JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

SIGNIFICANCE OF MACHINE LEARNING IN RISK IMPACT ANALYSIS FOR AGILE SOFTWARE ENTERPRISES

[1]M. Ayesha Ziana, [2]Dr. Charles J

[1]Research Scholar, [2]Associate Professor

[1]Department of Computer Science, Noorul Islam Centre for Higher Education, Kumaracoil, Thuckalay, Tamil Nadu, India.

[2]Department of Software Engineering, Noorul Islam Centre for Higher Education, Kumaracoil, Thuckalay, Tamil Nadu, India.

Abstract : Agile software development methodology has encouraged the application of various tools that mitigate the risks of the software project as much as possible. In spite of this, the risk analysis process in Agile software projects usually follows an implicit path without the use of explicit and structured methods. In short, explicit risk analysis methodologies are scarce in Agile software development. To perform a structured risk mitigation activity, it is essential to analyze the risks. The application of machine learning to risk assessment activities is growing. This motivated us to perform this study, which illustrates the significance of machine learning in the risk assessment process, particularly in the risk analysis phase. This study, apart from exhibiting the results of the implemented machine learning algorithms in the literature, serves as a starting point in developing our own risk analysis model by predicting the impacts of the risks identified, in terms of project performance goals and the level of negative impacts created. A survey has been planned as a continuation of this process in order to finalize the dataset with the required features to be given as inputs for our machine learning model in the future.

IndexTerms **– Agile Software Risk Analysis, Machine Learning for Software Risk Management, Risk Impact Analysis, Text classification algorithms.**

I. INTRODUCTION

Project management is a series of phases carried out in which the project will produce the expected outcome in a simple and cost-effective way. According to the famous Project Management Body Of Knowledge (PMBOK) standard, risk management is one of the most vital areas of assistance in project management. Estimating risks and the relationship between them is one of the nine phases of risk management as proposed by Chapman and Ward. The belief that a project is filled with uncertainties strengthens the fact that a project can fail in connection with its goals, profits, cost, and schedule. A structured approach to risk management is encompassed by the identification and classification, analysis, and mitigation of risks. The risk assessment is one of the two-stage processes involved in risk management, and includes the identification, analysis, and prioritization of risks (Barghi and Sikari, 2020). Risk analysis is a process that assists in learning about the nature and level of risks (as demonstrated by the risks' effects and causes). In addition, it acts as an input for the process of making decisions towards effective risk handling (Oehmen et al., 2020). Agile Software Development concentrates on delivering a working software prototype to the customer on a schedule basis to ensure the generation of a quality product. Better risk analysis has been discovered and reported as one of the vital success factors of an Agile Software Development project in a Systematic Literature Review conducted by Sinha, Shameem and Kumar (2020). Risk prediction can be carried out in two different ways for software projects: (1) by predicting the risk level of the individual risks of a project as high, low, medium, and so on, thus helping to distinguish between different levels of risk; (2) by predicting if a project is risky or not using a classifier, thus helping to identify risky projects at an early stage (Han, 2015).

II. BACKGROUND AND RELATED WORK

The application of machine learning algorithms in the risk management process has gained tremendous popularity in recent years among software companies (Sousa, Faria and Mendes-Moreira, 2021).

The study by Han (2015) uses a three-layered Neural Network (Input, Hidden, and Output layers) with a back propagation algorithm to increase the accuracy of prediction in order to identify if a project is risky or not. The model with an accuracy of 100% outperformed the previous Logistic Regression model whose accuracy was 87.5%.

The study by Malhotra (2014) provides a comparison between statistical and machine learning models in the prediction of software faults. Six machine learning models, including Decision Tree, Artificial Neural Network, and Support Vector Machine were used for comparison. Their findings state that the decision tree algorithm was better at prediction than the logistic regression model. The applied risk dimensions were cost, risk prioritization, and cost-risk analysis. Integrating GSRM with the requirements engineering process of a project helps identify and mitigate the risks in software development in a structured way right from the beginning (Islam**,** Mouratidis and Weippl, 2014). As indicated by Khan, Mirza, and Saleem (2020), 100% accuracy was achieved in classification with the Neural Network model and 87.5% by using logistic regression.

A novel model for classification of risks based on relational association rule mining was proposed by Czibula, Marian, and Czibula (2014), which outperformed several machine learning models in the task of software fault prediction. A two-stage method of classification was introduced by Li, Huang, and Li (2016), which provided a better solution for predicting software defects in a cost-sensitive fashion. The methodology applied a two-stage method of ranking on three-way decisions and obtained better accuracy at a lower cost. The experimental results by Calp and Akcayol (2019) illustrate the effectiveness of using Artificial Neural Network in the process of risk estimation. Around 45 risk factors from 20 projects were used as inputs for the model, and the derived outputs were variations in the project schedule, cost, success of the project, and so on. The model showed high performance with 45 inputs of risk factors, 15 hidden layer neurons, and 5 outputs, eventually producing a higher accuracy rate with an error rate almost as low as zero.

In the work of Hu et al., (2009), two machine learning algorithms Neural Network and Support Vector Machine were used in building a software risk prediction model, and the experiment results convey that SVM performed better with an accuracy of 85%, while the neural network model obtained an accuracy of 75%. The prediction of projects was based on three categories: successful, challenged, and failed.

The Support Vector Machine based software risk classification model by Zavvar et al., (2017) outperformed the Self-Organizing Map (SOM) and K-Means algorithms in terms of accuracy and Area Under Curve (AUC) metrics with percentage values of 99.51% and 98%, respectively. The model was useful in classifying the risks of a project as either low risk or high risk.

The software risk factors prediction model by Christiansen et al., (2015) applied multiple logistic regression in order to classify the project characteristics as either risky, or non-risky, with an accuracy rate of 90%. The inputs were collected from the questionnaires for 70 software projects.

The goal-based risk analysis methodology by Bhukya and S. Pabboju (2019) handles the risks occurring as a result of certain events with a few plans of action, like analyzing the cost of the solutions produced by the candidates, prioritizing the risks to discriminate the higher priority risks from the lower priority ones, and analyzing the cost and risk so as to give a clear picture of the budget and the risks associated with each of the solutions provided.

The risk analysis model by Hu et al., (2013) concentrates on the process of analyzing risks based on historical data and applies the random forest algorithm, believing that it is more precise than the decision trees.

From the above evidence, we find that there are research works that classify the risks based on the impact level and classify the projects as risky or not risky. However, as per our knowledge, we did not find any work that classifies the risks based on the project performance goals affected by those risks. We identified this as the research gap and will be proceeding further towards developing a better project goal impact prediction model along with the prediction of the level of impacts.

III. PROBLEM IDENTIFICATION

Quantitative risk models require massive amounts of data to implement statistical functions, and this is a great impediment to the success of hybrid approaches. Also, this requirement is hard to execute in Agile for two main reasons. Initially, Agile prefers as little documentation as possible, which is carried out just to preserve continuity and aid in maintenance. Secondly, Agile sprints are not dependent on each other, and so the data of one sprint may not be compatible with that of another sprint, and shorter sprints may produce less data, which may be insufficient for quantitative risk models (Anes, Abreu and Santos, 2020). However, from the recent research, we can see that machine learning has been started for use in Agile Software Development, knowing its immense benefits. For instance, an unsupervised machine learning model was applied in the task allocation process of distributed agile projects (Singh, Chauhan and Popli, 2022) and the supervised machine learning Naive Bayes algorithm was implemented in the effort estimation procedure (Ratke et al., 2019), where a reasonable accuracy of 83% was obtained.

Each risk analysis methodology comprises unique benefits. Regression analysis manifests the variable dependencies and assists in predicting the risks. Association rules can be used to find out which rules fulfil the minimum support and the counting-based confidence aforementioned by the user. Decision trees are uncomplicated and cushy. Neural networks are able to record the existing non-linear interdependencies among the variables. By applying fuzzy logic, one can bundle the individual scores of project risk factors into an overall risk score for the project using fuzzy set theory, which aids in imprecise risk assessment. Clustering analysis clumps subsets of observations based on mutual surveillance in the absence of predefined categories. However, it is unfortunate that none of these methodologies is capable of discriminating causality and correlation, though they can produce certain cause-effect relationships unknowingly (Hu et al., 2013).

Szwaczyk, Wrona and Amanowicz (2018) deal with the 2 major classifications of risk analysis methodologies, namely, Data-driven methods and Dependency-modeling methods. The Data-driven methods are, Statistical analysis (SA) which includes correlation and regression, which are useful in future prediction, and Pattern analysis and neural networks (PA&NN) which predict the future via machine learning and Artificial Neural Network (ANN) methodologies. The data-driven methodologies encompass higher performance and scalability features in addition to the ability to predict individual risks.

The systematic risk analysis methodologies can be classified into two groups, namely, the classic and conceptual models. The probabilistic approaches demand a large volume of data, which is quite impossible during the planning phase. They also face difficulties while implementing a real-time project due to the challenge of making accurate decisions in the absence of clear definitions of the problems, which are vague. Hence, subjective assessment is the most suitable way for situations in which the classic models are not applicable (Akhavan, Sebt and Ameli, 2021). Despite this, we can apply machine learning models by choosing the best algorithm that suits the limited amount of data generated.

The main purpose of risk classification is to obtain a united perspective regarding a group of factors, which in turn can assist the practitioners in discovering the group with the maximum risk. Risk classification is an inexpensive methodology for risk analysis along with their root causes by categorizing risks of similar nature into a class (Hoodat and Rashidi, 2009).

Choosing an appropriate method for automatic text analysis is a big challenge, which prevents further machine learning proliferation. Maintaining the balance of required objectives while selecting the best approaches is hard to attain. In order to maximize the accuracy of the classification process, it is suggested to perform a thorough examination of all the existing approaches to find out the best solution for every task. Recent studies in computer science reveal the suggestion that Artificial Neural Network (ANN) and Na¨ıve Bayes (NB) are found to be the best solutions for classifying text (Hartmann et al., 2019).

IV. PROPOSED WORK

Risk analysis takes into account the initiators and roots of risk, its repercussions, and their probability. The risk level can be computed by combining the probabilities and impacts of the risk (Bakhtavar et al., 2021). Discovering the most important risks may be a complicated task. Also, determining the likelihood and impact level for each and every risk may be subjective, making it tough to evaluate and gauge. Even though training is an important aspect, shortages of experts do exist and need to be minimized (Munte´s-Mulero et al., 2019).

Recently, the trend of using machine learning algorithms for assessing risks has become quite popular. Especially super- vised learning algorithms such as Decision Trees, Naive Bayes Classifiers, Neural Networks, and Support Vector Machines are commonly used. Analyzing factors such as reliability, availability, and quality is widely followed in the risk analysis phase of the risk management process. In spite of this, there exists some uncertainty in risk estimation, and hence a subjective approach to risk assessment is commonly preferred by practitioners (Sousa, Faria and Mendes-Moreira, 2021). However, we can experiment with machine learning models to obtain the desired outcome.

In the work of Marchwicka (2020), the risk analysis has formed an anchor point in observing and rescheduling software projects based on the three dimensions of the iron triangle: time, cost and quality. The Goal-driven Risk Management Framework (GSRM)- based study by Shrivastava and Rathod (2019) concentrated on the basic ‘iron triangle’ performance goals such as ‘Time’, and ‘Cost’ required to complete a project, in addition to the ‘Quality’ goal of the project, which is usually related to the satisfaction of the customer. Any risks that occur in a project can be a threat to one or more of these goals, thus leading to the downfall of the project. Hence, it is essential for project teams to have a clear understanding of the impact of the potential risks. The survey was conducted in two phases, the first of which revealed the impact of the 44 identified risk factors from the literature on the given project goals. In the second phase, the data collected from the first phase was analyzed to offer a rank based on its impact towards a specific goal of the project. The outcome of the first phase states that the risk factors were bound to attack the ‘Time’ and ‘Quality’ goals but not the ‘Cost’ goal. Keeping this in mind, we propose an additional attribute, which is ‘Employee satisfaction’. From the work of Chanda and Goyal (2020), we can find that the ‘Employee Satisfaction’ variable in terms of job nature, salary, and supervision is strongly connected to the profit growth, sales growth, and Return on Investment of an organization. So, our ML algorithm consists of three impact classes to predict, namely ‘Time’, ‘Quality’, and ‘Employee Satisfaction’. We are in search of other performance goals as well. In addition, the level of impact can also be predicted, namely ‘Low’, ‘Medium’ and ‘High’.

Inputs: Risk description, Risk Category, Risk Subcategory, and Risk Nature (the data set is obtained from our previous experimental research among three groups of participants)

Outputs: Impact goal and Impact level (to be find out using our proposed Machine Learning model in our future work)

As of now, no feature selection algorithm is going to be applied, and we will be dropping the Risk description and Risk Category columns if required since the Subcategory feature can be good enough for prediction. The steps of research methodology are depicted in Fig.1 and the process of building our own ML model is illustrated in Fig.2.

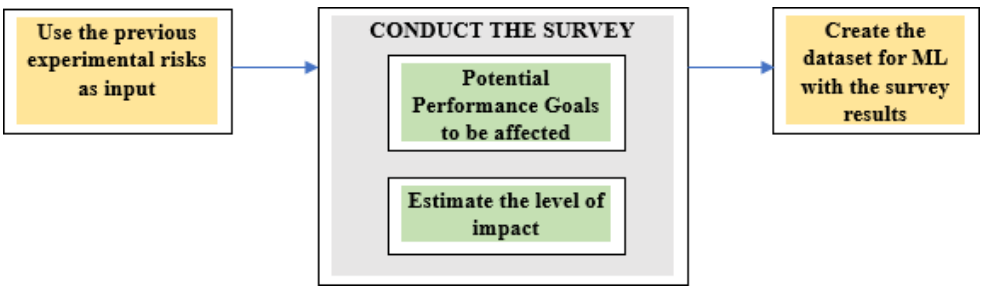

Fig. 1. Research Methodology

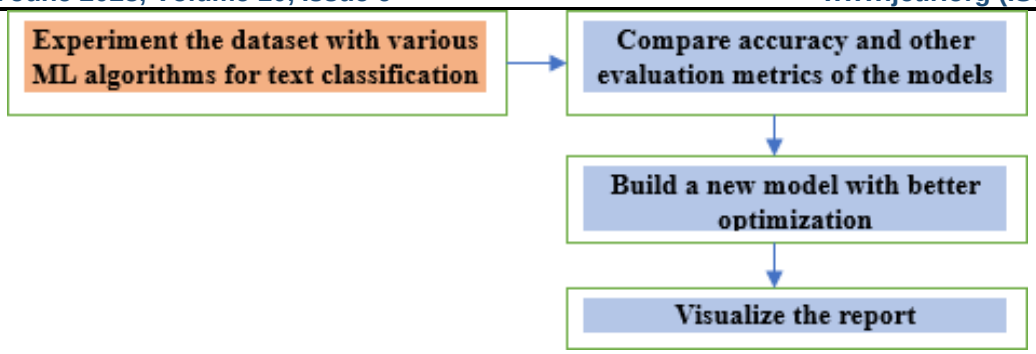

Fig. 2. Research Methodology

V. CONCLUSION

In this paper, we have discussed the significance of risk management, especially the risk analysis phase in the software development life cycle. Predetermination of the software risk consequences on the project performance and the level of consequences is beneficial for the agile teams in the managing and administration activities. The predictions will be effective when collaborating with the customer as well, since the customer can get a clear-cut representation of the consequences of the risks. Cause and consequences analysis of risks will be helpful in finding the sources and effects of risks. Our previous experiments on the risk identification phase were done based on the causal effects of risks, and hence we focused on the impact analysis of risks, especially with the project performance goals affected. In future work, we will conduct a survey based on this study and use the responses as part of the data set for our machine learning model, along with the outputs of our previous brainstorming session experiments. It is difficult to suggest a single classifier for the software risk classification and prediction problem. Simpler but more efficient algorithms needed to be discovered to provide better optimization.

REFERENCES

[1] B. Barghi, and S.S.Sikari. 2020. Qualitative and quantitative project risk assessment using a hybrid PMBOK model developed under uncertainty conditions. Heliyon, 6(1): e03097. doi: https://doi.org/10.1016/j.heliyon.2019.e03097.

[2] J. Oehmen, A. Guenther, J. W. Herrmann, J. Schulte, and P. Willumsen. 2020. Risk management in product development: risk identification, assessment, and mitigation - a literature review. Proceedings of the Design Society: DESIGN Conference, 1: 657–666, Cambridge University Press, May 2020. doi: https://doi.org/10.1017/dsd.2020.27.

[3] R. Sinha, M. Shameem, and C. Kumar. 2020. SWOT: strength,weaknesses, opportunities, and threats for scaling agile methods in global software development. Proceedings of the 13th innovations in software engineering conference on formerly known as India software engineering conference, 1-10.

[4] W. M. Han. 2015. Discriminating risky software project using neural networks. Computer Standards & Interfaces, 40:15–22. doi: http://dx.doi.org/10.1016/j.csi.2015.01.001.

[5] A. Sousa, J. P. Faria, and J. Mendes-Moreira. 2021. An analysis of the state of the art of machine learning for risk assessment in software projects. Proceedings of the 33rd International Conference on Software Engineering and Knowledge Engineering, 1–10, SEKE 2021. doi: 10.18293/SEKE2021-097.

[6] M. N. A. Khan, A. M. Mirza, and I. Saleem. 2020. Software risk analysis with the use of classification techniques: A Review. Engineering, Technology & Applied Science Research, 10(3): 5678–5682. doi: https://doi.org/10.48084/etasr.3440.

[7] R. Malhotra. 2014. Comparative analysis of statistical and machine learning methods for predicting faulty modules. Applied Soft Computing, 21: 286–297. doi: http://dx.doi.org/10.1016/j.asoc. 2014.03.032.

[8] S. Islam**,** H. Mouratidis, and E. R. Weippl. 2014. An empirical study on the implementation and evaluation of a goal-driven software development risk management model. Information and Software Technology, 56(2): 117–133. doi: http://dx.doi.org/10.1016/j.infsof.2013.06.003.

[9] G. Czibula, Z. Marian, and I. G. Czibula. 2014. Software defect prediction using relational association rule mining. Information Sciences, 264: 260–278. doi: http://dx.doi.org/10.1016/j.ins.2013.12.031.

[10] W. Li, Huang, and Q. Li. 2016. Three-way decisions-based software defect prediction. Knowledge-Based Systems, 91: 263–274. doi: http://dx.doi.org/10.1016/j.knosys.2015.09.035.

[11] M. H. Calp, and M. A. Akcayol. 2019. A novel model for risk estimation in software projects using Artificial Neural Network. In Artificial Intelligence and Applied Mathematics in Engineering Problems: Proceedings of the International Conference on Artificial Intelligence and Applied Mathematics in Engineering (ICAIAME 2019), 295–319. doi: https://doi.org/10.1007/978-3-030-36178-5_23.

[12] Y. Hu, X. Zhang, X. Sun, M. Liu, and J. Diu. 2009. An intelligent Model for software project risk prediction. In 2009 International Conference on Information Management, Innovation Management and Industrial Engineering, 1: 629–632, IEEE, 2009. doi: 10.1109/ICIII.2009.157.

[13] M. Zavvar, A. Yavari, S. M. Mirhassania, M. R. Nehi, A. Yanpi, and M. H. Zavvar. 2017. Classification of risk in software development projects using Support Vector Machine. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 9(1): 1–5. url: https://jtec.utem.edu.my/jtec/article/view/857.

[14] T. Christiansen, P. Wuttidittachotti, S. Prakancharoen, and S. A. O. Vallipakorn. 2015. Prediction of risk factors of software development project by using multiple logistic regression. ARPN Journal of Engineering and Applied Sciences, 10(3): 1324–1331.
[15] S. N. Bhukya, and S. Pabboju. 2019. Software engineering: risk features in requirement engineering. Cluster Computing, 22: 14789–14801. doi: https://doi.org/10.1007/s10586-018-2417-3.
[16] Y. Hu, J. Du, X. Zhang, X. Hao, E. W. T. Ngai, M. Fan, and M. Liu. 2013. An integrative framework for intelligent software project risk planning. Decision Support Systems, 55(4): 927–937. doi: http://dx.doi.org/10.1016/j.dss.2012.12.029.
[17] V. Anes, A. Abreu, and R. Santos. 2020. A new risk assessment approach for Agile projects. In 2020 International Young Engineers Forum (YEF-ECE), 67–72, IEEE. doi: https://doi.org/10.1109/ YEF-ECE49388.2020.9171808.
[18] Y. Hu, X. Zhang, E. W. T. Ngai, R. Cai, and M. Liu. 2013. Software project risk analysis using Bayesian networks with causality constraints. Decision Support Systems, 56: 439–449. doi: http://dx.doi.org/10.1016/j.dss.2012.11.001.
[19] S. Szwaczyk, K. Wrona, and M. Amanowicz. 2018. Applicability of risk analysis methods to risk-aware routing in software-defined networks. In 2018 International Conference on Military Communications and Information Systems (ICMCIS), 1–7, IEEE. doi: https://doi.org/10.1109/ICMCIS.2018.8398688.
[20] M. Akhavan, M. V. Sebt, and M. Ameli. 2021. Risk assessment modeling for knowledge based and startup projects based on feasibility studies: A Bayesian network approach. Knowledge-Based Systems, 222: 106992. doi: https://doi.org/10.1016/j.knosys.2021.106992.
[21] H. Hoodat, and H. Rashidi. 2009. Classification and analysis of risks in Software Engineering. International Journal of Computer and Information Engineering, 3(8): 2044-2050.
[22] J. Hartmann, J. Huppertz, C. Schamp, and M. Heitmann. 2019. Comparing automated text classification methods. International Journal of Research in Marketing, 36(1): 20-38. doi: https://doi. org/10.1016/j.ijresmar.2018.09.009.
[23] E. Bakhtavar, M. Valipour, S. Yousefi, R. Sadiq, and K. Hewage. 2021. Fuzzy cognitive maps in systems risk analysis: a comprehensive review. Complex & Intelligent Systems, 7: 621-637. doi: https://doi.org/10.1007/s40747-020-00228-2.
[24] V. Munte´s-Mulero, O. Ripolles, S. Gupta, J. Dominiak, E. Willeke, P. Matthews, and B. Somosko¨i. 2019. Agile risk management for multi-cloud software development. IET Software, 13(3): 172-181. doi: https://doi.org/10.48550/arXiv.2001.03356.
[25] E. Marchwicka. 2020. A technique for supporting decision process of global software project monitoring and rescheduling based on risk analysis. Journal of Decision Systems, 29(sup1):398-412. doi: https://doi.org/10.1080/12460125.2020.1790825
[26] S. V. Shrivastava, and U. Rathod. 2019. A Goal-driven risk management approach for Distributed Agile Development projects. Australasian Journal of Information Systems, 23. doi: https://doi.org/10.3127/ajis.v23i0.1843.
[27] U. Chanda, and P. Goyal. 2020. A Bayesian network model on the interlinkage between socially responsible HRM, employee satisfaction, employee commitment and organizational performance. Journal of Management Analytics, 7(1): 105-138. doi: https://doi.org/10.1080/23270012.2019.1650670.
[28] M. Singh, N. Chauhan, and R. Popli. 2022. Task allocation in Distributed Agile Software Development environment using unsupervised learning. Journal of Engineering Research, 10. doi: https://doi.org/ 10.36909/jer.ICMET.17167.
[29] C. Ratke, H.H. Hoffmann, T. Gaspar, and P.E. Floriani. 2019. Effort Estimation using Bayesian Networks for Agile Development. In 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS), 1-4, IEEE. doi: https://doi.org/10. 1109/CAIS.2019.8769455.