applied sciences

MDPI

Article

Software Project Management Using Machine Learning Technique—A Review

Mohammed Najah Mahdi [1,*], Mohd Hazli Mohamed Zabil [2], Abdul Rahim Ahmad [2], Roslan Ismail [2], Yunus Yusoff [2], Lim Kok Cheng [2], Muhammad Sufyian Bin Mohd Azmi [2], Hayder Natiq [3] and Hushalini Happala Naidu [4]

1 Institute of Informatics and Computing in Energy, Universiti Tenaga Nasional, Kajang 43000, Malaysia
2 College of Computing and Informatics (CCI), Universiti Tenaga Nasional, Kajang 43000, Malaysia; hazli@uniten.edu.my (M.H.M.Z.); Abdrahim@uniten.edu.my (A.R.A.); roslan@uniten.edu.my (R.I.); yunusy@yniten.edu.my (Y.Y.); Kokcheng@uniten.edu.my (L.K.C.); sufyian@uniten.edu.my (M.S.B.M.A.)
3 Department of Computer Technology, Information Technology Collage, Imam Ja'afar Al-Sadiq University, Baghdad 10064, Iraq; hayder.natiq@sadiq.edu.iq
4 Uniten R&D Sdn Bhd, Universiti Tenaga Nasional, Kajang 43000, Malaysia; hushalini@uniten.edu.my
* Correspondence: Najah.Mahdi@uniten.edu.my

Abstract: Project management planning and assessment are of great significance in project performance activities. Without a realistic and logical plan, it isn't easy to handle project management efficiently. This paper presents a wide-ranging comprehensive review of papers on the application of Machine Learning in software project management. Besides, this paper presents an extensive literature analysis of (1) machine learning, (2) software project management, and (3) techniques from three main libraries, Web Science, Science Directs, and IEEE Explore. One-hundred and eleven papers are divided into four categories in these three repositories. The first category contains research and survey papers on software project management. The second category includes papers that are based on machine-learning methods and strategies utilized on projects; the third category encompasses studies on the phases and tests that are the parameters used in machine-learning management and the final classes of the results from the study, contribution of studies in the production, and the promotion of machine-learning project prediction. Our contribution also offers a more comprehensive perspective and a context that would be important for potential work in project risk management. In conclusion, we have shown that project risk assessment by machine learning is more successful in minimizing the loss of the project, thereby increasing the likelihood of the project success, providing an alternative way to efficiently reduce the project failure probabilities, and increasing the output ratio for growth, and it also facilitates analysis on software fault prediction based on accuracy.

Keywords: machine learning technique; software project estimation; software estimation; software project management; project risk assessment

Citation: Mahdi, M.N.; Mohamed Zabil, M.H.; Ahmad, A.R.; Ismail, R.; Yusoff, Y.; Cheng, L.K.; Azmi, M.S.B.M.; Natiq, H.; Happala Naidu, H. Software Project Management Using Machine Learning Technique—A Review. *Appl. Sci.* **2021**, *11*, 5183. https://doi.org/10.3390/app11115183

Academic Editor: Vito Conforti

Received: 12 April 2021
Accepted: 5 May 2021
Published: 2 June 2021

1. Introduction

Improving the efficiency and maintaining the sustainability of a software project are obstacles that are faced by project managers. The probability of project failure is generally due to the lack of knowledge, skills, resources, and technology during project implementation [1–3].

The knowledge that is obtained from historical project data sets can be used for the development of predictive models by either utilizing a mathematical methodology, including linear regression and study of association or machine learning (ML) approaches, such as Artificial Network Network (ANN) and Support Vector Machines (SVM). Predictive methods provide a method that is focused on present and historical project evidence to forecast the project's future. Different ML algorithms have not yet been studied owing to a huge number of ML algorithms. According to the literature findings, the reason for

using automated projects, the issues of the evaluation of the project management, and the development ML methodology are addressed. The empirical results would be evaluated.

Although project literature describes the success and failure of the project, there are lengthy debates regarding how project improvements can be measured. The perceptions of project performance and the assessment of project success vary [4]. Hughes and other members [5] Project Management Institute (PMI) [6] discern between project success and project performance variables.

The project progress thresholds are evaluated to measure the success and failure of a project. Feedback is also taken into consideration for project progress. Historically, the delivery of the necessary outcomes and the utilization of the selected resources is distinguished by a successful project under a specific project duration [7]. PMI [6] identifies initiatives that successfully achieve the stakeholder's project goals, criteria, and ambitions. Researchers, such Como Aladwani [8], Cates and Mollaghasemi [9], Parsons [10], and Rosenfeld [11], describe the effects of the classical objective outcome metrics, such as project expense (above and below budget), project time (early, on or at late), and the project outcomes output (with less or better than required properties and functions).

The evaluation of project requirements also contributes to costs, time expenses, unfulfilled goals, or even cancellations of projects, becoming a natural, unwanted project danger of adverse effects on the reliability of software projects [12]. The requirements for amending the specifications (in terms of multiple extension, elimination, and modification) during the software development project are among the principal factors raising problems for the project [13–16].

Section 2 of this article includes an explanatory analysis on the principles of software project assessment and computer training technology. This article is further structured, as follows. Section 3 defines the approach, including the source of material, requirements for eligibility of research, the Systematic Literature Review (SLR), and the effects of search results of publications. It also identifies the research questions (RQs) for this research, with threats to validity pointing out major challenges to the effectiveness of SLRs. The queries of any object from three website papers were separated into four classes, the literary taxonomy on software project management utilizing an ML technique. Section 4 addressed inspiration, difficulties, and recommendations in that research area and a modern approach to the risk management of software projects. Finally, Section 5 presents the Conclusion.

2. Preliminary Study

In this section, we clarify some concepts of software project estimation and machine learning technique.

2.1. Software Effort Estimation

The pediction of software development effort and duration is the critical task for effective Software Project Management (SPM). The accuracy and reliability of prediction mechanisms are also essential. Having accurate effort assessments, especially at an early software project phase, may significantly reduce the high risks that are taken during the development of a software product. Unfortunately, most of the existing estimation techniques are often substantially wrong, and most of the projects encounter effort overruns. However, it was found that software project estimation based on ML algorithms can provide more accurate effort estimation.

2.2. Machine Learning (ML)

The ML is an application of artificial intelligence that provides systems to learn and improve from experience without being explicitly programmed automatically. In other words, the primary aim of ML is to allow the computers to learn automatically without human intervention or help and then adjust actions accordingly. Furthermore, ML enables the processing of massive volumes of information.

2.3. Software Project Management Estimation Based on ML

Figure 1 illustrates the procedure of software project management estimation, which can be summarized, as follows.

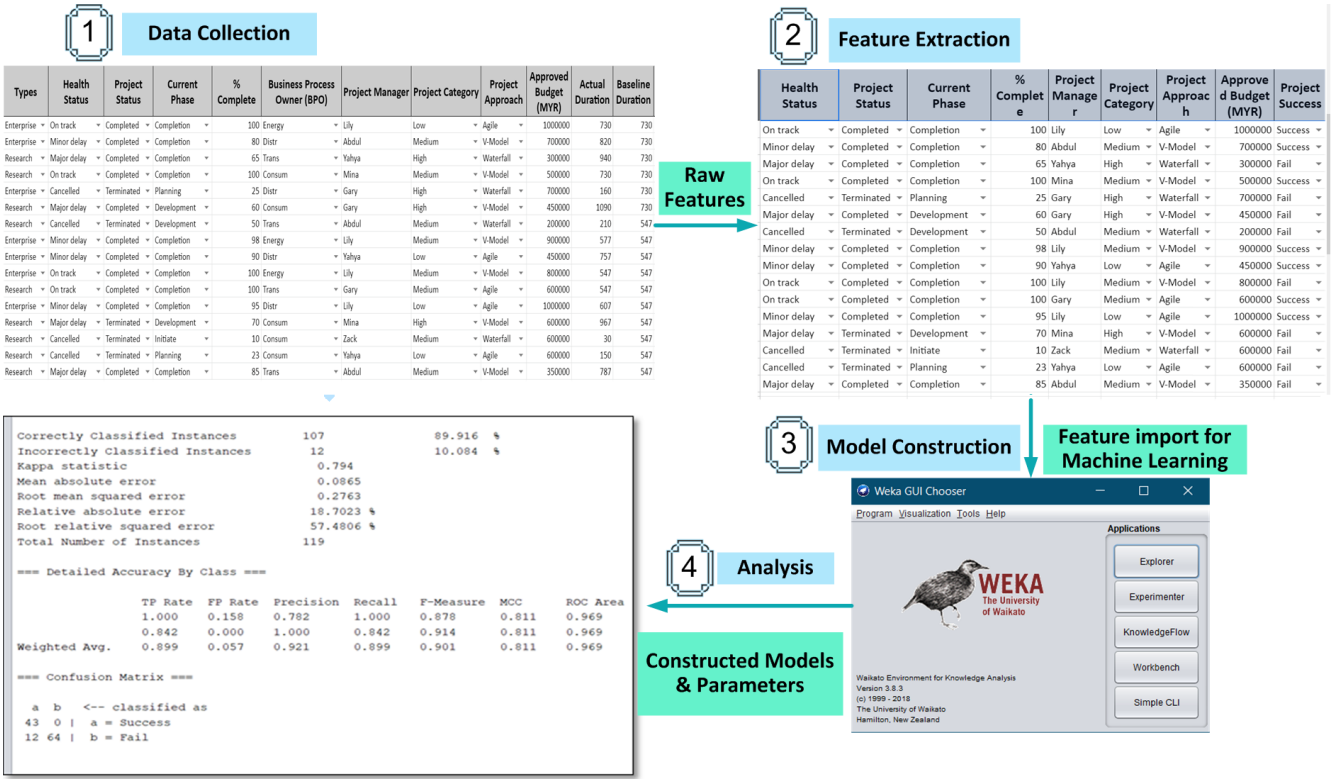

Figure 1. Example of Methodology for Developing Software Project Management Estimation.

Step 1. Data Collection: token extraction, word tokens are essential characteristics for calculating effort in the project. A key part of the project material is defined by Tokens. As a core component of the estimation model, Unigram language modelling concerning tokens was used.

Step 2. Feature Extraction: after the extraction of tokens, the project features were chosen for further analysis: Term Frequency–Inverse Document Frequency (TF-IDF) of each token 'Term Frequency' is a tool for calculating the sense of a phrase that considers word frequency and the reverse counts of records, including this word. This is a method of measuring a word's meaning.

Step 3. Model Construction: for different classification algorithms, the derived features are used as inputs to Weka. The following is an overview of how we chose the learning algorithms: Naïve Bayes: a classification is a probabilistic classification that is focused on the theorem of Bayes that takes independent features from the classroom. Naïve Bayes, with its simplicity, fights the high dimensionality of the data with the subjective assertion of freedom, which can also exceed more complex classification approaches. J48 Decision tree: Java's open source C4.5. It is a decision tree-generating algorithm where the tanning collection is not linearly separable, decision trees suit the training results well. Random Forest: it is a category classifier that consists of several decision trees and the class outputs, which are the statistical model of the individual trees output groups.

Step 4. Analysis: the study found the best ML models to outperform and showed that the project risk calculation using machine learning is more effective in minimizing the fault of the project, which improves the likelihood of the project answer, providing an alternative way to efficiently reduce the probabilities and increase the output ratio for growth.

3. Methodology

We obtained articles that were related to SPM by searching using two phrases: "Machine Learning" and "Software project management". The searches were done on three digital libraries; (1) Web of Science (WOS), as it caters to multi-disciplinary research articles in the fields of science, arts, etc., (2) IEEE explore which provides articles that are specialized in the field of electrical and electronics engineering, and (3) ACM digital library, which has a comprehensive database that contains scientific articles regarding computing and information technology.

The significant articles and literature referred by the search outputs were selected and categorized based on two criteria: (1) use three iterations in the filtering process, which removes the redundant and duplicated articles, excluding irrelevant articles using the title. (2) Undertake the initial screening, and the selected papers are following the screen using the SPM by carefully studying the narrowed search results.

Many searches on the three databases mentioned were done in March 2020 using several keywords (or phrases), such as "machine learning" OR ML OR "artificial intelligence" OR "classification" OR "clustering" OR "regression" AND "software project management" OR "SPM" OR "software development" OR "application development" OR "apps" OR "software" AND "techniques" OR Methods OR implementation OR guide OR algorithm."

Figure 2 shows a typical query text used. We have excluded the search results, which are correspondences, letters, book chapters, etc., using the advanced options of search engines. The exclusions are to obtain the most recent scientific articles and only those of great importance that enhance the SPM capability. The focus is to include all of the articles and scientific manuscripts that fulfil all the criteria of this work. Subsequently, they are then divided into classes, namely: general and coarse-grained. The latter is discussed in four subsequent sections obtained from the study results, in which the Google scholar search engine was utilized in defining the direction of the study.

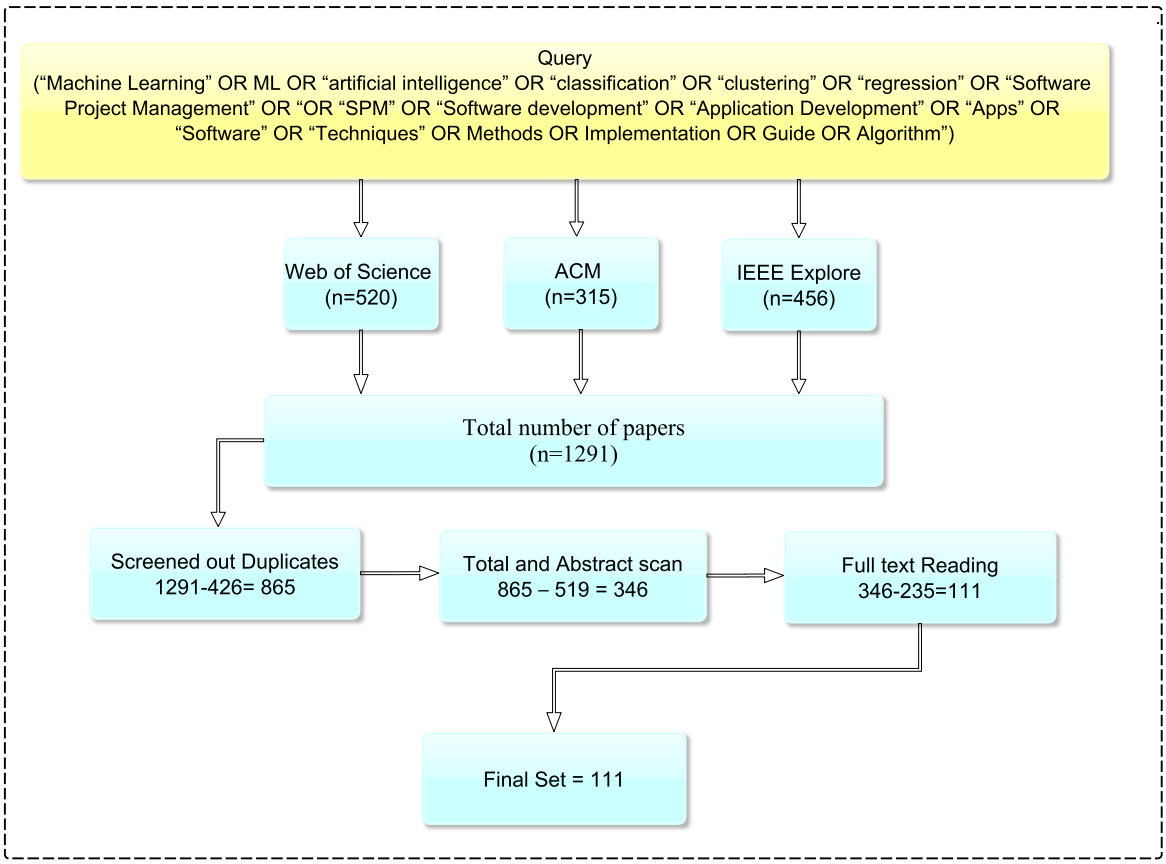

Figure 2. Research Methodology Guideline.

Figure 2 shows that there were 1291 papers gathered after the queries were performed, of which, from all the documents, 520 were obtained from WOS, 456 from IEEE, and 315 from ACM digital libraries. All of the selected articles were published between 2009 and 2020. These articles were later divided into three groups; (1) 426 redundant articles, (2) 519 were irrelevant based on the titles and abstracts, and 111 articles fall within the SPM criteria.

As highlighted earlier, an article is excluded if it does not satisfy the selection criteria, which are listed below: (1) the English language is not the language used to write the paper. (2) Techniques and/or methods were the focus of the article. (3) The research interest in the article is only concentrated on the SPM without Software Development or Machine Learning.

Moreover, after the second iteration, the articles can still be eliminated if SPM was not included or: (1) the paper's contribution does not consider any aspects of machine learning and project management. (2) The discussion on the paper is only focused on SPM and it does not discuss any other topic. In this work, articles undergo extensive ML, whereby the remaining articles are later categorized into categories that look into how to enhance the SPM.

3.1. Threats to Validity

Other studies have pointed out significant challenges to the effectiveness of SLRs [17,18], and highlighted trends of using Machine learning algorithms, benchmark data-sets, validation methods, and size metrics for software effort estimation. Four different strategies were used to minimize the risks that are posed by these TTVs strategically.

Firstly, construct validity: the framework was verified by implementing a manual and automated sentence search to minimize the calculated SPM data from data collection. Additionally, the selected articles evaluate the SPM by thoroughly analyzing the reduced search results.

Secondly, internal validity: the methods that were proposed by [17,18] were used to solve the internal validity. Additionally to avoid biases during the exhaustive search for journal articles, a technique that combines two phases of the search was used for a comprehensive selection approach. All articles of interest were extracted from databases used for related researches [17–19] and they were subjected to thorough selection processes that are shown in Figure 2.

Thirdly, external validity: external validity was addressed by integrating ten years time frame of SPM studies—leading to generalized results. There is a parallel relationship between the cumulative collection of papers and available papers, which suggests that this SLR can maintain a generalized report that is consistent with the external validity criteria of the research.

Lastly, conclusion validity: the conclusion validity was handled using SLR methods and guidelines applied by researchers from reputable publications, such as [19], which makes the results possible to reproduce the research chronologies of this SLR with quantifiable and identical results.

3.2. Research Questions

In view of conducting a systematic literature review, the research questions play a prominent role in deciding the search strategy and analysis. We identified the following research questions (RQs) for this research:

RQ1. What does the existing research literature reveal about Software Project Management using machine learning techniques?
RQ2. Can we build better machine learning-based models in terms of accuracy prediction by applying feature transformation and feature selection to reduce the project failure probabilities efficiently?
RQ3. What are the existing gaps for prospects of research in the field of Software Project Management?

RQ4. What are the prediction metrics and their current level of accuracy evidenced by different estimation techniques?
RQ5. Which machine learning algorithm tends to overestimate and which tends to underestimate?

3.3. Statistical Information of Collects Articles

The outcome of the review is addressed in the form of answers to the research questions.

RQ1. What does the existing research literature reveal about Software Project Management using machine learning techniques?

Figure 3 demonstrates the taxonomy. The records may be categorized into four fundamental classes. (1) Review and Survey; (2) Case Study; (3) Experimental; and, (4) Analysis and Architecture. The first class of research and questionnaire materials outlines the ML approaches and strategies employed in SPM to accomplish their objectives and address concerns. The second category discusses the effects, triggers, countermeasures, conditions, and proposes technologies for improved efficacy control. The third category presents the effects of a methodology used to classify multiple variables, which can affect different aspects of the method or the product as it is produced. The fourth category incorporates structures, methods, and expertise for a mission.

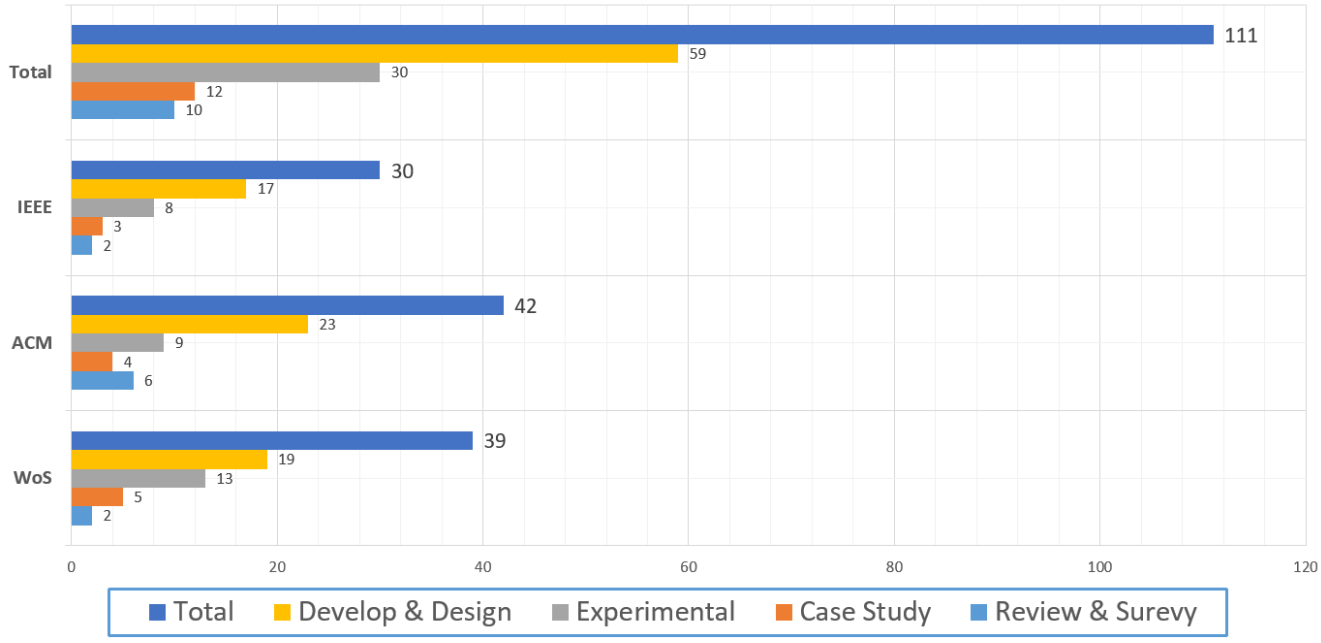

Figure 3. A number of included articles in different categories according to publication journals.

Figure 3 presents the statistics of the different categories above for the articles that are related to SPM. In the figure, the 111 articles from the three databases are divided into review and surveys (10), Case Study (12), Develop and Design (59), and those that describe the Experimental study (30).

Figure 4 presents the rapid number of publications based on the fields and region in which the study and studies are developed in SPM. The findings were split into 10 of the 111 papers, 12 of the 111 are relevant publications on ML methods of case analysis and SPM strategies, and 30 of the 111 articles, systematic steps and experimental criteria for the project management review ML-Software. The final groups of scientific contributions and outcomes in ML-SPM research design and growth, 59 of 111 articles. The figure even demonstrates the mathematical study of the multiple groups.

Figure 4, on the other side, includes papers that are dependent on the year of publishing and displays the science classified articles between 2009 and 2020. In 2009, only eight papers were written, and 49 were published, from 2010 to 2015. In comparison, for

2016, 2017, 2018, 2019, and 2020, respectively, 15, 16, 11, seven, and five papers have been written. Primary sources of analysis based on ML-SPM research were included, and its general guidelines were evaluated.

We found several trends and produced a taxonomy, as shown in Figure 5. We also distinguished many subcategories, but several main areas have been observed. We noticed some themes from the literature and rendered a taxonomy, as seen in Figure 5. While specific fields were overlapping, we established multiple subcategories.

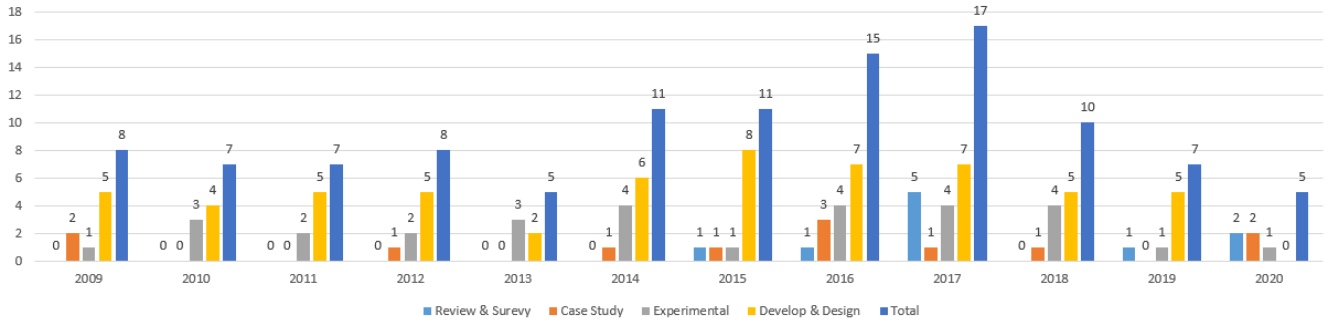

Figure 4. Published articles in between 2005 and 2020.

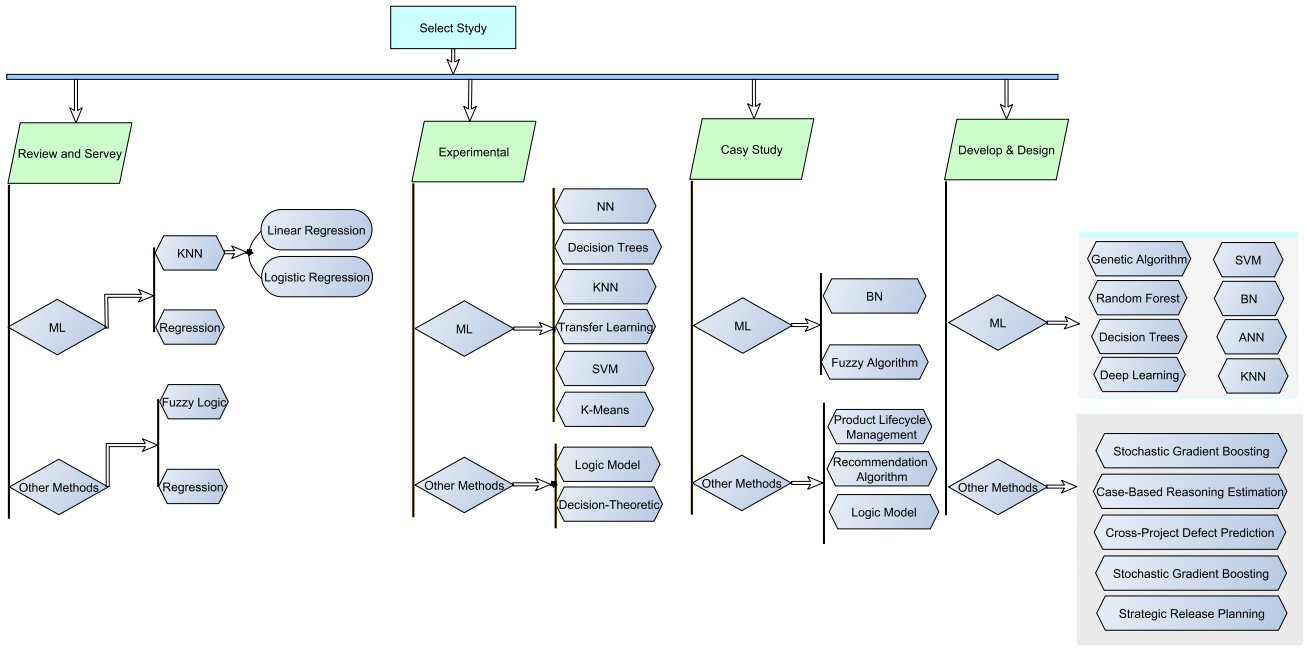

Figure 5. Taxonomy of literature on the Software Project Risk Assessment Using Machine Learning Technique.

3.4. Review and Survey Articles

The analysis and research documentation outlined the latest perception of ML technologies in SPM preparation and evaluation and the application of ML algorithms.

3.4.1. Studies Conducted on Machine Learning and Their Use in SPM

This segment discusses and uses the ML processes. These papers were broken into multiple subjects and implementations. Selected studies have been grouped into large groups, being focused on the ML methods of production techniques. There were three subcategories for the six publications in this group, respectively.

This sub-cluster was conducted on K-Nearest Neighbor Algorithm (KNN). In [20], the observations, metrics, data sets, calculation measures, ML challenges, various models of

forecasts, and ensemble models used in the region of the maintenance prediction were evaluated and analyzed. Another [21] has shown growing concern for ML technologies, with KNN for the management of missing values in information engineering data structures.

Other classification studies were conducted on Regression. The paper [19] identified the volatility prevision methods and predictors and the classification criteria. The characteristics that were used as indicators of literature volatility parameters and forecasting techniques used for boosting the precision of the volatility of prevision requirements have been established. Specifications with volatility are critical for software programs, since they lead directly to costs and overrun period. In [22], the SLR was suggested to help a formal mechanism of repeatable findings. The study cannot settle the precise application by the organization of a data set, like other data sets. Another [18] addressed the usage of the ML methods for calculating the program effort. The systemic study showed that ML approaches, size scales, comparative data sets, assessment procedures, etc., were influential.

One article on the Fuzzy Logic Studies [23] investigated the use of ML methods to test program effort. Additionally, he outlined a number of software work, expense assessments of systems-functioning methods, and the main conclusions were that there should no other methodologies be preferred by process and model.

3.4.2. Other Methods

This segment discusses and utilizes the other approaches. These papers were grouped into different subjects and implementations; listed works are grouped into a particular logical model group.

The author of [24] used the evaluation of financial returns on investment (ROI) network infrastructure . A commodity that does not generate a 'sale' benefit is difficult when implementing a ROI financial principle, as seen in the purchasing or sale of inventories in most academic environments. The paper [17] analyzes recent program maintenance research extensively. The study results showed that the use in maintenance forecasting of ML algorithms has increased since 2005. The problems were categorized according to Project Management Body of Knowledge (PMBOC) fields of expertise and they were also analyzed [25,26]. The problems were examined by using artificial intelligence tools and obstacles in agile PM. The contribution was related to the expected need to develop modern PM models and IT techniques that integrate ML-based methods with the treatment of inaccuracy, vagueness, or ambiguity with crucial performance indicators being correlated with critical areas of expertise.

3.5. Experimental Studies

This section classifies technologies that execute standard measurement tests and experimental parameters that are used in the ML-SPM analysis. These articles are split into different subjects and implementations.

RQ2. Can we build better machine learning-based models in terms of accuracy prediction by applying feature transformation and feature selection to reduce the project failure probabilities efficiently?

3.5.1. Studies Conducted on Machine Learning Methods

Based on the ML approaches of production methodology, the chosen articles were categorized into vast groups. The 27 publications were organised into seven subcategories in the following group. This segment includes 15 papers that are used for SPM with various ML algorithms. Table 1 contrasts the current ML, Definition, Domain, and other core aspects of established experimentally.

Table 1. Studies of Existing Experimental Studies.

Ref	Type of ML	Description	Domain	Feature Extraction	Limitation of Old System	Limitation of the New System
[27]	SVM	Evaluated two ML approaches to boost track consistency between regulatory codes and specifications at the commodity level	Security, and privacy in healthcare domain	Non	Limited success for tracing regulatory codes due to the disparity in terminology that can exist between the codes and product level requirements	Applied the data mining to a more fine-grained model of the HIPAA regulatory codes showing specific rights
[28]	Several type of ML	Argued that information analytics apply computational technologies	Broad spectrum of field experience and awareness	Non	Full machine analytics, software analysis, ML, data processing and knowledge visualization	Expertise to design and implement scalable data processing tools and learning tools
[29]	Several type of ML	Develop machine assessment, maximize the usage of capital	Effort and duration estimation	Non	Plan and commodity historical indicators depending on the learning method	Availability of granular data regarding project and product characteristics
[30]	Several type of ML	Demonstrates a novel solution to address this omnipresent dilemma through a modern synthesis of digitization and ML	Project evaluation, team pace and time estimation	Non	Creation of a waterfall concept about a decade ago	Extended to generate data on individual and team contribution, which can be helpful for management
[31]	Several type of ML	Complementing Agile manual planning poker	Software development effort estimation	Token Extraction	There is no framework for agile growth which is the most suitable	Larger data sets and functions in this experiment do not included
[32]	Several type of ML	Many solo strategies to forecast the software development effort were suggested System	Software effort estimation	Dataset figures include the number of ventures and the number of characteristics	It has been seen to be sufficient in any case	The goal was to evaluate the effect of the number of participants of the ensemble
[33]	Several type of ML	The goal was to reach a solution by implementing a smart device that assigns team members creatively to a specific mission	Software Project Management	CollabCrew ETL	Built primarily to tackle the software issue	Results of this research are a benefit to the real-time framework and provide insight into the efficiency, Precision and level of reliability
[34]	NB and SVM	Provided an extensive comparison of well-known data lters	Cross-project defect prediction	Feature based approaches	Data lter strategy significantly improves the efficiency of cross-project defect prediction and the hierarchical chosen method suggested significantly improves the performance	Find another classifier for the model building other than NB or SVM

Table 1. *Cont.*

Ref	Type of ML	Description	Domain	Feature Extraction	Limitation of Old System	Limitation of the New System
[35]	Several type of ML	Give an active online adaptation model solution to ACONA, which adapts a pool of categories dynamically to different projects	Software development process management; Risk Management	Non	Using well-trained classifications to render good forecasts for the current project with streaming data on vast historical data from other projects	Attains improved outcomes with less concerns regarding the actual CI scheme, which reveals that ACONA can dramatically minimise CI costs more than current methods
[36]	RF, Multilayer Perceptron and SVM	Purpose of predicting the effort	Software project effort	Non-linear features	Accurate estimations of software project effort	Incorporating other ML models like treeboost like XBoost etc. and validating with other diverse datasets
[37]	DT, FL	In certain instances, it provides reasonably reliable figures	Software cost estimation	Feature subsets from ISBSG	Built exact and useful models are constrained in fact even though they give tech stakeholders considerable financial benefits	Models in an area of actual growth
[38]	SVM	The externalised development project is one of the key approaches to build software that has a large rate of failure. Smart risk prediction model can assist in the timing of high-risk projects	Software project	Selected 25 risk factors	Existing models are focused primarily on the premise that all costs of misclassification are equivalent, which does not correlate to the fact that risk prediction exists in the software project region	Applies stronger classifiers to improve the prediction accuracy of outsourced software project risk
[39]	SVM	Investigates the impact of noisy domains on eight ML accuracy and the recognition algorithms for statistical trends	Software effort prediction	Randomly selected feature	Solutions for the problem of noisy domains in software effort prediction from a probabilistic point of view	Extended by considering a more detailed simulation study using much more balanced types of datasets required to understand the merits of STOCHS, especially larger datasets
[40]	K-Means	Used a particular information engineering design strategy to identify faulty software	Global Software Development	Feature Subset Selection	To promote PM software decisions by data mining and produce practical results	Investigation and comparison with other methods for data mining
[41]	DT	Software Effort Estimation is the most crucial task in software engineering and PM	Software Effort Estimation	Non	Given a comparison of ML algorithms to estimate effort in varying sized software	Augmented by applying other ML algorithms and validating with other diversified datasets

Table 1. *Cont.*

Ref	Type of ML	Description	Domain	Feature Extraction	Limitation of Old System	Limitation of the New System
[42]	kNN, DT, and LDA	Intelligent approach to predict software fault based on a Binary Moth Flame Optimization with Adaptive synthetic sampling was introduced	Software fault prediction (SFP)	Frequency of selecting each feature from all datasets using the EBMFOV3	Improved the performance of all classifiers after solving imbalanced problems	Studied the importance of features to enhance the performance of classifiers and SFP model accuracy
[43]	Neural network	ML was named the general neural network regression for the efficiency forecast in practices of apps	Software practitioners	Non	Developers and managers refer to tech professionals' output, which is typically calculated as the size/time ratio	The usage of a radial base feature neural network to forecast practitioners and developer teams' efficiency
[44]	ANN, SVM	Several ML algorithms to predict the software duration	SPM	Non	Evaluated the algorithms according to their correlation coefficient	Prediction operates according to current/past project details will estimate the potential work and length of the project
[45]	Decision-tree	Proposed evolving decisions through an evolutionary algorithm and the corresponding tree for the prediction of device maintenance effort	Software effort prediction	Non	Usage of HEAD-DT to create a judgement treaties-based algorithm that adapts to the maintenance of data Application	Effectiveness of hyperheuristics in evaluating other primary software indicators, data creation in private and public software
[46]	Decision tree	A tool proposed to boost predictive performance of program effort	Software prediction	Four-dimensional feature	Beginnings of better understanding and utilizing decision-making bodies as the part classification of ensemble imputation methods	Incomplete data and machine estimation theoretical and observational analysis
[47]	k-NN	To explore how parameters are more adaptive to their parameters and how often the output of MLs in SEE may be influenced	Software effort estimation	Non	Systemic tests on three data sets were conducted with five ML in multiple parameter settings	Investigating additional ML and data sets; other forms of action-size, including non-parametric ones; and additional window sizes for online learning assessment
[48]	Regression Trees	Cross-company (CC) machine effort calculation (WC) details aim to explicitly utilize CC knowledge or models to predict in WC situations CC data or model data	Software Effort Estimation	Number of ventures with each characteristic	This system will not only use far less WC knowledge than a comparable WC model, but also produce an equivalent/better output	Dycom's sensitivity to parameter values, simple pupils, inputs and separating CC ventures into separate parts

Table 1. *Cont.*

Ref	Type of ML	Description	Domain	Feature Extraction	Limitation of Old System	Limitation of the New System
[49]	SVM	Systematic studies indicate that RVM is very successful in contrast to advanced SEE approaches	Software effort estimation	Account specific features of SEE	It has shown that RVM is an outstanding indicator of SEE and requires more analysis and usage	Using the automated validity evaluation of RVM, three unique case cases were established and the advice on whether the effort needed was suggested
[50]	SVM	The right calculation of effort helps determine which challenges to be corrected or solved in the next round	Effort Estimation	Computed characteristics on the criteria for the classification task dependent on the initial attributes	The development features have been used to construct statistical models that analyze story points for open source projects	Predictions can be enhanced by taking into consideration new features relevant to human development characteristics
[51]	ANN	Calibration methods depend on linear adjustment forms except ANN based non-linear adjustment	Software development effort estimation	Non-normality and categorical features of different datasets	Considered as a base method for the software development effort estimation	Extension to this study, there are other options for the kernel function in LS-SVM other than radial basis function
[52]	K-Means	Clustering approaches are generalized to be used to construct CC subsets. Three separate methods of clustering are researched	Software Effort Estimation	Different features can be used to describe training projects for clustering 1- Productivity, 2- Size effort, 3- All project input and output attributes	Clustering Dycom with K-Means will help separate the CC programs, producing good or better predictive efficiency than Dycom	Clustering processes, simple learners, input project attributes, clustering project functions, parameter values

3.5.2. Studies Conducted on Other Methods

This segment discusses and it does not extend the system of ML; the chosen works are divided into such groups, including models or techniques. These papers in the groups identify various guidelines with parameters included in the assessment according to their analysis.

First Model, Logic Mode. The emphasis of [53] was on enhancing quality attributes, like faults, months, and tension. Parametric model proponents contend that domain-independent models may be adapted to local data. The authors of [54] recommended integrating reference + visualization into enhancements to the project.

Second Model, Parametric Model. The authors of [55] explored the advantages of cost miscalculation methods when developing templates for predictive software failures that utilize mutual repository project information. In this situation, figure out that cost-sensitive schooling does not have points that outweigh the cost-insensitive classifiers.

Third Model, Decision-Theoretic Optimization Techniques. In [56], decision-theoretic optimization techniques were presented that can select the best parameters for a range of workflows. The initial experiments show that optimized workflows are significantly

more economical than manually set parameters. They argue that Artificial Intelligence (AI) methods, such as ML, decision-theory, and optimization, can solve these problems, facilitating the rapid construction of effective crowd-sourced workflows.

3.6. Case Study

This third section analyzes a project, campaign, or company that identifies a situation, recommended solutions, implementation actions, and identify those factors that contributed to failure or success on SPM development techniques using the ML technique.

Articles in this section focus on ML methods, and selected works were classified into broad categories, depending on the ML methods in SPM development techniques. The seven articles in this category were divided into three subcategories.

This segment includes five ML papers utilizing many SPM algorithms. Based on the ML approaches of software development methodology, the work is divided into broad groups. The articles [57,58] focused on improving the predictability of estimation and allocation of effort required for accommodating various client, project management, and development issues. By way of remedy, the need to address these issues of reporting protocols and expertise and ensure blind analysis is routine will be argued. Others [59,60] proposed a methodology for evaluating stakeholders' perspectives, isolating sector topics, and building profiles that reflect the preferences of stakeholders across all subjects. Additionally, software's computing and predictive regression techniques were contrasted.

The articles in this category studies on Bayesian Networks Algorithm. In [61], a solution for value estimation is provided employing a combination of qualitative and ML solutions where a probabilistic model encompassing the knowledge from different stakeholders will be used to predict the overall value of a given decision relating to product management and development. The authors of [62] implemented a model that automatically identifies the relationship between risk factors and mitigation through an intelligent decision support system (DSS). The suggested methodology covers the widely cited current risk management limits, such as a lack of uniform DSS and the link between software risks and mitigation.

The articles in this category were conducted on the Fuzzy algorithm. The articles [63,64] introduced a fuzzy mathematics method into parametric modelling of risk influence diagram to solve severe problem that the probabilities of important events are not easy to obtain. The work describes the relationship of different influence factors in the risk management process of IT projects by establishing the topology structure of risk factors. The findings are contrasted with various assessment parameters.

Studies Conducted on Other Methods

This section reviews the other methods used. These articles were divided into various topics and applications. They are three articles in this category.

The first model, conducted on product lifecycle management (PLM). The authors of [65] presented PLM. The method builds a layer of functionality to allow for the next iteration of PLM around an established PLM network. This new PLM will then be integrated into a digital plant automation ecosystem using the case study of Ford Powertrain.

The second model, conducted on a recommendation algorithm. A new software algorithm is suggested by [66]. First, add a bug-based feature and a specific screening mechanism to validate the applicant fixer, build a network of multi-developer commits by taking a range of comments and promises, position them in the ranking, and then determine the most suitable bug fixer. The outcome indicates that the solution successfully executes the error triage function.

Third Model, Logic Model. Two articles [67,68] used an actual case study in a distributed domain and applied agile testing to a selected team, comparing their outcome with another three groups to determine the impact of involving a client in a testing process to overcome distributed development challenges. However, the group applying agile testing

verified more than 99% of all requests entered into the testing process, a notable difference supporting the productivity of any development project.

3.7. Develop and Design

Fifty-nine papers; is the study of a scheme, structure type, or architectural model to satisfy the requirements of the stage, where the research results on PMS are produced to be addressed and the methodology used by ML.

3.7.1. Studies Conducted on Machine Learning Methods

Selected works were classified into broad categories depending on the ML methods in SPM. The forty-three articles in this category were divided into nine subcategories.

This portion includes fifteen ML process papers that used various SPM algorithms. Based on the methodology of ML in software production, selected works are grouped into large categories.

Applications from the field of SPM were evaluated by the first category of papers [69–73]: behaviours are classified as working and pertinence. It indicated a multi-target learning problem in designing the model for estimating the system effort. This helps to understand the compromise between various performance metrics by creating SEE models that were simultaneously automated by several objective evolutionary algorithms. Naïve Bayes, Logistic Regression, and Random Forests are the strategies that are used in this analysis.

Two articles [74,75] introduced an automated ML-based method of estimating software effort based on task text. An ANN is used to simplify effort estimating functions. Evaluating software SPM from a software company obtains results that exceed the literature involved, and a system that promises to be much easier to integrate into any software SPM tool that stores textual task descriptions relies primarily on textual descriptions of tasks that, unlike various other methods, are nearly always available.

In [76], the authors showed the result of a reflection on applied data mining work social metrics, effort estimation, test case generation, and others. The results of that unofficial analysis were then formalized and systematized into the seven principles and a dozen other tips. The aim is to describe approaches to successful industrial data mining outcomes, but hasten to add that some of these principles may be true for academic data mining.

Research on [77–79] has more correctly established a new hybrid model. The model is ideal, even for a wider variety of activities, since it is usable in one database. Two ML algorithms, ANN and SVM, check the performance of our model. The experiments show a more robust version of our SVM danger forecasting model.

Others [80–82] have often included flippant explanations for discussing situation ambiguity and linguistic considerations to improve the technical reaction to project risk management methods. The current policy is applied to help electricity mitigation and investment. The project's dedication is calculated, and the performance of the planned method is analyzed based on parameters, such as the correct number, mean absolute mistake, source, and relative absolute mistake.

The authors of [83–85] presented a test framework of the source code metrics and chose the best metric set for the model's performance. The cost estimation method is used to test the predictive failure models. The goal was also to resolve these limitations by closing the distance between the revised test outcomes and the potential implementation in the activity of efficient ML algae in the initial project growth initiative life cycle.

Others [86–88] proposed that software models selection system be used by project manages to pick the software process model that is ideally fit for a current project at an early stage of development by utilizing historical software engineering proof. Reflect on the dimension of automated methods and articulate the problem as a sequence classification challenge that has been solved by implementing ML algorithms. The authors of [89] provide an architecture that develops automated failure analysis models using ML classification algorithms to test outcomes from the different techniques for Firefox and Netbeans. They

how that automated prediction models are more efficient in approximating these two parameters more realistically than a variety of baselines for specific loss modes and projects.

This section describes articles that used Artificial Neural Networks (ANNs). Two articles [90,91] on the establishment of ML risk stimulator systems offer the most appropriate risk incentives on requirements, scenarios, and taxonomy tags for the creation of a software project. The study should be viewed independently of any of these taxonomies, since the taxonomies are independent of the danger causes. The authors of [92,93], regarding the application of a software development team's ANN-driven and optimized programs to recognize capacity gaps and prepare the planning and scheduling of SPM skills, facilitate the predicting and anticipation of talent demands based on applied intelligence technologies, articulating staff development resources and techniques. One of the key aims of [94] is to help forecast SCE by utilizing the current ANN learning process. The effects are root average and median proportional magnitude of the error.

Another classification studies Support Vector Machine (SVM). Two papers clarify the work of project gating systems and their usage to clean the building line, the primary category for the CI systems [95,96]. Three heuristics for handling submissions are suggested and checked for the Gating system. I t is displayed that this results from utilizing a high success rate screening and continuous monitoring at a low level. This outcome is strong. The third and final heuristic appraisal tests ML leverage for selection optimization. Others [97–100] have developed the Less Square project risk evaluation model (LS-SVM). The simulation shows that the anticipated SVM outcome is successful. The LS-SVM approach was used to analyze the risk assessment model of the project. The expert risk assessment data are used to train the established LS-SVM regression model for the mapping relationship between danger and characteristics. The findings also show the strong precision and generalization of the LS-SVM model. The last paper [101] suggested that steps should be analyzed and graded by SVM learning methods at runtime. It has defined the mechanism that is indicated by choosing metrics from the existing schedule or reorienting the software of measurements, strengthening calculation programs with the metrics of versatility.

Articles in this section describe Random Forest. Two papers [102,103] have developed an extremely reliable prediction model. In the ongoing software creation phase and research initiative, the approach mentioned in a practice journal shall involve defect prediction. Compare and reexamine the teaching outcomes with the most accurate fault predictor in the sector in multiple classifications, including the NB, DT, or RF. One [104] complete team activity appraisal results, which involves over 40 objective and observable steps that were taken by student groups collaborating on class initiatives; also, the ML framework uses RF algorithm to forecast teammates' behaviours and team outcomes.

Only one article was conducted on the Bayesian network. One [105] used numerous databases to gather metrics that were taken from the design specifications for three separate NASA programs, built for the instruments for the Spacecraft, a real-time ground prediction framework and flight satellite applications. Explore the use in requirement engineering of BN, focusing specifically on identifying and evaluating the risky requirements.

Two articles are conducted on K-Nearest Neighbour (k-NN). The proposed model shown in [106] presents project managers with various places to select the best global manufacturing sites for individual tasks. The proposed job allocation model is also evaluated and checked for other approaches. There is a second [107] hybrid algorithm that combines optimization of COA-Cuckoo and KNN algorithms. The findings suggest that the projected expense is more reliable.

Another two classification articles are conducted on Decision Trees. In [108], a discrete variable was proposed, and a classification model algorithm was introduced. The findings demonstrate that the statistically elaborate policy trees surpass the evolutionarily conditioned and the standard logistical regression. A second [109] analyzes the homogeneity of cost data in the device domains and focuses on the embedding sense. Equatinf cross-domain data models with the domain data model creates three experimental installations.

One paper that was conducted on Genetic algorithm [110] suggested that a NN be used to create a list of experts organized for each criterion. A combination of semiautomatic discharge planning approaches and semi-automatic position assignment has also been implemented. The cumulative outcome is an iteration schedule for the details that the creator operates on.

One paper was conducted on Deep Learning. In [111], there were two reliable deep learning architectures: highway network and long-range memory. The prediction framework involves end-to-end training from rudimentary input data to forecast effects without manual function engineering. The analytical review reveals that the mean absoluteity, median absoluteness, and uniform precision of three criteria baselines and six alternatives are reliably outperformed.

3.7.2. Studies Conducted on Other Methods

This section reviews the other methods and how it is used. These articles were divided into various topics and applications.

First Model, Software Process Models. Two papers [112,113] explored research collaboration research in agile programs. The samples help to identify the conflicts above, including short-term alternatives to short-term adjustments and settle interest disputes using Agile techniques. Two papers [114,115] clarified the compilation of suggestions, the evaluation of the chosen features, and the implementation of an ML kit for the statistical sense of the R. Additionally, explain how data mining techniques can be used to construct a cost loss prediction classification model. The proposed model gathers data from a few project parameters and classifies a project into one of three classes. One paper [116] has proposed a software project risk evaluation approach that uses credits to measure the impacts of risk factors to fix imprecise opinions and inconsistencies among experts. The inference of the credal network contributed to a risk prediction and diagnosis.

In [117], the importance of agile methods and advanced systems, like IoT, Fog, and the cloud is addressed. Therefore, software integrating a design and risk mitigation structure was proposed to better pursue this purpose. ML innovations, which proves to be more favourable to a continuous mode for current steps in business risk evaluation and are implemented with IoT, are also included in the proposed framework. The authors of [118] systematically offer an integrated robust quantile system of risk management that focuses on the partnership between the project size and the engagement of risky decision-makers. The approach is shown in an actual project database through an analytical application.

Second model, Fuzzy Logic Controller. In two to six stages of the software effort estimation, two papers [119,120] suggested an increase in the efficiency of fluent, logical controllers by improving case-based fuel controls with a minimum size on the control basis. The result is the fluid logic controller cascade. The rules for models that are generated by the subtractive clustering allow a further reduction.

Third Model, Strategic release planning (SRP). The authors of [121] found that the software used as a plug-in for frequently used production frameworks helps to improve the process performance. SRP is a crucial phase in the creation of iterative applications. In the form of hard and soft constraints, such as time, commitment, consistency, and money, SRP includes assigning features or conditions to release.

Fourth Model, Maturity Model. In [122], massive unstructured data, which have been developed under the Digital Competency Maturity Model (DCMM) framework through a detailed analysis of the aim, management processes, or influencing factors of this project. The comparison of related data storage skills is projected to have a favourable impact. An established methodology for performance enhancement and the gold standard for program and device creation was built for over 20 years.

Fifth Model, Forward Sequential Weighting. The authors of [123] developed and evaluated efficient algorithms to generalize feature subset selection into a practical feature weighting approach. The algorithm improves the accuracy further, since not all of the features contribute equally in solving the problem to assign a weight to elements to improve

the estimation accuracy. Subsequently, perform experiments that are based on repeated measures design on real-world datasets to evaluate these algorithms.

Sixth Model, Case-Based Reasoning (CBR) Estimation Model. The authors of [124] proposed that the CBR clustering systems (CBR-Cs) could be established to provide an accurate cost estimation. The CBR-C approach aims to quantify errors and time and to allow managers to understand evaluation processes easily. The study reveals that the proposed CBR-C approach provides a comprehensive software project-design cost estimation structure.

Seventh Model, Earth-Moon model. The authors of [125] presented a risk management model for Earth Moon projects, which considered the characteristics of software creation in compliance with the implemented theory of the software life cycle project. The main developments that are used in the current paradigm are often analyzed and solved.

Eighth Model, Cross-Project Defect Prediction. In [126], the data simplification effects were described and quantified. Experiments have been conducted and they have been contrasted with, and without, the predictive ability of CPDP. A method for data simplification was introduced using the adaptive learning method for user interaction calculations.

Ninth Model, Treeboost (Stochastic Gradient Boosting). Treeboost's scale, efficiency, and sophistication are the inputs for the model, according to one report [127]. The Treeboost model was tested using four output criteria: MMRE, PRED, MdMRE, and MSE against the multi-linear regression model and the case point model, and it was established as a multi-linear regression model. The Treeboost model can be used for quantifying software output commitment.

4. Discussion

This research explores the essential studies in state-of-the-art project management utilizing ML technology. The goal of this analysis is to emphasize research patterns in this area. This research is not current and it would not cover implementation, but the literature itself. This study is different from previous estimates. The accompanying literature as taxonomy is suggested. In a research field, creating a literature taxonomy may have several benefits, a shifting one. On the one hand, the taxonomy of literary works is commonly publicized.

RQ3. What are the existing gaps for prospects of research in the field of Software Project Management?

A new investigator who is researching a software project assessment may be overwhelmed by the broad range of documentation in this sector, the absence of a certain kind of framework, and a review of this field—numerous articles on the issue address emerging trends in project management. Certain studies have established ML models and implementations today.

A literature taxonomy helps to organize these diverse works and incidents significantly, being usable and consistent. On the other hand, the methodology for taxonomy gives researchers useful insights into the topic. Next, it describes the future areas for study. The taxonomy in the current analysis of software evaluation indicates, for instance, that researchers appear to recommend a path to be taken in this sector to involve mechanisms for program development and operation. The usage and implementation of ML technologies, including the latest project evaluation, are also covered.

Secondly, a taxonomy can classify study deficiencies. The mapping of the literature illustrates a poor and strong study coverage into project review proposals in various categories. For instance, the taxonomy shows the value of review and assessment of groups of individual claims at the expense of consolidated methods and structures and growth activities (being expressed in the abundance of their categories). The taxonomy also showed the absence of research on the development of project reviews after an adequate inquiry. The literature is essential to research. Studies in this sector are aimed at improving and exchanging ML.

Statistical analysis on individual divisions of taxonomy classify industries participating in the ML methodology to contend with emerging developments and improve dormant fields. This study offers a taxonomy in which scholars may collaborate and analyze new technologies, such as developments, comparative studies, and project appraisal utilizing ML technology, being analogous to taxonomies in other fields. The analysis illustrates three parts of the literature: the factors behind the emergence of automated project management using ML technologies, the challenges to the successful use of those methods, and the recommendations for overcoming these problems.

4.1. Motivations

The advantages of using the project management ML platform are transparent and convincing. This section discusses some of the advantages in literature, being classified according to unique advantages. The appropriate sources are cited for further discussion (see Figure 6).

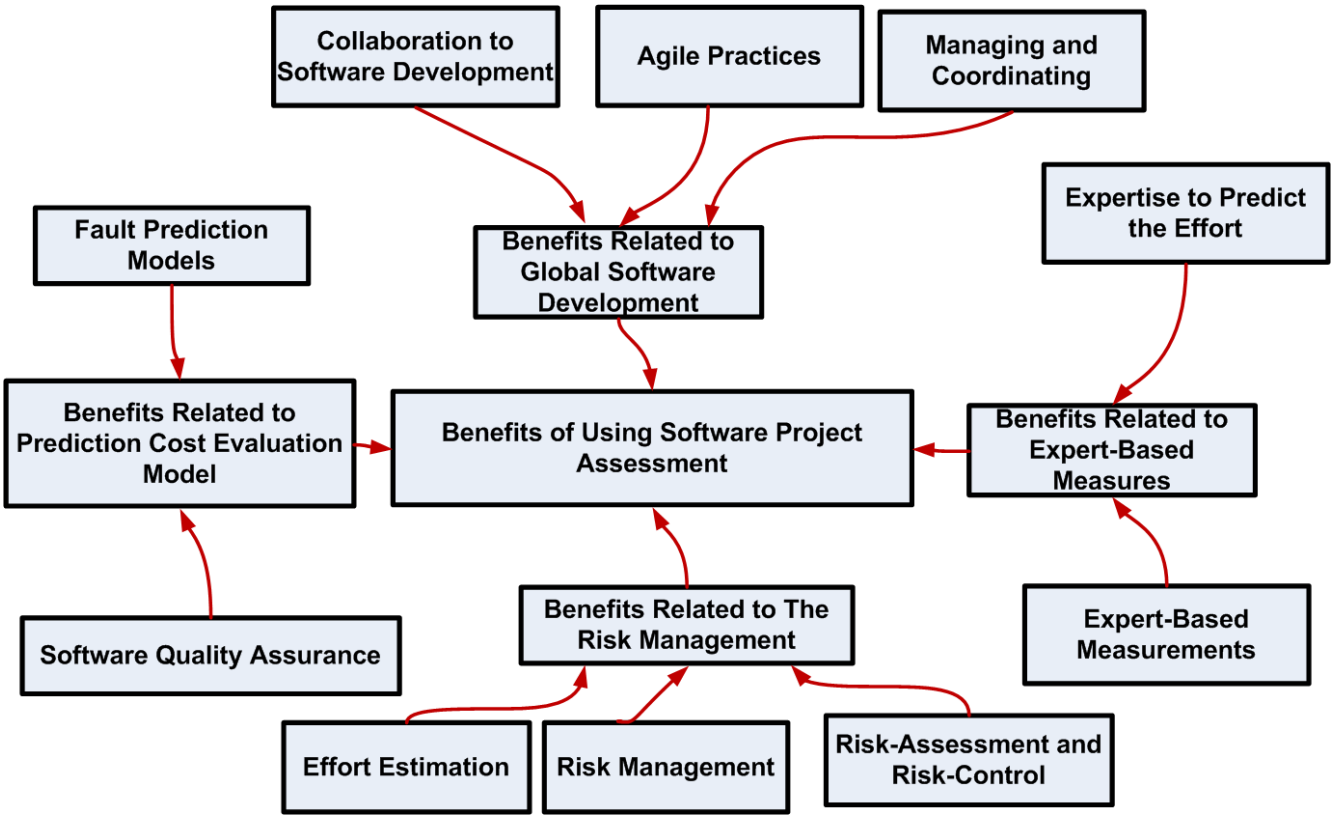

Figure 6. Benefits of Using Software Project Assessment.

4.1.1. Benefits Related to Prediction Cost Evaluation Model

The method was used to test the source code metrics and the right metrics to improve the fault-predicting model performance. The cost estimation method is used to test the predictive fault model. The preliminary results are [128]: (1) several approaches for voting overshadowed by other methods; (2) the source code mechanism metrics selected to use proposed device source metrics, which are helpful in software projects with percentages of error classes below the recommended threshold value, as compared to all other methods; and, (3) failure prediction approach.

4.1.2. Benefits Related to The Risk Management

Different activities of software planning may be divided into two effective practices, namely engagement appraisal and risk reduction [129]. The cost estimate of software effort is dependent on many cost considerations, and risk control requires the detection, treatment, and removal of software risk before unexpected results [130]. The projected commitment to software growth determines efforts. A risk appraisal, the main activity in the project's planning phase [131], is a critical element in evaluating the success of the software development project. However, human judgement and experience are essential in the success of risk management activities in traditional risk-reduction methods, and risk assessments are considered to be unnecessary and costly for the software project [132].

4.1.3. Benefits Related to Global Software Development (GSD)

The increased use of the GSD to minimize manufacturing costs and be open to an extensive package of professional analysi is another critical advancement in the market. GSD ventures frequently pose significant challenges, but they are becoming more popular. These involve connectivity concerns between the delivering project participants, problems in forming acceptable community contacts, cultural difficulties, and obstacles in the management and organization of work in the implementation projects. In short, intensive teamwork in the production of the software remains difficult in a distributed environment [131]. Being centred on face-to-face communication, which is demanding and complex to handle GSD contexts, some agile approaches seem challenging to integrate from the first attempt.

4.1.4. Benefits Related to Expert-Based Measures

The predictor utilizes its experience to forecast the work of ventures, such as experts. The skill of the estimator depends on the issue and his experience of similar and conventional undertakings. The centred model would benefit significantly if the limited number of steps required the expert measurements to be removed. It should not be overlooked the position of expert behaviour. The assessment concept built without professional measures does even worse than the consistency model and quality assurance approach used to prevent assessments that need specialist research—the evaluation model research presents many limitations in collecting the predictive measures for potential work. The potential drawbacks may be the number of specific tools available to assess behaviour, consistency, and adherence to the application of a system or measuring accuracy [132].

4.2. Challenges

Although computer training techniques that are used in SPM evaluation offer several advantages, such technologies are not considered to be the ideal solution for evolving projects [133]. The surveys indicate that the researchers are interested in assessing projects and their use of ML strategies. The main obstacles in the implementation of ML techniques are listed below, in addition to additional subjects. The difficulties are defined (see Figure 7).

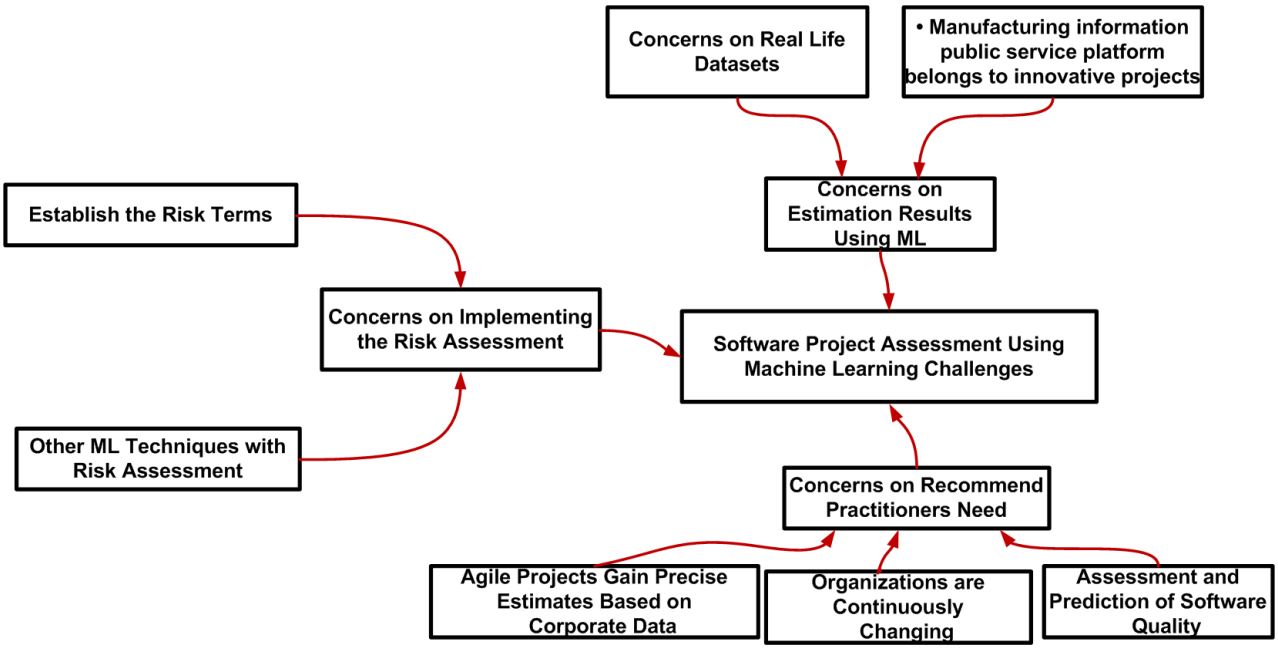

Figure 7. Software Project Assessment Using Machine Learning Challenges.

4.2.1. Concerns on Estimation Results Using ML

The analysis uncovered more questions regarding real-life data sets that lack fundamental software development approaches and need other accurate metrics that can be used to calculate the effort. Various evaluation approaches are required to verify the findings of program effort predictions in more research. Cross-validations, the Jacknife approach, and the Iterative method are mainly validated. In addition, study trends have shown that calculating methods need to be examined and augmented. In addition, real-life data sets may also research size metrics and other ML methods, including regression trees. The public service portal belongs to creative initiatives on manufacturing knowledge. Development procedures must be updated more often by our project team, and engineers cannot monitor the development platform and infrastructure that has a higher effect on product efficiency. Software Project Assessment Using Machine Learning Challenges.

4.2.2. Concerns on Implementing the Risk Assessment

Concerns on the risk assessment implementation: when the project starts. First, we need to define the project threats and risk factors and establish the main risk terms, i.e., risk demand shifts, the danger of infrastructure, coordination of personnel, and device protection approaches, etc. The risk evaluation network, coupled with the expertise of experts, is focused on the current cases to create the risk case learning process. In light of the two systems that have obtained the highest success results, it was also necessary to evaluate the degree of risk of one condition that the costs of revising its level of risk became the most appropriate framework for the Bayesian classification. It is recommended in this field, since, for other techniques, like ML, it provides improved estimation results [12]. There is comparatively more definite proof of higher precision, as data are extracted from a single entity dataset into various homogeneous categories, depending on the enterprise or sector. The accuracy of effort estimates was increased through stratification [134–136].

4.2.3. Concerns on Recommend Practitioners Need

Be careful regarding the kind of calculation methods that are used and the sort of data sets they use for their projects. Agile projects gain more precise estimates based on corporate data. When compared to national data, private businesses profit from a limited amount of internal project data [132]. As the company implements more productive and agile gradual growth methods, like XP, further reports are suggested on projects that apply

the technique. Current studies show relatively better effort management in XP projects, as demonstrated by low MMRE, despite retaining the same productivity [137].

Because organizations are continuously changing, Adaptive-project-control systems will be needed. New KPIs to be used or deleted, changing expert evaluation rules, are commonly known scenarios among project managers. PMIS needs to handle that kind of situation. Establish well-defined automated processes between high management levels of the organization and the PMIS internal settings, promising new relationship features in the way on which these two entities communicate between them [138].

Regarding the assessment and prediction of software quality in large software organizations, the general background for using ML methods in large computing enterprises is for the evaluation and estimation of product quality. The various product indicators can be used in the creation of the software quality model using ISO 15939 to measure and forecast the output of the software as well as to satisfy the quality information criteria of these organizations. The usefulness of ML methods is even documented for such an evaluation. In ISO 9126 [139], quality is defined as "the completeness of the features and features of a software product that is capable of satisfying stated or implied needs. In ISO 25000 [140], the capacity of software products to fulfil specified, indicated specifications under some circumstances" is the consistency approach adopted. To identify and allocate resources where most of the software is necessary to assess software quality early on in the development process [141].

4.3. Recommendations

This section presents some recommendations for solving the issues and challenges in the SPM assessment used in the ML technique (see Figure 8).

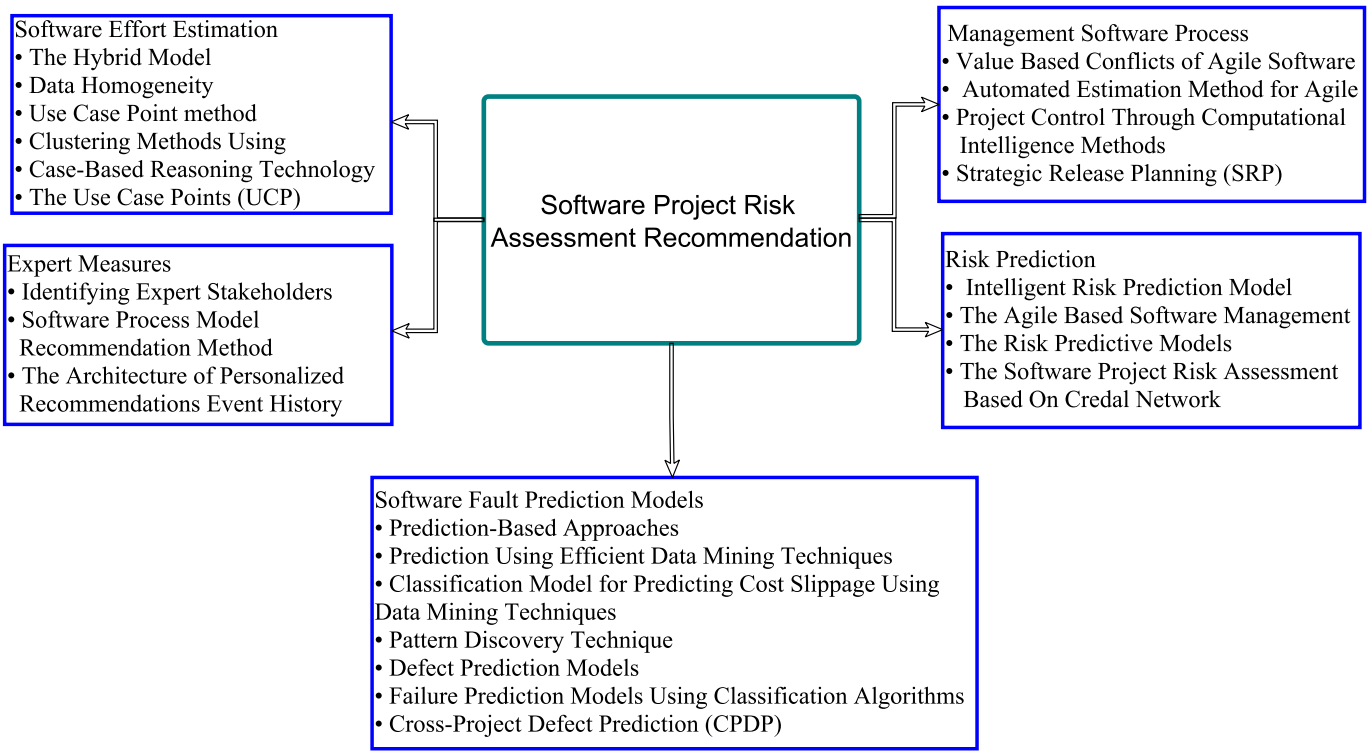

Figure 8. Categories of Recommendations for Using Software Project Assessment.

4.3.1. Recommendations for Software Effort Estimation

Measuring software participation is essential for successful software programs. Precise computing effort estimates are necessary if the preparation and spending of the project are to be proper if a specific budget is to be sought in software businesses. Overestimated

business gains can be missed, owing to price changes. On the other side, underestimation will overwhelm schedule and expenditure, costing the business a ton of capital. Becayse the expense of the effort reflects the high cost, the literature, instead, utilizes software effort calculation and software cost calculation terminology that refers to the approximate value.

RQ4. What are the prediction metrics and their current level of accuracy evidenced by different estimation techniques?

The Hybrid Model, for a long time, the estimation of completion of program activities, is an essential field of study. Therefore, we wanted several well-performing independent models to improve the precision and reliability of the prediction effects. A hybrid prototype includes three separate sets of attributes: (1) one is a text-based attribute known as summing; (2) one includes text-based attributes known as a definition; and, (3) and that has a variety of metadata-based characteristics. In comparison to previous model models in [137], the model is also suitable for a more extensive number of tasks, as it is not limited to just one kind of data source that is not always available. This segment concentrated on estimating the completion of activities, but further research is required to explore whether the same hybrid strategies extend to completion time estimation models.

Data Homogeneity the homogeneity of application domain cost results. This question was not previously discussed in the literature on machine estimates. Investigate the impact of the prediction success training data scale, and a vital topic that is explored in another research. The studies carried out to address this problem have yet to confirm that this is an unanswered issue due to differences in data quality and prediction algorithms. However, the latest experimental findings will direct project managers to determine how much data are necessary to train the algorithm [109]. Take advantage of different ML methods to measure program costs and analyze performance. It may also encourage and direct other researchers to perform work in this area. The potential work approach could aim for domain-specific attributes, such that the attributes' data quality becomes better and that their prediction output increases. That may have been achieved by the data analysis in the embedded device domain.

Use the Case Point method Treeboost model, centred on the three independent variables software scale, efficiency, and sophistication, foreshadows software commitment. The Treeboost model suggested forecasting program action based on the use case point process. The Treeboost model was used. Software size, productivity, and complexity include the inputs for the model. A multi-linear regression model was created, and the Treeboost model was evaluated using four performance criteria: MMRE, PRED, MdMRE, and MSE against the multi-linear regression model and the case point model. The Treeboost patternwas used for program effort with a positive performance [22]. The findings are positive and they will boost the accurate measurement of early tension.

Clustering Methods have a range of possible avenues. Additional clustering methods, simple students, project input properties, clustering functionalities, parameter values, and (automated) tuning protocols may be addressed as well as recommendations of a more organized update mechanism for CC projects [142,143]. Dycom extended the use of clustering approaches in the creation of the CC subsets. There are three methods of clustering, i.e., the hierarchy of clusters, K-means, and the maximization of preferences [144–146]. The Dycom Clustering is identical to the original Dycom, which is based on four CC sub-sets of four different sizes SEE tables. The research contains a toilet layout, for example. The K-mean clustering Dycom assists in separating CC programs, with the delivery being equal to Dycom or having better predictive efficacy. However, the number of CC subsets also needs predefinition, and a wrong choice may have a negative effect on the predictive results.

Using Case-Based Reasoning (CBR) Technology, such factors inspire one to suggest an interactive CBR for addressing the needs of managers. The method seeks to promote the comprehension of the cost estimate mechanism and provide managers with sufficient knowledge. The CBR cycle is simple to grasp, because it simulates the storing and retrieving of information in human memory. Managers can better understand how the calculation was

rendering with the measures that they have taken. The ANGEL method was outperformed by the CBR-C strategy [147]. The principal explanation for this is that the current CBR-C approach has an indexing function that fairly declines the prediction error. The CBR method is to overcome current issues by addressing related problems in the past. The estimation of CBR commitment is detailed. The basic rules of CBR are as follows. The most comparable past projects are selected to estimate how the expense of the current project can utilize comparison metrics [143]. 1. Is there a form of function weighting algorithm for evaluating casual reasoning effort that enhances the current function-selection methods? 2. How much does the data set rely on these results? 3. Can the accuracy of data sets be measured by seeking ways of demonstrating that our findings are accurate or free from the noise? 4. What to describe weight check phase size with characteristics of a dataset?

The Use Case Points (UCP), due to their profit having an approximation at the initial phases of product growth, is used. Before training models on the data collection, the pre-processing standardization strategy implemented. The estimation performance of the RF, MLP, and SVM models was assessed [148]. The UCP metric was used to measure commitment using universal modelling language (UML) case diagrams. . The UCP allows for predicting the program in the early stages of the design process. The uniform scale, efficiency, and difficulty values are the critical inputs for all models to forecast the last attempt. The RF technology and the input parameter were used as the previous reference for the simulation of the RF for effort prediction [149]. The research can expand by adding certain ML types, such as tree boost types, such as XBoost, etc.

4.3.2. Recommendations for Expert Based Measures

The expert's knowledge, expertise, experience, and intuition are based on the know-how of project groups in 10 fields, like prices, time, distance, productivity, and resources management (often contributes to over-optimistic estimates). This represents the compilation of processes, policies, and practices that create and gather the knowledge that is needed to execute a project. Two layers of knowledge—micro and macro—were established.

They identified expert stakeholders and the methodology that was used to assess stakeholder inputs and identified market topics that build profiles to reflect stakeholder involvement in each subject. Subsequently, considering the fundamental contributing factors of the particular stakeholders, often examine solutions in a broader range of initiatives rather than viewing each contributed equally. The defined approaches in [150] are focused almost entirely on the contents of the objects allocated to each topic. In certain instances, random word use requires the distribution of small series values for conceptually similar subjects. Shipping and monitoring subjects, for example, were not related, even though technically connected. Cooperation filtering techniques can mitigate this problem by constructing the communities of similar stakeholders to determine whether the stakeholder will inform of the targeted issues.

In the software process model recommendation method, project managers are advised to use the best software cycle model for the current project based on existing data science proof at an early point in the implementation process. In the project management approach, the framework challenges the community with the recommendation for the protocol structure. Secondly, it analyzes differences in alternate classification and selection algorithms, being accompanied by a proposal model comprising historic software creation figures to estimate a modern software project process model with only some details [151]. The use of this form of information entirely: (1) assists the project managers in selecting the right software system for the current project at a sharp point in the creation phase; (2) evaluates the reciprocal effect between phase models and various forms of factor project, to allow the project managers to decide the most suitable process model; and, (3) uses the best approach. An automated recommendation framework focused on the ML software process model is used to enable project management to decide, according to historical software engineering results, which software process model is better fitting in the early development phase for a new project [152].

The architecture of personalized recommendations offers an interface that helps data research teams to effectively engage in ML initiatives. The output gives millions of applicants' custom feedback, answers queries in less than one second, and brings new knowledge. Therefore, it runs on Antelope's open-source implementation, integrating the idea of case history, a flexible information engineering platform. It may also use a range of ML tools along with technology, which is developed here and that can allow for deeper integrations, even some that only have an interface with traditional data management systems [153].

4.3.3. Recommendations for Management Software Process

Agile strives to minimize the thin consistency impact by first generating the most important features. In comparison, architecture projects in larger project environments also face output problems provided overall system specifications that are perceived to be fair. This could apply to agile societies that clash with non-agile crops that trigger conflict between agile and non-agile levels [154]. Trends may better identify a disagreement to provide a short-term remedy with short-term improvements and a long-term value conflict-resolving plan when following the Agile approach. Different cultures are now converging and interacting, and other religious systems intersect. This can include tax, technological, functional, corporate, or communicative relations [155].

Value-Based Conflicts of Agile Software Agile systems are autonomous. It exists in the literature that an organization is a critical factor in the development of agility. The literature indicates very little support in doing what should be achieved if an agile project does not impact or modify its project environment. No guidance is provided as to which disputes and interferences may be monitored and managed. This approach incorporates agile product development, value-based conflicts, and institutional initiatives by [113], which recognize problems and solutions. My experience is that of an Agile software development team that functions in a non-project atmosphere and may not have a significant effect and/or modification. The ideals of people and institutions are part of the environment and, therefore, agile concepts must be evaluated.

The automated estimation method Agile efficiently applies the automatic card estimation method to historical predicted human data with the latest ML algorithms [31]. The "Auto Estimate" approach boosts the most popular form of manual poker preparation in agile environments [154]. Self-estimation utilizes story card features in an agile environment: (a) maximize the precision of estimation by reducing the effect of wrong estimates; (b) show that auto estimation increases poker preparation in the latter part of the project; and, (c) determine the value of writing properly designed story cards.

Project control through computational intelligence methods is linked to numeric and linguistic data management, calculation error noise, human appreciation, and ambiguous decision-making principles. It also explores new ways and technical instruments for the administration of ventures and open access applications in recent decades for computer intelligence. There is also an overview of emerging patterns and places to develop, evaluating niche sectors of strong thematic applicability [25]. The input refers to the projected need to establish modern project control models and IT resources that involve ML-based frameworks and information imprecision care, vagueness, or ambiguity by the main measures of success that is linked to the entire fields of knowledge. The introduction of modern learning assessment libraries and open-source development frameworks for project management opens up an area of study that is relevant to the technical convergence of IT resources [156,157].

Strategic Release Planning (SRP): a critical step in the growth of iterative software. SRP includes a delivery, like compounding, soft controls, including time, resources, price, or money, of release characteristics or requirements. The SRP-Plugin reveals that applications used with the common application help to improve the productivity of the development process [121]. The plugin boasts a rich visual space ecosystem with advanced release preparation capabilities, enhancing the capacity to prepare for launches, increase viability,

and boost collaboration amongst project stakeholders. With a robust, detailed, and organized approach and the ability of Release Planner to produce complex release schemes, SRP-Plugin enhances the Visual Lab. However, the preparation of release schedules is just the first phase in complicating ideas that offer broader guidance for strategic release planning decisions.

4.3.4. Recommendations for Risk Prediction

Risk management is necessary if software-related projects are to improve their efficiency independent of their field of business. Consumer expectations are still not addressed in proven systems subject to rigorous monitoring [158]. Determination of risks to the initiative is a critical consideration in the assessment of project success or regression. Nearly every organization utilizes sophisticated instruments to classify, minimize, and remove damage altogether.

Intelligent risk prediction model: will a high-risk initiative be detected in time? However, the current models are primarily focused on the premise that all costs of error classification are equivalent and that the probability estimation is in the software project. The expense for forecasting a failed project as a project that is likely to achieve success is different from predicting a project that is likely to succeed as a failed project. To the best of our understanding, while it is commonly utilized in several areas [159], the cost-sensitive learning approach is still not applied in the context of outsourcing software project risk management. In the study area of the software project risk prediction model, there are two major research holes. Firstly, risk prediction models that are unique to an outsourced software project are seldom explored. Second, even though software project risk prediction studies are comprehensive, no researchers have applied cost-sense-sensitive learning methods on software project risk prediction.

The Agile-based software management for the bulk of the project's performance was productive in recent times. Time-related threats, which impact the release of deliverables, represent schedule-based threats. This is because the finances, the unreliable forecasts of the time, and the positive actions of the project manager are not appropriately distributed. Budget threats reflect financial hazards that may occur from funds swarming. Such factors may be attributed to unintended project reach extension, low usage of existing outlets, and poor management [160].

Operational danger forms are associated with regular project procedures. Wrong procedures, inadequate planning, and team strength are the reasons for these threats. Discuss the value of agile methods and the application of modern frameworks in risk control by adequate resources. In the future, that would enable the team to determine the effects of the threats by evaluating the risk parameters. There is also a strong probability of sound output if such criteria are used. The predictive risk models: a 50% estimate of the software venture delays variant decreases the number of prompts for the 'keyword' scan. The accuracy of the received Bayesian models is measured and contrasted using several classification scales. An optimized network architecture for the tree illustrates successful experimental performance for all data sets. The relationship between the obtained variables, also specified by necessity engineers, to determine the danger level in a situation. Bayesian networks are essential methods in necessity engineering for risk management automation. Risk management in software development aims to detect, measure, prepare, and react to potential risks to avoid their influence on the software project.

The software project risk assessment is based on Credal network, a modern method introduced to assess the vulnerability of a project program focused on a credal network. It will contend with experts' contradictory views and their differences by using a credal set to measure the impacts of risk factors [116].

The credal network inference conducted risk prediction and risk diagnosis. The case results indicate that the method principle is correct and that the software project evaluation is well predicted. Threats are modelling aims to predict the risks and implications of hazards and to define the main risk factors that promote a risk strategy and risk manage-

ment. However, a long development period, high product sophistication, and tremendous method instability render it impossible to forecast and evaluate the danger of the software project. The current software project risk assessment philosophy often analyzes risks, when considering the features of software systems and information development methods, from general project management.

4.3.5. Recommendations for Software Fault Prediction Models

The prediction of software development effort and duration is the critical task for active SPM. The accuracy and reliability of prediction mechanisms are also important. Several ML algorithms were used to predict the software duration.

Prediction based approaches require a forecast feature that forecasts the potential project commitment and period according to current/past data of the project. Nonetheless, numerous ML algorithms are often not evaluated despite the vast number of ML algorithms. For the construct machine model, based on several project details, various ML algorithms are used [161]. The ML model was dubbed the global neural regression network (GRNN) to forecast tech professionals' efficiency. GRNN could be utilized for a prediction of practitioners' 'productivity for new and modified lines of code, codes, and programmers'. Experiences as independent variables predictive accuracy of GRNN used better than statistical regression when the two models are used to predict the productivity of individually devoted software practitioners.

Prediction using efficient data mining techniques for this portion of analysis measures the success/failure of projects with a phase-by-phase calculation, instead of the regular assessment style of the whole project aspect. The data-mining method is used to cascade clustering and grouping methods by gathering data from various initiatives through several computing sectors. They also suggested that the project completion was increasing and that the failed project be successful. To foresee the severity of a program error, Nagwani and Bhansali introduced a modern GUI model [162]. The clustering approach was used to build a software bug repository severity cluster with a problem fixing length. The suggested model is applied using the open-source code, which is often provided by MySql's open software issue repository [163].

The classification model for predicting cost slippage using data mining techniques uses the budget and schedule for the initial planning of an ICT project, and then forecasts a cost slip in the project category. There are three categories of fall that are deemed natural, of medium slip, and high fall, which require action [115,164]. The goal is to explain how a classification model is built using data mining techniques to forecast cost losses. The proposed model uses the input (for example, the initial budget and schedule) of a limited number of project parameters and divides a project into one of three categories (regular, medium, and large).

The pattern discovery technique is to implement an experiment in computational biology with a pattern discovery strategy effectively implemented. The technology exposes habits of relationship that are inherited from the records, enabling business practitioners to obtain meaningful knowledge and improve trust in decision-making. Statistically relevant trends with good grade results were generated for the data sets tested [165]. It also shows the effects on the results of numerous budgetary techniques. The first research uses the unique patterns mining methodology for faulty software identification in software engineering. The findings have shown the ability for such a strategy to offer positive ranking performance and helpful knowledge for decision-makers.

Defect prediction models creating software project prediction models is valuable for raising the effort to detect defects the development of a model of defect prediction for a large industrial software enterprise. The system and method measurements for model creation illustrate that, even though four percent of the program is flawed, the goal is to incorporate a malfunctioning factor into a massive program project's continuous development process and reduce the evaluation initiative. A study of experience offers the latest strategy [166].

The model provides a highly reliable fault prediction for a four percent faulty program. The model utilizes the RF, which is more reliable than NB, Logistic Regression, and DT.

Failure prediction models using classification algorithms of a framework implemented using ML-classification algorithms automatically construct failure prediction models and compare the performance of the different techniques for the Firefox and Netbeans projects. The calculation is based on a cost–benefit model to determine the importance of additional initial research. The importance of further early-stage research in this model was based on its probability performance in avoiding failures on the relative cost of faults that are related to its costs [167]. The rational projections of the two numerical forecast parameters give better utility for some fault forms and programs than a collection of baselines. That indicates that automatic failure prediction may be a beneficial solution to guidance requirements development activities in online environments during the elicitation of online requirements.

Cross-Project Defect Prediction (CPDP) is the research area in which data from other programs can be used by a software project with inadequate local data to construct fault predictors. The cross-project details must carefully screen to help CPDP before local implementation. Several specific CPDP efficiency enhancement filters have been developed and introduced by researchers [168]. However, the data filter technique, in general, and especially in CPDP, is still uncertain. It demonstrates that the data filter technique dramatically increases the efficiency of cross-project fault forecasting and that the hierarchical chosen filter is even more vital. Additionally, the defect predictor dependent on cross-project data may bypass the predictor trained using internal project data by utilizing the correct data filter technique. In reality, CPDP is required, because it uses labelled source/project data to construct a model and predict a fault for a target project [84].

RQ5. Which machine learning algorithm tends to overestimate and which tends to underestimate?

Software fault prediction Based Accuracy: the early prediction of faults in software by applying a specific prediction technique may minimize the cost and effort. Different ML techniques have been used for fault prediction, and they are proven to be helpful. Table 2 consolidates the prediction accuracy that was reported by primary studies. It is encouraging to note that, of the 111 preliminary studies covered by the SLRs, nearly 22 report model accuracy. SVR and KNN are the most widely used metric, with nearly 95% of the studies reporting their accuracy. There is relatively more evidence on better accuracy when data are segregated across different homogeneous groups based on organization type or industry type, rather than confining to a single company dataset. Stratification has improved the accuracy of effort estimates. We have compared the estimated result in varying software among algorithms. These algorithms can be used in the early stages of the software life cycle and they can help SPM to conduct effort estimation efficiently before starting the project. It avoids task overestimation and underestimation, among other benefits. Software size, productivity, complexity, and requirement stability are the input factors of these models.

Table 2. Software fault prediction Based Accuracy.

Ref	Type of ML	Datasets	Model	Achieve Prediction	Advantages	Limitation
[73]	kNN	IBM commercial projects called RQM and RTC	Hybrid model uses three independent attribute sets (1) early metadata based attributes, (2) title and (3) description of software tasks	Accuracy 88%	Automatic effort estimation to a larger number of tasks	Datasets of this study did not have historical snapshots to make sure that the final value of included attributes for all tasks are equal to their value before they were assigned to a developer
[48]	Logistic linear regression	KitchenMax CocNasaCoc81 ISBSG2000 ISBSG2001 ISBSG	DYCOM	Accuracy 66%	Made best use of CC data, so that can reduce the amount of WC data while maintaining or improving performance in comparison to WC SEE models	Investigation of Dycom's sensitivity to parameter values, base learners, input features and techniques for splitting CC projects into different sections
[72]	Naïve Bayes	Data sets University Student Projects developed in 2005) (USP05-FT) and USP05-RQ	Software Effort Estimation	Accuracy 87%	Based upon ML techniques for non-quantitative data and is carried out in two phases	Efficiency of other ML techniques such as SVM, Decision Tree learning etc. can be used for effort estimation
[47]	K-NN	PROMISE Repository	Software effort estimation	Accuracy 92%	Investigate to what extent parameter settings affect the performance of ML in SEE, and what learning machines are more sensitive to their parameters	Investigation of other learning machines and data sets; other types of effect size, in particular non-parametric ones; and other window sizes for the evaluation of the online learning procedure
[60]	SVR	NASA93 dataset	Software Effort Estimation	Accuracy 95%	Conduct a comparison between soft computing and statistical regression techniques in terms of a software development estimation regression problem	The need of more future research work to evaluate the efficiency of soft computing techniques compared to the popular statistical regression methods, especially in the context of software effort estimation
[81]	ANN	NASA 93	Experiments Models	Accuracy 95%	Examined the effect of classification in estimating the amount of effort required in software development projects	Implemented a model to estimate the final amount of effort required in new projects, to estimate the partial effort at various stages in the project development process
[37]	Fuzzy logic	ISBSG, COCOMO and DESHARNAIS datasets	HYBRID Models	Accuracy 97%	Addresses the issue of Software cost estimation proposing an alternative approach that combines robust decision tree structures with fuzzy logic	Investigate a wider pool of type of attributes, such as categorical attributes, and concentrate mostly on those that are available at the early project development phases, to address the issue of proposing better and more practical cost models

Table 2. *Cont.*

Ref	Type of ML	Datasets	Model	Achieve Prediction	Advantages	Limitation
[109]	SVR	International Software Benchmarking Standards Group (ISBSG) repository	Data homogeneity	Accuracy 98%	Investigate the homogeneity of cost data in terms of application domains, and to focus on the embedded domain	Data collection process in embedded systems domain may focus on searching for domain specific attributes, so that the information content of the attributes becomes richer and as a result prediction performance of the algorithm improves
[107]	KNN	KEMERER, MAXWELL, MIYAZAKI 1, NASA 60, NASA 63, NASA93	Software Cost Estimation (SCE) models	Accuracy 91%	Model-based methods use a single formula and constant values, and these methods are not responsive to the increasing developments in the field of software engineering	Has not a good performance compared to the comparative algorithms, and its reason can be the lack of consistent data
[89]	SVR	ISBSG dataset	Software project estimation	Accuracy 72%	Narrow the gap between up-to-date research results and implementations within organisations by proposing effective and practical ML deployment and maintenance approaches by utilization of research findings and industry best practices	Focused on verifying the proposed approach through proof-of concept with different organisations to validate the model's accuracy and adjust the deployment and maintenance framework
[46]	Decision tree	Kemerer Bank Test equipment DSI Moser, Desharnais Finnish, ISBSG CCCS, Company X	Software effort prediction	Accuracy 92%	Improving software effort prediction accuracy by generating the ensemble using two imputation methods as elements	In terms of the training parameters and the combination rules that can be employed. Second, empirical studies of the application of MIAMI to datasets from other areas of data mining should be undertaken to assess its performance across a more general field
[92]	Neural networks	Historical data	I-Competere	Accuracy 93%	Presented a tool developed to forecast competence gaps in key management personnel by predicting planning and scheduling competence levels	Centered on the inclusion of other types of projects in order to prove that the proposed framework can be adapted when predicting competency gaps in different projects

Table 2. *Cont.*

Ref	Type of ML	Datasets	Model	Achieve Prediction	Advantages	Limitation
[94]	ANN	ISBSG datasets	Software development effort estimation	Accuracy 97%	Investigated in conjunction with feature transformation, feature selection, and parameter tuning techniques to estimate the development effort accurately and a model was proposed as part of an expert system	Suggested model will be used on new datasets they become available for experiments and our analysis
[166]	Logistic linear regression	Cross-Project Software Fault Prediction Using Data-Leveraging Technique to Improve Software Quality	Source + target	Accuracy 95%	Building a predictive model using instant-based transfer learning through the data leveraging method	Include more datasets from the same domain and by applying other machine algorithms by comparing their results
[101]	Random Forest	Real data	Defect Prediction	Accuracy 90%	Building a defect prediction model for a large industrial software project	Implement model as an online algorithm, which learns with each release
[55]	Random forest	13 data sets	Misclassification cost-sensitive	Accuracy 95%	Analyze the benefits of techniques which incorporate misclassification costs in the development of software fault prediction models	Indicate that in projects where the exact misclassification cost is unknown, a likely scenario in practice, cost sensitive models with similar misclassification cost ratios are likely to exhibit performance which is not significantly different
[108]	Decision tree	Company effort data set	Evolutionary-based Decision Trees	Accuracy 64%	Employing an evolutionary algorithm to generate a decision tree tailored to a software effort data set provided by a large worldwide IT company	Determine its effectiveness in estimating other important software metrics, in private and public software development data sets
[83]	ANN	Experiments on 45 open source project dataset	Fault prediction model	Accuracy 98%	To validate the source code metrics and select the right set of metrics with the objective to improve the performance of the fault prediction model	Reduced feature attributes using proposed framework

Table 2. *Cont.*

Ref	Type of ML	Datasets	Model	Achieve Prediction	Advantages	Limitation
[42]	KNN	Several dataset	EBMFO	Accuracy 89%	Enhanced Binary Moth Flame Optimization (EBMFO) with Adaptive synthetic sampling (ADASYN) to predict software faults	Study the importance of features to enhance the performance of classifiers and SFP model accuracy
[86]	SVM	Quanxi Mi data set	Defect management (DM)	Accuracy 97%	Focused on the procedure aspect of software processes, and formulate the problem as a sequence classification task, which is solved by applying ML	Investigated extra aspects of software processes and other ML techniques to develop more advanced solutions
[77]	Random Forest	NASA namely CM1, PC1 and JM1	Software Effort Estimation	Accuracy 99%	Investigate the apt choice of data mining techniques in order to accurately estimate the success and failure rate of projects based on defect as one of the modulating factors	Process of project estimations and henceforth improves the quality, productivity and sustainability of the company in the industrial atmosphere

5. Conclusions

The literary analysis concluded that extensive study in software project management on ML methods was done. The spread of jobs over the years has been continuous. ANN, Fuzzy Logic, Genetic, and Regression Algorithms are the critical ML methods of automatic effort estimation. The precise calculation of effort is one of the leading software development practices. The software was specifically influenced by time and difficulty. Basic themes may be drawn from various ML works in software project management. These ventures are classified into roughly four groups: the first group involves reviews and surveys related to software project management; the second group covers papers that focus on case studies of software project management methods; the third category comprises of experimental publications that have been used in the management of ML, a type of structure or architectural model; and, the final group of the research contribution study is an analysis of a project, form of a structure, or architectural model. An in-depth review of these articles would help the software project management to review the ML approaches to define and explain the threats, advantages, and recommendations. However, because of the massive amount of ML algorithms, various machine study algorithms remain unanalyzed. The reasons for using automated SPM, the problems of project preparation assessment, and ML engineering technologies are then investigated based on literature findings. Although the literature on SPM explains the performance of projects and loss, there is a lengthy tradition of disagreements over whether project progress should be calculated. There are conflicts of opinion regarding what reflects the development of a project and how it is estimated; these guidelines will resolve the problems facing a software project in ML methods and open up work possibilities in this sector. Research has still to explore the estimation of effort based on ML approaches that focus on risk assessment. Another factor is using standard filtering methods to minimize the problem by creating districts with the same stakeholders and predicting whether a stakeholder is aware of the issue. This literature review provided preliminary answers to essential questions on Software Project Management Estimation that is based on ML.

Author Contributions: Conceptualization, M.N.M. and M.H.M.Z.; methodology, M.N.M. and M.H.M.Z.; validation, M.N.M. and M.H.M.Z. and A.R.A.; formal analysis, M.N.M. and R.I.; investigation, M.N.M. and Y.Y.; resources, M.N.M., M.H.M.Z., L.K.C. and M.S.B.M.A.; data curation, M.N.M., H.N. and H.H.N.; writing—original draft preparation, M.N.M. and M.H.M.Z.; writing—review and editing, M.N.M., M.H.M.Z., H.N., and A.R.A; visualization, M.N.M. and H.N.; supervision, M.N.M. and M.H.M.Z.; project administration, A.R.A. and M.H.M.Z.; funding acquisition, M.H.M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is generously funded by the Universiti Tenaga Nasional (UNITEN) and UNITEN R&D Sdn Bhd, under TNB Seed Grant project U-TC-RD-19-10 titled "Smart Software Projects Management Assessment System.

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. Oun, T.A.; Blackburn, T.D.; Olson, B.A.; Blessner, P. An enterprise-wide knowledge management approach to project management. *Eng. Manag. J.* **2016**, *28*, 179–192. [CrossRef]
2. Maimone, C. Good Enough Project Management Practices for Researcher Support Projects. In Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning), Chicago, IL, USA, 28 July–1 August 2019; pp. 1–8.
3. Saleem, N. Empirical analysis of critical success factors for project management in global software development. In Proceedings of the 2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE), Montreal, QC, Canada, 25–26 May 2019; pp. 68–71.

4. Gemünden, H.G. Success factors of global new product development programs, the definition of project success, knowledge sharing, and special issues of project management journal. *Proj. Manag. J.* **2015**, *46*, 2–11. [CrossRef]
5. Hughes, S.W.; Tippett, D.D.; Thomas, W.K. Measuring project success in the construction industry. *Eng. Manag. J.* **2004**, *16*, 31–37. [CrossRef]
6. Project Management Institute. *Guide to the Project Management Body of Knowledge (Pmbok Guide)*; Project Management Institute: Newtown Square, PA, USA, 2013.
7. Kirsch, L.J. Software project management: An integrated perspective for an emerging paradigm. In *Framing the Domains of IT Management: Projecting the Future... Through the Past*; Pinnaflex Educational Resources inc: Ann Arbor, MI, USA, 2000; pp. 285–304.
8. Aladwani, A.M. IT project uncertainty, planning and success. *Inf. Technol. People* **2002**, 210–226. [CrossRef]
9. Cates, G.R.; Mollaghasemi, M. The project assessment by simulation technique. *Eng. Manag. J.* **2007**, *19*, 3–10. [CrossRef]
10. Parsons, V.S. Project performance: How to assess the early stages. *Eng. Manag. J.* **2006**, *18*, 11–15. [CrossRef]
11. Rosenfeld, Y. Root-cause analysis of construction-cost overruns. *J. Constr. Eng. Manag.* **2014**, *140*, 04013039. [CrossRef]
12. Wang, J.; Li, J.; Wang, Q.; Zhang, H.; Wang, H. A simulation approach for impact analysis of requirement volatility considering dependency change. In Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, Essen, Germany, 19–22 March 2012; pp. 59–76.
13. Ferreira, S.; Collofello, J.; Shunk, D.; Mackulak, G. Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation. *J. Syst. Softw.* **2009**, *82*, 1568–1577. [CrossRef]
14. Tiwana, A.; Keil, M. The one-minute risk assessment tool. *Commun. ACM* **2004**, *47*, 73–77. [CrossRef]
15. Sommerville, I. *Software Engineering*, 9th ed.; Pearson: London, UK, 2011; ISBN 0137035152.
16. Ali, N.; Hwang, S.; Hong, J.E. Your Opinions Let us Know: Mining Social Network Sites to Evolve Software Product Lines. *Ksii Trans. Internet Inf. Syst.* **2019**, *13*. [CrossRef]
17. Malhotra, R.; Chug, A. Software maintainability: Systematic literature review and current trends. *Int. J. Softw. Eng. Knowl. Eng.* **2016**, *26*, 1221–1253. [CrossRef]
18. Sharma, P.; Singh, J. Systematic literature review on software effort estimation using machine learning approaches. In Proceedings of the 2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS), Jammu, India, 11–12 December 2017; pp. 43–47.
19. Alsalemi, A.M.; Yeoh, E.-T. A Systematic Literature Review of Requirements Volatility Prediction. In Proceedings of the 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), Mysore, India, 8–9 September 2017; pp. 55–64.
20. Alsolai, H.; Roper, M. A systematic literature review of machine learning techniques for software maintainability prediction. *Inf. Softw. Technol.* **2020**, *119*, 106214. [CrossRef]
21. Idri, A.; Abnane, I.; Abran, A. Systematic mapping study of missing values techniques in software engineering data. In Proceedings of the 2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Takamatsu, Japan, 1–3 June 2015; pp. 1–8.
22. Pillai, S.P.; Madhukumar, S.; Radharamanan, T. Consolidating evidence based studies in software cost/effort estimation—A tertiary study. In Proceedings of the TENCON 2017 IEEE Region 10 Conference, Penang, Malaysia, 5–8 November 2017; pp. 833–838.
23. Sangwan, O.P. Software effort estimation using machine learning techniques. In Proceedings of the 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence, Noida, India, 12–13 January 2017; pp. 92–98.
24. Stewart, C.A.; Hancock, D.Y.; Wernert, J.; Furlani, T.; Lifka, D.; Sill, A.; Berente, N.; McMullen, D.F.; Cheatham, T.; Apon, A.; et al. Assessment of financial returns on investments in cyberinfrastructure facilities: A survey of current methods. In Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning), Chicago, IL, USA, 28 July–1 August 2019; pp. 1–8.
25. García, J.A.L.; Peña, A.B.; Pérez, P.Y.P.; Pérez, R.B. Project control and computational intelligence: Trends and challenges. *Int. J. Comput. Intell. Syst.* **2017**, *10*, 320–335. [CrossRef]
26. Raharjo, T.; Purwandari, B. Agile Project Management Challenges and Mapping Solutions: A Systematic Literature Review. In Proceedings of the 3rd International Conference on Software Engineering and Information Management, Sydney, NSW, Australia, 12–15 January 2020; pp. 123–129.
27. Cleland-Huang, J.; Czauderna, A.; Gibiec, M.; Emenecker, J. A machine learning approach for tracing regulatory codes to product specific requirements. In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, Cape Town, South Africa, 2–8 May 2010; Volume 1, pp. 155–164.
28. Zhang, D.; Dang, Y.; Lou, J.-G.; Han, S.; Zhang, H.; Xie, T. Software analytics as a learning case in practice: Approaches and experiences. In Proceedings of the International Workshop on Machine Learning Technologies in Software Engineering, Lawrence, KS, USA, 12 November 2011; pp. 55–58.
29. Pospieszny, P. Software estimation: Towards prescriptive analytics. In Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement, Gothenburg, Sweden, 25–27 October 2017; pp. 221–226.

30. ManikReddy, P.; Iyer, J. Effective collaboration across the globe through digital dash boards and machine learning. In Proceedings of the 2018 IEEE/ACM 13th International Conference on Global Software Engineering (ICGSE), Gothenburg, Sweden, 6 December 2018; pp. 30–34.
31. Moharreri, K.; Sapre, A.V.; Ramanathan, J.; Ramnath, R. Cost-effective supervised learning models for software effort estimation in agile environments. In Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA, 10–14 June 2016; pp. 135–140.
32. Hosni, M.; Idri, A.; Nassif, A.B.; Abran, A. Heterogeneous ensembles for software development effort estimation. In Proceedings of the 2016 3rd International Conference on Soft Computing & Machine Intelligence (ISCMI), Dubai, United Arab Emirates, 23–25 November 2016; pp. 174–178.
33. Samath, S.; Udalagama, D.; Kurukulasooriya, H.; Premarathne, D.; Thelijjagoda, S. Collabcrew—An intelligent tool for dynamic task allocation within a software development team. In Proceedings of the 2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Malabe, Sri Lanka, 6–8 December 2017; pp. 1–9.
34. Li, Y.; Huang, Z.; Wang, Y.; Fang, B. Evaluating data filter on cross-project defect prediction: Comparison and improvements. *IEEE Access* **2017**, *5*, 25646–25656. [CrossRef]
35. Ni, A.; Li, M. Poster: ACONA: Active Online Model Adaptation for Predicting Continuous Integration Build Failures. In Proceedings of the 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), Gothenburg, Sweden, 3 June 2018; pp. 366–367.
36. Sharma, P.; Singh, J. Machine Learning Based Effort Estimation Using Standardization. In Proceedings of the 2018 International Conference on Computing, Power and Communication Technologies (GUCON), Greater Noida, India, 29 September 2018; pp. 716–720.
37. Papatheocharous, E.; Andreou, A.S. A hybrid software cost estimation approach utilizing decision trees and fuzzy logic. *Int. J. Softw. Eng. Knowl. Eng.* **2012**, *22*, 435–465. [CrossRef]
38. Hongming, Z.; Bin, F.; Xizhu, M.; Lijun, S.; Xiangzhou, X.Z.; Yong, H. A Cost-sensitive Intelligent Prediction Model for Outsourced Software Project Risk. In Proceedings of the WHICEB 2013 Proceedings, Wuhan, China, 25–26 May 2013.
39. Twala, B. Reasoning with Noisy Software Effort Data. *Appl. Artif. Intell.* **2014**, *28*, 533–554.
40. Wu, J.H.; Keung, J. Decision support for global software development with pattern discovery. In Proceedings of the 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 26–28 August 2016; pp. 182–185.
41. Rahman, M.T.; Islam, M.M. A Comparison of Machine Learning Algorithms to Estimate Effort in Varying Sized Software. In Proceedings of the 2019 IEEE Region 10 Symposium (TENSYMP), Kolkata, India, 7–9 June 2019; pp. 137–142.
42. Tumar, I.; Hassouneh, Y.; Turabieh, H.; Thaher, T. Enhanced binary moth flame optimization as a feature selection algorithm to predict software fault prediction. *IEEE Access* **2020**, *8*, 8041–8055. [CrossRef]
43. Lopez-Martin, C.; Chavoya, A.; Meda-Campaña, M.E. A machine learning technique for predicting the productivity of practitioners from individually developed software projects. In Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Las Vegas, NV, USA, 30 June–2 July 2014; pp. 1–6.
44. Han, W.; Jiang, H.; Zhang, X.; Li, W. A Neural Network Based Algorithms for Project Duration Prediction. In Proceedings of the 2014 7th International Conference on Control and Automation, Hainan, China, 20–23 December 2014; pp. 60–63.
45. Basgalupp, M.P.; Barros, R.C.; da Silva, T.S.; de Carvalho, A.C. Software effort prediction: A hyper-heuristic decision-tree based approach. In Proceedings of the 28th Annual ACM Symposium on Applied Computing, Coimbra, Portugal, 18–22 March 2013; pp. 1109–1116.
46. Twala, B.; Cartwright, M. Ensemble missing data techniques for software effort prediction. *Intell. Data Anal.* **2010**, *14*, 299–331. [CrossRef]
47. Song, L.; Minku, L.L.; Yao, X. The impact of parameter tuning on software effort estimation using learning machines. In Proceedings of the 9th International Conference on Predictive Models in Software Engineering, Baltimore, MD, USA, 9 October 2013; pp. 1–10.
48. Minku, L.L.; Yao, X. How to make best use of cross-company data in software effort estimation? In Proceedings of the 36th International Conference on Software Engineering, Hyderabad, India, 31 May–7 June 2014; pp. 446–456.
49. Song, L.; Minku, L.L.; Yao, X. The potential benefit of relevance vector machine to software effort estimation. In Proceedings of the 10th International Conference on Predictive Models in Software Engineering, Torino, Italy, 17 September 2014; pp. 52–61.
50. Scott, E.; Pfahl, D. Using developers' features to estimate story points. In Proceedings of the 2018 International Conference on Software and System Process, Gothenburg, Sweden, 26–27 May 2018; pp. 106–110.
51. Benala, T.R.; Bandarupalli, R. Least square support vector machine in analogy-based software development effort estimation. In Proceedings of the 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, India, 23–25 December 2016; pp. 1–6.
52. Minku, L.L.; Hou, S. Clustering Dycom: An online cross-company software effort estimation study. In Proceedings of the 13th International Conference on Predictive Models and Data Analytics in Software Engineering, Toronto, ON, Canada, 8 November 2017; pp. 12–21.
53. Brady, A.; Menzies, T. Case-based reasoning vs parametric models for software quality optimization. In Proceedings of the 6th International Conference on Predictive Models in Software Engineering, Timisoara, Romania, 12–13 September 2010; pp. 1–10.

54. Borges, R.; Menzies, T. Learning to change projects. In Proceedings of the 8th International Conference on Predictive Models in Software Engineering, Lund, Sweden, 21–22 September 2012; pp. 11–18.
55. Jiang, Y.; Cukic, B. Misclassification cost-sensitive fault prediction models. In Proceedings of the 5th International Conference on Predictor Models in Software Engineering, Vancouver, BC, Canada, 18–19 May 2009; pp. 1–10.
56. Weld, D.S.; Dai, P. Execution control for crowdsourcing. In Proceedings of the 24th Annual ACM Symposium Adjunct on User Interface Software and Technology, Santa Barbara, CA, USA, 16–19 October 2011; pp. 57–58.
57. Shepperd, M. The scientific basis for prediction research. In Proceedings of the 8th International Conference on Predictive Models in Software Engineering, Lund, Sweden, 21–22 September 2012.
58. Karim, M.R.; Alam, S.D.A.; Kabeer, S.J.; Ruhe, G.; Baluta, B.; Mahmud, S. Applying data analytics towards optimized issue management: An industrial case study. In Proceedings of the 2016 IEEE/ACM 4th International Workshop on Conducting Empirical Studies in Industry (CESI), Austin, TX, USA, 17 May 2016; pp. 7–13.
59. Castro-Herrera, C.; Cleland-Huang, J. A machine learning approach for identifying expert stakeholders. In Proceedings of the 2009 Second International Workshop on Managing Requirements Knowledge, Atlanta, GA, USA, 1 September 2009; pp. 45–49.
60. Abdellatif, T.M. A Comparison Study Between Soft Computing and Statistical Regression Techniques for Software Effort Estimation. In Proceedings of the 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), Quebec, QC, Canada, 16 May 2018; pp. 1–5.
61. Mendes, E.; Turhan, B.; Rodríguez, P.; Freitas, V. Estimating the value of decisions relating to managing and developing software-intensive Products and Projects. In Proceedings of the 11th International Conference on Predictive Models and Data Analytics in Software Engineering, Beijing, China, 21 October 2015; pp. 1–4.
62. 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]
63. Qu, Y.; Yang, T.-Z. Research on occurrence frequency of IT projects risk based on fuzzy influence diagram. In Proceedings of the 2016 International Conference on Machine Learning and Cybernetics (ICMLC), Jeju, Korea, 13 July 2016; pp. 166–171.
64. Sree, S.R.; Ramesh, S. Analytical Structure of a Fuzzy Logic Controller for Software Development Effort Estimation. In *Computational Intelligence in Data Mining—Volume 1*; Springer: Berlin, Germany, 2016; pp. 209–216.
65. Raza, M.B.; Kirkham, T.; Harrison, R.; Monfared, R.; Haq, I.; Wood, S. Evolving knowledge based product lifecycle management from a digital ecosystem to support automated manufacturing. In Proceedings of the International Conference on Management of Emergent Digital EcoSystems, Lyon, France, 27–30 October 2009; pp. 437–441.
66. Yang, G.; Zhang, T.; Lee, B. Utilizing a multi-developer network-based developer recommendation algorithm to fix bugs effectively. In Proceedings of the 29th Annual ACM Symposium on Applied Computing, Gyeongju, Korea, 28 March 2014; pp. 1134–1139.
67. Amasaki, S.; Lokan, C. A Virtual Study of Moving Windows for Software Effort Estimation Using Finnish Datasets. In Proceedings of the International Conference on Product-Focused Software Process Improvement, Innsbruck, Austria, 28 October 2017; pp. 71–79.
68. Qahtani, A.M. An Empirical Study of Agile Testing in A Distributed Software Development Project. In Proceedings of the 2020 3rd International Conference on Geoinformatics and Data Analysis, Marseille, France, 17 April 2020; pp. 110–114.
69. Bruegge, B.; David, J.; Helming, J.; Koegel, M. Classification of tasks using machine learning. In Proceedings of the 5th International Conference on Predictor Models in Software Engineering, British Columbia Canada, 18–19 May 2009; pp. 1–11.
70. Minku, L.L.; Yao, X. Software effort estimation as a multiobjective learning problem. *ACM Trans. Softw. Eng. Methodol. (TOSEM)* **2013**, *22*, 1–32. [CrossRef]
71. Shivhare, J.; Rath, S.K. Software effort estimation using machine learning techniques. In Proceedings of the 7th India Software Engineering Conference, Noida, Chennai, India, 21 February 2014; pp. 1–6.
72. Ramaswamy, V.; Suma, V.; Pushphavathi, T. An approach to predict software project success by cascading clustering and classification. In Proceedings of the International Conference on Software Engineering and Mobile Application Modelling and Development (ICSEMA 2012), Chennai, India, 21 December 2012.
73. Iwata, K.; Nakashima, T.; Anan, Y.; Ishii, N. Effort estimation for embedded software development projects by combining machine learning with classification. In Proceedings of the 2016 4th Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/1st Intl Conf on Big Data, Cloud Computing, Data Science & Engineering (ACIT-CSII-BCD), Las Vegas, NV, USA, 14 December 2016; pp. 265–270.
74. Ionescu, V.-S. An approach to software development effort estimation using machine learning. In Proceedings of the 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 9 September 2017; pp. 197–203.
75. BaniMustafa, A. Predicting software effort estimation using machine learning techniques. In Proceedings of the 2018 8th International Conference on Computer Science and Information Technology (CSIT), Amman, Jordan, 12 July 2018; pp. 249–256.
76. Menzies, T.; Bird, C.; Zimmermann, T.; Schulte, W.; Kocaganeli, E. The inductive software engineering manifesto: Principles for industrial data mining. In Proceedings of the International Workshop on Machine Learning Technologies in Software Engineering, Lawrence, KS, USA, 12 November 2011; pp. 19–26.
77. Dehghan, A.; Blincoe, K.; Damian, D. A hybrid model for task completion effort estimation. In Proceedings of the 2nd International Workshop on Software Analytics, Seattle, WA, USA, 13 November 2016; pp. 22–28.

78. Tollin, I.; Fontana, F.A.; Zanoni, M.; Roveda, R. Change prediction through coding rules violations. In Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, Karlskrona, Sweden, 15–16 June 2017; pp. 61–64.
79. 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, 27 December 2009; pp. 629–632.
80. Manalif, E.; Capretz, L.F.; Nassif, A.B.; Ho, D. Fuzzy-ExCOM software project risk assessment. In Proceedings of the 2012 11th International Conference on Machine Learning and Applications,Boca Raton, FL, USA, 15 December 2012; pp. 320–325.
81. Rana, R.; Staron, M. Machine learning approach for quality assessment and prediction in large software organizations. In Proceedings of the 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 25 September 2015; pp. 1098–1101.
82. Tariq, S.; Usman, M.; Wong, R.; Zhuang, Y.; Fong, S. On learning software effort estimation. In Proceedings of the 2015 3rd International Symposium on Computational and Business Intelligence (ISCBI), Bali, Indonesia, 9 December 2015; pp. 79–84.
83. Kumar, L.; Rath, S.; Sureka, A. An empirical analysis on effective fault prediction model developed using ensemble methods. In Proceedings of the 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), Turin, Italy, 8 July 2017; pp. 244–249.
84. Hu, Y.; Feng, B.; Mo, X.; Zhang, X.; Ngai, E.W.T.; 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]
85. Pospieszny, P.; Czarnacka-Chrobot, B.; Kobylinski, A. An effective approach for software project effort and duration estimation with machine learning algorithms. *J. Syst. Softw.* **2018**, *137*, 184–196. [CrossRef]
86. Lochmann, K.; Ramadani, J.; Wagner, S. Are comprehensive quality models necessary for evaluating software quality? In Proceedings of the 9th International Conference on Predictive Models in Software Engineering, Baltimore, MD, USA, 9 October 2013; pp. 1–9.
87. Chen, N.; Hoi, S.C.; Xiao, X. Software process evaluation: A machine learning framework with application to defect management process. *Empir. Softw. Eng.* **2014**, *19*, 1531–1564. [CrossRef]
88. Song, Q.; Zhu, X.; Wang, G.; Sun, H.; Jiang, H.; Xue, C.; Xu, B.; Song, W. A machine learning based software process model recommendation method. *J. Syst. Softw.* **2016**, *118*, 85–100. [CrossRef]
89. Fitzgerald, C.; Letier, E.; Finkelstein, A. Early failure prediction in feature request management systems. In Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference, Trento, Italy, 2 September 2011; pp. 229–238.
90. Joseph, H.R. Poster: Software Development Risk Management: Using Machine Learning for Generating Risk Prompts. In Proceedings of the 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Florence, Italy, 24 May 2015; pp. 833–834.
91. ERTUĞRUL, E.; Baytar, Z.; ÇATAL, Ç.; MURATLI, Ö.C. Performance tuning for machine learning-based software development effort prediction models. *Turk. J. Electr. Eng. Comput. Sci.* **2019**, *27*, 1308–1324. [CrossRef]
92. Colomo-Palacios, R.; González-Carrasco, I.; López-Cuadrado, J.L.; Trigo, A.; Varajao, J.E. I-Competere: Using applied intelligence in search of competency gaps in software project managers. *Inf. Syst. Front.* **2014**, *16*, 607–625. [CrossRef]
93. Nassif, A.B.; Azzeh, M.; Capretz, L.F.; Ho, D. Neural network models for software development effort estimation: A comparative study. *Neural Comput. Appl.* **2016**, *27*, 2369–2381. [CrossRef]
94. Desai, V.S.; Mohanty, R. ANN-Cuckoo Optimization Technique to Predict Software Cost Estimation. In Proceedings of the 2018 Conference on Information and Communication Technology (CICT), Jabalpur, India, 28 October 2018; pp. 1–6.
95. Schleier-Smith, J. An architecture for Agile machine learning in real-time applications. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 13 August 2015; pp. 2059–2068.
96. Volf, Z.; Shmueli, E. Screening heuristics for project gating systems. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, Paderborn, Germany, 8 August 2017; pp. 872–877.
97. Liyi, M.; Shiyu, Z.; Jian, G. A project risk forecast model based on support vector machine. In Proceedings of the 2010 IEEE International Conference on Software Engineering and Service Sciences, Beijing, China, 18 July 2010; pp. 463–466.
98. Lopez-Martin, C.; Banitaan, S.; Garcia-Floriano, A.; Yanez-Marquez, C. Support vector regression for predicting the enhancement duration of software projects. In Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 21 December 2017; pp. 562–567.
99. Chou, J.-S.; Cheng, M.-Y.; Wu, Y.-W.; Wu, C.-C. Forecasting enterprise resource planning software effort using evolutionary support vector machine inference model. *Int. J. Proj. Manag.* **2012**, *30*, 967–977. [CrossRef]
100. Song, L.; Minku, L.L.; Yao, X. Software effort interval prediction via Bayesian inference and synthetic bootstrap resampling. *Acm Trans. Softw. Eng. Methodol. (TOSEM)* **2019**, *28*, 1–46. [CrossRef]
101. Dahab, S.A.; Porras, J.J.H.; Maag, S. A Software Measurement Plan Management Guided by an Automated Metrics Suggestion Framework. In Proceedings of the 2017 European Conference on Electrical Engineering and Computer Science (EECS), Bern, Switzerland, 19 Novermaber 2017; pp. 9–16.
102. Koroglu, Y.; Sen, A.; Kutluay, D.; Bayraktar, A.; Tosun, Y.; Cinar, M.; Kaya, H. Defect prediction on a legacy industrial software: A case study on software with few defects. In Proceedings of the 2016 IEEE/ACM 4th International Workshop on Conducting Empirical Studies in Industry (CESI), Austin, TX, USA, 17 May 2016; pp. 14–20.

103. Azzeh, M.; Banitaan, S. An Application of Classification and Class Decomposition to Use Case Point Estimation Method. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 11 December 2015; pp. 1268–1271.
104. Petkovic, D.; Sosnick-Pérez, M.; Huang, S.; Todtenhoefer, R.; Okada, K.; Arora, S.; Sreenivasen, R.; Flores, L.; Dubey, S. Setap: Software engineering teamwork assessment and prediction using machine learning. In Proceedings of the 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, Madrid, Spain, 25 October 2014; pp. 1–8.
105. del Águila, I.M.; Sagrado, J.D. Requirement risk level forecast using Bayesian networks classifiers. *Int. J. Softw. Eng. Knowl. Eng.* **2011**, *21*, 167–190. [CrossRef]
106. Alsri, A.; Almuhammadi, S.; Mahmood, S. A model for work distribution in global software development based on machine learning techniques. In Proceedings of the 2014 Science and Information Conference, London, UK, 29 August 2014; pp. 399–403.
107. Miandoab, E.E.; Gharehchopogh, F.S. A novel hybrid algorithm for software cost estimation based on cuckoo optimization and k-nearest neighbors algorithms. *Eng. Technol. Appl. Sci. Res.* **2016**, *6*, 1018–1022. [CrossRef]
108. Basgalupp, M.P.; Barros, R.C.; Ruiz, D.D. Predicting software maintenance effort through evolutionary-based decision trees. In Proceedings of the 27th Annual ACM Symposium on Applied Computing, Riva del Garda, Italy, 29 March 2012; pp. 1209–1214.
109. Bakır, A.; Turhan, B.; Bener, A.B. A new perspective on data homogeneity in software cost estimation: A study in the embedded systems domain. *Softw. Qual. J.* **2010**, *18*, 57–80. [CrossRef]
110. Helming, J.; Koegel, M.; Hodaie, Z. Towards automation of iteration planning. In Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications, Orlando, FL, USA, 29 October 2009; pp. 965–972.
111. Choetkiertikul, M.; Dam, H.K.; Tran, T.; Pham, T.; Ghose, A.; Menzies, T. A deep learning model for estimating story points. *IEEE Trans. Softw. Eng.* **2018**, *45*, 637–656. [CrossRef]
112. Niinimäki, T.; Piri, A.; Hynninen, P.; Lassenius, C. Studying communication in agile software development: A research framework and pilot study. In Proceedings of the ICMI-MLMI'09 Workshop on Multimodal Sensor-Based Systems and Mobile Phones for Social Computing, Cambridge, MA, USA, 6 November 2009; pp. 1–4.
113. Pechau, J. Rafting the agile waterfall: Value based conflicts of agile software development. In Proceedings of the 16th European Conference on Pattern Languages of Programs, Irsee, Germany, 17 July 2011; pp. 1–15.
114. Gousios, G.; Zaidman, A. A dataset for pull-based development research. In Proceedings of the 11th Working Conference on Mining Software Repositories, Hyderabad, India, 18 May 2014; pp. 368–371.
115. Makris, C.; Vikatos, P.; Visser, J. Classification model for predicting cost slippage in governmental ICT projects. In Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, 17 April 2015; pp. 1238–1241.
116. Qu, Y.; Tang, X.-L. Software project risk assessing model based on credal networks. In Proceedings of the 2010 International Conference on Machine Learning and Cybernetics, Qingdao, China, 14 July 2010; pp. 1976–1979.
117. Gouthaman, P.; Sankaranarayanan, S. Agile Software Risk Management Architecture for IoT-Fog based systems. In Proceedings of the 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 14 December 2018; pp. 48–51.
118. Andrés, J.D.; Landajo, M.; Lorca, P. Using nonlinear quantile regression for the estimation of software cost. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, Oviedo, Spain, 22 June 2018; pp. 422–432.
119. Pa, R.S.; Snsvsc, R. Improving efficiency of fuzzy models for effort estimation by cascading & clustering techniques. *Procedia Comput. Sci.* **2016**, *85*, 278–285.
120. Nassif, A.B.; Azzeh, M.; Idri, A.; Abran, A. Software development effort estimation using regression fuzzy models. *Comput. Intell. Neurosci.* **2019**, *2019*. [CrossRef]
121. Mohebzada, J.G.; Ruhe, G.; Eberlein, A. SRP-plugin: A strategic release planning plug-in for visual studio 2010. In Proceedings of the 1st Workshop on Developing Tools as Plug-ins, Honolulu, HI, USA, 28 May 2011; pp. 36–39.
122. Baolong, Y.; Hong, W.; Haodong, Z. Research and application of data management based on Data Management Maturity Model (DMM). In Proceedings of the 2018 10th International Conference on Machine Learning and Computing, Macau, China, 10 February 2018; pp. 157–160.
123. Sigweni, B. Feature weighting for case-based reasoning software project effort estimation. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London, UK, 13–14 May 2014; pp. 1–4.
124. Huang, Z.-W. Cost Estimation of Software Project Development by Using Case-Based Reasoning Technology with Clustering Index Mechanism. In Proceedings of the 2009 Fourth International Conference on Innovative Computing, Information and Control (ICICIC), Kaohsiung, Taiwan, 7–9 December 2009; pp. 1049–1052.
125. Wang, Y.-H.; Jia, J.; Qu, Y. The "Earth-Moon" model on software project risk management. In Proceedings of the 2010 International Conference on Machine Learning and Cybernetics, Qingdao, China, 14 July 2010; pp. 1999–2003.
126. Amasaki, S.; Kawata, K.; Yokogawa, T. Improving cross-project defect prediction methods with data simplification. In Proceedings of the 2015 41st Euromicro Conference on Software Engineering and Advanced Applications, Madeira, Portugal, 28 August 2015; pp. 96–103.
127. Nassif, A.B.; Capretz, L.F.; Ho, D.; Azzeh, M. A treeboost model for software effort estimation based on use case points. In Proceedings of the 2012 11th International Conference on Machine Learning and Applications, Boca Raton, FL, USA, 15 December 2012; pp. 314–319.

128. Wagner, S. A literature survey of the quality economics of defect-detection techniques. In Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering, Rio de Janeiro Brazil, 21–22 September 2006; pp. 194–203.
129. Pressman, R.S. *Software Engineering: A Practitioner's Approach*; Palgrave Macmillan: London, UK, 2005.
130. Nassif, A.B.; Ho, D.; Capretz, L.F. Towards an early software estimation using log-linear regression and a multilayer perceptron model. *J. Syst. Softw.* **2013**, *86*, 144–160. [CrossRef]
131. Menzies, T.; Mizuno, O.; Takagi, Y.; Kikuno, T. Explanation vs performance in data mining: A case study with predicting runaway projects. *J. Softw. Eng. Appl.* **2009**, *2*, 221. [CrossRef]
132. Kitchenham, B.; Mendes, E.; Travassos, G.H. A systematic review of cross-vs. within-company cost estimation studies. In Proceedings of the 10th International Conference on Evaluation and Assessment in Software Engineering (EASE) 10, Swindon, UK, 11 April 2006; pp. 1–10.
133. Mahdi, M.N.; Yusof, M.Z.M.H.A.; Cheng, L.K.; Azmi, M.S.M.; Ahmad, A.R. Design and 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, 26 August 2020; pp. 354–362.
134. Lee, T.; Gu, T.; Baik, J. MND-SCEMP: An empirical study of a software cost estimation modeling process in the defense domain. *Empir. Softw. Eng.* **2014**, *19*, 213–240. [CrossRef]
135. Mitchell, S.M.; Seaman, C.B. A comparison of software cost, duration, and quality for waterfall vs. iterative and incremental development: A systematic review. In Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, Lake Buena Vista, FL, USA, 16 October 2009; pp. 511–515.
136. Jorgensen, M.; Shepperd, M. A systematic review of software development cost estimation studies. *IEEE Trans. Softw. Eng.* **2006**, *33*, 33–53. [CrossRef]
137. González-Ladrón-de-Guevara, F.; Fernández-Diego, M.; Lokan, C. The usage of ISBSG data fields in software effort estimation: A systematic mapping study. *J. Syst. Softw.* **2016**, *113*, 188–215. [CrossRef]
138. Iranmanesh, S.H.; Hojati, Z.T. Intelligent systems in project performance measurement and evaluation. In Proceedings of the Intelligent Techniques in Engineering Management; Springer: Berlin, Germany, 5 May 2015; pp. 581–619.
139. Mellegård, N.; Staron, M. Characterizing model usage in embedded software engineering: A case study. In Proceedings of the Fourth European Conference on Software Architecture: Companion Volume, Copenhagen, Denmark, 23–26 August 2010; pp. 245–252.
140. Antonellis, P.; Antoniou, D.; Kanellopoulos, Y.; Makris, C.; Theodoridis, E.; Tjortjis, C.; Tsirakis, N. A data mining methodology for evaluating maintainability according to ISO/IEC-9126 software engineering–product quality standard. Special Session on System Quality and Maintainability-SQM2007. 2007. Available online: https://www.ihu.edu.gr/tjortjis/A%20Data%20Mining%20Methodology%20for%20Evaluating%20Maintainability%20according%20to%20SQM07.pdf (accessed on 5 May 2021).
141. Azar, D.; Harmanani, H.; Korkmaz, R. A hybrid heuristic approach to optimize rule-based software quality estimation models. *Inf. Softw. Technol.* **2009**, *51*, 1365–1376. [CrossRef]
142. Mahdi, M.N.; Azmi, M.S.M.; Cheng, L.K.; Yusof, A.; Ahmad, A.R. Software Project Management Using Machine Learning Technique-A Review. In Proceedings of the 2020 8th International Conference on Information Technology and Multimedia (ICIMU), Selangor, Malaysia, 26 August 2020; pp. 363–370.
143. Zhang, H.; Dai, G. The strategy of traffic congestion management based on case-based reasoning. *Int. J. Syst. Assur. Eng. Manag.* **2019**, *10*, 142–147. [CrossRef]
144. Agrawal, A.; Menzies, T. "Better Data" is Better than "Better Data Miners" (Benefits of Tuning SMOTE for Defect Prediction). *arXiv* **2017**, arXiv:1705.03697.
145. Amasaki, S.; Takahara, Y.; Yokogawa, T. Performance evaluation of windowing approach on effort estimation by analogy. In Proceedings of the 2011 Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement, Nara, Japan, 4 November 2011; pp. 188–195.
146. Arcuri, A.; Briand, L. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In Proceedings of the 2011 33rd International Conference on Software Engineering (ICSE), Honolulu, HI, USA, 28 May 2011; pp. 1–10.
147. Wan, S.; Li, D.; Gao, J.; Li, J. A knowledge based machine tool maintenance planning system Using case-based reasoning techniques. *Robot. Comput. Integr. Manuf.* **2019**, *58*, 80–96. [CrossRef]
148. Kaur, A.; Kaur, K. Effort Estimation for Mobile Applications Using Use Case Point (UCP). In Proceedings of the Smart Innovations in Communication and Computational Sciences, Bangkok, Thailand, 30 June 2019; pp. 163–172.
149. Srivastava, A.; Singh, S.; Abbas, S.Q. Performance Measure of the Proposed Cost Estimation Model: Advance Use Case Point Method. In Proceedings of the Soft Computing: Theories and Applications, Lviv, Ukraine, 20 September 2019; pp. 223–233.
150. Larsson, S.; Jansson, M.; Boholm, Å. Expert stakeholders' perception of nanotechnology: Risk, benefit, knowledge, and regulation. *J. Nanoparticle Res.* **2019**, *21*, 57. [CrossRef]
151. Poth, A.; Sasabe, S.; Mas, A.; Mesquida, A.L. Lean and agile software process improvement in traditional and agile environments. *J. Software Evol. Process.* **2019**, *31*, e1986. [CrossRef]
152. Sievi-Korte, O.; Beecham, S.; Richardson, I. Challenges and recommended practices for software architecting in global software development. *Inf. Softw. Technol.* **2019**, *106*, 234–253. [CrossRef]

153. Lops, P.; Gemmis, M.D.; Semeraro, G. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*; Springer: Berlin, Germany, 2011; pp. 73–105.
154. Fauzi, S.S.M.; Ramli, N.; Nasir, M.H.N.M. Software Configuration Management A Result from the Assessment and its Recommendation. In Proceedings of the 2009 International Conference on Information Management and Engineering, Kuala Lumpur, Malaysia, 3–5 April 2009; pp. 416–419.
155. Khomyakov, I.; Mirgalimova, R.; Sillitti, A. An investigation of the project management approaches of agile and plan-based companies. In Proceedings of the 35th Annual ACM Symposium on Applied Computing, Brno, Czech Republic, 3 April 2020; pp. 1662–1665.
156. Prakash, B.; Viswanathan, V. A. Survey on Software Estimation Techniques in Traditional and Agile Development Models. *Indones. J. Electr. Eng. Comput. Sci.* **2017**, *7*, 867–876. [CrossRef]
157. Picha, P.; Brada, P. Software process anti-pattern detection in project data. In Proceedings of the 24th European Conference on Pattern Languages of Programs, Irsee, Germany, 19 July 2019; pp. 1–12.
158. Kappen, T.H.; Vergouwe, Y.; Wolfswinkel, L.V.; Kalkman, C.; Moons, K.; Klei, W.V. Impact of adding therapeutic recommendations to risk assessments from a prediction model for postoperative nausea and vomiting. *Br. J. Anaesth.* **2015**, *114*, 252–260. [CrossRef]
159. Kanimozhi, U.; Ganapathy, S.; Manjula, D.; Kannan, A. An intelligent risk prediction system for breast cancer using fuzzy temporal rules. *Natl. Acad. Sci. Lett.* **2019**, *42*, 227–232. [CrossRef]
160. Matharu, G.S.; Mishra, A.; Singh, H.; Upadhyay, P. Empirical study of agile software development methodologies: A comparative analysis. *ACM SIGSOFT Softw. Eng. Notes* **2015**, *40*, 1–6. [CrossRef]
161. Yang, M.Q.; Elnitski, L.L. Prediction-based approaches to characterize bidirectional promoters in the mammalian genome. *BMC Genom.* **2008**, *9*, S2. [CrossRef] [PubMed]
162. Nagwani, N.K.; Bhansali, A. A data mining model to predict software bug complexity using bug estimation and clustering. In Proceedings of the 2010 International Conference on Recent Trends in Information, Telecommunication and Computing, Kerala, India, 13 March 2010; pp. 13–17.
163. Shan, X.; Jiang, G.; Huang, T. A framework of estimating software project success potential based on association rule mining. In Proceedings of the 2009 International Conference on Management and Service Science, Beijing, China, 22 September 2009; pp. 1–4.
164. Khan, B.; Iqbal, D.; Badshah, S. Cross-Project Software Fault Prediction Using Data Leveraging Technique to Improve Software Quality. In Proceedings of the Evaluation and Assessment in Software Engineering, Trondheim, Norway, 17 April 2020; pp. 434–438.
165. Chelly, Z.; Elouedi, Z. Improving the dendritic cell algorithm performance using fuzzy-rough set theory as a pattern discovery technique. In Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA, Ostrava, Czech Republic, 25 June 2014; pp. 23–32.
166. Ghotra, B.; McIntosh, S.; Hassan, A.E. Revisiting the impact of classification techniques on the performance of defect prediction models. In Proceedings of the 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Florence, Italy, 16–24 May 2015; pp. 789–800.
167. Li, J.; Ji, X.; Jia, Y.; Zhu, B.; Wang, G.; Li, Z.; Liu, X. Hard drive failure prediction using classification and regression trees. In Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Atlanta, GA, USA, 26 June 2014; pp. 383–394.
168. Ryu, D.; Choi, O.; Baik, J. Value-cognitive boosting with a support vector machine for cross-project defect prediction. *Empir. Softw. Eng.* **2016**, *21*, 43–71. [CrossRef]