirements. Data preparation is concerned with cleaning, normalizing and transforming data into a clean, coherent, unambiguous, countable, correct, standardized and non-redundant dataset, i.e. quality data. Data exploration refers to carrying out various analysis scenarios using graphical and statistical methods to find patterns, connections and relationships in the data. Predictive modelling is the process in data analysis that creates or chooses a statistical model to best predict the probability of

an outcome. Finally, visualisation of results is the process of choosing a method to best present the results of the analysis process.

On the other hand, Oracle experts have outlined a seven-stage business analysis process (Bāliņa *et al.*, 2016) as stated below:

- Definition of requirements: the understanding of business process objectives and requirements, the definition of analysis target.
- Data acquisition: including data collection and preparation for further analysis.
- Data understanding: learning about the data in its context.
- Manipulation of data: the data filtering, elimination, as well as removing of incorrect data.
- Data publishing: making the data and information available to others.
- Data analysis: application of a model corresponding to a business process and given data, model estimation and learning;
- Decision making and taking action: model evaluation, testing, comparison, and application of the model.

The above categorisation of processes differs in how the various data analysis processes are grouped when compared to the definition by Cuesta (2013). The two approach methods by Cuesta (2013) and Bāliņa *et al.* (2016) are consolidated and discussed in detail below.

Step 1: Data Collection

Data collection is the acquisition of unprocessed data from a real-world environment and proficiently development of the data (Saggi and Jain, 2018). Depending on the analysis to be done and since data can come from different sources and can take different forms (i.e. heterogeneous data from multiple sources), the best practice is to transform the data with the aim of consolidating, sorting and cleaning it (Elshawi *et al.*, 2018). Data collection deals with all the data sources needed for analysis. Paakkonen and Pakkala (2015) suggest that data sources can be described using two dimensions, namely mobility and structure of data. The first dimension, namely, mobility refers to data that is immobile (in situ data that is stored in a file or database but it can be modified however, the rate of change is not high, for example a CSV file containing a set of data) and mobile (data that is continuously being generated for example streamed data, data flows in real-time,

transactional data will increase with time as new transactions are added) (Paakkonen and Pakkala, 2015). The second dimension, namely, structure refers to the structure of the data stored in databases, either structured, semi-structured or unstructured (Wang *et al.*, 2018b). Examples of structured data include digits, symbols and tables; semi-structured data examples include trees, graphs and extensible mark-up language (XML) documents and unstructured data examples include logs, audio, videos and images.

Data can be collected from where it is stored by systems that generate the data, such as databases (MySQL, SQL, Oracle etc.) or data files (such as Excel files, comma separated values (CSV) files, XML files, text etc.) In the most complex case, the data will be unstructured, multivariate and large in volume. With advances in information technology data collection is often limited by the researcher's imagination rather than by technological tools (George *et al.*, 2016). Technologies such as cloud computing, Internet-of-Things have positively affected scalability, flexibility and performance needed to analyse large volumes of data allowing companies to store data for longer periods and efficiently manage huge data sets (Oussous *et al.*, 2018). Several data sources for data collection are discussed below;

Text Files Text files are commonly used for data storage because of the flexibility to convert different file formats that can work on a varied range of applications. Moreover, text files are easier to recover (Cuesta, 2013). Large amounts of data come in different forms of text files such as CSV, XML and JavaScript Object Notation (JSON) files, while the data can be logs, emails, transactions, an so on.

Excel Files Microsoft Excel is a widely used office application for storing, organizing and analysing data (Needham, 2018). Microsoft Excel offers a work-space for data and a wide range of functions for data analysis through calculations, statistical operations and data presentation through different graphs and tables. Microsoft Excel files can also be converted to other formats such as CSV.

SQL Databases Cuesta (2013) defines a database as an organized collection of data. Structured Query Language (SQL) is a database language for managing and manipulating data in a database. A common configuration of a database is the relational database which is based on the relational model of data (i.e. based on logical relationships) (Cuesta, 2013). Databases store data in tables. The Data Definition Language (DDL) and the Data Manipulation Language (DML) form the SQL language. DDL allows for table creation, deletion and alteration, while DML allows for access and manipulation of data.

NoSQL Databases Not Only SQL (NoSQL) is a term used in technologies where the nature of data does not require a relational model (Cuesta, 2013). NoSQL is able to work with huge quantities of data with varied formats, while providing for acceptable performance, scalable and high availability.

Web Scraping Data can also be copied from websites using applications that read from different websites such as weather or market data.

Analytical tools are useless if they produce incorrect, incomplete and inconsistent results. Thus, examination of the source data is essential before an integration process of analytical tools and the data sources is carried out. This is a quality check that promotes desired analysis in later processes the data goes through (Azeroual *et al.*, 2018).

Throughout the data analysis process as discussed here, it is assumed that data requirements have been set and a design developed for the whole process. After successful identification of the data, the next phase begins. Data analysis requires data storage to store data being worked on. The type of data storage will depend on the type of data being analysed. Traditional data and big data storage systems both have their advantages and disadvantages. Data warehousing is a technique for management of data that has multiple sources, this data is the aggregated, organized and stored in a single database for more convenient access (Desiderio, 2019).

Step 2: Data Processing

Extraction Transformation and Loading (ETL) This process deals with handling of data stored from the data source process and includes three processes performed on the data, namely, extraction/data acquisition, data transformation and loading/storage (Wang *et al.*, 2018b). The ETL process is synonymous with data preparation and data preprocessing. Data preparation transforms data into a structure that can more easily and efficiently be processed (Losarwar and Joshi, 2012).

- Extraction deals with copying data from the data source to a data warehouse from where the next process will access it. In-situ or streamed data is captured into a temporary data store to be read, and where a rough analysis looking at the frequencies, size and formats is carried out. This process is referred to as preprocessing. The data may be compressed to improve efficiency in the processes to come (Paakkonen and Pakkala, 2015).

- In the transformation step the data is moved, cleaned, split, translated, merged, sorted and validated, this process is aimed at improving quality of analysis. The data is then stored into a preparation data store.
- The loading process deals with loading the data into another data warehouse where analysis will occur. This is referred to as data exploration where statistical methods are applied to the data. Data in the preparation data store can be replicated between data stores to allow for further analysis while maintaining data integrity thereby maintaining or improving quality of the results.

The ETL process requires an appropriate type of data store for all the requirements to be met. It is difficult to directly manage unstructured data within enterprise relational databases. Object-based storage architecture enables collections of data to be stored and managed as objects, which provides a more flexible solution for integrating semi-structured and unstructured data (Tao *et al.*, 2018).

Step 3: Data Analysis

Data analysis is then carried out to further process all kinds of data by applying statistical and mathematical methods using various analytical techniques to gain information from data. At this stage there is the need to refer to the requirements set at the beginning of the process and implement tools and methods as specified to address the objectives. For example there would be no need to use predictive algorithms if the aim is merely to look at the trend and identify a problem. For this research however, all available techniques will be considered and those generally applicable will be implemented. Below are some data analysis techniques used (Saggi and Jain, 2018):

- Classification: to predict the categories of input data for e.g. weather attributes are sunny, windy, rainy etc.
- Regression: to predict numeric value e.g. price of stocks.
- Clustering: to organize similar items in-to groups e.g. grouping a company in senior, adults, and teenagers.
- Association analyses: to find interesting relationships between sets of variables.
- Graph analyses: to use graphic structure to find connections between entities.
- Decision tree: to predict modelling insights of objective variables by learning simple decision rules inferred from the data features.

In addition to the above techniques there are other advanced techniques such as machine learning, advanced statistics and data mining which utilize a combination of data analysis techniques. Hadoop Map/Reduce, stream computing and in-database analysis can each be used independently of the others or in combination depending on the purpose of the analysis. Data analysis processes utilize work-flows to integrate analysis methods, which helps to simplify the process and reduce time of production (Wang *et al.*, 2018b).

Step 4: Data Visualisation

Data visualization is the presentation of data in a pictorial or graphical format, and a data visualization tool is the software that generates this presentation. Data visualization provides users with intuitive means to interactively explore and analyse data, enabling them to effectively identify interesting patterns, infer correlations and causalities, and supports sense-making activities (Bikakis, 2018). This step aims to present data in a way that is understandable to the user or to visualize the results of data analysis in a convenient way (Oussous *et al.*, 2018). "Visualizing data is about representing key information and knowledge more instinctively and effectively through using different visual formats such as in a pictorial or graphical layout" (Sivarajah *et al.*, 2017, p.273). Presentation of information deals with the production of output by using data modelling and visualizations. Models are developed and tested to maintain quality standards and to identify any patterns. Visualizations are tools used to represent the results of analysis. Visualizations, real-time information processing and meaningful business insights can be achieved with complete data (Wang *et al.*, 2018b).

According to Wang *et al.* (2015), there are several methods available that are used to present data, where each method is unique in the way data is displayed. There are conventional data visualization methods such as tables, histograms, scatter plots, line charts, bar charts, pie charts, area charts, flow charts, bubble charts, combination charts, time line, Venn diagrams, data flow diagrams, entity relationship diagrams, parallel coordinates, tree maps, cone trees, semantic networks and so on. Each method has a set of data requirements and how information is presented. Visualization can highlight problems with data, for example, uncovering errors in the data. However, visualizations cannot replace critical thinking and will not always show an accurate picture as throughout the whole data analysis process data can be manipulated (Wang *et al.*, 2015). There are several kinds of visualization tools that can be used to present data.

- Dashboards are applications that use a simple user interface to display key performance indicators but has limited user control presentation is on the data analysed.

- Visualization applications offer greater detail and control to analysed data (Paakkonen and Pakkala, 2015).

Data visualization in this context is to do with the design and implementation of graphical representation in different forms such as tables, images, diagrams as forms to present data (Saggi and Jain, 2018). Visualizations can be static, where they present the data from a pre-set configuration or can be interactive, where the user is able to refine the output, for example filter data.

2.4 Data Science

Data science is an inter- and cross-disciplinary field (trans-disciplinary) that studies data by developing a more sophisticated and systematic analysis of it (Cao, 2017b; Power, 2016). According to Cao (2017b), data science builds on and synthesizes statistics, informatics, computing, communication, sociology, management and the general knowledge domain. Data science focuses on systematic understanding of data and related problems (Saggi and Jain, 2018). Data must be analysed to inform decision-making, the right data (relevant, accurate and timely) is needed to conduct appropriate analysis (Power, 2016). We live in a world where we value data and most importantly we are able to collect varied data from many of the activities we do on a day-to-day basis; for example, websites can track most of our actions, our smart phones can monitor different aspects of our lives such as heart rate, sleeping patterns, screen time and other habits. The "Smart" term is coming into every aspect of life:smart home, smart car, smart office, etc. Most of these smart devices come with sensors to address our different needs automatically without us having to lift a finger in some instances. These devices might connect to the Internet for management and data collection and a collection of these devices and their systems is referred to as "The Internet of Things" (IOT) (referring to a connectivity of everyday objects to the internet) (Stojkoska and Trivodaliev, 2017). IOT devices generate vast amounts of data continuously and the data are collected at a high rate and volume (Grus, 2019). With all this data, different entities/organizations can use the data differently depending on need or aim. For example, some organizations might use data to analyse how to implement a presidential campaign by amongst other things, identifying which voters to focus on and how. Another example can be to track sales of goods and see which goods do well at certain times of the day, week, month or year. The ultimate goal for most organizations is to aid decision making, i.e. decision support by reporting on current state, aiding prediction of future events and prescribing or recommending appropriate actions.

Data science has come a long way, in the last 50 years with efforts that developed from statistics and further refinement that has made it to a trans-disciplinary field (Donoho,

2017). Data science is used in most industries; it has enabled data-driven theories, economies and professional development to be recognized with terms such as data analysis, data analytics, advanced analytics, big data, data science, deep analytics, descriptive analytics, predictive analytics, and prescriptive analytics continually being explored in data science (Cao, 2017a). In summary, Blei and Smyth (2017) state that data science focuses on exploiting the modern deluge of data for prediction, exploration, understanding, and intervention. It emphasizes the value and necessity of approximation and simplification. It values effective communication of the results of a data analysis and of the understanding about the world that we glean from it. It prioritizes an understanding of the optimization algorithms and transparently managing the inevitable trade-off between accuracy and speed. It promotes domain-specific analyses, where data scientists and domain experts work together to balance appropriate assumptions with computationally efficient methods.

2.4.1 Data and Analytics

Data analytics involves the process of analysing heterogeneous data to mine insightful information through discovery of unknown patterns by applying various techniques and methods such as predictive algorithms, semantic analysis, statistical analysis methods, and technologies (Saggi and Jain, 2018). Knowledge of analytical methods and techniques is essential for uncovering hidden patterns in data. Analytical techniques can range from simple to complex descriptive statistics, data visualization methods, mathematical methods and statistical analysis algorithms such as regression, correlation analysis, and support vector machines (Talabis *et al.*, 2014). Data analytics is part of a broader term; data science and data analytics incorporates data analysis (Molina-Solana *et al.*, 2017). Data analysis is used to study a dataset's individual parts and extract information while data analytics encompasses the complete management of data.

It is important to thoroughly understand data in the context it occurs (Hoelscher and Mortimer, 2018). In most cases data analytics is used where data is produced, in business, in personal life, government and many other disciplines or knowledge domains. Data analytics is used primarily and generally to get value from data which can address a certain aspects of a context or situation. Data analytics is nowadays used for business operation transformation with the aim of surviving and competing in the future (Hinde, 2018).

2.4.2 Approaches to Data Analytics

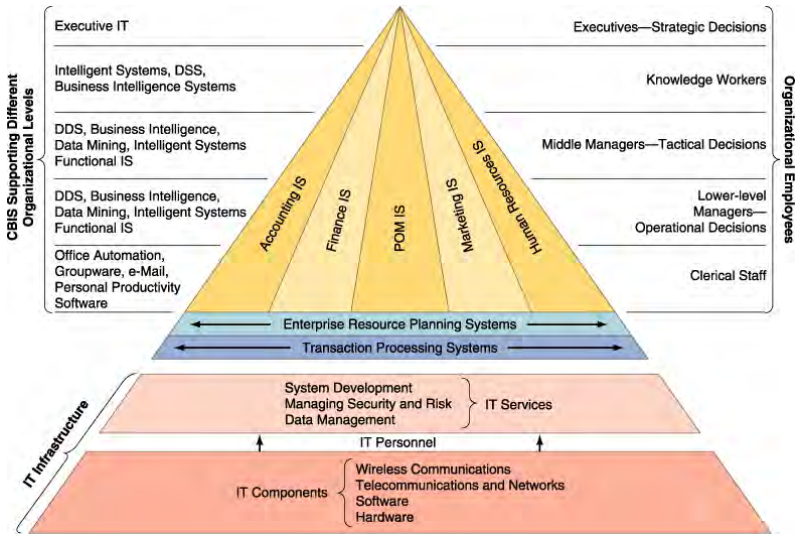


Figure 2.4: Data analytics architecture taken from Power (2016).

Data analytics is used in different contexts with varied forms and sizes of data but the aim is ultimately the same, and that is to gain insight that aids decision making (Power, 2016). Data analytics uses the data analysis steps as a foundation; data collection, data processing and data presentation (Zhang *et al.*, 2017b; Kumari *et al.*, 2019; Zhou *et al.*, 2018; Oussous *et al.*, 2018). Analytics applications use data from a variety of data sources of an organization's operational systems and external sources. Operational systems focus on day-to-day aspects of an organization hierarchy, collects, stores and maintains data generated from all business transactions. Operational systems are functional and most often focus on specific areas such as management of finance, human resources, supply chain, logistics etc. Data from functional units can be processed and aid in operational control and operational performance decisions. Operational systems are also referred to as Transaction Processing Systems (TPS). On the other hand Management Information Systems (MIS) gather data from TPS and processes the data to produce functional information. MIS are low-level analytical systems because they usually have access to data that relates to some (usually one) functional units. Executive Information Systems (EIS) gather information from MIS to have an overview of an organization through integration of information from most but usually all functional areas as illustrated in Figure 2.4.

According to a study done by LaValle *et al.* (2011) across 30 industries and 100 countries on how effective data analytics is, the study found that top-performing organizations use data analytics five times more than lower performers and one of the reason was the questions that were asked for analytics were effective in their context to produce positive outcomes (see Figure 2.5). Beginning with questions before collecting data has proven

to be effective and efficient, by defining the target areas first, organizations use available data and with insights from the outcome, gaps can be identified in data infrastructure and business processes (LaValle *et al.*, 2011).

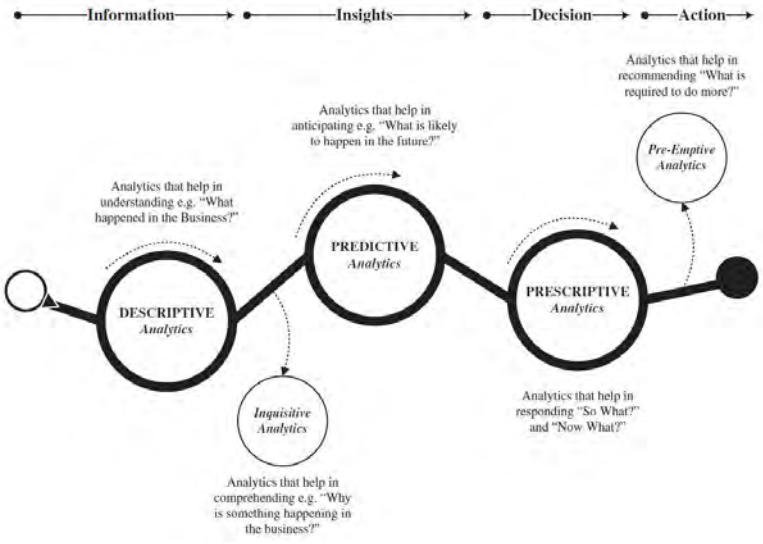


Figure 2.5: Data analytics methods, taken from LaValle *et al.* (2011)

The types of questions center around "what happened and/or what is happening? How things happened or are happening?" for the basic form of data analytics, i.e. descriptive, then to a more predictive form of "what will happen and/or why will it happen?" then based on the predictive information recommendations and solutions can be developed. Questions are asked based on an organization's objectives, ideally they must ask around issues that affect the objectives so that if they are answered they impact positively to the organization. The leading challenge to effective use of data analytics is the lack of understanding of data analytics to improve a business as a result, this makes it difficult to ask the right questions correctly (LaValle *et al.*, 2011).

2.5 Existing Software Tools

There are numerous tools and techniques used across industries with variations in their requirements due to differences in environments. For example, some common data analysis tools include R Programming[1], Tableau[2], Microsoft Excel[3], Python[4], Microsoft PowerBI[5],

[1]https://www.r-project.org/
[2]https://www.tableau.com/
[3]https://www.microsoft.com/en-za/microsoft-365/excel
[4]https://www.python.org/
[5]https://powerbi.microsoft.com/en-us/

SAS[6], Apache Spark[7], KNIME[8], Rapid Miner[9], OpenRefine[10] and many others. Some of these are discussed in more detail below.

2.5.1 Python

Python is an interpreted high-level general-purpose programming language. Python has a wide range of applications including web and Internet development, database access, desktop graphical user interface, scientific and numeric, education, network programming and software and game development. Python is developed under an Open Source Initiative approved open source license, making it freely usable and distributable, even for commercial use.

Python has the following libraries that enable it to handle data:

- Pandas[11] is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.
- SciPy[12] is a Python-based ecosystem of open-source software for mathematics, science, and engineering.
- NumPy[13] is a fundamental package for scientific computing with Python. It contains among other things:
  - a powerful N-dimensional array object
  - sophisticated functions
  - useful linear algebra, Fourier transform, and random number capabilities

  Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
- Matplotlib[14] is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

[6] https://www.sas.com/
[7] https://spark.apache.org/
[8] https://www.knime.com/
[9] https://rapidminer.com/
[10] https://openrefine.org/
[11] https://pandas.pydata.org/
[12] https://www.scipy.org/
[13] https://numpy.org/
[14] https://matplotlib.org/

- Django[15] is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source.
- SQLAlchemy[16] is a Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL. It provides a full suite of well known enterprise-level persistence patterns, designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.

2.5.2 R Programming

R is an open-source language and environment for statistical computing and graphics. It is a GNU project[17] which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R provides a wide variety of statistical (linear and non-linear modelling, classical statistical tests, time-series analysis, classification, clustering, etc.) and graphical techniques, and is highly extensible. R is available as free software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.

One of R's strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control. R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes:

- an effective data handling and storage facility,
- a suite of operators for calculations on arrays, in particular matrices,
- a large, coherent, integrated collection of intermediate tools for data analysis,
- graphical facilities for data analysis and display either on-screen or on hardcopy, and
- a well-developed, simple and effective programming language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

[15] https://www.djangoproject.com/
[16] https://www.sqlalchemy.org/
[17] https://www.gnu.org/home.en.html

The term *environment* is intended to characterize it as a fully planned and coherent system, rather than an incremental accretion of very specific and inflexible tools, as is frequently the case with other data analysis software. R is designed around a true computer language, and it allows users to add additional functionality by defining new functions. Much of the system is itself written in the R dialect which makes it easy for users to follow the algorithmic choices made. For computationally-intensive tasks, C, C++ and Fortran code can be linked and called at run time. Advanced users can write C code to manipulate R objects directly.

R an environment within which statistical techniques are implemented and can be extended via packages. There are about eight packages supplied with the R distribution and many more are available through the CRAN family of Internet sites covering a very wide range of modern statistics. R has its own LaTeX-like documentation format, which is used to supply comprehensive documentation, both on-line in a number of formats and in hard copy.

2.5.3 Microsoft Excel

Microsoft Excel is a commercial data analysis application developed by Microsoft. It can be purchased as a single product or together with other Microsoft Office products/applications such as Microsoft Word and Powerpoint. Microsoft Excel (or just Excel) is a spreadsheet program, which contains a number of columns and rows, where each intersection of a column and a row is a *cell*. Each cell contains one point of data or one piece of information. By organizing the information in this way, it is easier to find information or draw information from changing data.

Excel offers an array of data analysis features and functionality from data preparation to data visualisation. An important feature of the Excel spreadsheet program is that it allows for the creation of formulas that calculate results. Without formulas, a spreadsheet is not much more than a large table for displaying text or data. A formula is an equation that makes calculations based on the data in a spreadsheet. Formulas are entered into a cell and generates a result that is displayed in the cell. The formulas allow for statistical and mathematical measures and methods to be applied to data. Other functionality available in Excel included sorting and filtering of data. Charts provide functionality to visualise data, with a variety of charts such as bar charts, pie charts, pivot tables available in Excel.

Excel also has a feature that automates repetitive tasks called macros. A macro is a series of commands that one can use to automate a repeated task[18]. Macros consist of

[18]https://support.microsoft.com/en-us/office/macros-in-office-files-12b036fd-d140-4e74-b45e-16fed1a7e5c6

program code that runs in an Excel environment. Basic knowledge of the Visual Basic programming language is needed to make advanced modifications in the macro, but not for carrying out basic macros. Using macros basically involves initializing a recording, then performing the data analysis task and ending the recording when all tasks have been completed.

2.5.4 Microsoft Power BI

Power BI[19] is a collection of software services, applications, and connectors that work together to turn your unrelated sources of data into coherent, visually immersive, and interactive insights. Power BI can access various types of data sources such as Excel spreadsheets, or a collection of cloud-based and on-premises hybrid data warehouses, to connect to and visualise any data using the unified, scalable platform for self-service and enterprise business intelligence.

Power BI is a paid service from Microsoft, it consists of several elements that all work together to provide data analysis functionality, a Windows desktop application called Power BI Desktop, an online SaaS (Software as a Service) service called the Power BI service and Power BI mobile applications for Windows, iOS, and Android devices. Power BI offers functionality for the data analysis process via a desktop application, or mobile application or web application.

2.5.5 Apache Spark

Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size. It provides development Application Programming Interfaces (API) in Java, Scala, Python and R, and supports code reuse across multiple workloads batch processing, interactive queries, real-time analytics, machine learning, and graph processing. It is used by organizations from many industries. Apache Spark has become one of the most popular big data distributed processing frameworks.

Apache Spark started in 2009 as a research project at UC Berkley's AMPLab, a collaboration involving students, researchers, and faculty, focused on data-intensive application domains. The goal of Spark was to create a new framework, optimized for fast iterative processing like machine learning, and interactive data analysis, while retaining the scalability, and fault tolerance of Hadoop MapReduce.

[19] https://docs.microsoft.com/en-us/power-bi/fundamentals/power-bi-overview

Today, Spark has become one of the most active projects in the Hadoop ecosystem, with many organizations adopting Spark alongside Hadoop to process big data. Hadoop MapReduce is a programming model for processing big data sets with a parallel, distributed algorithm. Developers can write massively parallelized operators, without having to worry about work distribution, and fault tolerance. However, a challenge to MapReduce is the sequential multi-step process it takes to run a job. With each step, MapReduce reads data from the cluster, performs operations, and writes the results back to Hadoop Distributed File System (HDFS). Because each step requires a disk read, and write, MapReduce jobs are slower due to the latency of disk I/O.

Spark was created to address the limitations to MapReduce, by doing processing in-memory, reducing the number of steps in a job, and by reusing data across multiple parallel operations. With Spark, only one-step is needed where data is read into memory, operations performed, and the results written back resulting in a much faster execution. Spark also reuses data by using an in-memory cache to greatly speed up machine learning algorithms that repeatedly call a function on the same dataset. Data re-use is accomplished through the creation of DataFrames, an abstraction over Resilient Distributed Dataset (RDD), which is a collection of objects that is cached in memory, and reused in multiple Spark operations. This dramatically lowers the latency making Spark multiple times faster than MapReduce, especially when doing machine learning, and interactive analytics.

There are many benefits of Apache Spark to make it one of the most active projects in the Hadoop ecosystem. These include:

- Speed; through in-memory caching, and optimized query execution, Spark can run fast analytic queries against data of any size.
- Developer friendly; Apache Spark natively supports Java, Scala, R, and Python, providing a variety of languages for building applications. These APIs make it easy for developers, because they hide the complexity of distributed processing behind simple, high-level operators that dramatically lowers the amount of code required.
- Multiple workloads; Apache Spark comes with the ability to run multiple workloads, including interactive queries, real-time analytics, machine learning, and graph processing. One application can combine multiple workloads seamlessly.

Apache Spark has been deployed in every type of big data use case to detect patterns, and provide real-time insights. Example use cases include:

- Financial Services; Spark is used in banking to predict customer churn, and recommend new financial products. In investment banking, Spark is used to analyse stock prices to predict future trends.
- Healthcare; Spark is used to build comprehensive patient care, by making data available to front-line health workers for every patient interaction. Spark can also be used to predict/recommend patient treatment.
- Manufacturing; Spark is used to eliminate downtime of internet-connected equipment, by recommending when to do preventive maintenance.

2.5.6 Tableau

Tableau is a commercial visual analytics platform. It helps visualize data that can be understood by professionals at any level in an organization. It also allows non-technical users to create customized dashboards. Some features of Tableau software are data blending, real time analysis and collaboration of data. Tableau does not require any technical or any kind of programming skills to operate.

Tableau connects and extracts the data stored in various places. Simple databases such as an Excel or PDF files, to complex databases like Oracle or those in the cloud such as Amazon Web Services, Microsoft Azure SQL database, Google Cloud SQL can be extracted by Tableau. When Tableau is launched, data connectors are available to provide connections to any database. Depending on the version of Tableau that has been purchased the number of data connectors supported by Tableau will vary.

The accessed data can either be connected live or extracted to the Tableau's data engine, which is where data analysts and data engineers work with the data and develop visualizations. The created dashboards are shared with the users as a static file. The users who receive the dashboards view the file using the Tableau reader.

The data from the Tableau desktop can be published to the Tableau server. This is an enterprise platform where collaboration, distribution, governance, security model and automation features are supported. With the Tableau server, the end users have a better experience in accessing the files from all locations, be it a desktop or mobile or via email.

2.6 Data Analysis Opportunities and Challenges

The data science discipline is still evolving, with new ideas and new technologies continually being discovered and improved throughout various data science processes. Moreover,

a comprehensible understanding cannot be fully established due to the dynamic nature of the discipline and scope which leads to a significant amount of research in the area (Sivarajah *et al.*, 2017). Data analysis is seen to improve operational efficiency, strategic potential and discover new revenue streams to gain a competitive edge in a business environment. Big data is not a recent discovery as the data science movement started some 50 years ago; however, the potential impact of it is being realized due to the effect of new technologies such as cloud computing, more powerful hardware components (computer processing units, memory, storage) and research (Wang *et al.*, 2018a). Data analysis is made up of two components, the physical and logical. The physical aspect of data analysis is largely to do with the computing components stated above. The logical aspect relies on the physical aspect's capacity; however, the logical aspect is required to maximize utilization of physical resources to collect data, process data and present information efficiently and effectively (Desiderio, 2019) .

2.6.1 Opportunities

Data analysis enables competitive advantage by improving business efficiency and effectiveness because of its high operational and strategic potential and is thus an important aspect of decision making in business processes such that it is a major differentiator between high performing and low performing organizations (Wamba *et al.*, 2017). Wang *et al.* (2018a) highlight five opportunities for data analysis and data analytics: discovery of patterns from data, ability to gain value from unstructured data, decision support, predictive ability and traceability.

Another opportunity with respect to data analysis is the process of automation. Bezerra *et al.* (2020) define automation as the use of mechanical and instrumental devices to replace, refine, extend, or supplement human effort and faculties in the performance of a given process to achieve little to no human intervention in controlled operations. The authors further highlight that there are generally two types of automation: automized and automated devices. Automized devices are skilled and perform specific operations at a specific point or points in a process. Automated devices and processes can function with little or no human intervention.

2.6.2 Challenges

Data analysis problems require systematic thinking, methodologies and approaches to develop solutions. It requires data and intelligence-driven solutions and tools to represent, learn, stimulate, reinforce and transfer human-like intuition, imagination, curiosity and

creative thinking (Cao, 2017b). Sivarajah *et al.* (2017) identify various challenges and group these into three categories:

- Data challenges – which deal with data characteristics of the 5Vs.
- Process challenges – which deal with techniques applied in the data analysis process. Another challenge of the process is that researchers reduce, exclude or include and elaborate on data making the interpretation subjective because it then becomes shaped by the researcher's views and assumptions (Brown, 2019).
- Management challenges – which are to do with governance and ethical aspects.

2.7 Related Work

This section discusses some of the work carried out in the area of data analysis. Prior generations of data analysis products were mostly used as tools to perform retrospective analysis and forecasts on breaches after the fact. This is still an important function, but today's data analysis solutions have evolved to deliver business value across a much broader range of circumstances, and to address a number of critical issues facing organizations of all sizes. Tao *et al.* (2018) explain that various advances in Internet technology, such as Internet-of-Things, computing, big data and artificial intelligence, have affected the manufacturing industry positively. There has been an increase in data produced, collected and analysed, which has given the industry an opportunity to transform to data driven strategies and become more competitive. Continued development improves the efficiency of organizations' routine management on an operational level (Zhang *et al.*, 2017a), and suggests organizations need more advances in data analysis to develop solutions to different strategic issues.

Zhang *et al.* (2017a) propose a framework for a Big Data driven life cycle management for optimization of decisions of Product Life-cycle Management in manufacturing through discovery of knowledge from data. Within the proposed framework, the availability and accessibility of data and knowledge related to the life-cycle can be achieved using a variety of mathematical models, algorithms and data mining techniques. A case study was presented to demonstrate the proof-of-concept of the proposed framework. The authors identified an increasingly important objective for manufacturing enterprises to improve their sustainable competitive advantage. The approach was to integrate the business processes of an organization and more effectively manage and utilize the data generated during life-cycle studies. With emerging technologies, product-embedded information

devices such as radio frequency identification tags and smart sensors to improve the efficiency of enterprises' routine management at an operational level. The results show that the proposed framework was feasible for adoption in industry, and could provide an overall solution for optimizing the decision-making processes in different phases of the whole life-cycle. The key findings and insights from the case study were summarized as managerial implications, which could guide manufacturers to ensure improvements in energy saving and fault diagnosis related decisions in the whole life-cycle. It was discovered that manufacturing enterprises needed a more advanced analysis approach to develop a solution on a strategic level from using such a Big Data life-cycle.

The big data and information security study by the Business Application Research Centre (BARC) and Kuppinger Cole delivered insights into the level of awareness and current approaches in information security and fraud detection in organizations around the world (Haas and Lorcher, 2016). It measured the importance, status quo and future plans of big data security analytics initiatives across different sectors. Furthermore, it presented an overview of the various opportunities, benefits and challenges relating to those initiatives, and outlines the range of technologies currently available to address those challenges.

A study by Monahan (2015) suggests that information security has always been a large producer and consumer of data. More sophisticated best practices and expanding compliance and regulatory requirements have almost exponentially accelerated the production and consumption of data. Event and activity logs have grown to big data proportions and the diversity of data being consumed has become significantly more varied. As a result, traditional log and event management tools and monitoring practices are becoming increasingly insufficient. To add to this, the success record of maintaining security for an environment is at an all-time low. Executives are being dismissed or forced to resign after a breach whether they knew about security issues prior to the breach or not. Threats seem to come from every angle. Not only are attackers consistently probing, but the attacks themselves are more persistent and difficult to block; once a foothold is achieved, detection and removal are also more difficult. The research by Monahan (2015) highlights how both management- and operations-level IT and information security practitioners are impacted by among other issues, lack of visibility into their environments. Security analytics tools provide practitioners with a way to meet their actionable threat intelligence needs for an appropriately prioritized, timely response to attacks. It also demonstrates that the information security discipline needs next generation analytics capabilities to be successful in the age of advanced and persistent threats. Security analytics combines multiple capabilities for detection and response, each of which is a significant strength that is not found in combination elsewhere in the technology market.

Carbone *et al.* (2015) highlight how stream and batch processing can be achieved using a

single execution engine. They used Apache Flink[20] to show how a (seemingly diverse) set of use cases can be unified under a single execution model. Apache Flink is a framework and distributed processing engine for stateful computations over unbounded and bounded data streams. Flink has been designed to run in all common cluster environments, perform computations at in-memory speed and at any scale. Flink is built on the philosophy that many classes of data processing applications, including real-time analytics, continuous data pipelines, historic data processing (batch), and iterative algorithms (machine learning, graph analysis) can be expressed and executed as pipelined fault-tolerant dataflows. Batch computations are executed by the same runtime as streaming computations. The runtime executable may be parameterized with blocked data streams to break up large computations into isolated stages that are scheduled successively. The research found that "while batch computations are, in theory, a special case of a streaming computations, Flink treats them specially, by optimizing their execution using a query optimizer and by implementing blocking operators that gracefully spill to disk in the absence of memory" (Carbone *et al.*, 2015, p.37).

2.8 Summary

This chapter provided context as to the nature of this research. Section 2.1 added definitions of important aspects in this study. In Section 2.2 data was discussed as a foundation to data analysis. Section 2.3 discussed several aspects of data analysis; methods, approaches, processes and applications.

Section 2.5 mentioned a number of existing tools and techniques for data analysis with four specific examples provided. A high-level discussion of data analysis opportunities and challenges was provided in Section 2.6. The chapter closed with some highlights of related work carried out by different researchers.

[20] https://flink.apache.org/

Chapter 3

High Level Methodology and Research Design

This chapter gives a high level overview of the research and discusses the design of the SDA tool and the record and playback macro feature to satisfy the research objectives. The software and datasets used are also discussed as well as justification for these choices.

3.1 Research Methodology

Phase two of the research approach as documented in Section 1.4 relied heavily on the literature. Based on the tool analysis presented in Section 2.5, weaknesses and risks in the data analysis process were investigated to identify areas of potential enhancement such as, reducing the cost (both in acquisition of the analysis tool and gaining of knowledge to use such a tool), minimising user interaction where possible and making the tool scalable as shown in Table 3.1. In as much as information from data would be valuable, it is important for a cost benefit analysis to be carried out when making decisions of which objectivity is key (Wang *et al.*, 2018b). From the perspective of individuals and small businesses, every resource is important for growth, hence when Table 3.1 was populated the researcher learned from the important features of the tools such as efficiency, scalability, and varied data sources and improved on the areas specific to individuals and small businesses usability and the preferred open source nature. The categorisation of medium and high is a relative comparison between the tools based on what each tool offers.

For phases three (SDA tool development) and four (enhancement of SDA tool), a quasi-experimental approach was used for the iterative process of code implementation and

tool refinement, followed by data analysis experimentation to find patterns within the available datasets. This led to a prototype data analysis tool being implemented, and then refined to include additional functionality found to be lacking during the execution of case studies on different types of data.

Table 3.1: Data analysis tool comparison

	Microsoft Excel	**R**	**Python**	**Tableau**	**Microsoft Power BI**
Usability	Requires Training	Requires Training	Requires Training	Requires Training	Requires Training
Efficiency	Medium	Medium	High	High	High
Scalability	Medium	Medium	High	High	High
Data Sources	Medium	Medium	High	Medium	High
Development Environment	Desktop & Web	Desktop	Desktop & Web	Desktop	Desktop & Web
Data Preparation Tools	Yes	Yes	Yes	Yes	Yes
Open Source	No	Yes	Yes	No	No

Thereafter the prototype SDA tool was enhanced to include a record and playback macro feature that would allow recording of the data analysis steps used in a specific analysis process such that that these steps could later be rerun by using the recorded as input on a larger dataset. This implementation process is explained in detail in the sections that follow. Figure 3.1 illustrates the high level architecture of the system showing the SDA tool, containing the core elements of the SDA tool and the record and playback macro feature also referred to as the enhancement elements.

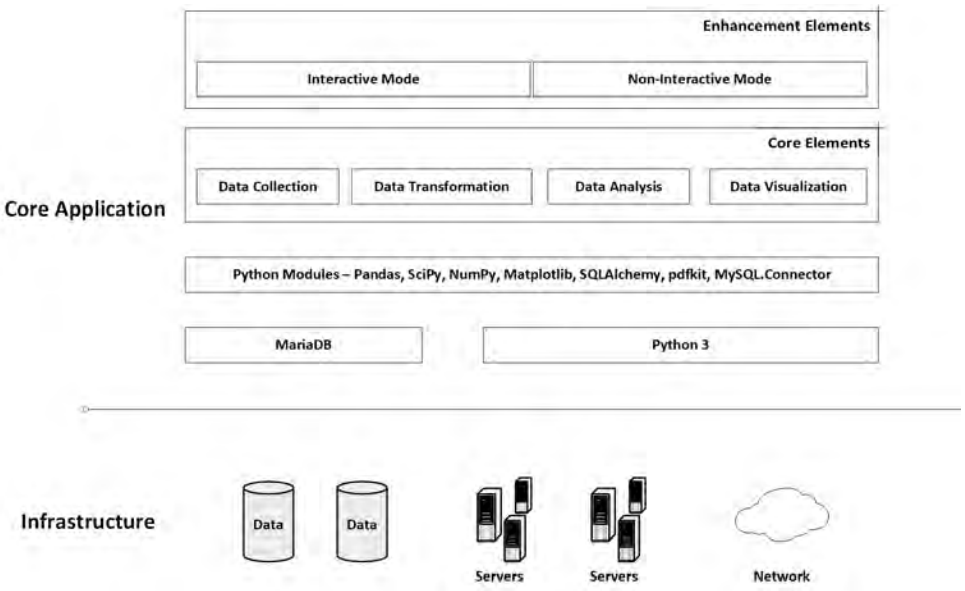


Figure 3.1: SDA tool architecture

Three case studies were identified. For the sake of validating the implementation, the first two case studies, namely a computer network-based data analysis process and a log-based data analysis process, were used to understand the data analysis process better so as to

improve the overall functionality of the SDA tool. The third case study was identified to assess the impact of the record and playback macro feature on the usability of the SDA tool. This case study was based on a general business area and used financial stock data. For each case study, two or three analysis scenarios were developed for testing the data analysis implementation. After testing, the implementation was tweaked to satisfy the scenario requirements and results were produced. Based on these results, example recommendations were made for each case study.

In documenting the case studies in Chapters 5-7, the structure given below has been followed in each case:

- Introduction – discusses the specific data, how it is typically analysed and why it is important.
- Overview – gives the structure and nature of the data used.
- Scenario planning and design – includes an adequate representation of the scenarios.
- Results – show outcomes for each scenario presented using various visualizations.

Three key performance indicators (KPIs) were identified to assess generality of the SDA tool and efficiency of the record and playback macro feature. These are:

- Usability – success rate (whether the user is able to perform a task), time taken to perform task and error rate.
- Scalability – the characteristic of the system to handle a growing amount of work by adding resources to the system.
- Parallelization – a metric used to show how much of the computer resources are being used in carrying out the analysis process.

3.2 Simplified Data Analysis Tool Design

The SDA tool was designed to have four main functional areas as dictated by the data analysis process given in Section 2.3.3. These are data collection, data transformation, data analysis and data visualization. Figure 3.2 shows a flow-chart of the processes included in the SDA tool, each of which is discussed in the following subsections.

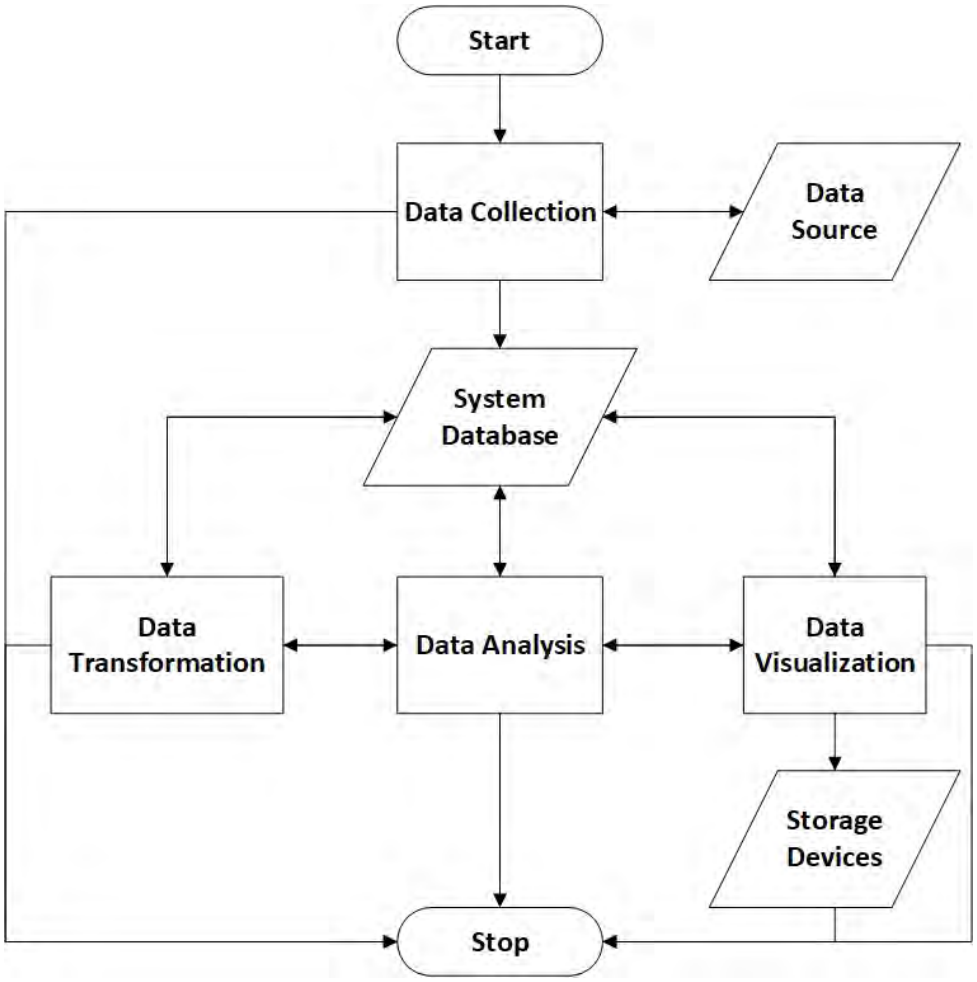


Figure 3.2: High-level system design of the SDA tool

3.2.1 Data Collection

This component is responsible for reading data from a source and storing it in the SDA system database for processing. It is divided into three functional areas: the input process, storage process and output process.

The input process obtains data source details and project details from a user. These details are used by the next process to access target data and copy the data to the system database. The input function requires the following information as input:

- Path: This can be a file path (local path, network path or Universal Resource Locator (URL) path) or database connection parameters such as DBMS type, localhost, username, password and database name(s), and optionally table name(s).

- Project details: In the context of this research, a project refers to a data analysis task to be done. Data being copied must be assigned to a new or existing project and the following project details must be provided: project name, database name (optional, default = project name) and table name(s).

Next, the storage process establishes a link to the data source using the path, creates a project using the project details provided and copies the data from the source into the system database. A user has the option to pre-process (filter, sort, clean, etc.) the data before it is saved into the database i.e. using the data transformation component that is discussed in the next section.

Finally, for the data collection function, the role of the output process is to provide feedback on the data retrieval process, summary statistics and the database schema for processing. The desired output for this function includes the data read from the data source, summary statistics (number of columns, number of rows, column data types, null value details (columns with null values), duplicate details and size of data (in Gb)).

3.2.2 Data Transformation

The data transformation component prepares the data for further processing and thus this component makes available several data preparatory processes to be applied depending on the context and data analysis requirements. This component was designed with three functionalities in mind: viewing of data, cleaning of data and maintenance of data. The primary input to this component is the data to be analysed. Two approaches are available for transforming data, some or all of the functions in this component can be carried out before or after data is written to the SDA tool's database (in the storage process during data collection). This allows for some quick and obvious corrections to be done prior to handling data. Efficiencies can be achieved by making good use of limited resources such as storage space and processing time.

View Data

A user has the option to view the actual data and details about it, for example, the data types, the columns names, summary statistics, and so on. Viewing of the data allows a user to make informed decisions on the cleaning process. Below is a list of the functions available:

- View the raw data.

- View summary statistics, which for all numeric columns the system prints out the count, mean, standard deviation, the minimum value, 25% percentile, 50% percentile, 75% percentile and the maximum value.
- Viewing the data structures, which shows the number of records, columns, data types of the columns, null count for each column and the memory usage of the data.
- Sorting data by a column or columns.
- Filtering data by a column or columns.
- Grouping data by a column or columns.
- Saving data, which allows the user to save temporary data at any point in most of the functions in the data transformation, data analysis and data presentation components.

Clean Data

Data is cleaned in order to make it accurate and complete (Power, 2016). To do so the following processes are available:

- Handle null values, which allows handling of null values by filling the null spaces, removing the record or leaving it as is.
- Handle duplicate values, which allows handling of duplicate values by checking whether the entire record is a duplicate or only certain attributes that should be unique so that an appropriate decision can be made to remove the record or amend part of the record.
- Handle inaccurate values, which allows for an assessment and maintenance of records for correction.
- Format data, which, by default, means the system formats the data depending on what exists in a column. However, due to errors some of the values can be inaccurate and prevent appropriate formatting. After cleaning such as working on missing or inaccurate values, this option allows a user to format the data as preferred.
- Merge data to allow concatenation of similarly structured datasets.
- Save the data.

Update Data

The third set of functions included in the data transformation component allows a user to update the data for one reason or another. For example, if after viewing the data a user notices an error that was not picked up in the functions above, the user has three options available, namely, update column names, update values (data in cells) and add or remove rows or columns.

3.2.3 Data Analysis

The data analysis component offers statistical analysis methods that can be applied to data to retrieve values and information. Data analysis makes available statistical measures for gaining information from data. The input is data specified in the system database and is processed using statistical measures chosen for analysis to produce information as output. This component is split into six functional groups as stated below:

- Managing data, which provides for three options, sampling, sorting and filtering.
- Calculating measures of central tendency, including mean, median and mode.
- Calculating measures of dispersion, including range, variance and standard deviation.
- Calculating measures of association, including correlation and regression.
- Other functions, such as, counting values, summary statistics, comparing columns and basic maths operations such as addition, subtraction, multiplication and division.

3.2.4 Data Visualization

The data visualization component provides methods for presenting the information obtained. Pattern discovery, reporting and general awareness of the state of data is thus presented visually by this component. The data visualization component is used to represent information and data in readable forms using various tools such as charts and tables. For this function, the input is data from the system database and visualisation options that are desired while the output is the actual visualisations. The types of visualizations available are listed below:

- Charts including scatter plot, line graph, bar graph, pie chart, horizontal bar graph, histogram, density estimation plot and table view (i.e. data is viewed in a table).
- Reports which provide the capability of customised reporting saved as either PDF or CSV files.

3.3 Record and Playback Macro Enhancement

The second part of the implementation of the SDA tool involved developing a record and playback macro feature aimed at enhancing the data analysis process by improving usability, scalability and resource utilization. With respect to usability, the record and playback macro feature should be able to make the data analysis process easier to use by amongst other things, minimizing user interaction. Some degree of automation was identified to enhance the generic application of the SDA tool by enabling a user to record the steps taken in any data analysis process, from start to finish and then execute the recorded steps using the playback feature to repeat data analysis that process on a larger dataset that would require more time and resources, thereby improving the scalability. Another use of the macro feature is to be able to run a data analysis process repeatedly on different attributes of the data, for example, using different time periods, this can be done by editing the recorded steps to change the time period involved in the analysis. The record and playback macro feature was designed to also utilize computer resources effectively through multiprocessing and interleaving of tasks discussed in Section 3.4 below, to improve scalability and allow for parallelisation.

The record and playback macro feature is similar to the Excel macros discussed in Section 2.5. However, there are several differences in that scaling of an analysis task using the SDA tool does not require any programming skills while the scaling capability of the SDA tool enables the tool to work on large and complex datasets effectively and efficiently using the power of Python such as parallelisation and data analysis libraries such as Pandas[1] which Excel does not offer currently.

The record and playback macro feature includes three main functions: a *controller* function, a *logger* function and a *reader* function, as shown in Figure 3.3. These functions are accessed by a user through one of two modes, namely interactive and non-interactive modes. Interactive mode is the default mode where the user interacts with the system step-by-step, using a menu driven approach. In this mode the user can choose to use the system in its most basic form i.e. using only the basic SDA tool, or alternatively to use the *logger* function, where a log file is created that records the steps with their input

[1] https://pandas.pydata.org/

taken to carry out data analysis (recording occurs as the user carries out a data analysis task).

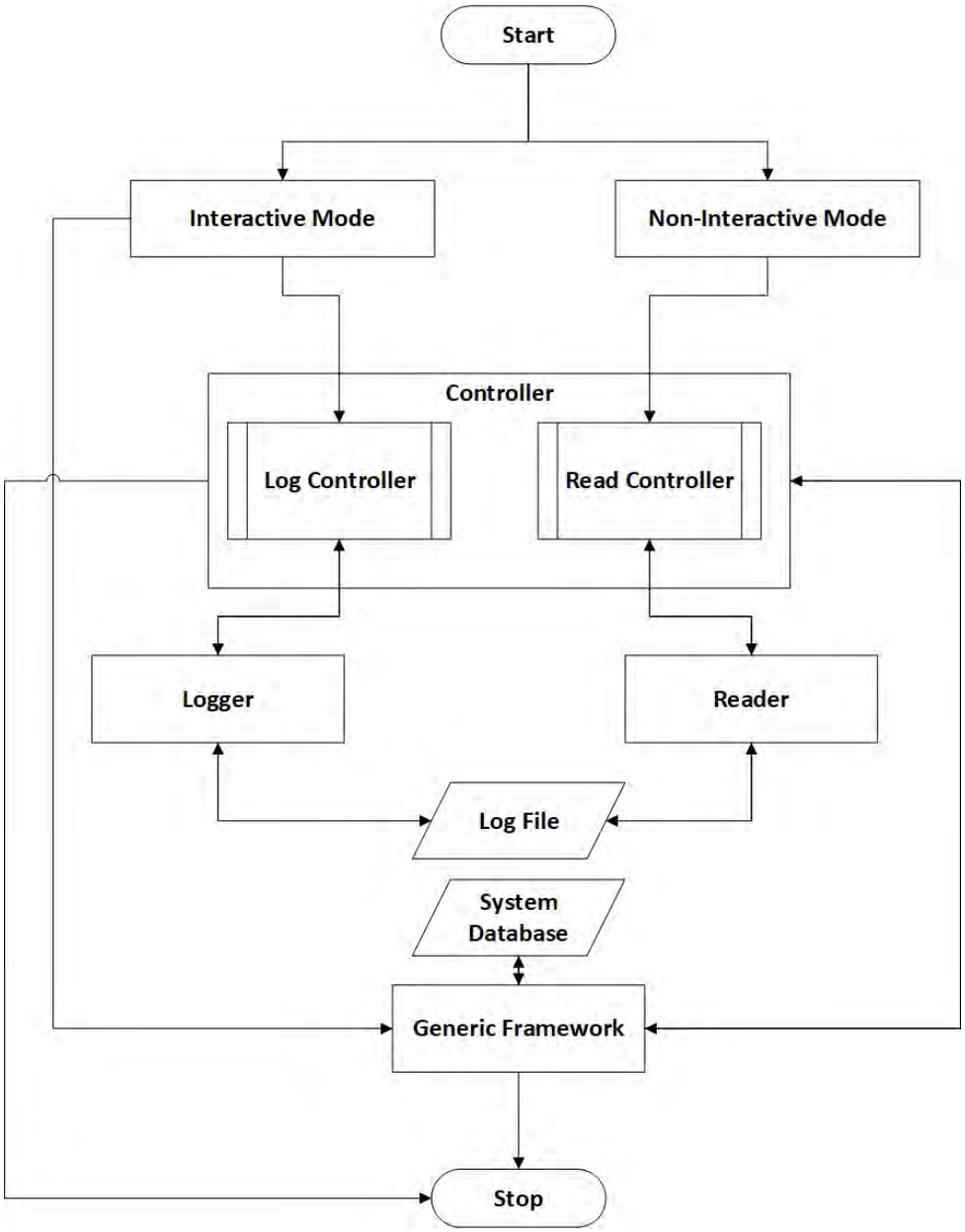


Figure 3.3: Control flow of the record and playback macro feature

On the other hand, the non-interactive mode is used to execute a data analysis task that has been recorded in a log file with user input, on a larger sample size or for a data analysis process that is run repeatedly. In this mode, the function uses multiprocessing wherever possible thereby allowing the user to continue working in interactive mode on other data analysis tasks. The non-interactive mode uses the *reader* function to execute data analysis steps with the associated input given in a specified log file. Thus, the ideal

approach would be for a user to create analysis tasks, record the steps while working on a sample dataset, review the recorded log file to refine the data analysis process and finally to use the SDA tool in non-interactive mode to execute the task on the full dataset.

3.3.1 Logger Function

One of the functions in the record and playback macro feature is the logger, which is responsible for the first of two main tasks in this feature, i.e. the recording.

This function records actions taken during a data analysis task and produces a log file as output. This function is only available in the interactive mode of the SDA tool. Logging is used to capture all user input and corresponding functions accessed; it carries out its processes in the background, recording user input into a log file for use later by the reader function. Each log record has the following format: date and time, user input, input format (i.e. the format the input is in) and function ID. Listing 3.1 shows a sample log file.

```
Date_and_Time, Millisecond, Input, Format, Function_ID
2020-10-11 08:48:20,622, 1, <class 'int'>, DC1
2020-10-11 08:48:28,340, 1, <class 'int'>, DC2
2020-10-11 08:48:31,551, data/sample_net.csv, <class 'str'>, DC3
2020-10-11 08:48:38,663, 1, <class 'int'>, TRF1
2020-10-11 08:48:40,112, 4, <class 'int'>, TRF3
2020-10-11 08:48:49,212, Flow-Rate, <class 'str'>, TRF4
2020-10-11 08:48:54,058, 3, <class 'int'>, DP1
2020-10-11 08:48:58,455, 2, <class 'int'>, DP2
2020-10-11 08:49:12,322, Time, <class 'str'>, DP2
2020-10-11 08:49:21,172, Flow-Rate, <class 'str'>, DP2
```

Listing 3.1: Sample log records

3.3.2 Reader Function

The second of the functions in the record and playback macro feature is the reader, this is responsible for the playback. This function plays back user input when carrying out a data analysis task in non-interactive mode.

The reader function reads a log file containing user input (compiled by the logger function) to execute analysis tasks on a specified dataset. This function is accessed in the non-interactive mode and the only input required from a user is the access path to the log

file. Each log record is executed using the input and function records and the processed data is then stored as various files and database tables as specified in the log file. This function uses a function map module designed to call the respective modules/functions in the SDA tool. The details of the implementation are given in the next chapter.

3.3.3 Controller Function

This is the main function that manages the logger and reader functions by acting as an interface between the basic SDA operations and the record and playback macro feature. Both the interactive and non-interactive modes call this function which manages both what is passed to the logger function to be logged and what is read from the reader function to be executed by the SDA tool. During the recording of a data analysis task, as the user captures input, the logger logs the input to a log file. Similarly, when using the non-interactive mode, as the log file is being read as input, each input is passed to the SDA tool at a time in sync with processes happening in the SDA tool through the controller.

3.4 Performance Enhancements

The final part of the implementation was to develop performance enhancements to the SDA tool, namely, multiprocessing and multithreading. The SDA tool should run in an efficient manner, taking advantage of available hardware resources by carrying out some processes in parallel wherever possible to enhance the data analysis process further.

Multiprocessing uses at least two central processing units (CPUs) to run at least two processes simultaneously, if possible. Multiprocessing initiates multiple operating system processes for each separate process/task to allow these to run in parallel by separate processor cores (Khot, 2017). For the SDA tool, if needed, multiprocessing allows one process to run in interactive mode and a number of processes in non-interactive mode (subject to hardware resources availability) because the processes would be independent (i.e. the data analysis tasks would be separate).

With multithreading, multiple processing cores work on shared input data, with threads working on different parts of a process with input/output (IO) bound processes (Khot, 2017). Threading was designed into the system, to create the effect of interleaving of tasks as is the case during data collection: when data is copied from a data source it is stored in computer memory and later copied to the system database. As the data is being copied to the database, data analysis operations can be carried out on the data in memory;

for example, after the system copies data from source, the user will be asked to provide project, database and table names, after which the main menu will be displayed in order to process the data further using transformation, analysis and presentation components of the SDA tool. When the copying of data to the database is completed, the user will be notified and can access that data at any time. Meanwhile a user will always process the data in memory, an option to save it to the database will be available (saving of data is designed to use multithreading) from the menus of all three components in order to maintain data integrity and to make it easy to roll-back if mistakes have been made. Processed data for a project is saved using the same database provided earlier in the process; note that the user should specify different table names for different completed analysis tasks that are saved into the database.

3.5 Design Considerations

The design of the system included various considerations to improve the SDA tool's usability, as discussed below:

- Menu-driven approach; this approach provides for applicable options at each stage of the data analysis process while allowing for a dynamic execution (i.e. data analysis tasks can be carried out based on context) of an analysis task. For example, when collecting data from source, only the options for that functionality are displayed, making it easier for non-technical users or those with little analytical know-how to carry out data analysis tasks.
- Data pre-processing; this feature is available when copying data from a source to the system. In some cases not all the data being copied is what is required and so this function allows for extraction of only the required data through filtering. This function also provides options to clean, format and update the data before it is saved to system database. In cases where this feature is applicable, it allows for faster processing as the data is reduced through filtering and cleaning of the data.
- In-memory data processing; as stated earlier, during the data analysis process, when the system is processing data, the data is stored in memory for quicker overall processing due to the faster speed of computer memory than storage such as hard disk drives.

A combination of the above features should make the SDA tool easier to use by minimizing the initial learning and providing for an environment with acceptable performance.

3.6 Implementation Considerations

Tool selection was based on several desired characteristics as determined by the researcher, such as, ease of use and learning, open-source nature and functional capabilities available in the tools for data processing. The system was implemented using the Python programming language with a MySQL database as the internal system database.

Python was chosen because it offered flexibility and scalability on data handling especially with the power of the Pandas library. Moreover, the open-source nature of Python was preferred as opposed to commercial tools because Pandas is adequate for most small business analysis requirements and is free to use.

"Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language" (Pandas Development Team, 2020). The Pandas library was used to implement all data processes, i.e. it provides an efficient dataframe for handling data, large or small. Other Python libraries such as NumPy and SciPy were used for data analysis functions, while Matplotlib was used for data presentation. SQLAlchemy, another Python library, was used for handling interaction between the system and the Database Management System, MySQL (using MariaDB[2] 10.4.6). SQLAlchemy interfaced with Python using MySQL.Connector[3].

MariaDB is an enhanced, drop-in replacement for MySQL developed by the original developers of MySQL and guaranteed to stay open-source. It is fast, scalable and robust, with a rich ecosystem of storage engines, plug-ins and many other tools that make it very versatile for a wide variety of use cases. It was developed as open-source software and as a relational database it provides an SQL interface for accessing data. The SDA tool utilizes a database system for storing data whilst it is being processed. For data collection from sources, the SDA tool is able to interact with four types of Database Management Systems, namely, MySQL, Microsoft SQL, Oracle and PostgreSQL. The system can provide output (i.e. processed data) to either a database or CSV or PDF files (PDF files are created using pdfkit[4]).

As execution infrastructure, the system requires an environment with the minimum specifications as given in the Table 3.2.

[2]https://mariadb.org/
[3]https://pypi.org/project/mysql-connector-python/
[4]https://pdfkit.org/

Table 3.2: Minimum system specifications

Operating System	Windows/Linux/MacOS with Python3
Python Modules	Pandas, SQLAlchemy, MySQL.Connector, Matplotlib, Scipy, Numpy, pdfkit
Database Management System	MySQL
Storage	1 GB of free space on hard disk drive (otherwise dependant on size of data analysed)
Main Memory - RAM	8 GB
Processor	Duo Core with 1.9 GHz

3.7 Experimental Datasets

Three datasets were identified and used in the case studies for this research.

For the first case study using network data, the network data used was labelled "IP Network Traffic Flows Labeled with 75 Apps" and owned by Rojas (2019). The dataset was collected in a network section from the Universidad Del Cauca, Popayán, Colombia by performing packet captures at different hours, during morning and afternoon, over six days in 2017. A total of 3,577,296 instances were collected and are currently stored in a CSV file. This network traffic flow dataset was chosen for this study because it provided records for a real world network data analysis environment, it is free to use and generally provides simple and complete data. This dataset was used to validate the SDA tool.

For the second case study using log data, the data file was a server *weblog* collected from RUET OJ [5], an open source online judge platform in Bangladesh. The data was collected between November 2017 and March 2018, and consists of 16008 rows with 4 columns of attributes, namely IP, Time, URL and Response Status. An access log dataset was chosen because this provided a different type of data for a real world data analysis environment. It is also free to use and generally provided simple and complete data. Similar to the first dataset, this was used to validate the SDA tool.

Finally, for the third case study a business example to test the prototype SDA tool and the record and playback macro feature used Apple Inc. stock data. Apple Inc. stock data for a 5-year period between 1st January 2016 to 31st December 2020 was retrieved from Yahoo Finance[6]. The data contains seven columns, *date, open, high, low, close* and *volume*. *Open, high, low* and *close* prices are stated in United States Dollars (USD).

[5] http://ruetoj-ruetoj.apps.us-east-1.starter.openshift-online.com/
[6] https://finance.yahoo.com/quote/AAPL/history?p=AAPL

3.8 Summary

The chapter provided a high-level overview of the design of the SDA tool, Section 3.1 discussed the methodology used and the architecture of the tool. This section also explained how the case studies were used together with the KPIs to validate the SDA tool. Section 3.2 outlined and discussed the four components of the tool, namely data collection, data transformation, data analysis and data visualization.

Section 3.3 introduced the record and playback macro feature. Some performance enhancements were then discussed in Section 3.4. Section 3.5 stated the design considerations that were implemented to make the SDA tool more usable, while Section 3.6 discussed the tools that were used in the SDA tool implementation. Finally, the datasets used in all three case studies were introduced.

Chapter 4

Prototype SDA Tool Implementation

This chapter complements the system design covered in the previous chapter by discussing various technical aspects of the SDA tool. Some of the functionality is illustrated by algorithms and code snippets. This chapter covers implementation of both the core elements/functions explained in Section 3.2 and the record and playback macro feature introduced in Section 3.3.

4.1 Implementing the Core Elements

The core elements are the functions within the SDA tool responsible for processing the data. There are four core modules, namely, data collection, data transformation, data analysis and data visualisation, which are discussed further in the subsections below.

4.1.1 Data Collection Module

This module copies data from a source (file, database, etc.) thereby making it available to the system for processing. The copied data is stored in a system database for future use, while a copy of the data resides in memory for processing. Three functions exist in this module, namely, the copy function to retrieve data from the source, write function to write data to the system storage and a data collection controller function to ensure efficiency when reading and writing the data. This module is illustrated in Figure 4.1 which expands on the data collection module shown in Figure 3.2.

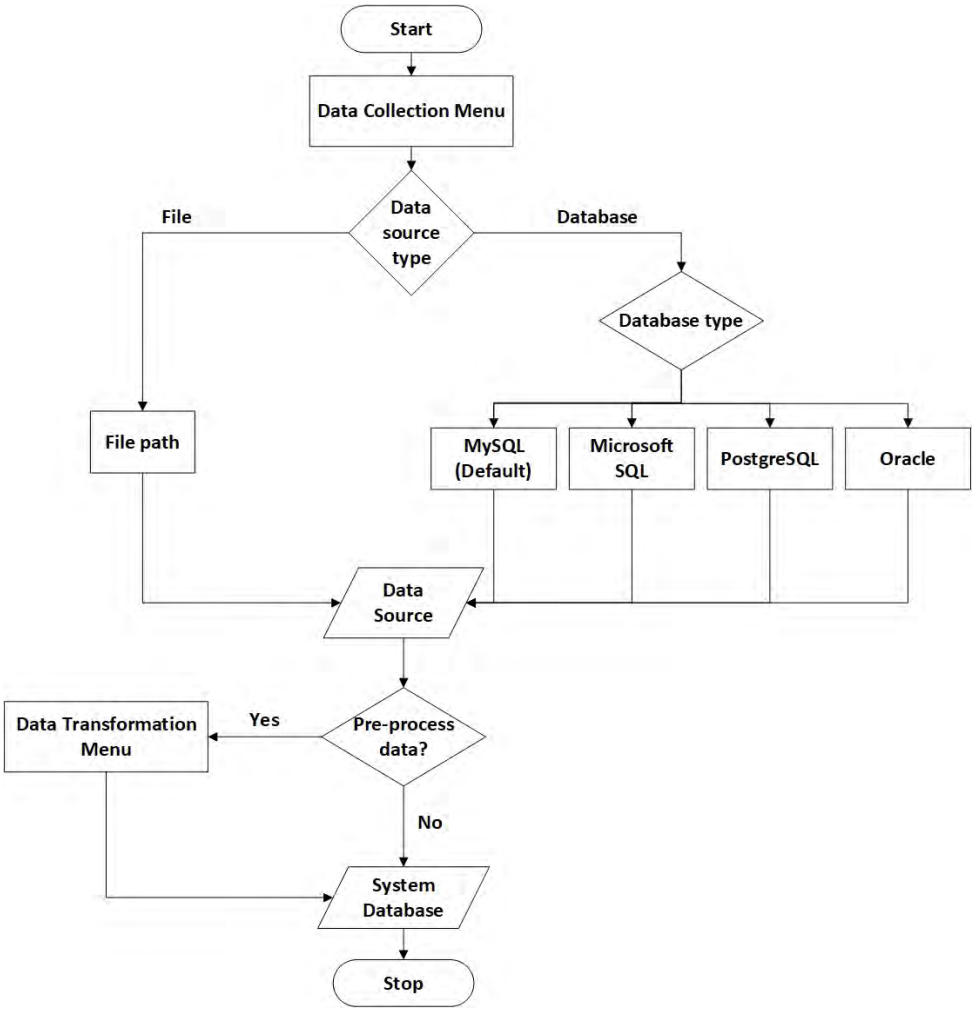


Figure 4.1: Data collection process flowchart

The starting point for data processing is when data is retrieved from where it was stored. For this research we considered only two data sources, namely text files and databases. The SDA tool provides the option to *read data* within the copy function, with the two source options available and displayed for a user to choose from, that is read from a file or a database. With the option to read from a file, text file types such as CSV, JSON, MS Excel, HTML, Statistical Package for the Social Sciences (SPSS) and Stata are supported. With the option to read from a database, support is provided for reading from databases using DBMSs such as MySQL, SQL, Oracle and PostgreSQL.

The data collection controller function is responsible for any interaction with the data collection module; this function uses the copy function to copy data and the write function to write data. Also available in this function is the ability to capture details of the analysis process which in this research we refer to as a project. Three key items are captured, namely, the project name, database name and table names, which are used to identify the

projects easily. Another facet of this function is to enable multitasking through multi-threading. Essentially this allows for copying and writing of data while the main task of data processing continues to other functions or modules.

Algorithm 1: Data collection - Controller

input : Data Source Details, Project Details
output: Path

begin

Ask User - Type of Data Source (file or database or data scraping);
source_type $\longleftarrow$ *user_input*;
if *source_type = 'file'* **then**
    path $\leftarrow$ [*source_type*, *file_path*];
    `/* Gets` *file_path* `from user */`
else if *source_type = 'database'* **then**
    Ask User - Type of DBMS (MySQL or SQL or PostgreSQL or Oracle);
    dbms_type $\longleftarrow$ *user_input*;
    if *dbms_type = 'sql'* **then**
        dialect $\leftarrow'$ *mysql* + *pymysql*$'$;
    else if *dbms_type = 'postgresql'* **then**
        dialect $\leftarrow'$ *postgresql* + *psycopg2*$'$;
    else if *dbms_type = 'oracle'* **then**
        dialect $\leftarrow'$ *oracle* + *cx_oracle*$'$;
    else
        dialect $\leftarrow'$ *mysql* + *pymysql*$'$;
    end
    Ask User - DBMS Parameters (Username, Password, Host, Port, Database, Table);
    dbms_parameters $\longleftarrow$ *user_input*[$'$*username*$'$,$'$ *password*$'$,$'$ *host*$'$,$'$ *port*$'$,$'$ *database*$'$,$'$ *tablename*$'$];
    path $\leftarrow$ [*source_type*, *dialect*, *dbms_parameters*];
Ask User - Project Details (Project Name, Database Name, Table Names);
`/* Default` *db_name* = *project_name* `if user does not specify */`
project_details $\longleftarrow$ *user_input*;
return *path*, *project_details*

end

Once data has been copied and needs to be stored, the write function is used to write the data either to a database or a file. With the option to write to a database, the same types of DBMSs used in the copy function are available and with the option to write to a file, a CSV or PDF formatted file can be created.

Algorithm 2: Data collection - Copy data

input : Path
output: Data stored in memory

begin

```
1  | Get path from Algorithm 1;
2  | source_type ⟵ path[0];
3  | if source.type = 'file' then
4  |     file_path ← path[1];
5  |     data ← pandas.read_file(file_path);
6  | else if source.type = 'database' then
7  |     dbms_dialect ⟵ path[1];
8  |     dbms_parameters ⟵ path[2];
9  |     db_username ⟵ dbms_parameters[0];
10 |     db_password ⟵ dbms_parameters[1];
11 |     db_host ⟵ dbms_parameters[2];
12 |     db_port ⟵ dbms_parameters[3];
13 |     db_database ⟵ dbms_parameters[4];
14 |     db_tablename ⟵ dbms_parameters[5];
15 |     eninge_parameters ⟵ dbms_dialect : //db_username :
   |         db_password@db_host : db_port/db_database /* typical form of a
   |         database URL with SQLAlchemy=dialect + driver : //username :
   |         password@host : port/database                              */
16 |     db_tablename ⟵ sqlalchemy.create_engine(eninge_parameters);
17 |     data ← pandas.read_sql_table(db_database, db_tablename);
18 | return data
```

end

Algorithm 1 illustrates how the process of data collection starts using the data collection controller function, with data source type and connection parameters as input from the user. Once the connection parameters are input into the SDA tool, a thread is created to retrieve the data using the copy function (*Algorithm 2*) while the project details are input to the SDA tool. The project details are used to create a new project, database and tables or access an existing database. In the latter case, new tables are created to store the new data in the SDA tool's DBMS. Once data is copied it resides in memory as a Python Pandas DataFrame. Two options are then available: either the data can be transformed and then saved or the data can be saved as is and in this case another thread is created to write the data to the project database using the write function as shown in Algorithm *3*. The main thread can then continue with the data processing.

Algorithm 3: Data collection - Write data

input : Data
output: Data stored on the system

begin

```
1   Get project_details from Algorithm 1;
2   project_name ⟵ project_details[0];
3   db_name ⟵ project_details[1];
4   table_name ⟵ project_details[2];
5   db_status ⟵ check_existing_db(db_name);
    /* Checks if database exists                                    */
6   if db_status = TRUE then
7       table_status ⟵ check_existing_table(table_name);
        /* Checks if table exists                                   */
8       if table_status=TRUE then
9           Ask User - New Table Name;
10          table_name ⟵ user_input;
11  else
12      sqlalchemy.create_db(db_name) ;
    end
13  Get data from Algorithm 2;
14  data ⟵ algorithim2 ;
15  eninge_parameters ⟵ dbms_dialect : //db_username : db_password@db_host :
        db_port/db_database /* uses default system parameters       */
16  engine ⟵ sqlalchemy.create_engine(eninge_parameters);
17  pandas.DataFrame.to_sql(data, table_name, engine);
    /* Writes Data to Database                                      */
18  return summary_stats
```

end

Algorithm 3 illustrates the creation of a database and storing of data using the write function. Other modules in the SDA tool can use the data collection module to read and write data, but every time the copy and write functions are called a thread is created while the main thread continues with the other core element functions such as data transformation, data analysis and data visualizations. In the other three modules comprising the core elements, the copy function is used in the background to access data in the SDA tool's database if no data is loaded into memory for use by those modules. The *Save Data* option is available in all modules to allow users to save data that has been altered.

4.1.2 Data Transformation Module

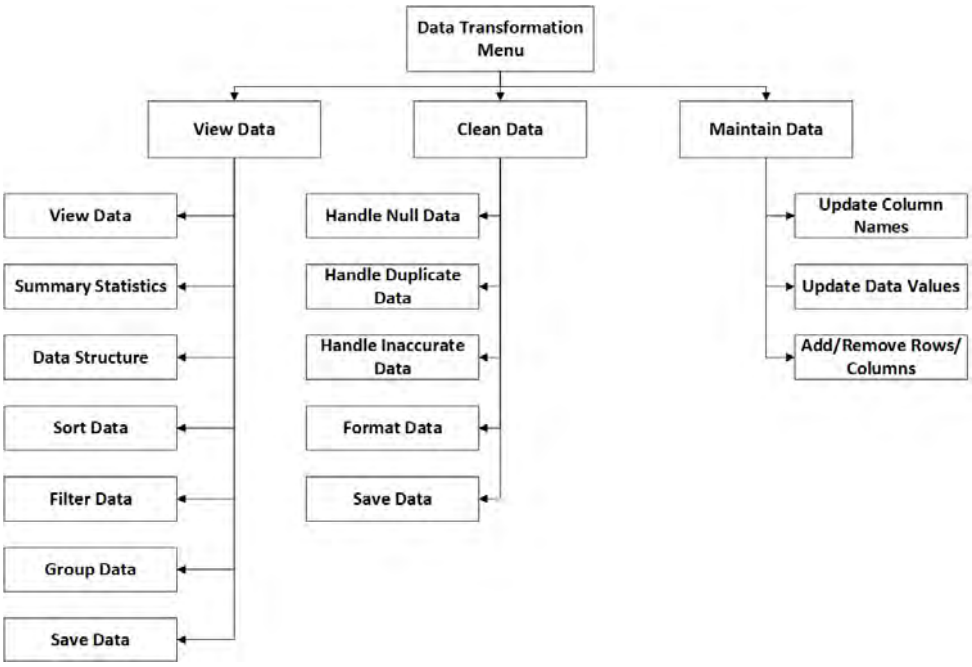


Figure 4.2: Data transformation main menu

The data transformation module offers several capabilities and options that can be applied to data depending on the context and data analysis requirements. The input to this module is the data to be processed and various user options available to the user. Transformations prepare the data through cleaning and updating of data for further processing, i.e. transformations are aimed at preparing data for analysis by ensuring that the data is accurate and complete (Power, 2016). This module has three functions: view data, clean data and maintain/update data as illustrated in Figure 4.2.

The data transformation module prepares the data for further processing and thus this module makes available several data preparation processes to be applied depending on the context and data analysis requirements. The primary input to this module is the data to be analysed. Two options are available for transforming data, namely, that some or all of the functions in this module can be carried out before or after data is written to the SDA tool's database. This allows for some quick and obvious corrections to be done prior to handling data. Efficiencies can be achieved by making good use of limited resources such as storage space and processing time.

4.1.3 Data Analysis Module

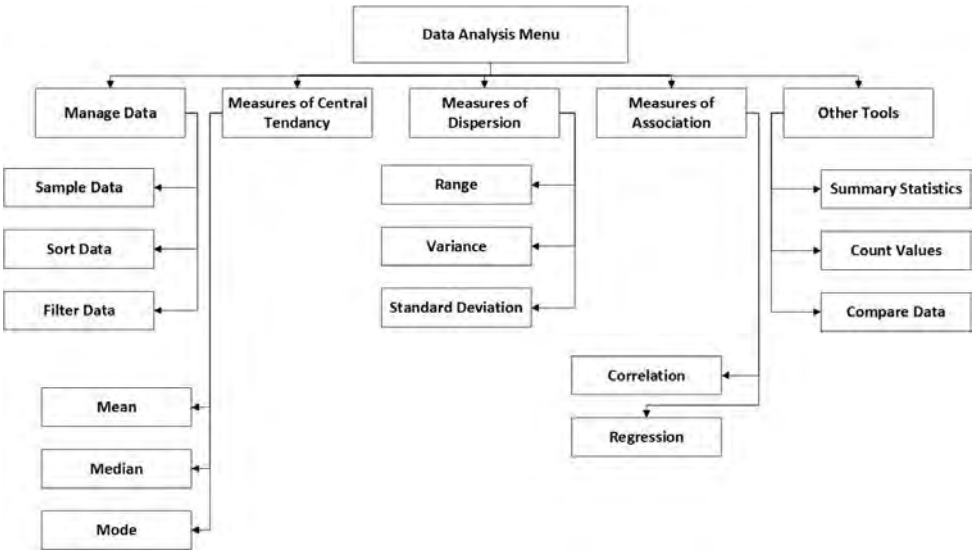


Figure 4.3: Data analysis main menu

The data analysis module offers statistical analysis methods that can be applied to data to retrieve values and information. Data analysis makes available statistical measures for gaining information from data. The input is data specified in the system database and is processed using statistical measures chosen for analysis to produce information as output. This module is split into five functional groups as shown in Figure 4.3; namely, managing data, measures of central tendency, measures of dispersion, measures of association and other tools that provide additional functionality.

4.1.4 Data Visualisation Module

The data visualisation module provides visualisations, i.e. methods for presenting information that has been deduced from the data. Pattern discovery, reporting and general awareness of the state of data is thus presented visually using this module. For this module, the input is data from the system database and visualisation options that are desired while the output is one or more visualisations options provided.

This function provides two menu options for visualization, namely, charts and reports. The primary input is data to be visualized while additional input is specified by the user depending on the option chosen. For example, if one desires to print out the information in table form then only the input of choosing either PDF or CSV format and the name of the file is requested by the system. For the chart option, the user has to choose the type of chart and then provide the name/title, labels and figure size (for quality of the figure).

4.2 Automation Enhancement

The core system functions do provide for an adequate environment to conduct data analysis, however, one of the objectives of this study is to develop enhancements to those core functions. Automation is one of the aspects that was considered to enhance the SDA tool by improving usability, scalability and resource utilization. This study developed an automation enhancement in terms of the record and playback macro feature. Together the logger and playback functions record the steps and capture the input when a user is carrying out a simple data analysis task and at a later stage can rerun the recorded steps on a larger dataset without user interaction.

The configuration of the enhancements and the relationship with the SDA tool shown in Figure 3.3, is further discussed in more technical detail in this section.

4.2.1 Controller Function

```
def add_it(x,y):
    sum_val = x + y
    return sum_val

def minus_it(x,y,reverse=False):
    if reverse is True:
        minus_val = y - x
    else:
        minus_val = x - y
    return minus_val

AddCode = 'ADD1'
SubCode = 'SUBTRACT1'

function_map = {'ADD1':add_it, 'SUBTRACT1':minus_it}
function_to_run = function_map[AddCode]

function_to_run(12,11)
```

Listing 4.1: An example of dynamic function calling in Python

The controller function provides an interface between the two modes (interactive and non-interactive) of the SDA tool and the logger and reader. In the interactive mode (the default mode), a user can enable recording of a data analysis process through the

controller which then coordinates the creation of a log file and logging of every input from the user. In the non-interactive mode, the controller coordinates the access of a log file specified by a user and then the execution of a data analysis as specified in the log file.

Since it is not possible in Python to call a function directly using another variable, a method known as dynamic function calling was implemented. Using what we have termed a function map, we use a Python dictionary to map functions and their identification numbers as illustrated in Listing 4.1.

When the logger logs a record, a *function-id* is recorded, which is the function's identification corresponding to a particular function in the map. When the reader reads a record from a log file, the *function-id* is used to identify the function to execute as the next process.

4.2.2 Logger Function

The logger function is accessed through the interactive mode of the SDA tool, which interfaces with a controller function to log user input passed to the SDA tool. There are three components that are initiated and created by the controller when a user enables recording in the interactive mode. The first is a global variable *record* that is checked if *TRUE* by the modules when they are run in order to send input records to a logger for recording into a log file. A logger function is then initiated to log user input to a log file using the Python *logging* module and, finally, a log file is created for storing the steps. The logger has the following three processes also illustrated in *Algorithm 4*:

1. Creating a log file for a specified project. The log file is created with a header, *date-time, input, input-format, function-id*. The *date-time* attribute is used to sort the list from oldest to newest, the *input* is input passed by the user, *input-format* is the format of the input and *function-id* is a number identifying a particular function in the SDA tool.
2. Logging every input from the user that is captured in the SDA tool.
3. Enabling review of the log file once logging is done.

Algorithm 4: The Logger Function

input : User Input on mode to use
output: Log and Log Records

begin

Get - System Mode (Interactive Mode or Non-interactive Mode);
system_type $\longleftarrow$ *user_input*;
if *system_type = 'interactive_mode'* **then**
    Ask User - With or Without Recording? (Recording or No Recording);
    record_type $\longleftarrow$ *user_input*;
    if *record_type = 'TRUE'* **then**
        `/* TRUE if recording else FALSE */`
        Get project_name from Algorithm 1;
        Initiate logger using project_name;
        Initiate header $\longleftarrow'$ $Date\_and\_Time, Millisecond, Input, Format, Function\_ID'$;
        Go to data_collection module;
        For every user_input record $'Date\_and\_Time, Millisecond, Input, Format, Function\_ID'$ *in log file created using Logger function*;
    else if *record_type = 'FALSE'* **then**
        Go to data_collection module;

end

4.2.3 Playback Function

The playback function reads the log file, passes one input at a time to the controller for execution by the SDA tool, i.e. the log file is used as input to execute data analysis tasks in the background. The whole process is synchronized.The reader has two main processes also illustrated in *Algorithm 5*.

1. In non-interactive mode, a user specifies a path for the log file to be executed. The path is passed to the controller which reads the contents of the log file and loads it into a list which is passed to the reader. The list is sorted with the oldest record first and after a record is read, it is removed from the list, i.e. first-in first-out type of list.

2. The next step is to get the first record from the list. Three items are retrieved from the list, namely the *input*, *input-format* and *function-id*. For the first record only,

the function corresponding to the specified function-id is called dynamically using the SDA tool's function map dictionary. The input is explicitly formatted using the input-format attribute and passed to the function.

3. Due to the menu-driven nature of the SDA tool, once the process starts, the system continually requests input. Thus the following will become an iterative process:

   - Get input request from the SDA tool.
   - Get the next record from the reader.
   - Check function-id from the record with the function-id from the function requesting input; if true, proceed; if not true, raise an error.
   - Format input with input-format.
   - Pass formatted input to the SDA tool.

Algorithm 5: The Playback Feature

input : User Input on mode to use
output: Log and Log Records

```
begin
    Get - System Mode (Interactive Mode or Non-interactive Mode);
    system_type ⟵ user_input;
    if system_type = 'non-interactive_mode' then
        if input_list is empty then
            Ask User - Path for log file;
            log_file_path ⟵ user_input;
            Initiate reader using log_file_path;
            Initiate input_list ⟵ log_file;
        for input_request = 'TRUE' do
            input_record ⟵ input_list[0];
            input_value ⟵ input_record[2];
            input_format ⟵ input_list[3];
            function_id ⟵ input_list[4];
            if input_format == 'int' then
                input_value ⟵ int(input_value);
            else
                input_value ⟵ str(input_value);
            Pop-out input_list[0];
            input_values ⟵ [function_id, input_value];
            return input_values
        end
end
```

4.3 Summary

This chapter provided some low-level detail of the implementation of the basic SDA tool (Section 4.1) and the automation enhancement (Section 4.2). Algorithms and some code snippets were used to illustrate the textual explanations.

Chapter 5

A Case Study using Network Data

This chapter discusses the first case study for this research, using network data to refine and validate the SDA tool. The first two sections discuss network data in general, and thereafter the focus is on aspects that apply to network data analysis. The third section describes the design and implementation of scenarios for validating the SDA tool.

5.1 Analysing Network Data

The backbone to technological communications is a network, specifically a computer network. For various communication devices such as phones and personal computers, networks are essential for transmitting messages to the intended recipients of the messages. Networks interface various devices, by providing a medium or channels through which messages travel between source or sender to destination or recipient over short or long physical distances.

Computer networks are varied in structure and configuration; they exist in different places for different purposes serving different needs or requirements and handle a varied range of data types and formats (Kurose and Ross, 2010). For effective communication there is a need for a robust network configuration that meets appropriate security standards. As networks continually evolve, they are becoming more and more complex, bringing about challenges in running them effectively (Alotaibi *et al.*, 2017).

A computer network is a collection and integration of a number of computing devices for interaction (sharing resources) (Kurose and Ross, 2010; Alotaibi *et al.*, 2017). Best management practices recommend that there are adequate levels of network security throughout data transmission with specific goals being confidentiality, integrity and availability

(Alotaibi *et al.*, 2017). Achieving this security requires several capabilities working together. One option is to use data analysis to enhance network availability by promoting the awareness about the network state through monitoring, managing and controlling of various performance parameters such as speed (rate at which data passes through the network), types of connections allowed to pass through and users allowed to access the network, etc. (Lin and Liao, 2017). Thus, it is important for computer networks to be monitored effectively using appropriate tools as the amount of data that is generated from various network applications is huge and is generated at an ever increasing rate (Hoplaros *et al.*, 2014).

The evolution of computer networks has greatly increased the data being transmitted and demand on networking resources which has led to more scaled environments that require efficient information delivery. This has caused a change in approach to the way networks are managed from a network-centric approach focused on network performance to a more modern user-centric approach focused on user experience (Kang *et al.*, 2012). Automation of the data analysis process allows for efficacy through availability of networking information by monitoring network traffic performance data and carrying out analysis using software (Oliver-Balsalobre *et al.*, 2017). A number of software tools for data analysis are available, such as, Wireshark, Nagios and NetSpot. Network software for data analysis enables data gathering from the logs generated by various network devices on aspects such as resource utilization. These logs can be analysed and reports produced on the state of a network environment, thus affording comprehensive information for decision making in network management. Performance and monitoring parameters can be customized based on a set of requirements of an environment in which a solution is to be deployed. Although the structure of the data depends on the network setup, the network data attributes are standard because of the use of a standard network model, the Transmission Control Protocol /Internet Protocol (TCP/IP), which is discussed in the next section.

Many studies have focused on network monitoring with various components being looked into, such as, to provide an accurate and current overview of the state of an environment, to provide additional visualization, to provide alerts when any state changes, to be easily maintainable as an environment scales, to examine the broader IT infrastructure, and so on (Solomon *et al.*, 2016; Rudman, 2016; Queiroz *et al.*, 2019; Fay *et al.*, 2017). The dynamic nature of the field drives the need for new methods and approaches which the evolving data analysis process offers. This case study provides the opportunity to obtain information from network data through data analysis by using the SDA tool developed and in the process to validate the functionality of the tool.

5.2 Overview of Network Data

Computer networks are complex, they grow from local area networks to wide area networks and to the Internet and comprise a number of network devices that use various types of mediums. Organizations and entities administer their own networks normally using private Internet Protocol (IP) addresses which usually interface to other networks on the public domain (the Internet) using public IP addresses. For general communication to occur there is a universal structure of how communication is governed. The TCP/IP model, which is a suite of communication protocols that carry out essential services running on the Internet and private networks, is normally used (Alotaibi *et al.*, 2017). This network model provides end-to-end connectivity of information between sender and receiver by establishing, maintaining and ending connections. Figure 5.1 illustrates the TCP/IP network model which is divided into four layers: layer 1 is the application layer, layer 2 the transport layer, layer 3 the Internet layer and layer 4 the network layer as discussed below.

- The application layer is the topmost layer and, is responsible for human interaction and how software implements data to be transmitted or received. This layer deals with protocols such as Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS), Simple Network Management Protocol (SNMP), File Transfer Protocol (FTP), and so on.
- The transport layer in the TCP/IP network model is responsible for transmission of data between two hosts, that is end-to-end connection. This layer has two protocols, TCP and User Data Protocol (UDP).
- The Internet layer is responsible for handling communication from one computer to another, being able to identify the computer intended to receive a packet. This layer has four main protocols, IP, Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP) and Internet Group Management Protocol (IGMP).
- The network access layer is responsible for host-to-host communication on the physical level.

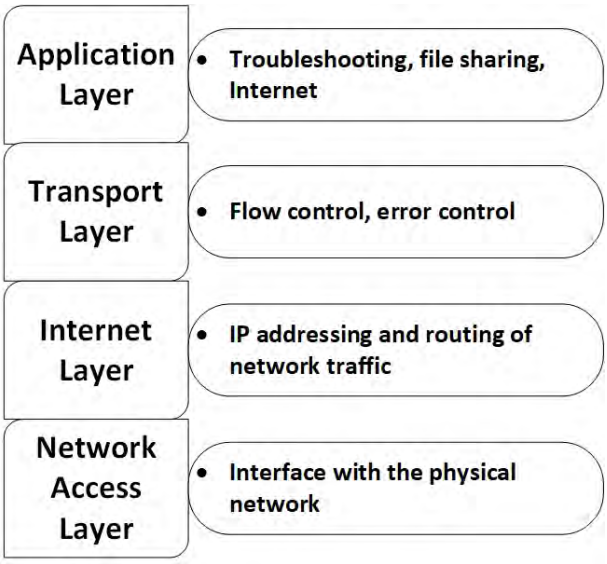


Figure 5.1: TCP/IP Network Model, adapted from Zhou *et al.* (2018)

For networks to function, network devices are used. These are hardware components that enable communication (transmission of data) over a network by interfacing links (through a medium) on a network. Network devices work with other network devices and host computers through a physical form of connection. Three types of physical connections exist, namely wired, wireless and optical technologies. The data generated by network devices may vary slightly because of differing configurations used and these devices work in layers 2, 3 or 4. There are various network devices, however, four are most commonly used:

- Routers connect networks, for example connecting a company network to the Internet.
- Switches connect host machines such as computers, printers, servers, mobile devices etc. to a private network such as a local area network (LAN).
- Firewalls control network traffic based on a set of rules to protect a network.
- Servers manage various network resources and administer the network by handling user requests on a network.

5.2.1 Network Management

Network management consists of a broad range of functions including activities, methods, procedures and the use of tools to administrate, operate, and reliably maintain

computer network systems (Alpern and Shimonski, 2010). Network management deals with reliability, efficiency and capacity of data transfer channels about a computer network. Monitoring is essential to IT infrastructure and systems health and ultimately to an organization's bottom line as information is analysed to make informed decisions in management (Hernantes *et al.*, 2015). Monitoring of networks is done to detect issues, ensure service and infrastructure availability and measure resource utilization while ensuring awareness of the state of a diverse set of components (Formoso *et al.*, 2009; Hernantes *et al.*, 2015). Through network monitoring, the data that is generated in a network and the data coming in and going out can be understood. Information derived from network data can shed more light on network quality measurement, traffic estimation and attack prevention (Zhou *et al.*, 2018). Moreover, using data analysis tools trends and patterns can be discovered to improve the overall operation of a computer network and identify problems, weaknesses and potential incidents, that can help to improve service delivery by making well informed decisions (Qiu *et al.*, 2010; Hoelscher and Mortimer, 2018).

5.2.2 Network Data Composition

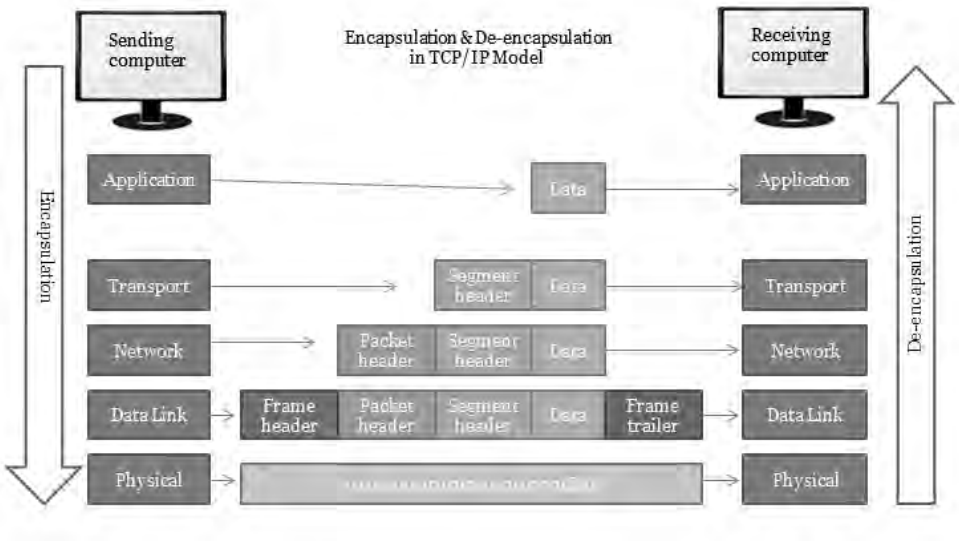
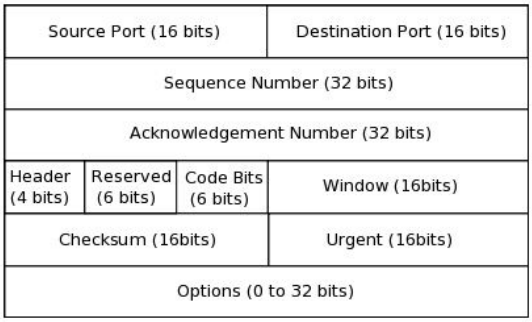


Figure 5.2: Encapsulation and de-encapsulation in TCP/IP Model, adapted from Zhou *et al.* (2018)

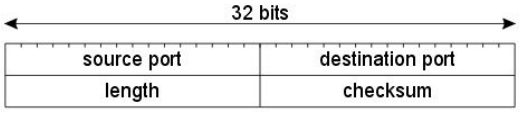
Two components are used for network monitoring, namely the data and the tools for analysing the data. Before data can be explored, it is essential to understand the environment in which it was generated. The basic unit of network data is a datagram (payload), which is the data that is to be sent over a network. Using the TCP/IP network model the data moves from one host to another, whilst undergoing a process called data encapsulation where transmission information (header) is added to the datagram by a sending host

and de-encapsulation where transmission information is removed by the receiving host as shown in Figure 5.2 (Kurose and Ross, 2010; Zhou *et al.*, 2018).

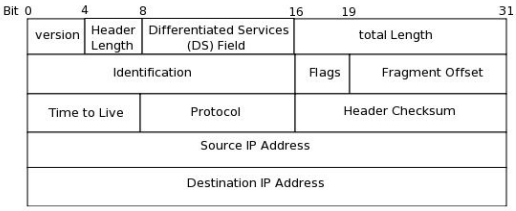
Data leaving the sending host travels from the application layer down to the network access layer with header data added at each layer. The data then travels out to the network and when it reaches the receiving host, the data moves from the network access layer up to the application layer while header information is removed at each layer. Together the header and datagram are known as a packet. The headers contain valuable data for transmission, to guide the packet to a recipient. The transmission data is stored on the hosts and networking devices on a network. Network transmission data is generated by the network layers; the headers contain data and analysis of this provides quality of service information about network transmission (information such as the port used, the type of end-to-end connection TCP or UDP etc.) (Zhou *et al.*, 2018). Thus, at the application layer the protocol being used appends its header to the data (e.g. HTTP, FTP, DNS), at the transport layer a TCP (see Figure 5.3a) or a UDP (see Figure 5.3b) header is appended. An IP header is appended at the network layer (see Figure 5.3c) and before data leaves the sending computer at the network access layer a frame header (also referred to as an Ethernet frame) (see Figure 5.3d), is appended to the data which is then converted into a bit-stream before being placed on a medium for transmission.



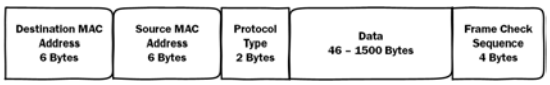
(a) TCP header format



(b) UDP header format



(c) IPv4 header format



(d) Ethernet frame format

Figure 5.3: Some TCP/IP model headers

Thus network data is generated from the headers appended to data as it travels from sender to recipient by various network devices in a network such as routers, switches,

servers, firewalls and hosts machines. Depending on the type of network being monitored, data is usually collected from packet data, flow data and log data. Each header can be collected and analysed to gain more understanding about the environment, however, for network performance monitoring, the headers normally used are the TCP, UDP and IP headers (Zhou *et al.*, 2018). This case study uses packet data and flow data with attributes like source and destination IP addresses, source and destination ports and protocol, packet size and flow duration used in the various scenarios.

5.3 Case Study Design

This section outlines the design of three scenarios developed as analysis tasks operating on network data, to validate the SDA tool. The case study utilizes all four modules of the SDA tool, namely data collection, transformation, analysis and visualisation. Network traffic analysis (NTA), which is the focus in this case study, involves the monitoring of network availability and activities to identify anomalies is an environment (Dutt *et al.*, 2020). Using a network dataset, analysis was done to understand what was happening in the environment by looking at the times the network was utilized and when it was not, to assess hosts with high traffic volumes and to assess the durations of transmissions. In the real world, NTA provides insights on how to optimize network performance and resources, playing an essential role in network management, and thus NTA is essential to IT environments (Abbasi *et al.*, 2021). Two data analysis scenarios were designed and carried out as listed below:

1. Determining peak and off-peak times: this analysis produced graphs showing peak and off-peak times for each weekday, i.e. Monday through Friday. The attributes that are used are time, flow size and flow duration.
2. Determining the top IP address for traffic in and out and flow rates for each weekday: the attributes that are used are source IP, destination IP, flow duration and flow size to show the amount of data transmitted.

5.3.1 Data Used

The dataset used in this study as described in Section 3.7, contains 87 attributes. Each instance holds the information of an IP flow generated by a network device i.e., source and destination IP addresses, ports, interarrival times, application layer protocol used on that flow as the class, amongst others. Most of the attributes are numeric but there

are also nominal types. The flow statistics (IP addresses, ports, inter-arrival times, etc) were obtained using CICFlowmeter[1]. The application layer protocol was obtained by performing a Deep Packet Inspection (DPI) processing on the flows with ntopng[2].

The dataset has data from several days in April 2017, namely Wednesday 26, Thursday 27 and Friday 28. The dataset also has data from May 2017, namely Tuesday 9, Thursday 11 and Monday 15. For purposes of this case study, we considered a week to consist of Monday 15 May, Tuesday 9 May, Wednesday 26 April, Thursday 11 May and Friday 28 April. We will assume that this data represents a typical week and that these are typical daily traffic records. Table 5.1 shows a sample of the data.

5.3.2 Network Data Preparation

The dataset was downloaded from the Internet [3] using the data collection module of the SDA tool and transformed before it was saved to the database using the data transformation module. This helped free up hardware resources such as hard drive space (storage) and processing time because the initial dataset was 1.7 GB and the transformed data ended up being 0.4 GB, i.e. 400 MB by filtering out data for dates other than those specified and unwanted columns. The preprocessing carried out to prepare the initial data are listed below:

- Load network data into the system.
- Filter out unwanted columns and rename the columns. Of the 87 attributes, all but 12 were filtered out which left the following; Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Flow Duration (Seconds), Flow Size (Bytes), Flow Packets, L7 Protocol, Protocol Name and Timestamp.
- Filter out unwanted days. We were left with 5 weekdays, Monday 15 May, Tuesday 9 May, Wednesday 26 April, Thursday 11 May and Friday 28 April.
- Rename column names as desired.
- Save data to new database, 'network_data'.

[1] http://www.unb.ca/cic/research/applications.html - https://github.com/ISCX/CICFlowMeter
[2] https://www.ntop.org/products/traffic-analysis/ntop/ - https://github.com/ntop/ntopn
[3] https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps

Table 5.1: Sample network data

Flow ID	Source IP	Source Port	Protocol	Flow Duration (Millisecond)	Flow Size (Bytes)	Flow Packets	L7 Protocol	Protocol Name	Timestamp
172.217.30.1-10.200.7.194-443-40147-6	10.200.7.194	40147	6	10532016	1635.7741955576	3.418149004	126	GOOGLE	5/15/17
192.168.60.77-10.200.7.5-62505-3128-6	192.168.60.77	62505	6	4192003	8247.3700519775	8.5877801137	7	HTTP	4/28/17
8.30.11.14-10.200.7.199-80-57408-6	10.200.7.199	57408	6	1	12000000	2000000	7	HTTP	11/5/17
192.168.220.105-10.200.7.9-51453-3128-6	192.168.220.105	51453	6	45001361	0.2666586017	0.0888862006	126	GOOGLE	5/15/17
192.168.60.44-10.200.7.5-49162-3128-6	192.168.60.44	49162	6	1003	17946.1615154536	2991.0269192423	131	HTTP_PROXY	4/28/17
172.19.1.97-10.200.7.8-53091-3128-6	172.19.1.97	53091	6	90015811	0.1999648706	0.1333099137	131	HTTP_PROXY	9/5/17
10.200.7.195-13.107.42.11-55227-443-6	13.107.42.11	443	6	80018024	6.111123164	0.1374690282	91	SSL	11/5/17
192.168.180.69-10.200.7.7-51241-3128-6	10.200.7.7	3128	6	6	23000000	666666.666666667	131	HTTP_PROXY	9/5/17
190.90.221.25-10.200.7.194-80-59054-6	10.200.7.194	59054	6	71300160	24603.8578314551	18.4291311548	147	WINDOWS_UPDATE	4/28/17
192.168.60.56-10.200.7.5-54882-3128-6	192.168.60.56	54882	6	1069	11225.4443405051	1870.9073900842	126	GOOGLE	5/15/17
192.168.10.45-10.200.7.7-49636-3128-6	192.168.10.45	49636	6	2676604	4676.8218234748	13.0762712751	70	YAHOO	4/28/17
178.250.0.75-10.200.7.217-443-53301-6	10.200.7.217	53301	6	1395066	5845.6015701049	17.9202991113	91	SSL	4/28/17
192.168.131.15-10.200.7.7-49768-3128-6	192.168.131.15	49768	6	11	1090909.09090909	181818.181818182	126	GOOGLE	4/26/17
192.168.40.30-10.200.7.6-53814-3128-6	192.168.40.30	53814	6	96933252	49598.9549592332	43.8342871237	7	HTTP	11/5/17
179.1.4.237-10.200.7.196-443-56398-6	10.200.7.196	56398	6	121406	96939.1957563877	230.6311055467	64	SSL_NO_CERT	4/28/17
192.168.72.136-10.200.7.8-52088-3128-6	192.168.72.136	52088	6	4400635	1123.2469859464	6.8171979726	120	TWITTER	11/5/17
192.168.72.62-10.200.7.7-52083-3128-6	192.168.72.62	52083	6	105910521	1782.8540377023	3.1441635529	131	HTTP_PROXY	11/5/17
192.168.220.177-10.200.7.7-58116-3128-6	10.200.7.7	3128	6	321	56074.7663551402	9345.7943925234	131	HTTP_PROXY	11/5/17
192.168.10.62-10.200.7.5-49572-3128-6	192.168.10.62	49572	6	79966852	27.8740496125	0.6127538946	126	GOOGLE	9/5/17
179.1.4.217-10.200.7.199-443-47596-6	10.200.7.199	47596	6	12914091	76.737882674	1.0066523459	126	GOOGLE	5/15/17

5.4 Data Analysis Scenario 1 - Determining peak and off-peak times

A peak time is a period of time in a network where the demand on resources is high causing the network to operate at capacity or near capacity while an off-peak time is a period of time where the network is operating at significantly lower than capacity (Morley *et al.*, 2018). This scenario analyses the network data to assess network traffic for each weekday's (Monday to Friday) peak and off-peak times in order to determine network utilization based on network traffic packet size and flow duration.

This scenario duplicates one of the data analysis tasks organisations use to administer network resources, for example, allocating higher bandwidth to more productive departments during the day and to other tasks that can be scheduled outside working hours to utilise network resources effectively. Similarly, organisations can obtain a good understanding of their network environment by analysing network traffic coming in and adequately allocate various resources to customers.

Input - The network data preprocessed as described in Section 5.3.2 and user commands for executing this scenario are outlined in the analysis steps given below.

Output - Graphs summarizing the information on peak and off-peak times.

5.4.1 Steps in ESxecuting Scenario 1

1. Load network data into the system.
2. View data and make decisions on the desired columns (Timestamp, Flow Size and Flow Duration) to select and check data types, nulls and duplicates.
3. Filter data for each day, Monday to Friday.
4. Apply appropriate statistical measures for analysis.
    - For each day group the time data into hours of the day.
    - For each day and for the whole week calculate the mean flow size and sum flow duration for each hour of the day with traffic recorded.
    - For each day and for the whole week calculate the sum flow size and sum flow duration for each hour of the day with traffic recorded.
    - Plot two bar graphs for each day, one showing mean flow size and mean flow duration over time and another showing sum flow size and sum flow duration.

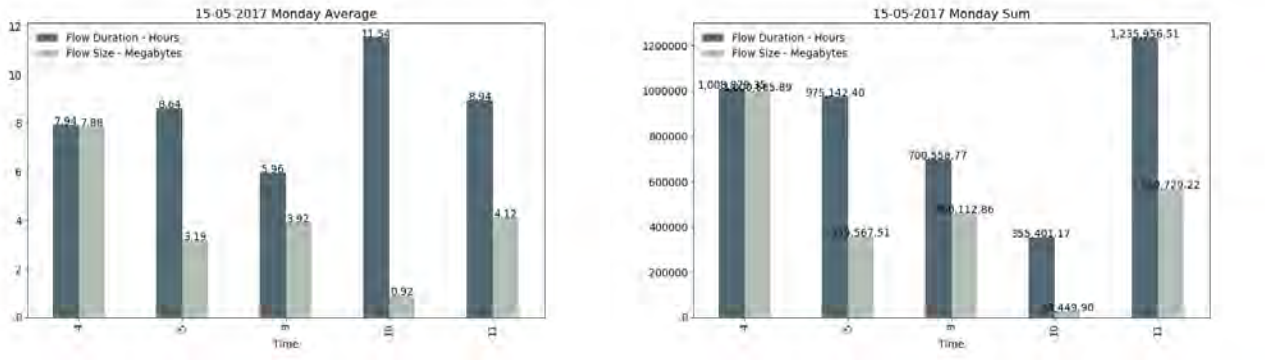
- Plot two bar graphs for the whole week, one showing mean flow size and mean flow duration over time and another showing sum flow size and sum flow duration.

5. Retrieve output information, graphs in this case.

6. Analyse the information produced.

5.4.2 Results and Discussion

A total of 12 graphs were plotted for this scenario, for each day two graphs, a detailed view and a broader view, as well as two for the whole week's view. Figure 5.4 shows the peak and off-peak analysis for Monday. Figure 5.4b gives the results for the total flow size in MB and flow duration in hours, where network traffic was recorded at 4 am, 5 am, 9 am, 10 am and 11 am with peak flow duration at 11 am and peak flow size at 4 am. A combination of flow duration and flow size shows that the peak time was at 4 am. Similarly, Figure 5.4a depicts the average flow duration in hours and the average flow size in MB; 10 am shows the highest average flow duration and 4 am the highest average flow size. Similar to Figure 5.4b, the combination of flow duration and flow size shows that the peak time was at 4 am. The other hours of the day that are not shown recorded little to no traffic.



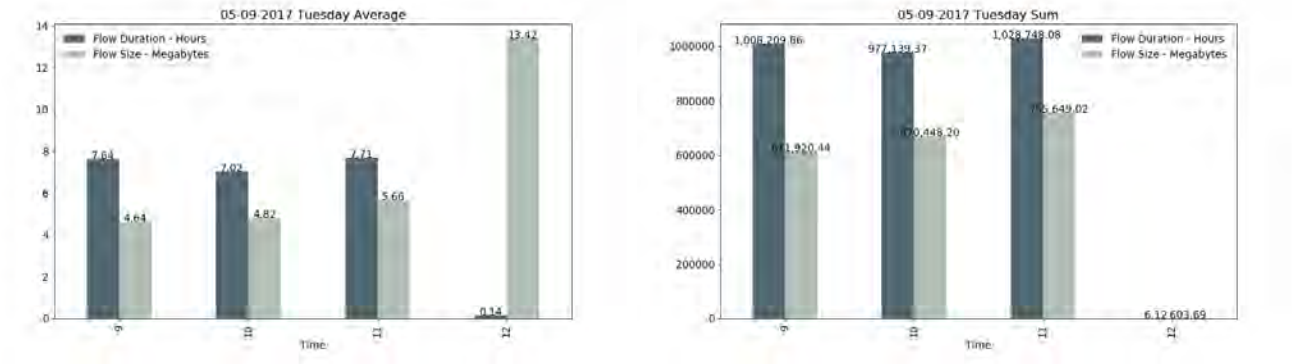
(a) Monday 15-05-2017 Average

(b) Monday 15-05-2017 Total

Figure 5.4: Monday 15-05-2017 peak and off-peak analysis

Considering Figure 5.4, a proportional distribution between flow duration and flow size is seen at 4 am while the other times show a disproportional distribution where the time taken to transmit data is significantly more than the amount of data relative to the standard 1 Gigabits/s network throughput (Queiroz *et al.*, 2019). This pattern would indicate that users were active during these hours as the computers would open sessions

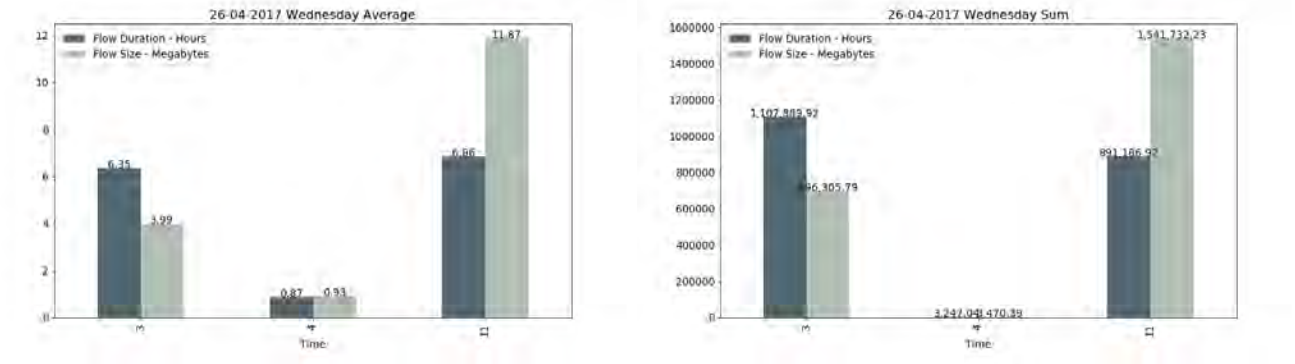
and mostly transmit upon instruction from a user, in other instances where throughput is measured, these results would suggest a bottle neck or slow connection speed.



(a) Tuesday 05-09-2017 Average (b) Tuesday 05-09-2017 Total

Figure 5.5: Tuesday 05-09-2017 peak and off-peak analysis

Figure 5.5 shows Tuesday's flow trends. Network traffic was recorded at 9 am, 10 am, 11 am and 12 pm. The distribution is similar to Monday's trend for the 9 am, 10 am and 11 am, however, for 12 pm the average flow duration is significantly lower than the average flow size as shown in Figure 5.5a. This would suggest that these were likely automated transmissions, for example servers within the network carrying out various tasks by transmitting a fair amount of data in a small period of time. Figure 5.5b shows a similar distribution for 12 pm. The peak time can been seen at 11 am where both flow duration and size are the highest as shown in Figure 5.5b, however, in Figure 5.5a, it is the flow duration that is highest.



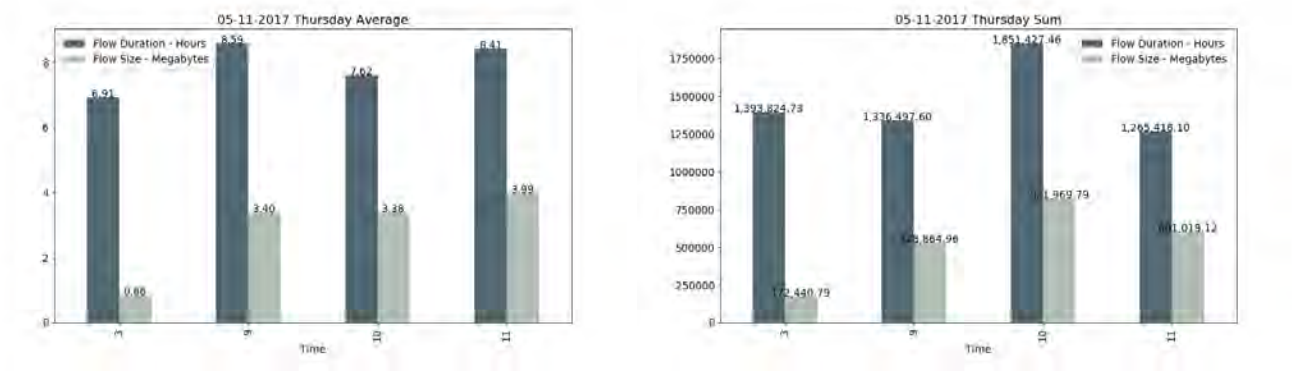
(a) Wednesday 26-04-2017 Average (b) Wednesday 26-04-2017 Total

Figure 5.6: Wednesday 26-04-2017 peak and off-peak analysis

Figure 5.6 shows Wednesday's trend, with network traffic recorded at 3 am, 4 am and 11 am. Both Figure 5.6a (showing averages) and 5.6b (showing totals), indicate similar

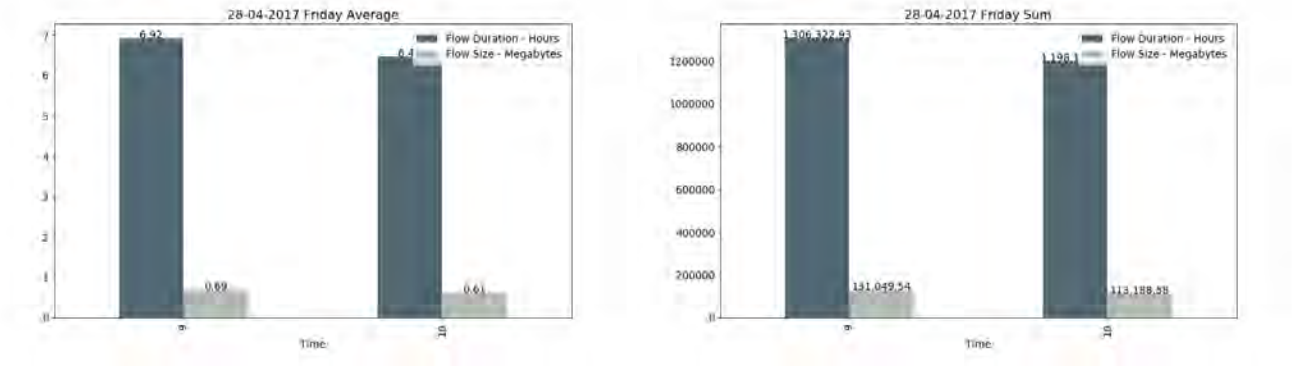
trends, where users are active at 3 am, both users and computers are active at 4 am and more computers than users are active and at 11 am. When both sub-figures are considered, the peak time is at 11 am with both averages and the total flow size being high.

Figure 5.7 shows Thursday's trend, with network traffic recorded at 3 am, 9 am, 10 am and 11 am. Figure 5.7a (showing averages) and 5.7b (showing totals) confirm that mainly users are active. The peak time is seen at 10 am in Figure 5.7b showing highest readings for both flow duration and size, however, according to Figure 5.7a, the highest averages do not occur at 10 am but rather highest flow duration is recorded at 9 am and highest flow size at 11 am.



(a) Thursday 05-11-2017 Average (b) Thursday 05-11-2017 Total

Figure 5.7: Thursday 05-11-2017 peak and off-peak analysis



(a) Friday 28-04-2017 Average (b) Friday 28-04-2017 Total

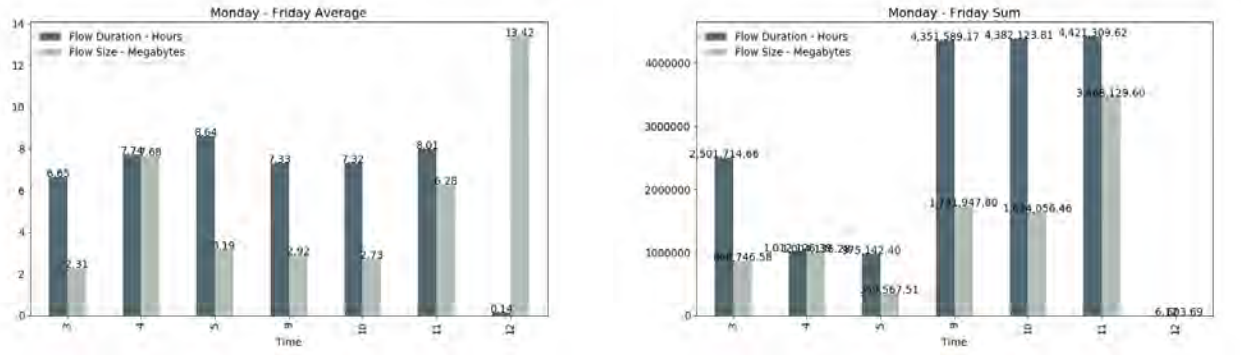
Figure 5.8: Friday 28-04-2017 peak and off-peak analysis

Figure 5.8 shows Friday's trend, with network traffic recorded between 9 am and 10 am only, the least number of hours recorded in all of the weekdays which indicates the lowest network activity of the week. Figure 5.8a (showing averages) and 5.8b (showing totals),

illustrates that mainly users are active. Peak time is seen at 9 am with both flow duration and size highest by total and on average.

Figure 5.9 shows the entire week's trend i.e. from Monday to Friday, with network traffic recorded at 3 am, 4 am, 5 am, 9 am 10 am, 11 am and 12 pm of the various days. Figure 5.9a (showing averages) and 5.9b (showing totals), illustrate that the period with the most automated transmissions (due to high amounts of data being transmitted and in a relatively short duration of time) is at 12 pm from a weekly point of view. Regarding network traffic, network traffic was recorded once (one week day) each at 5 am and 12 pm, twice (on two days each) at 3 am and 4 am and four times each at 9 am, 10 am and 11 am. From a weekly point of view, the highest network activity by total flow duration and size was recorded at 9 am, 10 am and 11 am with the peak time seen at 11 am.

For the weekdays, Monday recorded the highest number of hours with network traffic, five of the hours, while Friday recorded the lowest number of hours with network traffic, two of the hours. An average of 3.6 hours were recorded per day to have network traffic.



(a) Monday - Friday Average

(b) Monday - Friday Total

Figure 5.9: Monday to Friday peak and off-peak analysis

5.5 Data Analysis Scenario 2 - Determining the top IP address for traffic in and out

This scenario calculates the flow rate, that is the time taken for a given amount of data to flow. The aspects taken into consideration are flow duration and flow size to determine the flow rate in the network data based on source IP addresses and ports and destination IP addresses and ports.

This scenario analyses the network data to ascertain weekday and weekly information on top IP source addresses by looking at the amount of data transmitted and duration

of these transmissions. With this information from this type of network, analysis traffic flows can be better understood. For example hosts that send large volumes of data should normally be those that host applications that offer desired services to users.

Input - The network data preprocessed as discussed in Section 5.3.2, and user commands for executing this scenario as outlined in the analysis steps given below.

Output - Tables summarizing the information on top IP by traffic in and out and ports.

5.5.1 Steps in Executing Scenario 2

1. Load data into the system.
2. View data and make decisions on the desired columns to select and check data types, nulls and duplicates.
3. Filter data for each day, Monday to Friday.
4. Apply appropriate statistical measures for analysis.
    - Identify inbound traffic flows to determine IP source addresses.
    - Calculate the flow rate by converting flow duration column to seconds and flow size column to bytes then dividing the flow size column with the flow duration column, the result to be placed in a new column labelled "Flow Rate"
    - Group the data by source IP with an average flow rate calculated for each IP.
    - Sort the data by the flow rate column to identify the top 10 source and top destination IP addresses.
5. For inbound and outbound traffic generate a top IP list.
6. Retrieve information, in the form of tables in this case.
7. Analyse the information produced.

Note that as the automated enhancements had not been implemented at the time of running those analysis tasks, all weekday/weekly tasks were repeated manually. This would however have been an ideal example of how automated macro enhancement would have saved user time and effort.

5.5.2 Results and Discussion

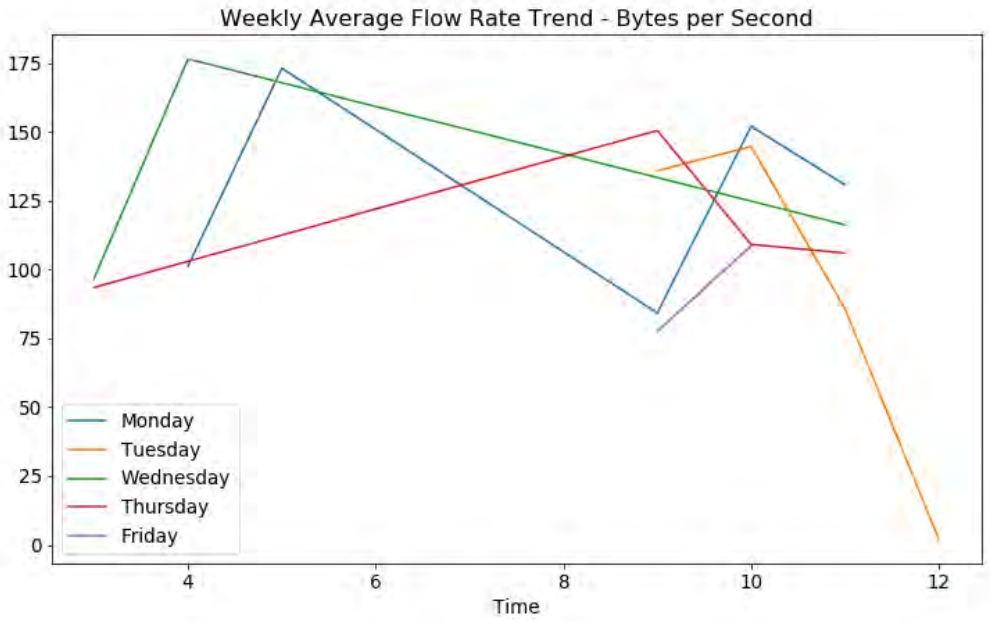


Figure 5.10: Weekly average flow rate

Figure 5.10 shows weekly average flow rates calculated as bytes per second for the peak hours. The highest flow rates were recorded between 4 am and 5 am while the lowest were recorded at 12 pm. The figure further shows that the lines indicating the flows for Monday through Friday are concentrated between 9 am and 11 am as earlier shown in scenario 1.

Firgure 5.11 shows the top 10 source IP addresses for each day from Monday to Friday and top 10 source IP addresses for a week, giving a broader view. Figure 5.11a to 5.11e shows the top 10 for Monday to Friday respectively, while Figure 5.11f shows the combined top 10 for the week.

From Figure 5.11, Tuesday has the highest flow rate while Friday has the lowest flow of the top 10 IP addresses. It is further noted that Tuesday has the highest average flow rate of all the days with five of the 10 IP addresses in the week's top 10 (Figure 5.11f) appearing in Tuesday's top 10. At least one of the IP addresses appearing in a single day's top 10 also appears in the week's top 10 indicating that network performance was somewhat consistent.

Source IP	Flow Rate
149.5.0.66	47,350.1895
8.253.68.87	24,282.2613
74.125.165.82	20,606.4371
209.85.224.151	9,431.0255
185.163.111.131	6,930.4857
158.69.228.114	6,447.8220
198.54.201.15	6,317.0367
185.163.110.100	6,129.3830
190.90.221.50	5,733.7469
173.194.136.169	5,547.2520

(a) Top 10 IP Addresses Average - Monday

Source IP	Flow Rate
199.91.152.26	108,451.1520
188.165.12.56	74,235.2472
31.216.144.21	37,603.1807
31.216.145.10	25,721.4374
192.168.10.62	22,126.3465
173.194.27.250	9,396.9970
173.194.143.53	9,189.8680
91.207.103.197	6,385.9933
173.194.27.187	4,192.9150
217.79.182.144	4,074.4943

(b) Top 10 IP Addresses Average - Tuesday

Source IP	Flow Rate
37.48.82.77	13,590.5295
190.60.12.28	13,514.2700
98.158.99.143	11,531.5302
173.194.27.247	11,365.2720
96.7.11.95	11,027.3255
190.90.221.26	10,685.8663
178.79.164.222	9,345.8389
185.163.111.167	8,832.8285
173.194.27.199	7,867.8730
217.20.152.74	5,494.8010

(c) Top 10 IP Addresses Average - Wednesday

Source IP	Flow Rate
8.253.69.241	50,638.6708
8.253.48.249	23,959.5295
185.163.111.135	22,012.2386
190.90.221.26	17,628.7965
173.194.141.220	17,182.5003
185.120.147.2	14,711.4891
185.120.144.228	13,035.3752
8.253.165.248	12,858.1357
185.120.147.19	12,383.7986
216.58.222.202	11,617.9639

(d) Top 10 IP Addresses Average - Thursday

Source IP	Flow Rate
216.58.222.106	31,617.1050
162.125.18.6	20,447.8256
91.207.103.11	17,237.9680
179.1.4.205	14,652.3496
173.194.24.252	14,609.5700
185.102.219.20	10,233.8476
192.168.32.65	6,651.2901
172.18.1.34	6,087.2542
93.184.216.178	5,530.0733
91.228.167.26	4,049.0505

(e) Top 10 IP Addresses Average - Friday

Source IP	Flow Rate
199.91.152.26	108,451.1520
188.165.12.56	74,235.2472
8.253.69.241	50,638.6708
149.5.0.66	47,350.1895
31.216.144.21	37,603.1807
31.216.145.10	25,721.4374
8.253.68.87	24,282.2613
8.253.48.249	23,959.5295
185.163.111.135	22,012.2386
162.125.18.6	18,295.4262

(f) Top 10 IP Addresses Average - Monday to Friday

Figure 5.11: Top 10 IP addresses based on flow rate

5.6 Summary

This chapter first discussed some general concepts of a typical network environment. Thereafter, analysis scenarios common to computer networks were developed and executed with the aim of testing the SDA tool. Morley *et al.* (2018) suggest that better understanding of everyday practices by network users would assist in the provision and design of computer network services. The scenarios covered in this case study are typical of some of the ways in which a computer network can be better understood.

For this case study, data processing and presentation were the focus in testing the SDA

tool. Data was preprocessed during data collection and transformed by cleaning and updating. Statistical analysis involving means and sums were applied and visualisations such as tables and graphs were used to present the results.

Chapter 6

A Case Study using Log Data

This case study reinforces the generality of the SDA tool's application by using a different type of data, log data. This type of data offers some challenges in its preparation which is one of the focus areas of this case study.

6.1 Data Logging in Context

Best practices in computing recommend that software, network devices and other IT hardware record and document events or activities in log files (Gottesdiener, 2003; Zhang, 2018). A log file is a collection of system generated, time-stamped records of events relevant to a particular system. Log files, also referred to as logs, are text files with chronologically sorted messages of events, which are operations occurring within a system. Logs exist in most IT environments and are used for various purposes such as security, troubleshooting and general resource monitoring (Wu *et al.*, 2019).

Log data analysis is the process of transforming log data into information, i.e. making sense of log files. System errors, anomalies and suspicious activities that deviate from the normal can be identified by various log analysis methods and techniques within the context of the environment in which the log has been generated. Log generation should be designed to aid log analysis, for example, by using simple and clear terminology, effective formatting and so on. Du *et al.* (2017) state that the primary purpose of a system log is to record system states and significant events at various critical points to help debug system failures and perform root cause analysis. Such log data is universally available in nearly all computer systems. Thus log data is an important and valuable resource for understanding a system's status and performance issues, and therefore, the various system

logs are naturally a good source of information for monitoring and anomaly detection and generally, for computer systems management (Svacina *et al.*, 2020).

Some common examples of log analysis related to computer security are discussed below:

- On web servers, access logs list all the individual files that users have requested from a website. These files include the HTML files and their embedded graphic images and any other associated files that get transmitted. From the server's log files, an administrator can identify the number of visitors to the website, the domains from which they are visiting, the number of requests for each page and usage patterns according to variables such as times of the day, week, month or year.
- An audit log (also known as an audit trail) records chronological documentation of any activities that could have affected a particular operation or event. Details typically include the resources that were accessed, destination and source addresses, a time-stamp and user login information for the user who accessed the resources.
- In Microsoft Exchange, a transaction log records all changes made to an Exchange database. Information to be added to a mailbox database is first written to an Exchange transaction log. Thereafter, the contents of the transaction log are written to the Exchange Server database, an approach which is common in many database systems.

Some common examples of log analysis related to resource monitoring and trouble shooting are discussed below:

- To make informed decisions about system performance such as identifying areas to add more resources or apply scheduling of some time insensitive processes, information on resource utilization can be gained from logs by analysing key trends across different resources. Various resources in a computer environment such as the CPU, memory, disks, Ethernet, wireless connection and database generate logs through the operating system or specific application or software.
- When troubleshooting a website using log analysis to handle HTTP errors, information gained helps in the understanding of HTTP errors and on what pages they occurred so that a solution to the problem can be developed and implemented.

Log data is an ideal example for validating the basic SDA tool as it typically requires significant cleaning and transformation of the data in order to access the information. Moreover, manipulation of data using various statistical techniques and a range of visualisations provide for a realistic test of this tool.

6.2 Overview of Log Data

The format of log data is highly unstructured, and at the very least such data consists of a date and message attributes, stored as text files. Some other attributes can be added to the format as required by different systems and applications in various environments. The semantics of log data can vary significantly from system to system (Du *et al.*, 2017). The description of the event being logged can be in natural language form and detailed to describe the event technically or in simple terms. Records are always appended to the existing file and by default logs are a time-ordered sequence of records; essentially logs record what happened and when it happened (Kreps, 2013). Three types of information are recorded, namely, system information, network information and user information.

Logs contain data with different attribute types, and not only high-dimensional data. Logs exist in large volumes because many events in an IT environment are logged continuously over time. Bush (2020) highlights some techniques for use when analysing log data, one of which is log classification. Log classification, also referred to as classification and tagging, is a process of ordering messages into different classes or tagging them with different keywords for analysis. Zou *et al.* (2016) explain that a number of algorithms exist for log classification. To be used for analysis, a good algorithm must be able to discover the structure of the log that exists in the subspaces of the data and ignore the order of the input data. Moreover, for log classification, the position of words in the log is more important than that in other normal texts, as the sequence of words is not overwhelming (Zou *et al.*, 2016). With relatively large sized log files, it is both time consuming and prone to erroneous results to go through each record manually, and thus several correlation techniques can be applied to logs using software applications to generate information from the log data. Bush (2020) explains the common techniques for log data analysis, as detailed below:

- Normalization is the process of cleaning logs so that they adhere to the same standards or formats. For example, if logs from various sources contain varying date-time formats, they should be normalized before proceeding with any analysis.
- Pattern recognition is the process of identifying patterns in logs, so that individual log entries can be handled appropriately. For example, considering the logs collected by an e-commerce platform, log entries that refer to users signing in should be separated from log entries that refer to users signing out.
- Classification and tagging is another process that involves categorizing individual log entries. In this case, log entries should be further classified, for example, based on keywords that may be present in the entries themselves.

- Correlation analysis is the process of finding log entries that are related. This may refer to identifying which entries pertain to a specific event, or identifying which entries (pertaining to separate events) are related. As in other types of data analysis, identifying correlations is an essential step in drawing meaningful conclusions from logs.
- Artificial ignorance is the process of ignoring entries that are not useful for analysis. In web-based applications, artificial ignorance may be applied to identify which logs relate to intended usage patterns. With the help of artificial ignorance, it is possible to significantly reduce the number of logs that must be analysed, which can speed up automated analysis processes or even make manual analysis a possibility.

Logs are simple resources that have significant value for informed decision making.

6.3 Case Study Design

This section outlines the design of two data analysis scenarios on log data, to validate the SDA tool. This case study focuses on data preparation and presentation of information, since up to 80% of the time required to carry out real world data analysis is usually spent on data preparation as this is the foundation of the analysis process (Losarwar and Joshi, 2012). Data preparation of log data represents the most time-consuming phase of the log analysis process due to complex data structures and thus, it is important to correctly and accurately prepare log data for analysis as quality of results depend highly on the quality of analysed data (Wu *et al.*, 2019; Munk *et al.*, 2010). Thus with data preparation and presentation as a focus, two scenarios were designed and carried out using a combination of normalisation and analysis techniques as listed below:

1. Implementing data cleaning and transformations in carrying out data preparation.
2. Implementing a general overview of events over time with specific analysis of web pages accessed to highlight some presentation techniques available.

The data used in this study was described in Section 3.7. A random sample of the log data is shown in Table 6.1.

Table 6.1: Sample log data (original raw data)

IP	Time	URL	Staus
10.128.2.1	[29/Nov/2017:06:58:55	GET /login.php HTTP/1.1	200
10.128.2.1	[29/Nov/2017:06:59:02	POST /process.php HTTP/1.1	302
10.128.2.1	[29/Nov/2017:06:59:03	GET /home.php HTTP/1.1	200
10.131.2.1	[29/Nov/2017:06:59:04	GET /js/vendor/moment.min.js HTTP/1.1	200
10.130.2.1	[29/Nov/2017:06:59:06	GET /bootstrap-3.3.7/js/bootstrap.js HTTP/1.1	200
10.130.2.1	[29/Nov/2017:06:59:19	GET /profile.php?user=bala HTTP/1.1	200
10.128.2.1	[29/Nov/2017:06:59:19	GET /js/jquery.min.js HTTP/1.1	200
10.131.2.1	[29/Nov/2017:06:59:19	GET /js/chart.min.js HTTP/1.1	200
10.131.2.1	[29/Nov/2017:06:59:30	GET /edit.php?name=bala HTTP/1.1	200
10.131.2.1	[29/Nov/2017:06:59:37	GET /logout.php HTTP/1.1	302
10.131.2.1	[29/Nov/2017:06:59:37	GET /login.php HTTP/1.1	200
10.130.2.1	[29/Nov/2017:07:00:19	GET /login.php HTTP/1.1	200
10.130.2.1	[29/Nov/2017:07:00:21	GET /login.php HTTP/1.1	200
10.130.2.1	[29/Nov/2017:13:31:27	GET / HTTP/1.1	302
10.130.2.1	[29/Nov/2017:13:31:28	GET /login.php HTTP/1.1	200
10.129.2.1	[29/Nov/2017:13:38:03	POST /process.php HTTP/1.1	302
10.131.0.1	[29/Nov/2017:13:38:04	GET /home.php HTTP/1.1	200
10.130.2.1	[29/Nov/2017:13:38:19	GET / HTTP/1.1	302

In the case study, relationships between the initial four attributes were assessed, to answer various questions about the data, including: the frequency of URLs accessed for the available IP addresses per month and for the whole period, existing URL and status relationships and common URLs that were accessed over the duration of the collection period. In addressing these questions, some key aspects of the SDA tool were utilised, such as data cleaning and transformation (preparing the data for the subsequent analysis) and the analysis and visualization components (generating information from the data).

6.4 Data Analysis Scenario 1 - Implementing data cleaning and transformations

This scenario describes a typical initial cleaning process prior to working on data for analysis. A look at the data and objectives of the analysis determine the processes to be executed. Planning and execution of the plans for data preparation for this case study was done by understanding the data analysis objectives developed from the questions

stated in the previous section, to assess the common IP addresses and URLs accessed per month and for the duration of the data collection period, to access the URLs and their relationships with statuses.

6.4.1 Steps in Executing Scenario 1

1. Load weblog data into the system.
2. View data, noting the column names and data in the cells.
3. Check the summary statistics which shows for each column the count (number of rows), unique value, top or most frequent value and frequency of the top value.
4. Clean and transform the data by performing the following:
    - Time column; Remove the leading character "[", split the date and time components into new Date and Time columns and format as such.
    - Correct the "Staus" column name to "Status".
    - Filter out all erroneous values using the Status column, those with unwanted values have a "No".
    - From the Date column, extract the year, month and day values and create new Year, Month and Day columns with the extracted values.
    - From the URL column extract the method attribute and file name to create two new columns Method and File_Name.
    - In the File_Name column, clean the data to be left with the file name and file extension only.
    - Filter out all file names not ending in .php as the website uses PHP for its web pages, the other files were .js for JavaScript.
    - Check data types, nulls and duplicates while making corrections where necessary and possible.
5. Extract a sample of the data to a CSV file.
6. Retrieve CSV output file.
7. Analyse the data produced with the summary statistics before and after the changes.

6.4.2 Results and Discussion

The initial summary statistics as shown in Table 6.2, highlights some important metrics of the log data. Firstly, *count* shows that there are 16,007 rows of data (excluding the header). Moreover, for each of the four attributes or columns there are also 16,007 rows, indicating that there are no null values in any of the cells. Next, *unique*, shows the number of unique values for each column, while *top*, indicates the most common value in each column. Finally, *frequency* indicates how often the top value appears in the data.

Table 6.2: Summary statistics from original data

	IP	**Time**	**URL**	**Staus**
count	16007	16007	16007	16007
unique	16	7307	314	13
top	10.128.2.1	cannot	GET /login.php HTTP/1.1	200
freq	4257	167	3284	11330

Table 6.3 shows the summary statistics of the cleaned log data similar to Table 6.2. After carrying out data cleaning and the transformation steps listed in the Steps paragraph in Section 6.4, there are now ten attributes with 9499 rows containing the desired data for each column. The number of unique values is indicated for each column with the top occurring values shown together with the associated frequencies.

Table 6.3: Cleaned data summary statistics

	IP	**URL**	**Status**	**Date**	**Time**	**Day**	**Month**	**Year**	**Methods**	**File_Name**
count	9499	9499	9499	9499	9499	9499	9499	9499	9499	9499
unique	5	261	2	56	5547	26	5	2	3	34
top	10.128.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:54:33	29	Jan	2017	GET	login.php
freq	2636	3284	6742	4045	15	4397	4125	4755	8808	3426

Table 6.4 shows a randomly selected sample of the cleaned log data. Note the differences from Table 6.1, with the number of attributes being ten instead of four. These were derived from the four initial attributes to meet the objectives of the case study.

Table 6.4: Sample cleaned log data

IP	URL	Status	Date	Time	Day	Month	Year	Methods	File_Name
10.130.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:29:42	29	Jan	2018	GET	login.php
10.131.0.1	GET /archive.php HTTP/1.1	200	1/16/18	06:03:04	16	Jan	2018	GET	archive.php
10.131.2.1	GET /home.php HTTP/1.1	200	11/19/17	06:54:20	19	Nov	2017	GET	home.php
10.128.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:29:28	29	Jan	2018	GET	login.php
10.128.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:52:28	29	Jan	2018	GET	login.php
10.128.2.1	GET /login.php HTTP/1.1	200	11/30/17	16:52:06	30	Nov	2017	GET	login.php
10.131.0.1	GET /home.php HTTP/1.1	302	1/29/18	20:35:48	29	Jan	2018	GET	home.php
10.130.2.1	GET /home.php HTTP/1.1	302	1/29/18	20:30:59	29	Jan	2018	GET	home.php
10.131.0.1	GET /login.php HTTP/1.1	200	1/29/18	20:39:45	29	Jan	2018	GET	login.php
10.128.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:53:18	29	Jan	2018	GET	login.php
10.130.2.1	GET /allsubmission.php HTTP/1.1	200	12/2/17	16:47:17	02	Dec	2017	GET	allsubmission.php
10.128.2.1	GET /home.php HTTP/1.1	302	1/29/18	20:41:33	29	Jan	2018	GET	home.php
10.130.2.1	GET /login.php HTTP/1.1	200	1/29/18	20:27:54	29	Jan	2018	GET	login.php
10.128.2.1	GET /contestproblem.php?name=RUET%20OJ%20Server%20Testing%20Contest HTTP/1.1	200	11/29/17	14:56:04	29	Nov	2017	GET	contestproblem.php
10.130.2.1	GET /archive.php HTTP/1.1	200	11/23/17	13:51:41	23	Nov	2017	GET	archive.php
10.131.0.1	GET /home.php HTTP/1.1	302	1/29/18	20:18:07	29	Jan	2018	GET	home.php
10.128.2.1	GET /contestproblem.php?name=RUET%20OJ%20Server%20Testing%20Contest HTTP/1.1	200	12/1/17	18:07:16	01	Dec	2017	GET	contestproblem.php
10.131.0.1	GET /login.php HTTP/1.1	200	11/30/17	12:16:49	30	Nov	2017	GET	login.php
10.128.2.1	GET /contestproblem.php?name=RUET%20OJ%20Server%20Testing%20Contest HTTP/1.1	200	11/30/17	18:00:03	30	Nov	2017	GET	contestproblem.php
10.128.2.1	GET /home.php HTTP/1.1	302	1/29/18	20:32:28	29	Jan	2018	GET	home.php
10.131.0.1	GET /login.php HTTP/1.1	200	1/29/18	20:20:39	29	Jan	2018	GET	login.php

The work carried out in this scenario highlighted some important requirements of the SDA tool to support the steps taken in typical log data analysis tasks. For example the view of data and its general composition (summary statistics) helps in better understanding the data, adding to an existing analysis plan to meet the objectives. Some other aspects included the filtering, splitting, slicing of data as well as removing of unwanted characters and data.

6.5 Data Analysis Scenario 2 - Implementing a general overview of events

A general overview of events logged was generated from the log data. Essentially this is some basic information that can be produced after analysis. In this scenario the log

data was analysed and generic information was produced. The SDA tool's analysis and presentation components were used to do this. In so doing the effectiveness of these components was explored. The information produced shows the relationship between the IP addresses and URLs over the duration of the data collection period and the URLs and status and other information such as most used method, URL and status returned.

6.5.1 Steps in Executing Scenario 2

1. Access the cleaned weblog data in the database system.
2. Load the cleaned and transformed data for analysis.
3. Apply appropriate statistical measures for analysis.
    - Generate a pivot table for IP addresses and files they accessed over time.
    - Generate a pivot table for URLs and Statuses.
    - Plot a horizontal bar graph showing the IP distribution.
    - Plot a horizontal bar graph showing the days of month that recorded most or least traffic.
    - Plot a bar graph showing monthly logins.
    - Plot a pie chart showing the methods used to access the URLs.
    - Plot a pie chart showing the statuses returned.
4. Retrieve output information, in this case the tables and graphs created.
5. Analyse the information produced.

Note that as explained in Section 5.5 no automation enhancements were used in executing these scenarios.

6.5.2 Results and Discussion

For the first part of this scenario, the relationship between the file accessed and the IP that accesses it is shown in Tables 6.5, 6.6 and 6.7. Table 6.5 shows a pivot table of 34 web pages and the five IP addresses for the entire duration the data was recorded, as indicated in Table 6.3 in the unique row. The data shows that IP 10.128.2.1 accessed the *login.php* web page the most of all the IP addresses (a total of 1,027 times), while IP 10.131.2.1 accessed the same web page the least number of times, with 182 times recorded. The most accessed web page was *login.php* while the least accessed web page was *adminpanel.php*.

Table 6.5: IP addresses and files accessed during the whole period of collection

File_Name	**10.128.2.1**	**10.129.2.1**	**10.130.2.1**	**10.131.0.1**	**10.131.2.1**
action.php	25	6	11	32	9
adminpanel.php	0	0	1	0	0
allsubmission.php	43	24	30	44	29
announcement.php	1	6	0	0	1
archive.php	64	51	71	69	54
compile.php	15	16	23	23	19
compiler.php	20	7	27	30	14
contest.php	55	46	54	44	50
contestproblem.php	116	100	111	117	112
contestshowcode.php	0	3	0	1	2
contestsubmission.php	44	53	46	52	33
contestsubmit.php	10	10	14	8	11
countdown.php	10	14	14	13	22
createadmin.php	2	0	2	0	0
description.php	22	25	33	24	20
details.php	59	60	63	60	55
edit.php	4	3	3	2	2
editcontest.php	1	0	0	2	0
editcontestproblem.php	3	5	0	2	2
home.php	875	103	790	794	93
index.php	3	0	1	1	1
login.php	1027	194	1002	1021	182
logout.php	11	6	7	11	9
pcompile.php	17	16	20	15	12
process.php	65	46	85	86	35
profile.php	48	20	24	31	24
setcontest.php	2	0	0	2	2
setcontestproblem.php	0	0	0	1	3
setproblem.php	4	1	0	5	2
showcode.php	13	7	5	18	11
sign.php	35	14	27	45	15
standings.php	32	27	35	37	36
submit.php	7	12	17	11	7
update.php	3	2	0	2	0

Table 6.6: IP addresses and files accessed during December 2017

File_Name	**10.128.2.1**	**10.129.2.1**	**10.130.2.1**	**10.131.0.1**	**10.131.2.1**
action.php	3	0	2	9	2
allsubmission.php	4	0	4	8	2
archive.php	9	2	14	10	7
compile.php	4	3	8	6	0
compiler.php	3	3	8	10	3
contest.php	9	2	6	3	4
contestproblem.php	13	9	14	19	4
contestsubmission.php	8	3	5	4	1
description.php	2	1	4	2	3
details.php	4	5	10	13	3
edit.php	0	1	0	1	0
editcontestproblem.php	0	0	0	1	0
home.php	26	7	41	42	7
index.php	0	0	1	0	0
login.php	72	17	91	105	22
logout.php	0	1	1	2	0
pcompile.php	3	2	1	2	1
process.php	14	4	18	23	3
profile.php	4	2	3	3	1
showcode.php	0	0	0	3	1
sign.php	3	0	4	12	2
standings.php	7	4	4	8	3
submit.php	2	1	1	2	3

Table 6.7: IP addresses and files accessed on 21 December 2017

File_Name	**10.128.2.1**	**10.130.2.1**	**10.131.0.1**
action.php	0	0	1
archive.php	1	0	0
compile.php	0	1	0
compiler.php	0	2	0
contest.php	0	2	0
description.php	0	2	0
home.php	1	6	0
login.php	2	1	9
process.php	2	1	1
sign.php	0	0	1

Table 6.6 shows a portion (drilled down) of the data in Table 6.5 but for the month of December 2017. This month recorded 23 of the 34 unique web pages accessed by the five IP addresses. The web page *login.php* was the most accessed while *index.php* was the least accessed.

Table 6.7 shows the same files accessed and IP address relationships but for one day, the 21st of December 2017. On this day, ten of the 34 unique web pages was accessed with *home.php* the most accessed and *sign.php, compile.php, archive.php* and *action.php* the joint least accessed web pages.

Table 6.8 shows a pivot table of the file names and status code columns. Two types of status were recorded. The 200 status code is the most commonly returned one and means that the request was received, understood and is being processed. The 302 status code is a redirect status response code that indicates that the resource requested has been temporarily moved to the URL given by the Location header, i.e. redirection to another URL. Table 6.8 shows some of the 34 web pages and the recorded status codes. The data is sorted by the 200 status code where the first ten rows are the top 10 records and the last five rows are the bottom 5 records. From the table, most web pages recorded the normal 200 code, however, we noted that three web pages, namely, *process.php, action.php* and *logout.php* did not return this code but rather have a number of redirects. The *admin-panel.php* file was accessed once and also returned a redirect. The web page *home.php* recorded 2,169 redirects as well as a large number of a normal status codes, which may indicate that it is a central point of the website/system as it does serve some requests itself and redirects to other web pages for other services.

Execution of the second part of this scenario resulted in the generation of some basic information about the data, some of which is similar to that generated in the first part of this scenario. The information presented and discussed relates to the IP distribution, days and months for logins, URL methods used and URL status codes returned.

Figure 6.1 shows that there are five IP addresses in the data, with IP addresses 10.128.2.1, 10.131.0.1 and 10.130.2.1 recording just over 2,500 events/accesses each on the website while 10.131.2.1 and 10.129.2.1 recorded about 800 events each. Address 10.128.2.1 accessed the website the most while 10.131.2.1 accessed the website the least.

Table 6.8: Files accessed and Status for full dataset

File_Name	**200**	**302**
login.php	3418	8
home.php	486	2169
contestproblem.php	481	75
archive.php	302	7
details.php	288	9
contest.php	240	9
contestsubmission.php	226	2
allsubmission.php	170	0
standings.php	161	6
profile.php	139	8
editcontest.php	1	2
process.php	0	317
action.php	0	83
logout.php	0	44
adminpanel.php	0	1

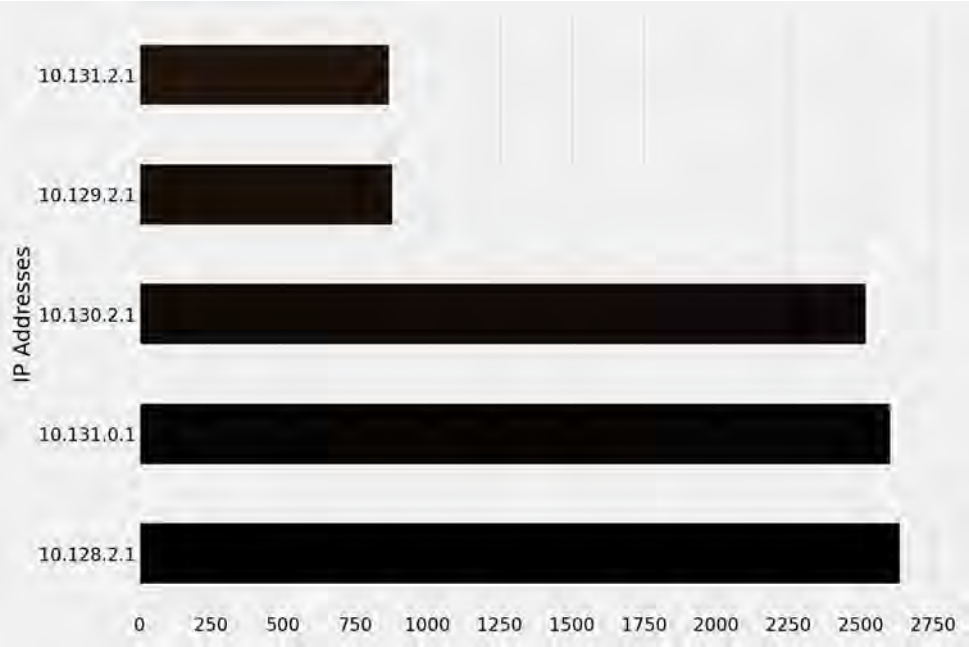


Figure 6.1: IP distribution

Analysis of accesses on different days of the month is presented in Figure 6.2. A typical month has 30 days but some have a 31st day. If we consider all 31 days of a month we

notice that the information in Figure 6.2 shows that over the period that the data was generated, only 26 days of the 31 days recorded any activity, i.e. users accessed the website on only 26 days in a month. The missing days are 4, 5, 6, 7 and 31; however, without further information about the website such as the operating environment, we cannot give any reasons why there was no activity recorded on certain days. Days 29 and 30 recorded the most activities while the rest of the recorded days show an average of fewer than 250 activities recorded. Day 10 recorded the least number of activities followed by days 24, 27, 8 and 3.

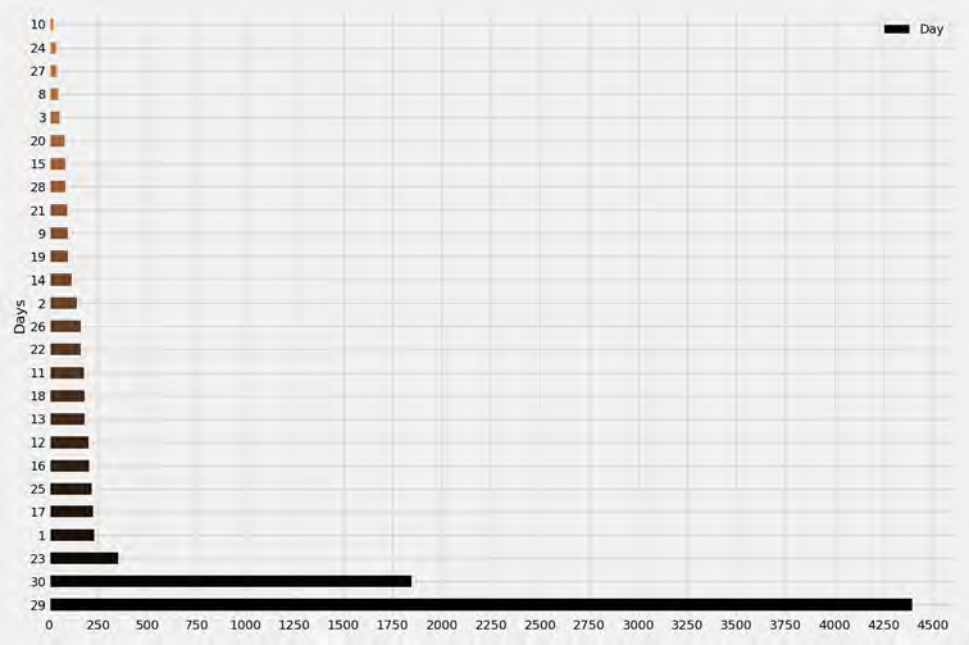
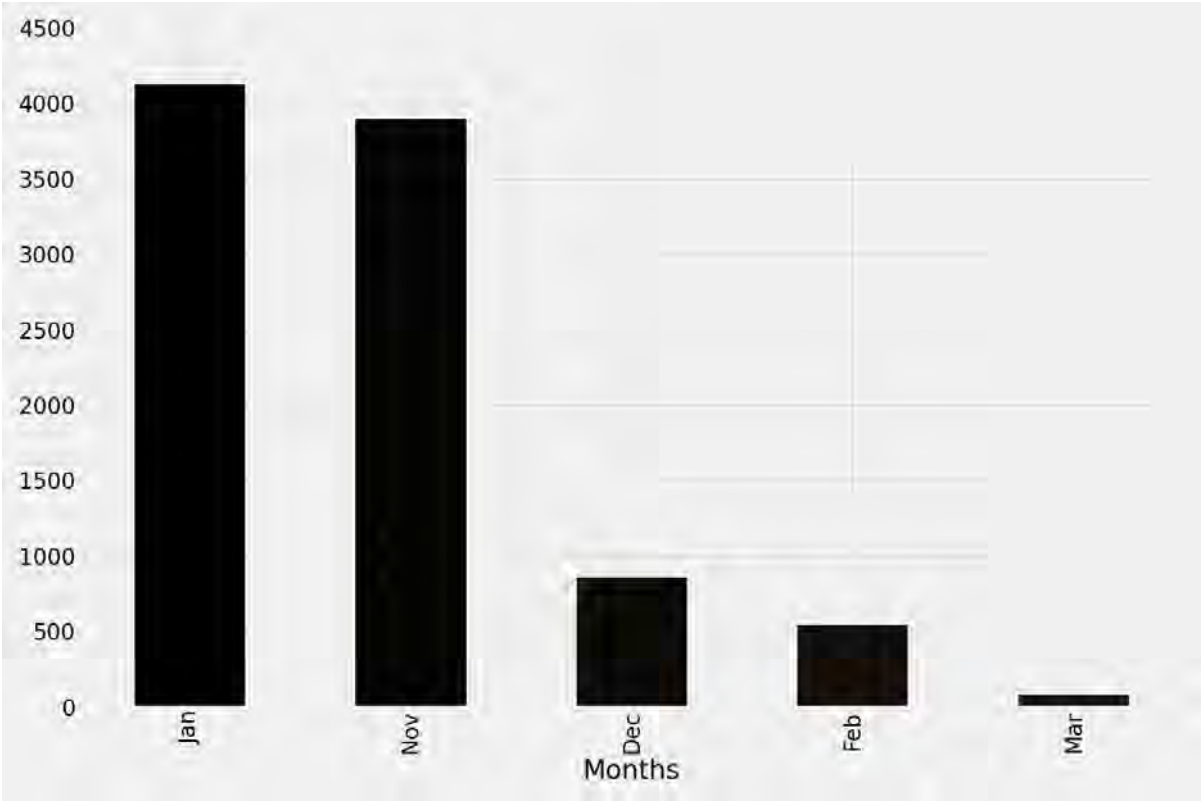


Figure 6.2: Days with the most frequent logins

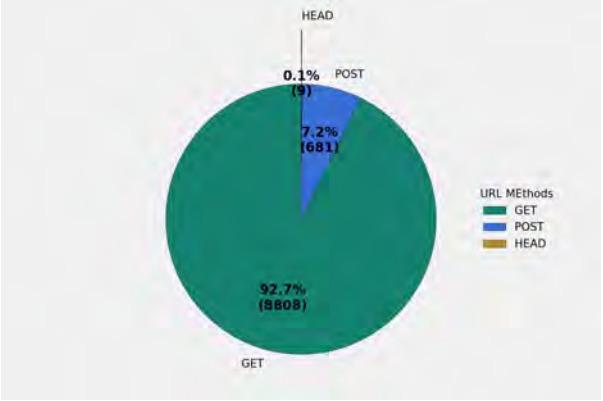
Frequency access to the website during the 5 months of data collection, November to March is shown in Figure 6.3a. January 2018 recorded the greatest number of activities with just over 4,000 activities, followed by November 2017 with just under 4,000 activities. The other three months recorded fewer than 1,000 activities each with March recording the smallest number, with approximately 100 activities.

Figure 6.3b shows that there were three URL methods used to access the web pages: GET was used 8,808 times, representing 92.7% of the data accesses, POST was used 681 times (7.2%) and HEAD accounted for 0.1% with 9 uses.

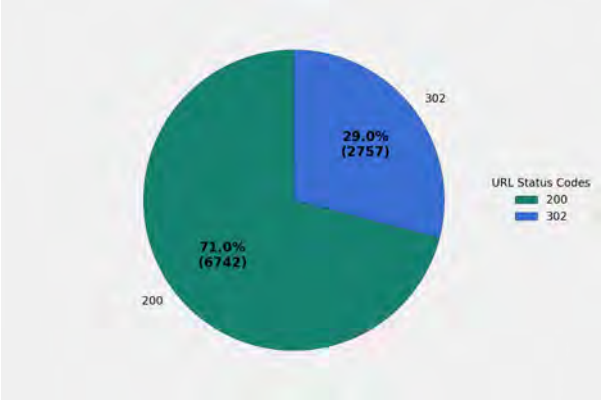
Figure 6.3c summarises the data in Table 6.8. Status code 200 was logged 6,742 times, representing 71% of the codes returned while status code 302 was logged 2,757 of the times, representing 29% of the codes returned.



(a) Most frequent months for user logins



(b) Most frequent Methods used



(c) Most frequent Status codes returned

Figure 6.3: General information obtained from the log data

6.6 Summary

The case study on log data answered the questions that were developed for the study. Thus, the case study achieved the objectives by using the SDA tool to assess the common IP addresses and URLs accessed per month and for the duration that the data was collected as well as the URLs and their relationship with the response status codes.

In executing the scenarios in the case study, the four main steps of a basic data analysis process were considered, namely data collection, data processing, data transformation, data analysis and results presentation. Data were obtained from a CSV file, transformed to a desired state, analysed and finally presented using a variety of formats including, tables, bar charts and pie charts, which offered different vantage points from which the information could be viewed and hopefully understood.

Chapter 7

A Case Study using Apple Stock Data

This chapter discusses the third and last case study for this research, using Apple Inc. stock data to refine and validate the record and playback feature macro which enhances the SDA tool. The first two sections discuss stocks in general and an overview of stock data including common stock analysis methods. The third section describes the design and implementation of data analysis scenarios for validation of the record and playback enhancement.

7.1 Understanding Stocks

A stock is a security that represents part of the ownership of an organization; stocks are also known as equity and units of stock are called shares (Hayes, 2020b). Stocks are created by organizations to raise funds for business operations in what is called a primary stock market. Stocks are traded by investors in what is called a secondary stock market and entitle the owner to a proportion of the organization's assets and profits equal to the number stocks owned.

Stocks are bought and sold predominantly on stock exchanges, mechanisms and infrastructure that facilitate buying and selling of stock, although there can be private sales as well and buyers and sellers make up the market (Hayes, 2020a). Examples of stock exchanges include the Malawi Stock Exchange (MSE), the Johannesburg Stock Exchange (JSE) and the New York Stock Exchange (NYSE). Stock markets are where individual and institutional investors come together to buy and sell shares in a public venue. Stock

prices are set initially by an evaluation of an organization's current state to come up with a present value which is used for an initial public offering (IPO) in a primary market. Thereafter, prices in the secondary market (are affected by the laws of supply and demand) as buyers and sellers interact.

The laws of supply and demand apply in stock transactions which require a buyer and a seller; if there are more buyers than sellers of a specific stock, the stock price will trend up and conversely, if there are more sellers than buyers of the stock , the price will trend down (Hayes, 2020a). The need to buy or sell stock is influenced by the perception of an organization's value, effects of an event to an organization or sector, performance of an organization and so on.

Stocks are the foundation of many individual investors' portfolios. A number of studies such as that by Berk and Van Binsbergen (2015) and McLean and Pontiff (2016), show that over long periods of time stock investments result in better returns compared to other asset types. Stock returns are due to capital gains (when a stock is sold at a greater price than when it was bought) and dividends (a share of a profit organizations give to shareholders). Stock traders aim to make the most value or money by trying to make good choices over time based on various aspects of the stock market. One way is to consider historical stock trading data for organization, analysing the data to learn from the information produced and making decisions based on past trends.

7.2 Overview of Stock Data

Xu and Berkely (2014) outline two types of analysis when dealing with stock data, technical analysis and fundamental analysis. Fundamental analysis involves a broader view by looking at underlying factors that affect an organizations business and its future prospects to determine its stock value, for example by taking into account the organization's market share, growth, revenue, return on equity etc. On the other hand, technical analysis considers historic price movements to predict future price movements. In this case study, we consider technical analysis for the validation of the record and playback feature.

A stock price is the point at which buyers and sellers meet. Any historical stock data normally includes the following attributes:

- Date, the day on which transactions occurred for the records.
- Open, representing the opening price, which is the stock price when the market opened on the day.

- High, representing the highest price, which is the highest stock price reached on the day.
- Low, representing the lowest price, which is the lowest stock price reached on the day.
- Close, denoting the closing price, which is the stock price when the market closed on the day.
- Volume, referring to the number of shares that were traded on the day.

7.3 Case Study Design

This section first discusses the Apple stock data used and the scenario design which outlines analysis scenarios selected for this case study. Thereafter, the process of data analysis using the record and playback macro feature is explained together with the results of its application. Three analysis scenarios were designed and carried out as listed below:

Table 7.1: Sample Apple stock data

Date	**Open**	**High**	**Low**	**Close**	**Volume**
1/3/16	11.63	11.795	11.601429	11.770357	445138400
1/4/16	11.872857	11.875	11.719643	11.831786	309080800
1/5/16	11.769643	11.940714	11.767858	11.928572	255519600
1/6/16	11.954286	11.973214	11.889286	11.918928	300428800
1/7/16	11.928214	12.0125	11.853572	12.004286	311931200
1/10/16	12.101071	12.258214	12.041785	12.230357	448560000
1/11/16	12.317142	12.32	12.123929	12.201428	444108000
1/12/16	12.258928	12.301071	12.214286	12.300714	302590400
1/13/16	12.327143	12.38	12.280357	12.345715	296780400
1/14/16	12.353214	12.445714	12.301429	12.445714	308840000
1/18/16	11.768572	12.312857	11.642858	12.166072	1880998000
1/19/16	12.441072	12.45	12.031428	12.101429	1135612800
1/20/16	12.015357	12.082143	11.79	11.881429	764789200
1/21/16	11.920357	11.96	11.665358	11.668571	754401200
1/24/16	11.673928	12.051785	11.668571	12.051785	574683200
1/25/16	12.011786	12.194285	11.948929	12.192857	546868000
1/26/16	12.248571	12.342857	12.196428	12.280357	506875600
1/27/16	12.277857	12.310357	12.243929	12.2575	285026000
1/28/16	12.291785	12.3	11.911786	12.003572	592057200

1. Analysing stock volumes traded from 1st January 2016 to 31st December 2020.
2. Analysing stock price trends over time between 1st January 2016 to 31st December 2020.
3. Analysing the high and low prices over time between 1st January 2016 to 31st December 2020.

A sample of the data is shown in Table 7.1. The data is spread over long periods of time and is ideal to highlight the record and playback feature through trend analysis of the individual years available in the data.

7.4 Steps in using The Record and Playback Macro Feature

This section highlights the general steps in using the record and playback macro feature.

1. Start up the SDA tool.
2. Enter interactive-mode
3. Choose option to use recording.
4. Carry out data analysis task.
5. End recording and exit SDA tool.
6. Review the log file and adjust parameters.
7. Start up the SDA tool.
8. Enter non-interactive mode
9. Input log file path.
10. Retrieve output information produced.

7.5 Data Analysis Scenario 1 - Analysing stock volumes traded over time

Stock volumes traded were analysed with a general view of year-on-year trends from 2016 to 2020. Months with lower or higher than usual trends were then analysed comparatively with corresponding months in other years to assess the differences.

Input - The Apple Inc. stock data described in Table 7.1 and user commands for executing this analysis scenario as outlined in the analysis steps given below.

Output - Graphs summarizing information on yearly stock volume trends are created from the analysis of specific months that had varied general trend.

7.5.1 Steps in Executing Scenario 1

1. Enter interactive-mode and initiate recording.
2. Load Apple stock data into the system.
3. View data and make decisions on the desired columns to select and check data types, nulls and duplicates and make corrections where necessary. Select date and volume columns.
4. Sample 5% of the data.
5. Format *Date* column to *dd-mmm-yy*.
6. Filter data for year(s) entered (2016).
7. Apply appropriate statistical measures for analysis.
    - From *Date* column, extract *month* and create *Month* column that indicates the month only.
    - Format *Volume* column to *integer*.
    - Repeat the steps above on data for each year(s).
8. Generate a graph showing yearly volumes against time.
9. Generate graphs on on specific periods by filtering data.
10. End recording and review the log (remove the sampling data step and add 2017, 2018, 2019 and 2020 to years entered).
11. Enter non-interactive mode and input log file path.
12. Retrieve output information, in this case graphs are selected.
13. Analyse the information produced.

7.5.2 Results and Discussion

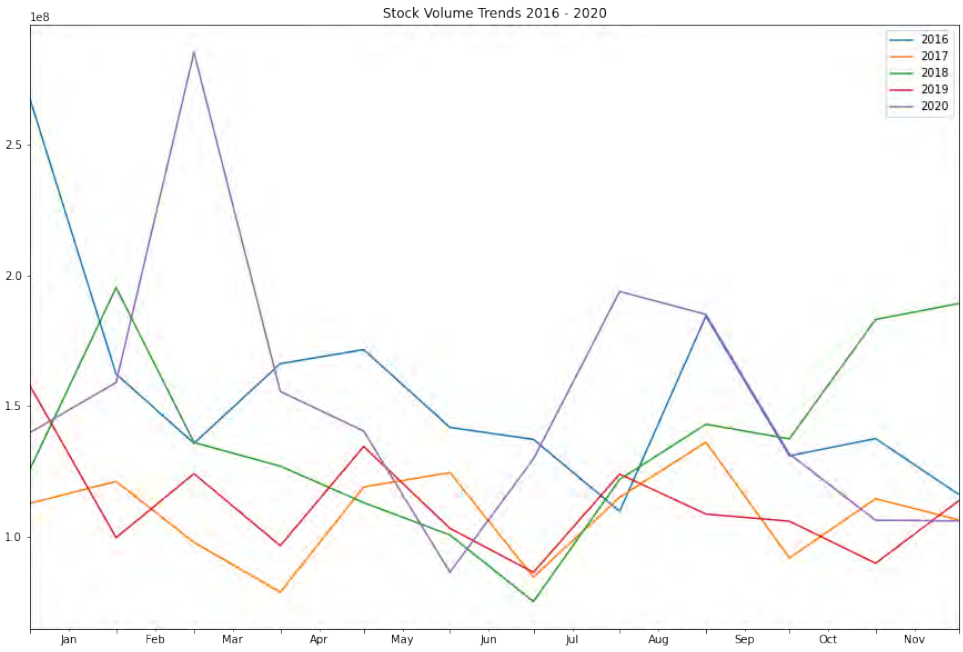
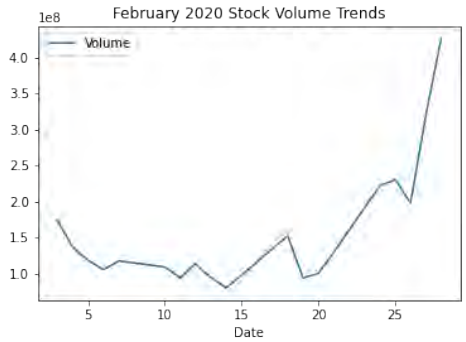


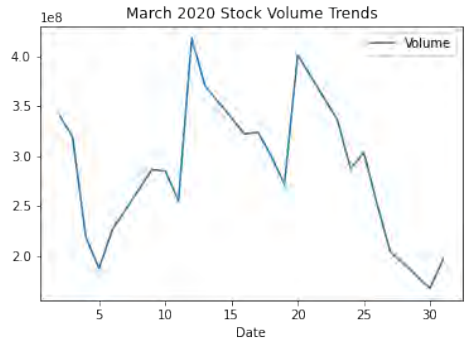
Figure 7.1: Stock volume trends 2016 - 2020

Considering Figure 7.1, which shows the trend of average stock volume traded, there is consistent movement over the years. Years 2017 and 2019 look consistent indicating relatively normal events that can affect stock in the course of the year, while 2016, 2018 and 2020 are volatile due to various events such as the first-ever iPhone year-over-year sales decline[1], potential slowdown in iPhone sales and a lack of innovation in other areas of Apple's business[2], and Covid-19 in 2020[3]. Some periods to note are January 2016, February 2018, February to March 2020, May to June 2020, June 2018, August to September 2020, September 2020 and November to December 2018.

[1]https://www.cnbc.com/2016/04/27/street-slashes-apple-price-targets-after-earnings-disappointment.html
[2]https://www.cnbc.com/2018/11/01/apple-shares-are-doing-something-unusual-ahead-of-earnings.html
[3]https://www.macrumors.com/2020/02/28/aapl-covid19-coronavirus/

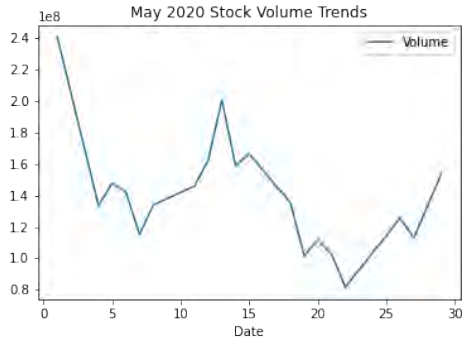


(a) February 2020 stock volume trends

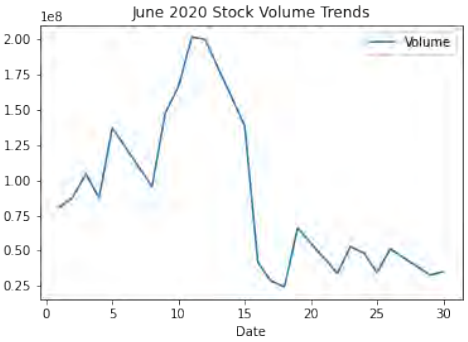


(b) March 2020 stock volume trends

Figure 7.2: High stock volume traded months



(a) May 2020 stock volume trends



(b) June 2020 stock volume trends

Figure 7.3: Low stock volume traded months

A drill down of the data for the year 2020 highlights some details. The drill down offered an opportunity to use the record and playback macro feature to carry out repetitive tasks such filtering data and plotting stock volume trends for each year and month. The highest number of shares traded in 2020 was recorded on 28th February with *426,510,000* and the lowest number on 27th November with *46,691,300*. February and March 2020 results are shown in Figure 7.2. February was a volatile month in terms of volume traded per day; the lowest volume traded in February was on the 14th, with *80,113,600* shares while the highest was on the 28th with *426,510,000* shares, a variance of *346,396,400*. In comparison Apple stock have an average of *77,670,000* shares traded over 10 days and *101,360,000* over 3 months. From this information we can see that there was an increase of demand in Apple Inc. stocks which attributed to an increase in the stock price.

Figure 7.3 shows the months with average low stock volumes traded and the volatility is still apparent and is most likely due to uncertainties of the first wave of COVID-19.

In this scenario, the record and playback macro feature eliminated the need for a user to

carry out a repetitive task, i.e. carrying out the same data analysis steps for each year and month in the data set.

7.6 Data Analysis Scenario 2 - Analysing stock prices over time

This scenario analyses the gains and losses of the stock price between 2016 and 2020.

Input - The Apple Inc. stock data described in Table 7.1 and user commands for executing this scenario as outlined in the analysis steps given below.

Output - A graph summarizing the information on yearly stock price trends created from the analysis of Apple stock data.

7.6.1 Steps in Executing Scenario 2

1. Load Apple stock data into the system.
2. View data and make decisions on the desired columns to select and check data types, nulls and duplicates and make corrections where necessary. Select date, open and close.
3. Format *Date* column to *dd-mmm-yy*.
4. From *Date* column, extract *month* and create *Month* column that indicates the month only.
5. Save data and go back to main menu.
6. Enter interactive mode and initiate recording
7. Load processed Apple stock data into the system.
8. Sample 5% of the data.
9. Filter data for years entered (2016).
10. Generate a graph showing yearly trend of *adjusted close* price.
11. End recording and review the log (remove the sampling data step and add 2017, 2018, 2019 and 2020 to years entered).
12. Enter non-interactive mode and input log file path.
13. Retrieve output information, in this case graphs are selected.
14. Analyse the information produced.

7.6.2 Results and Discussion

Stock prices are determined by two fundamentals, namely, an earning base (which provides earning power through dividends) and a valuation multiple (expression of expectations about the future) (Happer, 2021). A higher growth rate will earn stock a high valuation multiple and vice versa. The stock price trend shown in Figure 7.4 indicates that Apple Stock price has steadily increased from 26 USD a share in January 2016 to 135 USD a share in December 2020, thus an increase of 107 USD (412%) over five years. The volatility seen in 2016 and 2018 (see Figure 7.1) is not apparent in the stock price trend, however, for 2020 the volatility is apparent with a year low price of 55 USD and a year high of 135 USD.

Similar to scenario 1, the record and playback macro feature lessened the repetitive tasks to process yearly and monthly data. Note in this scenario the recording was started after processing the data, its application depends on how best a user assesses a situation.

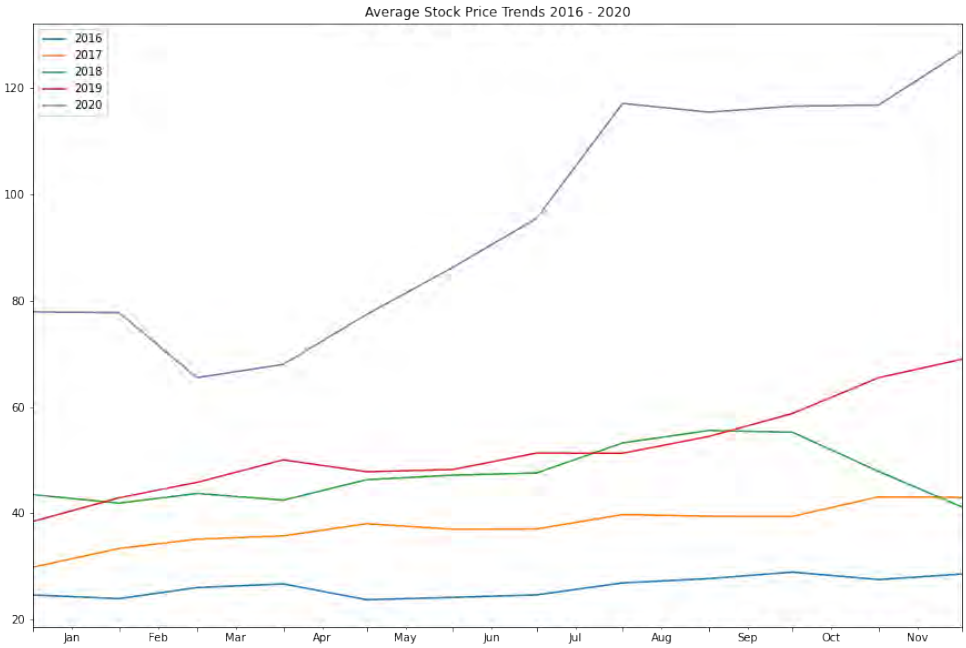


Figure 7.4: Average stock price trends 2016 - 2020

7.7 Data Analysis Scenario 3 - Analysing the high and low prices over time

This scenario analyses the movements between high and low prices in order to determine the volatility of the stocks from 2016 to 2020 relative to the volume of stock sold.

Input - The Apple Inc. stock data described in Table 7.1 and user commands for executing this scenario as outlined in the analysis steps given below.

Output - Graphs summarizing the information on yearly stock high and low price trends.

7.7.1 Steps in Executing Scenario 3

1. Enter interactive mode and initiate recording
2. Load Apple stock data into the system.
3. Create a random sample of 5% of the data.
4. View data and make decisions on the desired columns to select and check data types, nulls and duplicates and make corrections where necessary. Select date and volume columns.
5. Format *Date* column to *dd-mmm-yy*.
6. Filter data for each year from 2016 through 2020.
7. Generate a candlestick graph for the period 2016 to 2020 to obtain an overview of the trend in price movements.
8. Generate a candlestick graph for each year in the period 2016 to 2020 to obtain a detailed view of the price trend.
9. End recording and review the log (remove the sampling data step).
10. Enter non-interactive mode and input log file path.
11. Retrieve output information, in this case graphs are selected.
12. Analyse the information produced.

7.7.2 Results and Discussion

(a) 2016 - 2020 High & Low Price Trend

(b) 2016 52-Week High/Low Analysis

(c) 2017 52-Week High/Low Analysis

(d) 2018 52-Week High/Low Analysis

(e) 2019 52-Week High/Low Analysis

(f) 2020 52-Week High/Low Analysis

Figure 7.5: Scenario 3 Output: 2016 - 2020 High & Low Price Analysis

Figure 7.5 shows candlestick charts that are used by traders to determine possible price movement based on past patterns (Hayes, 2022). Candlesticks are useful when trading as they show four price points (open, close, high, and low) throughout the period of time the trader specifies. Figure 7.5 has six sub-figures.

Figure 7.5a shows price analysis from January 2016 to Decemebr 2020 so as to establish a

general trend of prices over the period and the trend is a steady positive climb in Applie Inc. stock price. The stock price opened at USD 25.65 on 4th of January 2016 and closing at USD 133.72 on 30th of December 2020, an increase of 421.32%.

Figures 7.5b through 7.5f drill down on information presented in Figure 7.5a. Each of the detailed figures shows a different trend for one of the years between 2016 and 2020. A *52-week high/low* is a technical indicator used by traders and investors to analyse stock's current value and as a predictor of its future price movement (Chen, 2021). An investor may show increased interest in a stock as its price nears either the high (to sale and make profit) or the low (to buy stock) end of its 52-week price range.

Note that the application of the record and playback macro feature in this scenario showed scaling of a dataset set, i.e. from 5% to 100% unlike the first two scenarios that were changing parameters to carry out repetitive tasks.

7.8 Summary

The case study using Apple Inc. stock data addressed the objectives that were developed for the case study. Thus, the case study achieved the objectives by using the enhanced SDA tool for technical analysis of historical stock data.

In executing the scenarios in the Apple stock case study, the SDA tool was further improved and validated and specifically made use of the record and playback feature in order to validate the enhancement functionality and provide technical computing information to assess the efficiency. The enhancement has two aspects, one aspect provides an option to carry out a repetitive task and this was validated and assessed using scenarios 1 and 2. The other aspect simulated using scenario 3 is that of carrying out analysis of tasks with large amounts of data by recording analysis steps on a fraction of the dataset and then using the recorded steps to carry out analysis on the full dataset.

Chapter 8

Subjective Analysis of the Software Tool

The study set out to develop a simplified data analysis tool that supported the data analysis process in such a way as to save user's time and computing resources in resource constrained environments such as those operated by individuals or small businesses.The SDA tool was developed and tested through a repetitive process of code improvement and SDA tool refinement. The outcome was an enhanced data analysis tool consisting of the core elements of the data analysis process (data collection, data transformation, data analysis and data visualization) and the enhancement element (the record and playback feature macro).

Three case studies were carried out, conducting a variety of data analysis tasks in order to provide a proof of concept. Two case studies, one on network data and another on log data validated the SDA tool's functionality. In these case studies, data was retrieved from a file and stored on the SDA tool's database, transformed then some statistical measures applied before the results were presented using charts or table formats. The enhancement element was validated using a third case study that used Apple Inc. stock data spread over five years, the record and playback feature used a small dataset to record steps and run on a larger dataset and recursive analysis tasks were used to test the enhancement element. The outcomes of each study and task was presented and discussed at the end of each task in the case studies.

This chapter presents a consolidated analysis of the enhanced SDA tool based on the results of the three case studies and provides a discussion of the tool's strengths and weaknesses.

8.1 Analysis of the Tool Performance

This section analyses the performance of the SDA tool using five metrics, namely, functionality, usability, scalability, parallelization and the efficiency of the SDA tool. The tests were carried out using a laptop with standard specifications; Intel Core i5-8260U Processor, 8 GB RAM and a 256 GB Solid State Drive (PCI Express 3.1a x4 NVMe).

8.1.1 Functionality

The Oxford Advanced Learner's Dictionary[1] defines functionality as the range of functions that a computer or other electronic system can perform. For the enhanced SDA tool a number of functions were designed and implemented. The functions were grouped into five main groups, namely, data collection, data transformation, data analysis, data visualization and the record and playback feature.

The dynamic nature of data analysis has led to an extensive set of functionality being developed to meet varying requirements across industries. Not all of this functionality was fully implemented in this research as the SDA tool was designed purely as proof of concept.

Data Collection

This component performed as intended based on the data analysis scenarios that were carried out. The reading function can read from flat files and databases, then copying the data into memory for the SDA tool to use. In all three case studies, the source data was copied from flat files (CSV and Microsoft Excel), however, reading from a database was tested when data was retrieved from the internal database to system memory for use. Note that users can save progress of their data analysis task at any point, so that they can continue at a later time where the system copies the stored data.

A feature available when copying data is that of data preprocessing. The data collection component interfaces with the data transformation component to allow for data preprocessing, thus the functionality in the data transformation component is made available after data is copied from source but before it is committed to the system database.

Another functionality within the data collection component is that of writing data, which also performed as intended. Data was written to the system database and flat files such

[1] https://www.oxfordlearnersdictionaries.com/definition/english/functionality?q=functionality

as PDF and CSV. This function makes use of multi-threading when writing data from memory to storage devices so that a data analysis task can proceed using data in memory at the same time.

Data Transformations

The data transformation component provides functionality for processing of data and has three main functions as discussed in Chapter 3. These functions performed as designed and were extensively tested in the case study using log data, (see Chapter 6).

The first functionality in this component allows users to view data in order to make decisions on how to carry out a data analysis task. A user can view all or part of the data, as well as summary statistics which show the count, unique value, the most frequent value and the frequency of the most frequent value for all columns in the data. An example is shown in Table 6.2. For numeric and date and time data, the summary statistics show the count, mean, minimum, maximum and 25th, 50th and 75th percentiles for all columns in the data. A user can also view the structure of the data, which shows the data types of columns in the data. User can sort, filter, group data and save data.

The second functionality provides functions to clean data. There are four functions that were tested. First, the ability to process data with null values in rows, columns and cells of a dataset. The null values can be filled with correct data or place-holders or the null records can be removed. The second function is to process duplicate data by identifying and providing means to correct the data either by removing the record or updating the record. A whole record or part of it can be a duplicate. The third function is to process inaccurate values; these values are normally identified when the data is viewed and the inaccurate data can be updated or removed. The fourth function is to format data to the correct type. Pandas formats data depending on data available in the columns; if there are inaccurate values in the columns, i.e. the column has different data types then in most cases they will be formatted as *string*. A normal solution to this is to work on the inaccurate values and then use this function to format the data as desired, for example to date or number format.

The third functionality in the transformations component is to maintain data. The normal approach is to view data and gain an understanding about it, and then if it is required for the data to be updated this component can be used. Three options are provided. The first is to update column names, the second is to update data values and the third is to add or remove columns and rows.

Data Analysis

The data analysis component provides mathematical and statistical methods such as arithmetic functions, measures of central tendency, measures of dispersions and other miscellaneous functions that aid data analysis. This component was tested in all three case studies and functioned as intended.

The first group of functions is *Manage data*, this provides options to sort and filter data using the same system functions in the data transformations. Also included in this group is the sampling capability. Sampling was used to show all the sample data given in the relevant scenario sections in Chapters 5 through 7.

Another group of functions includes statistical functions that were tested using the case studies, namely *mean*, *sum*, *count* and basic math operations.

Data Visualization

This component provided two options for visualizing data, namely, charts and tables. The visualizations can be in stored as image files (JPG) or PDF files. All the case studies produced visualizations and thus the functionality was determined to meet the design objectives.

The Record and Playback Feature

Table 8.1: Record and playback execution analysis

Number of Runs	2	5	7	9	11
Interactive Mode Duration (Seconds)	110	182	226	225	237
Non-Interctive Mode Duration (Seconds)	58	84	77	75	79
Variance	52	98	149	150	158
% Variance	47%	54%	66%	67%	67%
% Change		7%	12%	1%	0%
Change in Interactive Mode (Second)		72	44	-1	12
Change in Non-Interactive Mode (Seconds)		26	-8	-2	4

This component was tested in the case study using Apple Inc. stock data in Chapter 7. Using scenarios 1 (Section 7.5) and 2 (Section 7.6) recursive analysis tasks were carried out: analysis steps for 2016 were recorded, the steps were then used to run similar analysis on years 2017 through 2020. Using Scenario 3 (Section 7.7) a random sample dataset representing 5% of the original dataset was used to record the analysis steps, applied to analysis of low and high prices over time. The recorded steps were then used to analyse the whole dataset using the playback feature.

In order to assess the time saving and benefit of the record and playback feature, the times of execution for Scenario 1 and 2 are summarized in Table 8.1. The table shows the number of iterative processes carried out, the time it took to execute the processes using the record and playback feature and without using it, the variance of time and the percentage of variance.

Using Scenario 3, there are savings of time when carrying out data analysis using the record and playback feature by recording steps using a subset of data then running the data analysis task using the recorded steps on the whole dataset. Apple Inc. stock data from 2016 to 2020 has 1,258 records, a random sample of 5% of the data (63 records) was used to record steps. After recording, the steps were used to run analysis on the whole dataset.

8.1.2 Usability

Usability is the ability to use software products (Weichbroth, 2020). This is an important aspect when designing software products for the software to be used as intended. A popular standardized questionnaire for the assessment of perceived usability know as the System Usability Scale (SUS) accounts for 43% of post-test questionnaire usage (Lewis and Sauro, 2017). The SUS is shown in Figure 8.1. Several aspects of the design implementation of the SDA tool are aimed specifically at ease of usability, for example the menu driven approach where a user is only displayed options depending on the stage of analysis (this limits errors or difficulties in choice of next action to be taken), and removing as many complexities as possible to make it easy to learn.Users should learn how to use the system easily without computer technical skills and knowledge and the system should be attractive to users with respect to the functions as well.

	The System Usability Scale Standard Version	Strongly Disagree	1	2	3	4	5	Strongly Agree
1	I think that I would like to use this system frequently.		o	o	o	o	o	
2	I found the system unnecessarily complex.		o	o	o	o	o	
3	I thought the system was easy to use.		o	o	o	o	o	
4	I think that I would need the support of a technical person to be able to use this system.		o	o	o	o	o	
5	I found the various functions in this system were well integrated.		o	o	o	o	o	
6	I thought there was too much inconsistency in this system.		o	o	o	o	o	
7	I would imagine that most people would learn to use this system very quickly.		o	o	o	o	o	
8	I found the system very awkward to use.		o	o	o	o	o	
9	I felt very confident using the system.		o	o	o	o	o	
10	I needed to learn a lot of things before I could get going with this system.		o	o	o	o	o	

Figure 8.1: The standard System Usability Scale taken from Lewis and Sauro (2017)

In addition to the SUS three factors were also considered to as usability,these are:

- Success rate, whether or not an analysis process executed successfully i.e. output achieved.
- Error rate, system errors encountered during an analysis process.
- Time taken for the system to execute a task.

8.1.3 Scalability

Scalability is the ability of a system to handle increased workload by continuously applying a cost-effective strategy for extending a system's capacity (Weinstock and Goodenough, 2006). This capability was tested using case study using network data when network data was preprocessed. The original data was 1.7 GB and after processing the data was shrunk to 400 MB. The processing included retrieving the data from source and loading it into memory, filtering data and saving data to the system database.

It must be noted that the limits of scalability were not tested.

8.1.4 Parallelization

Parallelization in computing can be achieved by multi-processing of tasks and data using two or more processors (Tejedor *et al.*, 2017). This feature enhances the SDA tool's productivity by allowing more work to be done in a period of time, which positively affects the usability from a user's perspective of efficiency. The implementation of parallelization in the SDA tool is to make use of processors available in an environment by utilizing one processor for one data analysis task being carried out. Parallelization is only possible when carrying out two or more separate data analysis tasks.

The results found that this features allows for more work to be done in a period of time depending on computing resources available on a computing environment. The purpose of this is not to process a task more quickly but rather to process more tasks within a fixed period of time. From the tests we found that the resources that need to be considered when implementing this feature are the CPU, memory and storage. The CPU needs to be able to handle a number of tasks and there must be a multiprocessor. The SDA tool uses in-memory processing and thus adequate memory is required. Lastly, storage complements the memory and stores data in the database, hence this component requires consideration when using the SDA tool. Consideration of the aforementioned computing resources is for better operation, however, the minimum specifications given in Section 3.2 do suffice and provide optimal operations but with more limitations on the size of data able to be analysed and the number of analysis tasks that can be run concurrently.

8.1.5 Efficiency of the Tool Implementation

As discussed earlier, Python is a powerful programming language and is widely used in many areas, more so in data analysis. The Pandas library is widely used for handling various sizes of data efficiently and effectively. As the SDA tool includes use of the Pandas library, all the power and performance offered by Python and Pandas is available to the users of the SDA tool, but without the need to learn any programming themselves.

The efficiency of the SDA tool was also tested by carrying out data analysis tasks on low end hardware resources. This was shown in case study using network data. Data was processed using both a base model computer with specifications as shown in Figure 3.2, specifically the processor (duo core with 1.9 GHz), memory of 8 GB and storage having at least 1 GB free and a higher end model with typical specifications[2] of a quad core processor, 16 GB of memory and 512 GB of storage.

[2]https://www.choosewhat.com/how-to/select-the-best-small-business-computer

The performance on both computers was acceptable and the outcome was the same, however, there were notable performance differences. The base model computer was slower due to memory and storage, size and speed which affected the rate at which data was copied and stored. In this particular case in terms of storage, the base model computer had a hard disk Drive (HDD) while the top model computer had a solid state drive (SSD). HDDs have significantly lower read (55 MB/s) and write (30 MB/s) speeds as compared to SSDs read (3,000 MB/s) and write (2,000 MB/s) speeds[3] and these differences were observed.

8.2 Tool Strengths and Weaknesses

The SDA tool was developed by learning and analysing parts of data science and various data analysis tools available. Thus, in a resource limited environment context, it aimed to improve on existing processes and tools were compared with Excel. For example, the SDA tool has the record and playback feature similar to Excel macros, which addresses a limitation in data analysis. However the implementation in the SDA tool is different. The record and playback feature is scalable as it handles repetitive tasks and data analysis tasks with different sized datasets. The SDA tool makes use of parallelization to enhance the record and playback macro feature by making use of available processors in the computer environment. Another difference is that Python has a range of libraries that offer various data analysis capabilities such as Apache Spark that can handle more data types other than alpha-numeric and is efficiently designed for big data processing and machine learning. This means that the tool has potential capabilities that could be implemented to improve and adapt it to various environments. The SDA tool makes learning and use easier as it provides contextual menus during data analysis process.

From this background various strengths and weaknesses of the SDA tool are discussed below.

Strengths

- The tool is simple to use and effective in its capability.
- The menu-driven approach and contextualized menus provide users with appropriate options at each stage of an analysis process.
- Ability to process alphanumerics data types of various sizes.

[3]https://www.avast.com/c-ssd-vs-hdd

- Ability to carry out analysis using recorded steps while user does other tasks.
- Ability to carry out repetitive analysis tasks using recorded steps while user does other tasks.
- Ability to adapt recorded steps to changes.

Weaknesses

- The current limit in scope which generally affects areas where the SDA tool can handle.
- Limited in big data processing.
- Limited in real-time or near real-time data processing.

8.3 Summary

This chapter analysed and discussed the SDA tool's performance and its strengths and weaknesses using the three case studies that were carried out. Five areas were assessed and discussed, namely the tool's functionality, usability, scalability, parallelization and efficiency.

The tool is functional and able to carry out analysis tasks not requiring technical programming skills and knowledge but with only basic data analysis knowledge. The tool is usable, with its menu-driven approach giving available options to a user at each step of an analysis task. The tool is scalable both in workload and hardware. The parallelization designed allows for multiple analysis tasks to be carried out at a time. The efficiency in the SDA tool is as a result of the power of Python which has a range of libraries with efficient and effective data analysis capabilities.

Lastly the tool has a number of strengths that makes it suitable for use in a resource limited environment.

Chapter 9

Conclusion

Information is essential in decision making as it provides opportunity and a basis for effective choices and possibly favourable and intended outcomes. To gain information, data has to be collected and analysed; an objective and effective analysis process allows for accurate information. The implementation of the analysis process has been improved with time to meet the constantly changing demands on data requirements for various industries which has created opportunities in various contexts such as that of individuals and small businesses where resources are more limited larger businesses. This provides room for enhancement of the data analysis process to address some challenges in such resource limited environments. In this thesis a simplified data analysis tool was developed using Python. The tool is a proof of concept for development of analysis tools that meet the needs of such environments.

9.1 Summary of Work Done

The work that was carried out in this research was categorized into four phases introduced in Section 1.4. For the first phase various literature was reviewed to gain understanding of the data analysis process and various analytical tools/techniques available. From the understanding acquired it was determined that an enhanced simplified data analysis tool was feasible for resource limited environments such as those of individuals and small businesses. A variety of existing tools such as Python, R programming, Microsoft Excel, Apache Spark and Tableau and techniques suggested by Cuesta (2013) and Bāliņa *et al.* (2016) were reviewed.

In the second phase weaknesses such as high cost of acquiring a data analysis tool and knowledge and risks such as sustainability of such a solution were identified and impact

assessed. Through research of how various industries that use data analysis tools, the impact of such a solution was verified.

During the third phase of the research, the enhanced SDA tool was developed and validated using a variety of available datasets. The SDA tool was refined to address some of the weaknesses and risks identified in the second phase. The datasets used in this research were aimed at testing and validating a generic design for the SDA tool.

Finally, from the information gathered, enhancements were developed to the SDA tool. These included a record and playback macro feature and the incorporation of parallelization that improved overall efficiency and effectiveness of the SDA tool. This resulted in a simplified data analysis tool applicable to a broad spectrum of users with small tweaks and changes to conform to specific environments and save users time in carrying out data analysis.

9.2 Achievement of Objectives

The main objective of this research was to develop a simplified generic data analysis tool and techniques that can be manually applied to a small subset of data of various types and then rerun on the full dataset using the automated playback enhancement thereby saving a user time and computing resources. This is approached through three sub-objectives as discussed below.

The first sub-objective required that the SDA tool should be able to work with a variety of data types, although limited to alpha-numeric data this objective was achieved and this is illustrated by the three different case studies carried out (Chapters 5-7). The case studies used data from different industries and of different formats. Data used in the the first case study, was computer network data which was relatively large (2 GB) when considering the needs of individuals and small businesses. The data had 87 columns and just under 3.5 million records. The data for the second case study was log data which consisted of 4 columns and about 16,000 records, relatively small. Finally the third case study used stock data that had 7 columns and about 1,800 records. The SDA tool was able to correctly analyse these datasets.

The second sub-objective required a determination on the effect of the record and playback macro feature on the usability of the SDA tool. The determination was that the record and playback macro feature affected the usability positively. The feature reduced the interaction with the user on long and/or repetitive tasks, saving users time which can be spent on other tasks.

The third and final sub-objective was to determine the extent parallelization can enhance the data analysis process. Whilst it is difficult to quantify exactly what savings of time can result from using the record and playback macro feature, it was shown that using non-interactive mode is faster than using the interactive mode of the SDA tool.

9.3 Contributions of the Research

In this thesis, we have shown an enhanced SDA tool that is able to process a variety of data from various industries efficiently and effectively by applying the data analysis process using a record and playback macro feature. A data analysis model can be created by recording steps using a sample of data, the steps can then be refined and applied on a full dataset to carry out analysis tasks even those that are repetitive by configuring variables. Further enhancements include that of a menu driven design that limits options available at different steps of the data analysis process to reduce chances of users choosing incorrect actions and making the analysis process and SDA tool easy to learn. Through use of this tool, individuals and small businesses would have access to the analysis power of Python and Pandas with a very minor learning curve.

9.4 Future Work

There is potential to improve the SDA tool with features such as Apache Spark, an enhanced version of Pandas that works with more data types aside from alpha-numeric data and big data in general. Other features that can enhance such type of tools are the inclusion of machine learning and artificial intelligence techniques. Further investigations into the development of in-memory processing and efficient and effective input/output processing as well as the handling of big data in real-time needs to be done. With these developments, the SDA tool could be used to simplify the data analysis process and make it available to many industries and a wider range of users while still being easy-to-use.

References

Abbasi, M., Shahraki, A., and Taherkordi, A. Deep learning for network traffic monitoring and analysis (ntma): A survey. *Computer Communications*, 170:19–41, 2021. doi:10.1016/j.comcom.2021.01.021.

Abgaz, Y., Dorn, A., Piringer, B., Wandl-Vogt, E., and Way, A. A semantic model for traditional data collection questionnaires enabling cultural analysis. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan*, pages 7–12. 2018.

Alotaibi, A. M., Alrashidi, B. F., Naz, S., and Parveen, Z. Security issues in protocols of TCP/IP model at layers level. *International Journal of Computer Networks and Communications Security*, 5(5):96, 2017.

Alpern, N. J. and Shimonski, R. J. Network Management. In **Alpern, N. J. and Shimonski, R. J.**, editors, *Eleventh Hour Network+*, pages 143 – 154. Syngress, Boston, 2010. doi:10.1016/B978-1-59749-428-1.00010-2.

Azeroual, O., Saake, G., and Schallehn, E. Analyzing data quality issues in research information systems via data profiling. *International Journal of Information Management*, 41:50 – 56, 2018. doi:10.1016/j.ijinfomgt.2018.02.007.

Bāliņa, S., Žuka, R., and Krasts, J. Opportunities for the use of business data analysis technologies. *Economics and Business*, 28(1):20–25, 2016.

Berk, J. B. and Van Binsbergen, J. H. Measuring skill in the mutual fund industry. *Journal of Financial Economics*, 118(1):1–20, 2015.

Bezerra, M. A., Lemos, V. A., de Oliveira, D. M., Novaes, C. G., Barreto, J. A., Alves, J. P. S., da Mata Cerqueira, U. M. F., dos Santos, Q. O., and Araújo, S. A. Automation of continuous flow analysis systems – a review. *Microchemical Journal*, 155:104731, 2020. doi:10.1016/j.microc.2020.104731.

Bikakis, N. Big data visualization tools. *arXiv preprint arXiv:1801.08336*, 2018.

Blei, D. M. and Smyth, P. Science and data science. *Proceedings of the National Academy of Sciences*, 114(33):8689–8692, 2017.

Brown, N. Listen to your gut: A reflexive approach to data analysis. *The Qualitative Report*, 24(13):31–43, 2019.

Bush, T. What Is Log Analysis? (How-To, Techniques, and Tools). 2020. Accessed on 9 March 2020.
URL `https://pestleanalysis.com/log-analysis/`

Callaghan, P. and Bee, P. Quantitative data analysis. In *A research handbook for patient and public involvement researchers*. Manchester University Press, 2018.

Cao, L. Data science: a comprehensive overview. *ACM Computing Surveys (CSUR)*, 50(3):43, 2017a.

Cao, L. Data science: Challenges and directions. *Commun. ACM*, 60(8):59–68, July 2017b. doi:10.1145/3015456.

Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., and Tzoumas, K. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4), 2015.

Castleberry, A. and Nolen, A. Thematic analysis of qualitative research data: Is it as easy as it sounds? *Currents in Pharmacy Teaching and Learning*, 10(6):807–815, 2018.

Chen, J. 52-Week High/Low. 2021. Accessed on 14 March 2022.
URL `https://www.investopedia.com/terms/1/52weekhighlow.asp`

Cuesta, H. Practical Data Analysis. Packt Publishing Ltd, 2013.

Delen, D. and Demirkan, H. Data, information and analytics as services. *Decision Support Systems*, 55(1):359 – 363, 2013. doi:10.1016/j.dss.2012.05.044.

Desiderio, D. Data analysis techniques for enhancing the performance of Customs. *World Customs Journal*, page 17, 2019.

Donoho, D. 50 years of data science. *Journal of Computational and Graphical Statistics*, 26(4):745–766, 2017.

Du, M., Li, F., Zheng, G., and Srikumar, V. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1285–1298. Association for Computing Machinery, New York, NY, USA, 2017. doi: 10.1145/3133956.3134015.

Duan, Y., Cao, G., and Edwards, J. S. Understanding the impact of business analytics on innovation. *European Journal of Operational Research*, 281(3):673–686, 2020.

Dutt, I., Borah, S., and Maitra, I. K. Multiple immune-based approaches for network traffic analysis. *Procedia Computer Science*, 167:2111–2123, 2020. doi:10.1016/j.procs.2020.03.259.

Elshawi, R., Sakr, S., Talia, D., and Trunfio, P. Big data systems meet machine learning challenges: Towards big data science as a service. *Big Data Research*, 14:1 – 11, 2018. doi:10.1016/j.bdr.2018.04.004.

Fay, R., Bland, J., and Jones, S. Next generation monitoring: Tier 2 experience. In *J. Phys. Conf. Ser.*, volume 898, page 092035. 2017.

Formoso, V., Cacheda, F., Carneiro, V., and Valiño, J. Open source tool for network monitoring. *International Journal of Business Data Communications and Networking (IJBDCN)*, 5(1):22–39, 2009.

García, D. L., Nebot, À., and Vellido, A. Intelligent data analysis approaches to churn as a business problem: a survey. *Knowledge and Information Systems*, 51(3):719–774, 2017.

George, G., Osinga, E. C., Lavie, D., and Scott, B. A. Big data and data science methods for management research. 2016.

Gottesdiener, E. Use Cases: best practices. *Rational Software white paper*, 2003.

Grus, J. Data Science from Scratch: First Principles with Python. O'Reilly Media, 2019.

Haas, J. and Lorcher, W. Analyst companies publish big data & information security survey. 2016. Accessed on 26 March 2019.
URL `http://barc-research.com/analyst-companies-publish-survey-big-data-information-security`

Happer, D. R. Forces That Move Stock Prices. 2021. Accessed on 27 February 2022.
URL `https://www.investopedia.com/articles/basics/04/100804.asp`

Hayes, A. How Does the Stock Market Work? 2020a. Accessed on 5 January 2021.
URL `https://www.investopedia.com/articles/investing/082614/how-stock-market-works.asp`

Hayes, A. Stock. 2020b. Accessed on 5 January 2021.
URL `https://www.investopedia.com/terms/s/stock.asp`

Hayes, A. Introduction to Technical Analysis Price Patterns. 2022. Accessed on 14 March 2022.
URL `https://www.investopedia.com/articles/technical/112601.asp`

Hernantes, J., Gallardo, G., and Serrano, N. IT infrastructure-monitoring tools. *IEEE Software*, 32(4):88–93, July 2015. doi:10.1109/MS.2015.96.

Hinde, D. PRINCE2 Study Guide: 2017 Update. John Wiley & Sons, 2018.

Hoelscher, J. and Mortimer, A. Using Tableau to visualize data and drive decision-making. *Journal of Accounting Education*, 44:49–59, 2018.

Hoplaros, D., Tari, Z., and Khalil, I. Data summarization for network traffic monitoring. *Journal of Network and Computer Applications*, 37:194–205, 2014.

Jagadish, H. Big data and science: Myths and reality. *Big Data Research*, 2(2):49–52, 2015.

Kang, W., Sim, B., Kim, J., Paik, E., and Lee, Y. A Network Monitoring Tool for CCN. In *2012 World Telecommunications Congress*, pages 1–3. March 2012.

Khot, T. Parallelization in Python. *XRDS: Crossroads, The ACM Magazine for Students*, 23(3):56–58, 2017.

Kreps, J. The log: What every software engineer should know about real-time data's unifying abstraction. 2013. Accessed on 16 February 2021.
URL `https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying`

Kumari, A., Tanwar, S., Tyagi, S., Kumar, N., Parizi, R. M., and Choo, K.-K. R. Fog data analytics: A taxonomy and process model. *Journal of Network and Computer Applications*, 128:90 – 104, 2019. doi:10.1016/j.jnca.2018.12.013.

Kurose, J. and Ross, K. Computer Networking: A Top-Down Approach. Pearson Education, Limited, 2010. ISBN 9780131365483.

LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., and Kruschwitz, N. Big data, analytics and the path from insights to value. *MIT Sloan Management Review*, 52(2):21, 2011.

Lewis, J. J. R. and Sauro, J. Revisiting the factor structure of the system usability scale. *Journal of Usability Studies*, 12(4), 2017.

Lin, I.-C. and Liao, T.-C. A survey of blockchain security issues and challenges. *IJ Network Security*, 19(5):653–659, 2017.

Losarwar, V. and Joshi, D. M. Data preprocessing in web usage mining. In *International Conference on Artificial Intelligence and Embedded Systems (ICAIES'2012) July*, pages 15–16. 2012.

Maxwell, D., Norton, H., and Wu, J. The data science opportunity: crafting a holistic strategy. *Journal of Library Administration*, 58(2):111–127, 2018.

McLean, R. D. and Pontiff, J. Does academic research destroy stock return predictability? *The Journal of Finance*, 71(1):5–32, 2016.

Molina-Solana, M., Ros, M., Ruiz, M. D., Gómez-Romero, J., and Martín-Bautista, M. J. Data science for building energy management: A review. *Renewable and Sustainable Energy Reviews*, 70:598–609, 2017.

Monahan, D. EMA Research Report: Data-Driven Security Reloaded. Technical report, Colorado: Enterprise Management Associates, Inc, 2015.

Morley, J., Widdicks, K., and Hazas, M. Digitalisation, energy and data demand: The impact of Internet traffic on overall and peak electricity consumption. *Energy Research & Social Science*, 38:128–137, 2018.

Munk, M., Kapusta, J., and Švec, P. Data preprocessing evaluation for web log mining: reconstruction of activities of a web visitor. *Procedia Computer Science*, 1(1):2273–2280, 2010.

Needham, T. C. Excel 2016: A Comprehensive Beginners Guide to Microsoft Excel 2016. Independently published, 2018. ISBN 978-1-944684-08-2 978-1-71805-796-8.

Oliver-Balsalobre, P., Toril, M., Luna-Ramírez, S., and Garaluz, R. G. A system testbed for modeling encrypted video-streaming service performance indicators based on TCP/IP metrics. *EURASIP Journal on Wireless Communications and Networking*, 2017(1):213, 2017.

Oussous, A., Benjelloun, F.-Z., Lahcen, A. A., and Belfkih, S. Big Data technologies: A survey. *Journal of King Saud University-Computer and Information Sciences*, 30(4):431–448, 2018.

Paakkonen, P. and Pakkala, D. Reference architecture and classification of technologies, products and services for big data systems. *Big Data Research*, 2(4):166 – 186, 2015. doi:10.1016/j.bdr.2015.01.001.

Pandas Development Team. pandas-dev/pandas: Pandas. February 2020. doi:10.5281/zenodo.3509134.

Power, D. J. Data science: supporting decision-making. *Journal of Decision Systems*, 25(4):345–356, 2016. doi:10.1080/12460125.2016.1171610.

Qiu, T., Ge, Z., Pei, D., Wang, J., and Xu, J. What happened in my network: Mining network events from router syslogs. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 472–484. ACM, New York, NY, USA, 2010. doi:10.1145/1879141.1879202.

Queiroz, W., Capretz, M. A., and Dantas, M. An approach for SDN traffic monitoring based on big data techniques. *Journal of Network and Computer Applications*, 131:28–39, 2019.

Richmond, B. Introduction to Data Analysis Handbook. *Academy for Educational Development*, 2006.

Rojas, J. S. IP Network Traffic Flows Labeled with 75 Apps. 2019. Accessed on 15 April 2019.
URL `https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps`

Rudman, L. L. NetwIOC: a framework for the automated generation of network-based IOCS for malware information sharing and defence. MA thesis, Rhodes University, Faculty of Science, Computer Science, 2016.

Saggi, M. K. and Jain, S. A survey towards an integration of big data analytics to big insights for value-creation. *Information Processing & Management*, 54(5):758 – 790, 2018. doi:10.1016/j.ipm.2018.01.010.

Sato, A. and Huang, R. A generic formulated KID model for pragmatic processing of data, information, and knowledge. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE; 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE; 15th Intl Conf on Scalable Computing and Communications and its Associated Workshops (UIC-ATC-ScalCom)*, pages 609–616. Aug 2015. doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.120.

Sivarajah, U., Kamal, M. M., Irani, Z., and Weerakkody, V. Critical analysis of Big Data challenges and analytical methods. *Journal of Business Research*, 70:263–286, 2017.

Solomon, T., Zungeru, A. M., and Selvaraj, R. Network traffic monitoring in an industrial environment. In *2016 Third International Conference on Electrical, Electronics, Computer Engineering and their Applications (EECEA)*, pages 133–139. IEEE, 2016.

Stojkoska, B. L. R. and Trivodaliev, K. V. A review of Internet of Things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 140:1454–1464, 2017.

Sun, Z., Sun, L., and Strang, K. Big data analytics services for enhancing business intelligence. *Journal of Computer Information Systems*, 58(2):162–169, 2018.

Svacina, J., Raffety, J., Woodahl, C., Stone, B., Cerny, T., Bures, M., Shin, D., Frajtak, K., and Tisnovsky, P. On Vulnerability and Security Log Analysis: A Systematic Literature Review on Recent Trends, page 175–180. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450380256. doi:10.1145/3400286.3418261.

Talabis, M., McPherson, R., Miyamoto, I., and Martin, J. Information Security Analytics: Finding Security Insights, Patterns, and Anomalies in Big Data. Syngress, 2014.

Tao, F., Qi, Q., Liu, A., and Kusiak, A. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018.

Tejedor, E., Becerra, Y., Alomar, G., Queralt, A., Badia, R. M., Torres, J., Cortes, T., and Labarta, J. Pycompss: Parallel computational workflows in Python. *The International Journal of High Performance Computing Applications*, 31(1):66–82, 2017.

Wamba, S. F., Gunasekaran, A., Akter, S., Ren, S. J.-f., Dubey, R., and Childe, S. J. Big data analytics and firm performance: Effects of dynamic capabilities. *Journal of Business Research*, 70:356–365, 2017.

Wang, L., Wang, G., and Alexander, C. A. Big data and visualization: methods, challenges and technology progress. *Digital Technologies*, 1(1):33–38, 2015.

Wang, Y., Kung, L., and Byrd, T. A. Big data analytics: Understanding its capabilities and potential benefits for healthcare organizations. *Technological Forecasting and Social Change*, 126:3–13, 2018a.

Wang, Y., Kung, L., Wang, W. Y. C., and Cegielski, C. G. An integrated big data analytics-enabled transformation model: Application to health care. *Information & Management*, 55(1):64 – 79, 2018b. doi:10.1016/j.im.2017.04.001.

Weichbroth, P. Usability of mobile applications: a systematic literature study. *IEEE Access*, 8:55563–55577, 2020.

Weinstock, C. B. and Goodenough, J. B. On system scalability. Technical report, Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst, 2006.

Wu, P., Lu, Z., Zhou, Q., Lei, Z., Li, X., Qiu, M., and Hung, P. C. Bigdata logs analysis based on SEQ2SEQ networks for cognitive Internet of Things. *Future Generation Computer Systems*, 90:477–488, 2019. doi:10.1016/j.future.2018.08.021.

Xu, S. Y. and Berkely, C. Stock price forecasting using information from Yahoo finance and Google trend. *UC Brekley*, 2014.

Zhang, E. What is Log Analysis? Use Cases, Best Practices, and More. 2018. Accessed on 9 March 2020.
URL `https://digitalguardian.com/blog/what-log-analysis-use-cases-best-practices-and-more`

Zhang, Y., Ren, S., Liu, Y., Sakao, T., and Huisingh, D. A framework for Big Data driven product lifecycle management. *Journal of Cleaner Production*, 159:229–240, 2017a.

Zhang, Y., Ren, S., Liu, Y., and Si, S. A big data analytics architecture for cleaner manufacturing and maintenance processes of complex products. *Journal of Cleaner Production*, 142:626–641, 2017b.

Zhou, D., Yan, Z., Fu, Y., and Yao, Z. A survey on network data collection. *Journal of Network and Computer Applications*, 116:9 – 23, 2018. doi:10.1016/j.jnca.2018.05.004.

Zohuri, B. and Moghaddam, M. What is data analysis from data warehousing perspective? In *Business Resilience System (BRS): Driven Through Boolean, Fuzzy Logics and Cloud Computation*, pages 269–289. Springer, 2017.

Zou, D.-Q., Qin, H., and Jin, H. Uilog: Improving log-based fault diagnosis by log analysis. *Journal of Computer Science and Technology*, 31(5):1038–1052, 2016.