iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** VI **Month of publication:** June 2023

DOI: https://doi.org/10.22214/ijraset.2023.53747

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue VI Jun 2023- Available at www.ijraset.com

A Study of Software Defect Prediction Using Artificial Intelligent and Business Risk Management Approach

Ahmad Hafizu Sale[1], Dr Wei Wang[2]
[1, 2]*School of statistics, Zhejiang University of Science and Technology*

Abstract: ***The usage of several artificial intelligence algorithms in the area of software defect prediction is reviewed in the literature in this paper, it review the use of various techniques for artificial intelligence in the field of software defect prediction. Software system flaws remain a significant issue, and modern software development now faces many difficulties. Identification, estimation, and evaluation of the risks are crucial steps in order to avoid, minimize, and monitor the risks and their effects.This literature review is quite helpful because it improves understanding of the studied area. A software fault is a flaw in an executable product that results in software failure. Saving money, time, and effort may be accomplished by identifying general software process areas that may require attention from the start of a project and understanding the potential causes of problems. Planning, managing, and carrying out software development activities could be aided by the ability for early fault estimation. In order to study how many prediction approaches have been developed over this time to address the problems linked to software defect, this research conducts a literature review of works from the previous many years.In the field of software development, the study also discusses software risk management. This study makes a significant contribution by proposing a method for analyzing, organizing, and utilizing information technology initiatives in the context of software defect prediction and risk management.***
Keywords: Software defect, Artificial intelligence, Neural Network, Fuzzy logic, Data prediction,Software risks,Risk management, classification, Identification, Assessment, mitigation.

I. INTRODUCTION

In every software project, risk is a component that is there. Software development is still costly since it takes a lot of work. Over the past 15-20 years after survey has revealed essentially no change in the rate at which IT initiatives fail. For any industry to select the methods that will help it advance, it is necessary to evaluate where it stands in terms of technical maturity. Because time, money, quality, market conditions, or even a tiny error can derail an entire software project, addressing specific IT risks associated to our software projects is necessary to ensure project success. All software development firms, including Microsoft, SAP, Oracle, IBM, etc., have extensively acknowledged the need for project risk management. In order to reduce losses and increase software development success, it is important to have a thorough grasp of the issue domain, effective risk management tools, and consistent methods. The goal of project risk management is to enhance project performance by systematic risk identification and assessment, risk reduction or risk avoidance strategy development, and opportunity maximization. Risk management is a continuous process that is carried out during the course of a system's life cycle. Planning for risk management, early risk identification and analysis, prompt execution of remedial measures, ongoing monitoring and reevaluation, as well as communication, documentation, and collaboration, are all necessary for effective risk management.As part of the writing, several features or metric qualities were discovered and used to anticipate programming faults. It would be helpful and straight forward if we could choose the arrangement of metrics that are most important and use them more to anticipate the flaws, as opposed to managing such a variety of property. In order to determine the compelling relationships between the software metric qualities and defect prediction, we reviewed a number of methodologies in this research, including ANN, fuzzy logic, fuzzy art maps, etc. Many supervised machine learning algorithms, including neural networks, support vector machines, Naive Bayes, and others, are available for software defect prediction, which can shorten project duration and lower project costs.There is also a brand new supervised machine learning method called Fuzzy ARTMAP for predicting software fault. The metric qualities employed in the reviewed studies are often software datasets that have been made publicly available with the aim of enabling repeatable, evident, debatable, and improvable predictive models of software engineering. The majority of the time, it is possible to use the NASA MDP dataset to foresee flaws in software solutions.

The purpose of the proposed study is to determine whether metric based assessment in the early life cycle (i.e., necessity measurements), metric-based assessment in the late life cycle (i.e., code measurements), and metric based assessment in the early life cycle (i.e., necessity measurements) combined with metric based assessment in the late life cycle (i.e., code measurements) can be used to identify defectiveness issue inclined modules using various supervised and unsupervised methods Early program planning stages should kick off risk management, which should last the duration of the program. Additionally, for root cause and consequence management, risk management is most effective when it is fully integrated with the program's systems engineering and program management procedures. A typical misunderstanding about risk management is that issues should be tracked and identified first (instead of risks), and then the issues should be managed (instead of the root causes). For instance, if a program is running late in getting engineering drawings to the fabricator, this is not a danger; rather, it is a problem that has already arisen and needs to be fixed, and it may be difficult to do so. Failure of the components being tested or analyses that reveal a design flaw are further instances of problems. True dangers are often hidden by this method, which tracks risks rather than addressing or reducing them.

II. RELATED WORK

This section provides a concise review of many research that have been conducted in the previous few decades in the area of software fault prediction. We have discussed a number of methodologies, their advantages, and the ensuing findings that have made significant contributions to the creation of a system for predicting software defects at a stage where they are very dependable. Despite the enormous number of studies that have been published, according to Fenton and Neil (1991), the problem of software defect forecasting is still far from being solved. There are certain incorrect assumptions regarding how flaws are described or seen, and this has led to misleading outcomes.

Their argument is clearer when we consider that while some articles refer to faults as watched inadequacies, others refer to them as lingering ones. By anticipating the size and complex nature, testing, process quality information, multivariate approach, and other factors, they may evaluate the software's features. In their work from 1994, R. Chidamber and Kemerer introduced new categories of software metrics that should be used for object-oriented design. By evaluating these metrics, they discovered that they tend to have a variety of characteristics, and they offer a few ways that an object-oriented approach might differ from traditional technique.

They carry out six distinct sets of measurements, including WMC, RFC, NOC, DIT, CBO, and LOCM. Venkata U.B. Challagulla (2005) suggested a unique machine learning model for identifying problematic real-time software modules using a variety of NASA datasets, such as KC1,PC1,CM1,JM1. This model is to be used to predict the s/w product flaws. It turns out that the KC1 dataset is the best at predicting the fault when we look at the mean absolute error of the several software forecasts that are currently available. In five NASA datasets, CM1, JM1, KC1, KC2, and PC1, A. Gunnes Koru (2005) uses a few machine learning algorithms to predict programming errors in software modules. They made a mistake using class level information rather than method-level information to make predictions for KC1. When compared to using method-level information in this case, the use of class-level information improved forecast execution. To anticipate programming flaws, Anuradha Chug et al. (2010) present various grouping and bunching solutions. Three information mining classifier calculations, J48, Random Forest, and Naive Bayesian Classifier (NBC), are evaluated based on a variety of metrics, including ROC, Precision, MAE, RAE, and so on. After that, a grouping approach is attached to the data set using k-implies, hierarchical clustering, and density-based clustering With the primary goal of decomposing the execution of various classifiers on defect expectation based on open area NASA information set KC1, Shanthini et al. (2012) covered the work.

They deconstructed the execution of the classifiers using conventional measures, such as exactness, review, and F-measure. According to this study, SVM model construction is acceptable, adaptable to OO frameworks, and helpful in identifying classes that are prone to flaws for higher amounts of measurements (class). According to C. Akalya Devi et al. and Martin Sheppard (2014) defect forecast analysts, direct use of blind analysis improved reporting practices and led to more inter group investigations with the aim of identifying easier expertise issues.

In the end, study is necessary to determine whether this propensity is widespread across various application domains. The accuracy of the NASA MDP is superior than any other dataset included in this work for forecasting with imperfections. Pushpavathi T.P et al. (2014) describe a study for anticipating defect prediction in software modules using a coordinated approach of fuzzy c-means clustering based on evolutionary algorithms and random forest analysis. This method was developed using Fuzzy C-implies bunching with Random Forest grouping coupled on a specific information set and analysis was done. Finally, using five NASA open space software defect information sets, the results were approved.

III. RISK MANAGEMENT METHODOLOGY, PROCESS, AND TOOLS

Risk management in software engineering covers all facets of the various program phases as they relate to one another, from conception to disposal. Design requirements are integrated with other life cycle concerns, like manufacturing, operations, and support, through the risk management process. The project manager, the project team, and the contractor must all be committed for the risk management process to be successful. The risk management process assists in creating and achieving reasonable cost, schedule, and performance goals when appropriately resourced and put into practice. It also identifies hazards early enough for specific attention and mitigation.

1) The project team and management should establish a risk management process that includes risk identification, risk analysis, risk mitigation planning, risk mitigation plan implementation, and risk tracking. This process should be integrated and continuously used throughout the program, including during the design phase. There are five fundamental steps in the risk management process approach.
2) Identify the risks - Recognize the common issues that could have a negative impact on the project.
3) Assess the risks - Order the hazards according to their degree of certainty, likelihood that they will occur, and other factors.
4) Plan the risk response - Consider several approaches to risk assessment and change the project plan as necessary to account for the risk.
5) Monitor the risks - Keep coming back to the risk profile throughout the project, reassessing any significant risks, and updating the risk profile to reflect any actions taken.

Learn from the process of risk identification, assessment, and management by documenting your lessons learned. Early risk assessment and identification are essential for minimizing negative project development deviations and maximizing positive outcomes. In order to assess the level of potential risk to the project, identifying software risks entails gathering and categorizing information about the software development project.

This activity can make use of some CASE tools and machine learning algorithms. Based on risk identification information, risk strategy and planning for software risks include exploring possible risk outcomes and then creating backup plans for these outcomes or for risk reduction. Determining the impact of potential dangers is necessary for risk assessment of software. Based on the knowledge gathered from the earlier activities of identifying, planning, and analyzing risks, the action of mitigating and avoiding software risks is carried out. Based on data gathered from earlier subjects, risk reporting compares the current state of hazards to those that have already been identified. Risk forecasting entails leveraging the historical knowledge of previously identified hazards. Risk prediction is generated from the prior actions of identifying, planning, analyzing, mitigating, and reporting risks.

The book Continuous Risk Management by The Software Engineering Institute contains a very structured definition of the risk management process. The actions to be completed, the tools and procedures to be utilized at each level, and the products that are input into or output from activities are all described in this document, together with a glossary of risk management words. Additionally, SEI explains how to implement ongoing risk management within the company. Start, Install, and Improve are the three stages that make up the process.

During the start phase, management commitment is guaranteed, a foundation for facilitating the shift to risk management is constructed, project staff is introduced to the goal and ideas of risk management, and a baseline of project risks is identified and mitigation plans are developed. Infrastructure refers to the individuals who will play key roles in carrying out the implementation process. To oversee the initial process of risk identification, a baseline team is also required. A risk baseline has been created at the conclusion of the phase.

The objective of the install phase is to integrate risk management into the organization's current project and risk management practices. The resulting processes are then documented in a risk management strategy. Installed are any databases, forms, or instruments that are computerized. The people working on the project are trained on both the procedures to be followed and the techniques and equipment to be employed. The project must finally begin its risk management activities and make them a routine part of daily operations.

a) When fundamental risk management procedures have been applied to the project, the last phase, improve and expand, begins. The processes, methods, and tools must be improved in order to ensure that risk management is more integrated into standard project risk management. It is important to record the lessons learned. It's also necessary to provide ongoing training and facilitation. The organization should extend risk management to further projects.
b) The author recommends the key concepts for software risk management, taking into account the specifics of the IT sector and the background in the linked software fields:

c) To avoid big IT projects, you should break them down into manageable stages with milestones, clear results, and a person in charge of the overall deliverable achievement.

d) Recruit and include IT professionals with relevant experience for IT project management rather than technical workers to concentrate on the project management's quality.

e) Employ the chopper view and strategic judgment of impartial subject-matter experts who can examine the project from their own point of view and independently determine the project's state of health.

f) Take into account all potential hazards, not only those that have already occurred in past projects or are listed in check lists, risk databases, or other sources.

At a high, strategic level, many IT projects actually resemble one another quite a bit. Involved parties and specific events vary between them. This explains the wide range of potential risks associated with IT projects. Numerous experts have attempted to pinpoint the sources of IT project risk. The most frequent IT risks, according to Boehm, are:

- project team members with inadequate training;
- temporary planning and project budgets that are not practical;
- inappropriate product features generated;
- interfaces that are not user-oriented; and testing in real-world scenarios that are unsuccessful.

Not all identified risks should be treated the same. Some identified risks are more likely to occur, and some, if realized, would have a bigger impact. Risk analysis and management depends on the types of risks being considered. Within the context of the technological and business perspectives, there can be distinguished three main elements of software risk: technical, schedule/scope, cost.

- The functionality, quality, dependability, and timeliness problems with the software product are considered technical hazards. Even without scope adjustments in the middle of the project, unanticipated technical difficulties might nonetheless completely alter the course of the work. Even if project managers are quite familiar with the technology being used, surprises can still happen. For example, a component may have previously operated without issue, but when it is integrated with another component, the result is a total disaster. The risk of unanticipated technical constraints is reduced with increased technical expertise, although this risk is nonetheless always present.
- Timetable and domain Risks are related to the software product's development timetable and scope. Changes in scope are common in IT projects, and they are somewhat understandable since there are always suggestions after you have begun the implementation, regardless of how comprehensive your specification is. These recommendations frequently call for drastic adjustments and change requests that can completely upend any timeline. Software managers should examine the risks from a different perspective and then obtain comprehensive information in order to address the holistic view of hazards. Additionally, technical difficulties may have an impact on the scope. If a particular functionality is technically impractical to implement,
- Cost risks include concerns with budget, nonrecurring expenses, recurring costs, fixed costs, variable costs, profit/loss margin, and realism. Cost risks are related to the cost of the software product during software development, including its ultimate delivery. The risks should be identified and then evaluated in terms of both probability and impact. To simply rank and order the risks and allow the team and sponsors to discuss how to address each risk, the project team will take these two dimensions and multiply them together to produce a risk score. The Risk Score enables us to rank the dangers in order of importance. For instance, if the first risk scores at $100K and the second at $160K, the second risk is the better investment. risk represents a bigger threat to the project’s baselines and has bigger priority.

The project team must decide how the actual assessment (risk analysis and root cause analysis) will be carried out for each risk assessment. If the resources required to complete the assessment exceed those available from within the program team, it may be appropriate to have teams outside the project team. This team, which consists primarily of experts in systems engineering, logistics, manufacturing, testing, schedule analysis, and cost estimating, will be responsible for carrying out the risk assessment. Depending on the nature and priority of the risks, after they have been recognized and evaluated, they should be mitigated with one of the response steps. The four primary categories of responses can vary based on the approach used, but they are as follows:

- Reduce the risk by incorporating specific strategies into the project's scope that will address its occurrence or reduce the possibility that it will;
- Reduce Risk - exclude from the project any scope that contains Risk;
- Share the Risk - Give another party risk by giving them ownership of the scope;

- ❖ Accept the Risk - Take no action, let the risk happen, and deal with it if it happens. In order to identify the high, medium, and low categories based on the Risk Scores and assign responses to those categories, the successful project manager should facilitate a conversation among the team and sponsors.

The risk management procedures are completed by risk monitoring and recording lessons learned. The use of the risk database from previous projects to design upcoming projects can assist managers in avoiding the majority of known issues and enables them to apply best practice experience and project skills rather than learn from their own mistakes.

The relationship between software risk management and other project management domains, including quality management, requirements management, people management, etc., is widely established. This means that any dangers related to the connected locations should be investigated during the identification step. For instance, the questions listed below should be asked to identify hazards in the quality management area:

- ✓ Is the business prepared for the introduction and use of the new information technology, and are all business procedures in line with it?
- ✓ Is the change control process for IT applications totally transparent and understandable for the end users?
- ✓ Are there any experts who can provide rapid assistance and prevent downtime in the event that there are any technical issues with the IT solution?
- ✓ According to the authors' experience in software engineering, the majority of the critical project risks can be found by looking into critical success factors and/or critical IT risk failures. The author has chosen the following significant software project failures for the risk management procedure:
- ✓ The absence of user participation during the requirements gathering phase.
- ✓ The lack of time and attention from the primary project stakeholders, as well as the limited assistance from the project managers and leaders.
- ✓ A lack of competent project team members and a lack of software development experience
- ✓ Unrealistic timeline and scope expectations on the part of the client or customer.
- ✓ Conflicts that may arise during project coordination and interaction between management, employees, client functional departments, stakeholders, sponsors, contractors, and other third parties
- ✓

IV. METHODOLOGY INVOLVED IN S/W DEFECT PREDICTION

We looked at a number of studies to analyze the various methods used in this decade to anticipate software defects perfectly. It has been noted that a variety of software measures are utilized in conjunction with artificial intelligence algorithms to estimate software anomalies.

Table 1.1 Summarized Description of Technology used in s/w Defect Prediction in Year Wise Manner.

Publication year	Methodology applied	accuracy
Software defect prediction using boosting technique 2013	Boosting with decision stump	86.94%
	Boosting with REP tree	86.52%
	Boosting with M5	87.37%
Software defect prediction tool based on neural network	Linear function based neural network	80.3%
	Quadratic function based neural network	78.8%

	Levenberg marquardt LM algorithm base neural network	88.1%
Software faault prediction in objective oriented software system using density based clustering approach	Density based clustering approach	58.63%
Applying machine learning for fault prediction using software metrics	Naive bayes	68.27%
	Support vector machine	80%
	K- star	72.41%
	Random forest	70.3%
Comparative classifier for software quality assessment	Neural network	76.27%
	Radial basis function	69.52%
Empirical assessment of machine learning based softer defect prediction techniques	Support vectoor logic regression	--
	Nn for discrete goal field	--
	Logistic regression	--
	Naive bayes	--
	Instance based learning	--
	J48 tree	--
	1-rule	--

V. RISK MANAGEMENT ORGANIZATION AND ROLES

The entire program team must be involved in risk management in order for it to be effective, and it may also need outside assistance from specialists versed in key risk areas like threat, technology, design, manufacturing, logistics, schedule, cost, etc. The risk management procedure should also encompass hardware, software, the human factor, interfaces, and other integration-related problems. The project manager is in charge of risk management. However, the process of identifying and analyzing risks should involve all project stakeholders. It's crucial to comprehend the project's scope from the outset in order to avoid serious issues and properly position IT projects.

For instance, many managers are aware with the scenario in which a feature that was given low priority at the start of the project ended up being crucial to the system's performance. This resulted in the convoluted processes for modification requests, which may be essential for the project's finances and resource health. The possibility of requirements modification and project delays exists even when the requirements documents are transparent and well-documented. Knowing this, the managers should have formal project scope change procedures that they should have agreed upon with the customer throughout the project planning stage.

The extended project team executes risk management and mitigation procedures overall. Representatives from the user, lab, contract management, specialty engineering, test and evaluation, logistics, and industry are examples of external experts.The assessment process should include end users, who are crucial contributors to program trade analyses, in order to strike an acceptable balance between performance, schedule, cost, and risk. The project team will be better able to identify program risks and design and implement management strategies if they have deep ties to the industry and, eventually, the chosen contractor's.

The project team or program office (in the case of significant advances) must evaluate and manage each and every software risk because they are all of concern. The program office must analyze the entire risk taken on by the developing contractor after deciding which risks and how much of each risk to share with the contractor. It is necessary for the program office and the developer to share a database and software risk management methodology. The program office must present all program risks for consensus in order to achieve successful mitigation. A common risk database that is accessible to the program office is a very useful resource. Risk reduction entails choosing the course of action that best balances cost and performance. It is also possible that throughout the system life cycle there may be a need for different near-term and long-term mitigation approaches. Not all risks are transferred to the contractor; rather, the program office shares some risks with the development, production, or support contractor. A capable and maintainable system must always be developed by the program office as a duty to the system user, and the office cannot escape this obligation.

The Project Manager and members of the application development program office will carry out risk management procedures for identified hazards. The Project Manager will be in charge of keeping an eye on these activities with assistance from other team members. These actions consist of:

1) Develop and maintain a project software development plan.
2) Develop and maintain a project risk management plan.
3) Identify high-level risks applicable to the project.
4) Identify additional project-specific risks through the Risk Analysis tool.
5) Assess and analyze risks.
6) Incorporate risk mitigation / avoidance approaches in the Project Plan.
7) Maintain, monitor, and update a detailed project risk profile.

Most people assume that IT project managers have advanced technical knowledge. Technical prowess, however, is not the main factor in project management success. Particularly, IT projects typically fail unless they are led by experts with a sufficient balance of business and technical communication abilities. An integrated risk management method is established, used, and maintained by professional project managers. They should make sure that every discipline needed to support the system's life cycle is included in their integrated risk management approach. The project team, which documents and puts into practice the risk management methods, supports them.

Using established risk assessment criteria, the team members evaluate (identify and analyze) hazards and their underlying causes. It is strongly advised to conduct ongoing/continual risk assessments, which are helpful at every stage of a program's life cycle. For each of the pertinent technical reviews and each crucial program decision point, a customised program risk assessment should be done. As part of their reporting and recommendations, project team members also estimate the funds needed to undertake risk mitigation plans with more documentation and knowledge sharing.

VI. AUTOMATED TOOLS FOR RISK MANAGEMENT

Finding computer-based tools with high accuracy probabilities to aid managers in decision-making is crucial if high-quality software products are to be delivered to the market on schedule and in compliance with market specifications. Data analysis and data mining can be used to some extent in software risk management and analysis. Automated tools are made to help project managers with project planning and setup, resource assignment, task tracking, management of budgets, requirements, changes, and hazards, as well as workload analysis.

LAN or the internet are typically used for team collaboration. Today's project management technologies can automatically save all project outcomes in a single repository that is accessible to all users.

Editing, defining, and prioritizing requirements and adjustments are all possible. Requirements that are traceable across their full life cycle are used to build tasks. Software designed for risk management may also have functionality for project quality control and test management. Project managers can assign resources to tasks even in a multi-project context with the aid of specialized views and customized reports. Most of risk management software supports the core risk methodologies, such as CMMI, SPICE, PRINCE2, COBiT etc. For example, the Software Tools Complex for Evaluation of Information Systems Operation Quality (CEISOQ) is designed with accordance to standard ISO/IEC 15288 and uses to evaluate probabilities of "success", cost, time and quality risks and related profitability and expenses. This assists in resolving the following practical issues in the system life cycle on a scientific basis: analysis of quality management systems for enterprises; justification of quantitative system requirements to hardware, software, users, staff, and technologies; requirements analysis; evaluation of project engineering decisions; investigation of issues regarding potential threats to system operation, such as information security and anti-terrorist protection; evaluation.

The majority of the time, risk analysis and management are based on data gathered from conventional knowledge, analogies to well-known situations, common sense evaluation, findings from experiments or tests, and reviews of unintentional exposure. The automated technologies must first gather historical data in order to create a database. Once the database is established, it will evaluate the information and extract some helpful data to aid managers in assessing risks and making decisions. There are numerous approaches to studying machine learning. For instance, clustering abilities are utilized to identify various dangers with a risk category. Risks within each cluster could share characteristics. Each cluster is examined using the association rule approach to determine how risks and risk variables are related. Building risk assessment models and predicting risks of software development are two further applications of artificial intelligence (9K-near neighbor technique, ID3 decision tree, Neuro-Network, etc.).

There is a wide variety of widely used decision-making software available on the market that, albeit with some restrictions, can also be used for risk management in software risk analysis. Successful decision-making software used in the business industry includes Microsoft SQL Server, Crystal Decisions, and Microsoft OLAP/Analysis Services. Although distinct from the banking and trading industries, software development risks can be seen from a variety of angles, making it a hybrid industry. Chartis, the leading research and advisory services firm focused exclusively on the risk technology market annually names the top risk management solution providers. Among specialized risk management software the most popular are: Acuity Risk Management LLP, Risk Watch, IBM Rational Portfolio Manager, OCTAVE-S, CRAMM, Citicus ONE, SCIENTECH

VII. CONCLUSION AND RECOMMENDATIONS

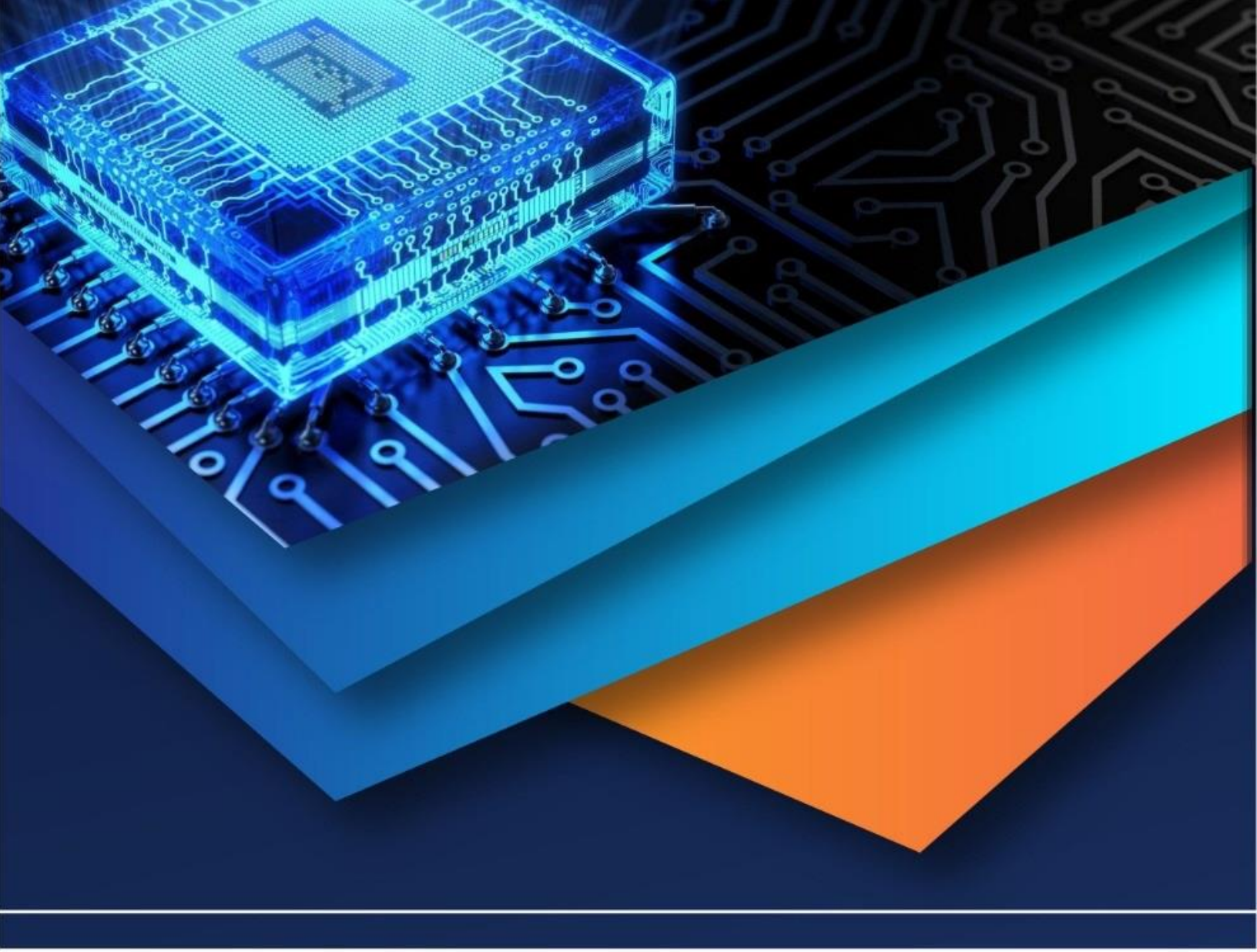
The usage of several artificial intelligence algorithms in the area of software defect prediction is reviewed in the literature in this essay. This paper's significant contribution is the improvement of our understanding of the field of study provided by the literature review. From the review of the literature, it can be inferred that this topic is quite interesting to researchers, and there are a ton of algorithms available for software defect prediction-based coding in the field of software engineering. We looked at a number of studies to assess the various approaches used in this decade to anticipate software defects perfectly. So, we draw the conclusion that there is a huge potential for applying fuzzy logic and neural network based techniques for software fault prediction in future study. Although many algorithms, including Support Vector Logic Regression, neural networks, boosting, and others, have been examined for defect prediction, none of them have been shown to provide accuracy greater than 88 percent. Therefore, the field of defect prediction has a lot of potential. Future work on developing more dependable defect prediction algorithms may take into account advanced hybrid mechanisms like fuzzy artmaps.

An IT project is frequently thought to be unreliable, overbudget, and behind schedule. This occurs as a result of the complicated nature of software development and implementation, which involves numerous parties with varying expectations. With so many interconnected parts and revisions in a typical IT project, delays in one part can easily effect the entire project. Risk management does not ensure success, but instead focuses on anticipating and addressing potential problems before they become crises, making it possible to complete projects that meet their objectives and instilling greater trust in IT among users and managers. Project managers may deal with risk management programs in the most effective and efficient way possible with the help of the suggested risk management tools and methodologies. It is obvious that every software project is different and requires modification and customization to its actual execution. In order to identify risk related to project poor quality, organizational failures, constant requirements change, contractors management, and many others that are directly influence the project performance and success, the author of this article advises paying more attention to the key project management areas such as contract management, requirements management, quality control, HR, etc.

While identifying the important risk categories and groups, it may also be helpful to investigate the project success elements. A methodology-based expert's even part-time assistance would significantly boost the likelihood that the project would be completed on time, as planned, and to the satisfaction of the client.

REFERENCES

[1] Martin Shepperd, Qinbao Song Zhongbin Sun Carolyn Mair "Data Quality: Some Comments on the NASA Software Defect Data Sets" IEEE Transaction on Software Engineering, Vol. 28, Oct 2012.
[2] Haimes Y.Y., Kaplan S., Lambert J.H. Risk filtering, ranking and management framework using hierarchical holographic modeling. Risk Analysis. 2002, 22(2): 381-395.
[3] Pipattanapiwong, J. Development of Multi-party Risk and Uncertainty Management Process for An Infrastructure Project. PhD Thesis, Kochi University of Technology. Kochi, Japan. 2004.
[4] R.Chidamer and Kemerer, "A Metrics Suit for Object Oriented Design" IEEE Transaction Software Engineering, vol.20, no.6, June 1994.
[5] Shanthini. A, Chandrasekaran.RM, "Applying Machine Learning for Fault Prediction Using Software Metrics", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 6, June 2012.
[6] C. Akalya Devi, K. E. Kannammal and B. Surendiran, "A Hybrid Feature Selection Model for Software Fault Prediction", International Journal on Computational Sciences & Applications (IJCSA) Vo2, No.2, April 2012.
[7] Kwak Y.A. Stoddard J. Project risk management: lessons learned from software development environment. Technovation, 2003, 24: 915-920.
[8] Anuradha Chug, Shafali Dhall "Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm", International Journal of computer science and engineering Vol.27, no.5 2010.
[9] Venkata U.B. Challagulla, Farokh B. Bastani, I-Ling Yen, Raymond A. Paul, "Empirical Assessment of Machine Learning based Software Defect Prediction Techniques", IEEE, 2005.
[10] Software project risk management Technical Report, by DET NORSKE VERITAS, 1999.
[11] PMI (Project Management Institute). A Guide to the Project Management Body of Knowledge (PMBoK). Newtown Square. Pennsylvania, USA. 2004
[12] Ahmet Okutan Olcay Taner Yıldız, "Software defect prediction using Bayesian networks", Empirical Software Eng (2014) 19:154–181 © Springer Science+Business Media, LLC, 2012.
[13] Del Cano A., De La Cruz M.P. Integrated methodology for project risk management. Journal of Construction Engineering and Management. 2002, 128(6): 473-485.
[14] Conrow E.H. Effective Risk Management: Some Keys to Success, 2nd Edition. American Institute of Aeronautics and Astronautics. Reston, USA. 2003
[15] A.Gunes Koru, "An Investigation of the Effect of Module Size on Defect Prediction Using Static Measures" ACM SIGSOFT Software Engineering Notes, 2005.
[16] Norman Fenton, N.E. and Neil, M. (1999), "A Critique of Software Defect Prediction Models", IEEE Transactions on Software Engineering, vol.25, no. 5, pp. 675-689.
[17] Avdoshin S., Pesotskaya E., Business informatization. Managing risks, Moscow: DMK Press, 2011, 176 p. [in Russian].
[18] Chapman C.B., Ward, S.C. Project Risk Management, Processes, Techniques and Insights, 2nd Edition. John Wiley. Chichester, UK. 2003.
[19] Boehm B., Software Risk Management: Principles and Practices, IEEE Software, 1991.
[20] SEI Continuous Risk Management HandBook, http://www.sei.cmu.edu /programs/sepm/risk/risk.mgmt.overview.html
[21] Cooper D. Tutorial Notes: The Australian and New Zealand Standard on Risk Management (AS/NZS 460). Retrieved: may 2004 from http://www.broadleaf.com.
[22] Xiaomeng Lian, Software Project Management – Risk Management (Abstraction),http://www.docstoc.com/docs/24840578/Software-Project-Management.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY