Department of Information Technology

Failures Prediction Approach in Agile Software Development

Prepared by:

Bulqees Thani Alajaleen

Supervised by

Dr. Aysh Alhroob

This Thesis is submitted to the Faculty of Information Technology as a Partial Fulfilment of the Requirement for Master Degree in Software Engineering

August 2020

The undersigned have examined the thesis entitled "FAILURES PREDICTION APPROACH IN AGILE SOFTWARE DEVELOPMENT" presented by Bulqees Al-Ajaleen, a Candidate for the degree Master of Science in Software Engineering and hereby certify that it is worthy of acceptance.

د. عايش الحروب

31-8-2020

Date

Dr. Aysh Alhroob

31.8.2020

Date

prof. Dr. Mohammad Al Fayoumi

31.8.2020

Date

Dr. Mohammad Muhairat

إقرار تفويض

أنا بلقيس العجالين ـ افوض جامعة الاسراء بتزويد نسخ من رسالتي ورقيا والكترونيا للمكتبات او المنظمات او الهيئات او المؤسسات المعنية بالأبحاث والدراسات العليا عند طلبها.

التوقيع :

التاريخ : 31/8/2020

AUTHORIZATION STATEMENT

I am Bulqees Al-Ajaleen, authorize Isra University to provide hard copies or soft copies of my thesis to libraries, institutions or individuals upon their request.

Signature :

Date : 31/8/2020

DEDICATION

I dedicate this message to the great mother, the honorable nanny, the generous lady, who endured the pressures of life without getting bored to realize the wishes, who did not hesitate for a moment to provide all means of assistance, comfort and continuous support for the purposes of my arrival at this moment..

To the great one , my father, who planted in us all the good, beautiful meanings that he seeks in life trying to provide all means of comfort and success for us, I cannot help to describe you at this moment.

To my sisters, brothers, and all friends for their constant encouragement and continuous support in the most difficult times.

To colleagues and colleagues of the Master's Trip, all thanks and respect, and I wish you success in your journey..

ACKNOWLEDGMENT

After a long time of intensive research, preparation and work, today I am reaping the fruits of this fun and arduous fatigue at the same time, it has been an enjoyable period of work and continuous learning .

Writing this letter has had positive effects not only on the scientific level, but also on a personal level. I would like to extend great gratitude and thanks to all the people who provided various kinds of support, assistance and motivation non-stop over the course of this period.

First of all, I extend my sincere thanks, full of feelings of respect and affection to my scientific example, my supervisor and counselor, Dr. Aysh AlHroub, who had a great role in reaching this moment through our meetings in the office or permanent contact that was available at all times and I do not remember not responding at any time. A time despite the difficult conditions, we are going through.

I would also like to extend my deepest gratitude and sincere thanks and love to my parents, my brothers, and sisters for the constant support and motivation at the most difficult times I hope to be a source of pride and happiness to you , Were it not for your presence, this achievement would not have been.

Table of Contents

LIST OF ABBREVETIONS

Abbreviation	**Full Expression**
SVM	Support Vector Machine
CC	Correlation Coefficient
MTBF	Mean time between failures
ADT	Administrative Delay Time
V&V	Validation and Verification
AI	Artificial Antelligence
WWW	World Wide Web
ASD	Agile software development
XP	Extreme Programming
FDD	Feature-Driven Development
DSDM	Dynamic Systems Development Method
LNN	Linguistic Neutrosophic Numbers
DP-CNN	Predictive Defect over the Convolutional Neural Network
HCI	Human-Computer Interaction

Table of Figures

ABSTRACT

Software failure prediction is an important activity during agile software development as it can help managers to identify the failure modules. Thus, it can reduce the test time, cost and assign testing resources efficiently. To ensure that the development of the software is likely to fail in a specific level, there are two techniques are used in this work, Support Vector Machine (SVM) to determine the factors leading to failure, and to define the dependent and independent variables the correlation coefficient (CC) has been used. RapidMiner Studio9.4 has been used to perform all the required steps from preparing the primary data to visualizing the results and evaluating the outputs, as well as verifying and improving them in a unified environment.

Two datasets are used in this work, the results for the first one indicate that the percentage of failure to predict the time used in the test is for all 181 rows, for all test times recorded, is 3% for Mean time between failures (MTBF). Whereas, SVM achieved a 97% success in predicting compared to previous work whose results indicated that the use of Administrative Delay Time (ADT) achieved a statistically significant overall success rate of 93.5%. At the same time, the second dataset result indicates that the percentage of failure to predict the time used or experiment in the test is for all 1091 rows, for all test times recorded, is 1.5% for MTBF, SVM achieved 98.5% prediction.

Keywords: Software Failure, Agile, Support Vector Machine, Correlation Coefficient

CHAPTER ONE
INTRODUCTION

CHAPTER 1:

INTRODUCTION

1.1. Overview

Because of the will increases in size and complexity of software products, organizations have become more interested in detecting failures before starting the developing stage to reduce the effort and cost.

Prediction of failures during the software development process is a crucial activity because it helps development teams to focus on the sprint that has a high probability to have a defect. It also helps management teams to assign and distribute resources efficiently during testing, and thus reducing development costs.

Software failures detection during software development in Agile and after releasing the software product has bad consequences on the reputation of software organizations. Therefore, the organizations are imposed on fixing these defects to avoid cost inflation. In this regard, both the software industry and academic researchers are working hard to collect datasets and build models to predict the failures before starting the development phase and monitor failures during software development.

1.2. Problem Statement

Software failures detection during software development in Agile and after releasing the software product has a bad consequence on the reputation of software organizations. Therefore, the organizations are imposed on fixing these defects to avoid cost inflation.

In this regard, both the software industry and academic researchers are working hard to collect datasets and build models to predict the failures before starting the development phase and monitor failures during software development.

1.3. Research Question

The question to answer in this research is: How does the historical data analysis of scrum agile help in the prediction of software failures?

The agile software improvement lifestyles cycle is based totally on the perception of modern improvement, iterative deliverables, and context-based assessments as nicely. At some stage in the process of agile software program improvement, a big quantity of information is being generated inside, sprints definition, making plans achieved through the Scrum master, test cases suites, time-relevant metrics, number of failures, time spent in development, check out the cost and other such procedures and documents (Cooper, 2014).

Scrum Agile can be defined as a framework that is iterative and incremental for projects and product or application development. Its job is to structure development in work cycles called Sprints.

The scrum agile procedure includes numerous sprints; in every dash, a positive software function is evolved, examined, delicate after which documented. But, s agile development is specially structured at the undertaking context, checking out is completed variously in every sprint data on each other (Glaiel, 2012).

However, Agile is fundamentally an iterative and lightweight methodology for designing and developing programs that changed that was established in the late nineties of the nineteenth century to be very similar to the high development of the WWW network (World Wide Web). (Al-Zewairi, Biltawi, Etaiwi, & Shaout, 2017).

Agile methodologies are a group of incremental and iterative means which might be quite efficient and had been applied in venture control. Kanban and Scrum Agile are considered effective techniques of agile challenge control in the field of software improvement. (Lei, Ganjeizadeh, Jayachandran, & Ozcan, 2017).

The goal of Scrum Agile and Kanban is achieved by improving the development process by discovering tasks, gaining additional time control, and arranging the required combinations. The statement is an important source of agile standards (Batarseh & Gonzalez, 2018).

Scrum Agile principles that are related are :

1) Welcome to change requirements, Even if we are in an advanced stage of development. Agile technologies take advantage of the change that serves the customer's, competitive advantage.
2) Supply operating software regularly, from multiple weeks to more than one months, with a choice to the shorter timescale.
3) operating software application is the number one degree of development.
4) At ordinary intervals, the staff will be reflected in how they appear with more force, to adjust their behaviour and adjust them accordingly.

- Why Scrum Agile?

The following reasons show why the Scrum Agile was selected:

1) Scrum agile is a flexible and popular method.
2) Scrum Agile is context-based - which can be done without modifying and modifying problems so that a mainly context-based experience can be added

3) The scrum agile life cycle model is the most used, as illustrated in Figure (1). These days by software corporations, and recently it has proved to obtain good reputation, attraction and success (Batarseh & Gonzalez, 2018).

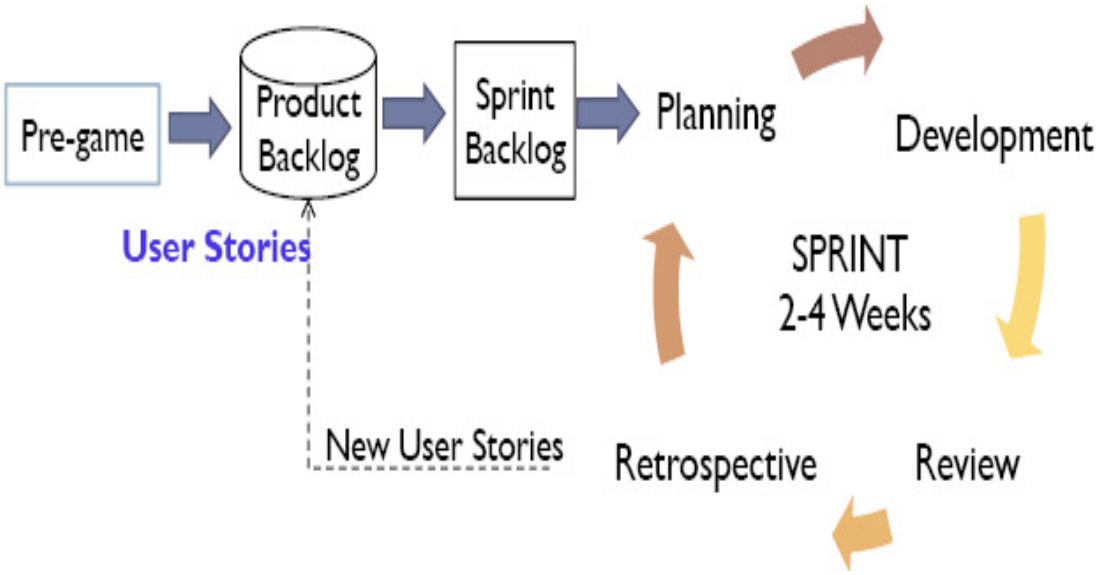


Figure 1 Scrum Agile life cycle (Schwaber, 2004) (Schwaber, 2004)

Assigning and forecasting errors is the most time-consuming activity. It is highly recommended to have confidence, Particularly in instances of software program that fail at the project's approved time, mistaken confidence is likely to cause serious issues such as financial losses and software failures.. preventive measures can be applied by the Scrum master to avoid further errors. It is not easy for this approach to provide preventive measure in the context of the system, but it also leads the team to anticipate errors before they take place, and even make in time changes to avoid them completely.

The importance of predicting failure before it occurs is to minimize the time and cost of the program implementation (Batarseh & Gonzalez, 2018).

What Is the Point in Using SVMs as a Classification Technique?

All type strategies have advantages and disadvantages, which can be greater or much less crucial consistent with the facts which are being analysed, and for this reason have a relative relevance. SVMs may be a useful device for Insolvency analysis, in the case of non-regularity within the statistics, as an instance when the facts are not often Distributed or have an unknown distribution. It could help examine facts, financial ratios which Should be converted prior to entering the rating of classical type techniques. The advantages of The SVM method can be summarised as follows :

1- By using introducing the kernel, SVMs gain flexibility inside the choice of the form of the brink setting apart Solvent from bancrupt businesses, which wishes no longer be linear and even needs no longer have the same practical shape for all statistics, due to the fact that its characteristic is non-parametric and operates domestically. Hence they Can work with financial ratios, which show a non-monotone relation to the score and to the probability Of default, or which might be non-linearly established, and this without needing any unique paintings on each Non-monotone variable.
2- Because the kernel implicitly incorporates a non-linear transformation, no assumptions approximately the practical Form of the transformation, which makes records linearly separable, is essential. The transformation happens implicitly on a robust theoretical foundation and human expertise judgement beforehand isn't wanted.
3- SVMs offer an awesome out-of-pattern generalization, if the parameters C and r (in the case of a Gaussian Kernel) are appropriately chosen. Which means, by way of selecting the proper generalization grade,SVMs can be robust, even if the schooling sample has a few bias.

4- SVMs deliver a unique solution, for the reason that optimality trouble is convex. This is an advantage in comparison to Neural Networks, that have more than one answers related to local minima and for that reason can also not be strong over different samples.

5- With the selection of the right kernel, along with the Gaussian kernel, you will placed greater stress on the similarity between corporations, due to the fact the more similar the monetary structure of corporations is, the better is the cost of the kernel. For this reason whilst classifying a new organisation, the values of its economic ratios are in comparison with the ones of the aid vectors of the education pattern which can be extra similar to this new company. This business enterprise is then categorised in line with with which organization it has the greatest similarity.

A not unusual disadvantage of non-parametric strategies together with SVMs is the dearth of transparency of consequences. SVMs can not represent the rating of all organizations as a simple parametric function of the financial

ratios of the monetary ratios, on the grounds that its measurement may be very excessive. It's miles neither a linear combination of single economic ratios nor

Has it another easy useful shape. The weights of the monetary ratios are not steady. Accordingly the marginal contribution of every financial ratio to the score is variable. The usage of a Gaussian kernel each company has its very own weights in step with the difference among the fee of their very own financial ratios and those of the guide vectors of the training records pattern. (Auria, 2008)

In this work, the importance of predicting failure in agile software development was discussed, and methods were developed to anticipate project failures through the use of analytical and statistical techniques. Research Aim and Objectives:

This research aims to predict the failures in scrum agile in advanced to serve the agile principles.

1) Reading related works and highlighting the approaches, methods, and techniques that are related to the failure prediction.
2) Selecting a research dataset (Multi Projects) that is related to agile scrum.
3) Using the correlation coefficient to measure the impact of sprint data to each other.
4) Using SVM. to keep the system learning and enhance the prediction of failures.
5) Comparing the result with previous work to measure our approach against similar work.

1.4. Motivations

Agile development relies upon at the context of the project, checking out is finished otherwise in each sprint data on each other. Furthermore, because of the absence of a detailed project plan, the prediction of the failure is very difficult. Once the failure happens in any stage of the scrum phases, the overcome of the failure will be a big challenge. In this work, we propose a technique to predict critical failure factors that assess the fulfilment or failure of tasks correctly using correlation coefficient analysis and support vector machine For figuring out critical failure factors to avoid project failure, as a result saving time and money.

1.5. The Contribution

Our main contribution here was achieved by implementing SVM. on two sets of data and identifying failure factors affecting them. The SVM. algorithm has proven highly effective in predicting failure cases before they happen, compared to other algorithms, which helps analysts and engineers work in a faster way and save time and effort.

1.6. Research Approach

Our approach includes the following steps:

1- Data pre-processing, which It is a data mining approach that restructures primary data into the concept and structured data. Preprocessing is the proven way to solve issues facing "Real-world data" because these data usually contain incomplete / or inconsistent data or lack any specific directions or behaviours and are expected to contain many errors

2- Identifying dependent and independent variables using correlation coefficient (CC).

3- Defining the dependent and independent variables the usage of the correlation coefficient. An independent variable is a variable that is changed or managed in a scientific test to check the results on the dependent variable. The dependent variable being measured and examined in a scientific experiment.

4- Analysing the results from the CC operation to greater or less than 0.5; in case it is greater, we change the concept of failure factors or emphasize the existing failure factors.

5- After confirming the presence of failure factors, SVM. is applied to determine the factors leading to failure by separating the data into a training group and a test group. Maximum of the records are used for training and using the usage of a

smaller part of the test in this process is to reduce the contrast between the data and higher recognize its characteristics.

6- At the end of the approach, critical factors of failure for projects will be listed.

1.7. Thesis Organization

This thesis is organized as follows:

- Chapter 1 is the introduction in which we are introducing our work.
- Chapter 2 includes a set of previous work related to the topic.
- Chapter 3 explains the steps that we have implemented in our proposed methodology.
- Chapter 4 includes a presentation of the results that were reached and analyzed.
- Chapter 5 presents conclusions and recommendations for future work.

CHAPTER TWO
BACKGROUND AND RELATED WORK

CHAPTER 2:

BACKGROUND AND RELATED WORK

3.1.Overview

Given the importance of being able to predict failures before they occur in terms of saving effort and time, many suggested methods that are highly reliable in forecasting have been discussed in this chapter. Main issues related to this work are highlighted in term of increasing the awareness of the importance of the research area. Furthermore, this research has covered some related work that had a role in the prediction process in several different areas.

2.1.Background

Programming testing has generally been separated into approval and check (V&V). Approval is the way toward guaranteeing that the product coordinates the prerequisites, and check guarantees that the framework meets the useful determination building one of the most well-known definitions of approval is building the right frame, and checking is building the right frame.

Programming building is loaded with vulnerability: A significant wellspring of issues during programming advancement is a vulnerability about necessities; a significant wellspring of issues during programming development is a vulnerability about plan and execution choices made during improvement (Barstow, 1988). Programming is one significant job which is driving the numerous electronic and business items. The

advancement of these items makes an expansion in the requirement for the product. Testing is one important period of a product improvement life cycle.

Scanning errors are performed in programming tools. Reliability is an important aspect of distinguishing characterizes quality. Programming unwavering quality models are numerical models that portray the sensible marvel of programming testing during the product improvement life cycle. These models are installed with flaw location, adjustment and shortcoming presentation. Numerous papers are proposed in writing right now (Rafi, 2012). Checking out should be done at all levels of the framework development system.

There are various techniques for checking out that might be applied for various purposes. Professionals have investigated making use of customary programming approval models into AI frameworks.

At the point when prescient approval is utilized, the outcomes are spared, and for the following cycle, a similar arrangement of tests is performed and contrasted with the past outcomes. Prescient approval can't be utilized in detachment; it is just a way to deal with and think about test outcomes, and assess the improvement procedure (Batarseh & Gonzalez, 2018).

More recent methods applied:

1. Particle swarm optimization-based artificial neural networks (PSO-ANN)
2. Artificial immune recognition systems (AIRS)
3. Gene expression programming (GEP)
4. Genetic programming (GP)
5. Multiple regressions (MR) (Batarseh & Gonzalez, 2018).

2.2. Agile software development (ASD)

The software engineering in 2001, depending a lot on four fundamental beliefs and twelve standards, spread out in the “Agile Manifesto” Agility, or the capacity to quickly adjust to unpredictable prerequisites is a foundation of ASD. This stands as a conspicuous difference to the plan-driven technique recommended by traditional models of software improvement, such as the cascade model. Other recognizing highlights of ASD encompass member-upgraded centres for human's _ social parts of programming designing; expanded coordinated effort between business customers and programming businesses; and a solid accentuation on visit conveyance of business esteem.

Since its beginning around two decades prior, ASD has quickly become a standard programming advancement model being used today through the mechanical reception of a few of its solid indications, for example, Scrum and Extreme Programming. A comparative emotional effect has been seen in the exploration network with productions of plentiful quantities of essential research concentrates on ASD points. There has likewise been an impressive number of auxiliary investigations as writing surveys and mappings distributed inside this space over the previous decade or something similar to that (Hoda, 2017).

Agile Methodologies are considered a gathering of software improvement methods which depend on incremental and iterative improvement. There are significant attributes that are essential to all those methodologies: adaptive planning, events development that is evolutionary and iterative, As well as rapid reactions and flexible ones to improve communication. Complying with the standards of "Light but sufficient" and emphasizing on the notion of humans-orientated and communique-centred is the primary accentuation. Being labelled as a lightweight procedure, it is progressively appropriate for the improvement of little activities. Agile software program development takes the view that

creation businesses want to start with simplified approximations which might be predictable to the final requirement and then keep on increasing these requirements information at some stage in the lifestyles of the turn of events (Aggarwal, 2020).

Agile methodologies are used to put in force numerous obligations which include appearing better programming in a shorter period, purchaser coordinated effort, self-organizing teams, documentation, saving time in and advertising and marketing for investing it in other regions. It's far clean that Agile method carries a group of lightweight methods that consist of Scrum, Crystal clean, extreme Programming (XP), feature-driven improvement (FDD), and Dynamic systems improvement method (DSDM) Crystal, Adaptive software program improvement (ASD), Lean software program improvement etc. (Aggarwal, 2020).

Scrum is an agile procedure ordinarily utilized for the development of a product, software development in particular. Popular-purpose project control framework that is the fabric to any amplify with forceful cutoff times with state-of-the-art requirements and a level of being precise (Aggarwal, 2020).

Agile benefits in the method of Software Development:

1. The important thing advantages of agile technique in the approaches of software development because of which agile method should be acquired even as developing..*Dealing with change requirements*

The plan development section is very advanced. to deal with change requirements

The planning process is improved. First of all, because clients are legally linked to making progress, meaning clients control the actions or activities on the site, the requirements to reflect the final needs.

2. *Fault Detection*

As trying out is applied throughout each stage, faults are diagnosed earlier and can be constant earlier than inflicting results with a plan-pushed process model. Additionally, such continuous testing helps in getting required remarks, which further improves the code introduction method in future emphases.

3. *Extended performance*

Ongoing meetings provide a great opportunity to exchange big data and continually modify improvements. Teamwork is surprisingly influenced by simplifying and discussing complex projects with simple tales and designs. Moreover, continuous communication improves data exchange, self-arranged groups and intimate friendship as employees begin to build an agreement with their colleagues and establish confidence in them. This expands the efficiency of the team and leads to a better implementation of a greater return on investment than the individual

4. *Incremental and Iterative Delivery*

To control risk and to get early feedback from clients and final customers, delivery of the project is allocated into little functional releases or increments. These little things are inserted into a thoughtful schedule to use iterations that are normally kept between one and four weeks at a time. Plans, requirements, planning, symbol and tests are initially established and updated gradually as needed to adapt to mission adjustments. As a final result, the development and monitoring of program functions can often be closely monitored and monitored over their preference upon completion of phases or long periods.

5. *Flexibility of Design*

Flexibility defines the capacity to trade headings rapidly. The main feature of Agile Technology is alternative handling in imperatives, the layout must be made flexible which could cope with changes effortlessly. Flexibility is based on the development manner used for the tasks.

6. Improvement in Quality

Paid development and restructuring is achieved through inspection engines. Restructuring results in better reuse of code and better quality. All components of this system have been optimized, from design and build to the performance of every racing product. Constant communication increases the speed of the error blocking response (Aggarwal, 2020).

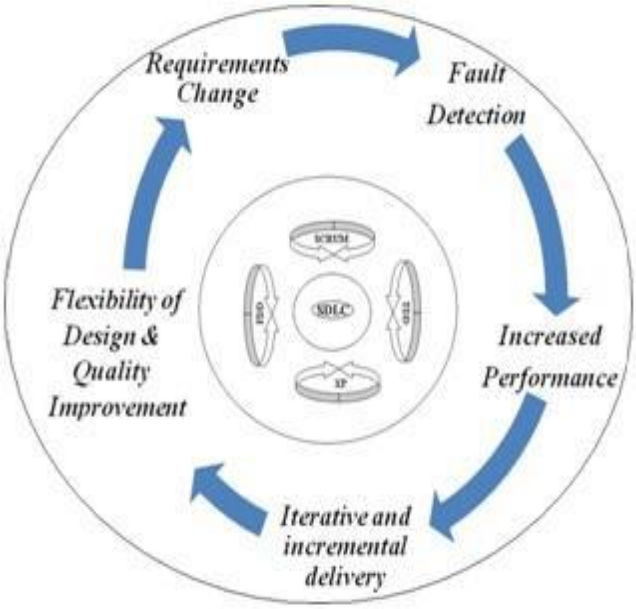


Figure 2 Agile Software Development Methodologies with Benefits (Kumar, 2012)

2.3.SCRUM Model for Agile Methodology

In Scrum Agile, projects progress employing a progression of iterations known sprints. Each sprint is ordinarily two-four weeks lengthy. An average scrum group has somewhere

in the range of five and nine people. However, Scrum Agile undertakings certainly undoubtedly scale into the hundreds.

In Scrum Agile, projects progress using a progression of emphases called runs. Each run is normally two months long. An average scrum group has somewhere in the range of 5 and 9 individuals. However, Scrum projects can undoubtedly develop into hundreds. The organization does not currently consist of any of the roles of traditional software engineering, along with the programmer, laboratory or architect. The product owner is the primary stakeholder of the project and represents users, customers and others in the tool. Scrum Master is responsible for making sure the institution is profitable and should be expected in moderation (Aggarwal, 2020).

Scrum is essentially an agile, lightweight structure that gives steps to manipulate and control the software and product improvement procedure.

It is the mix of the Incremental model and the Iterative model because the constructs are progressive and gradual as far as the highlights develop an object-oriented program.

Scrum Agile became intended to speed up advancement, adjust person and associations' maxims, characterize a culture concentrating on execution, bolster investor esteem creation, have the great correspondence of execution at all levels, and enhance singular turn of events and nature of life (Srivastava, 2017).

It is a structure of agile methodology, which gives adaptability to control and manage the requirements such as the improvement of programming. It is an iterative and gradual base model which constructs programming and characterizes system, like the one a module of programming can create in little chunks in an iterative manner (Hayat, 2019).

Scrum became meant to build the produce-capacity of development manner, alter character and associations dictums, characterize a culture targeting presentation, guide

investor esteem improvement, to have an awesome message of execution in any respect degrees, and improve wonderful improvement and class of existence. Scrum is such an adaptable version which may be followed to any amplify of any form of an enterprise (Hayat, 2019).

Scrum has an effective impact on the records territories of software program project management. Scrum has a high-quality impact on the time, price, scope, first-class, threat and scope of the task. Some association middle around the aim orientated recruiting of representatives and a few aren't, so it has a tremendous effect of a scrum on H.R the executives additionally that scrum reduces hazard, manage the fee, and evolved best product helps in well-timed of completion of the mission (Hayat, 2019).

Figure 3 Phases of project management lifecycle

2.4.Support Vector Machine

Support Vector Machines algorithm was presented by Vladimir Vapnik in 1995 (Jain, 2020). SVM. is a directed AI method that is utilized for grouping just as relapse. It endeavours to sort information by finding reasonable hyperplanes that can separate information by the most noteworthy edge. Because of the preparation sets, the new qualities are segregated and analyzed.

SVM. is a basic and effective classifying algorithm which is utilized for classification and an example of acknowledgement. The fundamental point of this algorithm is to obtain the capacity that develops hyperplanes or limits. These hyperplanes are utilized to isolate various classifications of input information focuses. SVM. utilizes binary classification (Jain, 2020) (Huang, 2018).

SVMs. are frameworks that utilize hyperplanes in highlight space of high measurements to separate values dependent on a specific determination. Hyperplanes are prepared with specific algorithms to utilize factual learning. SVM. method of classification is a supervised learning that involves the feature extraction and creates attractive outputs.

The advantage of SVM. is that it is very smooth to train. It can scale high-dimensional information better than neural networks. There are commonly two sorts of SVM. classifiers: Linear and Nonlinear (Jain, 2020). SVM. has been utilized in many research examines that depended on this technique for classifying four fundamental sorts, namely, severity, happiness, anger, and impartiality talk (Jain, 2020).

SVM. was used to classify genomic cancer, which helped to discover new biomarkers, drug targets, and an understanding of the genes that cause cancer. The artificial intelligence of SVM. helped to identify genomic patterns or features that might represent

subtypes of cancer, to predict results, to predict medicinal benefits, and to generate a tumour or a biological process specific to the tumour (Huang, 2018).

2.5. Correlation Coefficients

Correlation is characterized as a connection that exists between phenomena or objects, or between scientific or real elements that generally change, are related, or occur together with an unexpected method (Akoglu, 2018).

The relationship between the two variables is measured by a number, which is between -1 and +1. Zero, indicates no correlation and 1 indicates complete correlation. The correlation strength increases from (0 to +1), and from (0 to -1) (Akoglu, 2018).

Attention should be drawn to the fact that the interpretation of correlation coefficients varies greatly between areas of scientific research. There are no absolute rules for interpreting the power of attachment (Akoglu, 2018). The correlation coefficient was applied to find the relationship between apparent diffusion coefficient (ASD), specifically ADC mean and cellularity in different tumours, and the results that had a significant impact in this area (Surov).

A correlation coefficient is a useful tool in decision-making. Three correlation coefficients were proposed between linguistic neutrosophic numbers (LNNs). Accordingly, a decision-making method was developed for multi-attribute group decision-making (MAGDM), multi-traits based on correlation coefficients decision-making (MAGDM), and multi-traits based on correlation coefficients in (LNN). The application and effectiveness of the correlation coefficient method were presented (Shi, 2018).

The use of the correlation of the relationships in one of the largest economies in the world was verified and the standard of conversion was the US dollar. A negative relationship is

normal, which implies an expansion of access on the advantage that it will lead to a decrease in the access of the other. Standalone use and link confirmation animated normalization of embedded correlation analysis and correlation coefficients. We analyzed this opportunity dissecting behaviour according to distinct time scales. The main results appeared in European markets; the conversion standard has little effect. This exceptional impact occurs only because of the Indian Stock Exchange, while at the expense of the Japanese market, the relationship is certain (Ferreira, 2019).

2.6.Semi parametric Regression

Semiparametric regression can be a significant incentive in the arrangement of complex logical issues. This present a reality which is excessively confounded for the human brain to fathom in detail. Semiparametric regression models diminish complex informational collections to rundowns that we can understand. Appropriately applied, they hold basic highlights of the information while disposing of irrelevant subtleties, and hence they aid sound decision-making (Ruppert, 2003).

More than one linear regression model and many experimental design models are unique examples of a linear regression model. Regression is a look at the response variable Y by looking at the vector p x 1 of the predictions x. In a linear regression version, $Y = \beta T x + e$, and Y are not biased to x, given one linear formula $\beta T x$ (Olive, 2017).

Linear regression assumptions are illustrated using simulated statistics and an empirical instance at the relation between the time given that type 2 diabetes and glycated haemoglobin (HbA1c) prognosis. Simulation consequences had been evaluated on insurance; e.g, the number of instances the 95% self-assurance c program language period covered the genuine slope coefficient (Schmidt, 2018).

2.7. Failures Prediction In Agile Software Development

The main goal of forecasting failures is to reduce the impact and costs of the defect in software development projects by eliminating the defect in the development phase. In this paper, a method for predicting failure through a Bayesian model based on genes has been suggested to improve the true positive rate (Sangeetha, 2017).

The research can be conducted using a higher real positive rate with the expectation of errors in the program with the help of the gene model. Techniques found in the gene model discussed in the research included:

1- Direct Monitoring Techniques in Software Systems,

2- Fuzzy Rule-based Algorithm for estimation,

3- Morphology Model for Event-Based Testing (Sangeetha, 2017).

Given the importance of agile software development in software projects, work continues proposing methodologies to predict the failure of agile software projects. This paper presented a questionnaire on machine learning methodologies and their role in predicting the failure and success of agile software projects and a summary of conclusions. The paper reviewed machine learning such as logistic regression, linear regression, neural network, and fuzzy logic (Abdelaziz, 2017).

There are different types of errors that lead to software failures and affect the quality of the software, such as communication errors, runtime errors, and other errors, so the prediction of errors in the early stages of the program is to develop high-quality and cost-effective programs.

In the prediction process, the error-prone units are separated from the non-error-prone units.

In this paper, a methodology for predicting software errors using SVM. with a cross-sectional verification procedure, K-Fold was suggested and the results demonstrated that the proposed model was the best performing in terms of accuracy of classification to classify program units into units that may expose programs to fail or programs not to fail (Raju, 2018).

2.8.Related Works

Anderson (2015) The proposed type essentially fully examines failure forecasting techniques and provides a case study using a business tool. The results showed that rating models that mainly rely on attributes may be better used to understand the environment in which the program is superior and to discover deficiencies that may cause more value in this environment. Moreover, the consequences indicated that traditionally relied on measures, along with deactivation, might not be the first-order predictors of a failed test condition. Additionally, the results indicated that the use of additional characteristic model traits should enhance the ability to anticipate future failures.

The success of the device's use of automated learning algorithms in forecasting applications is presented in Chigurupati (2016). However, there was one drawback that the model got, which was not being trained in units that did not experience any failure.

The work of Li (2017) focused on predicting potential programming code flaws in software implementation. A prediction framework called DP-CNN (Predictive Defect over the Convolutional Neural Network) used a generation of source vacancy with semantics and structural involvement.

Preserving information with the use of word inclusion and combining the features with traditionally handcrafted attributes to further improve failure prediction, CNN's results highlighted that the proposed method improved the modern method by 12%. Predicting

errors to investigate risks is a critical task in software engineering. Therefore, an in-depth approach to learning technique has been applied to the release of Automatic Encryption Program (VAE) to predict error by obtaining (VAE) to generate new failure samples A good way to enhance predictability (Sun, 2018). In some different problems, software application engineers have continuously strived and tested to achieve the best software program. In the program verification program, approval and confirmation are the most fundamental pillars of high-quality programming and evaluation. The ability to detect errors and disasters that are already in the program is valuable, but what is more useful is the ability to anticipate cases of software failure (and their locations) sooner than they arise, especially if this is implemented with a high statistical self-assurance. ADT predictive analyzes are used to perform the reduction of future failures/errors. Several distinct styles within the literature are delivered to minimize errors after they are truly discovered. However, in this paper, a proactive approach to predict program failures was presented and evaluated. Meantime between failures (MTBF) and regression evaluation are used to predict future errors and their locations in a completely enemy-dependent life cycle (scrum) (Batarseh F. A., 2018). Effective and highly predictive methods of machine learning have been found by using some algorithms to enhance the accuracy of failure prediction. Researchers have developed a prediction system using time and instrument study. Which of these algorithms is SVM? A set of rules that were tested effectively by predicting 90% compared to different algorithms (Mohammed, 2019).

For more focusing on most related works to our proposed approach, Table 1 shows a summary of previous and related work.

Table 1: Related Works Summary

Author-Year	**Approach**	**Results**
Anderson, J. a. (2015). Striving for failure: an industrial case study about test failure prediction. In 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (pp. 49--58). IEEE.	The use of an industrial software	The results indicated that the use of additional capabilities of classification models can improve the ability to anticipate malfunctions in the event of a future test. The results show that fully feature-based species models can be used to increase knowledge The surrounding environment where the program is developed and the selection of deficiencies in this surrounding environment Extra value
Chigurupati, A. a. (2016). Predicting hardware failure using machine learning. In 2016 Annual Reliability and Maintainability Symposium (RAMS) (pp. 1--6). IEEE.	Using automated learning algorithms in applications	This paper illustrated the fulfilment of the tool's use of the device getting to know algorithms in forecasting programs. However, there has been one downside that the version got: now not being educated in units that did not experience any failure.
Li, J. a. (2017). Software defect prediction via a convolutional neural network. In 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS (pp. 318--328). IEEE.	Prediction framework referred to as DP-CNN(disease Prediction via Convolutional	DP-CNN improves completely contemporary DBN-based methods - which mainly rely on traditional methods by 12% and 16%, respectively, in terms of F

	Neural community))	measurement in predicting failures
Sun, Y. a. (2018). Utilizing deep architecture networks of VAE in software fault prediction. In 2018 IEEE Intl Conf on Parallel \& Distributed Processing with Applications, Ubiquitous Computing \& Communications, Big Data \& Cloud Computing, Social Computing \& Networking, Sustainable Computing \& Communications (ISPA/IUCC/BDCloud/SocialCom/SustainC (pp. {870--877}). IEEE.	Adopt an in-depth learning approach to Variationa Autoencoder (VAE) to predict program faults.	The ability to take advantage of VAE to generate new failure samples so that the user can improve the prediction of program errors
Batarseh, F. A. (2018). Predicting failures in agile software development through data analytics. Software Quality Journal(Springer), 49--66.	ADT predictive analyzes are used to identify future failures /Reducing errors. ADT MTBF and regression evaluation are used to predict future errors and their location in the scrum-dependent life cycle.	Using ADT had an ordinary statistical rate of 93.5 %
Mohammed, B. a. (2019). Failure prediction using machine learning in a virtualised HPC system and application. Cluster Computing, 22(Springer), 471--485.	Support vector machine (SVM.), regression trees (CART), random forest (RF), and, and k-nearest neighbours (KNN), classification linear discriminant analysis (LDA).	SVM. In the expectation of failure, 90% are valid and effective, compared to other algorithms

.

CHAPTER THREE
METHODOLOGY

CHAPTER 3:

METHODOLOGY

3.1. Overview

This chapter comes in five parts: the first part about investigates the description of the data sets that the methodology has implemented. The second chapter explains the pre-processing process that took place on the data sets. The third part tackles the dependent and independent variables. The fourth part presents the obtained results, value analysis and classification. In the fifth part, critical values that express the rate of failure, and whether they are influencing, or not, according to the criteria previously identified are determined.

3.2. Data Set Description

First Dataset

The dataset is selected based on a previous study (Batarseh & Gonzalez, 2018). A test was performed on an example AI system; the system has six essential modules:

1- Human-Computer Interaction (HCI) Module: This unit contains the basic intelligence and instinct that interact with the movements of individuals (mouse clicks in this example). The unit monitors human intervention with the device and uses the correct judgment included to respond to that

2- Intelligent Module 1: This is the primary of 3 clever gadgets, and it consists of logic that revolves around conclusions, questioning, and problem-solving. It prepares statistics for besides processing within the 2nd and third smart units.

3- Intelligent Module 2: This is the second unit (out of three) that consists of algorithms that can be collected from machine learning. This unit contains built-in algorithms that include aggregation, unusual inductive experience, knowledge acquisition and class

4- Intelligent Module 3: This unit contains an icon representing advanced AI technologies. The algorithms in this unit include "herbal language processing (NLP), genetic programming, and neural networks".

5- Knowledgebase: The knowledge base contains a set of specific principles that represent the information included in the device. This unit may be very vital because it represents "the logic of the middle system, intelligence and knowledge of the system".

6- User Interface: This unit contains all of the GUI plugins, for example, designs, images, tools on the side of the button, dropdown menus ... etc. A good UI validation effort is required (Batarseh & Gonzalez, 2018).

The historical failure data that has been collected as follows:

1- Definition for each row, common database practice (a single-row document is assigned that matches the definition)(ID).

2- "Software Module" (all modules are described in Sect. 3.2)

ID	Software Module	Sprint Number	TestType	Sprint Start Day	Sprint Failure Day	Total Time	Run Time in Hours	MTBF@ 60% Confidence	Failure Forecast
1	Intelligence Module 1	1	FunctionalityTest	Day 1	Day2	1	8	4	?
2	Intelligence Module 1	1	PerformanceTest	Day 1	Day2	1	8	4	?
3	Intelligence Module 1	1	StressTest	Day 1	Day2	1	8	4	?
4	Intelligence Module 1	1	GuiTest	Day 1	Day2	1	8	4	?
5	Knowledge Base	1	FunctionalityTest	Day 1	Day2	1	8	4	?
6	Knowledge Base	1	PerformanceTest	Day 1	Day2	1	8	4	?
7	Knowledge Base	2	StressTest	Day 3	Day 4	1	8	4	?
8	Knowledge Base	2	GuiTest	Day 3	Day 4	1	8	4	?
9	User Interface	2	FunctionalityTest	Day 3	Day 4	1	8	4	?
10	User Interface	2	PerformanceTest	Day 3	Day 4	1	8	4	?
11	User Interface	2	StressTest	Day 3	Day 4	1	8	4	?
12	User Interface	2	GuiTest	Day 3	Day 4	1	8	4	?
13	Human-Computer Int	2	FunctionalityTest	Day 3	Day 4	1	8	4	?
14	Human-Computer Int	2	PerformanceTest	Day 3	Day 4	1	8	4	?
15	Human-Computer Int	2	StressTest	Day 3	Day 4	1	8	4	?

Figure 4 "Initial dataset (data collected from the scrum development lifecycle") (Batarseh & Gonzalez, 2018)

3- Sprint number: As formerly referred to, scrum testing is primarily based on sprints, and so recording the sprint number is vital. It's far used as an entry to the forecasting version.

4- Time dimension, Sprint starts and day of failure. These proven values are used to measure the time of the race in which the failure occurred earlier.

5- General time in days, hours, and MTBF (inputs to the regression version as well).

"The dataset consists of 181 rows; the total difference for all rows is 360, with an average of MTBF = 114 min. Average testing runtime in hours for the system that was used for experimentation is 1770 min (29.5 h)" (Batarseh & Gonzalez, 2018).

Software module	Sprint failure day	Sprint number	Test type	Sprint start day	Run Time In Hours	Total time	Mtbf @60% Confidence	MTBF Prediction	Difference	Mtbf @60% Confidence Predictor
Human-Computer Interaction Module	Day 19	5	FunctionalityTest	Day 14	40	5	20	20	0	99.942%
			GuiTest	Day 14	40	5	20	20	0	99.942%
			PerformanceTest	Day 14	40	5	20	20	0	99.942%
			StressTest	Day 14	40	5	20	20	0	99.942%
	Day 4	2	FunctionalityTest	Day 3	8	1	4	4	0	99.942%
			PerformanceTest	Day 3	8	1	4	4	0	99.942%
			StressTest	Day 3	8	1	4	4	0	99.942%
	Day 49	13	FunctionalityTest	Day 45	32	4	16	15	(1)	99.942%
	Day 50	14	GuiTest	Day 50	0	0	0	0	0	99.942%
			PerformanceTest	Day 50	0	0	0	0	0	99.942%
			StressTest	Day 50	0	0	0	0	0	99.942%
	Day 60	19	FunctionalityTest	Day 60	16	2	8	8	0	99.942%
	Day 67	20	FunctionalityTest	Day 61	48	6	24	24	0	99.942%
			GuiTest	Day 61	48	6	24	48	24	99.942%
				Day 61	48	6	24	48	24	99.942%
			PerformanceTest	Day 61	48	6	24	48	24	99.942%
				Day 61	48	6	24	48	24	99.942%
			StressTest	Day 61	48	6	24	48	24	99.942%
				Day 61	48	6	24	48	24	99.942%
	Day 7	3	GuiTest	Day 5	16	2	8	8	0	99.942%
	Day 77	25	FunctionalityTest	Day 70	56	7	28	28	0	99.942%

Figure 5 MTBF forecasting results (Batarseh & Gonzalez, 2018)

Second Dataset

The ISBSG release (IDRI, 2012) Examining the performance of the proposed models is the reason for employing dataset. In step with Jorgensen and Sheppard (Jorgensen, 2006), utilizing real-existence reliable projects in SEE results in growing the credibility of the study. 1e dataset contains more than 5,000 commercial projects written using various programming languages and additionally evolved using various software improvement existence cycles.

Projects are categorized as new or improved development. Moreover, the size of enterprise software has become measured through functional points for using global standards including "IFPUG, COSMIC, etc.". 1, therefore, to make the research homogeneous, tasks with modified job factors of IFPUG were better taken into account. Dataset 1e consists of more than 100 attributes for each project as well as tools such as:

"project size, project start and end date, and many other features. Additionally, ISBSG classifies mission statistics from first to 4^{th} levels," A "to" D " Where “A” proposes projects with the best classification first, followed by “B” and so on.

After examining the dataset, it was noticed that despite some projects were similar in size of software, effort varied extensively.

The productivity ratio is the ratio among software program effort (output) and software program size (the primary input). It was noticed that a giant difference inside the productivity ratio among projects with similar software size. For instance, productivity (effort/size) various from 0.2 to 300 for the same adjusted feature factor (AFP). The big variant came in productivity.

The dataset is affected by the ratio, meaning that the ratio makes the dataset heterogeneous. It became true that applying the equal version of full projects became impractical. To solve this problem, projects were grouped according to productivity into more homogeneous data sets.

The number one dataset became divided into sub-datasets, in which projects in every sub dataset had the simplest small variations in productivity [50]. For this research, the dataset becomes divided into the following 3 datasets as follows:

i. "Dataset 1: small productivity ratio (P), where $0.2 \leq P < 10$".
ii. "Dataset 2: medium productivity projects, where $10 \leq P < 20$ ".
iii. "Dataset 3: high productivity ($P \geq 20$)".

Additionally, to compare the effect of mixing projects with diverse productivity together, a fourth dataset was added. This additional data set specifically mixes the three data sets. In addition to converting the third dataset to visual as heterogeneous compared to the first number, due to the productivity in this dataset, which ranges between 20 and 330.

The dataset has become used to look at the effect of heterogeneous statistics on the overall performance of inconspicuous models of common sense, due to the multiplicity of datasets of the ISBSG set described above, difficult and fast indicators for project selection have turned into the ability to arrange the dataset. The characteristics selected for the analysis were as follows :

.

i. "AFP: adjusted function points, which indicates software size";
ii. Development type: This suggests whether or now not the undertaking is a new improvement, enhancement, or redevelopment;
iii. Team size: This represents the type of participants in every improvement team
iv. Useful resource degree: This identifies the company that has changed to an interest in developing these challenges, along with the development team's attempt, development assistance, laptop operating support, and termination of users or clients.
v. Software effort: that is an effort in individual-hours in software strive estimation, it is vital to pick non-beneficial requirements as unbiased variables, in addition to user requirements. [51].

All the above functions are non-stop variables except beneficial resource degree it is truly express. The original dataset contained 5052Projects. .

The subsequent guidelines were used to clear out the datasets. Additionally, projects had been decided on primarily based on the following:

1) Data quality: Easier projects with data quality A and B as illustrated using ISBSG identified, reducing the size of the dataset to 4,747 projects;
2) Software program size in function points;

3) Four inputs:" resource level, development type, team size, and AFP; and one output variable: software effort";
4) New development projects only: Projects that were considered development improvements or redevelopment or different types were taken into consideration, which made the total of the entire projects, 1,805.
5) Missing information: In this step the data set was filtered by deleting all the rows containing missing data, leaving 468 very specific projects
6) Divide data sets according to their productivity, as mentioned previously, to create three great data sets and one mixed group.
7) Dividing each dataset into datasets for the training experiment is randomly divided into 70 /% and 30%, where 70% of each dataset is used for education and 30% for the experiment.

After implementing plans 6 and 7, the data sets were as follows:

a) The first dataset, with a productivity of $0.2 \leq P < 10$, comprises a complete set of 245 initiatives that can be customized in 172 training projects and 73 test projects.;
b) The second data set, with productivity $10 \leq P < 20$, consists of 116 projects divided into 81 training projects and 35 test projects
c) The third dataset, with a yield of better than or identical to 20 ($P \geq 20$), was made up of 107 projects with 32 test projects and 75 training projects.
d) The fourth dataset, with the combination of projects from the three datasets, consisted of 468 projects with forty trial missions and 328 training projects (Nassif, 2019).

G	F	E	D	C	B	A
MTBF	NUMBER OF FAILUER	Normalise	Resource	Adjusted F	Language	Developme
22828.67805	0.498145358	11372	1	859	4	1
177862.8011	0.497883759	88555	1	1306	3	1
20487.96146	0.497853338	10200	1	465	3	1
16938.80485	0.506529243	8580	1	359	4	1
5988.704415	0.506620429	3034	2	199	3	1
3662.929759	0.506698223	1856	3	225	3	1
34738.76261	0.506638656	17600	1	4272	4	1
21636.17467	0.506559046	10960	3	599	3	1
3448.063445	0.506371193	1746	1	357	4	1
17886.03029	0.506428752	9058	1	599	3	1
780.0484046	0.506378832	395	1	331	3	1
2505.912359	0.506402387	1269	3	212	3	1
7769.491595	0.506339437	3934	4	194	3	1
7551.642827	0.50624746	3823	4	185	6	1
7404.037252	0.506210311	3748	1	537	3	1
4168.560917	0.506169885	2110	1	426	4	1
7641.808747	0.50616289	3868	2	484	3	1
1876.981154	0.50613188	950	1	610	4	1
1234.894247	0.506116213	625	1	556	3	1
11755.13025	0.50616198	5950	1	170	3	1
7595.973515	0.506189232	3845	1	778	3	1

Figure 6 The results of MTBF process

In the second data set, work was initially made to generate a new feature for the data that expresses the Number of Failures so that we can apply the MTBF equation and reach results that are classified within a new column of the Number of failures reached through a statistical equation. This is presented in Figure (6).

Measuring the average time between failures (MTBF) is a common practice in the industry. In many certification strategies, MTBF is described as the time elapsed between device failure and the time when the device started working again. This dimension is achieved through calculation: run time ÷ number of failures.

Then, the values of the MTBF were obtained, as shown in Figure (6). On these values, we applied the remaining steps that were mentioned previously.

3.3.Proposed Approach Phases

Phase One: Data set Pre-processing:

The first step to identify failures is the pre-processing of data, which is a manipulation method. The manipulation method includes a redesign of heterogeneous data in conceptual planning. Global data is often incomplete or inconsistent and/or lacks positive behaviours or developments, and is likely to include many errors. Data pre-processing is a well-established approach to solve such problems. The pre-treatment process is discussed in detail.

The first set of data, consisting of about 181 records, was processed by preprocessing inside the RapidMiner Studio.4 tool, and it found about 30 lost values that were processed and replaced with values inside the feature.

In the second set of data, there are approximately 1090 records processed as we mentioned in the process of processing the data set with RapidMiner Studio.4.

This tool provides data mining and procedures identification, which includes "uploading and converting records (ETL)", processing and visualizing records, predictive analyzes, statistical modeling, evaluation and publication.

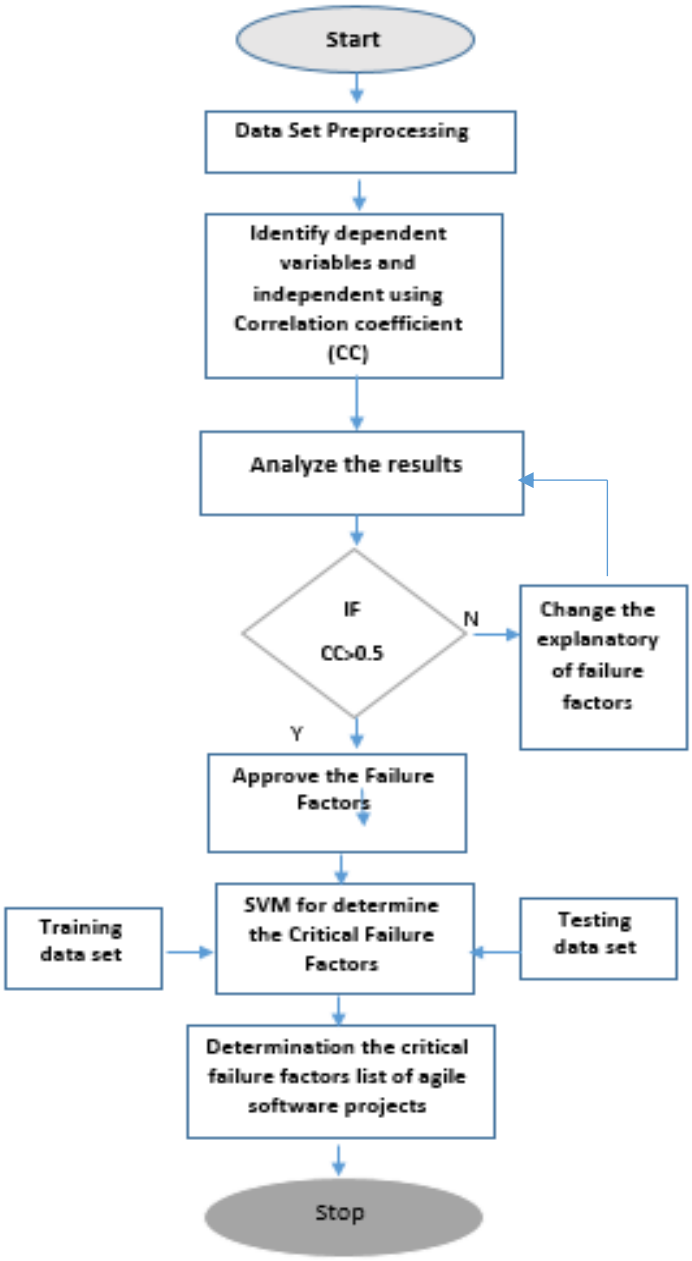


Figure 7 Proposed Approach

3.4.Phase Two: Dependent and Independent Variables

The next step for pre-treatment is to define the dependent and independent variables using the correlation coefficient (CC).

The two main variables in scientific experiments are the dependent variable and the independent variable. An independent variable is a variable that is modified or managed

in a scientific test to test results in the dependent variable. The dependent variable is the variable that modifies a response to the independent variable.

While searching for a type of relationship between variables, we try to find out if the independent variable causes a kind of substitute within its dependent variables, and based on it, the independent variable and the dependent variable are certain.

Within the first dataset, we considered MTBF as the dependent variable (since we want to expect the value). All different variables (besides identity) are used as standalone variables (input).

In the second dataset, we considered the dependent variable MTBF and the rest of the variables to be independent variables.

The dependent and independent variable were defined by a previous work whose datasets were used in this work, Then, we went on to find CC values between the variables and define a new goal to work on, and access to the power of the influence of the independent variable on the dependent variables.

In the third step, we will analyze the results from the CC process to greater or less than 0.5. looking at the results of the CC process, we found in the first dataset (the Run Time In Hours) and the second dataset, the most effective property was the Normalized Work Effort. Based on the condition set to confirm failure factors or changing the concept of failure factors, these values lead to failure and the SVM. an algorithm is applied to them.

In step four, we applied SVM. to determine the factors leading to failure by separating the data into a training group and a test group; the data was separated into 30% of the test data set and 70% of the training data set, Data is divided Most of the data is used for training purposes and a smaller part is used for testing in this process to reduce the discrepancy between the data and to a better understanding of its characteristics.

The last step identifies the critical failure factors for projects, which confirmed that the most influencing factor in the first data set is (Run Time In Hours) and in the second group is (Normalized Work Effort).

3.5.Phase Three: Result Analysis

Given the importance of being able to predict failures, this approach aims to predict failure cases expected to occur in programs, and this is achieved through the use of analytical and statistical methods and the activation of SVM., as SVM. Analysts and engineers have been able to anticipate errors with a high guarantee of the probability of occurrence through official statistics based on Artificial Intelligence, thus allowing engineers and analysts to take protection to avoid future dangers.

Correlation coefficient statistics are used to measure the strength of the relationship between two variables.

Its values range from (-1,+1) using a statistical program. The two variables are determined based on the independent variable and the dependent variable (the value that we want to forecast).

The process of analyzing the results of the CC:

We analyzed the results to less than 0.5 and greater than 0.5 and compared the values between each. As values greater than 0.5 are critical values and are classified as influencing factors, values less than 0.5 are non-critical values and are excluded.

The results of the CC process showed that in the first dataset, the Run-Time In Hours feature was the most influential; the CC value was 0.9996. In the second dataset, the most effective property was the Normalized Work Effort with a ratio of 0.9208. In other words, these features are the most influential and SVM algorithm is applied to them and to other features that had CC values greater than 0.5.

In the fourth part of this chapter, we addressed the SVM. algorithm and its implementation through the RapidMiner tool. This is presented in Figure (8).

3.6.Phase Four: Using SVM.:

One of the supervised machine learning algorithms is SVM. Training the system means giving it data and data outputs, and the system builds relationships and patterns between data and outputs so that it can later predict new outputs.

SVM. was used in this work to train the system to deal with the idea of predicting failures in the way that we mentioned previously. Because of the strength of this algorithm in terms of classification, as its performance is in the best classification, many tools and programs are used to implement SVM. In this work, we have used RapidMiner Studio 9.4. This is presented in Figure (8).

This use proved its effectiveness through the results reached from the process of predicting failures in specific factors in two sets of data by granting weights to data points in the training data set for each data set. High values indicate important data points that have the greatest impact and Low values indicate extreme points and their effect is less.

When we implemented the SVM algorithm in the first data set, the Run-Time IN Hours feature had the highest value of 5.295. This means that its effect on failures that can occur because of it is the highest while the other features come in a lower order, meaning they are extreme points that do not affect the data.

For the second dataset, the Normalized Work Effort attribute has the highest value (89.309), which means that its effect on failures that can occur because it is s the highest; other features come in a lower order, meaning they are extreme.

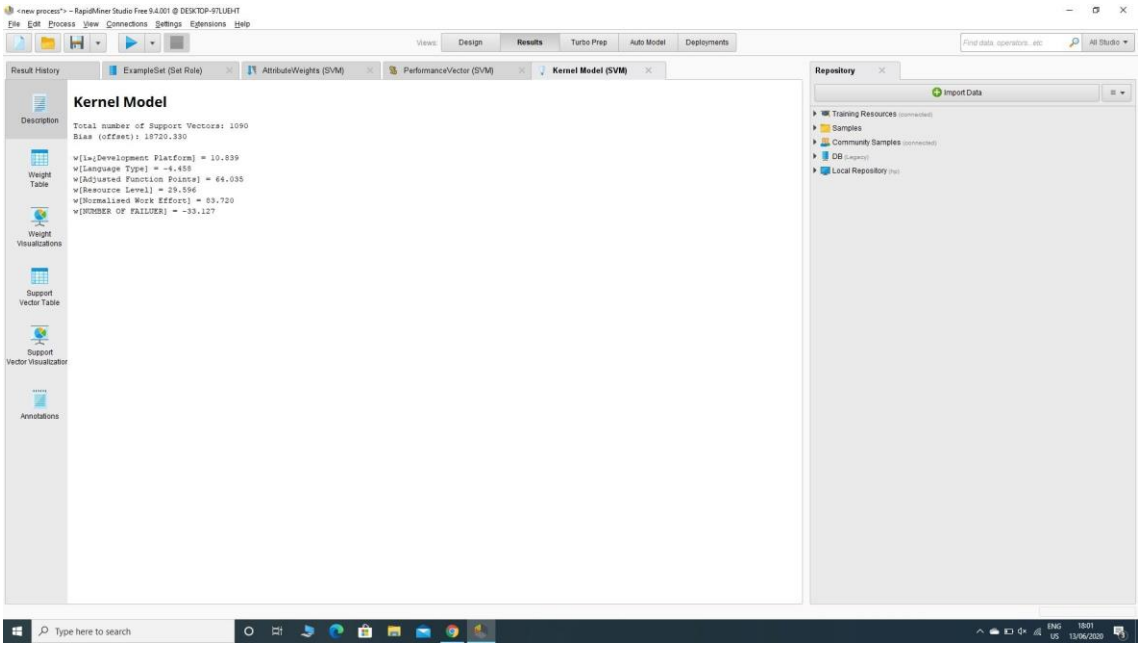


Figure 8 The RapidMiner tool

RapidMiner Studio9.4:

The RapidMiner Tool is an effective tool for data mining, knowledge acquisition, analysis, forecasting and business evaluation. This is because the software includes many programs, in the field of commercial and industrial projects, in research, training, knowledge acquisition and many useful means for researchers and curious people about extracting information and acquiring machine knowledge. We used this program to perform all the required steps from preparing the primary data to visualizing the results and evaluating the outputs, as well as verifying and improving them in a unified environment.

Predicting serious values through the R squared equation and comparing them with previously extracted values:

The coefficient of determination R refers to the ratio of variance in the dependent variable that can be expected through independent variables or dependent variables and is used in statistical models that aim to predict future results and test assumptions.

In our research, R was used to predict the feature of MTBF (shown in Figure (13) and Figure (14)) which indicates failure prediction through the R squared equation, whose elements in the first dataset were MTBF Confidence (expressed by the value of X) and Run Time In Hours (expressed by the value of Y) and represented in Figure (10).

In the second dataset, the elements of the MTBF equation (represented by the value of X) and the standard voltage (represented by the value of Y) are shown in Figure (11). The better the model, the better the distance between the actual data line and the expected data line, as this distance determines the required value.

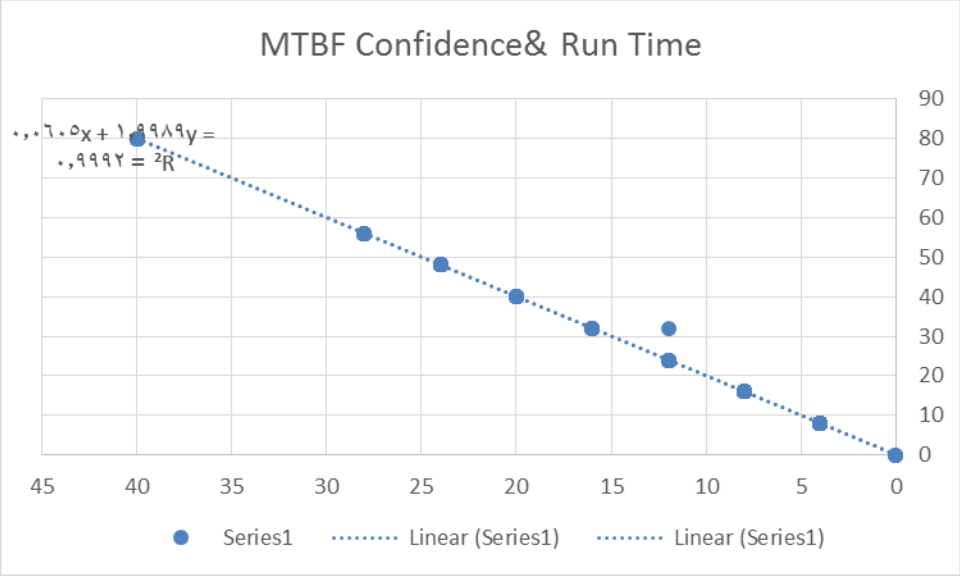


Figure 9 Finding the best fit line, forecasting and measuring R2 (for the first dataset)

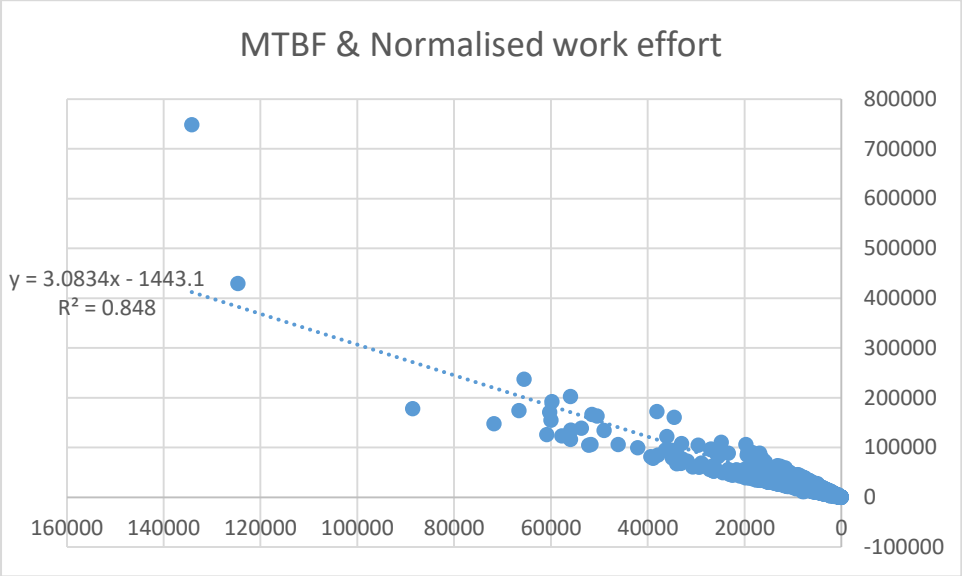


Figure 10 Finding the best fit line, forecasting and measuring R2 (for the second dataset)

3.7.Phase Five: Critical Failure Factors

In this last part, the elements were identified and categorized into critical and non-critical elements based on the outputs of the prediction process, discussed in the fourth chapter.

Based on the results reached, it was found that the critical component (the most influencing in the failure of the first dataset) was Run Time In Hours, and the most influential component in the failure of the second dataset was Normalized Work Effort.

Results evaluation:

Results were evaluated by calculating the probability value and the Absolute Error.

P-Value: This value is used to test hypotheses statistically, and it means the possibility of obtaining results close to the edges of the actual results. Accepting or rejecting hypotheses is given by Absolute Errors, distinguishing between measured or inferred values for the quantity and its true value.

CHAPTER FOUR
RESULTS ANALYSIS

CHAPTER 4:

RESULTS ANALYSIS

This section presents the results of the experiment study, which has been conducted to validate our approach. The evaluation has been performed on Support Vector Machine (SVM.) with regression, using public datasets obtained from related works as described in Chapter 3.

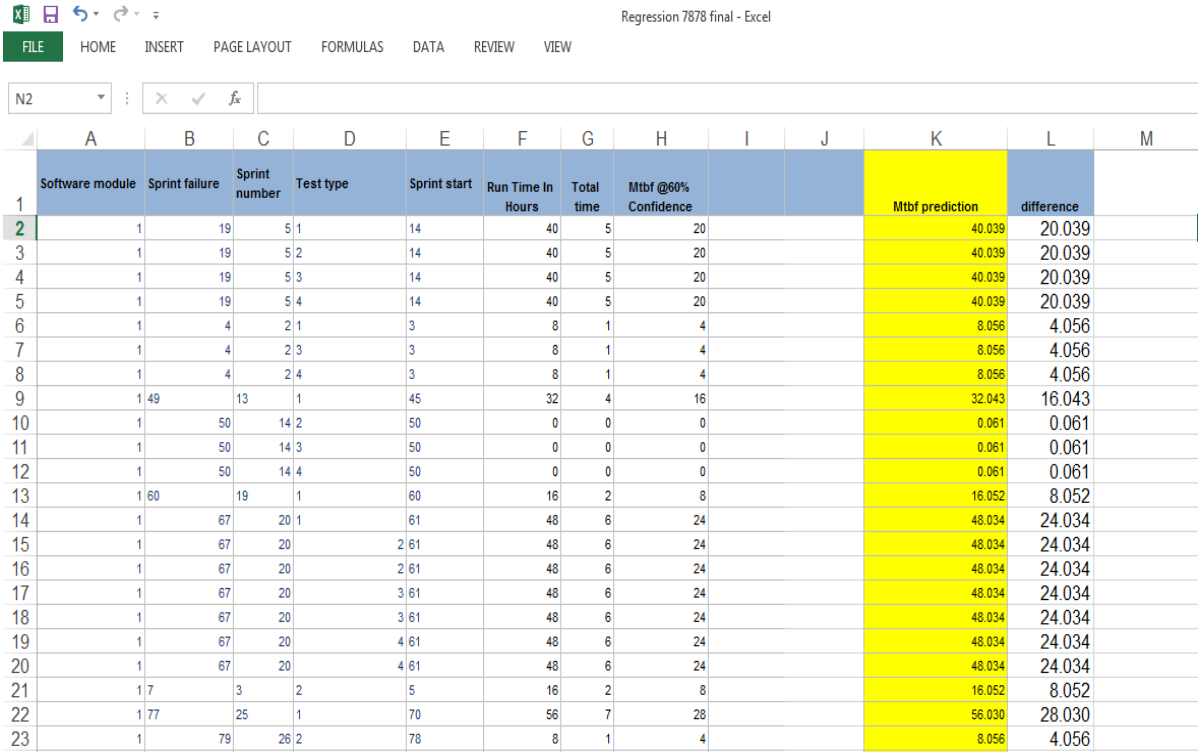
When using the first set of data from the ones mentioned in the previous chapter, after performing the experiment and recording the expected results, which were expected from the failure cases later. For this reason, Mr. Scrum can take the appropriate precautionary measure to avoid future mistakes. This method provides a safety measure throughout the optimization period and guides the team to identify the error before it occurs.

When looking at the results of the experiment, in Figs 12 and 14, the new data columns contain the following:

1- MTBF values recorded from previous experiences measured at 60%
2- Expected values for MTBF: Using the regression model, new values are created and used for forecasts. This is calculated using the results of CC and SVM. To this end a new column of data has been added, (representing an entirely new statistical element) in this regression version (represented in the Excel sheet), and the model will provide a final result (within the MTBF prediction statistics

column) based mainly on the regression line formula (which has been demonstrated in Previous sections), so this wide range is a prediction for the next graceful fast race.

3- Distinguish between real and expected values, equal to: historical MTBF values - MTBF prediction values.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Software module	Sprint failure	Sprint number	Test type	Sprint start	Run Time In Hours	Total time	Mtbf @60% Confidence			Mtbf prediction	difference	
2	1	19	5	1	14	40	5	20			40.039	20.039	
3	1	19	5	2	14	40	5	20			40.039	20.039	
4	1	19	5	3	14	40	5	20			40.039	20.039	
5	1	19	5	4	14	40	5	20			40.039	20.039	
6	1	4	2	1	3	8	1	4			8.056	4.056	
7	1	4	2	3	3	8	1	4			8.056	4.056	
8	1	4	2	4	3	8	1	4			8.056	4.056	
9	1	49	13	1	45	32	4	16			32.043	16.043	
10	1	50	14	2	50	0	0	0			0.061	0.061	
11	1	50	14	3	50	0	0	0			0.061	0.061	
12	1	50	14	4	50	0	0	0			0.061	0.061	
13	1	60	19	1	60	16	2	8			16.052	8.052	
14	1	67	20	1	61	48	6	24			48.034	24.034	
15	1	67	20	2	61	48	6	24			48.034	24.034	
16	1	67	20	2	61	48	6	24			48.034	24.034	
17	1	67	20	3	61	48	6	24			48.034	24.034	
18	1	67	20	3	61	48	6	24			48.034	24.034	
19	1	67	20	4	61	48	6	24			48.034	24.034	
20	1	67	20	4	61	48	6	24			48.034	24.034	
21	1	7	3	2	5	16	2	8			16.052	8.052	
22	1	77	25	1	70	56	7	28			56.030	28.030	
23	1	79	26	2	78	8	1	4			8.056	4.056	

Figure 11 Mtbf forecasting result (1)

I	H	G	F	E	D	C	B	A
difference	mtbf predicition	MTBF	NUMBER OF	Normalise	Resource	Adjusted F	Language	Developme
13678.84675	36507.5248	22828.67805	0.49814536	11372	1	859	4	1
96630.7859	274493.587	177862.8011	0.49788376	88555	1	1306	3	1
12405.81854	32893.78	20487.96146	0.49785334	10200	1	465	3	1
10959.86715	27898.672	16938.80485	0.50652924	8580	1	359	4	1
4809.431185	10798.1356	5988.704415	0.50662043	3034	2	199	3	1
3502.960641	7165.8904	3662.929759	0.50669822	1856	3	225	3	1
20972.17739	55710.94	34738.76261	0.50663866	17600	1	4272	4	1
13600.98933	35237.164	21636.17467	0.50655905	10960	3	599	3	1
3378.652955	6826.7164	3448.063445	0.50637119	1746	1	357	4	1
11486.50691	29372.5372	17886.03029	0.50642875	9058	1	599	3	1
1880.994595	2661.043	780.0484046	0.50637883	395	1	331	3	1
2850.022241	5355.9346	2505.912359	0.50640239	1269	3	212	3	1
5803.704005	13573.1956	7769.491595	0.50633944	3934	4	194	3	1
5679.295373	13230.9382	7551.642827	0.50624746	3823	4	185	6	1
5595.645948	12999.6832	7404.037252	0.50621031	3748	1	537	3	1
3780.513083	7949.074	4168.560917	0.50616989	2110	1	426	4	1
5727.882453	13369.6912	7641.808747	0.50616289	3868	2	484	3	1
2495.348846	4372.33	1876.981154	0.50613188	950	1	610	4	1
2135.330753	3370.225	1234.894247	0.50611621	625	1	556	3	1
8034.19975	19789.33	11755.13025	0.50616198	5950	1	170	3	1
5702.799485	13298.773	7595.973515	0.50618923	3845	1	778	3	1

Figure 12 MTBF forecasting result (2)

The results indicate that data consisting of 181 rows overall difference for all grades 2663 with an average of 148 minutes. Average running time Test in hours for the system that was used was 1770 minutes (29.5).

The results indicate that the percentage of failure to predict the time used or experiment in the test is for all 181 rows, for all test times recorded, is 3% for MTBF.

SVM achieved a 97% success in predicting compared to previous work whose results indicated that the use of ADT achieved a statistically significant overall success rate of 93.5%.

Therefore, we conclude that our methodology at work has achieved better results than the other method.

As shown in Figure (13), the most critical element affecting the program's failure in the first dataset is the Run Time In Hours component.

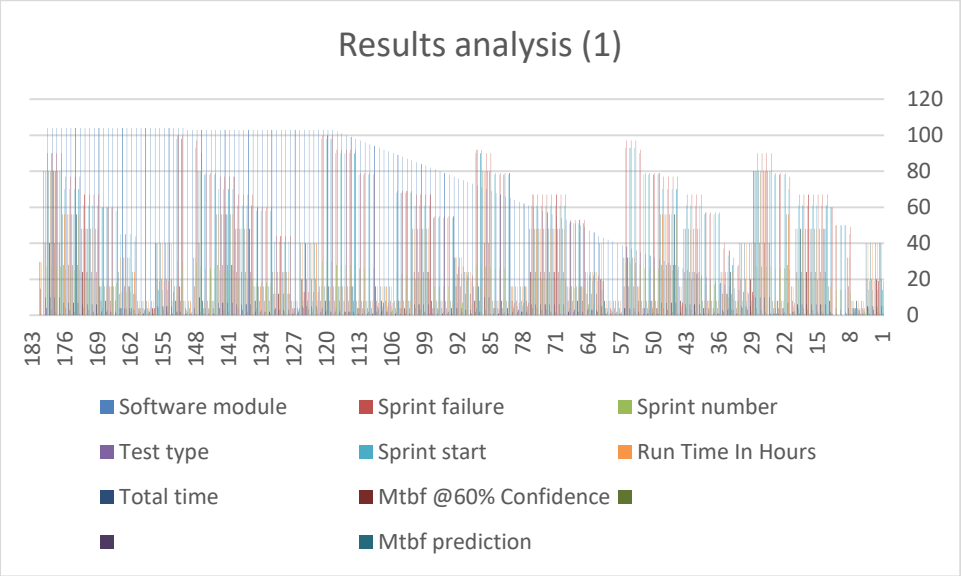


Figure 13 Results Analysis (1)

In the second dataset, which consisted of 1091 rows, the total difference for all grades is 2886.36059, with an average of MTBF = 18721 minutes, with an average running time for the 6539 minutes test for the system that was used.

The results of the second dataset indicate that the percentage of failure to predict the time used or experiment in the test is for all 1091 rows, for all test times recorded, is 1.5% for MTBF, SVM achieved 98.5% prediction.

The volume of data has an effective role in the efficacy of forecasting failures, as we note that the larger the volume of data, the more accurate the prediction.

As shown in Figure (14), the most critical element affecting the program's failure in the second dataset is Normalized Work Effort.

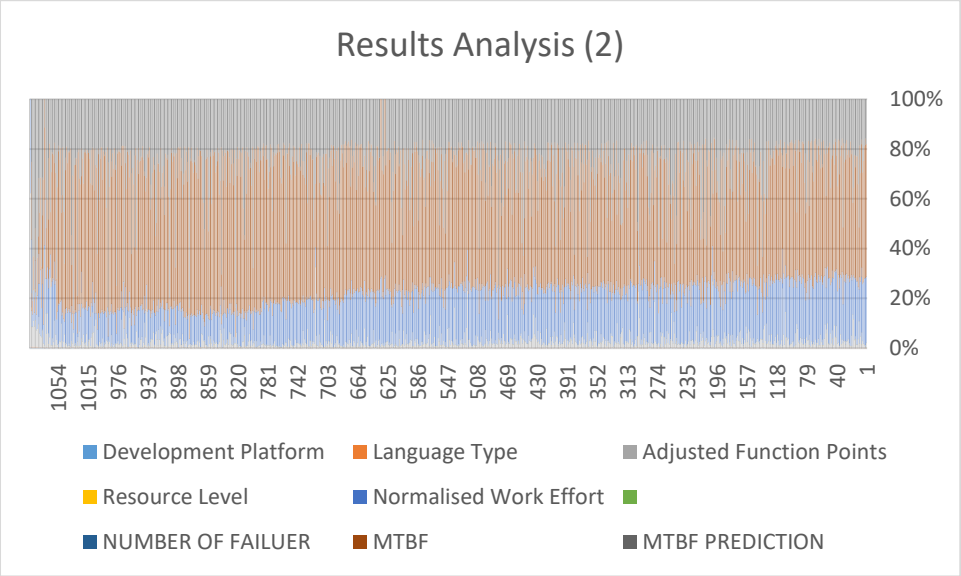


Figure 14 Result Analysis (2)

CHAPTER FIVE
RESULTS ANALYSIS

CHAPTER 5:

CONCLUSIONS AND FUTURE WORKS

Software failure prediction is a vital task during software development to help testing team to focus on defect proneness modules. To support that, various methods have been used to build models that can predict faulty modules based on datasets collected from software industries. Among them, Support vector machine has shown good performance for this problem. Thus, this research shows that the support vector machine and coefficient correlations help to increase the classification accuracy and improve the percentage of failure prediction probability over different datasets collected from software data repositories. As shown in this research, the size of the dataset is key to increasing the accuracy and enhancing the classification. The second dataset (the larger one) indicates that the percentage of failure to predict the time used for all test times recorded, is 1.5% for MTBF, SVM achieved 98.5% prediction. Whereas, the first dataset gives 3% for Mean time between failures and SVM achieved a 97%

The main challenge in such research is to obtain the related dataset. Most of the agile datasets are published under payment constraints, which minimized the chance to use more datasets. Our plan in future is to obtain more and variant datasets to examine our approach for more investigation. Firefly Algorithm is nominated to be used in the future to enhance the SVM parameters input selection, which hopefully increase the classification accuracy in different datasets.

REFERENCES

Aggarwal, P. a. (2020). Agile Methodology Influence on SDLC (Software Development Life Cycle). *Studies in Indian Place Names, 40*, 4579--4589.

Akoglu, H. (2018). User's guide to correlation coefficients. *Turkish journal of emergency medicine*(Elsevier), 91--93.

Al-Zewairi, M., Biltawi, M., Etaiwi, W., & Shaout, A. (2017). Agile software development methodologies: survey of surveys. *Agile software development methodologies: survey of surveys, 5*(.), 74--97.

Anderson, J. a. (2015). Striving for failure: an industrial case study about test failure prediction. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (pp. 49--58). IEEE.

Barstow, D. (1988). Artificial intelligence and software engineering. In *Exploring artificial intelligence* (pp. 641--670). Elsevier.

Batarseh, F. A. (2018). Predicting failures in agile software development through data analytics. *Software Quality Journal*(Springer), 49--66.

Chigurupati, A. a. (2016). Predicting hardware failure using machine learning. In *2016 Annual Reliability and Maintainability Symposium (RAMS)* (pp. 1--6). IEEE.

Cooper, R. G. (2014). What's Next?: After Stage-Gate. *Research-Technology Management, 57*, 20--31.

Diaz, J. a.-M. (2009). Mapping CMMI level 2 to scrum practices: An experience report. In *European Conference on Software Process Improvement* (pp. 93--104). .: Springer.

Ferreira, P. a. (2019). Detrended correlation coefficients between exchange rate (in dollars) and stock markets in the world’s largest economies. *Economies, 7*(Multidisciplinary Digital Publishing Institute), 9.

Fronza, I., Sillitti, A., Succi, G., Terho, M., & Vlasenko, J. (2013). Failure prediction based on log files using random indexing and support vector machines. *Journal of Systems and Software, 86*(.), 2--11.

Glaiel, F. (2012). Agile project dynamics: a strategic project management approach to the study of large-scale software development using system dynamics. *Massachusetts Institute of Technology,* .(.), .

Hayat, F. a. (2019). The Influence of Agile Methodology (Scrum) on Software Project Management. In *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)* (pp. {145--149}). IEEE.

Hoda, R. a. (2017). Systematic literature reviews in agile software development: A tertiary study. *Information and Software Technology, 85*(Elsevier), 60--70.

Huang, S. a. (2018). Applications of support vector machine (SVM.) learning in cancer genomics. *Cancer Genomics-Proteomics*(International Institute of Anticancer Research), 41--51.

IDRI, A. a. (2012). Software cost estimation by fuzzy analogy for ISBSG repository. In *Uncertainty Modeling in Knowledge Engineering and Decision Making* (pp. 863--868). World Scientific.

Jain, M. a. (2020). Speech emotion recognition using support vector machine. *arXiv preprint arXiv:2002.07590*.

Jorgensen, M. a. (2006). A systematic review of software development cost estimation studies. *IEEE Transactions on software engineering, 33*(IEEE), 33--53.

Lei, H., Ganjeizadeh, F., Jayachandran, P. K., & Ozcan, P. (2017). A statistical analysis of the effects of Scrum and Kanban on software development projects. *Robotics and Computer-Integrated Manufacturing, 43*(.), 59--67.

Li, J. a. (2017). Software defect prediction via convolutional neural network. In *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS* (pp. 318--328). IEEE.

Mohammed, B. a. (2019). Failure prediction using machine learning in a virtualised HPC system and application. *Cluster Computing, 22*(Springer), 471--485.

Nassif, A. B. (2019). Software development effort estimation using regression fuzzy models. *Computational intelligence and neuroscience*(Hindawi).

Olive, D. J. (2017). *Linear regression.* Springer.

Priyadarsini, P. I., Sai, M. S., Suneetha, A., & Santhi, M. V. (2018). Robust Feature Selection Technique for Intrusion Detection System. *International Journal of Control and Automation, 11*(.), 33--44.

Rafi, S. M. (2012). Incorporating fault dependent correction delay in SRGM with testing effort and release policy analysis. In *2012 CSI Sixth International Conference on Software Engineering (CONSEG)* (pp. 1--6). IEEE.

Ruppert, D. a. (2003). *Semiparametric regression.* Cambridge university press.

Schmidt, A. F. (2018). Linear regression and the normality assumption. *Journal of clinical epidemiology*, *98*(Elsevier), 146--151.

Shi, L. a. (2018). Multiple attribute group decision-making method using correlation coefficients between linguistic neutrosophic numbers. *Journal of Intelligent \& Fuzzy Systems*(IOS Press), 917--925.

Srivastava, A. a. (2017). SCRUM model for agile methodology. In *2017 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 864--869). IEEE.

Sun, Y. a. (2018). Utilizing deep architecture networks of VAE in software fault prediction. In *2018 IEEE Intl Conf on Parallel \& Distributed Processing with Applications, Ubiquitous Computing \& Communications, Big Data \& Cloud Computing, Social Computing \& Networking, Sustainable Computing \& Communications (ISPA/IUCC/BDCloud/SocialCom/SustainC* (pp. {870--877}). IEEE.

Surov, A. a. (n.d.). Correlation between apparent diffusion coefficient (ADC) and cellularity is different in several tumors: a meta-analysis. Oncotarget. 2017; 8: 59492--59499.

المـلخـص

يُعد التنبؤ بفشل البرمجيات نشاطًا مهمًا أثناء تطوير البرمجيات الرشيقة حيث يمكن أن يساعد المديرين على تحديد وحدات الفشل. وبالتالي، يمكن أن تقلل من وقت الاختبار والتكلفة وتعيين موارد الاختبار بكفاءة. للتأكد من احتمال فشل تطوير البرنامج في مستوى معين، هناك طريقتان تستخدمان في هذا العمل، Support Vector Machine (SVM) لتحديد العوامل التي تؤدي إلى الفشل، ولتحديد المتغيرات التابعة والمستقلة تم استخدام معامل الارتباط (CC).

في هذا البحث ، تم استخدام RapidMiner Studio9.4 لتنفيذ جميع الخطوات المطلوبة من إعداد البيانات الأولية إلى تصور النتائج وتقييم المخرجات ، وكذلك التحقق منها وتحسينها في بيئة موحدة.

تم استخدام مجموعتي بيانات في هذا العمل، وتشير نتائج المجموعة الأولى إلى أن النسبة المئوية للفشل في التنبؤ بالوقت المستخدم في الاختبار هي لجميع الصفوف البالغ عددها 181 صفًا، ولجميع أوقات الاختبار المسجلة، وهي 3٪ لمتوسط الوقت بين الإخفاقات (MTBF). حيث حقق SVM نجاحًا بنسبة 97٪ في التنبؤ مقارنة بالأعمال السابقة التي أشارت نتائجها إلى أن استخدام وقت التأخير الإداري (ADT) حقق معدل نجاح إجمالي ذو دلالة إحصائية بنسبة 93.5٪. في الوقت نفسه، تشير نتيجة مجموعة البيانات الثانية إلى أن النسبة المئوية للفشل في التنبؤ بالوقت المستخدم أو التجربة في الاختبار هي لجميع الصفوف 1091، لجميع أوقات الاختبار المسجلة، 1.5٪ لـ MTBF، حقق SVM توقع 98.5٪.