Software Risk Prediction: Systematic Literature Review on Machine Learning Techniques

Mahmudul Hoque Mahmud 1, Md. Tansirul Haque Nayan 1, Dewan Md. Nur Anjum Ashir 1 and Md Alamgir Kabir 2,*

1 Department of Computer Science, American International University-Bangladesh, 408/1, Kuratoli, Dhaka 1229, Bangladesh
2 Artificial Intelligence and Intelligent Systems Research Group, School of Innovation, Design and Engineering, Malardalen University, Hogskeleplan 1, 722 20 Vasteras, Sweden
* Correspondence: md.alamgir.kabir@mdh.se

Abstract: The Software Development Life Cycle (SDLC) includes the phases used to develop software. During the phases of the SDLC, unexpected risks might arise due to a lack of knowledge, control, and time. The consequences are severe if the risks are not addressed in the early phases of SDLC. This study aims to conduct a Systematic Literature Review (SLR) and acquire concise knowledge of Software Risk Prediction (SRP) from the published scientific articles from the year 2007 to 2022. Furthermore, we conducted a qualitative analysis of published articles on SRP. Some of the key findings include: (1) 16 articles are examined in this SLR to represent the outline of SRP; (2) Machine Learning (ML)-based detection models were extremely efficient and significant in terms of performance; (3) Very few research got excellent scores from quality analysis. As part of this SLR, we summarized and consolidated previously published SRP studies to discover the practices from prior research. This SLR will pave the way for further research in SRP and guide both researchers and practitioners.

Keywords: systematic literature review; software risk; software risk prediction model; machine learning model; review

1. Introduction

The process of software engineering is a systematic method to develop software [1–3]. It involves the development and maintenance of software. There is always the possibility of unexpected events occurring during the Software Development Life Cycle (SDLC) that may result in loss or failure in software development [4–6]. Furthermore, the unknown circumstance refers to the software risk [7]. Numerous software risks could be generated due to incomplete and unclear requirements [8,9]. In addition, the software risk prediction function is the most sensitive and crucial in the SDLC, and must be performed flawlessly [10,11]. Risk management is a critical step in software engineering that must be followed for the project to be successful [12–14]. Furthermore, all phases of SDLC can potentially introduce software risk [15]. Regardless of how much work we devote to ensuring the success of software projects, many software projects have an unusually high rate of risk [16,17]. SDLC involves all the factors that can occur risks (e.g., cost, schedule, and quality) [18,19]. Factors should not be neglected, even if a single element has the potential to have a significant impact on the whole software development process [20]. As a result, an effective risk management model should be able to identify risks and evaluate how they evolve as the project proceeds. [21]. However, without risk management, significant risks may probably be overlooked [22]. For this reason, risk analysis is significant in the SDLC, where risks are identified and necessary steps are taken [23–25]. Moreover, it is vital to take the necessary precautions to prevent project failures due to the aggravated complexity of modern software systems [26–28]. If all of these risks are not properly identified, these could be the
reason for the project’s failure [29–31]. Risk assessment should be a continuous process in SDLC [32]. If these issues can be resolved in the early phases of software development, both effort and cost may be reduced [33].

In this study, the objective is to conduct a comprehensive literature review on the need and purpose of Software Risk Prediction Models (SRPM). We believe that this Systematic Literature Review (SLR) will provide a critical view of SRPM research. In this SLR, we selected and analyzed 16 SRPM articles published from the year 2007 to 2022. In addition, we classified the articles based on their publication details and investigated further from several points of view. To the best of our knowledge, this SLR is one of the initial efforts for the review of SRP studies. Our findings in this SLR include:

- Identified and extracted the features (e.g., type and size of datasets, data analysis approaches, techniques for detection, performance metric, and proposed ideas) from primary studies (PS) linked to SRPM research.
- Provided an overview of the current status of SRPM research.
- Conducted an SLR and quality analysis of the SLR from the published articles (publication year: 2007 to 2022).

The rest of the paper is organized as follows: Sections 2, 3, and 4 describe the method, study selection and criteria for quality assessment, respectively. Section 5 is the most vital section that provides the results and commentary of the investigation. Finally, Section 6 concludes this review and provides recommendations based on our findings.

2. Review Method

The objective of this study is to an SLR to provide an overview of SRPM research. The procedures for performing this SLR are designed in accordance with basics established by Kitchenham [34,35]. Malhotra [36] and Son et al. [37] had the most effect on the review design, questions and data that are presented in this section, and therefore, their findings. As recommended by Kitchenham, we conducted this review in the following stages: preparing the review, doing it and reporting our results and conclusions. Figure 1 depicts in detail the unique stages that must be completed. The first step in the planning phase is to determine whether or not a systematic review of the literature is required. As stated in the prior section, the purpose of this SLR is to identify and evaluate the preceding in step 1. We devised the evaluation process to ensure the validity of the study and to eliminate research bias from the findings. The second step of Figure 1 mentions the major measures that were taken to conduct this review study in greater detail. In the first stage of the SLR, we presented research questions that would be used to solve the issues that will be encountered (step 4). An automated search technique was then devised and used in the digital library to acquire the Primary Studies data. The following phase is the study selection strategy, in which we perform an inclusion-exclusion analysis to identify which papers to include. Afterward, a questionnaire for quality assessment was used to explore and evaluate the overall quality of each of the research studies conducted. Last but not least, we gathered information from each study. The processes for conducting the SLR are delivered in the greater description in the following subsections.

2.1. Research Questions

We determined eleven research questions for the SLR to determine the findings gained from the SRPM. Table 1 shows the research questions that were created systematically. RQ-1 investigates the needs and objectives of SRPM research. RQ-2, RQ-3, and RQ-4 provide an analyzed report on the datasets and methods of analyzing the data employed in the publications. We evaluated several detecting approaches considered in earlier studies in RQ-5, RQ-6, and RQ-7. In the following inquiry, we looked at the most often-used performance metrics in the field of SRPM. RQ-9 evaluates the performance of the researchers proposed in the SRPM models, which also include the values of the performance measures. RQ10 investigates the research emphasis chosen in primary investigations. The final question
discusses the challenges and limitations that the researchers have identified in the chosen primary investigation.

![Image](https://via.placeholder.com/150)

Figure 1. Review method.

Table 1. Research questions.

| Q#  | Research Questions                                                                 | Motivation                                                                 |
|-----|------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| RQ01| What are the purposes and reasoning for SRPM research?                              | Investigate the SRPM’s citation-based analyses.                            |
| RQ02| What is the average number of SRPM studies each year?                               | Estimate the number of articles published on SRPM every year.              |
| RQ03| What kinds of data sets are used in the detecting process?                          | Determine which datasets are utilized in SRPM.                             |
| RQ04| What is the size of the data sets?                                                  | Examine the research’s credibility of the study to see if it is valid.     |
| RQ05| What data analysis approaches are utilized to develop SRPM models?                 | The data analysis approaches used to construct SRPM are listed.            |
| RQ06| What detecting techniques are employed in the development of SRPM models?           | The detection strategies utilized to create SRPM models are identified.    |
| RQ07| How many of the SRPM papers employ the Machine Learning approach?                   | Determine how the ML approach is used in SRPM.                            |
| RQ08| What are the various performance metrics used in SRPM studies?                      | Determine the performance parameters used for assessing the SRPM.          |
| RQ09| What is the efficiency of the SRPM that has been proposed?                         | Evaluated the performance of the SRPM that has been proposed.             |
| RQ10| What is the main research emphasis of the papers?                                   | Determine the particular aspects of the research papers.                   |
| RQ11| What are the limits and problems of the SRPM highlighted in the studies?            | Determine the limitations and challenges raised in the primary study.       |

2.2. Search Strategy

When it is necessary to incorporate all relevant articles on a certain topic, the search phase is the most significant stage of the process. Keyword searches have been taken into consideration because they are a common method of searching for items in electronic databases. In the first phase, we chose digital libraries for our research. Once we had the keywords acquired from relative article titles, abstracts, and keywords, which contained equivalent phrases and equivalents, we used Boolean “AND” or “OR” expressions to pick the most relevant articles to include in the database. Our final search string included the following keywords:
1. “Software Risk” AND (“Prediction” OR “Detection’) AND “Machine Learning”
2. “Software Risk” AND (“Prediction” OR “Detection’)
3. “Software Risk” AND (“Prediction” OR “Detection’) AND “Systematic Literature Review’

In the next steps, the search string was modified to fit the unique requirements of each database, which was done in the previous step. We searched each database using the titles, abstracts, and keywords that were provided. It is important to mention that the search for this study’s findings was undertaken in line with the study’s publication date. The collection included journal articles and conference proceedings written in English.

3. Study Selection

In Figure 2, the search approach produced a preliminary set of 26 Primary Studies (PS). However, the list may include some research that is irrelevant to the SLR or does not fit within the study’s objectives. To exclude studies that do not meet the SLR’s objectives, we have to develop inclusion-exclusion criteria prior to the early assessments.

Figure 2. Process for searching and selecting studies.

Criteria for inclusion:

• Select the studies that focuses on Software Risk Prediction.
• Detailed information on risk prediction, such as test–train samples and prediction rate, must be included in the research result report of the study.

Criteria for exclusion:

• Studies in which the SRPM was not treated as the main topic.
• Studies which are not published in English.
• Studies that lacked empirical analysis or clarity on the experimental results.
According to Figure 2, articles are rejected based on their title, abstract, and full text. A list of 19 articles was compiled from the abstracts and titles of 26 investigations. After analyzing the whole text of these 19 publications, the list is narrowed to a final 16 Primary Studies. To summarize, the final studies were acquired using the quality evaluation criteria listed in the following section.

4. Criteria for Quality Assessment

We prepared some questions to assess the overall quality of the PS we had selected. The recommendations of Sohan and Basalamah [38] were taken into account when developing this difficult task. There are a total of 11 quality evaluation questions in Table 2, which were applied to a total of 19 articles in the study. Each question has three possible answers: “Yes” (1 points), “Partly” (0.5 points), or “No” (0 points). A ‘Yes’ (1 points) response indicates that the researcher was completely in agreement with the question for the paper, a ‘Partly’ (0.5 points) response indicates that the researcher somewhat agreed, and a ‘No’ (0 points) response indicates that the researcher was fully resistant to the question for the article. The ultimate score is determined by combining 209 (19 × 11) question-answer matches. Each item may have a maximum score of 11 points and a minimum score of 0.

Table 2. Questions for Quality Analysis.

| QQ#  | Quality Assessment Questions                                      |
|------|------------------------------------------------------------------|
| QQ01 | Is it apparent what the study’s main objective is?              |
| QQ02 | Is the sample of the dataset large enough to support this sort of investigation? |
| QQ03 | Whether the procedure for data collection is adequately specified and documented? |
| QQ04 | Is there enough information on the experiment provided by the author? |
| QQ05 | Are the potential risks to the validity of the claim described? |
| QQ06 | Are the study’s limitations discussed in detail?                |
| QQ07 | Is there a clear definition of the performance parameters employed in the study? |
| QQ08 | How well-defined are the learning strategies?                   |
| QQ09 | Are the results presented in a clear and concise manner?        |
| QQ10 | Is there a way to compare and contrast different techniques?    |
| QQ11 | Are there any new insights from this study that can be added to the existing literature? |

5. Results and Discussions

This section contains information on primary studies and the findings of the studies that were conducted in response to the research questions. An overview of the research and their citations were provided in this section. After that, we delivered the answer to each question, along with the relevant discussion and interpretation.

5.1. Description of Primary Studies (PS)

An SLR inquiry in the realm of SRPM is new to us to the extent that we are aware. The 16 PS were rated according to a variety of selection criteria. Each study has its unique identification and reference number, which are listed in Table 3.

Partially oriented studies were discarded in favor of the 16 articles devoted exclusively to SRPM research. Primary studies detection models are described in the following lines, including study methods, findings, and their effectiveness:

Kumar and Yadav [18]’s Bayesian Belief Network (BBN)-based probabilistic software risk estimate model, which focuses on the most significant software risk indicators, was developed to be used for risk assessment in software development projects. An empirical experiment was carried out to evaluate the model that had been built, using data obtained from software development projects that were used by the business.
Table 3. Selected PS with references.

| Study No. | Authors and Title | Ref No. |
|-----------|-------------------|---------|
| PS01      | Cost-sensitive and ensemble-based prediction model for outsourced software project risk prediction—Yong Hu et al. on 2015 | [39] |
| PS02      | Software Risk Estimation Through Bug Reports Analysis and Bug-fix Time Predictions—Hussain Mahfoodh, Qasem Obiediat on 2020 | [40] |
| PS03      | A Dataset for Software Requirements Risk Prediction—Zain Shaukat et al. on 2018 | [7] |
| PS04      | Risk management in the software life cycle: A systematic literature review—Jhon Masso et al. on 2020 | [41] |
| PS05      | Software project risk analysis using Bayesian networks with causality constraints—Yong Hu et al. on 2013 | [42] |
| PS06      | Prediction of Risk Percentage in Software Projects by Training Machine Learning Classifiers—Gouthaman P, Suresh Sankaranarayanan on 2021 | [43] |
| PS07      | An integrative framework for intelligent software project risk planning—Yong Hu et al. on 2013 | [44] |
| PS08      | Software Risk Prediction Based on the Hybrid Algorithm of Genetic Algorithm and Decision Tree—Zhihong Xu et al. on 2007 | [45] |
| PS09      | Prediction of project problem effects on software risk factors—M. Özgür Cingiz et al. on 2013 | [46] |
| PS10      | Prioritizing Software Components Risk: Towards a Machine Learning-based Approach—Mrwan BenIdris et al. on 2020 | [47] |
| PS11      | An integrated approach using IF-TOPSIS, fuzzy DEMATEL, and enhanced CSA optimized ANFIS for software risk prediction—K. Suresh, R. Dillibabu on 2021 | [48] |
| PS12      | Research on Software Project Risk Assessment Model based on Fuzzy Theory and Improved—Zhenyu Yu et al. on 2017 | [49] |
| PS13      | Design and Development of Machine Learning Technique for Software Project Risk Assessment—A Review—Mohamed Najah Mahdi et al. on 2020 | [50] |
| PS14      | A Risk Assessment Method based on Software Behavior—Guorong Chen et al. on 2019 | [51] |
| PS15      | A Probabilistic Software Risk Assessment and Estimation Model for Software Projects—Chandan Kumar, Dilip Kumar Yadav on 2015 | [18] |
| PS16      | Risk Group Prediction of Software Projects Using Machine Learning Algorithm—Asim Kerem Hanci on 2021 | [52] |

Hu et al. [39] reviewed similar work from the previous two decades and discovered that all available models for prediction assume equal misclassification costs, by neglecting the effects of real-world events in the software project management industry. Indeed, failing to recognize project failure is even more dangerous than incorrectly labeling a project with a high possibility of success as a failure, which is much more common. Furthermore, ensemble learning, which is a well-established technique for improving prediction performance in other areas, has not been substantially examined in the context of software project risk prediction. Their research aimed to fill knowledge gaps in the field by investigating cost-sensitive analysis and classifier ensemble approaches, among other things, both of which were investigated. Using 327 outsourced software project examples, a T-test comparison of 60 alternative risk prediction models revealed that the optimal model is a homogenized set of decision trees (DT) adopting bagging. The findings of the proposed framework reveal that while DT beat Support Vector Machine (SVM) in terms of accuracy (i.e., assuming equal misclassification costs), it outperformed SVM in terms of cost-sensitive analysis. For a quick overview, this paper proposes the first cost-sensitive and ensemble-based hybrid modeling approach for predicting the risk associated with software development projects. A cost-of-misclassification evaluation criterion was also created to evaluate software risk and prediction models.

According to Hu et al. [44], software project risk assessment and planning have no empirical models. The researchers developed an integrated framework for intelligent software project risk planning to help reduce project risks and increase predictability (IF-
ISPRP). IF-ISPRP consists of two fundamental elements: the risk analysis module and the risk planning module. The risk analysis module predicts project success. It creates a cost-effective set of risk control activities from the risk analysis results. They suggested a breakthrough MMAKD approach for complex risk planning. They also utilized the framework to decrease project risk in Guangzhou Wireless City, a social media platform. Other social software projects might help from the risk-management practices discussed here. They believed that integrating risk analysis and planning would help project stakeholders manage project risks.

As part of their effort to describe and explain the present level of knowledge on this topic, Masso et al. [41] conducted a comprehensive review of the literature on software risk to identify any gaps or areas that may need future investigation. The findings of their SLR revealed that the scientific community’s emphasis had migrated away from the concept of research effort addressing an integrated risk management process and toward work concentrating on specific activities within this process, according to their analysis of the data. It was also feasible to observe an obvious lack of scientific integrity in the validation procedures of the various studies, as well as a weakness in the use of standards or de facto models to characterize the results of these.

In the paper of Hu et al. [42], a new model for risk analysis of software development projects based on Bayesian networks with causality restrictions were proposed by the authors (BNCC). They showed that when they applied unrestricted automatic causality learning to 302 collected software project data, the proposed model not only discovered causal relationships consistent with expert knowledge, but also outperformed other algorithms such as logistic regression, Naive Bayes, and general BNs in prediction. BNCC is being used in their study to establish the first causal discovery framework for assessing the risk causality of software projects, as well as a model for managing software project risk based on this framework.

BenIdris et al. [47] proposed an alternative model for software development project risk analysis that is based on BNs with causality constraints (BNCC). They proved that, when combined with expert information, the suggested model is not only capable of detecting causal relationships congruent with expert knowledge, but also outperforms other algorithms such as logistic regression, Naive Bayes, and generic BNs in terms of prediction performance. They established the first framework for studying the risk causality of software projects as well as a model for risk management in software projects based on BNCC theory as a consequence of their research.

Hanci [52] employed computer-learning techniques to forecast which group of software projects will be at risk. Using the criteria “development source as count”, “software development life cycle model”, and “project size”, they then used ID3 and Naive Bayes algorithms to forecast which group would be in danger. They were able to acquire a variety of accuracy ratios by implementing the holdout model.

Mahfoodh and Obediat [40] designed a new technique for risk estimation to assist internal stakeholders in software development in analyzing current software risks anticipating a quantitative software risk value. To establish the significance of the risk, it was estimated using historical software bug reports and compared to current and forthcoming bug-fix times, duplicated bug records, and software component priority level. Machine learning was used to determine the risk value on a Mozilla Core dataset (Networking: HTTP software component) and a risk level value for specific software faults was forecasted using the Tensorflow tool. The overall risk was calculated using this approach to be between 27.4% and 84%, with a maximum prediction accuracy of 35%. The researchers observed a strong association between risks derived from bug-fix time estimates and risks derived from duplicated bug reports as a consequence of their investigation.

Cingiz et al. [46] specifically intended to estimate the effects of project difficulties that could result in project losses in software projects in terms of their risk factor values, as well as rank the risk factors to determine if they could provide specific information about the effects of project problems on an individual basis. To achieve these objectives, five
classification algorithms were used to forecast the impact of problems and two methods for filter feature selection to classify the importance of risk variables were used in this study.

Mahdi et al. [50] summarized the literature on creating and using machine learning algorithms for risk assessment in software development projects, as well as a study of the literature. According to the findings of the review, major developments in machine learning methodology, size measures, and study outcomes have all contributed to the growth and advancement of machine learning in project management over the past decade or more. Additionally, their research provided a more in-depth understanding of software project risk assessment, as well as a vital framework for future work in this area. Furthermore, they discovered that machine learning is more successful in reducing project failures than traditional risk assessment methods. As a result, the probability of the software project’s forecast and the reaction was increased, so giving an additional way to effectively reduce the probability of failure and raise the software development performance ratio.

Shaukat et al. [7] provided a risk dataset comprising the bulk of the risk prediction parameters as well as software needs for the new software requirements. The collection comprises the vast bulk of the requirements derived from the Software Requirement Specification (SRS) of numerous open-source projects (SRS). The study was split up into three primary phases, the first of which was the collecting of data with a risk-oriented focus. The other two phases were the validation of datasets by IT professionals and the filtration of datasets.

Chen et al. [51] devised a method for detecting the hazard of a system based on the software behavior of the system’s components. The behavior of untrusted software when it calls other untrusted software is intimately related to system risk; specifically, the more and more untrusted software is called, the greater the risk the system faces, and the converse is true. Therefore, illegal computer operation is a subset of system risk, and the two are inversely proportional to each other in terms of likelihood. A quantitative analytical method (HMM) was used to assess the system’s risk level because the number and scope of untrusted program calls can be accurately monitored, but their risk level cannot be clearly seen. This method guarantees the objectivity and correctness of the results, and it was used in their article. Also included are experiments to study and explain the risk assessment method based on software behavior, which was carried out as part of the article.

Xu et al. [45] devised a hybrid learning approach that employed evolutionary algorithms and decision trees to evolve optimum subsets of software metrics for risk prediction during the early phase of the software life cycle, and this strategy was deployed. When compared to the use of all metrics for decision tree risk prediction, the experimental results indicate the feasibility and enhanced performance of their method.

Gouthaman and Sankaranarayanan [43] provided a novel framework for analyzing the dataset gathered through a questionnaire, in which machine learning classifiers were applied and risk assessments were generated for each of the software models that were identified. Software product managers can use the results to select the most appropriate software model based on the software requirements and the probability of risk prediction.

Yu et al. [49] used the correlation coefficient to combine historical data based on conceptions such as risk weight, expert trust, and risk consequence, allowing the assessor to measure the impact of risk factors at the macro and micro level. According to the findings of the case study, the model was objective and scientific, realistic, and provided a solid framework for risk prediction, mitigation, and control activities.

In the paper of Suresh and Dillibabu [48], the Development of a new hybridized fuzzy-based risk assessment framework was used in software projects. During decision-making, the proposed technique discovered and prioritized project dangers. Using intuitionistic fuzzy-based TOPSIS, adaptive neurofuzzy inference system-based multi-criteria decision-making (ANFIS MCDM), and fuzzy decision-making trial and evaluation laboratory processes improved software project risk assessment. An improved crow search method was used to modify ANFIS parameters for a more accurate software risk rating (ECSA). Integrating ANFIS with ECSA led to solutions that stayed inside the local optimum and required
only minor ANFIS parameter modifications. NASA’s 93 dataset contained 93 software project variables for experimental validation. The experimental results showed that the proposed fuzzy-based framework properly evaluated software development project risks.

Quality Assessment

Here, in Table 4, we now show the results of the QQ questionnaire, which we had previously presented in the paper.

Table 4. Results of the quality questionnaire.

| No. | Question                                                                 | Yes            | Partly          | No             |
|-----|---------------------------------------------------------------------------|----------------|-----------------|----------------|
| QQ01| Is it apparent what the study’s main objective is?                       | 17(89.47%)     | 2(10.53%)       | 0(0%)          |
| QQ02| Is the sample of the dataset large enough to support this sort of investigation? | 12(63.16%)     | 6(31.58%)       | 5.26(0%)       |
| QQ03| Is the procedure for data collecting adequately specified and documented? | 10(52.63%)     | 7(36.84%)       | 2(10.53%)      |
| QQ04| Is there enough information on the experiment provided by the author?     | 15(78.95%)     | 2(10.53%)       | 2(10.53%)      |
| QQ05| Are the threats to the validity of the claim stated?                      | 4(21.05%)      | 7(36.84%)       | 8(42.11%)      |
| QQ06| Are the study’s limitations discussed in detail?                          | 4(21.05%)      | 7(36.84%)       | 8(42.11%)      |
| QQ07| Is there a clear definition of the performance parameters employed in the study? | 14(73.68%)     | 3(15.79%)       | 2(10.53%)      |
| QQ08| How well-defined are the learning strategies?                            | 9(47.37%)      | 8(42.11%)       | 2(10.53%)      |
| QQ09| Are the results presented in a clear and concise manner?                 | 13(68.42%)     | 4(21.05%)       | 2(10.53%)      |
| QQ10| Is there a way to compare and contrast different techniques?             | 8(47.37%)      | 7(36.84%)       | 4(15.79%)      |
| QQ11| Are there any new insights from this study that can be added to the existing literature? | 10(52.63%)     | 6(31.58%)       | 3(15.79%)      |

It is evident from the data that the great majority of QQs received positive responses to their questions. What we do know for certain is that the answer to QQ01 has clearly stated the purpose of every primary study to which it has been applied. QQ04, QQ07, and QQ09 all had results that were largely unsatisfactory. The validity and scope of the vast majority of research have not been questioned so far. It is estimated that the majority of publications are valuable additions to the existing literature, with only a few papers being only somewhat valuable, according to QQ11.

Table 5 shows the results of the quality analysis, which were classified into four categories: very high (9.5 or more), high (8 to 9), average (6.5 to 7.5), and low (6 and below). In addition, the proportion of PS and the number of studies in each of the four categories are summarized in the table below.

Table 5. Scores assigned to quality questions.

| Score            | No. of PS | Percentage |
|------------------|-----------|-----------|
| Very high (9.5 and above) | 5         | 26.32%    |
| High (8 to 9)    | 6         | 31.58%    |
| Average (6.5 to 7.5) | 6         | 31.58%    |
| Low (6 and below) | 2         | 10.53%    |

Table 6 provides a list of PS and their quality scores, as well as the quality scores of PS with ‘very high’ or ‘high’ quality scores. These two categories comprise 11 studies that obtained an average of 8 or more quality analysis points during the evaluation procedure.
Table 6. Scores of quality of PS under the categories ‘very high’ and ‘high’.

| Study No. | Authors and Title |
|-----------|-------------------|
| PS01      | Yong Hu et al. Cost-sensitive and ensemble-based prediction model for outsourced software project risk prediction 2015 [39] |
| PS02      | Hussain Mahfoodh, Qasem Obediat, Software Risk Estimation Through Bug Reports Analysis and Bug-fix Time Predictions 2020 [40] |
| PS03      | Zain Shaukat, et al. A Dataset for Software Requirements Risk Prediction 2018 [7] |
| PS04      | Jhon Masso, et al. Risk management in the software life cycle: A systematic literature review 2020 [41] |
| PS05      | Yong Hu, et al. Software project risk analysis using Bayesian networks with causality constraints 2013 [42] |
| PS06      | Gouthaman P, Suresh Sankaranarayanan, Prediction of Risk Percentage in Software Projects by Training Machine Learning Classifiers 2021 [43] |
| PS07      | Yong Hu, et al. An integrative framework for intelligent software project risk planning 2013 [44] |
| PS08      | Zhihong Xu, et al. Software Risk Prediction Based on the Hybrid Algorithm of Genetic Algorithm and Decision Tree 2007 [45] |
| PS09      | M. Ozgür Cingiz, et al. Prediction of project problem effects on software risk factors 2013 [46] |
| PS10      | Mrwan BenIdris et al. Prioritizing Software Components Risk: Towards a Machine Learning-based Approach 2020 [47] |
| PS11      | K. Suresh, R. Dillibabu, An integrated approach using IF-TOPSIS, fuzzy DEMATEL, and enhanced CSA optimized ANFIS for software risk prediction 2021 [48] |
| PS12      | Zhenyu Yu et al. Research on Software Project Risk Assessment Model based on Fuzzy Theory and Improved 2017 [49] |
| PS13      | Mohamed Najah Mahdi et al. Design and Development of Machine Learning Technique for Software Project Risk Assessment—A Review 2020 [50] |
| PS14      | Guorong Chen et al. A Risk Assessment Method based on Software Behavior 2019 [51] |
| PS15      | Chandan Kumar, Dilip Kumar Yadav, A Probabilistic Software Risk Assessment and Estimation Model for Software Projects 2015 [18] |
| PS16      | Asim Kerem Hanci, Risk Group Prediction of Software Projects Using Machine Learning Algorithm 2021 [52] |

5.2. Answers to the Research Questions

RQ01: What are the purposes and reasons for SRPM research?

Software Risk Prediction is very important to develop software with fewer hassles within an efficient budget and time. The main purpose of the study is to reduce the risks during SDLC using Machine Learning models or algorithms.

RQ02: What is the average number of SRPM studies each year?

Figure 3 illustrates a year-by-year presentation of the studies that were selected. From 2007 to 2021, a total of 15 years’ worth of data for 16 articles was displayed in this section. Each year, there is a noticeable disparity in the distribution of articles. Our first acquired research publication on SRPM was published in 2007. Since then, a large number of research articles on this topic have been published. During the year 2013, three papers were published. In the years that followed, the rate of publication decreased substantially, with five new publications being produced until 2019. The figure also shows the vast majority of articles published in 2020 and 2021, with article counts ranging from 4 to 3 in each of those years. The overall picture implies that published papers are unequally distributed. We have not detected any pattern of sequential distribution over time, which is consistent with this conclusion.

RQ03: What kinds of data sets are used in the detection process?

Depending on the type of dataset used, we divided the studies into two categories: public and private data. In contrast to public datasets, which are made available to the public, private datasets are collected and used by individuals rather than being made available as public datasets. As a result of our investigation, we learned that 37.5% of the datasets utilized in the research were publicly accessible and 62.5% were private. An important challenge is the use of a private dataset to train models for the detection or prediction. As a rule, exposure to a private dataset is confined, making it hard to compare the outputs of different machine learning models in practice.
RQ04: What is the size of the datasets?

The purpose of examining the size of datasets is to determine the external validity of the research under discussion. When a large sample of a dataset is used rather than a small dataset, the external validity of the results is improved. In addition, the size of the dataset might have an impact on the results of detection models when used in conjunction with other techniques. When developed on large-scale training data, a detection model has a large learning area, which increases the likelihood of providing more positive outcomes. The studies were separated into three groups according to the size of the datasets that were analyzed. We were able to gather the relevant information about the sample size as well as the size of the data sets that were used in each of the 16 articles we reviewed. Our definition of a “Large” study was one that used more than 200 samples, a “Medium” study was one that used 100 to 200 samples, and a “Small” study was one that used zero to 100 samples. Another class has been defined as “Unknown” because the sample sizes of the dataset used in those studies were missing. Table 7 summarizes the number of samples in each of the datasets that were used, the number of studies that were conducted, and the percentage of studies that fell into each of the three categories that were conducted.

Table 7. Category of datasets based on the number of projects.

| Category | Sample Size in Dataset | No. PS | Percentage |
|----------|------------------------|--------|------------|
| Small    | Less than 100          | 7      | 43.75%     |
| Medium   | 100 to 200             | 2      | 12.50%     |
| Large    | More than 200          | 4      | 25.00%     |
| Unknown  | Unknown Size           | 3      | 18.75%     |

RQ05: What data analysis approaches are utilized to develop SRPM models?

In the primary studies we have selected, we found two major approaches for data analyses: (1) Machine Learning Approach and (2) Statistical approach. Although most of the papers conducted a machine learning approach. In Figure 4, the ratio of machine learning and statistical approaches is shown:
RQ06: What detecting techniques are employed in the development of SRPM models?

Detection techniques, which are employed in the development of SRPM models, are divided into two major categories: (1) Classification model and (2) Regression model. Some other techniques were also applied, those not for detection but for descriptive analyses. Figure 5 shows the statistical description of these techniques:

RQ07: How many of the SRPM studies employ the ML approach?

As our main concern in this study is software risk prediction using machine learning, we need to know how many of the studies used a machine learning approach. The ratio of the ML approach studies can be seen in Figure 6.
RQ08: What are the various performance metrics used in SRPM studies?

To evaluate the performance of the prediction models, performance metrics are used. There are several performance metrics that are accessible for evaluation, in general. These performance metrics are also used in the realm of SRPM research to assess and compare the findings obtained using various prediction approaches. The following are the key performance indicators and their descriptions:

Correctly Classified Instances: The sum of True Positive (TP) and True Negative (TN) refers to correctly classified instances.

Incorrectly Classified Instances: The sum of False Positive (FP) and False Negative (FN) refers to incorrectly correctly classified instances.

Accuracy: The number of correctly classified instances out of all the instances is known as accuracy. Accuracy can be expressed as:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]  

Precision: The precision is measured as the ratio of the number of correctly classified positive instances to the total number of positive instances. Precision can be expressed as:

\[
\text{Precision} = \frac{TP}{TP + FP}
\]

Recall: The recall is obtained by dividing the total number of positive instances by the number of positive instances correctly classified as Positive. Recall can be expressed as follows:

\[
\text{Recall} = \frac{TP}{TP + FN}
\]

F-Measure: F-Measure is a method of combining precision and recall into a single measure that incorporates both. F-Measure can be expressed as:

\[
F\text{-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]
**Receiver Operating Characteristic (ROC):** A graphical way to evaluate the performance of a classifier is a receiver operating characteristic (ROC) analysis. It evaluates a classifier’s performance using two statistics: true positive rate and false positive rate [53].

**Mean Absolute Error (MAE):** The Mean Absolute Error (MAE) is a regression model assessment indicator. The MAE of a model with respect to a test set is the average of all individual prediction errors on all instances in the test set [54]. The discrepancy between the real value and the expected value for each instance is called a prediction error [55]. Mean Absolute Error (MAE) can be expressed as

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|
\]  

where,
\[
\hat{y}_i = \text{predicted value},
\]
\[
y_i = \text{true value},
\]
\[
n = \text{total number of instances}.
\]

**Mean Squared Error (MSE):** Model evaluation metric Mean Squared Error (MSE) is frequently used with regression models. The MSE of a model with respect to a test set is the average of all squared prediction errors in the test set. The difference between the real value and the expected value for an example is the prediction error [55].

**Root Mean Squared Error (RMSE):** The standard deviation of the errors that occur while making a prediction on a dataset is known as the Root Mean Squared Error (RMSE). This is similar to MSE, except that the root of the number is taken into account when calculating the model’s accuracy.

**Matthews Correlation Coefficient (MCC):** The Matthews correlation coefficient (MCC) is a metric that indicates how closely true classes and projected instances are related [56]. It can be expressed as

\[
MCC = \frac{(TP \times TN - FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}
\]  

**Kappa Statistic:** As the Kappa statistic takes the chance factor into account, it is essential to consider the outcomes using this method. If the kappa statistic is near to one, the classification without change factor was successful.

**Median Absolute Error (MedAE):** The median absolute error is not affected by outliers. The loss is derived by averaging all of the absolute deviations between the true value and the prediction. It can be expressed as:

\[
MedAE(E, E') = median(|E_1 - E'_1|, \cdots, |E_n - E'_n|)
\]  

where,
\[
E = \text{true value},
\]
\[
E' = \text{predicted value},
\]
\[
n = \text{total number of instances}.
\]

**R²:** This measure indicates how well a model matches a certain dataset. It shows how close the regression line is to the actual data values. The R² value ranges from 0 to 1, with 0 indicating that the model does not match the given data and 1 indicating that the model fits the dataset correctly.

Table 8 shows the performance metrics that were used in the 16 papers chosen.
Table 8. Employed performance metrics.

| Performance Metrics                  | Primary Studies |
|--------------------------------------|-----------------|
| Correctly Classified Instances       | PS03, PS09      |
| Incorrectly Classified Instances     | PS03            |
| Accuracy                             | PS01, PS05, PS06, PS07, PS08, PS10, PS12, PS14, PS16 |
| Precision                            | PS07, PS09      |
| Recall                               | PS07, PS09      |
| F-Measure                            | PS07, PS09, PS10|
| Receiver Operating Characteristic (ROC) | PS10            |
| Mean Absolute Error (MAE)            | PS02, PS03      |
| Mean Squared Error (MSE)             | PS02            |
| Root Mean Squared Error (RMSE)       | PS03            |
| Matthews Correlation Coefficient (MCC) | PS10          |
| Kappa Statistic                      | PS09            |
| Median Absolute Error (MedAE)        | PS02            |
| R²                                   | PS02            |

We could discover that five performance metrics, Accuracy, Precision, Recall, F-Measure, and Mean Absolute Error (MAE), are the most often-used measures. Among them, Accuracy is the most used among the studies. Moreover, Precision, Recall, F-Measure, and MCC are also very commonly used when datasets are unbalanced. Therefore, considering the type of datasets, one can determine which performance measures should be chosen for the prediction models.

RQ09: What is the efficiency of the SRPM that has been proposed?

In this part, we summarize the performance of the PS of SRPM in this part. We looked at the results of all 16 PS to get an answer to this question. Allowing for any rating of studies on the basis of performance is highly difficult. We discovered that they use a variety of performance indicators, making it impossible to compare their results.

In Table 9, the value of the most considered and five performance metrics (Accuracy, Precision, Recall, F-Measure, and MAE) employed in the investigations was shown. For each performance metric, the highest performer’s values are highlighted.

Table 9. Top five studies based on the results of the performance metrics.

| Primary Study | Performance |
|---------------|-------------|
| PS16          | Accuracy: 100% |
| PS06          | Accuracy: 99.67% |
| PS09          | Accuracy: 97.2%, Recall: 97.9%, F-Measure: 97.5% |
| PS10          | Accuracy: 97%, F-Measure: 97% |
| PS08          | Accuracy: 87.77% |

RQ10: What is the main research emphasis of the papers?

The authors took various approaches in the primary studies, but the main research emphasis of the articles was to predict or assess software risk. Machine learning, statistical methods, and complex systems were all employed in the publications that were considered. Different kinds of classification and regression algorithms were employed in the machine learning approach. Inference and HMM techniques were employed for the statistical procedure. The Fuzzy Dematel approach was utilized for the complex system.
RQ11: What are the limits and problems of the SRPM highlighted in the studies?

The debate on the limitations and challenges of the software risk prediction model (SRPM) stated in the primary studies is summarized in the following paragraphs:

PS05 mentioned that the study had two limitations. For starters, the suggested technique cannot ensure that the data would provide a complete causal Bayesian network. The causalities discovered could only build a partial causality network due to the sample size constraint. Second, the suggested approach can only detect a fraction of the causalities that are underlying them.

PS01 used a diverse variety of datasets, including industrial projects that were not restricted to a single software firm or type of software. The research, on the other hand, had a hard time deciding on the right $\lambda$ value.

PS04 is a systematic literature review, and the authors mentioned that they might have only added chosen articles from Scopus and missed some of the important articles. They also mentioned that they might have missed some more articles during the study selection process as well.

PS07 indicates that the study developed a model for predicting risk control activities, but it was unable to establish the risk control activities’ execution sequence.

The studies mention the limitations and difficulties mentioned above in predicting SRPM. Future researchers can take these considerations into account when developing prediction models.

6. Conclusions

Requirement engineering is one of the flourishing phases of the Software Development Life Cycle (SDLC), and risk analysis is another crucial part of SDLC. Different types of risks exist in software development and must be considered. The primary objective of risk analysis is not only to identify hazards, but also to attempt to manage them. Additionally, it may offer specific information about hazards and make recommendations for mitigating them. Risk analysis’ major objective is to identify hazards accurately. Risk analysis should incorporate critical components such as problem description, formulation, and data collection. If the risk analysis is not performed properly, even a single risk factor can be the cause of system failure. As is usual, software risks cause problems for users, researchers have proposed several approaches to predict and prevent software risks. This study conducts a systematic review of the literature (SLR) and quality assessment of previously published software risk prediction models (SRPM) research publications. We studied by collecting articles from some digital libraries and then doing a search. The 16 most relevant articles were then designated as primary studies (PS), while the rest of the articles were omitted because they were not specifically written on the topic. To meet the SLR and quality analysis requirements, we investigated 16 different research questions and 11 research questions. The questions are answered in this SRPM research paper, which provides publication information, dataset information, detection tactics, data analysis methodology, performance metrics, detection model performance, targeted scopes, and the use of machine learning in investigations. To reply to the questionnaire, we observed SRPM from different aspects. The following are the key conclusions from the chosen PS:

- The demographics data implies that there were very few journal articles among the publications, indicating a scarcity of publication work.
- Based on the findings of the quality evaluation, the PSs were classified into four groups. Little research received significant scores, but the majority of studies had average scores according to the category.
- The majority of the researchers employed their own privately obtained datasets for their detection models, according to the findings. Furthermore, the majority of the research used large datasets, implying that the number of samples in the datasets was significant.
Only five research studies considered signature-based detection strategies to develop detection models.

For predicting software risks, machine learning algorithms provide an acceptable detection capability. According to the results of performance measures, the majority of ML-based research does exceptionally well.

Overall, we find that there is a lack of high-quality work in the SRPM literature and a lack of consistency in the approach used to forecast software risk detection investigations. In the future, we have a plan to conduct a systematic review based on the PRISMA method [57] and will also compare the method with this SLR.

Author Contributions: Conceptualization, M.H.M., M.T.H.N., D.M.N.A.A. and M.A.K.; review method, M.H.M., M.T.H.N. and D.M.N.A.A.; study selection, M.H.M., M.T.H.N. and D.M.N.A.A.; validation, M.H.M., M.T.H.N., D.M.N.A.A. and M.A.K.; writing—original draft preparation, M.H.M.; writing—review and editing, M.H.M., M.T.H.N., D.M.N.A.A. and M.A.K.; visualization, M.H.M., M.T.H.N., D.M.N.A.A. and M.A.K.; supervision, M.A.K.; project administration, M.H.M., M.T.H.N., D.M.N.A.A. and M.A.K.; funding acquisition, M.A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Mall, R. Fundamentals of Software Engineering; PHI Learning Pvt. Ltd.: Kharagpur, India, 2018.
2. Roger, S.P.; Bruce, R.M. Software Engineering: A Practitioner’s Approach; McGraw-Hill Education: New York, NY, USA, 2015.
3. Hu, Y.; Zhang, X.; Sun, X.; Liu, M.; Du, J. An intelligent model for software project risk prediction. In Proceedings of the 2009 International Conference on Information Management, Innovation Management and Industrial Engineering, Xi’an, China, 26–27 December 2009; Volume 1, pp. 629–632.
4. Teklemariam, M.A.; Mnkandla, E. Software project risk management practice in Ethiopia. Electron. J. Inf. Syst. Dev. Ctries. 2017, 79, 1–14. [CrossRef]
5. Jaafar, J.; Janjua, U.I.; Lai, F.W. Software effective risk management: An evaluation of risk management process models and standards. In Information Science and Applications; Springer: Boston, MA, USA, 2015; pp. 837–844.
6. Foo, S.W.; Muruganathan, A. Software risk assessment model. In Proceedings of the 2000 IEEE International Conference on Management of Innovation and Technology, ICIMT 2000, ‘Management in the 21st Century’ (Cat. No. 00EX457), Singapore, 2–15 November 2000; Volume 2, pp. 536–544.
7. Shaukat, Z.S.; Naseem, R.; Zubair, M. A Dataset for Software Requirements Risk Prediction. In Proceedings of the 2018 IEEE International Conference on Computational Science and Engineering (CSE), Bucharest, Romania, 29–31 October 2018; pp. 112–118. [CrossRef]
8. Bista, R.; Karki, S.; Dongol, D. A new approach for software risk estimation. In Proceedings of the 2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Malabe, Sri Lanka, 6–8 December 2017; pp. 1–8. [CrossRef]
9. Filippetto, A.S.; Lima, R.; Barbosa, J.L.V. A risk prediction model for software project management based on similarity analysis of context histories. Inf. Softw. Technol. 2021, 131, 106497. [CrossRef]
10. Naseem, R.; Shaukat, Z.; Irfan, M.; Shah, M.A.; Ahmad, A.; Muhammad, F.; Glowacz, A.; Dunai, L.; Antonino-Daviu, J.; Sulaiman, A. Empirical assessment of machine learning techniques for software requirements risk prediction. Electronics 2021, 10, 168. [CrossRef]
11. Bakhsh, S.T.; Shahzad, B.; Tahir, S. Risk management approaches for large scale software development. J. Inf. Sci. Eng. 2017, 33, 1547–1560.
14. Janjua, U.I.; Jaafar, J.; Lai, F.W. Expert's opinions on software project effective risk management. In Proceedings of the 2016 3rd International Conference on Computer and Information Sciences (ICCIONS), Kuala Lumpur, Malaysia, 15–17 August 2016; pp. 471–476.

15. Arnuphaptrairong, T. Top ten lists of software project risks: Evidence from the literature survey. In Proceedings of the International MultiConference of Engineers and Computer Scientists, Hong Kong, China, 16–18 March 2011; Volume 1, pp. 1–6.

16. Lytyinen, K.; Mathiassen, L.; Ropponen, J. A framework for software risk management. J. Inf. Technol. 1996, 11, 275–285. [CrossRef]

17. Elzamly, A.; Hussin, B. Classification and identification of software risk techniques for mitigating risks with factor analysis technique in software risk management. Rev. Comput. Eng. Res. 2015, 2, 22–38. [CrossRef]

18. Kumar, C.; Yadav, D.K. A Probabilistic Software Risk Assessment and Estimation Model for Software Projects. Procedia Comput. Sci. 2015, 54, 353–361. [CrossRef]

19. Sundararajan, S.; Bhasi, M.; Pramod, K. An empirical study of industry practices in software development risk management. Int. J. Sci. Res. Publ. 2013, 3, 1–11.

20. Khatavakhotan, A.S.; Ow, S.H. An innovative model for optimizing software risk mitigation plan: A case study. In Proceedings of the 2012 Sixth Asia Modelling Symposium, Bali, Indonesia, 29–31 May 2012; pp. 220–224.

21. Khatavakhotan, A.S.; Ow, S.H. Development of a software risk management model using unique features of a proposed audit component. Malays. J. Comput. Sci. 2015, 28, 110–131.

22. Tavares, B.G.; da Silva, C.E.S.; de Souza, A.D. Practices to improve risk management in agile projects. Int. J. Softw. Eng. Knowl. Eng. 2019, 29, 381–399. [CrossRef]

23. Asif, M.; Ahmed, J. A novel case base reasoning and frequent pattern based decision support system for mitigating software risk factors. IEEE Access 2020, 8, 102278–102291. [CrossRef]

24. Shahzad, B.; Al-Mudimigh, A.S. Risk identification, mitigation and avoidance model for handling software risk. In Proceedings of the 2010 2nd International Conference on Computational Intelligence, Communication Systems and Networks, Liverpool, UK, 28–30 July 2010; pp. 191–196.

25. Mohamud Sharif, A.; Basri, S. Software risk assessment: A review on small and medium software projects. In Proceedings of the International Conference on Software Engineering and Computer Systems, Kuantan, Malaysia, 27–29 June 2011; pp. 214–224.

26. Li, S.; Duo, S. Safety analysis of software requirements: Model and process. Procedia Eng. 2014, 80, 153–164. [CrossRef]

27. Gupta, D.; Sadiq, M. Software risk assessment and estimation model. In Proceedings of the 2008 International Conference on Computer Science and Information Technology, Singapore, 29 August–2 September 2008; pp. 963–967.

28. Keshlaf, A.A.; Riddle, S. Risk management for web and distributed software development projects. In Proceedings of the 2010 Fifth International Conference on Internet Monitoring and Protection, Barcelona, Spain, 9–15 May 2010; pp. 22–28.

29. Salih, H.A.; Ammar, H.H. Model-based resource utilization and performance risk prediction using machine learning Techniques. JOIV Int. J. Inform. Vis. 2017, 1, 101–109.

30. Verma, B.; Dhanda, M.; Verma, B.; Dhanda, M. A review on risk management in software projects. Int. J. 2016, 2, 499–503.

31. Chang, C.P. Mining software repositories to acquire software risk knowledge. In Proceedings of the 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI), San Francisco, CA, USA, 14–16 August 2013; pp. 489–496.

32. Bhujang, R.K.; Suma, V. A Comprehensive Solution for Risk Management in software development projects. Int. J. Softw. Eng. Knowl. Eng. 2018, 17, 153–175. [CrossRef]

33. Chowdhury, A.A.M.; Arefeen, S. Software risk management: Importance and practices. IJICT ISSN 2011, 2, 2078–5828.

34. Kitchenham, B.; Charters, S. Guidelines for performing systematic literature reviews in software engineering. In EBSE Technical 547 Report; University of Durham: Durham, UK, 2007; Volume 2.

35. Kitchenham, B. Procedures for Performing Systematic Reviews; Keele University: Keele, UK, 2004; Volume 33, pp. 1–26.

36. Malhotra, R. A systematic review of machine learning techniques for software fault prediction. Appl. Soft Comput. 2015, 27, 504–518. [CrossRef]

37. Son, L.H.; Pritam, N.; Khari, M.; Kumar, R.; Phuong, P.T.M.; Thong, P.H. Empirical study of software defect prediction: A systematic mapping. Symmetry 2019, 11, 212. [CrossRef]

38. Sohan, M.F.; Basalamah, A. A Systematic Literature Review and Quality Analysis of Javascript Malware Detection. IEEE Access 2020, 8, 190539–190552. [CrossRef]

39. Hu, Y.; Feng, B.; Mo, X.; Zhang, X.; Ngai, E.; Fan, M.; Liu, M. Cost-sensitive and ensemble-based prediction model for outsourced software project risk prediction. Decis. Support Syst. 2015, 72, 11–23. [CrossRef]

40. Mahfoodh, H.; Obediat, Q. Software Risk Estimation Through Bug Reports Analysis and Bug-fix Time Predictions. In Proceedings of the 2020 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), Sakheer, Bahrain, 20–21 December 2020; pp. 1–6. [CrossRef]

41. Maslo, J.; Pino, F.J.; Pardo, C.; García, F.; Piattini, M. Risk management in the software life cycle: A systematic literature review. Comput. Stand. Interfaces 2020, 71, 103431. [CrossRef]

42. Hu, Y.; Zhang, X.; Ngai, E.; Cai, R.; Liu, M. Software project risk analysis using Bayesian networks with causality constraints. Decis. Support Syst. 2013, 56, 439–449. [CrossRef]

43. P, G.; Sankaranarayanan, S. Prediction of Risk Percentage in Software Projects by Training Machine Learning Classifiers. Comput. Electr. Eng. 2021, 94, 107362. [CrossRef]
44. Hu, Y.; Du, J.; Zhang, X.; Hao, X.; Ngai, E.; Fan, M.; Liu, M. An integrative framework for intelligent software project risk planning. *Decis. Support Syst.* 2013, 55, 927–937. [CrossRef]

45. Xu, Z.; Yang, B.; Guo, P. Software Risk Prediction Based on the Hybrid Algorithm of Genetic Algorithm and Decision Tree. In *Proceedings of the Advanced Intelligent Compuering Techniques and Applications. With Aspects of Contemporary Intelligent Computing Techniques*; Huang, D.S., Heutte, L., Loog, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 266–274.

46. Cingiz, M.O.; Unudulmaz, A.; Kalıpsız, O. Prediction of project problem effects on software risk factors. In Proceedings of the 2013 IEEE 12th International Conference on Intelligent Software Methodologies, Tools and Techniques (SoMeT), Budapest, Hungary, 22–24 September 2013; pp. 57–61. [CrossRef]

47. BenIdris, M.; Ammar, H.; Dzielski, D.; Benamer, W.H. Prioritizing Software Components Risk: Towards a Machine Learning-Based Approach. In Proceedings of the 6th International Conference on Engineering & MIS 2020, Almaty, Kazakhstan, 14–16 September 2020; Association for Computing Machinery: New York, NY, USA, 2020. [CrossRef]

48. BenIdris, M.; Ammar, H.; Dzielski, D.; Benamer, W.H. Prioritizing Software Components Risk: Towards a Machine Learning-Based Approach. In Proceedings of the 6th International Conference on Engineering & MIS 2020, Almaty, Kazakhstan, 14–16 September 2020; Association for Computing Machinery: New York, NY, USA, 2020. [CrossRef]

49. Yu, Z.; Yang, K.; Luo, Y.; Deng, Q. Research on software project risk assessment model based on fuzzy theory and improved. In Proceedings of the 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 25–26 March 2017; pp. 2073–2077. [CrossRef]

50. Mahdi, M.N.; Zabil M.H., M.; Yusof, A.; Cheng, L.K.; Mohd Azmi, M.S.; Ahmad, A.R. Development of Machine Learning Technique for Software Project Risk Assessment—A Review. In Proceedings of the 2020 8th International Conference on Information Technology and Multimedia (ICIMU), Selangor, Malaysia, 24–26 August 2020; pp. 354–362. [CrossRef]

51. Chen, G.; Wang, K.; Tan, J.; Li, X. A Risk Assessment Method based on Software Behavior. In Proceedings of the 2019 IEEE International Conference on Intelligence and Security Informatics (ISI), Shenzhen, China, 1–3 July 2019; pp. 47–52. [CrossRef]

52. Hancı, A.K. Risk Group Prediction of Software Projects Using Machine Learning Algorithm. In Proceedings of the 2021 6th International Conference on Computer Science and Engineering (UBMK), Ankara, Turkey, 15–17 September 2021; pp. 503–505. [CrossRef]

53. Tan, P.-N. *Receiver Operating Characteristic*; Springer: Boston, MA, USA, 2009.

54. Sankalp, S.; Sahoo, B.B.; Sahoo, S.N. Deep learning models comparable assessment and uncertainty analysis for diurnal temperature range (DTR) predictions over Indian urban cities. *Results Eng.* 2022, 13, 100326. [CrossRef]

55. Sammut, C.; Webb, G.I. *Encyclopedia of Machine Learning*; Springer Science & Business Media: New York, NY, USA, 2011.

56. Chicco, D.; Tötsch, N.; Jurman, G. The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation. *BioData Min.* 2021, 14, 13. [CrossRef]

57. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *Syst. Rev.* 2021, 10, 89. [CrossRef]