DETERMINING THE BEST-FIT PROGRAMMERS USING PROGNOSTIC ATTRIBUTES

SORADA PRATHAN

FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2018

DETERMINING THE BEST-FIT PROGRAMMERS USING PROGNOSTIC ATTRIBUTES

SORADA PRATHAN

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SOFTWARE ENGINEERING

FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR

2018

UNIVERSITY OF MALAYA
ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: SORADA PRATHAN

Matric No: WGC130001

Name of Degree: MASTER OF SOFTWARE ENGINEERING

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"): **Determining the Best-Fit Programmers Using Prognostic Attributes**

Field of Study:

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date:

Subscribed and solemnly declared before,

Witness's Signature Date:

Name:

Designation:

DETERMINING THE BEST-FIT PROGRAMMERS USING PROGNOSTIC ATTRIBUTES

ABSTRACT

The software development industry depends significantly on human capital to maintain competitiveness through the development of quality software systems and project a company's operational and service excellence. However, software companies find it difficult to identify and employ the best-fit computer programmers. This research is aimed at using a data mining technique to identify the best-fit programmers who fulfil the relevant eligibility criteria. The best-fit programmers were predicted using both the Bayes' Theorem and Artificial Neural Network (ANN). The predicted best-fit programmers were compared to the good programmers who were identified based on the past annual performance appraisal results of two software companies in India. The datasets from the two companies (Company 1 and Company 2) covered the years 2010-2015. The Bayes' Theorem was used to analyse the relevant programmer's attributes, while, the Artificial Neural Network (ANN) was used to predict the best-fit programmers. The research established that the Bayes' Theorem is useful in recognising the prognostic attributes of the best-fit programmers for software companies while Artificial Neural Network (ANN) classifier was effective in the predicting the best-fit programmers. Using a confusion matrix, the Artificial Neural Network (ANN) classifier performance is 97.2% and 87.3%, 95.8% and 54.5%, and 100% and 75% with regard to accuracy, precision, and recall on the two test datasets of Company 1 and Company 2, respectively. The results are satisfactory enough to introduce a new technique to identify relevant attributes for predicting the best-fit programmers. Software companies can use this technique in their recruitment and selection process to determine the best-fit employees for the programmer posts. The proposed technique can be adapted to be applied in other disciplines such as

sports, education, etc, to identify and employ the most suitable person to fill a particular position.

Keywords: Bayes’ Theorem, Artificial Neural Network (ANN), Performance, Prediction, Programmers.

MENENTUKAN PENGATURCARA YANG PALING SESUAI MENGGUNAKAN ATRIBUT RAMALAN

ABSTRAK

Industri pembangunan perisian bergantung tinggi kepada modal insan untuk mengekalkan daya saing melalui pembangunan sistem perisian yang berkualiti dan mempamerkan kecemerlangan operasi dan perkhidmatan syarikat. Namun begitu, agak sukar bagi syarikat perisian untuk mengenalpasti dan mengambil pengaturacara yang paling sesuai. Kajian ini bertujuan menggunakan teknik perlombongan data bagi mengenalpasti pengaturcara yang paling sesuai untuk memenuhi kriteria kelayakan yang berkaitan. Pengaturcara yang paling sesuai diramalkan melalui Teori Bayes dan Rangkaian Neural Buatan (AAN). Pengaturcara yang paling sesuai yang diramal dibandingkan dengan pengaturcara yang bagus yang dikenalpasti berdasarkan keputusan penilaian prestasi tahunan terdahulu di dua buah syarikat perisian di India. Set data daripada dua buah syarikat ini (Syarikat 1 dan Syarikat 2) merangkumi data dari tahun 2010-2015. Teori Bayes digunakan untuk menganalisa atribut pengaturcara yang berkaitan, manakala Rangkaian Nueral Buatan (AAN) digunakan untuk meramalkan pengaturcara yang paling sesuai. Kajian ini menetapkan bahawa Teori Bayes berguna untuk mengenali atribut ramalan pengaturcara yang paling sesuai untuk syarikat perisian manakala pengkelas Rangkaian Neural Buatan (AAN) berkesan untuk meramalkan pengaturcara yang paling sesuai. Menggunakan matrik kekeliruan, prestasi pengkelas Rangkaian Neural Buatan (AAN) ialah 97.2% dan 87.2%, 95.8% dan 54.5%, dan 100% dan 75%, yang berkaitan dengan ketepatan, kepersisan dan perolehan kembali bagi kedua-dua set data uji Syarikat 1 dan Syarikat 2, masing-masing. Keputusan ini adalah cukup memuaskan untuk memperkenalkan teknik baru ini bagi mengenalpasti atribut yang berkaitan untuk meramalkan pengaturcara yang paling sesuai. Syarikat perisian boleh menggunakan teknik ini dalam proses mencari dan mengambil pekerja bagi

menentukan pekerja yang paling sesuai untuk jawatan pengaturcara. Teknik yang dicadangkan ini boleh diadaptasikan untuk digunakan dalam bidang lain seperti sukan, pendidikan, dan lain-lain, bagi mengenalpasti dan mengambil pekerja yang paling sesuai untuk mengisi sesuatu jawatan.

Kata kunci: Teori Bayes, Rangkaian Neural Buatan (AAN), Prestasi, Ramalan, Pengaturcara.

ACKNOWLEDGEMENTS

I am very grateful to a number of people, who have been instrumental in getting this dissertation to its successful completion.

First and foremost, I would like to express my sincere gratitude to my dedicated supervisor, Associate Professor Dr. Ow Siew Hock, for her patience, guidance, encouragement, suggestion, and tremendous and unending assistance throughout my research. Her unwavering supports and suggestions have steered me in the right direction when I felt lost and faced problems in the research. She has always encouraged and motivated me and gave me unending inspiration to propel me to complete this research successfully. I have been extremely lucky to be under her supervision, for her concern about my research and prompt response to my questions. This research journey would not have been possible without her great support. Her role in my research was very essential to my success.

I also would like to thank all the staff at the Faculty of Computer Science & Information Technology, University of Malaya. In particular Mrs. Norhafidzan Ahmad for her extensive instruction, advice and help in my Master's programme.

My sincere gratitude to Mr. K.H. Teh, who helped in proofreading my dissertation.

I also would like to express my very profound gratitude to my mother, father, grandmother, aunts, and my lovely sister for their unconditional love and unfailing support for me throughout my years of study and throughout the process of researching and writing this dissertation.

Last but not the least, I would also like to express my sincere and special gratitude to Dr. Athicha Eakamornpan and Ms. Chanaphan Thammarut, who always believe in me, and give me words of encouragement and motivation. They have been a great source of inspiration that supports me in my spiritual journey to feel calm and composed during difficult times in life, when I felt hopeless and discouraged. I sincerely thank them for their kindness and generosity.

I would also like to gratefully acknowledge the University of Malaya for supporting and funding via Postgraduate Research (PPP) Grant (Account No. PO032-2015A) to produce and complete this dissertation successfully.

Finally, my thanks go to all the people who have supported me in one way or another in completing this research work.

Sorada Prathan

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

HRM	:	Human Resource Management
OS	:	Operating System
HR	:	Human Resource
DP-AHP	:	Dynamic Programming Analytic Hierarchy Process
AHP	:	Analytic Hierarchy Process
DSS	:	Decision Support System
TOPSIS	:	Technique for Order Preference by Similarity to Ideal Solution
IT/IS	:	Information Technology/Information System
NB	:	Naive Bayes
ID3	:	Iterative Dichotomiser 3
NN	:	Neural Network
LMA	:	Levenberg-Marquardt Algorithm
ICT	:	Information and Communication Technology
IS	:	Information System
IT	:	Information Technology
ANN	:	Artificial Neural Network
LM	:	Levenberg-Marquardt
MLP	:	Multilayer Perceptron
TP	:	True Positive
TN	:	True Negative
FN	:	False Negative
FP	:	False Positive
MSE	:	Mean Squared Error
SVM	:	Support Vector Machine

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

1.1 Overview

In recent years, human capital has become a major competitive aspect for the software development industry. The staff recruitment procedure and selection process of an organisation directly affect employees' quality. Therefore, to maintain the quality of employees, various techniques have been applied to the recruitment process by the Human Resource Management (HRM) of organisations in order to identify the most talented employees (Agrawal, Khatri, & Srinivasan, 2012; Kalugina & Shvydun, 2014; Thompson & Ahrens, 2015; Kiantoa, Sáenzb, & Aramburu, 2017; Collins, 2018). However, during the recruitment process, most organisations focus on the nature of the work and analyse various tasks related to work nature without using relevant attributes that can be used to help the organisation in identifying and determining the best-fit employees for the jobs that they want to fill. The Best-fit employees are those candidates who meet the assessment selection criteria for relevant positions and can potentially produce the expected outcome for the benefit of the company.

In software development, the personnel aspect is a major concern for the organisation to produce high quality software systems. Such concerns include being able to identify the best employees who can contribute in achieving quality objectives, as quality software depends on personnel and work procedures in software development (Gupta & Suma, 2015).

Inconsistency in the level of knowledge and attitudes of employees in software development are other obstacles for the HRM to frame the selection procedures. Moreover, the advancement in technology and globalisation has given rise to various new cross-functional tasks which created new jobs. The requirement process involved in identifying quality personnel in software organisation has become stringent as the nature of work are new and complicated. Thus, the existing selection procedures which are framed based on constant work nature are not suitable for the current situation. Hence, to identify the best-fit employees for the right job at the right time, it is necessary to develop an effective selection framework for the organisation to recruit quality employees (Strohmeier & Piazza, 2013).

In the current situation, software organisations need to be more efficient in providing quality service at less cost and with more innovative ideas in order to stay competitive. Hence, the success of each software organisation depends on considering these factors seriously in order to employ the best-fit employees. In this context, managing talents within an organisation is an important challenge in HRM as it involves various managerial decisions in selecting the best-fit employees at the right time (Jantan, Hamdan, Othman, & Puteh, 2010).

1.2 Research Background

Software companies seek to develop better quality software that are easy to maintain and reliable (Atalag, Yang, Tempero, & Warren, 2014; Durmus, et al., 2016; Amara & Rabai, 2017; Khosravi, Hussin, & Nilashi, 2018). Therefore, software companies consider every quality aspects to ensure success of project through process efficiency using various techniques. However, even with better hardware and operating systems (OS), there are still reports of disastrous failures in software development. Most software failures are attributed to the lack of focus of people involved in the software development process. The recruitment of new personnel in software development is one of the important processes in business which can influence the human capital quality within a software company. It is highly important for software companies to ensure that good and highly qualified talents are employed to maintain a competitive edge over others in the industry.

Handling an organisation's talent is one of the challenges faced by human resource professionals. Many sound managerial decisions have to be made to select the right person for the right job and at the right time. Occasionally, it is difficult to reach clear decisions on certain issues, and thus it is crucial to rely on various factors such as human experience, knowledge, preference and judgment (Segall & Champness, 2015).

Human Resource (HR) in software development is a major issue to consider in the software industry in order to attain quality outcomes. In recent years, software companies have been concentrating on identifying the right people who are well-versed in software architecture and can perform efficiently, as the quality of software relies on the quality of the developers (Nair, Suma, & Tiwari, 2012). Success of a software organisation depends to a great extent on its talent management and deploying people with the best skills for the most appropriate tasks. This aspect of talent management is the biggest challenge as it involves making decisions on assigning the right person to the right job (Jantan et al., 2010).

Most software organisations have problems in identifying appropriate talent due to lack of appropriate selection criteria in the recruitment process. The primary objective of a software organisation is to provide quality software to the stakeholders (Gupta & Suma, 2013). Software development involves many methods and processes in order to ensure quality outcomes. Thus, software quality can be achieved by having personnel commited to applying quality processes in software development (Thakur & Gupta, 2015). Many studies have suggested that the software development process should focus more on the people involved in developing software (Gupta & Suma, 2015). Some software companies are using methods such as non-parametric and parametric data mining techniques to recruit their employees.

In recent years, many researchers have developed data mining-based frameworks to support decision-making for improving the efficiency of HR management. Data mining has been widely used in many sectors and achieved great outcomes (Strohmeier & Piazza, 2013). The technique is used to extract all necessary information from data. The data mining process is a combination of analytics of knowledge base and domain knowledge, which extracts the unknown information hidden in data by applying various data mining algorithms (Thakur & Gupta, 2015).

1.3 Research motivation

Grant and Sackman (1967) expressed that out of 28 programmers, only one programmer can truly meet the programmer ability required. In another study, Prechelt (2000) stated that the ratio is 4:1. These findings show that finding good programmers is not an easy task. The Standish Group's CHAOS Report (Johnson, 2018) also reported that having competent employee is one of the key factors which drive the software project to successful completion. Hence, understanding the software project success factors is crucial to enable the software companies to invest wisely on each factor, leading to better chances of software project success and to avoid costly errors. Data mining, which can provide a wealth of knowledge, has proven to be one of the emerging techniques that offers a way to discover the interesting patterns covert in a large amount of data stored in repositories. Generally, organisations often have a large set of data related to employee performance stored in the organisations' database. Thus, data mining technique can be used to analyse these data as it will have the potential to discover the hidden trends and patterns from these data, and also to predict the future trends, and make a wise decision on the selection of the right programmers. Hence, this research is aimed at using data mining techniques to find the best-fit programmers with the skills required by the software companies.

1.4 Problem Statements

Software companies have always strived to provide operational and service excellence as an inherent part of their mission, vision or goals. Human resource is a crucial aspect of that operational and service excellence to the company and findings from many studies have advocated that those involved in the process of developing a software, have a significant role to play in ensuring software quality, and in contributing to the overall performance of the company (Kosti et al., 2018). The computer programmers in particular, directly influence the overall software productivity and quality (Kamma, 2014).

As stated in the overview above, human resource has been a major challenge to the software industry in recent years. Even though organisations have been using suitable frameworks for recruiting the right and talented employees, the problem of identifying best-fit programmers still exits. In a survey of software companies, Pratt (2014) found that the computer programmer position was the hardest job position to fill. Although, some programmers can be employed, it is still difficult to get suitable programmers who can competently handle the complexity and scale of the tasks. They acknowledged that the problem arises from their lack of understanding and not using the right selection criteria for recruiting new computer programmers.

Current recruitment and selection criteria are merely based on the job requirement, personality traits such as a person's behaviour, social ability and so on, as well as the personal information details which include demographics, gender, age, GPA, experiences, and so forth. However, these job requirements, personality traits, and personal information details alone cannot identify the right programmers to be recruited to work to the level expected by the company (Strohmeier & Piazza, 2013). Various studies have revealed the relationship between the recruitment criteria and job performance (Vianen & Pater, 2012; Kristof-Brown & Billsberry, 2013; Farooqui & Nagendra, 2014; Swider, Zimmerman, & Barrick, 2015; Pradhan & Jena, 2016). Therefore, to obtain all the pertinent selection criteria for a programmer, past data on job performance must be considered as it contains the results of performance evaluation on the tasks the position requires.

1.5 Objectives of the research

Research objectives are the affirmative statements which are intended to describe the goals related to the research study, and how the goals will be accomplished to generate feasible outcomes (Chua, 2016). The objectives of this research are:

1. To identify the attributes that can be used in assessing the type of programmers a software company needs;
2. To propose a recruitment and selection technique that can be used to determine the best-fit programmers for a software company; and
3. To evaluate the proposed technique.

1.6 Research Questions

The research questions of this study are as follows:

1. What are the attributes that a programmer must possess to meet the needs of a software company?
2. How to determine the best-fit programmers for a software company?
3. How can a software company evaluate the technique to determine the best-fit programmers?

1.7 Scope of the Research

This study is limited to software companies and software programmers only. The data used were collected from two software companies in India. The data include the programmers' annual performance appraisal results for the period 2010-2015. In this research, the criteria used in evaluating the programmers' annual performance appraisal vary in the two software companies but generally, they do not focus on personnel details such as gender, age groups, education background, personality traits, experience, etc.

1.8 Research Contributions

This research introduces a technique for identifying the prognostic attributes (high potential attributes) which are used as selection criteria in predicting the best-fit programmers. The technique can be used by software companies to evaluate their existing programmers or when recruiting new programmers who fulfill their requirements. Hence, this proposed technique can be a useful guide to software companies for identifying accurately the best-fit programmers who can contribute positively to the company and ensure its profitability and enhance its professional status.

The human resource management department could use this proposed technique for determining the best candidate for the programmer post. This technique could possibly be applied in other disciplines such as sports, financial/banking, education, and medical domains for predicting the best-fit candidates for specific posts.

1.9 Organisation of Dissertation

This dissertation consists of six chapters:

Chapter 1: This chapter provides an overview of the study conducted, following with research motivation, research objectives, research questions, research scope and organisation of dissertation. The content of this chapter will give an insight on the significance of this study.

Chapter 2: This chapter presents a review of the literature pertaining to employee/programmer recruitment process and the existing problems faced with the current recruitment process. It then provides detailed explanation on the recruitment technique, Bayes' Theorem and Artificial Neural Network (ANN).

Chapter 3: This chapter explains the methods used to achieve the research objectives and describes the research phases and activities. It also provides an overview of the workflow of the proposed technique.

Chapter 4: This chapter describes in detail the proposed technique used to address the problem statements, together with a discussion of the system implementation of the proposed technique.

Chapter 5: This chapter presents the major findings from the proposed technique. Besides, the evaluation process for assessing the outcome and performance of the proposed technique will be described.

Chapter 6: This chapter summarises the findings of the study, describes the limitations and constraints of this study, followed by conclusion of the research and recommendations for further study.

CHAPTER 2: LITERATURE REVIEW

2.1 Overview

Today, the growing demand for reliable and quality software has led to greater emphasis on effective project management to achieve better quality assurance in software. Software systems play an important role in every aspect of our daily life. Software development companies are very much aware that major software project failures are related to the human factor, and thus, this issue must be urgently addressed. For successful software development, the right people should be appointed for the right jobs. One of the major concerns of companies today is recruiting the right people for the right jobs (Jantan, Hamdan, Othman, & Puteh, 2010). Some data mining approaches based on knowledge discovery and machine learning have been used to classify the employees. These classification methods have been useful in predicting the employees' performances (Thakur, Gupta, & Gupta, 2015). Some of the methods are reviewed in this chapter.

2.2 Employee/Programmer Recruitment and Selection Process

Recruitment is "the process of attracting individuals on a timely basis, in sufficient numbers and with appropriate qualifications to apply for jobs with an organisation" (Mondy & Martocchio, 2016). Every organisation that subscribes to this definition will take the recruitment process very seriously. It is a fundamental policy of the human resource management to treat the employees as one of the important factors of production (Faerber, Weitzel, & Keim, 2003) and thus to hire the best employees for the organisation (Laumer & Eckhardt, 2010). Organisations today are making drastic changes internally to cope with a highly turbulent external environment. Frequent reorganizing, downsizing, rightsizing, hierarchical flattening, teaming and outsourcing shape the staff recruitment and selection process. This is influenced by the fact that many people are experiencing major difficulties in their attempts to adapt to the uncertainties of career life (Larsson, Brousseau, Kling, & Sweet, 2007). Furthermore, increasing competition in the global markets drives organisations to put more emphasis on the personnel selection process. In software organisations, important issues such as changes in the organisations, work, society, regulations and marketing influence programmer/employee selection and recruitment. Problems such as getting the right employee/programmer for the right job, reducing recruitment time and advertisement costs are some of the issues in the recruitment process (Al-Otaibi & Ykhlef, 2012).

Some organisations make a strategic decision when choosing the best candidate by going through rigorous and costly selection processes, while others decide on recruiting employees as soon as they could based solely on the information stated on the applicant's form. Gusdorf et al. (2008) proposed other alternatives for employee recruitment such as outsourcing or contingent labour, part-time labour or overtime work by existing employees due to the staggering costs of the recruitment and selection processes.

Some organisations also differ with respect to the procedures and budgets for recruiting, selecting and orienting people (Karsak, 2001; Afshari, Mojahed, Yusuff, Hong, & Ismail, 2010). Many organisations also expand their employee/programmer search using recruitment platform such as posting the job ads on the career unit of a corporate website, social media or internet job portal. On the other hand, job seekers use job portals to declare their curriculum vitae and suitability for specific jobs as desired by the recruiting organisations (Fazel-Zarandi & Fox, 2009). These initial steps in personnel recruitment create better understanding of the recruitment process leading to other changes and thus, a more enhanced recruitment process. The use of recruitment platforms is one of the most successful approaches which companies used in employing the best candidates (Al-Otaibi & Ykhlef, 2012). Recruiting the best employee/programmer is a challenge faced by most software companies. The unavailability of people with specific skills such as in programming, has long been known to be a major drawback to companies' success (Laumer & Eckhardt, 2010).

The global attachments of companies and organisations to employing the best candidate/programmer during a recruitment and selection process, have made it necessary for analytical decision-making regarding the matter. Some issues to seriously consider in the programmer recruitment process include the way to assign weights to important recruitment and selection criteria (desired programmer attributes), how to use linguistic or numerical scales for assessing the applicants against multiple criteria, as well as the method of aggregating the evaluation results and ranking the applicants, etc.

Carroll, Marchington, Earnshaw, and Taylor (1999) suggested that the recruiting process should consist of four phases: (i) an assessment of job position that needs to be filled; (ii) a description of the job profile; (iii) the construction of a job description; and (iv) a candidate/programmer specification. Breaugh and Starke (2000) suggested that the recruitment process should consist of four major tasks: (i) short-term and long-term candidate/programmer attraction; (ii) applicant management; (iii) pre-selection; and (iv) final selection of candidates/programmers. Short-term and long-term marketing assessments involve projecting an attractive employer image to attract the best candidates. Other approaches used in the recruitment processes include use of the internet for advertising, use of online internet job boards, social media applications or the organisation's own website (Laumer, Eckhardt, & Weitzel, 2009). For applicant selection, organisations could use IT-based resumes and various e-assessment systems. Lee (2007) suggested using a holistic electronic recruiting system to support the recruiting process from the time of advertising a vacancy, and also using new applications such as social media, etc.

Some studies have been conducted on the consequences associated with the employee's recruitment and selection process. Gusdorf (2008) emphasized that an organisation's promotion policy can have a significant effect on: its recruitment policy; reduction in the costs for recruitment and selection (Chapman & Webster, 2003; Malinowski, Keim, & Weitzel, 2005; Musaa, Junaini, & Bujang, 2006); increase in the number of suitable and unsuitable candidates (Chapman & Webster, 2003; Parry & Tyson, 2008); high implementation costs of e-recruitment system by organisations (Chapman & Webster, 2003); time savings for organisation and applicants (Malinowski, Keim, & Weitzel, 2005); misuse of applicants data (Lin & Binshan, 2002); perceived fairness of programmer selection (Chapman & Webster, 2003); perceived loss of individuality (Chapman & Webster, 2003); improved corporate image (Parry & Tyson, 2008); and higher independence of other methods for programmer recruitment (Parry & Tyson, 2008). However, careful HRM planning must be considered for the overall growth prospects of the organisation and accurate forecasting of future programmer needs. Recruitment planning begins only when other alternatives have been considered and eliminated (Gusdorf, 2008). Table 2.1 presents the internal and external environmental factors that can hinder the recruitment process in an organisation, as well as the advantages and disadvantages of those factors. The strengths and weaknesses of the recruitment process in an organisation are presented in Table 2.2.

Table 2.1: Internal and External Environmental Factors that Can Hinder the Recruitment Process in an Organisation (Gusdorf, 2008; Mondy & Martocchio, 2016)

Factors	Merits	Demerits
Internal Environmental: Promotion from within	• Encourages employees to be more aware of their career opportunities when they see their co-workers being promoted. • Promoted employee is already comfortable with the corporate culture, knows organisation policies and will likely work much faster than new employee in the organisation.	• Promoted person leaves a staffing gap in his/her former position and position needs to be filled. • Organisation may lose out on chances for new ideas and creativity that can come from employing new employee.
Internal Environmental: Nepotism	• Most employers require family members to work in different areas of the company to avoid issues of favouritism and possible morale problems among employees	• It could be dangerous to refuse hiring a close relative due to inappropriate and illegal employment decision. • It is never appropriate to allow family members in a supervisory position because of difficulty in managing relatives.
External Environment: Legal issues	• The law applies to organizations with 15 or more employees	• Legislation prohibits discrimination based on race, gender, religion, or national origin
Labour Market Conditions	• With high level of unemployment, organisation may hire cheap labour	• Increased compensation to attract quality applicants if there is a limited number of skilled employees and the economy is very strong • Managing huge number of applications that must be pared down to find only a few good ones.

Table 2.2: Employee/ Programmer Recruitment Process (Lang, Laumer, Maier, & Eckhardt, 2011; Ozdemir, 2013)

Process	Strength	Weaknesses
Employer Branding /Selection/suitability to the position	• Select the best candidate with desired qualification • Determine the input quality of programmers • Establish/improve employer's /company's image • Save time for both organisation and the applicants	• Changes in work ethics and organisation regulation and marketing affect programmer selection • Increased competition
Personal Attraction	• Programmer attraction could be from channels like internet job boards, social media applications or organization's website. • Provides structured and in-depth analysis of worker skills	• Some information presented in the application may not necessarily reflect the true personality of applicant
Applicant Management/Examination	• Will identify the best candidates after rigorous assessment	• Time-consuming to get the right candidate/programmer
Pre-Selection/ Individual interview/ Group interview	• Interacting with the employees before final decision is taken in the company's interest. • Provides reference/background checks, along with second interviews to gather additional information as well as temporal differences in skills and behaviour	• Panels may be subjective in their decision, if there is no standardization in the selection process

Table 2.2: (Continued)

Process	Strength	Weaknesses
Evaluation/Final Selection	• Assess the best programmer using multiple criteria	• Set criteria weights to reflect the best

Hiring the best candidates is a good way of reducing high employee turnover. The use of ICTs and the Internet has helped to reduce the stress faced by the HR department in an organisation. However, the recruitment process is still not perfect even with the use of information technologies to process the applicants. In a typical organisation, the manager defines the target job and decides what minimal requirements, preferred qualifications and additional skills he desires of a candidate. Once the job requirements have been defined, the company identifies the most appropriate assessment methods to be used in the hiring process. Table 2.2 displays different recruitment processes or approaches for hiring the best candidate for an organisation. Ozdemir (2013) stated that the recruitment process starts with the employer's branding, personnel attraction, applicant management, preselection and finally selection or evaluation. He further elaborated that to determine the weight for the recruitment criteria in order to hire the best programmer, comparison matrices are used by experts involved in the programmer selection process in the organisation. He also calculated a criteria comparison using criteria value (*CR value)* which found that group interview and individual interview both play a key role in the employee selection process for a company with the highest weights compared to other steps.

2.3 Employee/Programmer Recruitment and Selection Technique

Although the employee recruitment policy differs among companies, it remains an important and decisive objective to hire the best personnel for the company. Lin (2010) stated that the intrinsic value and complexity of the candidate selection process require the use of effective analytical methods to provide an operational or tactical decision framework. Ozdemir (2013) believed that a more effective and rational personnel selection decision would be better than an intuitive decision when selecting the best employee. Thus, he proposed a two-phase multi-criteria Dynamic Programming Analytic Hierarchy Process (DP-AHP) mathematical method for personnel selection. Likewise, Kwak, McCarthy, and Parker (1997) developed an analytical hierarchy method which they believed would be more effective and rational for selecting hospital personnel. Seol and Sarkis (2005), on the other hand, used Analytic Hierarchy Process (AHP) for internal auditor selection, and found that organisations try to identify candidates' potential for success using AHP. In addition, Gibney and Shang (2007) showed how the AHP can provide a well-structured, coherent, and justifiable selection practice. Employee selection problems require a systematic approach to incorporate an organisation's values as well as the needs of its stake-holders. They concluded that AHP is a practical, versatile and powerful tool that can explicitly identify the factors that matter and provide a consistent structure and process for evaluating candidates.

Shih, Huang, and Shyur (2005) proposed a two-phase method with eight steps for decision-making in selecting the best personnel for an organisation - the Decision Support System (DSS) using AHP, and the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) methods. Also, Chen and Cheng (2005) established a Fuzzy group decision support system for selecting IS personnel. Their method helped in ranking fuzzy numbers. Kelemenis and Askounis (2010) proposed a new TOPSIS–based multi-criteria technique for personnel selection.

Faerber, Weitzel, and Keim (2003) used the relationship between models in recruiting the best candidates. In their proposed model, the recruiting tasks are divided into two main phases - the attraction phase and the selection phase - with both phases having a planning part as well as an execution part. The planning part determines the overall strategy and actual measures to attract valuable employees/programmers. The execution part involves recruiter branding activities, which include all long-term marketing strategies for attracting suitably qualified or best candidates. The attraction phase is aimed at generating a description for open job positions, while the selection phase starts with the pre-screening of resumes and other submitted materials. The final selection of candidates is then conducted by comparing the remaining candidates who have not been short-listed in the screening phase. Table 2.3 shows some of the existing applications which use different techniques for recruiting employee.

Table 2.3: Existing Applications for Recruiting Employee

Author	**Techniques/Methods**	**Application Areas**	**Source for Identifying Recruitment Criteria**
(Kelemenis & Askounis, 2010)	New TOPSIS–based multi-criteria techniques in personnel selection	Information System (IT) segment	Job descriptions
(Hsiao, Chang, Huang, & Chen, 2011)	Analytic Hierarchy Process (AHP)	Information System (IT) segment	Questionnaire for information software specialists (based on own perception on defining the eligibility of the recruitment criteria)

Table 2.3: Continued

Author	Techniques/Methods	Application Areas	Source for Identifying Recruitment Criteria
(Ozdemir, 2013)	Dynamic Programming - Analytic Hierarchy Process (DP-AHP)	Manufacture-Marketing department position	Manufacturer defines the set of criteria for determining suitability for a position (based on own perception on defining the eligibility of the recruitment criteria)
(Mammadova, Jabrayilova, & Mammadzada, 2016)	Fuzzy Multi-scenario	Information System (IT) segment	Conducted a survey of IT companies to determine the set of recruitment criteria (based on own perception on defining the eligibility of the recruitment criteria)
(Qin, et al., 2018)	Expert System - ANN	Information System (IT) segment	Job descriptions and resume

The traditional method of determining the recruitment criteria is guided by the job description based on human perception on the trait requirement for a job. The decision to determine a set of eligibility criteria for selection of an employee would have a human bias associated with it (Strohmeier & Piazza, 2013; Al-Kassem, 2017). This is because the human brain can be constrained in making fair judgment, and also in making an assumption based on the existing belief (Kruglanski & Higgins, 2007). This hinders efforts in recruiting the best-fit employee/programmer who meets the real requirements of the job as well as of the organisation (Al-Kassem, 2017). However, there still remains the issue of determining a set of eligibility criteria for selection of employees in the Information Technology/Information System (IT/IS) domain (Hsiao, Chang, Huang, & Chen, 2011; Radant, Palacios, & Stantchev, 2015).

2.4 Bayes' Theorem

Bayes' Theorem was introduced by an English mathematician, Thomas Bayes, who submitted an essay on inverse probability to the Royal Society, London (Bayes & Price, 1763). The main feature of Bayes' Theorem is that it is capable of interpreting evidences from prior experience or knowledge as well as considering new evidences to update a previous experience or knowledge. In the classical theory of probability, the probability of every events that had occurred also has equal probability or likelihood of happening. This probability is considered merely as a finite number of possible outcomes. However, based on pragmatic real-world situation, there is no equal probability of every event happening. However, Bayes' Theorem allows the possibility distribution for every possible event determined based on available information or knowledge (Spanos, 1986; Peterson, 2017). Bayes' Theorem explains the usage of a conditional probability expressed as an occurrence of certain event that affects the probability of another event, which assumes that H is a hypothesis and E is an observed evidence related to H. It can be stated mathematically as (Stone, 2015):

$$P(H|E) = \frac{P(E|H) * P(H)}{P(E)} \qquad \text{———————} \quad (1)$$

where, i) $P(H|E)$ is a probability that a hypothesis is true given its observed evidence values and it is called posterior probability; ii) $P(E|H)$ is a probability of seeing an evidence when a hypothesis is true, and it is called likelihood; iii) $P(H)$ is a probability of a hypothesis being true before seeing an evidence and it is called prior probability; iv) $P(E)$ is a probability of that evidences of all possible events and it is called marginal probability. Bayes' Theorem is a powerful technique for updating the beliefs of a hypothesis when additional evidence is only obtained later (Spačková & Straub, 2012; Zhang, Wu, Skibniewski, Zhong, & Lu, 2014; Zhang, Ekyalimpa, Hague, Werner, & Abourizk, 2015).

A number of current published studies have advocated the use of Bayes' Theorem over ordinary statistics. Many of them claimed that the Bayes factor (based on Bayes' Theorem) has outperformed the p-value in testing a hypothesis (Nanz & Furia, 2015; Furia, 2016; Perezgonzalez, 2016; Furia, 2017; Assaf & Tsionas, 2018). Bayes' Theorem allows one to perceive the whys and wherefores of interpretation of the hypothesis testing to be more understandable (Assaf & Tsionas, 2018). It also produces accurate and reliable estimation on testing a hypothesis (Furia, 2016). On the other hand, ordinary statistical methods only decide on whether to accept or reject a hypothesis based on the initial assumption which relies on the fixed threshold value (Rouder, Speckman, Sun, Morey, & Iverson, 2009; Gelman, 2016). Since this threshold value cannot guarantee that a p-value of greater than 0.05 is a suitable value to test the hypothesis in all cases inasmuch as different cases might require different scale of p-value to test the hypothesis (Goodman, 2008). Another limitation is that if the p-value is a small value, this can trigger rejection of the null hypothesis. However, this does not help in determining which alternative hypothesis should be accepted (Furia, 2017). In addition, Bayes' Theorem has been applied in a wide range of practical uses in the real-world situation, specifically to identify and analyse the relevant features, attributes or criteria used in certain applications (Watanabe, 2012; Mimović, Stanković, & Milić, 2015; Sarna & Bhatia, 2016). Table 2.4 presents some of the existing applications which use Bayes' Theorem to identify and analyse the relevant features.

Table 2.4: Existing Application Using Bayes' Theorem

Authors	Application Area	Technique Use	Findings
(Balamurugan & Rajaram, 2009)	Data mining techniques (Identifying the feature selection)	Bayes' Theorem	Increase in classification accuracy of NB, ID3 and Neural Network (NN), decrease in the running time for large-scale data.
(Appavu, Rajaram, Nagammai, Priyanga, & Priyanga, 2011)	Data mining techniques (Identifying the feature selection)	Bayes' Theorem and Information Gain	Improves classification accuracy of C4.5 and NB classifier; Reduces the computational time to run the algorithm.
(Mani & Kalpana, 2016)	Data mining techniques (Identifying the feature selection)	Bayes' Theorem, Self-Information and Sequential Forward Selection	Increase in predictive model performance at 1% of accuracy, precision and recall.
(Fang & Lan, 2018)	Performance Evaluation Model of Computing System (Modifying weight of the feature)	Bayes' Theorem	Significantly improves the overall accuracy of the evaluation performance model of computing system.

Table 2.4 clearly shows that Bayes' Theorem is a helpful mathematical approximation technique for identifying and analysing the relevant features in a classification model, and it significantly improves classification accuracy as well as reduces computational time of the algorithm. These advantages of Bayes' Theorem should be incorporated and applied in different domains and subjects to observe the results from various areas of studies (Furia, 2016; Furia, 2017; Fang & Lan, 2018).

2.5 Artificial Neural Networks (ANN)

The Artificial Neural Network (ANN) is considered to be a suitable technique for prediction, classification, and pattern recognition (Guo, et al., 2013; Wu, et al., 2018). ANN is also useful in handling noisy, nonlinearity and outliers in datasets (Rossel & Behrens, 2010; Beucher, Møller, & Greve, 2017). The learning method is the most significant characteristic of an ANN, as it is capable of metacognition about what it has learnt from experience to perform a given task. The learning method of a neural network can improve its performance after each iteration process through updating the weight of interconnected neurons to find the desired weight that can minimise the difference between its resulting output and the known output. These learning methods can be classified into two major types: supervised and unsupervised learning methods. Many studies have reported that the supervised learning method is preferred for solving a classification problem (Akritidis & Bozanis, 2013; Amato, et al., 2013; Szalkai & Grolmusz, 2018; Vidya, Sonawane, & Bhakti, 2018). The input data for classification provides an output class label associated with it. In the supervised learning method, the correct predefined outputs have been communicated to the corresponding input pattern during the training process of an ANN. Each iteration of the learning process is intended to minimise the error between the resulting output from ANN and the known correct output (Prieto, et al., 2016). On the other hand, an unsupervised learning method does not provide a correct predefined output to the ANN. It must learn by discovering the patterns from the input data for detecting the patterns, generating the rules, grouping and summarising of input data. This learning method may assist in describing and acquiring valuable meaningful insights into the data (Silva, Spatti, Flauzino, Liboni, & Alves, 2016).

The different objectives to solve a certain task of an ANN can be achieved by the different topologies of the ANN. There are three types of ANN topologies based on the interconnected pattern of the neural network: Feed-Forward Neural network, Recurrent Neural network and Hopfield Neural network (Arockiaraj, 2013).

i Feed-Forward Neural Network

The Feed-forward neural network was the first topology to be introduced and is the simplest type of ANN topology (Arockiaraj, 2013). The information signal in this ANN topology only travels in one direction, which is from the input layer through the hidden layer to the output layer. Each neuron node in a layer is connected to every neuron nodes in the adjoining layer. There is a weight associated with each of the connection between the neuron nodes of every layer. These weights are adjusted to obtain a strong connection signal between the neural nodes at the time the ANN is learning (Uddin, Jameel, & Razak, 2015). Feed-forward neural networks are commonly used for pattern classification, pattern matching, and decision-making in data mining applications (Wren, 2012).

A review of the literature indicates that the performance of the Feed-forward neural network is better than the performance of the traditional statistical approach (Singh & Kumar, 2010; Mohd & Yahya, 2018). Singh and Kumar (2010) applied the Feed-forward neural network to seven software failure data collected from software projects to observe the predictive accuracy performance between the proposed Feed-forward neural network and three software reliability growth prediction models that are based on the traditional statistical approach. In the proposed Feed-forward neural network, there are three layers which consist of single neuron node in the input layer (execution time of the testing phase), one hidden layer with several hidden neuron nodes and single neuron node in the output layer (cumulative failures). The experimental result of this research shows that the proposed Feed-forward neural network outperforms the three software reliability growth prediction models in terms of better predictive accuracy.

Mohd and Yahya (2018) attempted to construct the Feed-forward neural network with backpropagation algorithm to compare the accuracy in predicting depression among MIT students with the conventional logistic regression model. The dataset was obtained from the analysed questionnaire data collected from the MIT students. The Feed-forward neural network consists of three layers - input layer, hidden layer, and output layer. The results of the study show that the Feed-forward neural network achieves higher accuracy rate of 71.8% in predicting depression among the students compared with the logistic regression model which achieves accuracy rate of 62.5%.

ii Recurrent Neural network

A recurrent neural network also known as feedback loop, allows information signal in the network to travel in a directed cycle backward to itself (Chellappa, Diniz, Suykens, & Theodoridis, 2014). This neuron with self-feedback loop connection will create an internal state that enables them to remember past information with a temporal processing behaviour that stores information in short-term memory. This type of neural network topology is efficient for solving a task where the order of the data is of paramount importance. Generally, it is mostly applied in handwriting recognition, speech recognition, natural language processing, and motion generation (Arockiaraj, 2013).

Qin et al. (2018) used the Recurrent neural network to predict the degree of matching between the job position requirements and the applicant's experiences stated in the resume to overcome human bias in the data. There are three main components: Word-level, Hierarchical Ability-Aware, and Person-Job fit, in the study. Recurrent neural network was constructed to represent the semantic in a word level for job posting requirements and the applicant's experiences. The Ability-Aware component was designed to capture the semantic relationship between job requirements and candidate experience. The Person-Job fit component uses the semantic relationships generated from the previous step to measure the matching degree level between the job requirements and candidate experience. The finding of this research shows that the proposed model based on Recurrent neural network outperforms the other five baseline methods.

iii Hopfield Neural network

A Hopfield neural network is a fully connected single layer network where each individual neuron is connected to every neuron but not itself (Lipton, Berkowitz, & Elkan, 2015). The weight connection between two neurons are the same in both direction (symmetric weight). The input vectors are in binary format with activation values of 0, 1 or 1, -1. The desired pattern is sent to the input pattern to train a network iteratively until the network converges to a stable state and after a few iterations the pattern value does not change (Haykin, 2016). The typical applications of Hopfield neural network include: image processing, speech processing, signal processing, pattern recognition, and etc (Sulehria & Zhang , 2007). Table 2.5 summarises relevant studies on the application of Artificial Neural Network (ANN) in the Human Resource (HR) field.

Table 2.5: Summary of the Existing Applications of Artificial Neural Network in Human Resource Field.

Authors	Activity in HR	Technique/Task	Application
(Wang & Jiang, 2010)	Compensation management	• ANN/ Prediction	Employee evaluation
(Waheed, Zaim, Zaim, Sertbas, & Akyokus, 2013)	Talent management	• ANN/ Classification	To identify talented personnel for a job posting
(Heiat, 2016)	Employee attrition	• ANN/ Classification • Decision tree- C&R tree	To predict employee attrition rate

Table 2.5: Continued

(Wang & Shun, 2016)	Employee retention	• ANN/ Prediction	To predict the probability of employee retention (staying) in the organisation for management position
(Kavitha & Thomas, 2017)	Employee retention	• ANN/ Classification • Decision tree- Random Forest	To develop a decision support system for predicting an employee retention to assist the Human Resource Management System (HRMS)
(Qin, et al., 2018)	Recruitment and selection	• ANN/ Prediction	Develop a recruitment system that seeks to fit a person to a job based on past job application data

2.5.1 Levenberg-Marquardt Algorithm (LMA)

A Levenberg-Marquardt Algorithm (LMA) - one of the training algorithms of the Artificial Neural Network (ANN) - is the fastest training algorithm for solving nonlinear least squared problems (Madsen, Nielsen, & Tingleff, 2004). It has been widely used in various disciplines (Bazaraa, Sherali, & Shetty, 2006). It is built partially on Gauss-Newton and gradient descent methods (Hagan & Menhaj, 1994; Chen, Billings , & Grant, 2007). Many studies on ANN have asserted that LMA has the fastest ANN convergence rate compared to other algorithms such as Steepest Descent (Kumar & Rajasekhar, 2017), Standard Backpropagation (Min, Xiao, Cao, & Yan, 2017), Quickprop (Khan, Gaurav, & Kandl, 2013), Delta-Bar-Delta (Khan, Gaurav, & Kandl, 2013), Simple Scaled Conjugate Gradient (Mukherjee & Routroy, 2012; Mohamad, Zaini, Johari, Yassin, & Zabidi, 2010), and Quasi-Newton (Mukherjee & Routroy, 2012). Table 2.6 summarises relevant studies which applied the Levenberg-Marquardt algorithm.

Table 2.6: Application of Levenberg-Marquardt Algorithm

Authors	Domain	Algorithm	Application
(Cui, Xiong, Zheng, & Chen, 2012)	Medical	Levenberg-Marquardt Algorithm	To construct a mental diagnostic system
(Khan, Gaurav, & Kandl, 2013)	Medical	Levenberg-Marquardt Algorithm	To classify diabetes conditions among patients
(Billah , Waheed, & Hanifa, 2016)	E-capital market	Levenberg-Marquardt Algorithm	To predict the stock market
(Mammadli, 2017)	E-capital market	Levenberg-Marquardt Algorithm	To predict the stock market
(Min, Xiao, Cao, & Yan, 2017)	Medical	Levenberg-Marquardt Algorithm	To compare the predictive accuracy between Backpropagation and LMA in the diagnosis of breast cancer
(Djelloul, Sari, & Sidibe, 2018)	Manufacturing management	Levenberg-Marquardt Algorithm	An ANN model for fault detection and diagnosing problems in manufacturing systems

2.6 Chapter Summary

This chapter discusses in detail the review of the existing literature on the recruitment and selection process of employee/programmer. It expounds on the use of Bayes' Theorem for solving problems in various domains and subjects, and the use of Artificial Neural Network (ANN) technique and its training Levenberg-Marquardt Algorithm (LMA) for solving nonlinear least squared problems in various disciplines.

Human resource is one of the crucial factors in software development. Selecting the right person for the right job ensures better outcome for the organisations. It is important for a software organisation to select an appropriate recruitment and selection process/method in order to recruit the best-fit employee/programmer. Determining the right set of eligibility criteria is the most important step for success in recruiting the best-fit employee/programmer. Although there are traditional methods for determining the right recruitment criteria such as job description, job specification, job requirements, there are other issues that need to be solved beforehand to ensure fairness in the recruitment process. Determining the appropriate set of recruitment criteria from the previous employee job performance records in the organisation database would be useful to determine the eligibility criteria to assess a new candidate. Suitable eligibility criteria are defined based on the job performance record.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

This chapter explains the different phases, activities and methods used to carry out the research to achieve the objectives. The four phases of the research are outlined in Figure 3.1, and discussed briefly in the following sub-sections.

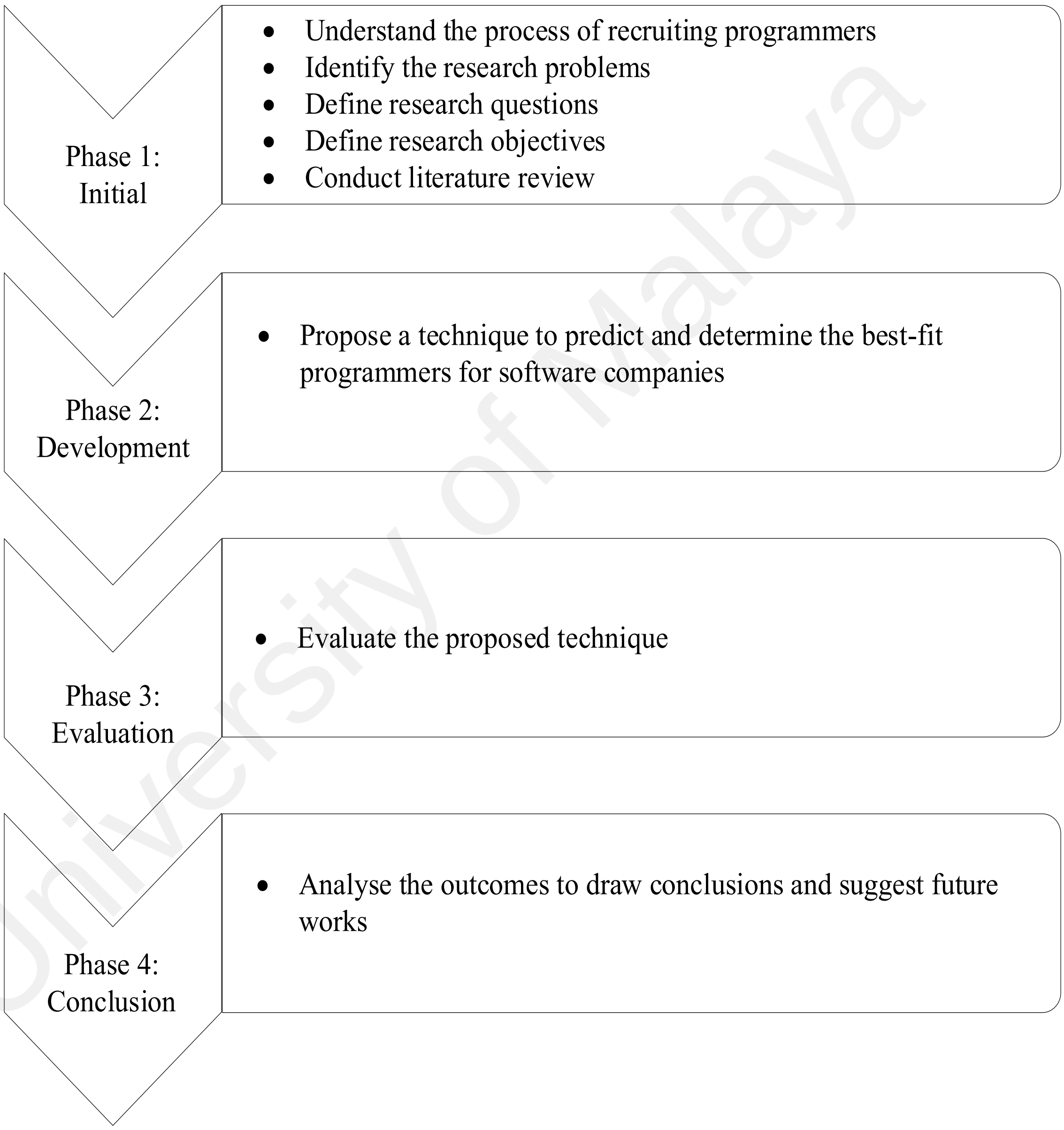


Figure 3.1: The Research Phases and Activities

3.2 The Research Phases

3.2.1 Phase 1: Initial

This phase involves gathering information on the recruitment process of programmers in order to understand the problems faced. This helps to define the research problem for this study. A literature review will also be conducted on issues related to the recruitment of programmers by software companies. These issues include the approaches and techniques used in recruiting programmers, and the problems encountered in the recruitment process. This will help to identify the weaknesses in the existing recruitment process. The research questions and objectives will then be defined to address the weaknesses.

3.2.2 Phase 2: Development

This phase involves the development of a technique to predict and determine the best-fit programmers to meet the needs of software companies. The development of the proposed technique is divided into two phases, which are explained in detail in chapter 4.

3.2.3 Phase 3: Evaluation

The accuracy of the proposed technique will be evaluated by comparing the best-fit programmers determined based on their past annual performance appraisal with the predicted best-fit programmers determined by the proposed technique. The evaluation will be conducted using the datasets provided by two software companies in India. These datasets consist of the past annual performance appraisal of the programmers working in the two software companies from 2010-2015. The best performing programmers based on the annual performance appraisal will be compared to determine whether they matched the best-fit programmers predicted by the proposed technique. In addition, the standard measurement matric called Confusion Matrix (Han, Kamber, & Pei, 2011) will be used to measure accuracy, precision, and recall of the predictions made by the proposed technique.

3.2.4 Phase 4: Conclusion

This phase presents a summary of the proposed technique, discusses the weaknesses and limitations of this research as well as makes recommendations for future study.

3.3 An Overview of the Workflow of the Proposed Technique

Figure 3.2 shows the workflow of the proposed technique, which consists of three main components. The first component is data acquisition and processing, which involves processing and manipulating the data on the programmers' annual performance appraisal and converting the data into applicable datasets for storage in a data warehouse. The second component involves the development of a technique to predict and determine the best-fit programmers to meet the needs of software companies. The proposed technique incorporates the use of Bayes' Theorem and Artificial Neural Network (ANN). The third component involves assessing the accuracy of the proposed technique in predicting the best-fit programmers. These three components are elaborated in the following sub-sections.

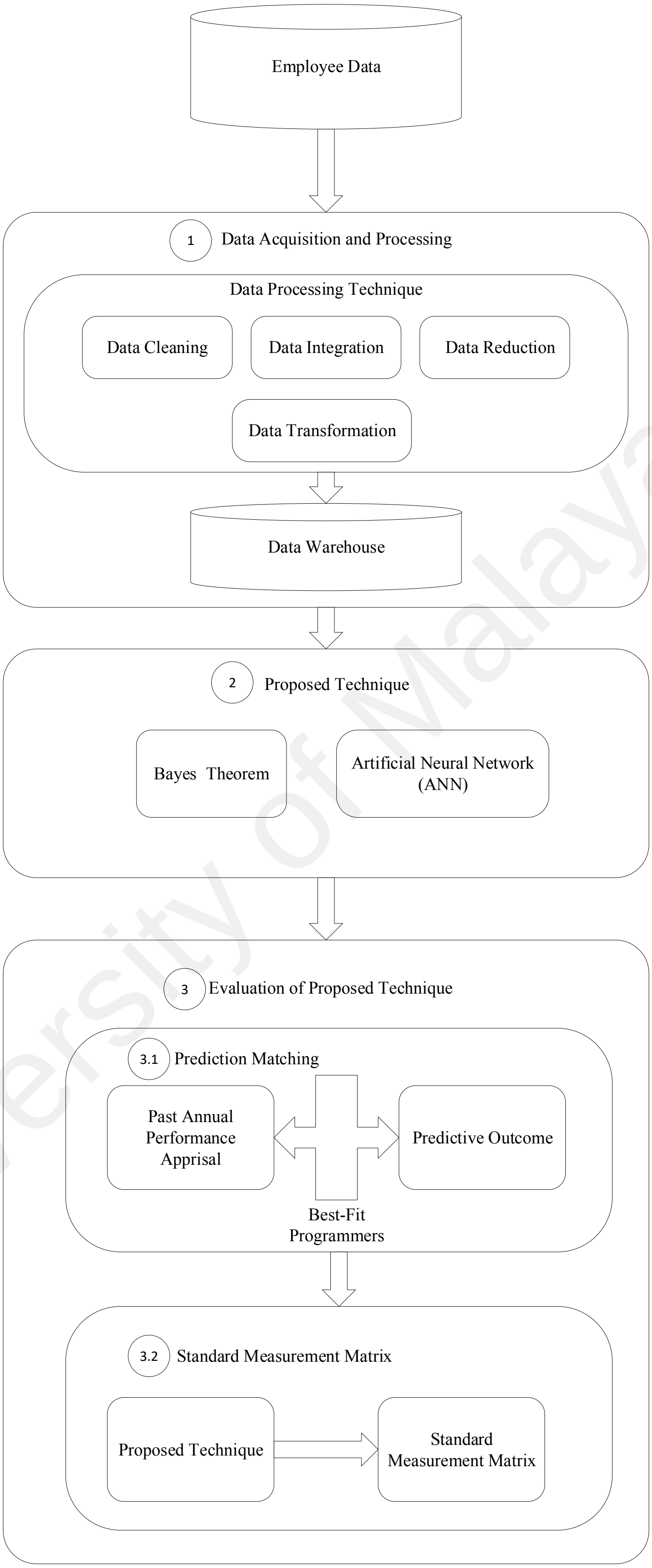


Figure 3.2: Workflow of the Proposed Technique

3.3.1 Data Acquisition and Processing

The first component of the workflow is data acquisition which is the extraction of the relevant subsets of data, which in this research, include information on the programmers' annual performance appraisal from the two companies. The data are first processed to eliminate inaccurate, noisy, incomplete and inconsistent information to ensure data quality, thereby contributing towards the accuracy of the predicted results (Chakrabarti, et al., 2009). The data processing techniques include data cleaning, data integration, data reduction and data transformation (Han, Kamber, & Pei, 2011). These processing techniques are explained in the sub-sections below.

3.3.1.1 Data Cleaning

The original data often contains many 'unclean' data, such as inconsistent values, duplicate data records, incorrect attribute value type, null value, use of various abbreviations, and so forth (García, Luengo, & Herrera, 2015). 'Dirty' employee data obtained from the software companies is cleaned using the following steps:

Step 1: Eliminating unclean data: There are 2,468 employee records from 2010-2015 in Company 1. However, only 2,230 records are in the annual employee performance appraisal table. Therefore, 238 (2,468-2,230) employee records do not contain information on their performance appraisal and are not used for data analysis. On the other hand, Company 2 has 2,653 employee records, but only 2,222 records are in their annual employee performance appraisal table. Thus, 431 (2,653-2,222) employee records that do not contain their performance appraisal are not used in data analysis.

Step 2: Dealing with null value: Data tables in the employee annual performance appraisal database that have null values in certain evaluation criteria fields, are replaced by the average score of that evaluation criteria field.

Step 3: Amending incorrect data: The data is analysed to explore the inconsistency of the value in the field. In the employee performance appraisal database, there were inconsistent values in some of the evaluation criteria fields. For example, the value in the first evaluation criterion should contain a score of between 1 and 10, but in some tuples (rows), the first evaluation criterion has a score of 70. It could have been a human or computer error which occurred during data entry. Hence, all the invalid data records (rows) were removed.

Step 4: Eliminate duplicate data: This is done by applying the “DISTINCT” keyword to remove all the duplicate data from the employee performance appraisal database. For instance, duplicate records for the same employee were found in an annual employee performance appraisal table. For this reason, “DISTINCT” was used to eliminate duplicate records and retain only one employee record.

3.3.1.2 Data Integration

The data collected from the two software companies were stored in different formats. In Company 1, the employee annual appraisal data from 2010-2011 were stored in MS EXCEL spreadsheets, but the data from 2012-2015 were stored in a database, as shown in Figure 3.3 (a). Similarly, in Company 2, the employee annual appraisal data from 2010-2012 were stored in MS EXCEL spreadsheets, and data from 2013-2015 were stored in a database, as shown in Figure 3.3 (b). In this research, all the data which are stored in different formats were merged into one data file for further processing. Merging the data from two heterogenous data formats poses a challenge to the researcher. For example, in the MS EXCEL spreadsheet, Attribute A is actually “Ability to read and write technical document”, whereas Attribute A in the annual employee performance appraisal table is the “Document Code”. This makes it difficult to determine which Attribute A should be stored in the database. Hence, metadata or "data about data" was applied when merging the data to reduce inconsistent data.

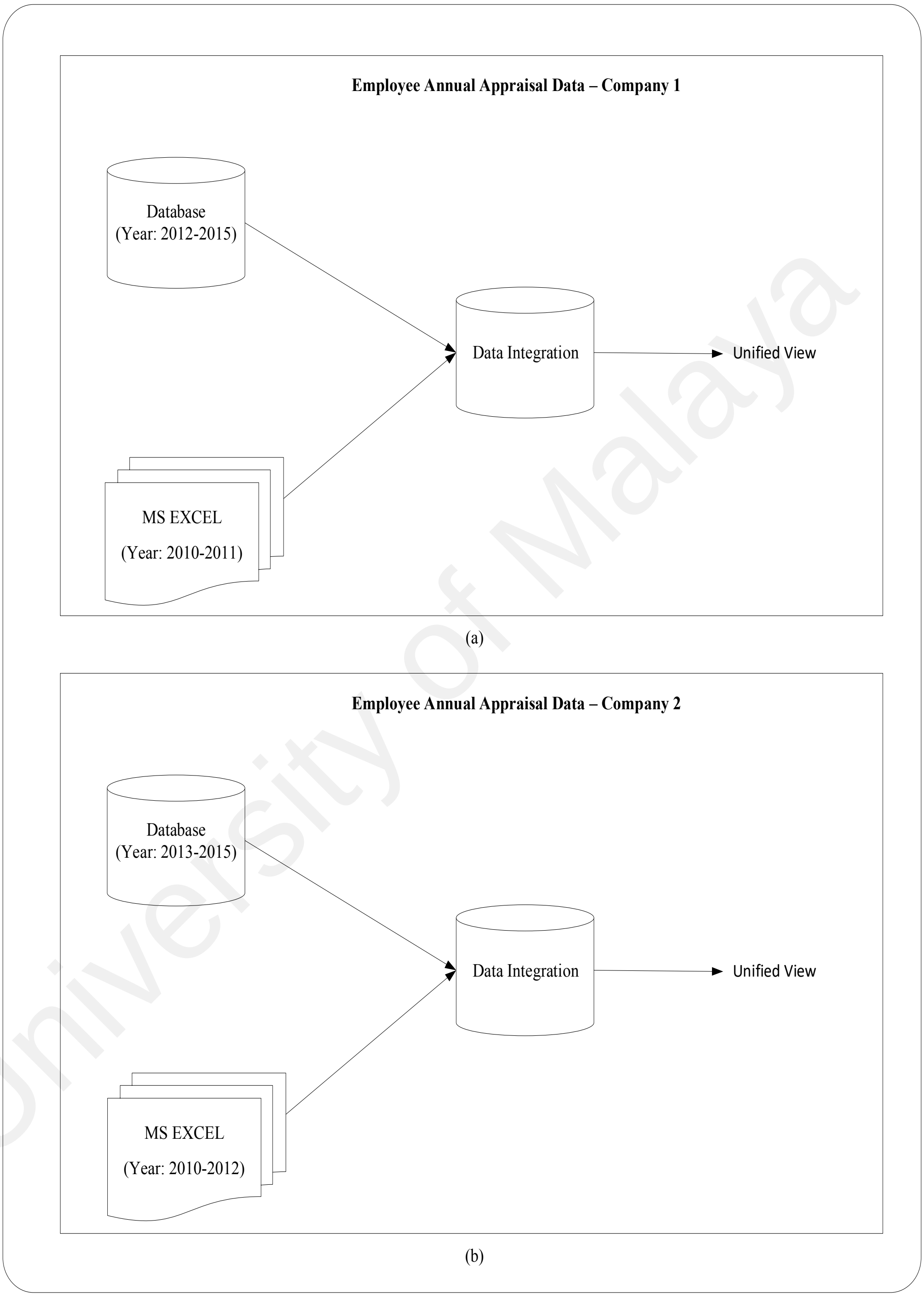


Figure 3.3: Data Integration of Employee Annual Appraisal Data (Company 1 and Company 2)

3.3.1.3 Data Reduction

The data collected from the two software companies include information on the annual performance appraisal for each programmer. Hence, programmers who have been working in the companies for more than one year will have more than one annual performance appraisal records stored in the database. As this research is concerned with the overall performance of a programmer (single record data) rather than specific yearly performance, the average score of all the annual performance appraisal results is used. Table 3.1 shows the original dataset of the annual performance appraisal of the programmers ($P_1 \ldots P_n$) from 2011-2015. This data reduction process is aimed at improving the accuracy of determining and predicting the best-fit programmers and also in improving storage efficiency and in reducing processing time, while maintaining the integrity and structure of the original datasets (García, Luengo, & Herrera, 2015).

Table 3.1: Programmers' Annual Performance Appraisal

Programmer ID	Year	A1	A2	A3	…	A_n
P1	2011	7	3	5	…	9
P1	2012	8	4	6	…	8
P1	2013	7	3	7	…	7
P2	2012	6	2	7	…	7
P2	2013	7	3	8	…	6
P3	2014	9	4	6	…	7
P3	2015	8	3	7	…	6
…	…	…	…	…	…	…
P_n	2015	8	4	6	…	7

A1: Attribute 1 (refers to the assessment criterion number 1 of programmers' annual performance appraisal)

A2: Attribute 2 (refers to the assessment criterion number 2 of programmers' annual performance appraisal)

A3: Attribute 3 (refers to the assessment criterion number 3 of programmers' annual performance appraisal)

$\mathrm{A_n}$*: Attribute n (refers to the assessment criterion number n of programmers' annual performance appraisal)*

Programmer P1 has three annual performance appraisal records from 2011-2013. Similarly, programmers P2, P3 until P_n also have multiple annual performance appraisal records, respectively. The average score of each attribute in the annual performance appraisal record of each programmer is calculated and presented in Table 3.2.

Table 3.2: Average Score of Programmers' Annual Performance Appraisal

Programmer ID	A1	A2	A3	…	A_n
P1	7.3	3.3	6	…	8
P2	6.5	2.5	7.5	…	7
P3	8.5	3.5	6.5	…	6.5
…	…	…	…	…	…
P_n	8	4	6	…	7

Hence, the original dataset shown in Table 3.1 has been compressed and reduced to a smaller data size, as shown in Table 3.2. This improves data processing time and at the same time reduces the size for data storage.

3.3.1.4 Data Transformation

The quality of data depends on how data meet its intended use in a given situation or context (Han, Kamber, & Pei, 2011). Data transformation is a data processing technique for improving data to make it more effective for determining and predicting the best-fit programmers in this research. Using this technique, the data is converted or consolidated into applicable forms for the intended use in specific data mining algorithms, by performing normalisation, discretisation or aggregation operations. In this research, data discretisation was applied on the datasets collected from the two software companies in India. The purpose of data discretisation is to transform the discrete value of each attribute into a set of intervals (e.g., 0–5, 6–10, etc.) or class label (e.g., low, medium, high, etc.). Since Bayes' Theorem is applied to handle the nominal and ordinal attributes, discretisation is applied on the two datasets. To estimate the probability of each performance attribute, the discrete value of each attribute of the programmer performance appraisal is transformed into ordinal type. Figure 3.4 shows the programmer ID, P_1 until P_{10} and their respective performance attributes, Attribute 1 until Attribute $_n$. The score (numerical value) of each attribute is recorded, respectively.

Performance Appraisal Attribute \ Programmer ID	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Attribute 1	7	8	5	9	6	15	11	12	9	13
Attribute 2	9	11	8	6	7	10	13	8	15	10
Attribute 3	13	7	9	8	10	12	15	13	4	12
Attribute 4	12	8	7	15	7	9	11	12	9	13
Attribute 5	3	6	8	11	12	9	6	8	14	9
...	...	...	...	...	...	...	...	...	...	...
Attribute n	10	9	12	14	6	15	11	12	9	13

Figure 3.4: The Value of Each Attribute in Programmer Performance Appraisal

The unordered values of attribute 1 for programmers P_1-P_{10} are: 7, 8, 5, 9, 6, 15, 11, 12, 9, 13, 10, respectively. The sorted values of attribute 1 shown in Figure 3.5 are: 5, 6, 7, 8, 9, 9, 10, 11, 12, 13, 15. These numerical values are partitioned into a set of ranges. Each range is treated as an ordinal attribute (order of values). The equal-width binning technique in data discretisation is then applied on these values to divide them into N equal width of intervals. The width of an interval is calculated using the following formula:

$$width = (max - min)/N$$

Where:

- max is the highest value among all the attribute values.
- min is the lowest value among all the attribute values.
- N is the specified number of bin.

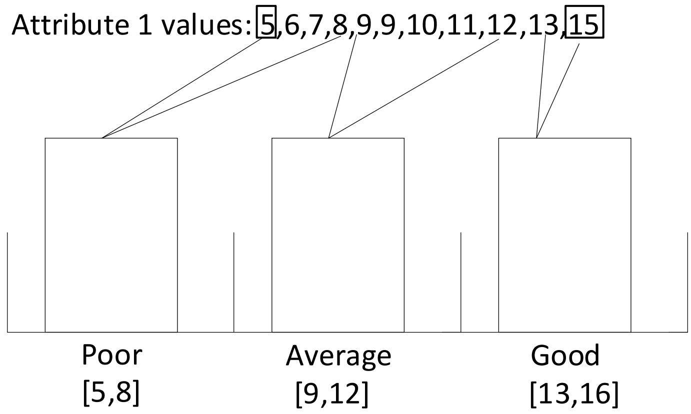


Figure 3.5: Equal-Width Binning

The highest value of Attribute 1 is 15, and the lowest value is 5. This research uses three groups (three bins) called Poor, Average, and Good on the range of values, the width of Attribute1 is thus rounded to 3, as illustrated below in Bin 1 to Bin 3. The number of bin represents the scores' class of the programmer appraisal performance.

Bin 1 (Poor): 5-8	all values between 5 and 8.
Bin 2 (Average): 9-12	all values between 9 and 12.
Bin 3 (Good): 13-16	all values between 13 and 16.

The discretised data from numerical values into ordinal types are shown in Figure 3.6. For example, Attribute 1 of P1 is transformed from the value 7 into "Poor" (Bin1). The values of the other attributes are also transformed in the same manner.

Performance Appraisal Attribute \ Programmer ID	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Attribute 1	Poor	Poor	Poor	Average	Poor	Good	Average	Average	Average	Good
Attribute 2	Average	Average	Poor	Poor	Poor	Average	Good	Poor	Good	Average
Attribute 3	Average	Poor	Average	Poor	Average	Average	Good	Average	Poor	Average
Attribute 4	Average	Average	Poor	Good	Poor	Average	Average	Average	Average	Good
Attribute 5	Poor	Poor	Average	Average	Average	Average	Poor	Average	Good	Average
...	...	...	...	...	...	...	...		...	...
Attribute n	Average	Average	Average	Good	Poor	Good	Average	Average	Average	Good

Figure 3.6: The Ordinal Type of Each Attribute of the Programmer Performance Appraisal

3.3.1.5 Data Warehouse

Data warehouse is the last step of data acquisition and processing components of the workflow in this research. The results of data processing – data cleaning, data integration, data reduction and data transformation - are consistent, and the quality data is stored in the data warehouse for further use.

3.3.2 Proposed Technique

In the proposed technique, Bayes' Theorem is applied on the data stored in the warehouse. Artificial Neural Network (ANN) with Levenberg-Marquardt (LM) training algorithm together with the Multilayer Perceptron (MLP) neural network architecture is developed to predict and classify the programmers' performance and assessed against the attributes.

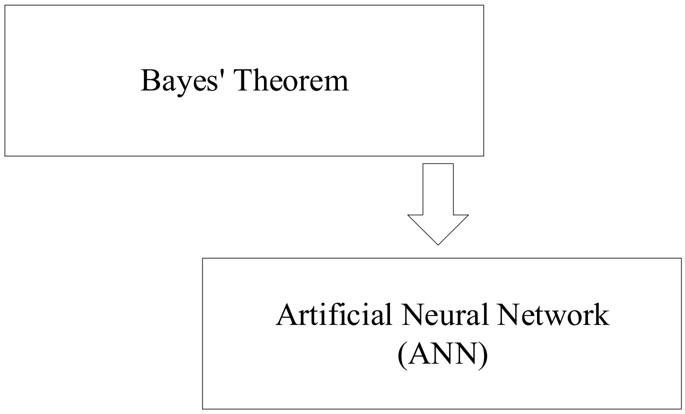


Figure 3.7: Two Algorithms Applied in the Proposed Technique

As shown in Figure 3.7, Bayes' Theorem is used to find the prognostic attributes – the attributes are ranked according to the most important attributes pertaining to work performance of programmers. Artificial Neural Network (ANN) with Levenberg-Marquardt (LM) algorithm is applied on the results produced following the application of Bayes' Theorem. ANN is used to refine the weightage of each attribute, in order to shortlist candidates who are qualified for the job. Detailed explanation on the proposed technique is presented in Chapter 4.

3.3.3 Evaluation of Proposed Technique

The evaluation process of the proposed technique involves:

(i) Comparing the predicted results with the annual performance appraisal of the programmers; and

(ii) Applying a standard measurement Metric to measure the performance of ANN constructed classifiers.

3.3.3.1 Comparison of the Predicted Results with Annual Performance Appraisal

The predicted outcome of the proposed technique in determining and predicting the best-fit programmers, is evaluated by comparing the predicted results with the results of the past annual performance appraisal of the programmers.

3.3.3.2 Standard Measurement Metric

To determine the performance of the proposed technique in predicting the best-fit programmers, a standard measurement metric, also known as a confusion matrix, can be applied to measure the performance of the proposed technique's ANN classifier. There are three standard indicators in the confusion matrix for evaluating the performance of the ANN classifier in terms of accuracy, precision, and recall (Catal, 2012). These indicators are calculated using the following formulas:

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

$$Precision = \frac{\mathrm{TP}}{TP + FP}$$

$$Recall = \frac{\mathrm{TP}}{TP + FN}$$

Whereby,

- True Positive (TP) indicates that the proposed technique predicts the programmer's performance class value correctly, and that the actual class value of the programmer's performance is valid. For example, the actual class value of programmer's performance is good and the proposed technique also predicted it as good.
- True Negative (TN) indicates that the proposed technique predicts the programmer's performance class value correctly, and that the actual class value of the programmer's performance is invalid. For example, the actual class value of programmer's performance is not good and the proposed technique predicted it as not good.

- False Positive (FP) indicates that the proposed technique predicts the programmer's performance class value incorrectly, and that the actual class value of the programmer's performance is invalid. For example, the actual class value is not good but the proposed technique predicted it as good.
- False Negative (FN) indicates that the proposed technique predicts the programmer's performance class value incorrectly, and that the actual class value of the programmer's performance is valid. For example, the actual class value is good but the proposed technique predicted it as not good.

The accuracy of the proposed technique is the ratio of the total number of programmer performances' classes that are correctly classified by the proposed technique. Precision is the proportion of predicted programmer performance's classes that are predicted correctly as it actual programmer performance's classes. On the other hand, recall is the proportion of the actual classes of programmer performance that are correctly predicted programmer performance's classes. Precision and recall are measured based on each individual class. The confusion matrix for assessing the performance of the proposed technique using the four parameters (TP, TN, FP and FN) is shown in Table 3.3.

Table 3.3: Confusion Matrix for Measuring the Performance of the Proposed Technique

<table>
<tr><td rowspan="4">Predicted Class</td><td colspan="3">Actual Class</td></tr>
<tr><td></td><td>Valid</td><td>Invalid</td></tr>
<tr><td>Valid</td><td>TP</td><td>FP</td></tr>
<tr><td>Invalid</td><td>FN</td><td>TN</td></tr>
</table>

Actual class is the real class value of the programmer's performance from the software companies' datasets. Predicted class is the predicted class value of the programmer's performance from the proposed technique. Moreover, the three standard indicators - Accuracy, Precision, and Recall - are computed from the confusion matrix, with the values ranging between 0 and 1 and the value has significant implication. For example, if the accuracy value is close to 1, it implies that the proposed technique's ANN classifier is more accurate in classifying the programmer performance's class. Similarly, the other two standard indicators - Precision and Recall - are calculated in the same way. The evaluation on the proposed technique using the confusion matrix is elaborated further in Chapter 5.

CHAPTER 4: PROPOSED TECHNIQUE

4.1 Introduction

This chapter provides an in-depth explanation on the proposed technique to address the problem statements defined in this research. It begins with an overview section, whereby the proposed technique is briefly explained. Each step in the proposed technique is then further elaborated. The following section will describe the system implementation of the proposed technique. Finally, Section 4.4 summarises the contents of this chapter.

4.2 Overview

This research is aimed at providing a technique for identifying the pertinent attributes that affect the programmers' work performance and for predicting programmers who are most likely to meet the job requirements of software companies. The proposed technique consists of three main steps, as shown in Figure 4.1. The initial step is to analyse the programmers' performance attributes using Bayes' Theorem for the purpose of identifying the prognostic attributes (high potential attributes), which can affect the work performance of programmers. The second step is to extract the attribute value obtained in the previous step, to determine the best-fit candidate for a software company. The third step is to extract the prognostic attributes identified from step 2 to be the input into the Artificial Neural Network (ANN) for predicting and determining the performance of new programmers. These three steps are illustrated in Figure 4.1, below.

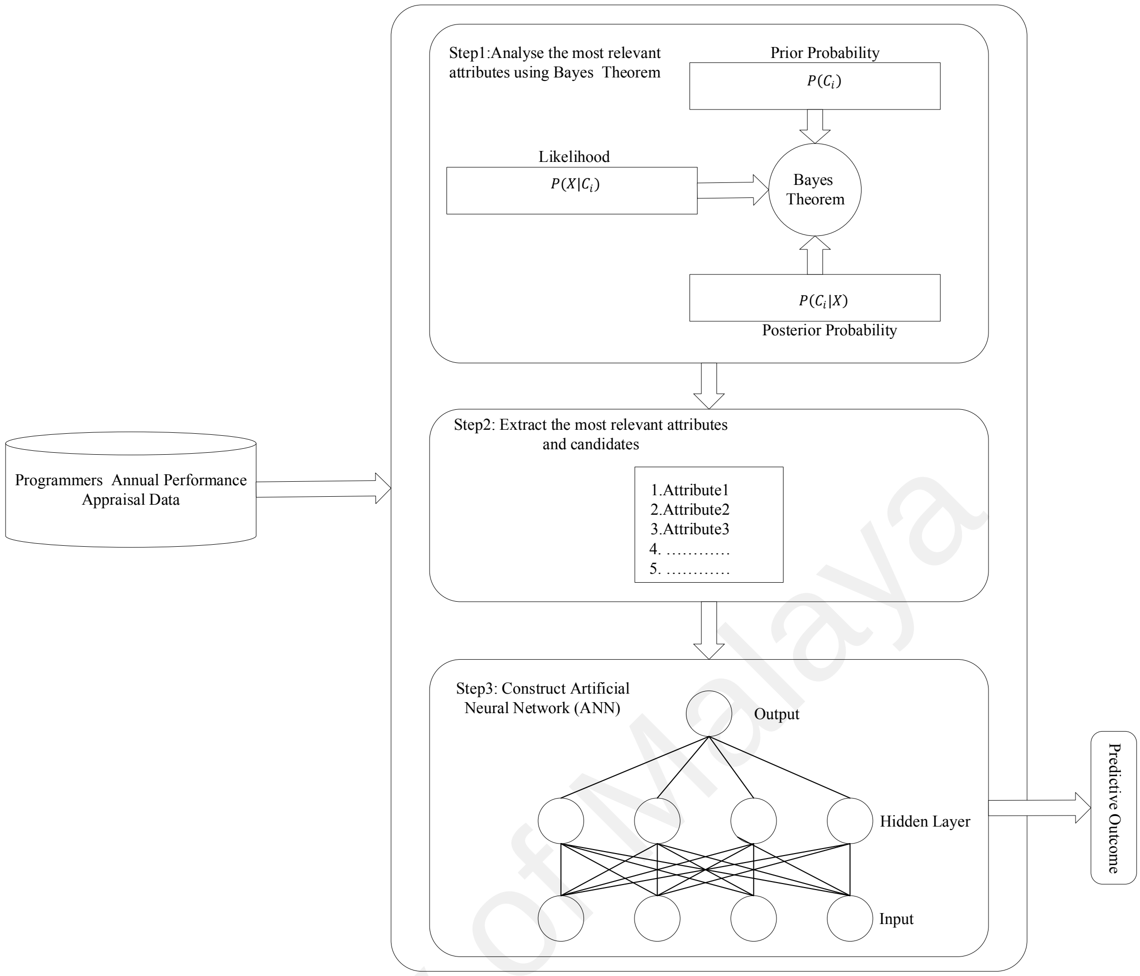


Figure 4.1: Steps of the Proposed Technique

4.2.1 Step 1: Analyse the most relevant attributes using Bayes' Theorem

The initial step of the proposed technique is to analyse the programmers' performance attributes using information from their annual performance appraisal records obtained from the software companies. This is done in order to determine those attributes that would most likely affect the performance of programmers. In this step, the Bayes' Theorem is used to analyse the programmers' performance attributes. Bayes' Theorem is very suitable for estimating the possible outcome of a hypothesis by inferring the evidence (a set of data) in the case of known knowledge or experience (Stone, 2015). Bayes' Theorem is defined by the following equation:

$$P(A|C_i) = \frac{P(C_i|A) * P(A)}{P(C_i)} \qquad (1)$$

In the equation, $P(A|C_i)$ is the posterior probability which refers to the probability of attribute value A, given that the programmers' work performance is from class i (C_i). $P(C_i|A)$ is the likelihood or chance a programmer will achieve performance of class i (C_i), given that it has attribute value A. $P(A)$ is the prior probability which is the fundamental belief of a chance of getting attribute A, whereas, $P(C_i)$ is the chance that a programmer is from class i (C_i). The results of applying Bayes' Theorem on the data collected from the two software companies (Company 1 and Company 2) in India are presented in Table 4.1 and Table 4.2, respectively.

Table 4.1: Results of Applying Bayes' Theorem (Company 1)

Attributes	Attributes Values	Probability $P(A\|C_i)$		
		Good	Average	Poor
Attribute 1: Write Functionally Correct Code	Good	0.770	0.483	0.261
	Average	0.198	0.394	0.348
	Poor	0.033	0.123	0.391
		Good	Average	Poor
Attribute 2: Writes Aesthetically Pleasing Code	Good	0.640	0.256	0.043
	Average	0.279	0.540	0.402
	Poor	0.082	0.205	0.554

Table 4.1: (Continued)

		Good	Average	Poor
Attribute 3: Performs Satisfactory Unit Test	Good	0.590	0.306	0.228
	Average	0.344	0.565	0.598
	Poor	0.066	0.130	0.174
Attribute 4: Documents Code Well	Good	0.738	0.442	0.141
	Average	0.246	0.505	0.467
	Poor	0.016	0.054	0.391
Attribute 5: Asks Questions When Needed	Good	0.738	0.252	0.217
	Average	0.230	0.435	0.457
	Poor	0.033	0.312	0.326
Attribute 6: Communication Skills	Good	0.705	0.464	0.250
	Average	0.279	0.284	0.315
	Poor	0.016	0.252	0.435
		Good	Average	Poor
Attribute 7: Punctuality	Good	0.230	0.284	0.240
	Average	0.393	0.363	0.402
	Poor	0.377	0.353	0.761

Table 4.1: (Continued)

		Good	Average	Poor
Attribute 8: Corporate responsibility	Good	0.525	0.429	0.315
	Average	0.344	0.278	0.315
	Poor	0.131	0.293	0.370
Attribute 9: Project innovation	Good	0.443	0.199	0.065
	Average	0.410	0.344	0.272
	Poor	0.148	0.457	0.663
Attribute 10: Leadership	Good	0.361	0.227	0.109
	Average	0.443	0.309	0.304
	Poor	0.197	0.464	0.587

Each value in Table 4.1 was calculated as follows. The probability of attribute 1 being "Good" and the programmers' work performance is also "Good" is denoted as $P(A = Good|C_{Good})$. Applying Bayes' Theorem, this can be expressed as:

$$P(A = Good|C_{Good}) = \frac{P(C_{Good}|A = Good) * P(A = Good)}{P(C_{Good})}$$

To find the $P(C_{Good}|A = Good)$, derive the formula for conditional probability as:

$$P(C_{Good}|A = Good) = \frac{P(C_{Good} \cap A = Good)}{P(A = Good)}$$

The Venn Diagram, below, illustrates: the $P(A = Good), P(A = Good)$ and $P(C_{Good} \cap A = Good)$.

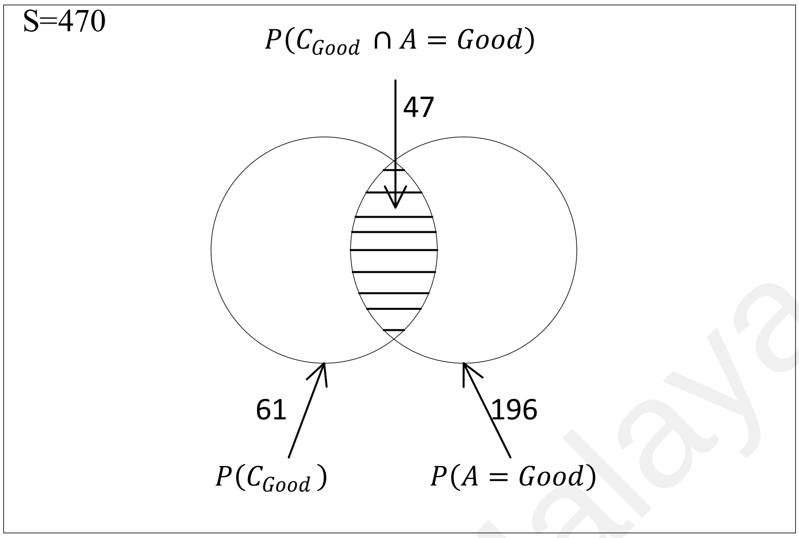


Figure 4.2: Venn Diagram of P(C$_{Good}$|A=Good)

Hence, $P(C_{Good}|A = Good) = \frac{47}{196} = 0.24$. The probability of $P(A = Good)$, from the Venn Diagram above is $\frac{196}{470}$ and that is equivalent to 0.417. Similarly, the probability of C_{Good} is $\frac{61}{470}$ which is equal to 0.13. Therefore, the value for $P(A = Good|C_{Good})$ is:

$$P(A = Good|C_{Good}) = \frac{P(C_{Good}|A=Good)*P(A=Good)}{P(C_{Good})} = \frac{0.24 * 0.417}{0.13} = 0.77$$

Similarly, the calculation for attribute 1 being “Average” and the programmers’ work performance is “Good”, i.e., $P(A = Average|C_{Good})$ is as follows:

$$P(A = Average|C_{Good}) = \frac{P(C_{Good}|A = Average) * P(A = Average)}{P(C_{Good})}$$

The Venn Diagram in Figure 4.3 illustrates the calculation for the probability, below:

$$P(A = Average|C_{Good}) = \frac{0.08*0.323}{0.13} = 0.198$$

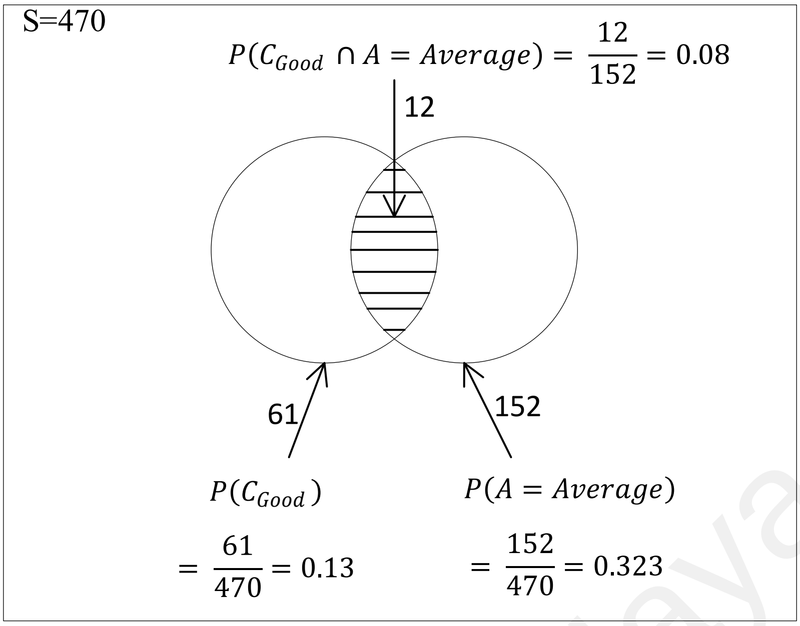


Figure 4.3: Venn Diagram of P(C_{Good}|A=Average)

All other values shown in Table 4.1 were also calculated in the same manner. The same way of calculating, was also applied on the data set obtained from Company 2, as shown in Table 4.2.

Table 4.2: Results of Applying Bayes' Theorem (Company 2)

Attributes	Attributes Values	Probability $P(A\|C_i)$		
		Good	Average	Poor
Attribute 1: Technical Skill	Good	0.645	0.390	0.2
	Average	0.197	0.396	0.470
	Poor	0.158	0.214	0.330
Attribute 2: Time Management	Good	0.789	0.146	0.043
	Average	0.211	0.6	0.348
	Poor	0	0.254	0.609

Table 4.2: (Continued)

		Good	Average	Poor
Attribute 3: Documentation/ Presentation	Good	0.697	0.204	0.052
	Average	0.303	0.693	0.557
	Poor	0	0.104	0.391
Attribute 4: Teamwork/ Cooperative	Good	0.75	0.329	0.139
	Average	0.25	0.45	0.348
	Poor	0	0.221	0.513
		Good	Average	Poor
Attribute 5: Attitude and Self Growth	Good	0.684	0.336	0.052
	Average	0.316	0.461	0.287
	Poor	0	0.204	0.661

4.2.2 Step 2: Extract the most relevant attribute and candidate

The purpose of this step is to determine the attribute that has the highest probability (potential) related to the performance of a programmer. Bayes' Theorem is used to find the probability of the phenomenon where each attribute and the programmer's work performance occur. A probability of 1 indicates that a phenomenon will definitely happen, while a probability of 0 indicates that phenomenon will not happen. The chances that a phenomenon will happen is indicated by a value of between 0 and 1. Therefore, the larger the value, the higher the chances of getting or identifying programmers of that particular performance class. The most relevant attributes (prognostic attributes) are thus the extracted values which are greater than 0.5. Based on the values shown in Table 4.1 and

Table 4.2, the prognostic attributes of Company 1 and Company 2 are listed in Table 4.3 and Table 4.4, respectively.

Table 4.3: Prognostic Attributes of Company 1

	Attribute
Prognostic Attributes (High Potential Attributes)	Attribute 1: Write Functionally Correct Code
	Attribute 2: Writes Aesthetically Pleasing Code
	Attribute 3: Performs Satisfactory Unit Test
	Attribute 4: Documents Code Well
	Attribute 5: Asks Questions When Needed
	Attribute 6: Communication Skills
	Attribute 8: Corporate Responsibility

Table 4.3 shows that based on the values calculated using Bayes' Theorem (Table 4.1), seven out of the 10 attributes are the prognostic attributes. For example, the chance of a programmer being rated 'good' in "Write Functionally Correct Code" is 0.770, which is greater than 0.5. Similarly, the probabilities for "Writes Aesthetically Pleasing Code" (0.640), "Performs Satisfactory Unit Test" (0.590), "Documents Code Well" (0.738), "Asks Questions When Needed" (0.738), "Communication Skills" (0.705), and "Corporate responsibility" (0.525), are all greater than 0.5 and hence, they are among the list of prognostic attributes. However, the probabilities for the three attributes: "Punctuality" (0.230), "Project innovation" (0.443) and "Leadership" (0.361) are less than 0.5, implying that these attributes do not have any impact on a programmer's performance.

Similarly, the list of prognostic attributes of Company 2 is presented in Table 4.4. These attributes include "Technical Skill" (0.645), "Time Management" (0.789), "Documentation/Presentation" (0.697), "Teamwork/Cooperative" (0.75) and "Attitude and Self Growth" (0.684). This means that all the attributes are selected as the prognostic attributes of Company 2 as all the values are greater than 0.5 implying high probability of getting programmers who can perform well. For instance, the attribute "Time Management" (0.789) could be interpreted to mean that the programmers possess good time management attribute, and thus there is high possibility that they will achieve good performance. On the other hand, if a programmer is rated 'average' in time management (0.6), he/she is also expected to achieve average performance. Similarly, a programmer who is rated 'poor' in time management (0.609), is likely to show poor performance. A 0 value for the four attributes: "Time Management", "Documentation/Presentation", "Teamwork/Cooperative" and "Attitude and Self Growth" implies that if a programmer is rated 'poor' for these attributes, then his/her probability of showing good performance in these attributes is nil, respectively.

Table 4.4: Prognostic Attributes of Company 2

Prognostic Attributes (High Potential Attributes)	Attribute
	Attribute 1: Technical Skill
	Attribute 2: Time Management
	Attribute 3: Documentation/ Presentation
	Attribute 4: Teamwork/ Cooperative
	Attribute 5: Attitude and Self Growth

4.2.3 Step 3: Construct Artificial Neural Network (ANN)

The structure of the ANN with Multi-Layer Perceptron (MLP) network is shown in Figure 4.4. This step involves building the classifier for predicting and differentiating the programmers' performance based on the attributes. The prognostic attributes from the previous step are used to develop the ANN.

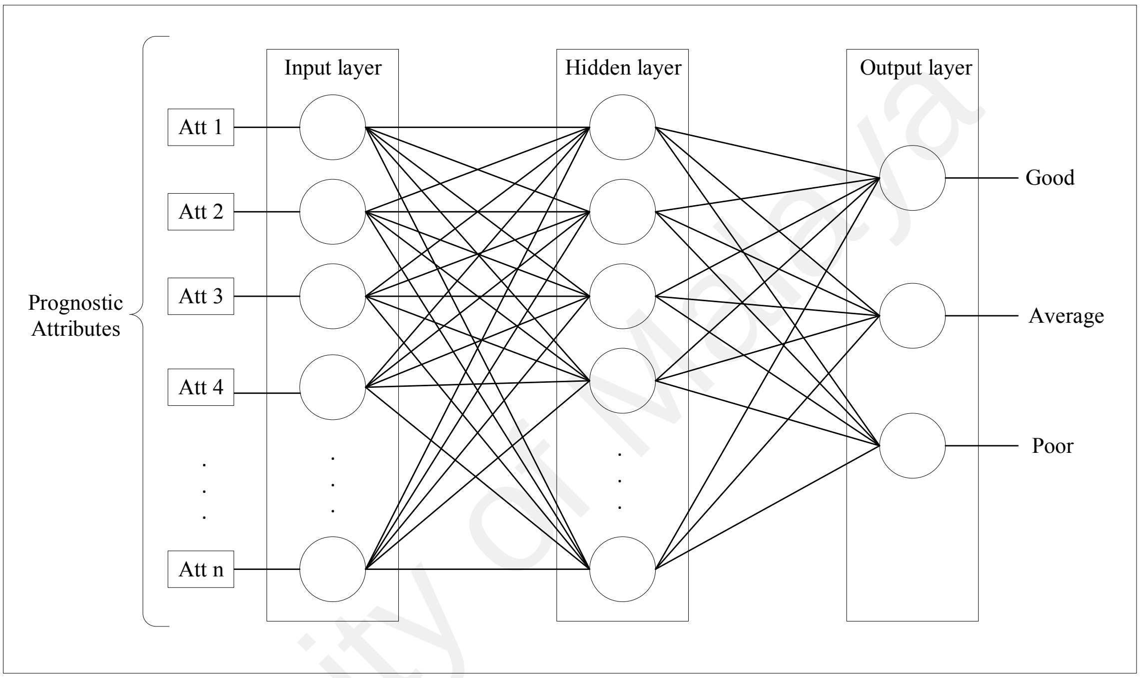


Figure 4.4: The Structure of Multi-Layer Feed-Forward Neural Network

Figure 4.4 shows the ANN structure of two datasets from Company 1 and Company 2. The structure consists of an input layer, a hidden layer, and an output layer. In the Company 1 dataset, the input layer has seven neurons which are the seven prognostic attributes, the hidden layer is set to contain 10 neuron nodes, and the output layer consists of three output vectors that represent the programmers' performance classes - "Good", "Average" and "Poor". In Company 2 dataset, the input layer has five neuron nodes, the hidden layer contains 10 neuron nodes, and the output layer consists of the same three output vectors of the programmers' performance classes as in Company 1.

The data used in constructing an ANN is divided into three sets - training set, validation set, and test set. In this research, the training set is a sample set that contains 70% of the total data and is used to adjust the weights of the neuron nodes to build an ANN classifier. The validation set is set at 15% of the total data to prevent the classifier from fitting closely to the training data. The test set is the remaining 15% of total data and is used for evaluating the performance of the developed trained classifier network using unseen data. Table 4.5 shows the results of the experiment conducted to split the data set. The test set was evaluated to ensure that the 70:15:15 ratio for training, validation, and test set, respectively, was followed, as this is the best ratio for predicting and determining the best-fit programmers based on their performance.

Table 4.5: Experiment on Split Data Set

Ratio	Set of Data	Observed Performance on Test Set (Company 1)	Observed Performance on Test Set (Company 2)
Experiment 1			
60	Training set	83.7%	90.5%
20	Validation set	78.7%	85.1%
20	Test set	86.2%	85.1%
Experiment 2			
70	Training set	86.3%	90.3%
15	Validation set	77.5%	78.9%
15	Test set	87.3%	87.3%
Experiment 3			
80	Training set	91.2%	89.7%
10	Validation set	80.9%	83.0%
10	Test set	85.1%	84.6%

4.3 System Implementation of the Proposed Technique

A tool to facilitate the proposed technique for predicting the best-fit programmers was developed using MATLAB. Figure 4.5 shows the user interface design when the system is used for predicting the best-fit programmer for Company 1. The interface has four parts - Load Data, Prognostic Attributes, Best-fit Programmers, and Classification.

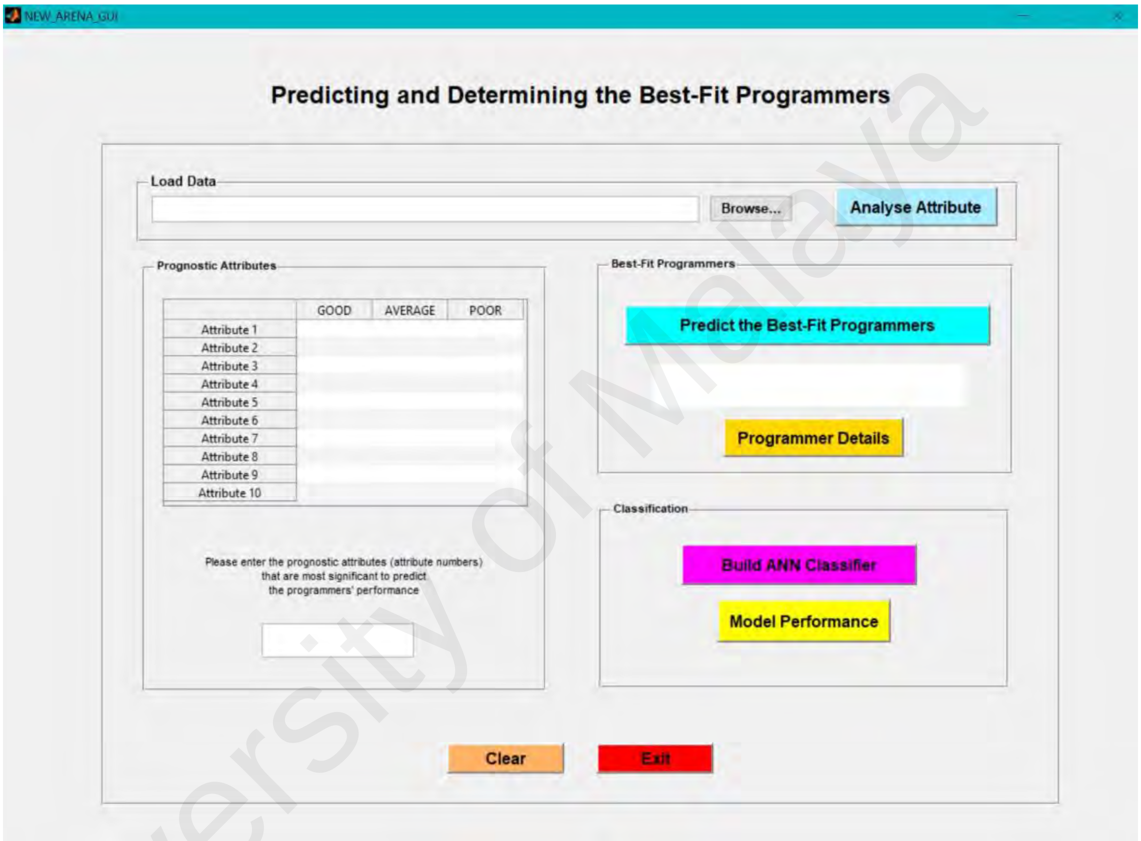


Figure 4.5: User Interface of the Predicting and Determining the Best-fit Programmers System (Company 1)

Figure 4.6 highlights the "Load Data" function, which is used for importing the dataset of Company 1 from a file path. When the "Analyse Attribute" button (located beside the "Load Data" part) is clicked, the backend code of the system will execute to analyse the most relevant attributes (prognostic attributes) using the Bayes' Theorem, explained above.

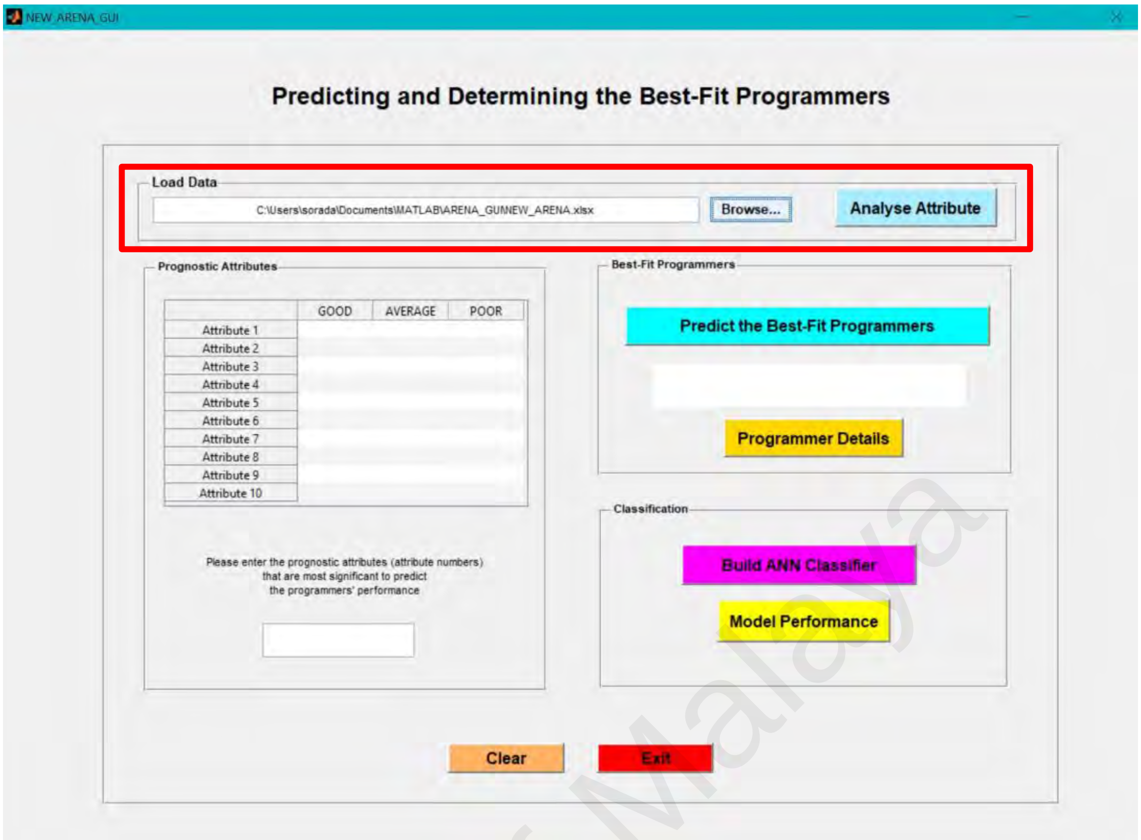


Figure 4.6: User Interface - Load Data Part (Company 1)

Figure 4.7 highlights the boxes in the user interface for analysing the "Prognostic Attributes" and determining the "Best-fit Programmers". The upper section of the "Prognostic Attributes" box tabulates the results of the analysed prognostic attributes. The lower section shows a textbox for inserting the prognostic attributes that are most significant (influential) on the programmers' work performance. In the "Best-fit Programmers" box, when the "Predict the Best-Fit Programmers" button is clicked, the programmers who show good performance in the prognostic attributes will be extracted and displayed (underneath the button) as "Programmer No.". The "Programmer Details" button when clicked, will show the details for each of the programmer displayed, above. In our research, Company 1, has seven prognostic attributes and three best-fit programmers.

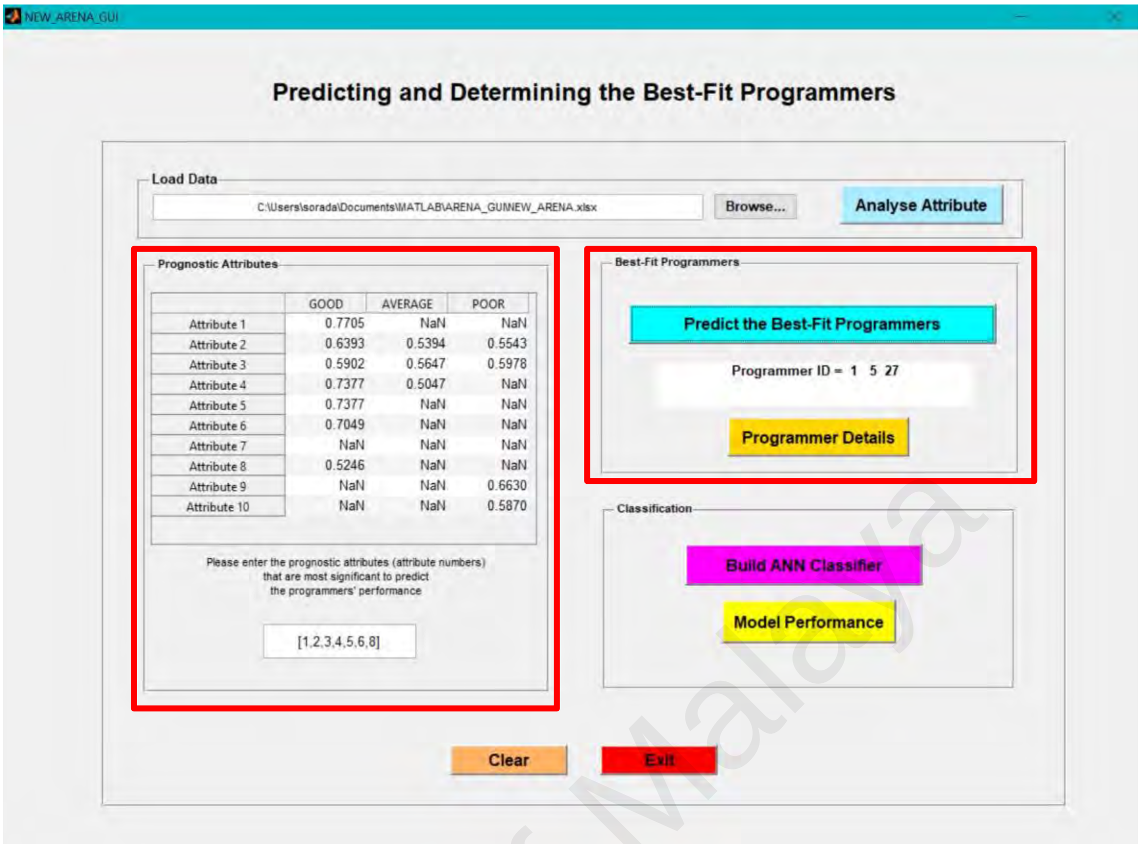


Figure 4.7: User Interface of Prognostic Attributes and Best-fit Programmers Analysing Parts (Company 1)

In the "Classification" box in Figure 4.8, clicking the "Build ANN Classifier" button enables the user to create an ANN classifier for predicting and determining the performance of programmers based on the assessing attributes.

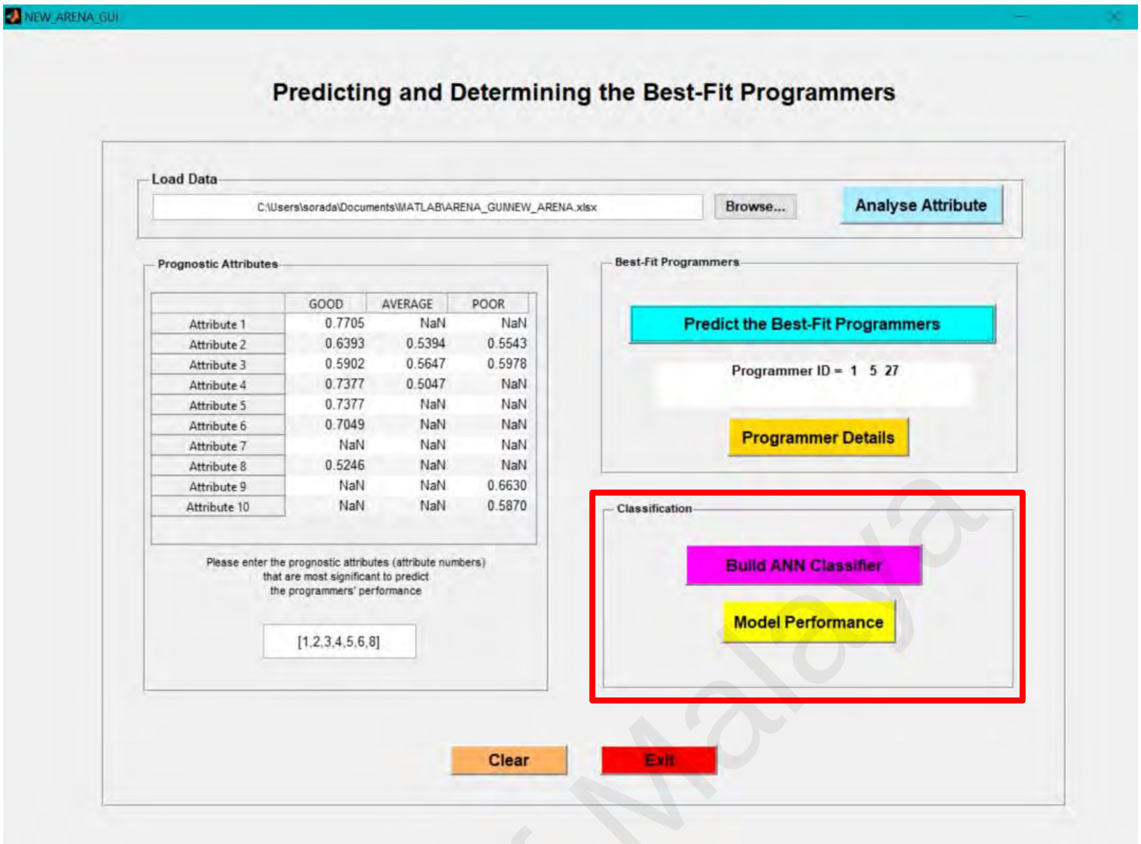


Figure 4.8: User Interface of Classification (Company 1)

Figure 4.9 shows the user interface for analysing the “Prognostic Attributes” and determining the “Best-Fit Programmers” for Company 2. There are five prognostic attributes and seven best-fit programmers for Company 2. Details on training an ANN classifier are presented below.

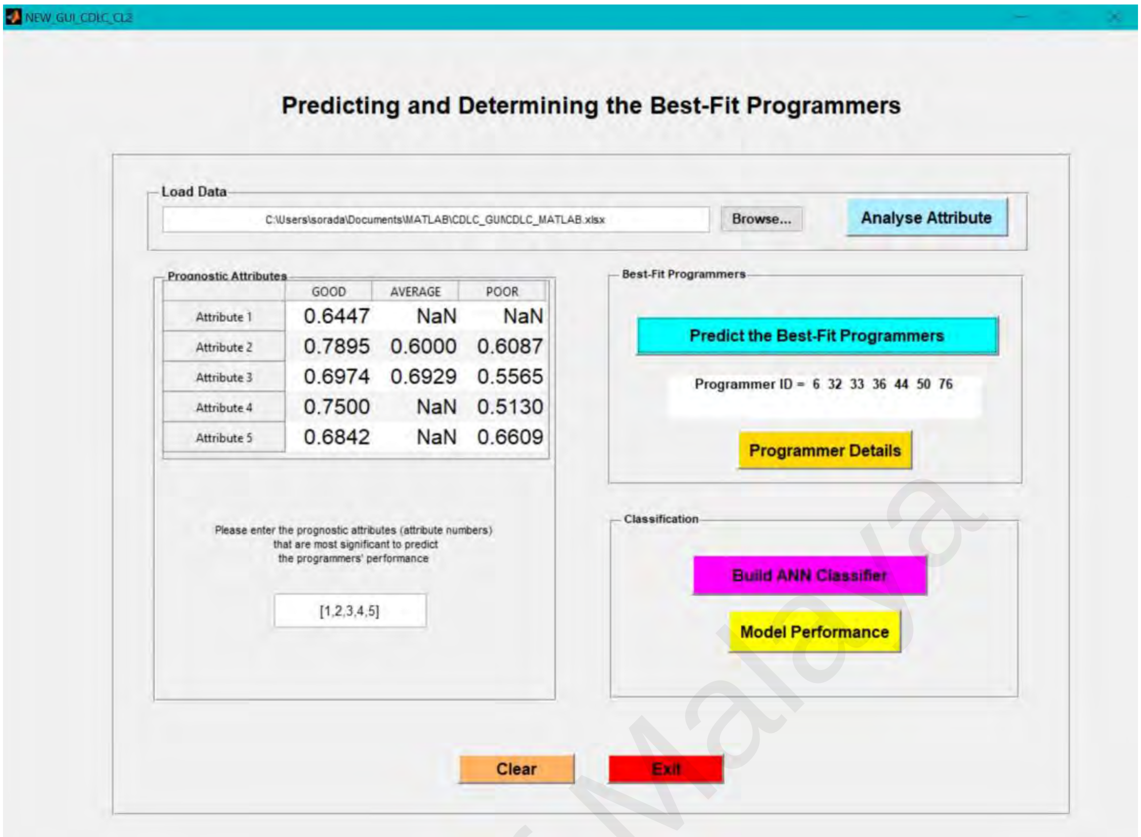


Figure 4.9: User Interface of Company 2

Figure 4.10 shows partial MATLAB codes used to train an ANN classifier in Company 1. The ANN classifier was constructed using pattern recognition network since this type of network is suitable for performing multi-class classification task. The dataset was split into three sets of data – 70% is allocated as the training set, and the remaining 30% is divided equally and allocated as the validation set (15%) and test set (15%), respectively.

```
% Input_prog_Atts - input data.
% target - target data.
%Input_data_prime
load('target.mat');
load('Input_prog_Atts.mat');
x = Input_prog_Atts;
t = target;
% Create a Pattern Recognition Network
hiddenLayerSize = 10;
net = patternnet([10, 20],'trainlm');
net.performParam.regularization = 0.01;
% Setup Division of Data for Training, Validation, Testing
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
% Train the Network
[net,tr] = train(net,x,t);
% Test the Network
y = net(x);

perf = perform(net,t,y);

e = gsubtract(t,y);
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);
performance = perform(net,t,y);
% View the Network
view(net)
% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, plotconfusion(t,y)
%figure, plotroc(t,y)
%figure, ploterrhist(e)
```

Figure 4.10: MATLAB Code for Training ANN Classifier of Company 1

The construction of ANN classifier for Company 2 is done in the same way as for Company 1. However, the difference between training the ANN for Company 1 and for Company 2 is in the number of neuron nodes in the hidden layers. Company 2 has two hidden layers - the first hidden layer has five neuron nodes while the second hidden layer has ten neuron nodes, as shown in the partial MATLAB codes in Figure 4.11.

```
% Input_prog_Att_C2 - input data.
% target_PPNET_CL2 - target data.
load('data_CL2.mat');
x = Input_prog_Att_C2;
t = target_PPNET_CL2;
% Create a Pattern Recognition Network
hiddenLayerSize = 10;
net = patternnet([5 10],'trainlm');
% Setup Division of Data for Training, Validation, Testing
net.divideParam.trainRatio = 80/100;
net.divideParam.valRatio = 10/100;
net.divideParam.testRatio = 10/100;
% Train the Network
[net,tr] = train(net,x,t);
```

Figure 4.11: Partial MATLAB Code for Training ANN Classifier of Company 2

Figure 4.12 shows the ANN training process for Company 1. The ANN structure consists of seven input neuron nodes, two hidden layers with 10 neuron nodes in the first hidden layer, and 20 neuron nodes in the second hidden layer. The output has three neuron nodes which represent the programmers' performance level.

The learning algorithm used to adjust the weight and bias values in this ANN training is based on the Levenberg-Marquardt algorithm together with the Mean Squared Error (MSE) for evaluating the performance of an ANN. Figure 4.12 shows the maximum number of iterations (Epoch) - 1,000 iterations - to train an ANN. The number of iterations can help in determining the cycle of iteration for updating the weights of the network. Details of the "Plots" (lower section of Figure 4.12) are elaborated in Figure 4.14.

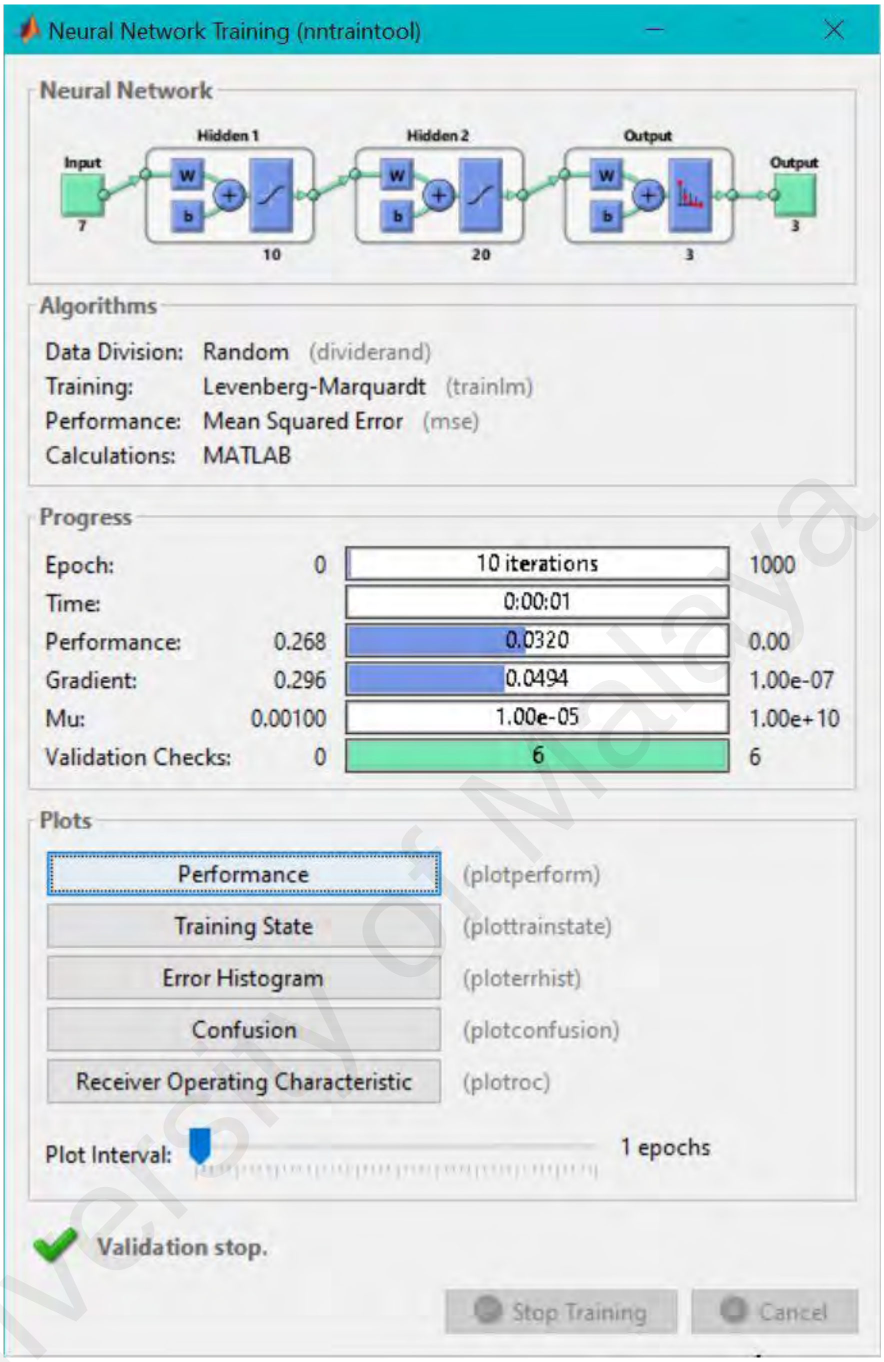


Figure 4.12: MATLAB's ANN Training Toolbox (Company 1)

Figure 4.13 shows the ANN training process in Company 2. The training process is the same as in Company 1 except for the number of neuron nodes in each layer. Company 2, has five input neuron nodes, two hidden layers with five neuron nodes in the first hidden layer and 10 neuron nodes in the second hidden layer. Similarly, there are three neuron nodes, which represent the programmers' performance level.

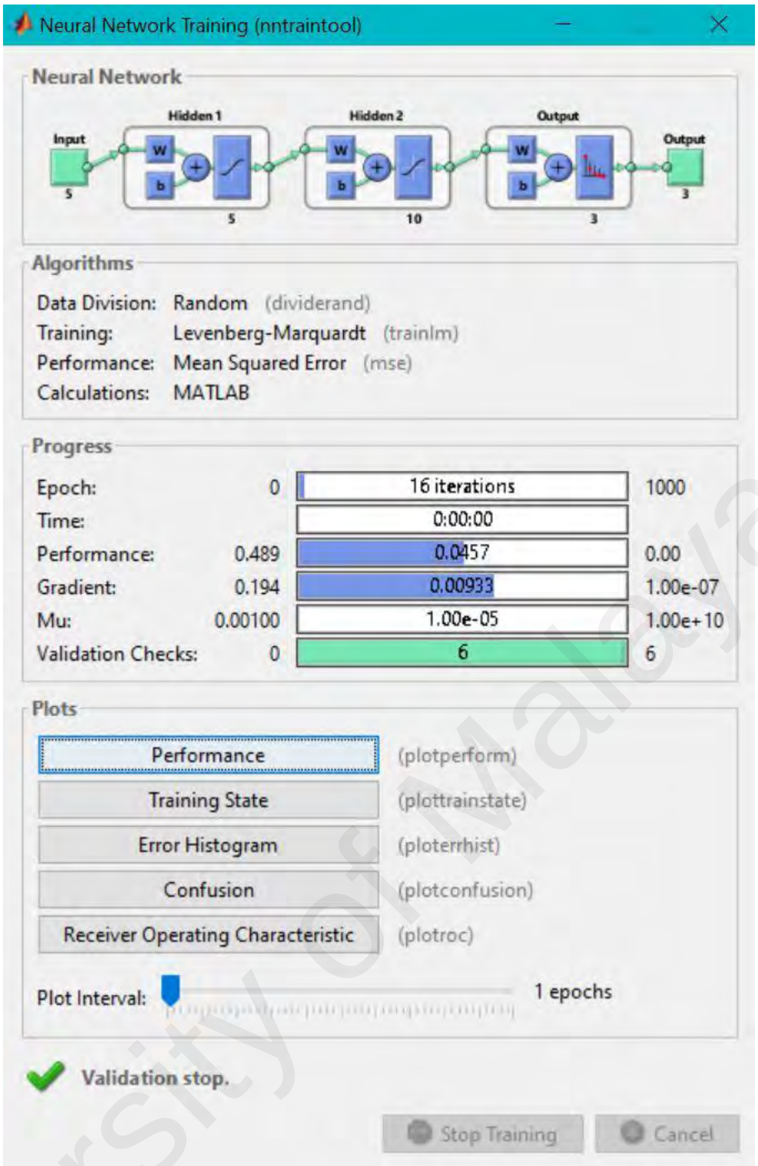


Figure 4.13: MATLAB's ANN Training Toolbox (Company 2)

Figure 4.14 shows the results of how an ANN performed. After iterative training of the ANN, the mean squared error (MSE) is reduced to the minimum value when an error of the validation dataset starts to increase. This indicates that ANN training has resulted in overfitting (over-training) the training dataset. By default, ANN training will stop when a validation error occurs up to six times. The threshold value is automatically set to six - the default value of MATLAB - to trigger the ANN to stop training. The validation dataset is used to monitor that error occurs for six times consecutively. ANN training performs best at the epoch when there is minimum error in the validation dataset. Therefore, in the

ANN training in Company 1, the result shows that the best performance is at epoch 4 with a minimum MSE value of 0.10456.

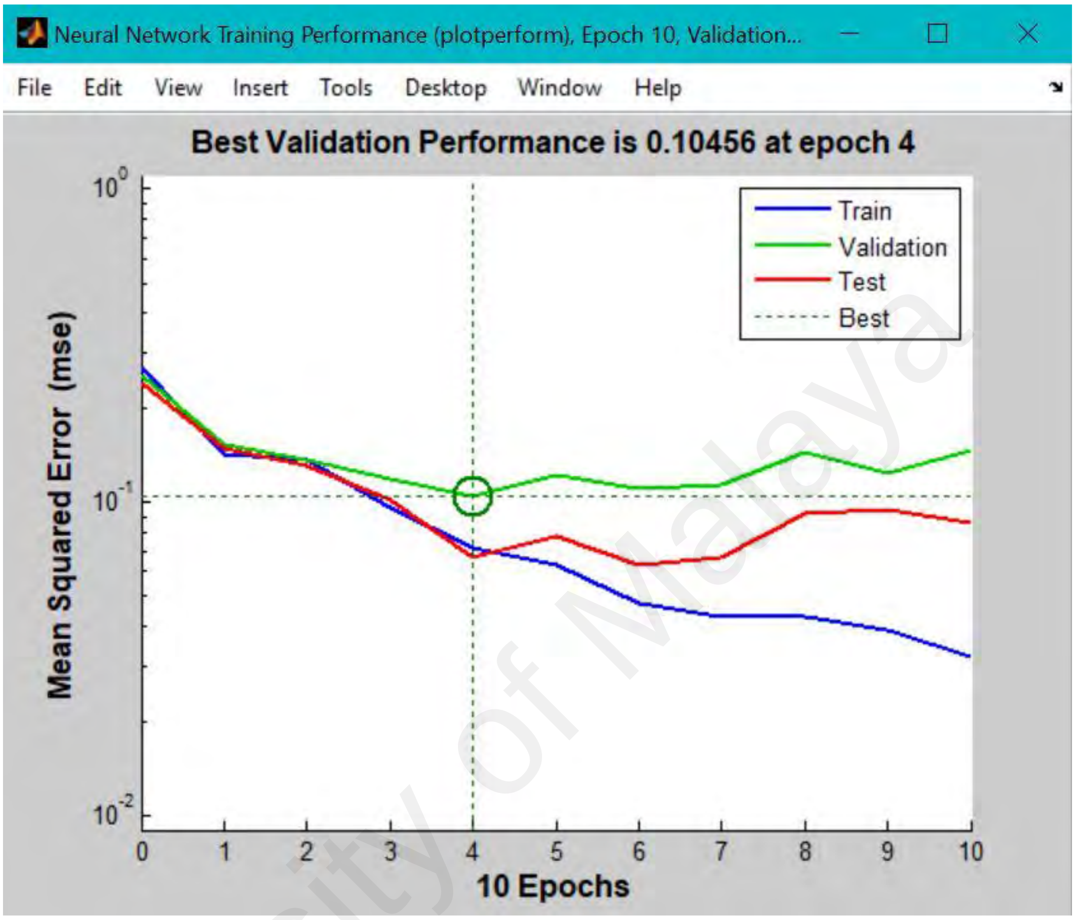


Figure 4.14: ANN Training Performance Plot (Company 1)

In Company 2, after training the ANN repeatedly, the MSE has decreased to the minimum. In ANN training, the best performance is achieved at epoch 10 with minimum MSE value of 0.086242, as shown in Figure 4.15.

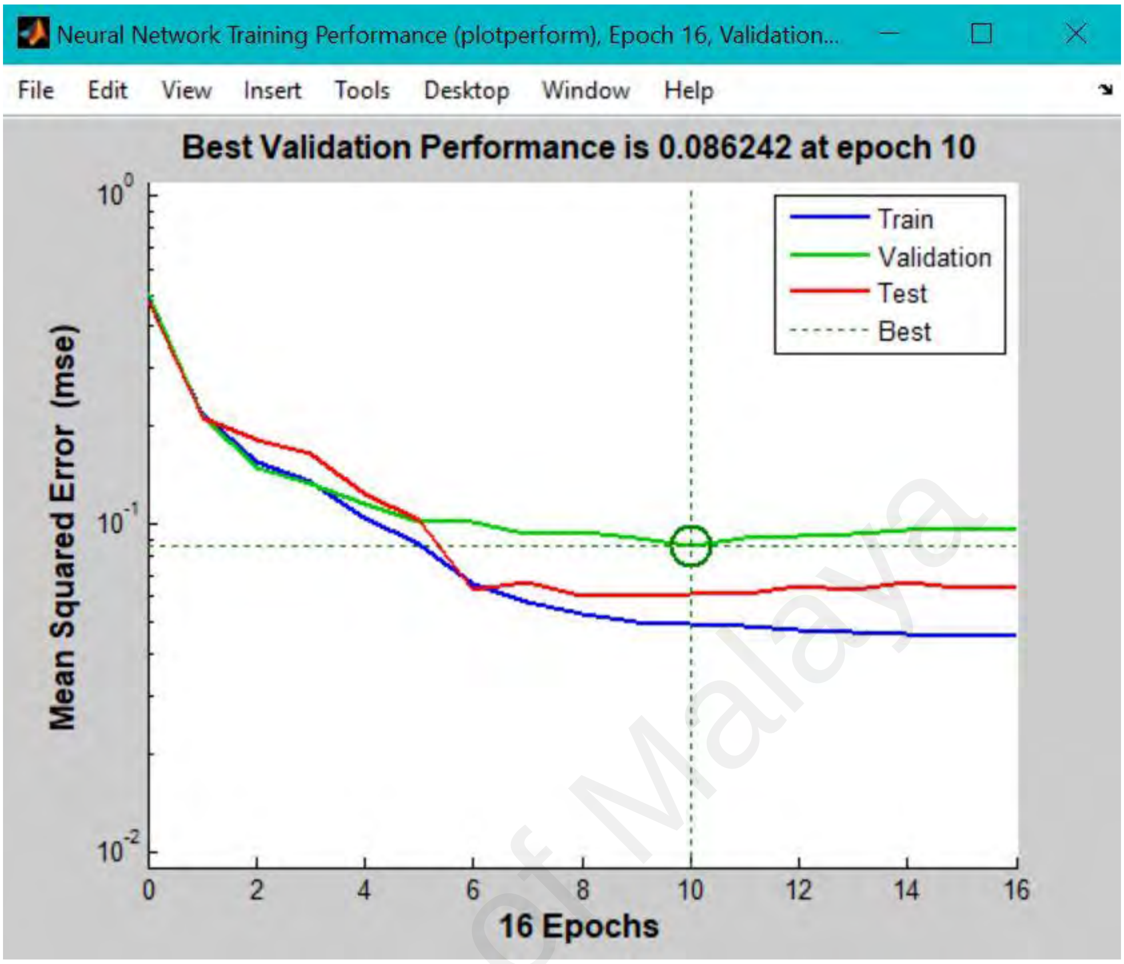


Figure 4.15: ANN Training Performance Plot (Company 2)

Figure 4.16 shows the training state of Company 1. The first line graph shows the gradient value is 0.049363 at epoch 10; the second line graph shows the value of mu is equal to 0.00001 at epoch 10; and the third line graph shows the validation check value is 6 at epoch 10. This line graph shows that after the fourth iteration (epoch), the validation check value has increased and reached the value 6 at epoch 10. These three values (gradient, mu, and validation check) were set as an ANN training stop condition, which is provided by default from the network training function of Levenberg-Marquardt algorithm in MATLAB. If any of these three values meet certain conditions, the ANN training process will be terminated. In this case, the validation check (validation errors have occurred consecutively for six times) has met the condition to the stop ANN training. If the gradient value is the value of the backpropagation gradient of each iteration (epoch) that assists in adjusting the ANN weight to find the lowest value of the cost function (the function that shows the errors that had occurred from the predicted ANN output value and the actual output value), then the final value of gradient is 0.049363 at epoch 10. Also, the mu value is used as a learning rate to offer a choice of weight adjusting process.

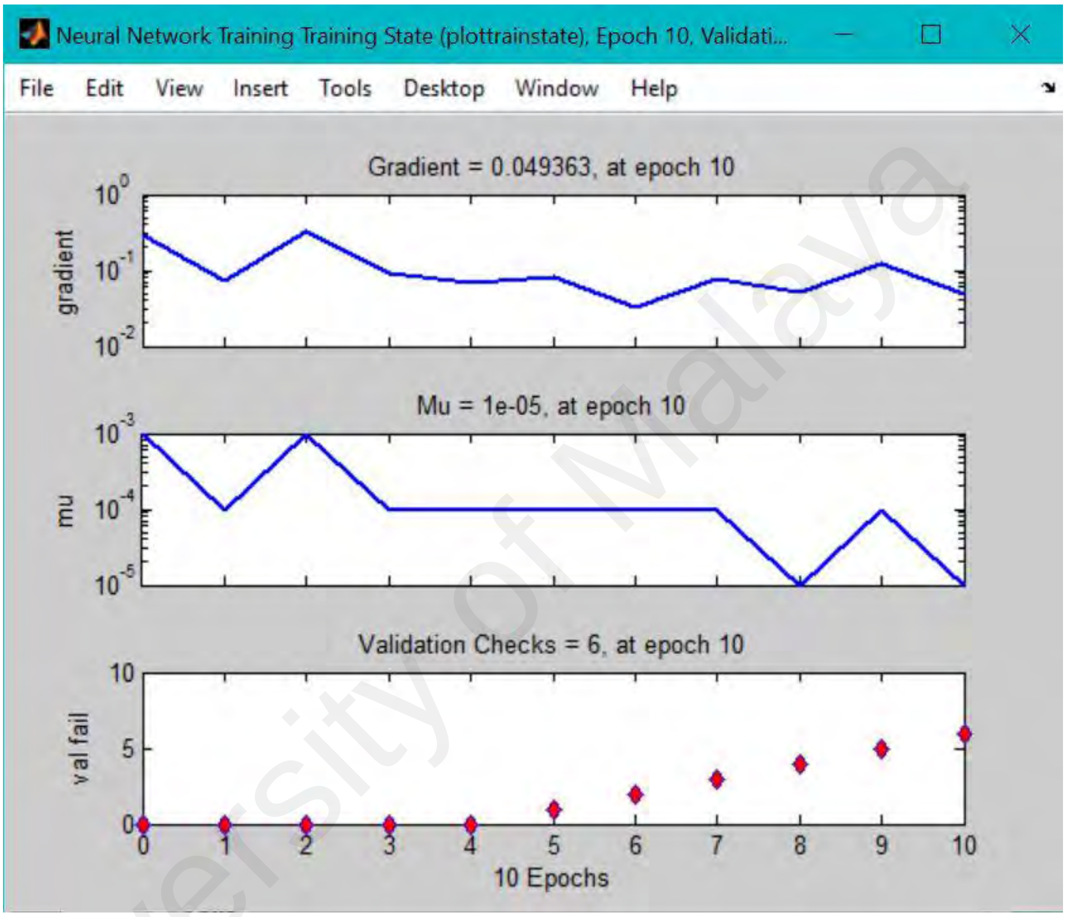


Figure 4.16: ANN Training State (Company 1)

For Company 2, the gradient value is 0.0093314 at epoch 16, the value of mu is 0.00001 at epoch 16 and the validation check value is 6 and the ANN stops training at epoch 16, as shown in Figure 4.17. At the eighth iterations (epochs), the error of the validation dataset started to increase but did not occur up to six times (default threshold value of MATLAB). In this case, the number of errors of the validation dataset increased after the tenth iterations (epochs), and the error of the validation dataset increased and reached value of 6 at epoch 16. The repetition is six times the error of the validation dataset, implying that the ANN training has resulted in over-fitting the training dataset. Hence, at epoch 16, the ANN has decided to stop training the network.

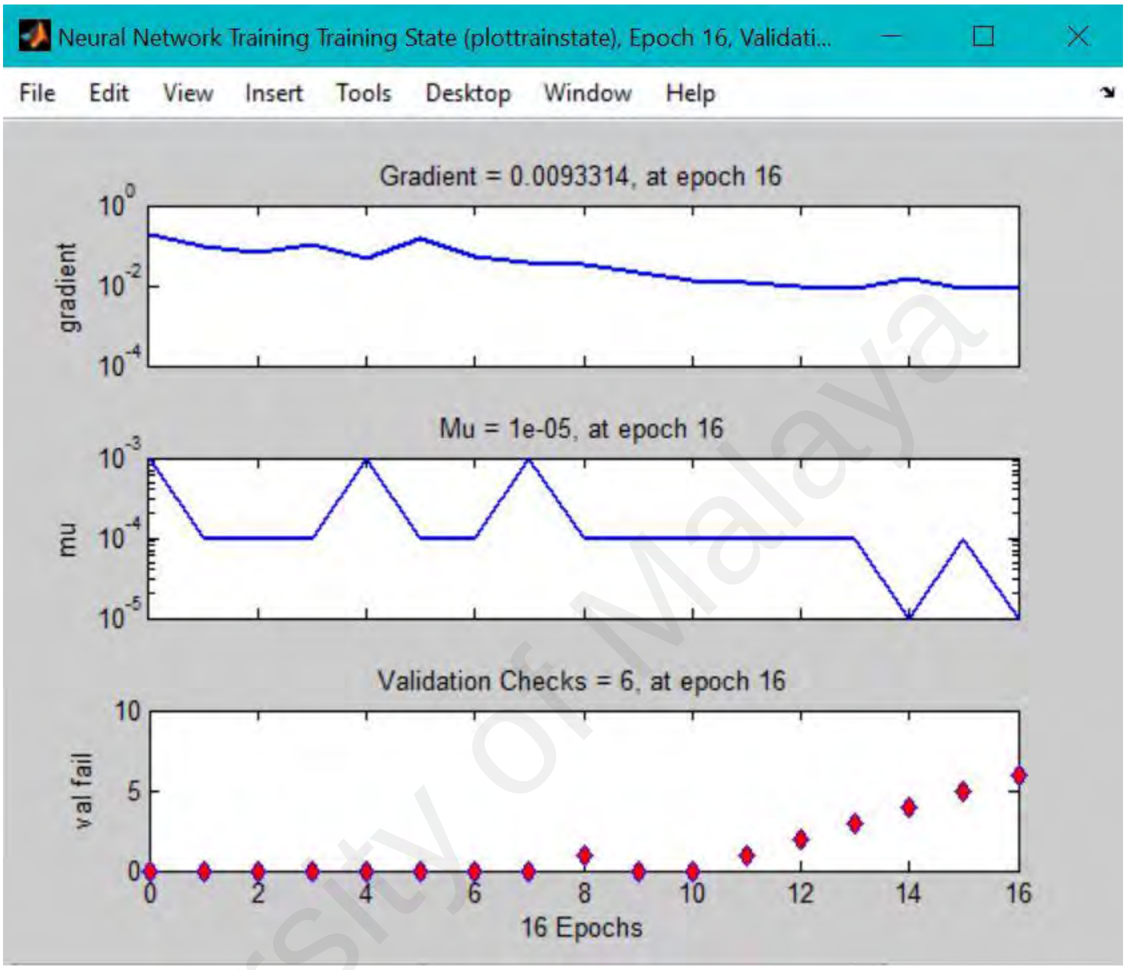


Figure 4.17: ANN Training State (Company 2)

Figure 4.18 shows the Error Histogram for Company 1. It helps in visualising the difference between the target values and the actual output values after training our ANN. The training data, validation data, and test data are represented by the blue bars, green bars, and red bars, respectively. The histogram shows that the majority of errors fall between -0.2 and +0.2, with most of the error values distributed approximately around 0 (zero). This implies that the trained ANN has good data-fitting error within a reasonable range.

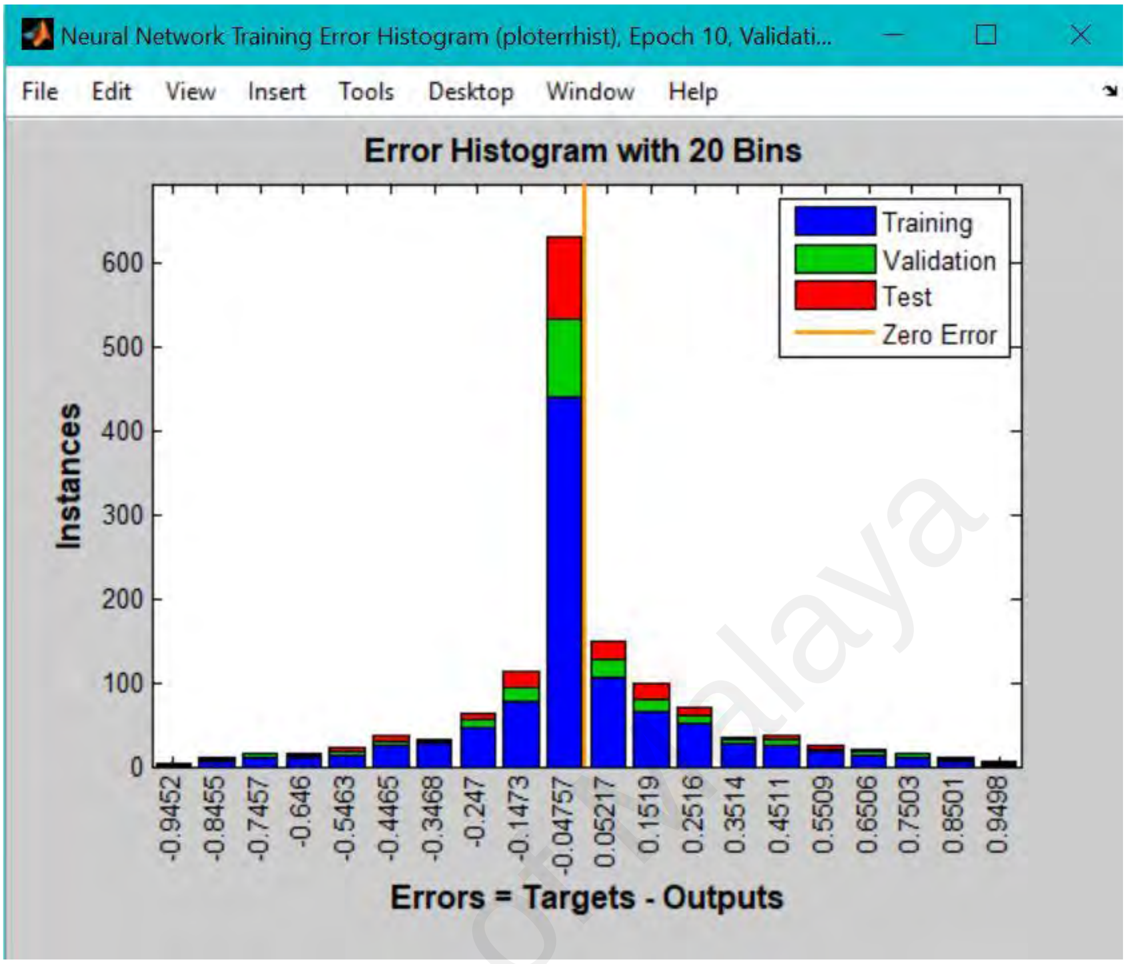


Figure 4.18: ANN Training Error Histogram (Company 1)

Figure 4.19 shows the Error Histogram for Company 2. The majority of the errors fall between -0.3 and +0.3, with the most error values distributed approximately around 0 (zero). This indicates that the trained ANN has good data-fitting error within a reasonable range.

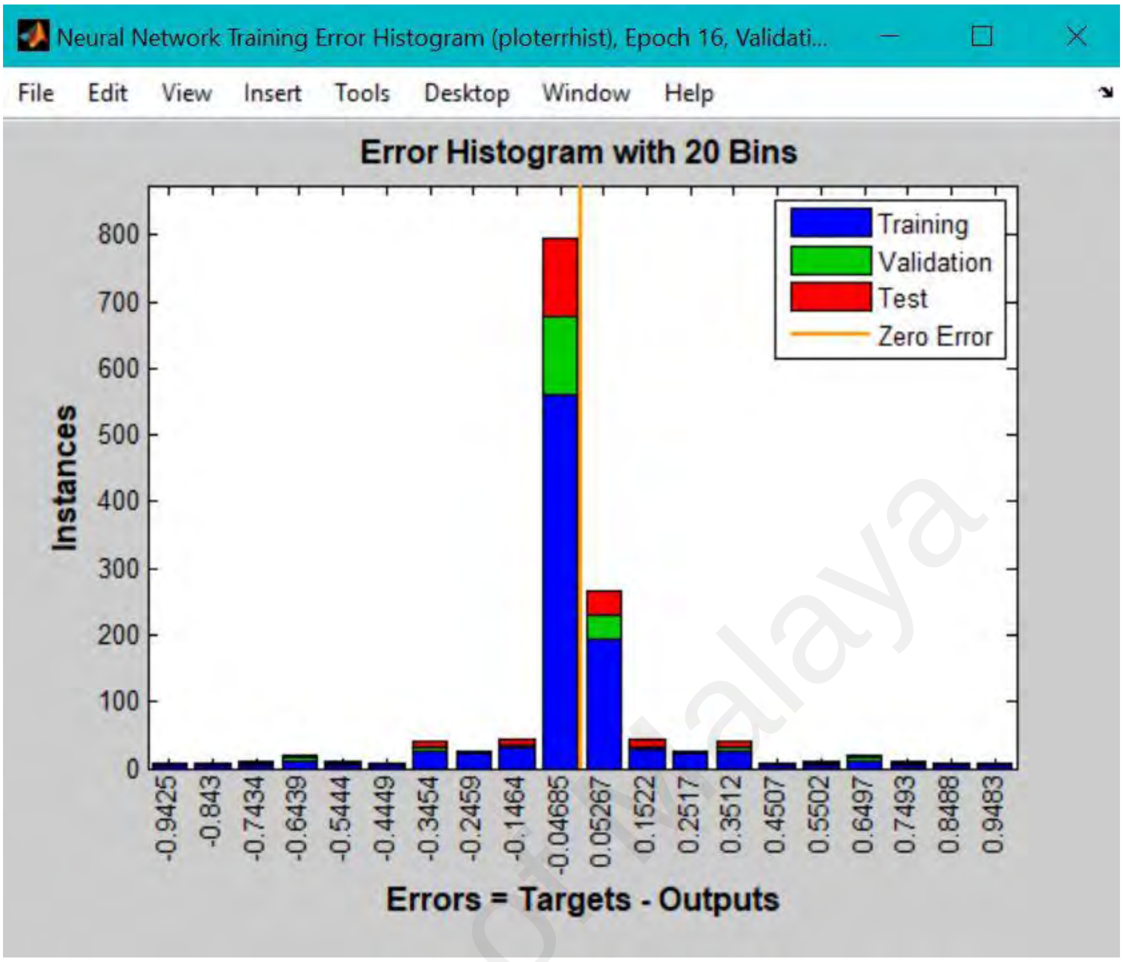


Figure 4.19: ANN Training Error Histogram (Company 2)

Figure 4.20 shows the confusion matrix for Company 1. It consists of four confusion matrices - training, validation, testing and all confusion matrices of the ANN. The performance of the constructed ANN classifier can be evaluated using the confusion matrices.

The relationship between the predicted class (output class) and the actual class (target class) of each set of data is represented in the confusion matrices, where the rows represent the predicted class (output class) and the columns represent the actual class (target class) of the network. The labelled numbers 1, 2, and 3 represent the class “Good”, “Average” and “Poor”, respectively, pertaining to the programmers’ performance. The information inside each cell of the confusion matrix can show the performance of the constructed ANN classifier - the diagonal cells (highlighted green blocks) - showing the number of correctly classified class members, and the cell elements that are not on the diagonal cells (highlighted red blocks) showing the number of incorrectly classified class members; the top cells (bold text) show the instances and percentages underneath it; the last column (highlighted grey blocks) represents the precision values of ANN classifier predicting each class, indicating the number of times that the classifier prediction result is correct; the last row (highlighted grey blocks) represents the recall rates of the ANN classifier at each class used for measuring the rate of correctly predicted class of the actual class for a specific class; and the last bottom right blocks in each matrix (highlighted blue blocks) represent the accuracy rate which indicates the overall rate of the ANN classifier correctly predicted. In this case, the performance of the constructed ANN classifier of Company 1 produced a result which shows that the training, validation and test datasets have achieved performance level of 86.3%, 77.5% and 87.3% respectively. Overall, the average performance level of the classifier in the three datasets (training, validation, and test set) is 85.1 %.

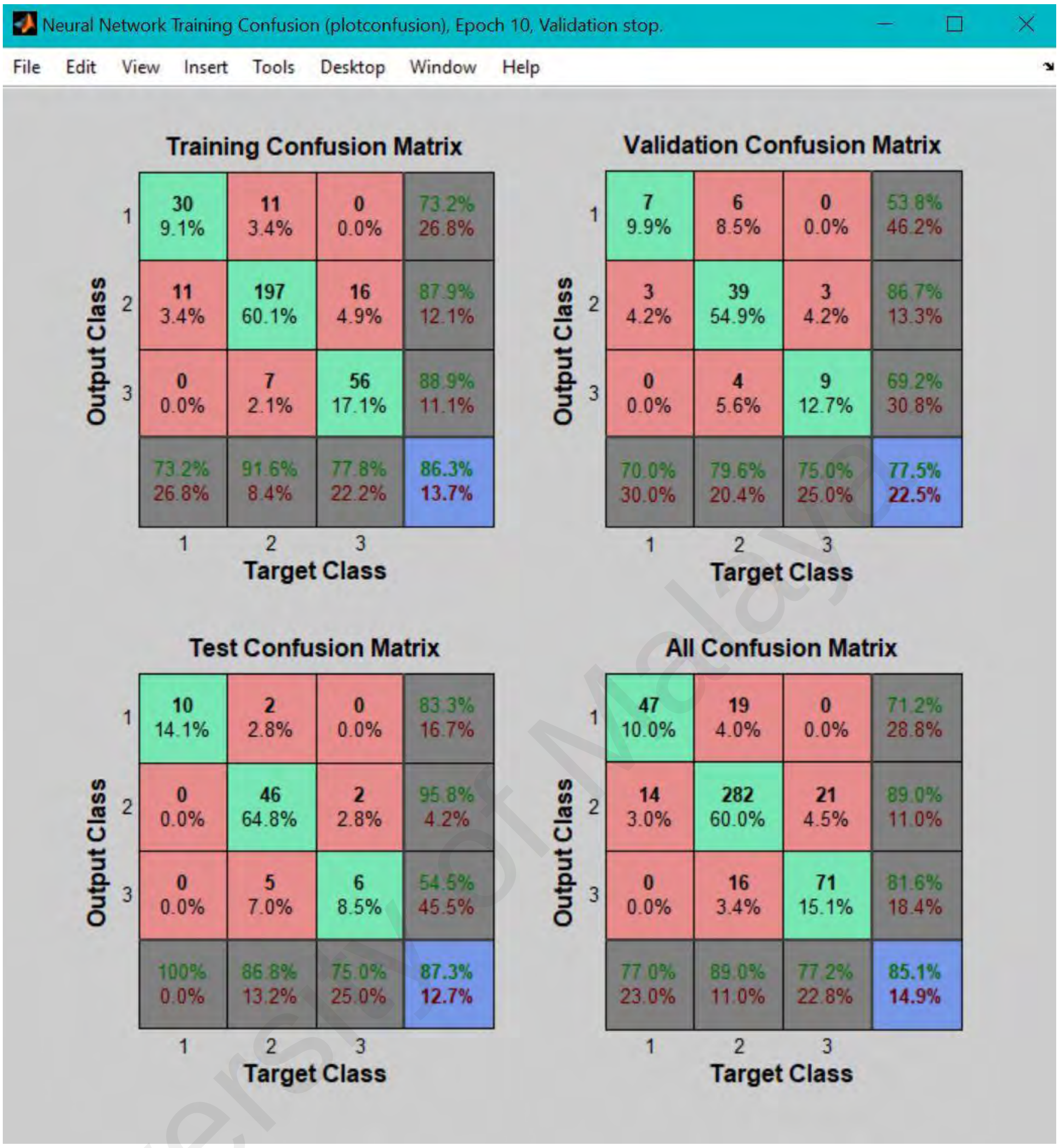


Figure 4.20: ANN Training Confusion Matrices (Company 1)

For Company 2, the confusion matrix shown in Figure 4.21, uses the same colour scheme as in the confusion matrix of Company 1. The latter confusion matrix consists of four matrices - training, validation, testing, and all confusion matrices of the ANN but with different values in each cell. However, in Company 2, the constructed ANN classifier produced a result in which the training, validation, and test datasets achieved performance level of 90.3%, 78.9% and 87.3% respectively. Overall, the average performance level of the classifier in the three datasets (training, validation, and test sets) is 88.1 %.

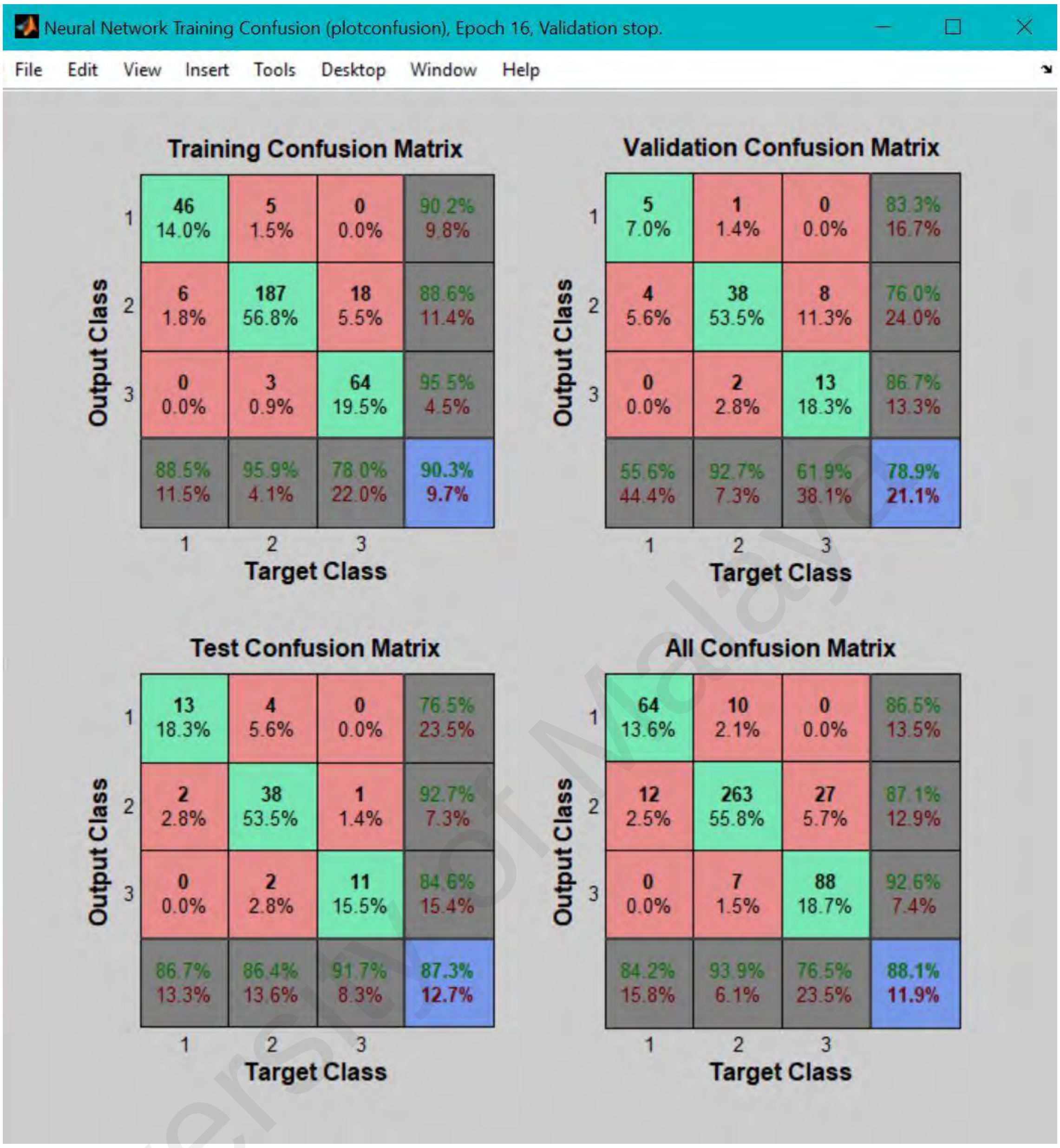


Figure 4.21: ANN Training Confusion Matrices (Company 2)

4.4 Summary

This chapter describes the proposed technique for predicting and determining the best-fit programmers for software companies. The chapter first begins with an overview of the proposed technique, and then describes each step of the proposed technique in detail. The proposed technique is implemented using MATLAB (R2014a). It also discusses the user interface design, the datasets from Company 1 and Company 2 used in the ANN training. The mean squared error (MSE) and the confusion matrices of both companies are also explained in detail. The next chapter discusses the evaluation result of the proposed technique.

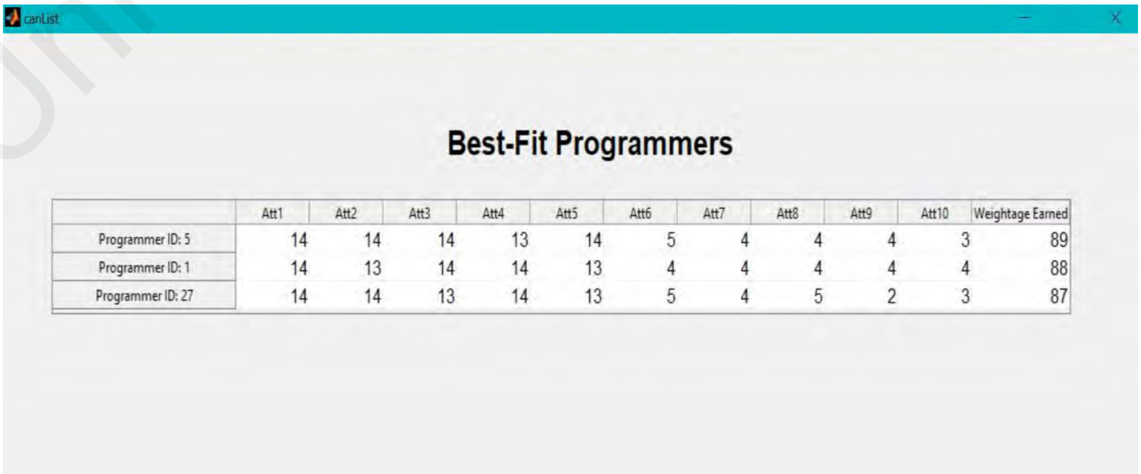
CHAPTER 5: EVALUATION OF THE PROPOSED TECHNIQUE

5.1 Overview

This chapter presents the predicted result of the best-fit programmers using the proposed technique and describes the evaluation process on the accuracy of the predicted outcomes as well as the performance of the proposed technique. The predicted outcome is evaluated by comparing the predicted result with the result from the original dataset. The performance of the proposed technique is evaluated using the standard measurement metrics - confusion matrix. The performance evaluation focuses on the accuracy, precision and recall of the predictive result from the proposed technique classifier. The evaluation of the predicted outcome and the performance of the technique is discussed in the sections, below.

5.2 Results of the Predicted Best-Fit Programmer

Figure 5.1 shows the predicted results of the best-fit programmers for Company 1. The best-fit programmers were predicted based on the prognostic attributes, i.e., the programmers who showed good performance and were rated 'good' for each of the prognostic attributes. The following subsections explain the results of the predicted best-fit programmers for Company 1 and Company 2, respectively.



	Att1	Att2	Att3	Att4	Att5	Att6	Att7	Att8	Att9	Att10	Weightage Earned
Programmer ID: 5	14	14	14	13	14	5	4	4	4	3	89
Programmer ID: 1	14	13	14	14	13	4	4	4	4	4	88
Programmer ID: 27	14	14	13	14	13	5	4	5	2	3	87

Figure 5.1: Predicted Best-Fit Programmers (Company 1)

5.2.1 Predicted Best-Fit Programmers of Company 1

In Company 1, the prognostic attributes are numbered 1, 2, 3, 4, 5, 6 and 8. Figure 5.1 shows that only three programmers (Programmer IDs 1, 5 and 27) are the best-fit programmers of Company 1 from a total of 61 programmers who showed good performance. As mentioned in Chapter 3, the value of each of these attributes is first transformed into a set of equal intervals that represent a class label. All the attribute values from the original dataset of Company 1 were substituted by the class label that each value represents, as shown in Figure 5.2. The values in the square brackets below each bar are the values within each equal-width binning of an attribute. For example, a programmer who scores 14 for Attribute 1 - Write Functionally Correct Code - is labeled as “Good”. Similarly, the values for all the other attributes were classified and labeled in the same manner.

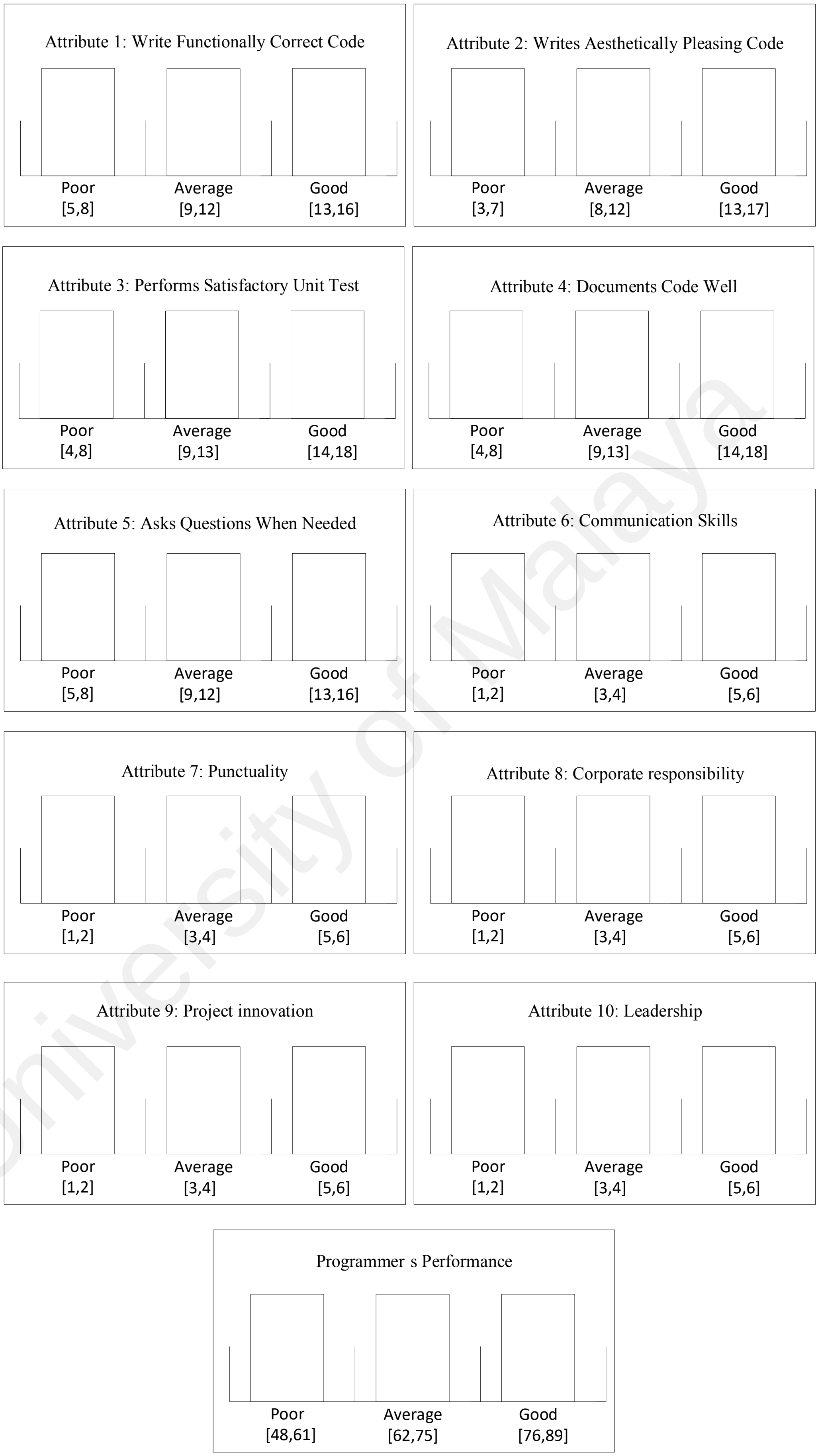
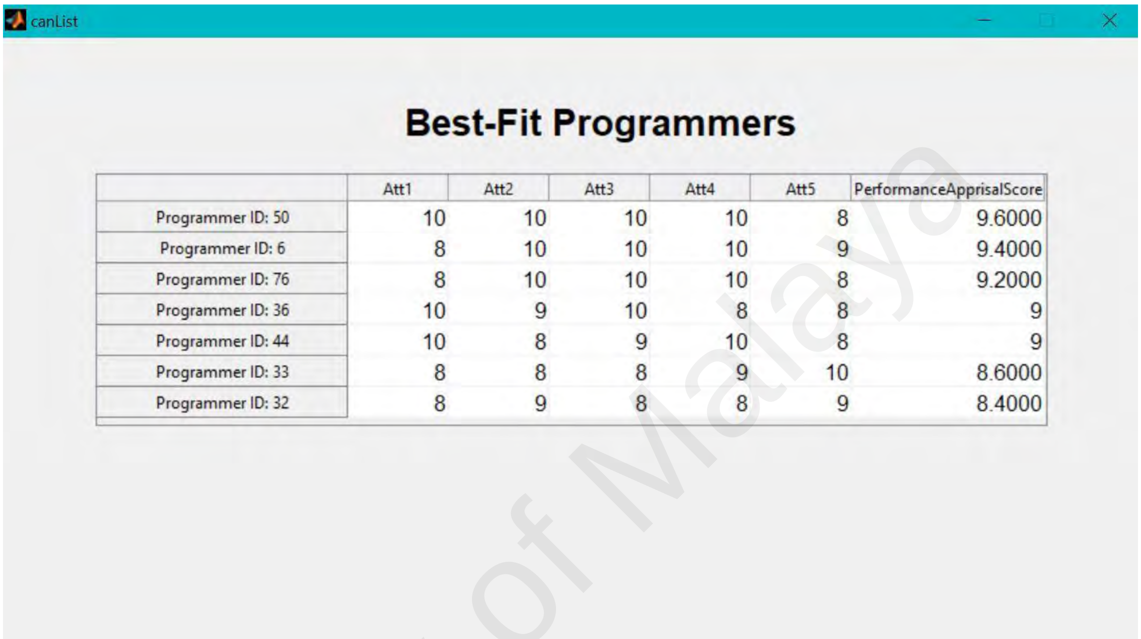


Figure 5.2: Equal - Width Binning of Attributes (Company 1)

5.2.2 Predicted Best-Fit Programmers of Company 2

In Company 2, all the prognostic attributes are numbered 1, 2, 3, 4 and 5, as shown in Figure 5.3. This is because the analysed results of the attributes show that they all have high probability of influencing the programmers' work performance.



	Att1	Att2	Att3	Att4	Att5	PerformanceApprisalScore
Programmer ID: 50	10	10	10	10	8	9.6000
Programmer ID: 6	8	10	10	10	9	9.4000
Programmer ID: 76	8	10	10	10	8	9.2000
Programmer ID: 36	10	9	10	8	8	9
Programmer ID: 44	10	8	9	10	8	9
Programmer ID: 33	8	8	8	9	10	8.6000
Programmer ID: 32	8	9	8	8	9	8.4000

Figure 5.3: Predicted Best-Fit Programmers (Company 2)

Figure 5.4 shows the attribute values of the dataset of Company 2 which had been converted into their respective class label based on the equal-width binning of the attributes. There are some differences in the range of values shown in the square brackets. For example, programmers whose score is equal to or more than 8 in Attribute - Technical Skill - will be classified and labeled as "Good". The other attributes were also determined in the same way. Figure 5.3 shows the seven programmers (Programmer IDs 50, 6, 76, 36, 44, 33 and 32) who have been determined to be best-fit programmers of Company 2 from a total of 76 programmers who showed good performance.

Figure 5.4: Equal - Width Binning of Attributes (Company 2)

5.3 Preliminary Analyses

As the main aim of this research study is to identify the programmer's attributes that affect the recruitment and selection decision to determine the best-fit programmers in a software company, it is necessary to conduct a statistical analysis to show the association between these attributes. These statistical analysis methods include chi-square, correlation analysis, regression analysis, to name a few (Cooper, Hedges, & Valentine, 2009; Hair, Babin, Anderson, & Black, 2018). Among these statistical methods, the correlation analysis test is the most suitable method to test the degree of connection strength between the pair of variable when the variable data is quantitative data (numerical data). If the variable data is categorical data (nominal data), the chi-square test is a more suitable method. However, if the assumption on causality in the relationship is required, the regression is a more appropriate method. In this study, the programmer's attributes data from the two software companies are quantitative data (numerical data). Furthermore, this research study only requires obtaining information on the relationship between the pair of programmer's attributes, and the variables being studied are not designated as dependent or independent. Thus, it is appropriate to use the correlation analysis method in this research study. The correlation analysis test requires that the data being tested are normally distributed. Hence, the normality test was performed prior to conducting the correlation analysis test to determine the appropriate type of correlation analysis test (Sekaran & Bougie, 2016).

5.3.1 Normality Test

Normal distribution is described by a symmetrical bell-shaped that has the greatest frequency of scores in the centre, with smaller frequencies decrease towards on either side (Sekaran & Bougie, 2016). There are several ways to evaluate data distribution from mathematically or graphically. Mathematically, these include Shapiro-Wilk statistic, Kolmogorov-Smirnov statistic, skewness and kurtosis. For the graphical approaches, these include histograms, stem-and leaf plots, boxplots, normal probability plots, and detrended normal plots. In this research study, skewness values, kurtosis values and histograms are used to assess the normality distribution of the two software companies' datasets. Sekaran and Bougie (2016) asserted that the values of skewness test and kurtosis test that fall within the range of -2 to +2 are deemed acceptable within the normal range. However, Coakes (2013) indicated that the kurtosis valued that fall between -3 to +3 is considered satisfactory for social science.

5.3.1.1 Normality Test of Company 1

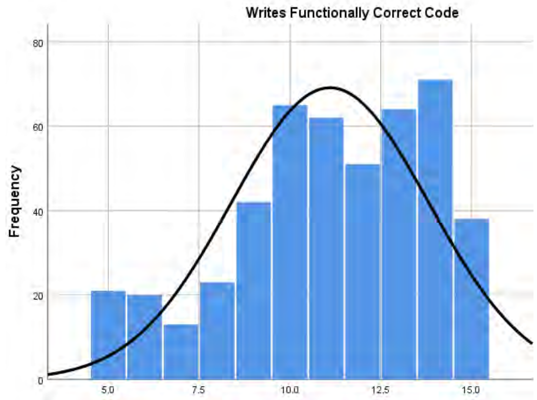
Table 5.1 shows the results of skewness and kurtosis values of the programmers' prognostic attributes of Company 1, with the total sample size of 470 programmers' past annual performance appraisal data records. All the seven programmers' prognostic attributes of Company 1 have the skewness and kurtosis values within the range of -2 to +2. This indicates that data for all the seven programmers' prognostic attributes of Company 1 are normally distributed and hence, parametric test can be used to analyse the data (Sekaran & Bougie, 2016).

Table 5.1: Normality Test Results of Company 1 Dataset (Sample Size, N=470)

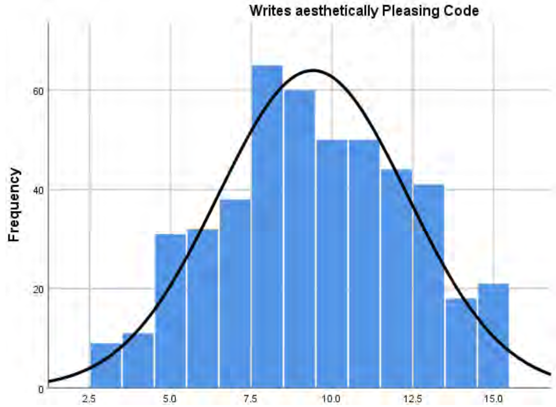
Programmer's Attributes	Mean	Std. Deviation	Skewness	Kurtosis
ATT_1	11.10	2.713	-0.540	-0.468
ATT_2	9.41	2.931	-0.039	-0.684
ATT_3	10.41	2.373	-0.486	0.054
ATT_4	10.80	2.759	-0.351	-0.577
ATT_5	10.14	2.484	-0.105	-0.618
ATT_6	3.29	1.190	-0.242	-0.795
ATT_7	3.23	1.130	-0.056	-0.854

Note: ATT_1: Writes Functionally Correct Code
ATT_2: Writes Aesthetically Pleasing Code
ATT_3: Performs Satisfactory Unit Testing
ATT_4: Documents Code Well
ATT_5: Asks Questions When Needed
ATT_6: Communication Skills
ATT_7: Corporate Responsibility

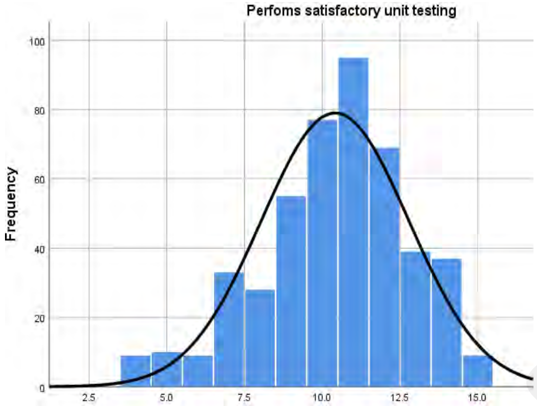
The test of normality using histograms is also used to visualise the distribution (normality) of the programmers' past annual performance appraisal data records of Company 1, graphically. Figure 5.5 shows the histograms of all the seven programmers' prognostic attributes of Company 1. Almost all the prognostic attributes are having a symmetric bell-shaped, except for Prognostic Attribute 1 (Figure 5.5 (a)) and Prognostic Attribute 4 (Figure 5.5 (d)) which are slightly skewed to the left. However, the skewness does not show a large difference in the normality of the data. Thus, this can be deemed acceptable for normal distribution (Ali & Akayuure, 2016).



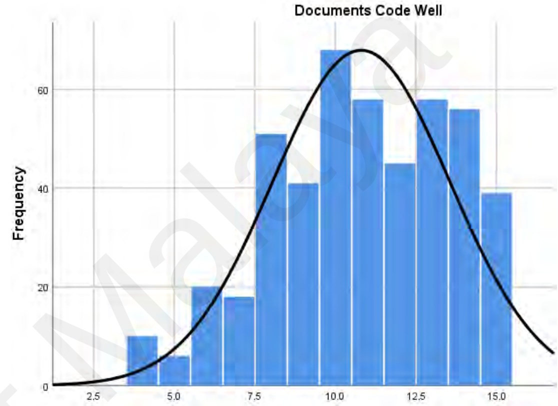
(a) Prognostic Attributes 1 (Company 1)



(b) Prognostic Attributes 2 (Company1)

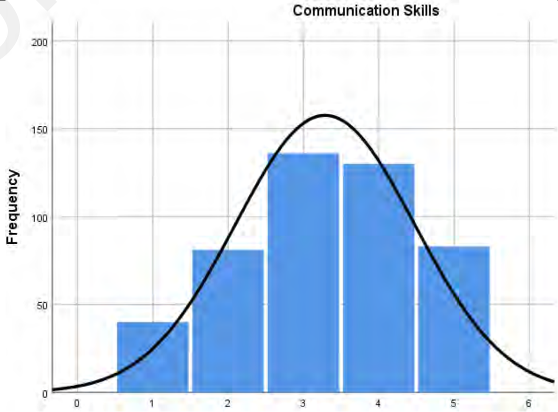


(c) Prognostic Attributes 3 (Company 1)

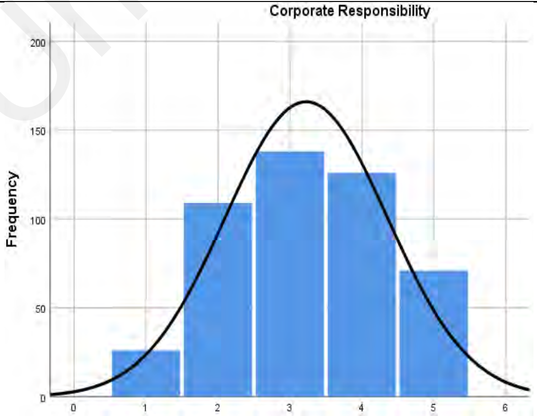


(d) Prognostic Attributes 4 (Company1)

(e) Prognostic Attributes 5 (Company1)



(f) Prognostic Attributes 6 (Company1)



(g) Prognostic Attributes 7 (Company1)

Figure 5.5: Results of the Normality Test on the Past Annual Performance Appraisal Programmers Dataset (Company 1)

5.3.1.2 Normality Test of Company 2

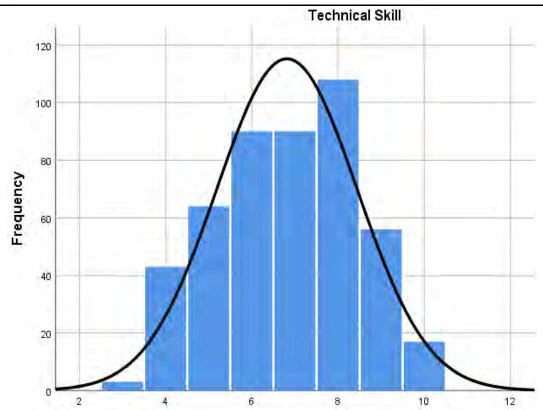
Table 5.2 shows the results of skewness and kurtosis values of the programmers' prognostic attributes of Company 2, with the total sample size of 471 programmers' past annual performance appraisal data records. All the five programmers' prognostic attributes from Company 2 have the skewness and kurtosis values within the range of -2 to +2. This indicates that data for all the five programmers' prognostic attributes of Company 2 are normally distributed and hence, parametric test can be used to analyse the data (Sekaran & Bougie, 2016).

Table 5.2: Normality Test Results of Company 2 Dataset (Sample Size, N=471)

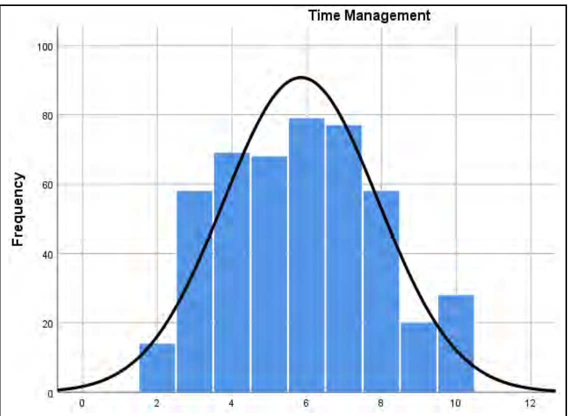
Programmer's Attributes	Mean	Std. Deviation	Skewness	Kurtosis
ATT_1	6.81	1.631	-0.117	-0.792
ATT_2	5.85	2.071	0.169	-0.748
ATT_3	6.34	1.874	-0.170	-0.207
ATT_4	6.74	1.722	-0.054	-0.699
ATT_5	6.60	1.737	-0.092	-0.725

Note: ATT_1: Technical Skill
ATT_2: Time Management
ATT_3: Documentation/Presentation
ATT_4: Teamwork/ Cooperative
ATT_5: Attitude and Self Growth

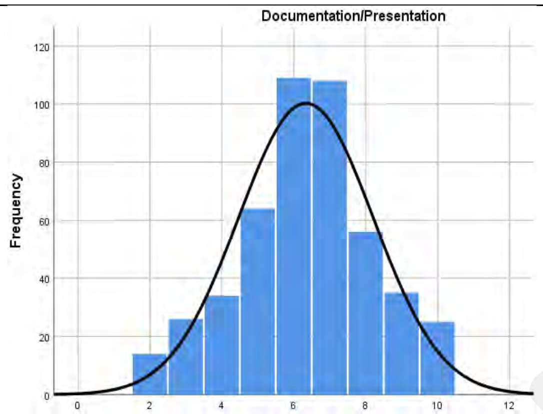
Besides skewness and kurtosis values, the histograms shown in Figure 5.6 were used to confirm the normality of the data for all the five programmers' prognostic attributes of Company 2. All the histograms of all five programmers' prognostic attributes are having a symmetric bell-shaped. Thus, this also confirmed that the distribution of all the datasets of Company 2 are in normal distribution.



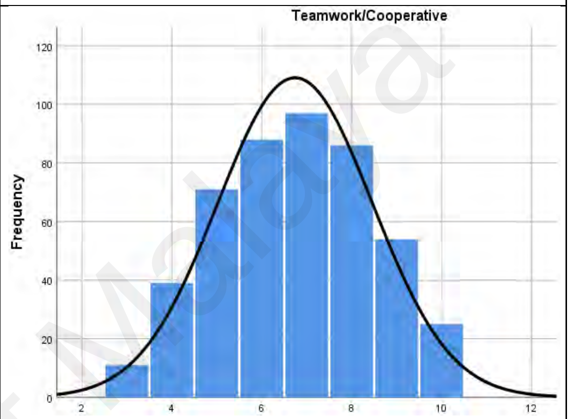
(a) Prognostic Attributes 1 (Company 2)



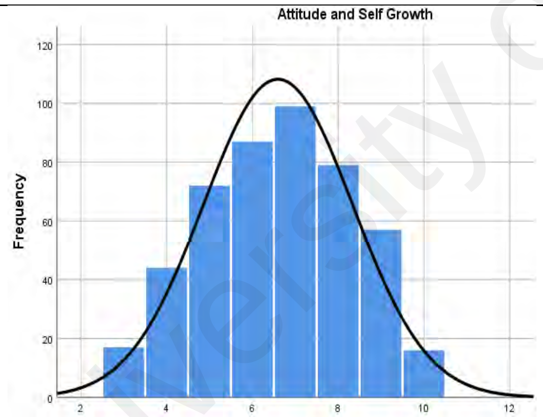
(b) Prognostic Attributes 2 (Company 2)



(c) Prognostic Attributes 3 (Company 2)



(d) Prognostic Attributes 4 (Company 2)



(e) Prognostic Attributes 5 (Company 2)

Figure 5.6: Results of the Normality Test on the Past Annual Performance Appraisal Programmers Dataset (Company 2)

5.3.2 Correlation Analysis

As the dataset of programmers' prognostic attributes from Company 1 and Company 2 are normally distributed, the Pearson's correlation (a parametric test) can be used to investigate the relationship between the pair of programmers' prognostic attributes. Pearson's correlation coefficient is often denoted by the letter r, with range from -1 to +1 to indicate the positive or negative relationship. The closer value to -1 or +1 indicates the stronger the relationship between the pair of variable. The value 0 indicates that there is no relationship between the pair of variables. Table 5.3 shows the Rule of Thumb to interpret the strength of the correlation between programmers' prognostic attributes (Hinkle, Wiersma, & Jurs, 2002, p. 109).

Table 5.3: Rule of Thumb for Interpretation of the Correlation Value Ranges (Hinkle, Wiersma, & Jurs, 2002, p. 109)

Correlation (r) Values	Interpretation
0.91 to 1.00 (-0.91 to –1.00)	Very high positive (negative) correlation
0.71 to 0.90 (-0.71 to -0.90)	High positive (negative) correlation
0.51 to 0.70 (-0.51 to -0.70)	Moderate positive (negative) correlation
0.31 to 0.50 (-0.31 to -0.50)	Low positive (negative) correlation
0.01 to 0.30 (-0.01 to -0.30)	Negligible correlation
0 (Zero)	No correlation

5.3.3 Correlation Analysis of the Prognostic Attributes of Company 1

As shown in Table 5.4, the positive Pearson's correlation values (r) of the prognostic attributes of Company 1 are ranging from 0.012 to 0.222. The smallest positive Pearson's correlation, r = 0.012 is found between "Documents Code Well" and "Asks Questions When Needed". The largest positive Pearson's correlation, r = 0.222 is found between "Asks Questions When Needed" and "Communication Skills". The negative Pearson's correlation values (r) of the dataset are ranging from -0.174 to -0.035. The smallest negative Pearson's correlation, r = -0.035 is found between "Writes Functionally Correct Code" and "Asks Questions When Needed". While the largest negative Pearson's correlation, r = -0.174 is found between "Writes Functionally Correct Code" and "Communication Skills". As all these values are less than 0.5 and close to 0, it can be concluded that all the prognostic attributes of Company 1 are not positively or negatively correlated.

Table 5.4: The Correlation Between the Pair of Programmers' Prognostic Attributes of Company 1 (N=470)

		Writes Functionally Correct Code	Writes Aesthetically Pleasing Code	Performs Satisfactory Unit Testing	Documents Code Well	Asks Questions When Needed	Communication Skills	Corporate Responsibility
Writes Functionally Correct Code	Pearson Correlation	1	.117*	.077	.104*	-.035	-.174**	-.147**
	Sig. (2-tailed)		.011	.097	.024	.445	.000	.001
Writes Aesthetically Pleasing Code	Pearson Correlation	.117*	1	-.047	.089	-.053	.040	.039
	Sig. (2-tailed)	.011		.308	.053	.248	.384	.401
Performs Satisfactory Unit Testing	Pearson Correlation	.077	-.047	1	-.066	.168**	-.044	-.153**
	Sig. (2-tailed)	.097	.308		.151	.000	.340	.001
Documents Code Well	Pearson Correlation	.104*	.089	-.066	1	.012	.106*	.019
	Sig. (2-tailed)	.024	.053	.151		.787	.021	.680
Asks Questions When Needed	Pearson Correlation	-.035	-.053	.168**	.012	1	.222**	.044
	Sig. (2-tailed)	.445	.248	.000	.787		.000	.345
Communication Skills	Pearson Correlation	-.174**	.040	-.044	.106*	.222**	1	.219**
	Sig. (2-tailed)	.000	.384	.340	.021	.000		.000
Corporate Responsibility	Pearson Correlation	-.147**	.039	-.153**	.019	.044	.219**	1
	Sig. (2-tailed)	.001	.401	.001	.680	.345	.000	

*. Correlation is significant at the 0.05 level (2-tailed).
**. Correlation is significant at the 0.01 level (2-tailed).

5.3.4 Correlation Analysis of the Prognostic Attributes of Company 2

As shown in Table 5.5, the positive Pearson's correlation values (r) of the prognostic attributes of Company 2 are ranging from 0.005 to 0.322. The smallest positive Pearson's correlation, r = 0.005 is found between "Documentation/Presentation" and "Technical Skill". The largest positive Pearson's correlation, r = 0.322 is found between "Documentation/Presentation" and "Attitude and Self Growth". The negative Pearson's correlation value (r) of the dataset is -0.008. The negative Pearson's correlation, r = -0.008 is found between "Teamwork/ Cooperative", the "Technical Skill". As all these values are less than 0.5 and close to 0, it can be concluded that all the prognostic attributes of Company 2 are not positively or negatively correlated.

Table 5.5: The Correlation Between the Pair of Programmers' Prognostic Attributes of Company 2 (N=471)

		Technical Skill	Time Management	Documentation /Presentation	Teamwork/ Cooperative	Attitude and Self Growth
Technical Skill	Pearson Correlation	1	.040	.005	-.008	.035
	Sig. (2-tailed)		.385	.908	.860	.449
Time Management	Pearson Correlation	.040	1	.320**	.309**	.282**
	Sig. (2-tailed)	.385		.000	.000	.000
Documentation/Presentation	Pearson Correlation	.005	.320**	1	.222**	.322**
	Sig. (2-tailed)	.908	.000		.000	.000
Teamwork/ Cooperative	Pearson Correlation	-.008	.309**	.222**	1	.169**
	Sig. (2-tailed)	.860	.000	.000		.000
Attitude and Self Growth	Pearson Correlation	.035	.282**	.322**	.169**	1
	Sig. (2-tailed)	.449	.000	.000	.000	

5.4 Evaluation of Predicted Outcome

To evaluate the accuracy of the predicted outcome using the proposed technique, the list of predicted best-fit programmers was compared with the list of programmers who showed good performance in their past annual performance appraisal. The evaluations for Company 1 and Company 2 are presented in the following sections.

5.4.1 Predicted Outcome Evaluation of Company 1 and Company 2

To evaluate the predicted outcomes, the list of programmers from the original datasets of Company 1 who showed good performance were recorded in an EXCEL sheet. Those records were compared with the results of the past annual performance appraisal of the programmers who showed 'good' performance. The past annual performance appraisal of Company 1 contained the performance records of 470 programmers. However, for evaluation purposes, only the records of good performing programmers were considered. The list of good programmers who showed 'good' performance was then sorted in descending order of their overall scores in their respective annual performance appraisal, as shown in Table 5.6, below. A total of 61 programmers showed good performance. Their programmer IDs are listed in the first column. They were assessed based on 10 attributes and the last column shows the overall annual performance appraisal score of each programmer. These results were then compared with the predicted results produced by the proposed technique, as shown in Figure 5.1 (section 5.2). The predicted best-fit programmers are Programmer IDs 5, 1 and 27. For these three programmers, their results from the past annual performance appraisal match their predicted results using the proposed technique.

Table 5.6: Results of the Past Annual Performance Appraisal of Programmers (Company 1)

Programmer ID	Att1	Att2	Att3	Att4	Att5	Att6	Att7	Att8	Att9	Att10	Weight Earned
Programmer ID: 5	14	14	14	13	14	5	4	4	4	3	89
Programmer ID: 1	14	13	14	14	13	4	4	4	4	4	88
Programmer ID: 27	14	14	13	14	13	5	4	5	2	3	87
Programmer ID: 34	14	12	13	15	11	4	5	3	5	4	86
Programmer ID: 54	14	15	13	10	11	5	3	5	4	5	85
Programmer ID: 60	14	11	14	12	14	5	3	3	4	5	85
Programmer ID: 41	14	14	15	8	13	4	4	4	5	3	84
Programmer ID: 47	12	14	14	15	11	4	3	3	3	5	84
Programmer ID: 4	14	13	12	14	13	4	3	3	4	3	83
Programmer ID: 37	15	11	11	14	14	5	2	5	4	2	83
Programmer ID: 40	12	13	12	15	13	3	3	3	5	4	83
Programmer ID: 43	11	15	14	10	14	5	3	3	3	5	83
Programmer ID: 46	12	14	10	15	15	3	3	5	4	2	83
Programmer ID: 29	13	15	9	12	13	5	3	4	3	5	82
Programmer ID: 38	12	15	13	13	10	5	2	5	5	2	82
Programmer ID: 48	14	12	9	14	15	4	4	3	5	2	82
Programmer ID: 49	15	12	10	15	15	3	3	3	3	3	82
Programmer ID: 2	11	14	14	11	14	4	3	3	4	3	81
Programmer ID: 30	11	15	15	9	9	4	5	4	4	5	81
Programmer ID: 39	9	11	14	14	14	5	3	4	5	2	81
Programmer ID: 56	10	14	15	13	12	4	5	2	3	3	81
Programmer ID: 58	13	9	11	12	14	4	4	5	4	5	81
Programmer ID: 3	13	12	13	11	13	4	4	3	4	3	80
Programmer ID: 7	13	13	12	13	11	4	4	4	3	3	80
Programmer ID: 15	13	15	14	6	14	4	4	4	3	3	80
Programmer ID: 31	15	13	5	13	15	5	2	3	4	5	80
Programmer ID: 33	9	15	11	14	13	3	3	5	3	4	80
Programmer ID: 21	11	9	14	14	15	3	5	2	3	3	79
Programmer ID: 23	14	9	14	12	13	4	5	3	2	3	79
Programmer ID: 24	14	7	13	14	14	4	4	4	2	3	79
Programmer ID: 32	15	11	8	15	9	5	4	4	5	3	79
Programmer ID: 44	14	15	8	10	13	4	5	3	4	3	79
Programmer ID: 53	10	6	13	15	15	3	5	4	3	5	79
Programmer ID: 57	13	13	9	11	14	5	4	4	3	3	79
Programmer ID: 8	14	13	11	12	13	4	4	3	3	1	78
Programmer ID: 11	15	14	14	12	8	3	4	3	3	2	78
Programmer ID: 18	15	12	9	15	9	2	5	2	4	5	78
Programmer ID: 20	13	9	14	11	13	4	4	5	2	3	78
Programmer ID: 45	10	15	12	14	11	5	2	3	3	3	78
Programmer ID: 51	6	11	15	13	14	5	4	5	3	2	78
Programmer ID: 61	7	9	15	15	13	3	5	2	4	5	78
Programmer ID: 9	13	13	12	11	10	4	4	4	3	3	77
Programmer ID: 10	13	12	14	13	12	3	4	2	2	2	77
Programmer ID: 12	11	14	13	11	11	3	4	4	3	3	77
Programmer ID: 17	14	12	11	15	7	4	5	2	3	4	77
Programmer ID: 19	9	8	11	13	15	5	5	5	3	3	77
Programmer ID: 25	12	6	11	14	15	3	2	5	4	5	77
Programmer ID: 26	15	7	13	12	14	3	3	5	2	3	77
Programmer ID: 28	9	15	12	8	14	5	3	4	4	3	77
Programmer ID: 35	12	7	13	10	13	5	5	3	4	5	77
Programmer ID: 36	12	9	7	14	14	3	5	5	3	5	77
Programmer ID: 42	12	15	5	12	14	3	3	5	3	5	77
Programmer ID: 50	13	14	7	15	12	4	4	3	2	3	77
Programmer ID: 52	14	8	10	15	13	4	3	5	2	3	77
Programmer ID: 55	12	8	9	15	15	4	3	3	3	5	77
Programmer ID: 6	14	11	13	11	9	3	4	4	4	3	76
Programmer ID: 13	14	10	11	13	12	3	5	3	3	2	76
Programmer ID: 14	15	12	11	13	13	3	4	1	2	2	76
Programmer ID: 16	13	12	10	14	10	4	4	3	4	2	76
Programmer ID: 22	13	9	12	13	12	5	4	1	3	4	76
Programmer ID: 59	14	15	8	9	11	5	3	5	3	3	76

Table 5.7 shows the list of programmers from Company 2 who showed good performance in their overall annual performance appraisal, sorted in descending order of their scores. There were 76 programmers who showed good performance. Of these, there were seven best-fit programmers (ID: 50, 6, 76, 36, 44, 33 and 32) when compared with the results of their past annual performance appraisal. Of these seven best-fit programmers, only two programmers (ID 50 and ID 6) matched with the top two programmers based on the annual performance appraisal results. The other five programmers are ranked 4th (programmer ID 76), 8th (programmer ID 36), 9th (programmer ID 44), 16th (programmer ID 33) and 24th (programmer ID 32) in the annual performance appraisal. These results show that not all best-fit programmers are the top programmers based on the annual performance appraisal. Similarly, not all the 76 good programmers in the annual performance appraisal are also the best-fit programmers. This is because 69 of the programmers who were rated 'good' in the annual performance appraisal, were not rated 'good' for all the prognostic attributes required by Company 2.

Table 5.7: Results of the Past Annual Performance Appraisal of Programmers (Company 2)

Programmer ID	Att1	Att2	Att3	Att4	Att5	Weight Earned
Programmer ID: 50	10	10	10	10	8	9.6
Programmer ID: 6	8	10	10	10	9	9.4
Programmer ID: 11	10	9	10	10	7	9.2
Programmer ID: 76	8	10	10	10	8	9.2
Programmer ID: 10	10	7	10	8	10	9
Programmer ID: 14	9	10	10	9	7	9
Programmer ID: 24	6	10	10	10	9	9
Programmer ID: 36	10	9	10	8	8	9
Programmer ID: 44	10	8	9	10	8	9
Programmer ID: 8	9	10	7	10	8	8.8
Programmer ID: 26	7	7	10	10	10	8.8
Programmer ID: 37	10	10	9	9	6	8.8
Programmer ID: 7	10	7	9	10	7	8.6
Programmer ID: 15	9	9	7	8	10	8.6
Programmer ID: 22	8	10	7	8	10	8.6
Programmer ID: 33	8	8	8	9	10	8.6
Programmer ID: 58	10	8	10	7	8	8.6
Programmer ID: 64	5	9	10	10	9	8.6
Programmer ID: 66	8	9	10	7	9	8.6
Programmer ID: 69	10	10	9	7	7	8.6
Programmer ID: 70	9	7	10	10	7	8.6
Programmer ID: 5	8	9	8	7	10	8.4
Programmer ID: 25	5	10	9	8	10	8.4
Programmer ID: 32	8	9	8	8	9	8.4
Programmer ID: 40	10	7	7	10	8	8.4
Programmer ID: 46	7	8	10	8	9	8.4
Programmer ID: 47	8	7	10	9	8	8.4
Programmer ID: 53	8	8	6	10	10	8.4
Programmer ID: 55	8	9	8	7	10	8.4
Programmer ID: 60	5	10	7	10	10	8.4
Programmer ID: 61	10	8	9	7	8	8.4
Programmer ID: 62	10	8	8	9	7	8.4
Programmer ID: 63	8	10	9	9	6	8.4
Programmer ID: 72	5	9	10	9	9	8.4
Programmer ID: 73	9	10	7	7	9	8.4
Programmer ID: 75	9	7	8	9	9	8.4
Programmer ID: 9	6	8	10	10	7	8.2
Programmer ID: 13	6	9	8	8	10	8.2
Programmer ID: 27	7	9	6	9	10	8.2
Programmer ID: 34	8	10	8	9	6	8.2
Programmer ID: 39	7	8	9	8	9	8.2
Programmer ID: 51	8	10	7	7	9	8.2
Programmer ID: 52	5	10	9	8	9	8.2
Programmer ID: 54	8	8	9	9	7	8.2
Programmer ID: 57	10	8	8	8	7	8.2
Programmer ID: 65	10	8	6	8	9	8.2
Programmer ID: 74	9	10	9	7	6	8.2
Programmer ID: 12	7	8	10	7	8	8

Table 5.7: (Continued)

Programmer ID: 16	9	10	7	7	7	8
Programmer ID: 17	8	10	6	7	9	8
Programmer ID: 18	6	8	8	10	8	8
Programmer ID: 21	10	7	8	7	8	8
Programmer ID: 23	6	10	8	8	8	8
Programmer ID: 35	5	7	10	8	10	8
Programmer ID: 71	5	7	9	9	10	8
Programmer ID: 1	9	6	7	8	9	7.8
Programmer ID: 2	9	5	9	8	8	7.8
Programmer ID: 3	5	9	6	10	9	7.8
Programmer ID: 4	9	10	6	8	6	7.8
Programmer ID: 19	10	9	6	7	7	7.8
Programmer ID: 20	8	9	7	7	8	7.8
Programmer ID: 28	9	7	7	9	7	7.8
Programmer ID: 29	4	10	8	9	8	7.8
Programmer ID: 30	6	10	6	8	9	7.8
Programmer ID: 31	6	10	8	7	8	7.8
Programmer ID: 38	5	9	6	10	9	7.8
Programmer ID: 41	9	8	7	7	8	7.8
Programmer ID: 42	6	8	10	7	8	7.8
Programmer ID: 43	7	8	9	8	7	7.8
Programmer ID: 45	9	8	8	7	7	7.8
Programmer ID: 48	7	8	8	10	6	7.8
Programmer ID: 49	5	7	10	8	9	7.8
Programmer ID: 56	10	7	6	10	6	7.8
Programmer ID: 59	8	9	6	10	6	7.8
Programmer ID: 67	5	10	9	9	6	7.8
Programmer ID: 68	8	7	9	9	6	7.8

5.5 Performance Evaluation

This section presents the use of the confusion matrix to evaluate the performance of the constructed ANN classifier of the predicted outcomes. The performance of the constructed ANN classifier was evaluated using the information residing in the confusion matrix to find their value vis-à-vis the three standard indicators - accuracy, precision, and recall. The datasets of Company 1 and Company 2 were used to construct the ANN classifier in the proposed technique. Each company's dataset was divided into training set, validation set, and test set. The first two sets were used to construct the ANN classifier in the proposed technique. The third set - test set - was used to evaluate the classifier performance because all instances of this set of "neutral" test data were not seen by the classifier before. The constructed ANN classifier has three classes in the predicted output, and thus produces three possible classes - Good, Average, and Poor - in the confusion matrix table. To ease understanding of the classifier performance of each corresponding output class, the calculation of the three classes of the classifiers (3×3 confusion matrix) is simplified by converting each output class into a 2×2 confusion matrix (Sadawi, 2014). This can be treated as converting a multi-class classification problem into a binary classification problem (Thomas, Magand, Handmann, & Gepperth, 2015). By transforming each class into two alternative classes, for example, for the class "Good", the ANN classifier should predict the potential class of a programmer to be either "Good" or "Not Good" in his/her overall work performance. Based on the data value of each class in a 3×3 confusion matrix converted into a 2×2 confusion matrix, the values of the following statements can be formulated as follows (Sadawi, 2014):

i. True Positive (TP): obtain the value for a certain class from the diagonal value of the corresponding class (TP_G, TP_A, or TP_P).
ii. True Negative (TN): obtain the value for a certain class by adding all the values in the columns and rows, except its own class column and row (Sum of $TP_A + E_{AP} + E_{PA} + TP_P$).
iii. False Positive (FP): obtain the value for a certain class by adding all the values in their corresponding row except for its own TP value (Sum of $E_{GA} + E_{GP}$).
iv. False Negative (FN): obtain the value for a certain class by adding all the values in their corresponding column except for its own TP value (Sum of $E_{AG} + E_{PG}$).

The calculation for TN, FN and FP, as indicated in brackets, refers to the class, "Good" (TP_G), as shown in Figure 5.7.

		Actual Class		
		Good	Average	Poor
Predicted Class	Good	TP_G	E_{GA}	E_{GP}
	Average	E_{AG}	TP_A	E_{AP}
	Poor	E_{PG}	E_{PA}	TP_P

Keys: TP_G – True Positive (Good), TP_A - True Positive (Average), TP_P - True Positive (Poor).

Figure 5.7: The Structure of a 3×3 Confusion Matrix

Figure 5.7 shows the structure of a 3×3 confusion matrix which contains three classes - Good, Average, and Poor. The rows represent the predicted class, and the columns represent the actual class output from the ANN classifier. The true positive (TP) values of each class (TP_G, TP_A, TP_P) can be determined from the diagonal values, as shown in the figure. The mis-classified class is denoted by the letter E. The first subscript letter represents the predicted class and the second subscript letter indicates the actual class. For example, if the class "Average" is mis-classified into class "Good", it is denoted as E_{GA}. The other mis-classified classes are also denoted in the same manner, respectively.

5.5.1 Performance Evaluation of Company 1

As mentioned in Section 5.5, above, the test dataset is used for measuring the performance of the ANN classifier of the proposed technique.

		Actual Class		
		Good	Average	Poor
Predicted Class	Good	10	2	0
	Average	0	46	2
	Poor	0	5	6

Figure 5.8: 3 × 3 Confusion Matrix of Company 1

Figure 5.8 shows the 3×3 confusion matrix of the dataset of Company 1, which consists of three possible classes: Good, Average and Poor. As mentioned above, the confusion matrix indicates how accurate are the results predicted by the ANN classifier of the proposed technique. The values of the actual classes are indicated in the columns while the rows contain values of the predicted classes. This 3×3 confusion matrix can be transformed into a 2×2 confusion matrix for each corresponding class, as explained below.

i) Class “Good”

As shown in Figure 5.9, the class “Good” of Company 1 was converted into a two-classification problem, i.e., a 2×2 confusion matrix which has two possible classes - “Good” and “Not Good”. Using the values in Figure 5.6, we can calculate the values of TP, FN, FP and TN of the class “Good” as each of the values is represented in the 2×2 confusion matrix, as shown in Figure 5.9 (b). The accuracy, precision and recall indicators for the class “Good” can be calculated as follows:

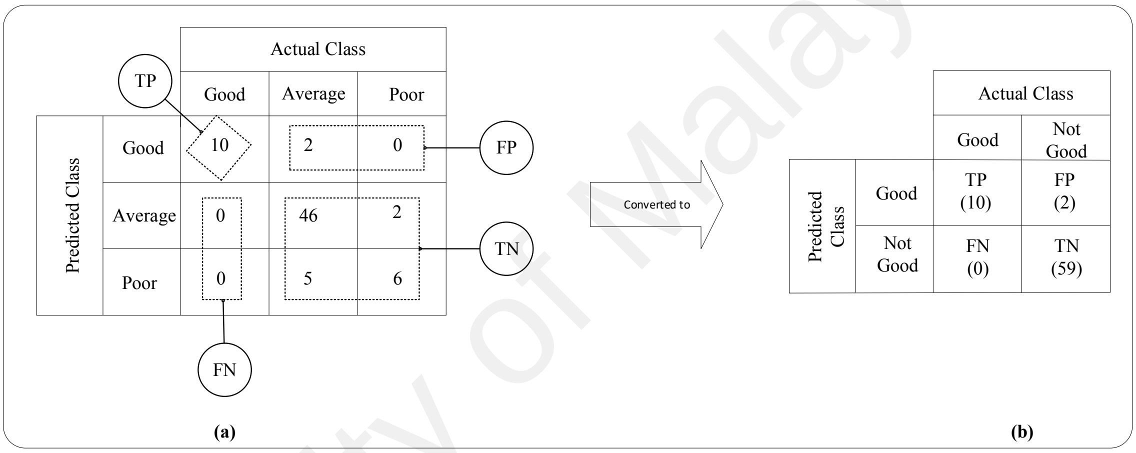


Figure 5.9: 2 × 2 Confusion Matrix of Class Good – Company 1

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{10+59}{10+59+2+0} = \frac{69}{71} = 0.972 \times 100, \text{ which is equal to } 97.2\%$$

$$Precision = \frac{\text{TP}}{TP+FP} = \frac{10}{10+2} = 0.833 \times 100, \text{ which is equal to } 83.3\%$$

$$Recall = \frac{\text{TP}}{TP+FN} = \frac{10}{10+0} = 1 \times 100, \text{ which is equal to } 100\%$$

In evaluating the performance of the proposed technique of class “Good” with regard to accuracy, precision and recall, the results show that the classifier prediction has achieved accuracy rates of 97.2%, 83.3%, and 100%, respectively. This means that the proposed technique’s ANN classifier has achieved an average classification rate (accuracy rate) of 97.2% in predicting the programmer’s performance for the class “Good” on the test dataset correctly; 83.3% frequency (percent of times) accuracy in classifying the programmers in the class “Good” correctly; and 100% recall ability in predicting correctly the class “Good” when its actual class is the class “Good”.

ii) Class “Average”

The performance of the proposed technique of the class “Average” is evaluated in the same manner with the class “Good” and class “Poor”, as shown in Figure 5.10.

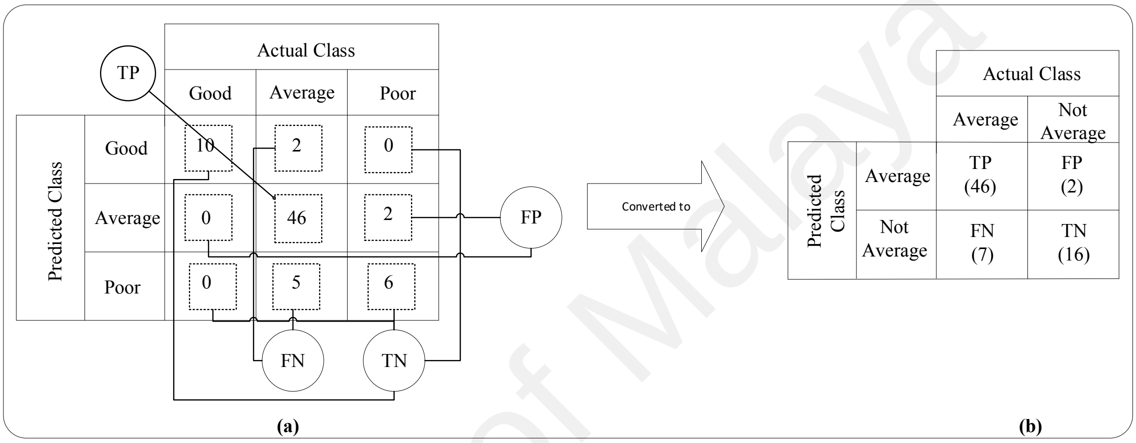


Figure 5.10: 2 × 2 Confusion Matrix of Class Average – Company 1

$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{46+16}{46+16+2+7} = \frac{62}{71}$ = 0.873 × 100, which is equal to 87.3%

$Precision = \frac{\text{TP}}{TP+FP} = \frac{46}{46+2} = \frac{46}{48}$ = 0.958 × 100, which is equal to 95.8%

$Recall = \frac{\text{TP}}{TP+FN} = \frac{46}{46+7} = \frac{46}{53}$ = 0.868 × 100, which is equal to 86.8%

Based on the calculated values of accuracy, precision and recall of class “Average”, the proposed technique’s ANN classifier achieved average 87.3% accuracy of the classification rate (accuracy rate) in predicting the programmer’s performance in the class “Average” on the test dataset correctly; 95.8% frequency (percent of times) accuracy in classifying the programmers in the class “Average” correctly; 86.8% accuracy in the recall ability to correctly predict as the class “Average” when its actual class is the class “Average”.

iii) Class “Poor”

Similarly, the performance of the proposed technique of the class “Poor” is calculated in the same manner as the class “Good” and class “Average”, as shown in Figure 5.11.

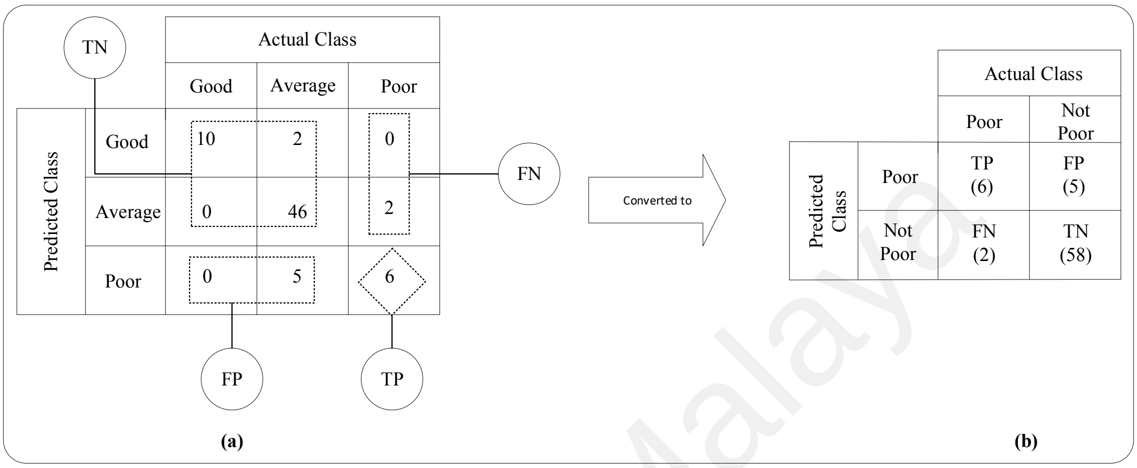


Figure 5.11: 2 × 2 Confusion Matrix of Class Poor – Company 1

$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{6+58}{6+58+5+2} = \frac{64}{71}$ = 0.901 × 100, which is equal to 90.1%

$Precision = \frac{\text{TP}}{TP+FP} = \frac{6}{6+5} = \frac{6}{11}$ = 0.545 × 100, which is equal to 54.5%

$Recall = \frac{\text{TP}}{TP+FN} = \frac{6}{6+2} = \frac{6}{8}$ = 0.75 × 100, which is equal to 75%

The proposed technique’s ANN classifier has achieved 90.1%, 54.5%, and 75%, with regard to accuracy, precision and recall, respectively. The precision values of the class “Poor” are lower than the precision values of the class “Good” and the class “Average”. This is due to the nature and size of the dataset of Company 1, as it contains an unequal distribution among its classes (imbalanced dataset), and this interferes with the ability to achieve frequency accuracy in classifying the programmers in the class “Poor”, correctly. Figure 5.11 shows higher instances of true negatives. In this context, however, it is more important to capture the positive cases. The ability to correctly predict the programmer’s performance class with 90.1%, accuracy rate, and the recall is able to correctly predict as the class “Poor” when its actual class is the class “Poor” with 75%, recall accuracy rate, indicates high predictive performance. Therefore, the performance of an ANN classifier of the class “Poor” is considered to be acceptable.

5.5.2 Performance Evaluation of Company 2

The performance evaluation of the ANN classifier of the proposed technique in Company 2 is shown in Figure 5.12. Similarly, the 3 × 3 confusion matrix using the test dataset was first constructed and then converted into a 2 × 2 confusion matrix. The same method of calculation was used to find the values for the three performance indicators – accuracy, precision and recall.

		Actual Class		
		Good	Average	Poor
Predicted Class	Good	13	4	0
	Average	2	38	1
	Poor	0	2	11

Figure 5.12: 3 × 3 Confusion Matrix of Company 2

i) Class “Good”

Figure 5.13 shows the steps to obtain the values for accuracy, precision and recall of class “Good”, respectively.

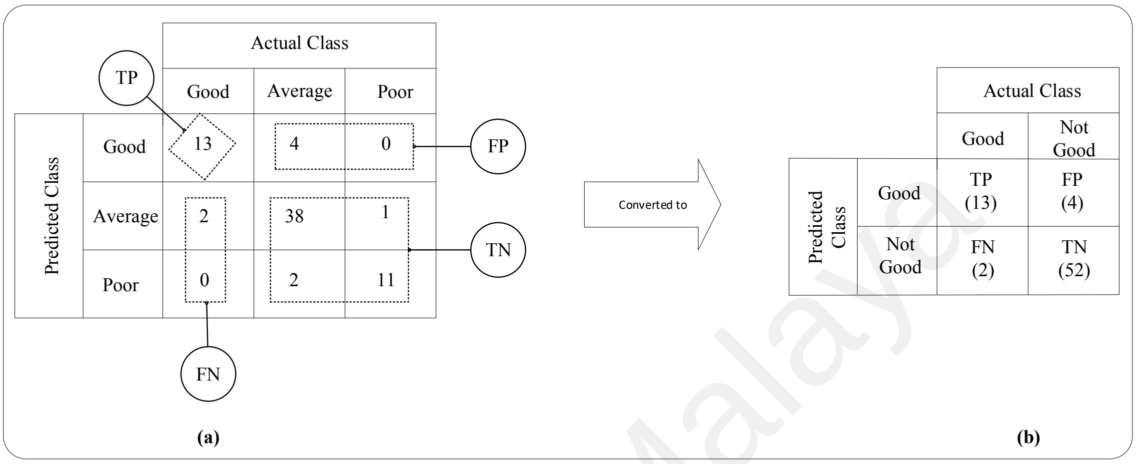


Figure 5.13: 2 × 2 Confusion Matrix of Class Good – Company 2

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{13+52}{13+52+4+2} = \frac{65}{71} = 0.915 \times 100, \text{ which is equal to } 91.5\%$$

$$Precision = \frac{\text{TP}}{TP+FP} = \frac{13}{13+4} = \frac{13}{17} = 0.765 \times 100, \text{ which is equal to } 76.5\%$$

$$Recall = \frac{\text{TP}}{TP+FN} = \frac{13}{13+2} = \frac{13}{15} = 0.867 \times 100, \text{ which is equal to } 86.7\%$$

In evaluating the performance of the proposed technique of class “Good” with regard to accuracy, precision and recall, the results show that the classifier prediction has achieved accuracy rates of 91.5%, 76.5%, and 86.7%, respectively. This means that the proposed technique’s ANN classifier has achieved an average classification rate (accuracy rate) of 91.5% in predicting the programmer’s performance for the class “Good” on the test dataset correctly; 76.5% frequency (percent of times) accuracy in classifying the programmers in the class “Good” correctly; 86.7% recall ability in predicting correctly the class “Good” when its actual class is the class “Good”.

ii) Class “Average”

Figure 5.14 shows the steps to obtain the values for accuracy, precision and recall of class “Average”, respectively.

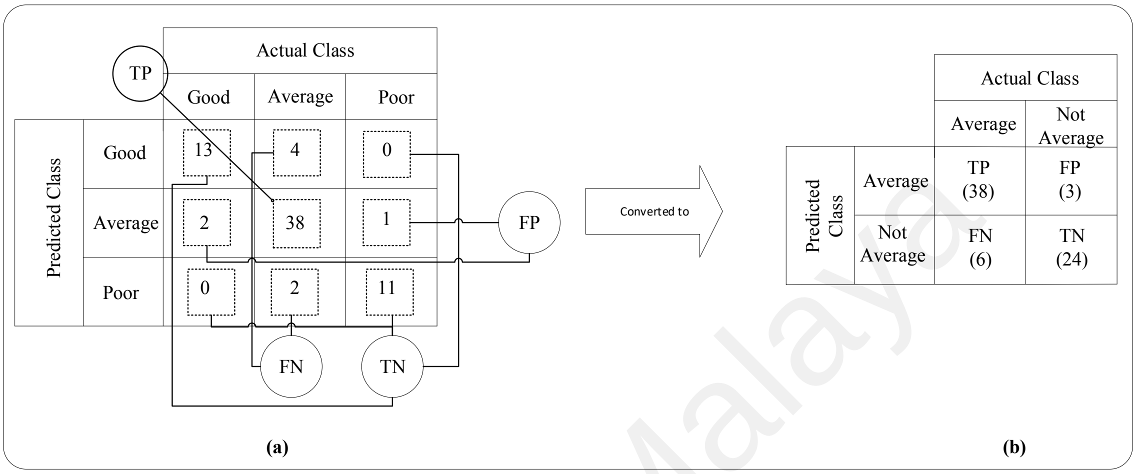


Figure 5.14: 2 × 2 Confusion Matrix of Class Average – Company 2

$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} = \frac{38+24}{38+24+3+6} = \frac{62}{71}$ = 0.873 × 100, which is equal to 87.3%

$Precision = \frac{\text{TP}}{TP+FP} = \frac{38}{38+3} = \frac{38}{41}$ = 0.927 × 100, which is equal to 92.7%

$Recall = \frac{\text{TP}}{TP+FN} = \frac{38}{38+6} = \frac{38}{44}$ = 0.864 × 100, which is equal to 86.4%

Based on the calculated values for accuracy, precision and recall of class “Average”, the proposed technique’s ANN classifier achieved average 87.3% accuracy in the classification rate (accuracy rate) in predicting the programmer’s performance in the class “Average” on the test dataset correctly; 92.7% frequency (percent of times) accuracy in classifying the programmers in the class “Average” correctly; 86.4% accuracy in the recall ability to correctly predict as the class “Average” when its actual class is the class “Average”.

iii) Class “Poor”

Figure 5.15 shows the steps to obtain the values for accuracy, precision and recall of class “Poor”, respectively.

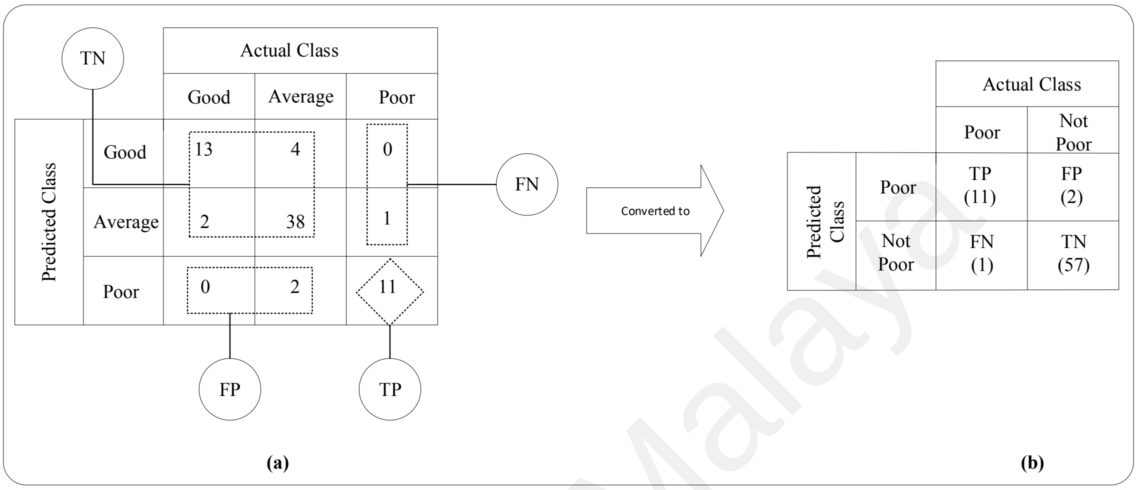


Figure 5.15: 2 × 2 Confusion Matrix of Class Poor – Company 2

$$Accuracy = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}} = \frac{11+57}{11+57+2+1} = \frac{68}{71} = 0.958 \times 100 \text{, which is equal to } 95.8\%$$

$$Precision = \frac{\mathrm{TP}}{TP+FP} = \frac{11}{11+2} = \frac{11}{13} = 0.846 \times 100 \text{, which is equal to } 84.6\%$$

$$Recall = \frac{\mathrm{TP}}{TP+FN} = \frac{11}{11+1} = \frac{11}{12} = 0.917 \times 100 \text{, which is equal to } 91.7\%$$

Based on the calculated values of accuracy, precision and recall of class “Poor”, the proposed technique’s ANN classifier achieved average 95.8% accuracy of the classification rate (accuracy rate) in predicting the programmer’s performance in the class “Poor” on the test dataset correctly; 84.6% frequency (percent of times) accuracy in classifying the programmers in the class “Poor” correctly; 91.7% accuracy in the recall ability to correctly predict as the class “Poor” when its actual class is the class “Poor”.

5.6 Summary

The results of the performance evaluation for both the companies are tabulated in Table 5.8 below.

Table 5.8: Comparison of Performance Evaluation Between Company 1 and Company 2

Company 1				Company 2			
Performance Evaluation	Class			Performance Evaluation	Class		
	Good	Average	Poor		Good	Average	Poor
Accuracy	97.2%	87.3%	90.1%	Accuracy	91.5%	87.3%	95.8%
Precision	83.3%	95.8%	54.5%	Precision	76.5%	92.7%	84.6%
Recall	100%	86.8%	75%	Recall	86.7%	86.4%	91.7%

A comparison of the performance evaluation results from Company 1 and Company 2 reveals different levels of performance between class "Good" and class "Poor" of these two software companies. It is clear from Table 5.8 that Company 1, class "Good" shows higher performance evaluation values on accuracy (97.2%), precision (83.3%), and recall (100%) compared to the performance evaluation values of class "Poor" with accuracy (90.1%), precision (54.5%), and recall (75%). On the other hand, Company 2, class "Good" shows lower evaluation performance values with accuracy (91.5%), precision (76.5%), and recall (86.7%) compared to the evaluation performance values of class "Poor" with accuracy (95.8%), precision (84.6%), and recall (91.7%). These differences are due to the analysis of different prognostic attributes (high potential attributes) that were selected by each company. For example, in Company 1, the prognostic attributes (as shown in Table 4.1) show the high likelihood in achieving a good performance when the selected prognostic attributes are rated "good". The prognostic attributes (as shown in

Table 4.1) did not show any significant value in the class “Poor”. On the other hand, in Company 2, the prognostic attributes (as shown in Table 4.2) show the high likelihood of achieving good performance when the selected prognostic attributes are rated “good”. Also, some significant values of the prognostic attributes (as shown in Table 4.2) were produced for the class “Poor” and thus there is high likelihood of poor performance when the selected prognostic attributes are rated “poor”.

Another possible reason for the different levels of performance shown in Table 5.8, could be due to the number of best-fit programmers who matched with their annual performance appraisal of Company 1 (Table 5.6 and Table 5.7) and also could be related to the different evaluation performance results. As in Company 1, there is matching between all predicted best-fit programmers and the top three programmers who obtained high scores in the annual performance appraisal. On the other hand, in Company 2, the first top two predicted best-fit programmers matched with the first top two programmers who obtained high scores in the annual performance appraisal. The remaining programmers with high scores were among the predicted best-fit programmers, but they were not matched in the order of their scores. Considering the results, overall, we believe that they reflect influence in the selection of the prognostic attributes on the performance evaluation results of the ANN classifier in the proposed technique of this research work.

CHAPTER 6: CONCLUSION AND DISCUSSION

This chapter discusses how the objectives of the research defined in Chapter 1 have been achieved. It also highlights the research contributions, its limitations, and suggests future research that can be undertaken.

6.1 Fulfillment of Research Objectives

Three research objectives were defined in Chapter 1, and each of these objectives was fulfilled as discussed, below.

Objective 1: To identify the attributes that should be used in assessing the type of programmers a software company needs.

To achieve this research objective, Bayes' Theorem was applied on the programmers' annual performance appraisal to identify the attributes that greatly influence the performance of programmers in a software company. For Company 1, seven attributes - Write Functionally Correct Code; Writes Aesthetically Pleasing Code; Performs Satisfactory Unit Test; Documents Code Well; Asks Questions When Needed; Communication Skills; and Corporate responsibility - were identified as the prognostic attributes (high potential attributes), which can affect a programmer's performance. For Company 2, five attributes - Technical Skill; Time Management; Documentation/Presentation; Teamwork/Cooperative; and Attitude and Self Growth - were identified as the prognostic attributes. Based on the result of a programmer's past annual performance appraisal, the programmers of both the companies who obtained high performance appraisal scores also showed good performance when evaluated against the identified prognostic attributes for their respective company. This shows that Bayes' Theorem is a good approach for identifying the prognostic attributes which a software company would like their programmers to possess.

Objective 2: To propose a recruitment and selection technique that can be used to determine the best-fit programmers for a software company.

This objective is achieved by constructing an ANN classifier that incorporates the identified prognostic attributes obtained from applying Bayes' Theorem. The identified prognostic attributes were imported into MATLAB as an input to implement the ANN classifier. The datasets of the two software companies (Company 1 and Company 2) were used to train the ANN classifier. The ANN classifier was used to predict the best-fit programmers for the two software companies.

Objective 3: To evaluate the proposed technique.

To fulfill this research objective, the predicted best-fit programmers identified using the proposed technique were compared with the programmers who were rated "good" in their annual performance appraisal. The results of the comparison show that for the two companies, all the predicted best-fit programmers were all among the list of programmers who were rated "good" in the annual performance appraisal - a 100% matching. In addition, the performance of the proposed technique's ANN classifier was evaluated using confusion matrix with regard to its accuracy, precision and recall. The results show that its performance is 97.2% and 87.3%, 95.8% and 54.5%, and 100% and 75% with regard to accuracy, precision and recall on the two test datasets of Company 1 and Company 2, respectively.

6.2 Research Limitations

This research has the following limitations:

i) The proposed technique was only used for predicting best-fit programmers software companies to employ. Other key personnel positions such as project managers, system engineer, system analysts, etc., in software or IT companies were not considered.

ii) Due to time constraint and difficulty in getting cooperation from software companies to participate in this study, only two software companies in India were willing to provide the needed datasets for us to implement and evaluate the proposed technique.

iii) The annual performance appraisal data only covered five years, 2010-2015, and were collected independently. Hence, this might have given rise to data error and bias in the data collection process as the researcher was not involved in the process directly.

6.3 Future Works

Future research on this topic should focus on expanding the research scope, which could include the following:

i) The proposed technique should be applied for predicting the best-fit personnel for other IT positions or even in other disciplines such as the best-fit athlete for a particular type of sports or the best-fit teacher to teach a particular subject in schools.

ii) To obtain more convincing findings on the evaluation of the proposed technique, larger size of sample data should be collected from more software companies of varying sizes, and from different countries. This would produce better comparative results from different countries and from different sizes of software companies.

iii) Other prediction techniques such as Decision Tree, Support Vector Machine (SVM), K-Nearest Neighbours, etc., can be incorporated in the proposed technique. In addition, to identify the most likely selection criteria that could affect an employee's work performance, other approaches such as linear regression, Naïve Bayes, fuzzy logic, etc., can be included in the proposed technique.

iv) The proposed technique should be more thoroughly evaluated. In our study, the evaluation was based on the comparison with past data (retrospective data) from the two software companies in India. This limited size of their historical data might not produce a representative result. Therefore, conducting a longitudinal study and then comparing the findings from the proposed technique with current data from the software companies could improve the accuracy of the study outcome.

6.4 Conclusion

This research proposes a technique to predict the best-fit programmers based on the pertinent skills (attributes) desired by software companies. Bayes' Theorem was used in identifying the pertinent attributes for assessing programmers' performance as required by the software companies. Artificial Neural Network (ANN) was used to classify the performance classes of the programmers based on the selected attributes to predict and determine the best-fit programmers. The accuracy of the proposed technique was evaluated using data collected from two software companies in India. The results show that the proposed technique can predict the best-fit programmers accurately. Hence, this proposed technique can contribute to the software development industry in terms of evaluating the existing programmers or new recruits as best-fit programmers. The proposed technique can be adapted to be applied in other discipline such as sports, education, etc.

REFERENCES

Afshari, A. R., Mojahed, M., Yusuff, R. M., Hong, T. S., & Ismail, M. Y. (2010). Personnel selection using ELECTRE. *Journal of Applied Sciences, 10*(23), 3068-3075. doi:10.3923/jas.2010.3068.3075

Agrawal, N. M., Khatri, N., & Srinivasan, R. (2012). Managing growth: Human resource management challenges facing the Indian software industry. *Journal of World Business*, 47(2), 159-166. doi:10.1016/j.jwb.2011.04.002

Akritidis, L., & Bozanis, P. (2013). A supervised machine learning classification algorithm for research articles. *Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC 13*. doi:10.1145/2480362.2480388

Ali, C. A., & Akayuure, P. (2016). Exploring postgraduate students' research knowledge and skills in normality tests and verifications. *US-China Education Review, 6*(1), 53-62. doi:10.17265/2161-623X/2016.01.005

Al-Kassem, A. H. (2017). Recruitment and selection practices in business process outsourcing industry. *Archives of Business Research,5*(3). doi:10.14738/abr.53.2180

Al-Otaibi, S. T., & Ykhlef, M. (2012). Job recommendation systems for enhancing e-recruitment process. In Proceedings of the International Conference on Information and Knowledge Engineering (IKE) (p. 1). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Amara, D., & Rabai, L. B. (2017). Towards a new framework of software reliability measurement based on software metrics. *Procedia Computer Science, 109*, 725-730. doi:10.1016/j.procs.2017.05.428

Amato, F., López, A., Peña-Méndez, E. M., Vaňhara, P., Hampl, A., & Havel, J. (2013). Artificial neural networks in medical diagnosis. *Journal of Applied Biomedicine,11*(2), 47-58. doi:10.2478/v10136-012-0031-x

Appavu, S., Rajaram, R., Nagammai, M., Priyanga, N., & Priyanka, S. (2011). Bayes Theorem and Information Gain based feature selection for maximizing the performance of classifiers. In N. Meghanathan, K. B. Kaushik & D. Nagamalai (Eds.), *Advances in Computer Science and Information Technology* (pp. 501-511). Berlin, Heidelberg: Springer.

Arockiaraj, C. (2013). Applications of neural networks in data mining. *International Journal Of Engineering And Science,3*(1), 8-11. Retrieved August 2, 2018, from http://www.researchinventy.com/papers/v3i1/B03108011.pdf

Assaf, A. G., & Tsionas, M. (2018). Bayes factors vs. P-values. *Tourism Management,67*, 17-31. doi:10.1016/j.tourman.2017.11.011

Atalag, K., Yang, H. Y., Tempero, E., & Warren, J. R. (2014). Evaluation of software maintainability with openEHR – a comparison of architectures. *International Journal of Medical Informatics*, 83(11), 849-859. doi:10.1016/j.ijmedinf.2014.07.006

Balamurugan, S. A., & Rajaram, R. (2009). Effective and efficient feature selection for large-scale data using Bayes' theorem. *International Journal of Automation and Computing,6*(1), 62-71. doi:10.1007/s11633-009-0062-2

Bayes, T., & Price, R. (1763). An essay towards solving a problem in the doctrine of chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S. *Philosophical Transactions of the Royal Society of London,53*, 370-418. doi:10.1098/rstl.1763.0053

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (2006). *Nonlinear Programming: Theory and Algorithms* (3rd ed.). Hoboken, NJ: Wiley-Interscience.

Beucher, A., Møller, A., & Greve, M. (2017). Artificial neural networks and decision tree classification for predicting soil drainage classes in Denmark. *Geoderma*. doi:10.1016/j.geoderma.2017.11.004

Billah, M., Waheed, S., & Hanifa, A. (2016). Stock market prediction using an improved training algorithm of neural network. In *International Conference on Electrical, Computer & Telecommunication Engineering (ICECTE)*. Rajshahi, Bangladesh: IEEE. doi:10.1109/ICECTE.2016.7879611

Breaugh, J. A., & Starke, M. (2000). Research on employee recruitment: So many studies, so many remaining questions. *Journal of Management,26*(3), 405-434. doi:10.1177/014920630002600303

Carroll, M., Marchington, M., Earnshaw, J., & Taylor, S. (1999). Recruitment in small firms. *Employee Relations,21*(3), 236-250. doi:10.1108/01425459910273080

Catal, C. (2012). Software mining and fault prediction. Wiley Interdisciplinary Reviews: *Data Mining and Knowledge Discovery*, 2(5), 420-426. doi:10.1002/widm.1067

Chakrabarti, S., Cox, E., Frank, E., Güting, R., Han, J., Jiang, X., . . . Witten, I. (2009). *Data mining: know it all*. Burlington, MA: Morgan Kaufmann.

Chapman, D. S., & Webster, J. (2003). The use of technologies in the recruiting, screening, and selection processes for job candidates. *International Journal of Selection and Assessment,11*(2-3), 113-120. doi:10.1111/1468-2389.00234

Chellappa, R., Diniz, P. S., Suykens, J. A., & Theodoridis, S. (2014). Chapter 15: Neural Networks. In *Signal processing theory and machine learning*(Vol. 1, pp. 817-855). Amsterdam: Elsevier. doi:10.1016/B978-0-12-396502-8.00015-2

Chen, L., & Cheng, C. (2005). Selecting IS personnel use fuzzy GDSS based on metric distance method. *European Journal of Operational Research,160*(3), 803-820. doi:10.1016/j.ejor.2003.07.003

Chen, S., Billings, S. A., & Grant, P. M. (2007). Non-linear system identification using neural networks. *International Journal of Control,*1191-1214. doi:https://doi.org/10.1080/00207179008934126

Chua, Y. P. (2016). *Mastering research methods* (2nd ed.). Shah Alam, Selangor: McGraw-Hill (Malaysia).

Coakes, S. J. (2013). *SPSS version 20.0 for windows: Analysis without anguish* (20 ed.). Milton, Queensland, Australia: John Wiley & Sons.

Collins, J. (2018). Recruitment, Evaluation, and Selection. In *HR Management in the Forensic Science Laboratory* (1st ed., pp. 203-226). London, UK: Academic Press. doi:10.1016/B978-0-12-801237-6.00012-9

Cooper, H., Hedges, L. V., & Valentine, J. C. (2009). *The handbook of research synthesis and meta-analysis* (2 ed.). New York, USA: Russell Sage Foundation.

Cui, Y., Xiong, H., Zheng, K., & Chen, J. (2012). On the application of BP neural network based on Levenberg-Marquardt algorithm in the diagnosis of mental disorders. *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*. doi:10.1109/cecnet.2012.6201941

Djelloul, I., Sari, Z., & Sidibe, I. D. (2018). Fault diagnosis of manufacturing systems using data mining techniques. *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT),*198-203. doi:10.1109/codit.2018.8394807

Durmus, T., Luhur, R., Daqqaq, T., Schwenke, C., Knobloch, G., Huppertz, A., . . . Lembcke, A. (2016). Individual selection of X-ray tube settings in computed tomography coronary angiography: Reliability of an automated software algorithm to maintain constant image quality. *European Journal of Radiology, 85*(5), 963-971. doi:10.1016/j.ejrad.2016.03.004

Faerber, F., Weitzel, T., & Keim, T. (2003). An automated recommendation approach to selection in personnel recruitment. In *Americas Conference on Information Systems (AMCIS)* (pp. 2329-2339). Tampa, Florida. Retrieved from http://aisel.aisnet.org/amcis2003/302

Fang, D., & Lan, Y. (2018). Weight modification method for performance evaluation model of computing system based on Bayes Theorem. *Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences - ICMSS 2018,*207-211. doi:10.1145/3180374.3180378

Farooqui, M. S., & Nagendra, A. (2014). The impact of person organization fit on job satisfaction and performance of the employees. *Procedia Economics and Finance*, 11, 122-129. doi:10.1016/s2212-5671(14)00182-8

Fazel-Zarandi, M. & Fox, M.S. (2009). Semantic matchmaking for job recruitment: An ontology-based hybrid approach. In Proceedings of the 3rd International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web at the 8th International Semantic Web Conference (ISWC 2010), Washington DC, USA.

Furia, C. A. (2016, August 28). Bayesian statistics in software engineering: Practical guide and case studies. Retrieved from https://arxiv.org/abs/1608.06865v2.

Furia, C. A. (2017). What good is Bayesian data analysis for software engineering? *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. doi:10.1109/icse-c.2017.92

García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining* (Vol. 72). Cham, Switzerland: Springer International Publishing. doi:10.1007/978-3-319-10247-4

Gelman, A. (2016). The problems with p-values are not just with p-values. The American Statistician. Online discussion: http://www.stat.columbia.edu/~gelman/research/published/asa_pvalues.pdf

Gibney, R., & Shang, J. (2007). Decision making in academia: A case of the dean selection process. *Mathematical and Computer Modelling,46*(7-8), 1030-1040. doi:10.1016/j.mcm.2007.03.024

Goodman, S. (2008). A dirty dozen: Twelve p-value misconceptions. *Seminars in Hematology,45*(3), 135-140. doi:10.1053/j.seminhematol.2008.04.003

Grant, E., & Sackman, H. (1967). An exploratory investigation of programmer performance under on-line and off-line conditions. *IEEE Transactions on Human Factors in Electronics, 8*(1), 33-48. doi:10.1109/thfe.1967.233303

Guo, P., Wu, W., Sheng, Q., Li, M., Liu, H., & Wang, Z. (2013). Prediction of soil organic matter using artificial neural network and topographic indicators in hilly areas. *Nutrient Cycling in Agroecosystems,95*(3), 333-344. doi:10.1007/s10705-013-9566-9

Gupta, S., & Suma, V. (2013). Empirical study on selection of team members for software projects – data mining approach. *International Journal of Computer Science and Informatics, 3*(2), 2231-5292.

Gupta, S., & Suma, V. (2015). Application and assessment of classification techniques on programmers' performance in software industry. *Journal of Software, 10*(9), 1096-1103. doi:10.17706//jsw.10.9.1096-1103

Gusdorf, M. (2008). Recruitment and selection: Hiring the right person a two-part learning module for undergraduate students. (Unpublished instructor's manual). Society for Human Resource Management (SHRM), Alexandria, Virginia, USA.

Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks,5*(6), 989-993. doi:10.1109/72.329697

Hair, J. F., Babin, B. J., Anderson, R. E., & Black, W. C. (2018). *Multivariate data analysis* (8 ed.). Hampshire, UK: Cengage Learning EMEA.

Han, J., Kamber, M., & Pei, J. (2011). *Data mining: Concepts and techniques* (3rd ed.). Amsterdam: Elsevier; Morgan Kaufmann.

Haykin, S. (2016). *Neural networks and learning machines* (3rd ed.). Delhi, INDIA: Pearson.

Heiat, A. (2016). Predicting employee attrition through data mining. *Journal of Business and Management,*228-232. Retrieved August 18, 2017, from http://wdsinet.org/Annual_Meetings/2016_Proceedings/papers/Paper228.pdf

Hinkle, D. E., Wiersma, W., & Jurs, S. G. (2002). *Applied statistics for the behavioral sciences* (5 ed.). Boston, USA: Houghton Mifflin.

Hsiao, W. H., Chang, T. S., Huang, M. S., & Chen, Y. C. (2011). Selection criteria of recruitment for information systems employees: Using the analytic hierarchy process (AHP) method. *African Journal of Business Management,5*(15), 6201-6209. doi:DOI: 10.5897/AJBM11.185

Jantan, H., Hamdan, A. R., Othman, A. Z., & Puteh, M. (2010). Applying data mining classification techniques for employee's performance prediction. *In Knowledge management 5th International Conference (KMICe2010)* (pp. 645-652). Kuala Terengganu, Malaysia: University Utara Malaysia.

Johnson, J. (2018). *Decision latency theory: It is all about the interval* (CHAOS Report, pp. 1-70, Rep.). The Standish Group. Retrieved November 8, 2018, from https://www.standishgroup.com/store/services/10-chaos-report-decision-latency-theory-2018-package.html

Kalugina, E., & Shvydun, S. (2014). An effective personnel selection model. *Procedia Computer Science, 31*, 1102-1106. doi:10.1016/j.procs.2014.05.365

Karsak, E. E. (2001). Personnel selection using a fuzzy MCDM approach based on ideal and anti-ideal solutions. In: Köksalan M., Zionts S. (Ed.), *Multiple Criteria Decision Making in the New Millennium: Lecture Notes in Economics and Mathematical Systems* (Vol. 507, pp. 393-402). Berlin, Heidelberg: Springer.

Kamma, D. (2014). Study of task processes for improving programmer productivity. In *Proceedings of the 36th International Conference on Software Engineering* (pp. 702-705). Hyderabad, India: ACM Press. doi:10.1145/2591062.2591084

Kavitha, M., & Thomas, J. (2017). Predicting employee retention using decision trees and neural networks. *International Journal of Contemporary Research in Computer Science and Technology (IJCRCST),3*(3), 86-88.

Kelemenis, A., & Askounis, D. (2010). A new TOPSIS-based multi-criteria approach to personnel selection. *Expert Systems with Applications,37*(7), 4999-5008. doi:10.1016/j.eswa.2009.12.013

Khan, N., Gaurav, D., & Kandl, T. (2013). Performance evaluation of Levenberg-Marquardt technique in error reduction for diabetes condition classification. *Procedia Computer Science,18*, 2629-2637. doi:10.1016/j.procs.2013.05.455

Khosravi, A., Hussin, A. R., & Nilashi, M. (2018). Toward software quality enhancement by customer knowledge management in software companies. *Telematics and Informatics, 35*(1), 18-37. doi:10.1016/j.tele.2017.09.007

Kianto, A., Sáenz, J., & Aramburu, N. (2017). Knowledge-based human resource management practices, intellectual capital and innovation. *Journal of Business Research*, 81, 11-20. doi:10.1016/j.jbusres.2017.07.018

Kosti, M. V., Georgiadis, K., Adamos, D. A., Laskaris, N., Spinellis, D., & Angelis, L. (2018). Towards an affordable brain computer interface for the assessment of programmers' mental workload. *International Journal of Human-Computer Studies*, 115, 52-66. doi:10.1016/j.ijhcs.2018.03.002

Kristof-Brown, A. L., & Billsberry, J. (2013). *Organizational fit: Key issues and new directions*(1st ed.). Hoboken, NJ: Wiley-Blackwell.

Kruglanski, A. W., & Higgins, E. T. (2007). *Social psychology : Handbook of basic principles* (2nd ed.). New York, USA: Guilford Press.

Kumar, G. S., & Rajasekhar, K. (2017). Performance analysis of Levenberg - Marquardt and Steepest Descent algorithms based ANN to predict compressive strength of SIFCON using manufactured sand. *Engineering Science and Technology, an International Journal,20*(4), 1396-1405. doi:10.1016/j.jestch.2017.07.005

Kwak, N. K., McCarthy, K. J., & Parker, G. E. (1997). A human resource planning model for hospital/medical technologists: An analytic hierarchy process approach. *Journal of Medical Systems,21*(3), 173-187.

Lang, S., Laumer, S., Maier, C., & Eckhardt, A. (2011). Drivers, challenges and consequences of E-recruiting. *Proceedings of the 49th SIGMIS Annual Conference on Computer Personnel Research - SIGMIS-CPR 11*. doi:10.1145/1982143.1982152

Larsson, R., Brousseau, K. R., Kling, K., & Sweet, P. L. (2007). Building motivational capital through career concept and culture fit. *Career Development International,12*(4), 361-381. doi:10.1108/13620430710756753

Laumer, S., & Eckhardt, A. (2010). Analyzing IT personnels perception of job-related factors in good and bad times. *Proceedings of the 2010 Special Interest Group on Management Information Systems 48th Annual Conference on Computer Personnel Research on Computer Personnel Research - SIGMIS-CPR 10*. doi:10.1145/1796900.1796939

Laumer, S., Eckhardt, A., & Weitzel, T. (2009). Status quo and trends in e-recruiting - Results from an empirical analysis. *Proceedings of the 2009 International Conference on Information Resources Management (CONF-IRM)*, Dubai, United Arab Emirates (UAE)

Lee, I. (2007). An architecture for a next-generation holistic e-recruiting system. *Communications of the ACM,50*(7), 81-85. doi:10.1145/1272516.1272518

Lin, B., & Stasinskaya, V. S. (2002). Data warehousing management issues in online recruiting. *Human Systems Management,21*(1), 1-8.

Lin, H. (2010). Personnel selection using analytic network process and fuzzy data envelopment analysis approaches. *Computers & Industrial Engineering,59*(4), 937-944. doi:10.1016/j.cie.2010.09.004

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015, October 17). A critical review of recurrent neural networks for sequence learning. Machine Learning, 1-38. Retrieved August 2, 2018, from https://arxiv.org/abs/1506.00019v4.

Madsen, K., Nielsen, H. B., & Tingleff, O. (2004). *Methods for non-linear least squares problems* (2nd ed.). Lyngby, Denmark: Informatics and Mathematical Modelling, Technical University of Denmark.

Malinowski, J., Keim, T., & Weitzel, T. (2005). Analyzing the impact of IS support on recruitment processes: An E-recruitment phase model. *Proceedings of the 2005 PACIFIC ASIA CONFERENCE ON INFORMATION SYSTEMS (PACIS)* 81, 977-988. https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1193&context=pacis2005

Mammadli, S. (2017). Financial time series prediction using artificial neural network based on Levenberg-Marquardt algorithm. *Procedia Computer Science,120*, 602-607. doi:10.1016/j.procs.2017.11.285

Mammadova, M. H., Jabrayilova, Z. Q., & Mammadzada, F. R. (2016). Fuzzy multi-scenario approach to decision-making support in human resource management. *Recent Developments and New Direction in Soft-Computing Foundations and Applications Studies in Fuzziness and Soft Computing,342*, 19-36. doi:10.1007/978-3-319-32229-2_3

Mani, K., & Kalpana, P. (2016). An efficient feature selection based on Bayes' Theorem, self information and sequential forward selection. *International Journal of Information Engineering and Electronic Business,8*(6), 46-54. doi:10.5815/ijieeb.2016.06.06

Mimović, P., Stanković, J., & Milić, V. J. (2015). Decision-making under uncertainty – the integrated approach of the AHP and Bayesian analysis. *Economic Research-Ekonomska Istraživanja,28*(1), 868-878. doi:10.1080/1331677x.2015.1092309

Min, Z., Xiao, L., Cao, L., & Yan, H. (2017). Application of the neural network in diagnosis of breast cancer based on levenberg-marquardt algorithm. *2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC),*268-272. doi:10.1109/spac.2017.8304288

Mohamad, N., Zaini, F., Johari, A., Yassin, I., & Zabidi, A. (2010). Comparison between Levenberg-Marquardt and Scaled Conjugate Gradient training algorithms for breast cancer diagnosis using MLP. *2010 6th International Colloquium on Signal Processing & Its Applications*. doi:10.1109/cspa.2010.5545325

Mohd, N., & Yahya, Y. (2018). A data mining approach for prediction of students depression using logistic regression and artificial neural network. *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication - IMCOM 18*. doi:10.1145/3164541.3164604

Mondy, R. W., & Martocchio, J. J. (2016). *Human resource management* (14th ed.). Harlow: Pearson Education Limited.

Mukherjee, I., & Routroy, S. (2012). Comparing the performance of neural networks developed by using Levenberg–Marquardt and Quasi-Newton with the gradient descent algorithm for modelling a multiple response grinding process. *Expert Systems with Applications,39*(3), 2397-2407. doi:10.1016/j.eswa.2011.08.087

Musa, N., Junaini, S. N., & Bujang, Y. R. (2006). Improving usability of e-recruitment website: A preliminary study on Sarawak government website. *Proceedings of Pacific Asia Conference on Information Systems (PACIS 2006)*, Kuala Lumpur, Malaysia

Nair, T. R., Suma, V., & Tiwari, P. K. (2012). Analysis of test efficiency during software development process. 2 nd Annual International Conference on Software Engineering and Applications (SEA 2011). Retrieved December 6, 2017, from https://arxiv.org/ftp/arxiv/papers/1203/1203.6445.pdf

Nanz, S., & Furia, C. A. (2015). A comparative study of programming languages in Rosetta code. *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. doi:10.1109/icse.2015.90

Ozdemir, A. (2013). A two-phase multi criteria dynamic programing approach for personnel selection process. *Problems and Perspectives in Management,11*(2), 98-108.

Parry, E., & Tyson, S. (2008). An analysis of the use and success of online recruitment methods in the UK. *Human Resource Management Journal,18*(3), 257-274. doi:10.1111/j.1748-8583.2008.00070.x

Perezgonzalez, J. D. (2016). Commentary: How Bayes factors change scientific practice. *Frontiers in Psychology,7*. doi:10.3389/fpsyg.2016.01504

Peterson, M. (2017). *An introduction to decision theory* (2nd ed., Cambridge Introductions to Philosophy). Cambridge, UK: Cambridge University Press.

Pradhan, R. K., & Jena, L. K. (2016). Employee performance at workplace: Conceptual model and empirical validation. *Business Perspectives and Research*, 5(1), 69-85. doi:10.1177/2278533716671630

Pratt, M. K. (2014, November 18). 10 hottest IT skills for 2015. *Computerworld Digital Magazine*. Retrieved August 18, 2016, from https://www.computerworld.com/article/2844020/it-careers/10-hottest-it-skills-for-2015.html.

Prechelt, L. (2000). An empirical study of working speed differences between software engineers for various kinds of task. *IEEE Transactions on Software Engineering,* 1-11. doi:10.1.1.43.4788

Prieto, A., Prieto, B., Ortigosa, E. M., Ros, E., Pelayo, F., Ortega, J., & Rojas, I. (2016). Neural networks: An overview of early research, current frameworks and new challenges. *Neurocomputing,214*, 242-268. doi:10.1016/j.neucom.2016.06.014

Qin, C., Zhu, H., Xu, T., Zhu, C., Jiang, L., Chen, E., & Xiong, H. (2018). Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval - SIGIR 18,*25-34. doi:10.1145/3209978.3210025

Radant, O., Palacios, R. C., & Stantchev, V. (2015). Demographic change – Towards a framework to manage IT-personnel in times of scarcity of talent. *Proceedings of the 3rd International Conference on Technological Ecosystems for Enhancing Multiculturality - TEEM 15,*623-630. doi:10.1145/2808580.2808676

Rossel, R. V., & Behrens, T. (2010). Using data mining to model and interpret soil diffuse reflectance spectra. *Geoderma,158*(1-2), 46-54. doi:10.1016/j.geoderma.2009.12.025

Rouder, J. N., Speckman, P. L., Sun, D., Morey, R. D., & Iverson, G. (2009). Bayesian t tests for accepting and rejecting the null hypothesis. *Psychonomic Bulletin & Review,16*(2), 225-237. doi:10.3758/pbr.16.2.225

Sadawi, N. (2014, August 30). Evaluating classifiers: Confusion matrix for multiple classes [Video file]. Retrieved from https://www.youtube.com/watch?v=FAr2GmWNbT0

Sarna, G., & Bhatia, M. P. (2016). A probalistic approach to automatically extract new words from social media. *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. doi:10.1109/asonam.2016.7752316

Segall, M. D., & Champness, E. J. (2015). The challenges of making decisions using uncertain data. *Journal of Computer-Aided Molecular Design, 29*(9), 809-816. doi:10.1007/s10822-015-9855-2

Sekaran, U., & Bougie, R. (2016). *Research methods for business: A skill building approach* (7 ed.). Chichester, West Sussex, UK: John Wiley & Sons.

Seol, I., & Sarkis, J. (2005). A multi-attribute model for internal auditor selection. *Managerial Auditing Journal,20*(8), 876-892. doi:10.1108/02686900510619700

Shih, H., Huang, L., & Shyur, H. (2005). Recruitment and selection processes through an effective GDSS. *Computers & Mathematics with Applications,50*(10-12), 1543-1558. doi:10.1016/j.camwa.2005.08.026

Silva, I. N., Spatti, D. H., Flauzino, R. A., Liboni, L. H., & Alves, S. F. (2016). Artificial neural network architectures and training processes. *Artificial Neural Networks,*21-28. doi:10.1007/978-3-319-43162-8_2

Singh, Y., & Kumar, P. (2010). Prediction of software reliability using feed-forward neural networks. *2010 International Conference on Computational Intelligence and Software Engineering,35*(5), 1-6. doi:10.1145/1838687.1838709

Spačková, O., & Straub, D. (2012). Dynamic Bayesian network for probabilistic modeling of tunnel excavation processes. *Computer-Aided Civil and Infrastructure Engineering,28*(1), 1-21. doi:10.1111/j.1467-8667.2012.00759.x

Spanos, A. (1986). *Statistical foundations of econometric modelling* (1st ed.). Cambridge, UK: Cambridge University Press.

Stone, J. V. (2015). *Bayes rules with MATLAB: A tutorial introduction to Bayesian analysis*. Sheffield, UK: Sebtel Press.

Strohmeier, S., & Piazza, F. (2013). Domain driven data mining in human resource management: A review of current research. *Expert Systems with Applications, 40*(7), 2410-2420. doi:10.1016/j.eswa.2012.10.059

Sulehria, H. K., & Zhang, Y. (2007). Hopfield neural networks: A survey. In *Proceeding AIKED'07 Proceedings of the 6th Conference on 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases* (Vol. 6, pp. 125-130). Corfu Island, Greece: Association for Computing Machinery.

Swider, B. W., Zimmerman, R. D., & Barrick, M. R. (2015). Searching for the right fit: Development of applicant person-organization fit perceptions during the recruitment process. *Journal of Applied Psychology*, 100(3), 880-893. doi:10.1037/a0038357

Szalkai, B., & Grolmusz, V. (2018). Near perfect protein multi-label classification with deep neural networks. *Methods,132*, 50-56. doi:10.1016/j.ymeth.2017.06.034

Thakur, G. S., & Gupta, A. (2015). Developing a framework for prediction of human performance capability using ensemble techniques. *Computer Science & Information Technology (CS & IT),*163-173. doi:10.5121/csit.2015.50214

Thakur, G. S., Gupta, A., & Gupta, S. (2015). Data mining for prediction of human performance capability in the software industry. *International Journal of Data Mining & Knowledge Management Process,* *5*(2), 53-64. doi:10.5121/ijdkp.2015.5205

Thomas, K., Magand, S., Handmann, U., & Gepperth, A. (2015). A pragmatic approach to multi-class classification. In *2015 International Joint Conference on Neural Networks (IJCNN)*(pp. 1-8). Killarney, Ireland: IEEE. doi:10.1109/IJCNN.2015.7280768

Thompson, H., & Ahrens, L. (2015). Identifying talent in your selection decisions. *Nurse Leader*, 13(4), 48-51. doi:10.1016/j.mnl.2015.05.011

Uddin, M. A., Jameel, M., & Razak, H. A. (2015). Application of artificial neural network in fixed offshore structures. *Indian Journal of Geo-Marine Sciences,44*(3), 394-403. Retrieved August 18, 2016, from http://nopr.niscair.res.in/bitstream/123456789/34697/1/IJMS 44(3) 394-403.pdf

Vianen, A. E., & Pater, I. E. (2012). Content and development of newcomer person-organization fit: An agenda for future research. In *The Oxford handbook of organizational socialization* (1st ed., Oxford Library of Psychology, pp. 139-157). New York, NY: Oxford University Press. doi:10.1093/oxfordhb/9780199763672.013.0008

Vidya, A. R., Sonawane, B., & Bhutad, S. (2018). Computer-aided breast cancer detection and diagnosis system. *International Journal of Modern Trends in Engineering & Research,4*(12), 304-310. doi:10.21884/ijmter.2017.4418.tj1jx

Waheed, S., Zaim, A. H., Zaim, H., Sertbas, A., & Akyokus, S. (2013). Application of neural networks in talent management. In *International Conference on Electrical Information and Communication Technology (EICT)*. Khulna, Bangladesh: IEEE. doi:10.1109/EICT.2014.6777833

Wang, X., & Jiang, Y. (2010). The study of enterprise human resource evaluation research based on neural network. In *2010 Third International Symposium on Information Processing*. Qingdao, China: IEEE. doi:10.1109/ISIP.2010.13

Wang, K. Y., & Shun, H. Y. (2016). Applying back propagation neural networks in the prediction of management associate work retention for small and medium enterprises. *Universal Journal of Management,4*(5), 223-227. doi:10.13189/ujm.2016.040501

Watanabe, S. (2012, August 31). A widely applicable Bayesian information criterion. *Journal of Machine Learning Research*. Retrieved from https://arxiv.org/abs/1208.6338v1

Weitzel, T., Eckhardt, A., & Laumer, S. (2009). A framework for recruiting IT talent: lessons from Siemens. *MIS Quarterly Executive (MISQE),8*(4), 123-197.

Wren, G. P. (2012). Ai tools in decision making support systems: A review. *International Journal on Artificial Intelligence Tools,21*(2), 1-13. doi:10.1142/s0218213012400052

Wu, W., Li, A., He, X., Ma, R., Liu, H., & Lv, J. (2018). A comparison of support vector machines, artificial neural network and classification tree for identifying soil texture classes in southwest China. *Computers and Electronics in Agriculture,144*, 86-93. doi:10.1016/j.compag.2017.11.037

Zhang, L., Wu, X., Skibniewski, M. J., Zhong, J., & Lu, Y. (2014). Bayesian-network-based safety risk analysis in construction projects. *Reliability Engineering & System Safety,131*, 29-39. doi:10.1016/j.ress.2014.06.006

Zhang, L., Ekyalimpa, R., Hague, S., Werner, M., & Abourizk, S. (2015). Updating geological conditions using Bayes theorem and Markov chain. *2015 Winter Simulation Conference (WSC)*. doi:10.1109/wsc.2015.7408498

LIST OF PUBLICATIONS AND PAPERS PRESENTED

During this research, a paper has been submitted, accepted and published in an international conference, as follows:

1. Prathan S., Ow S.H. (2017, November). A model for predicting and determining the best-fit programmers using prognostic attributes. In: Alfred R., Iida H., Ag. Ibrahim A., Lim Y. (Ed.), Paper presented at International Conference on Computational Science and Technology (ICCST), Kuala Lumpur, Malaysia (Vol. 488, pp. 294-301). Singapore: Springer.

The first page of the published paper is given as reference below.