Journal of Technology Informatics and Engineering
Vol 2 No 3 December 2023
e-ISSN :2961-8215;.p-ISSN :2961-9068 Hal 22-44
DOI: https://doi.org/10.51903/jtie.v2i3.163

Innovation In Project Management Utilizing Machine Learning Technology

Joseph Teguh Santoso
University of Science and Computer Technology (STEKOM University),
Semarang, 57166, Indonesia
Email: joseph_teguh@stekom.ac.id

Budi Raharjo
University of Science and Computer Technology (STEKOM University),
Semarang, 57166, Indonesia
Email: budiraharjo@stekom.ac.id

Abstract. *The successful adoption of programmable machines for complex tasks opens up opportunities for productivity and more efficient communication, but also poses serious challenges in IT project management. This study aims to tackle the issue of high project failure rates caused by inadequate planning. It aims to assist project managers in enhancing their project planning by implementing real-time solutions through the utilization of machine learning (ML) algorithms and a user-friendly graphical interface. This research is divided into two key phases. The initial phase involves an examination of existing literature in the field of machine learning to identify relevant concepts applicable to project management. In the subsequent stage, two distinct types of ML algorithms, namely example-based learning and regression modeling, will be integrated into a user-friendly platform. This research develops a system that utilizes machine learning algorithms to assist project managers in real-time or near real-time through a user-friendly graphical interface, with a focus on improving project planning and risk mitigation. This research shows that machine learning algorithms provide positive results in overcoming human factors and preventing risks based on the experience of project managers.*

Keywords: *Project Management, Machine Learning, Information Technology*

INTRODUCTION

In the rapidly growing era of globalization, challenges in project management are increasingly complex and require innovative approaches to ensure project success. The use of computers and information technology has changed the paradigm in various aspects of life starting from the latter part of the 20th century, including in the world of project management. The effective implementation of machines capable of being programmed to execute intricate tasks in a short period has opened the door to increased productivity, more efficient communication, and rapid growth of the global economy. However, the proliferation of software projects in the global internet age also brings serious challenges to IT project management. According to [1] [2] and [3] many IT project failures are closely related to problems in project management itself. These failures are often caused by a lack of proper planning, including issues like inadequate requirement specification and insufficient risk assessment. Risk analysis, which should be a crucial step from the start of a project, is often sidelined. The importance of identifying potential problems that may occur during a project and having a plan to address these situations cannot be underestimated. Moreover, when these

Received: Oktober 29, 2023; Accepted: November 27, 2023; Published: December 30, 2023
* Joseph Teguh Santoso, joseph_teguh@stekom.ac.id

problems arise at a more advanced stage of project development, the consequences can be devastating, even to the point of project cancellation after a large investment of time and resources has been made.

In this context, one of the promising innovations in project management is the use of Machine Learning technology. This technology has made a profound impact on multiple facets of our lives, spanning the realms of business and project management (PM). As a field, PM encompasses the tasks of planning, organizing, overseeing, and regulating resources to attain predetermined project objectives. However, although there have been many frameworks and methodologies that have been applied in project management over the years, there are still many challenges to overcome, such as uncertainty in the project environment, changes in requirements, and complex decision-making. Machine Learning can be used to analyze large and complex project data, predict risks, identify unseen patterns, and even provide suggestions for better decision-making. Thus, leveraging Machine Learning within project management has the potential to enhance efficiency, lower expenses, and elevate project success rates. In this research, a platform will be developed that can help project teams enhance their project planning, supervision, and oversight. The platform will have the ability to learn from each user's specific planning and management style, provide intelligent suggestions, and identify risks based on the user's history and experience. As such, this research will address the urgent need to design innovative solutions that can address the increasingly complex challenges of IT project management in this digital age. To ensure the effectiveness, relevance, and adoption of the proposed platform in practical situations, it is imperative that machine learning algorithms produce real-time solutions and seamlessly integrate them into a user-friendly graphical interface. To achieve these goals, an in-depth analysis of existing works in the field of machine learning was conducted and adapted to the problem at hand.

LITERATURE REVIEW

IT Project Management (PM)

PM is characterized as a field with the primary objective of initiating, planning, executing, monitoring, and ultimately accomplishing the tasks of a team, geared towards achieving predetermined success benchmarks. A project can be described as an individual or cooperative undertaking that is strategically organized to accomplish a particular objective. However, a PM is an individual who is responsible for planning and executing a specific project. In this research, attention will be centered on project management techniques applied to IT projects.

Waterfall Model

This model represents a methodical project design approach wherein the project progresses in an orderly manner through a series of predetermined stages, commencing from product conceptualization or requirements analysis and culminating in system testing [4] Frequently, this model forms the basis for more complex and streamlined project lifecycle frameworks. The number of stages or steps within the waterfall model can be tailored and is contingent upon the project's attributes and how the project manager orchestrates its advancement. Nevertheless, the conventional waterfall model typically includes five key steps: Requirements Analysis, Design, Implementation, Verification, and Maintenance. During the Requirements Analysis phase, the PM conducts discussions with all project stakeholders and the project team to gather all requirements, consolidating them into a thorough document outlining the project's requirements. In the Design phase, the project requirements are meticulously scrutinized by the project team, and the project's architectural blueprint is strategized. During the Implementation phase, the development team begins building the software by the design specifications gathered from the previous phase. The software is fragmented into more manageable components and once all these components are developed, they are combined. During the Verification phase, the software undergoes rigorous testing. Finally, the Maintenance Phase entails making minor adjustments to the software to accommodate any changing requirements that have arisen during the earlier phases.

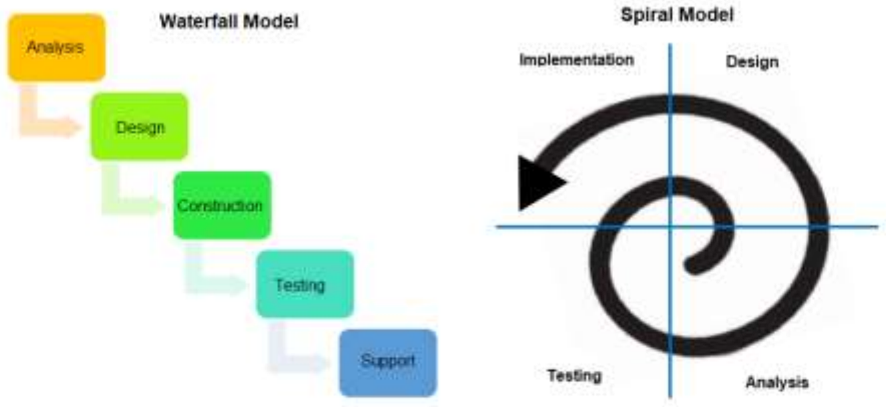


Fig 1. Waterfall Model and Spiral Model (Source: Santoso [6])

The primary drawback of the waterfall model is its inherent challenge in fully specifying project requirements at the project's outset, before any design or development work. Frequently, the individuals or organizations commissioning the project may not possess a comprehensive understanding of the problem they aim to address in the initial stages. Non-technical users and organizations often embark on projects to solve specific problems, but some of the requirements to achieve this objective may only become clear during the actual design or development phases. Despite these noted limitations, the waterfall model tends to function effectively as a lifecycle model in cases where the technical aspects of the project are well-understood, and the requirements remain stable right from the outset of project planning.

Spiral Model

This model, introduced by [6], is a process model that places a significant focus on the management of potential risks. This model involves dividing the project development process into a series of adaptable iterations, often organized into four main steps: Planning, Identification, Development and Testing, and, Iteration Planning. Hail a planning/goal setting, all the project requirements are collected in a requirement specification document. Next in Risk analysis/Risk identification and resolution, during this phase, if any risks are identified, the team will present alternative approaches to mitigate these risks, or they will develop an appropriate contingency plan to address them. In the development and testing phase, the software will be developed, after which the software will be tested properly. In planning the next iteration as the last stage, before progressing to the next spiral iteration, the customer assesses the project's current state. Projects implemented using the spiral model are designed to commence with a small scope and then progressively widen the project's boundaries. Expanding the project's scope only occurs once the project manager has managed to lower the risk associated with the upcoming project phase to a level deemed acceptable. The spiral model places significant emphasis on risk analysis, a task that typically necessitates specialized expertise. In this model, project costs typically rise as the project advances, but the level of risks involved tends to decrease [7] In the context of projects that require rapid development using the spiral model, as more time and resources are invested, the project leader tends to assume less risk. Because the spiral model revolves around risk management, it can offer crucial insights even in the initial project stages. If technical or financial limitations render the project unviable, the project manager can detect this during an initial phase, and choose to halt the project before substantial time and resources are invested. Similar to numerous project management models, the iterative cycle of the spiral model is flexible and can be customized to match the particular requirements of each project.

Software Tools for Project Management

At present, the market boasts numerous software tools tailored to aid project managers in project planning, task organization, risk evaluation, and various related functions. Some of these tools will be briefly described in the subsequent sections. Although there is an abundance of robust project management software available, it's noteworthy that the majority of these tools do not incorporate "machine learning" capabilities to support users.

Microsoft Project

Microsoft Project is a project management software created by Microsoft and, as of 2017, has been recognized as the most extensively utilized project management application globally. This software empowers PMs to construct project plans, allocate resources to tasks, oversee project advancement, and encompass various other functionalities.

Gantt Chart

This is a horizontal bar chart utilized as a production control tool in PM. It serves to visually represent the project schedule [8] [9. This diagram provides a clear and accessible means for project managers and Team members to determine which tasks must be accomplished and their corresponding timing within the project.

Trello

Trello is an online project management platform that employs the Kanban methodology to facilitate project organization. In Trello, projects are structured by establishing numerous boards, with each board consisting of lists. Within these lists, you find cards, and as the project progresses, these cards transition from one list to another. Trello's flexibility makes it adaptable for a wide range of project types and applications.

PM and Machine Learning (ML)

ML is a branch of Computer Science focused on enabling computers to learn without requiring explicit programming. This thesis will delve into the real-world implementation of ML techniques within PM software. The goal is to aid PMs in enhancing project planning and proactively mitigating risks during the initial phases of a project.

Instance-based Learning (IbL)

IbL encompasses a group of ML algorithms that evaluate new problem instances by comparing them to previously observed instances from the training dataset. This approach differs from other algorithms like artificial neural networks, which rely on explicit generalization techniques [10 [11]. In our specific domain, our objective is for the system to grasp the unique patterns of individual users and offer guidance throughout their project management tasks. In contrast to other types of machine learning, which involve collecting substantial data for training models like logistic regression or artificial neural networks, our practical domain seeks to assist users promptly by providing advice and identifying project risks. This assistance is highly user-specific and project-specific. Consequently, creating a single extensive dataset for the entire platform is not suitable in our context. The critical information that is user-specific within the system should be utilized to deliver personalized guidance. Hence, we should consider exploring IbL in our system, as it excels at performing machine learning with modest amounts of dynamically evolving data.

k-NN Algorithm

The k-NN algorithm [12] [13] [14] [15] is a straightforward IbL approach where each training instance is depicted as a vector in RN and is stored in a training example list whenever it is observed. All computations related to classification are deferred until the system receives a classification request. This request involves carrying out classification or regression for a specified search point. In Classification, the algorithm produces an integer that indicates the class membership, which is determined by the majority of the k nearest neighbors to the input point. In the k-NN algorithm, the output is a real numerical value (y), which represents the average of the values derived from the k nearest training instances to the input query point. The commonly employed distance metric in the k-NN algorithm is the Euclidean distance.

$$(x_1, x_2) = \sum_{r=1}^{n} (a_r(x_1) - a_r(x_2))^2$$

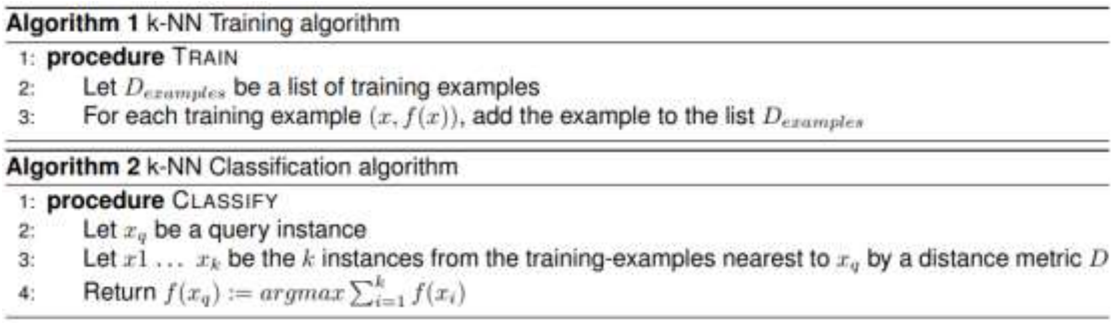
where x_1 and x_2 are two data points in R^n and $a_r(x)$ denotes the r_{th} coordinate value of data point x.

$$similarity(x_1, x_2) = \cos(\theta) = \frac{x_1 \cdot x_2}{||x_1||\ ||x_2||}$$

An alternative distance metric that can be employed and is independent of vector magnitude is the cosine distance. Cosine similarity quantifies the cosine of the angle formed by two points, x_1 and x_2, with a range of values between -1 and 1. A cosine similarity of 1 denotes complete similarity, a value of 0 signifies that the vectors are perpendicular (orthogonal), and -1 indicates that they are opposite to each other. This metric is particularly useful when comparing the similarity of vectors in a multi-dimensional space while disregarding their magnitudes.

$$d(x_1, x_2) = \frac{cos^{-1}(similarity(x_1, x_2)}{p^i}$$

While cosine similarity doesn't function as a conventional distance metric, it can be transformed into one using the formula. Algorithm 1 is utilized for training the k-NN model, and Algorithm 2 is employed for k-NN classification.



```
Algorithm 1 k-NN Training algorithm
1: procedure TRAIN
2:     Let $D_{examples}$ be a list of training examples
3:     For each training example $(x, f(x))$, add the example to the list $D_{examples}$

Algorithm 2 k-NN Classification algorithm
1: procedure CLASSIFY
2:     Let $x_q$ be a query instance
3:     Let $x1 \ldots x_k$ be the $k$ instances from the training-examples nearest to $x_q$ by a distance metric $D$
4:     Return $f(x_q) := argmax \sum_{i=1}^{k} f(x_i)$
```

Distance Weighted-NNA (Nearest Neighbour Algorithm)

The k-NN algorithm can be adapted to assign varying levels of importance to individual demand points (xq) based on their proximity. In this modification, greater weight is assigned to nearby neighbors compared to distant ones. When applying the k-NN algorithm with k = 5 for classifying a demand point (xq), if, for instance, three out of the five closest neighbors are situated at a considerable distance from the demand point *xq*, while the other 2 neighbors are closer but different, it can potentially introduce inaccuracies into the classification due to noise. To mitigate this issue, we can weigh the influence of each point based on its distance. This results in closer neighbors having a greater contribution to the classification or regression of the demand point. By assigning this weight, we can pay more attention to neighbors that have a greater impact on the demand point, while distant neighbors will have less impact.

$$w_i = \frac{1}{d(x_1, x_2)^2}$$

Managing data points in a high-dimensional hyperspace poses challenges for nearest-neighbor search algorithms. For instance, consider a vector x in R^n, where n = 50, representing a dataset with fifty attributes within a particular field or domain. Suppose we intend to carry out a classification task on this dataset, with only 3 out of the 50 vector dimensions being pertinent to our classification objective. While these 3 pertinent dimensions can cluster objects from the same category (implying that vectors or points belonging to the same category are proximate to each other in space, as indicated by a distance metric denoted as 'd'), the remaining 47 dimensions might introduce substantial separation between points of the same class. This renders the standard nearest neighbor algorithm ineffective without prior data preprocessing. This issue is referred to as "the curse of dimensionality" [16]. It arises because, as additional dimensions are introduced into the mathematical space, there is an exponential volume growth. Consequently, the traditional implementation of the k-NN algorithm may lose its utility in high-dimensional vector spaces. In such scenarios, the primary challenge shifts from merely finding the k-NN of the input query within the algorithm to identifying an appropriate "distance function" that can accurately capture the similarity between two feature vectors. This distance function should consider the relevance of each feature in the context of the problem, with each feature contributing varying weights to the distance function calculation.

Nearest Neighbour weighted (w-NN) features

One way to overcome the curse of dimensionality is to assign a contribution weight to different features [17] [18] [19] [20] in a k-nearest neighbor search, a feature vector represents a meaningful representation of an individual model instance. However, not all features within the vector hold the same level of significance for a given classification or regression problem. Certain features may be selected for consideration while others are deemed irrelevant to the specific problem at hand. By assigning appropriate weights, more important features will have a greater impact on the distance calculation, while less important features will have less impact. This approach allows us to pay greater attention to more relevant features and ignore less important features in the nearest neighbor search process. This can help overcome the curse of the dimensionality problem and improve the performance of the nearest neighbor algorithm in high-dimensional feature spaces.

Regression Model

Logistic Regression (LR)

LR is a form of regression model where the dependent variable, representing the model's output, is categorical [21]. LR can involve a dependent variable that is either binomial (for scenarios involving only two target categories) or multinomial (in cases where there are more than two categories). The primary goal of Logistic Regression (LR) is to create a model that can take an input vector x, which might represent a real-world object, and produce an estimated outcome y. The subsequent expression can be employed to compute predictions for a given LR model.

$$y = \sigma(Wx + b)\sigma = \frac{1}{1 + e^{-z}}$$

The result generated by a LR model is a weighted average of the input values after passing through a sigmoid activation function. The parameters that the model needs to learn are the weight (W) and the bias term (b). When a nonlinear activation function is not applied, LR behaves in the same way as LR [22]. The sigmoid function σ is constrained within the range of 0 and 1, offering a straightforward method to introduce non-linearity into the model and represent more intricate patterns. To train the regression model, which means determining the optimal values for the parameters W and b, we can apply numerical optimization methods like SGD [23] [24]. Regression serves a dual purpose; it not only facilitates the training of models for making predictions but can also be employed to assign weights that determine the significance of each feature within a feature vector for a specific problem. It's important to note that in binomial LR, the magnitude of each parameter linked to a feature does not directly convey the "semantic importance" of that particular feature within the context of the modeled problem. For instance, within a feature vector of dimension n, one feature might simply represent a scaling or transformation of other features, which can affect how the parameters are associated with each feature (e.g.: $f_1 = 100f_0$). In such a scenario, the acquired coefficients linked to those features could exhibit a difference of approximately 100-fold, yet the underlying semantic significance of the features f0 and f1 remains identical.

Stochastic Gradient Descent (SGD)

SGD is a commonly used first-order numerical optimization algorithm for training differentiable models [23 [24]. In such a scenario, the acquired coefficients linked to those features could exhibit a difference of approximately 100-fold, yet the underlying semantic significance of the features f_0 and f_1 remains identical:

$$\theta \coloneqq \theta - \eta \frac{\delta}{\delta\theta} J(\theta)$$

Certainly, the rule is applied iteratively, where η represents the learning rate, which controls the size of parameter updates, θ denotes the parameter being learned, J signifies the cost function, a measure of how well the model performs. Through this iterative process, the rule adjusts the parameter values until the model converges to a local minimum of the cost function, improving its accuracy and effectiveness in capturing patterns in the data.

$$W_{ij} \coloneqq W_{ij} - \eta \frac{\delta}{\delta W_{ij}} J(W,b)$$

$$b_i \coloneqq b_i - \eta \frac{\delta}{\delta b_i} J(W,b)$$

W stands for the weight matrix, while *b* represents the bias term.

Overfitting and Underfitting Model

In statistical modeling, overfitting refers to a phenomenon where the model incorporates noise or random errors into its structure, rather than accurately capturing the true relationship between input vectors and output values. Overfitting typically arises when the model is overly complex or when there's an insufficient amount of training data available for the training algorithm to generalize effectively.

Dynamic Learning Rate

The dynamic learning rate is usually performed during the training process, similar to the Simulated Annealing algorithm [25] [26]. The subsequent rule, when employed at each training epoch t, leads to an exponential reduction in the learning rate:

$$\eta_t + 1 \coloneqq r\eta_t$$

where r is a constant that indicates the learning rate attenuation ratio at each epoch.

Momentum

During the standard gradient descent optimization of a cost function, the geometric structure of the model's hyperspace can sometimes resemble a cliff with steep walls. This results in the gradient updates oscillating back and forth between these steep walls, posing a challenge, especially in training DNN models. This behavior can considerably slow down the training process or even lead to model divergence. To address this issue, a technique involves storing updates for all parameters across the model at a given time step, and in the subsequent step, the update becomes a combination of the current gradient and the most recent update. This approach helps to prevent abrupt changes in the update direction, typically resulting in quicker model convergence [27]. The following expression represents the update rule for adjusting parameter weights in Stochastic Gradient Descent (SGD) with momentum:

$$W_{ij}^{(l)}(t) \coloneqq W_{ij}^{(l)}(t) - \eta \frac{\delta}{\delta W_{ij}^{(l)}} J(W, b) + \ \alpha \Delta W_{ij}^{(l)}(t-1)$$

where $W_{ij}^{(l)}$and α are coefficients (0 vs 1) it's common to adjust the momentum scaling coefficient α to control the level of momentum applied during training. This adjustment involves starting with a relatively small momentum coefficient, which allows for rapid initial parameter updates. As training progresses over a few epochs, we often aim to ensure the network converges and takes smaller incremental steps. Consequently, increasing the momentum coefficient helps avoid abrupt changes in parameter updates, facilitating a smoother convergence process for the model.

$$\alpha_t + 1 \coloneqq r\alpha_t$$

Here, the constant r signifies how quickly the momentum coefficient will vary with each successive epoch.

METHODOLOGY

This research uses an analytical approach with the first stage of in-depth analysis of relevant literature on machine learning, this is done by collecting literature sources related to machine learning and its application in various contexts, identifying key concepts in machine learning that have the potential to be adapted in the context of project management and evaluating the identified concepts to determine their relevance and applicability in project management.

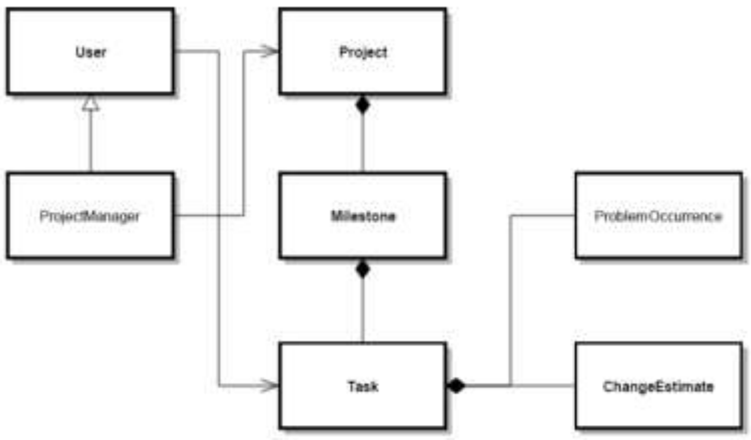


Fig 3. UML representation of the two data models

Analysis - Adaptive Risk Analysis Tool

Problem Formulation

The issue entails performing a risk analysis at the "milestone" level. To execute this risk analysis for a specific milestone planned at a particular time, it is necessary to pinpoint milestones from the project manager's past project experiences that resemble the current ones and evaluate the documented occurrences of problematic incidents.

Data Model

To develop a solution aiding teams in project management by offering guidance and risk assessment rooted in each user's historical data, it is vital to organize data into several suitable data models. Below, we will elucidate the structure and objectives of these models. These three data models serve as repositories for tracking the project's structural aspects within our platform. A milestone may operate independently, but it can also be linked to other milestones that require prior completion. A project consists of, ID, Name, Status, and Date Completed. A model that will store feedback from users during the project is also a must-have.

The Technique Used

By representing the 'Type' milestone feature using a one-hot vector, we ensure that each milestone type is treated as equally distinct, eliminating the issue of suggesting misleading proximity between types based on their position in a list. This one-hot encoding method creates a binary representation, allowing us to effectively apply distance metrics like the Euclidean distance, where the distance between any two milestone types is consistently equal, accurately reflecting their categorical independence. For instance, to represent type A ⊂ c, we employ the following one-hot vector:

$$Ftype = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

This allows the differences between the types to be properly accounted for in distance or similarity calculations.

Learning Algorithm

To solve this problem, the two categories of ML algorithms in use: IbL algorithms and regression models. An approach employed to address this issue is utilizing nearest neighbor search using the Nearest Neighbor Algorithm (NNA) to identify similar milestones and conduct risk analysis during the current project planning phase. The result of NNA consists of the k nearest neighbors of the query point. In this research domain, the nearest neighbor is the vector of milestones. The milestones associated with this vector having related problem occurrences are carefully examined and issues related to specific tasks are linked to individual tasks. As a milestone comprises several tasks, a milestone can be associated with the same problem multiple times. Given that the system is intended to function as an open web platform accessible to numerous project managers simultaneously and necessitates the provision of advice tailored to each user, it's imperative to establish a means of representing occurrences of problems within the algorithm. In practical situations, various problem categories may arise over the course of a project. In this research, problem types are denoted by integers, with each integer serving as the index for the corresponding problem within the user-specific problem list. To illustrate, if a user encounters problem types *K, L,* and *M* in their history, the mapping of problems to numbers would be as follows: *{K:0, L:1, M:2}.*

Selection and Weighting

NN algorithms like k-NN can encounter challenges related to the curse of dimensionality, especially in cases where only a handful of features in the feature vector are relevant to the particular problem being addressed. In our context, a project manager might be in the process of planning a project and encounter project management-related challenges only with particular types of milestones, such as those with a specific duration. Within our vector representation, the "duration" of a milestone takes up a single dimension in the vector space, while the remaining part of the vector comprises features that might not be especially pertinent in this context. To address this challenge, I propose employing a regression-based feature selection method, wherein each feature will be assigned a coefficient. This coefficient essentially quantifies the "level of importance" assigned to a specific feature for an individual user. A higher coefficient amplifies the significance of that feature within the feature vector, whereas a lower coefficient diminishes its importance. Think of this technique as a way to expand or shrink the vector space, thereby influencing the NN search. Because each project manager user has a distinct work history and experience, the weighting of features must be tailored to each project manager. To accomplish this, we've established a training set comprising milestone vectors created and stored by each specific project manager. We employ the SGD algorithm with momentum to train an LR model, and it converges in real-time within real-world scenarios. This is attainable because the number of stored milestones for each project manager is usually small enough to permit rapid computations. In our present use of LR, we aren't concerned with the model's predictions; instead, we're keen on the learned parameters represented by 'wr.' Each of these parameters, 'wr,' serves as a coefficient signifying the importance of the corresponding dimension within the milestone vectors.

Regression Model

We also examined an alternative approach to the nearest neighbor algorithm by building an LR model capable of predicting potential project risks based on an input milestone, which is a milestone just entered into the system by a project manager. To accomplish this, we created a training set 'T' consisting of pairs (x, y), where 'x' represents a vector of milestones and y is an integer indicating the occurrence of problems encountered during the project team's work on that specific milestone. The probability associated with each type of problem can be calculated as follows:

$$P(Y = i|x, W, b) = softmax_i(W_x + b) = \frac{e^{W_i x + b_i}}{\sum j \; e^{W_j x + b_j}}$$

The softmax activation function is employed because it serves as a generalization of the logistic function, commonly used in multicategory classification (as mentioned in Bishop, 2006). The prediction for the top risk can be computed using the following expression:

$$y_{prediction} = argmax_i(P(Y = i|x, W, b))$$

This particular model underwent training using the Stochastic Gradient Descent algorithm with momentum and demonstrated its capability to predict risks straightforwardly. It's crucial to highlight that this LR model varies from the one previously discussed in the previous section. It is now engaged in multi-category classification instead of merely determining whether an input milestone contains a potential risk or not. In this instance, the model is trained to forecast the degree of risk linked to an input milestone. However, it's important to note that the model itself arises from a global optimization process, which means it doesn't retain specific information about each training example. This characteristic represents a limitation of this approach compared to IbL methods, where all information is preserved and may, in specific situations, offer more comprehensive insights to the project manager. For instance, in cases where two specific problems occurred during a particular past milestone and could potentially recur, the model may not have the information needed to predict this situation accurately.

Hybrid Approach

An intriguing approach to address our problem involves developing a hybrid solution that utilizes both IbL methods and regression models. Despite the regression model no longer having access to the user's past experiences, it remains capable of predicting the primary risks linked to the given milestone input vector. The hybrid solution might involve creating a table of potential risks generated from the results of both algorithms, followed by assigning weights to determine the contribution of each algorithm to all identified risks. Ultimately, the top risk would be the one with the highest overall contribution when considering both algorithms. This approach seeks to capitalize on the strengths of both IbL and regression modeling to provide a more comprehensive and accurate risk prediction.

Chastising Old Experiences

Since this system is designed for use by project managers in project planning, it's crucial to consider that Project Managers (PMs) learn from their past experiences. Initially, a project manager may tend to underestimate the time required to complete certain types of tasks, but over time, they acquire better estimation skills. The impact of this human factor means that when our system conducts risk analysis, solely relying on the most similar past milestone data might not always yield the best results. To address this issue, we can introduce a mechanism to penalize query results based on the relative timing of milestones being entered into the system. This means giving greater importance to more recent results, which align better with the current expertise level of the project manager. Several approaches can be explored to achieve this objective. For instance, consider the following expression:

$$c = \frac{k}{d(x_1, x_2).\Delta t}$$

This expression signifies a potential reduction in the distance and time dimensions, where the larger the spatial distance *($d(x_1, x_2)$)* and the time difference *(Δt)* between the demand point and any point in the dataset, the smaller the assigned importance. It's crucial to take into account both spatial distance and time difference to address scenarios in which an older milestone bears a strong resemblance to the one currently planned by the project manager, and no similar milestones have been recently planned. To accommodate such scenarios, we aim to incorporate this aspect into our system and to impose penalties on milestones that are distant in both space and time, we are contemplating an expression in which the penalty increases quadratically as the spatial separation between two vectors grows.

$$c = \frac{k}{d(x_1, x_2).(\Delta t)^2}$$

This would lead to the algorithm disregarding the older records within the system, which may not be ideal since the project manager might be planning a new milestone that closely resembles one they worked on a while back, and the recent system entries might not offer pertinent insights for their specific situation.

RESULTS

Implementation

The system was designed to be adaptable and easily modifiable, allowing for potential future expansions through the introduction of new modules aimed at enhancing this solution and transforming it into a more valuable system. To incorporate all the requested functionalities, several decisions were made regarding the choice of programming language, platform, and tools. The objective was to opt for a cross-platform, open-source, and well-documented solution to ensure the portability, extensibility, and reliability of the proposed system. The proposed system is divided into three primary components, as depicted in Figure 4, and these components operate independently of each other. First is the Web Platform, designed for end-users, primarily project managers, to manage projects, input data, and receive feedback. The second component is the Database, responsible for persistent data storage. Lastly, there's the PMLO module, which handles queries requiring responses generated through machine learning techniques. The arrows in Figure 4 illustrate the flow of data between these components. When interacting with the database, it involves both reading and writing actions. In contrast, the interaction between the web platform and the software responsible for generating the solution primarily follows an activation-based approach. In this context, the platform initiates the software, the results are logged in the database, and they are not directly transmitted back to the web platform. Both the web platform and the machine learning software solely utilize the database component for data storage, devoid of any logic associated with this particular component.

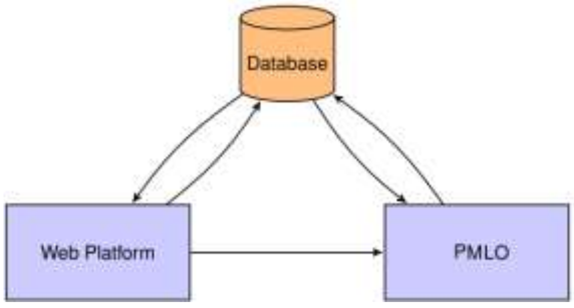


Fig 4. PMLO Scheme Architecture

Web Platform

After exploring various options, the decision was made to implement the system on a web-based platform. Utilizing a web platform enables users to access it from any location worldwide and via any device that has a web browser, including mobile devices. A system designed to assist teams in project management must offer an easily accessible interface. Users can perform tasks such as creating, modifying, and deleting projects, milestones, and tasks, as well as reporting events related to tasks. The web platform was created using the Django framework, which is renowned for its adherence to the model-view-template (MVT) design pattern (as described in [28]). Django was selected because of its simplicity and reliability as an open-source framework, supported by a large and dynamic community. It features comprehensive documentation and provides a broad range of plugins that can be utilized to enhance the platform's functionalities. Moreover, Django is compatible across various platforms and offers support for multiple database management systems, including PostgreSQL, MySQL, and SQLite. Moreover, Django projects can be readily deployed on popular web servers such as Apache and Nginx. Given that the platform's purpose is to facilitate project management for project managers and teams, ensuring a user-friendly, straightforward, and visually appealing interface was of paramount importance. To meet these requirements, Bootstrap, a framework encompassing Twitter's HTML, CSS, and JS, was employed. Bootstrap enables the creation of visually appealing user interfaces that adapt well to various screen sizes, including those of mobile devices.

Database

The database component serves the fundamental purpose of storing data persistently and consistently without any unique or specialized features. It serves as a crucial element within this platform because it acts as the repository for all user-provided data. Basic database operations, including Select, Insert, Update, and Delete, are employed to access and manage the data. A relational database is employed as the storage system. While there are various other database possibilities available today, relational databases were selected due to their widespread use and their compatibility with the presented model with minimal modifications. Different Database Management Systems (DBMS) implement the relational model, and although they offer varying features, they share a common foundation. In this research, only basic features are utilized, so the choice of DBMS is primarily influenced by the ease of infrastructure accessibility. For the development of this platform, SQLite was used as the DBMSSQLite is an extremely lightweight database management system that stores data in a single compact file within the operating system's file system. This design makes it highly portable and resource-efficient. One of the advantages of SQLite is its minimal configuration requirements, and it doesn't necessitate a dedicated database server. However, it's worth noting that SQLite may exhibit poorer performance and lacks support for user management. These limitations were not considered significant during the development phase. In contrast, PostgreSQL was chosen as the DBMS for platform implementation. Transitioning from PostgreSQL to another DBMS would have no performance implications, as the system does not rely on complex queries that might benefit from advanced query optimization. All queries used in this system are basic and do not involve sophisticated operators.

PMLO

The PMLO (Project Manager Learning Operations) component plays a pivotal role in executing all machine-learning tasks within this platform. This element establishes connections involving both the database and the web platform. When a user initiates an action that necessitates a machine learning response, like completing the planning of a project milestone and soliciting a risk analysis for that particular milestone, the web platform sends a query to the PMLO system. Subsequently, the PMLO system conducts a database query, computes the results utilizing the appropriate algorithm, and then furnishes the response back to the web platform. The rationale behind keeping the PMLO component separate from the web platform, both in the system architecture and its implementation, is to facilitate the platform's evolution while it's in production. Although the web platform is technically capable of performing all tasks executed by the PMLO component, the objective is to ensure that the machine learning component, responsible for intensive computational tasks, can be effortlessly replicated across multiple servers, based on the number of users concurrently accessing the platform, the web platform may utilize a load balancer to assess which PMLO servers are ready to efficiently manage user requests in such a situation.

In conclusion, the PMLO module was developed in Python, aligning with the choice made for the web platform module. This decision was driven by Python's extensive adoption in the fields of ML and data science, supported by a rich ecosystem of optimized and actively maintained libraries such as sci-kit-learn, TensorFlow, Theano, Pylearn2, and Caffe. Additionally, the module relies on Python's NumPy library for efficient mathematical computations. However, it's important to note that Python's interpretative nature can introduce performance challenges, potentially leading to slower code execution when compared to compiled languages. To mitigate this issue, the NumPy library has many of its operations implemented in the C language, enabling Python to achieve faster execution times. Additionally, libraries such as TensorFlow and Theano were designed with a strong focus on efficiency. They allow developers to construct computational graphs from mathematical operations, which are then compiled and optimized for both CPUs and GPUs (as detailed in [30] [31]. These computational graphs are highly adept at performing intricate functions, such as training deep neural networks. GPUs, in particular, offer substantial performance gains over CPUs for linear algebra operations, thanks to their ability to parallelize matrix operations.

DISCUSSION

Evaluation

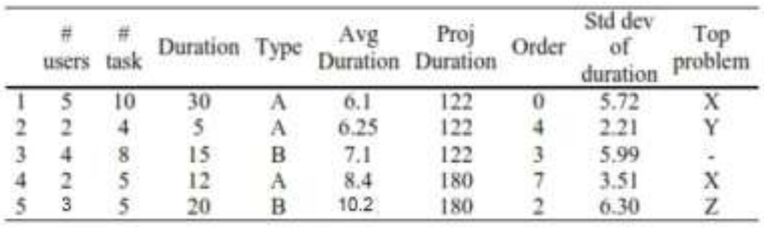
To test the validity of the offered solution, the program generates several sets of data. It is significant to note that the effectiveness of the project manager's auxiliary tasks during project planning relies not just on the project type and the features of the project but also largely on the capabilities of the manager of the project. For example, an inexperienced project manager may overlook the estimated time required to complete certain types of tasks, which may result in delays in achieving project milestones, and ultimately, the project itself. Hence, datasets are needed to take this variability into account this context, the dataset comprises diverse sets of projects, milestones, and tasks. Each milestone may have a connection with at least one occurrence of a problem, representing the circumstances encountered during the completion of that milestone. Each component in this dataset has various properties defined. The realism of the data is maintained as much as possible by assigning appropriate values to the milestone and task properties, as assigning random values would lead to uselessness in learning. In addition, certain "biases" are introduced into the dataset generation algorithm to make the data more relevant and interesting. For example, in certain scenarios, milestones of a certain type can have a certain probability of reporting certain issues. Another crucial aspect to take into account when constructing a dataset is the learning curve of project managers. For instance, a less experienced project manager utilizing our platform might initially make inaccurate estimations regarding the time required to complete a starting milestone. However, with time, they are likely to refine their duration estimates. To deliver this real-world scenario, a portion of the evaluation dataset was deliberately designed with this in consideration. Table 1 shows a highlight of the datasets employed to assess the learning algorithms in this research. Subsequently, these datasets were randomly Separated into a training set and a validation set, approximately in a 7:3 proportion.

Table 1. Description of datasets for testing our algorithm in various scenarios

	Number of project	Number of milestones	Number of task	PM learn over time
Dataset 1	1	16	100	No
Dataset 2	3	30	200	No
Dataset 3	3	30	200	Yes

Sample Data

Table 2. Every block shows the attributes of a milestone extracted from the records in the databas



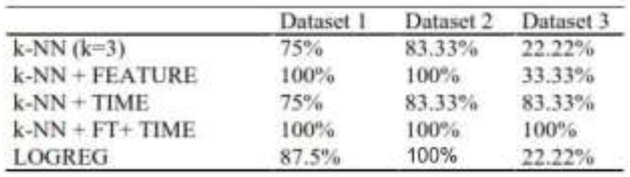
	# users	# task	Duration	Type	Avg Duration	Proj Duration	Order	Std dev of duration	Top problem
1	5	10	30	A	6.1	122	0	5.72	X
2	2	4	5	A	6.25	122	4	2.21	Y
3	4	8	15	B	7.1	122	3	5.99	-
4	2	5	12	A	8.4	180	7	3.51	X
5	3	5	20	B	10.2	180	2	6.30	Z

Solution Quality

Accuracy

Table 3 showcases the performance accuracy of several applied techniques when employed to forecast the primary potential problems associated with project milestones. In summary, these three methods—'k-NN + TIME,' 'k-NN + FEATURE,' and 'LOGREG'—employ different strategies for analyzing data and making predictions, with 'k-NN + TIME' emphasizing recent history, 'k-NN + FEATURE' incorporating feature weighting, and 'LOGREG' focusing on predicting primary potential problems related to input milestones.

Table 3. Accuracy of various techniques applied to the evaluation dataset



	Dataset 1	Dataset 2	Dataset 3
k-NN (k=3)	75%	83.33%	22.22%
k-NN + FEATURE	100%	100%	33.33%
k-NN + TIME	75%	83.33%	83.33%
k-NN + FT+ TIME	100%	100%	100%
LOGREG	87.5%	100%	22.22%

From the results obtained, it can be seen that all the techniques used give satisfactory results. The k-NN algorithm with feature weighting performed particularly well when the data was limited, as in Dataset 1. However, the LR model also performed reasonably well, albeit slightly lower when the amount of training data was reduced. However, it should be noted that LR performed less well on datasets where users learned over time, as expected. This is because the model is trained with the entire user history, making it unsuitable for real-world scenarios.

CONCLUSION AND RECOMMENDATION

Conclusion

This research introduces a system designed to assist project managers in enhancing their project planning process. The system achieves this by conducting a risk analysis based on the project manager's prior professional experiences each time they plan a project milestone. The literature review conducted in this study highlights the two primary categories of ML algorithms that contribute to addressing this challenge: example-based learning and regression models. Models utilizing both of these approaches were developed. Test results demonstrate that both techniques can deliver satisfactory outcomes when applied in real-world scenarios. Moreover, the algorithmic solution was effectively integrated into a platform accessible to project managers. This platform empowers project managers to enhance their efficiency and improve their project success rate by proactively identifying and mitigating potential risks before they manifest.

Recommendation for Future Research

To optimize the performance of this proposed research, potential improvements can be further explored through the analysis of global patterns in future user history. In future research, one interesting approach that can be taken through the application of ML algorithms is k-means Clustering, this can be used to cluster users into groups based on the similarity of their profiles and experiences. Furthermore, enhancing the precision of similar user profiles and histories based on the past experiences of users who share similar profiles will contribute to increased efficiency and effectiveness in similar tasks in the future. Concerning the web platform introduced in this study, the mechanism for linking the web platform with the ML back-end is designed to be quite versatile and adaptable, so that, for future research, it can be changed according to the needs to be more developed.

Acknowledgements

The authors would like to thank the University of Computer Technology (STEKOM University) for supporting this study.

Conflict of Interest

We, Joseph Teguh Santoso; Mars Caroline Wibowo; and Budi Raharjo, authors of the manuscript entitled "The Effect of Information in Cost Accounting Service Companies" declare there have been no known competing financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Authors' Contribution

The authors confirm their contribution to the paper as follows; Author 1: Idealization and conception of the subject and theme of the research, Development of the Theoretical Platform, Design of the methodological approach of the research, Analysis, and interpretations of the collected data, Critical revision of the manuscript, Final writing of the manuscript, Outline of the research methodological approach, Analysis and interpretation of the collected data, Final writing of the manuscript according to the rules established by the Journal, Author 2: Idealization and conception of the subject and theme of the research, Design of the methodological approach of the research, Critical revision of the manuscript, Final writing of the manuscript according to the rules established by the Journal. Author 3: Data collection, Analysis, and interpretation of the collected data, Research conclusions, Critical revision of the manuscript, and Final writing of the manuscript according to the rules established by the Journal

REFERENCES

A. Gupta, P. Roy, P& V. Dutt, "Evaluation of Instance-Based Learning and Q-Learning Algorithms in Dynamic Environments", *IEEE Access*, *9*, 138775–138790, 2021. https://doi.org/10.1109/access.2021.3117855

A.R. Peslak. (2012, July 1). "Information Technology Project Management and Project Success", *International Journal of Information Technology Project Management, 3*(3), 31–44, 2012. https://doi.org/10.4018/jitpm.2012070103

B. Martens, "The Importance of Data Access Regimes for Artificial Intelligence and Machine Learning", *SSRN Electronic Journal*. 2018. https://doi.org/10.2139/ssrn.3357652

B. W. Boehm, "A spiral model of software development and enhancement", *Computer*, *21*(5), 61–72, 1988. https://doi.org/10.1109/2.59

J. Sirignano & K. Spiliopoulos, “Stochastic Gradient Descent in Continuous Time: A Central Limit Theorem”, *Stochastic Systems*, *10*(2), 124–151, 2020. https://doi.org/10.1287/stsy.2019.0050

J.C. Stoltzfus, “Logistic Regression: A Brief Primer”, *Academic Emergency Medicine*, *18*(10), 1099–1104, 2011. https://doi.org/10.1111/j.1553-2712.2011.01185.x

J.M. Wilson, “Gantt charts: A centenary appreciation”, *European Journal of Operational Research*, *149*(2), 430–437, 2003. https://doi.org/10.1016/s0377-2217(02)00769-5

J.T/ Marchewka, *Information Technology Project Management*. John Wiley & Sons, 2016.

JT. Santoso. Introduction to Information Technology Project Management. Prima Agus Teknik Foundation: Indonesia. ISBN: 978-6238-12046-8.2023

K. Thirumoorthy, & K. Muneeswaran, “Feature selection using hybrid poor and rich optimization algorithm for text classification”, *Pattern Recognition Letters*, *147*, 63–70, 2021. https://doi.org/10.1016/j.patrec.2021.03.034

M. Abadi, “TensorFlow: learning functions at scale”, *ACM SIGPLAN Notices*, *51*(9), 1–1, 2016. https://doi.org/10.1145/3022670.2976746

M. Islam, G. Hu & Q. Liu, “Online Model Updating and Dynamic Learning Rate-Based Robust Object Tracking”, *Sensors*, *18*(7), 2046, 2018. https://doi.org/10.3390/s18072046

M.A. Tahir, A. Bouridane, A., & F. Kurugollu, “Simultaneous feature selection and feature weighting using Hybrid Tabu Search/K-nearest neighbor classifier”, *Pattern Recognition Letters*, *28*(4), 438-446, 2007.

Masahiro Shibuya, & Xuebin Chen, “Production Planning and Management Using Gantt Charts”, *Journal of Mechanics Engineering and Automation*, *11*(3), 2021. https://doi.org/10.17265/2159-5275/2021.03.002

O. Lemke, & B. Keller, “Common Nearest Neighbor Clustering—A Benchmark”, *Algorithms*, *11*(2), 19, 2018. https://doi.org/10.3390/a11020019

P. Bugata & P. Drotár, “Weighted nearest neighbors feature selection”, *Knowledge-Based Systems*, *163*, 749–761, 2019. https://doi.org/10.1016/j.knosys.2018.10.004

P. Drotár, “Weighted k-Nearest Neighbors Feature Selection (WkNN-FS)”, *Science Trends*, 2019. https://doi.org/10.31988/scitrends.48207

P.W. Cahyo & A. I. Wicaksono, “Django Framework and Python-Gammu as Middleware SMS Broadcast”, *Compiler*, *8*(1), 27, 2019. https://doi.org/10.28989/compiler.v8i1.430

R. Archibald, “A Stochastic Gradient Descent Approach for Stochastic Optimal Control”, *East Asian Journal on Applied Mathematics*, *10*(4), 635–658, 2020. https://doi.org/10.4208/eajam.190420.200420

R.E. Bellman, *Dynamic Programming*. Courier Corporation. 2003.

R.V. Babu, Ayyappan, G., & A. Kumaravel, “Comparison of Linear Regression and Simple Linear Regression for critical temperature of semiconductor”, *Indian Journal of Computer Science and Engineering*, *10*(6), 177–183, 2020. https://doi.org/10.21817/indjcse/2019/v10i6/191006050

S. Castelo, M. Ponti & R. Minghim, “A Visual Mining Approach to Improved Multiple-Instance Learning”, *Algorithms*, *14*(12), 344, 2021. https://doi.org/10.3390/a14120344

S. McConnell, *Rapid development: taming wild software schedules*. Pearson Education. 1996

T. DeMarco, & A. Miller. Managing Large Software Projects. *IEEE Software*, *13*(4), 24, 1996. https://doi.org/10.1109/ms.1996.526827

W. Yaokumah & E. Biney, “Integrated Financial Management Information System Project Implementation in Ghana Government Ministries”, *International Journal of Information Technology Project Management*, *11*(1), 17–31, 2020. https://doi.org/10.4018/ijitpm.2020010102

X. Peng, R. Chen, K. Yu, F. Ye, & W. Xue, “An Improved Weighted K-Nearest Neighbor Algorithm for Indoor Localization”, *Electronics*, *9*(12), 2117, 2020. https://doi.org/10.3390/electronics9122117

Y.T.J Kiran, “Computer vision accuracy analysis with deep learning model using tensorFlow”, *International Journal of Innovative Research in Computer Science & Technology*, *8*(4), 2020. https://doi.org/10.21276/ijircst.2020.8.4.13

Z. Hajizadeh, M. Taheri & M.Z. Jahromi, “Nearest Neighbor Classification with Locally Weighted Distance for Imbalanced Data”, *International Journal of Computer and Communication Engineering*, *3*(2), 81–86, 2014. https://doi.org/10.7763/ijcce.2014.v3.296

Z. Wang, Y. Zhao, YY. Liu, & C. Lv, “A speculative parallel simulated annealing algorithm based on Apache Spark”, *Concurrency and Computation: Practice and Experience*, *30*(14), e4429, 2018. https://doi.org/10.1002/cpe.4429

Zhang, C., Yao, J., Hu, G., & Sch鵖t, T. (2020). Applying Feature-Weighted Gradient Decent K-Nearest Neighbor to Select Promising Projects for Scientific Funding. *Computers, Materials & Continua*, *64*(3), 1741–1753. https://doi.org/10.32604/cmc.2020.010306