ISSN 2442-6571

Machine learning-based B2C software project success prediction model in Indonesia

Rudi Setiawan [a,1,*], Titik Khawa Abdul Rahman [b,2]

[a] Department of Information Systems, Trilogi University, South Jakarta, 12760, Indonesia
[b] School of Science and Technology, Asia e University, 47500, Shah Alam, Malaysia
[1] rudi@trilogi.ac.id; [2] titik.khawa@aeu.edu.my
* corresponding author

ARTICLE INFO

Article history
Received June 10, 2025
Revised July 3, 2025
Accepted July 30, 2025
Available online August 31, 2025

Keywords
Software project prediction
Project success prediction
Success and failure software project
Software prediction
Software project management

ABSTRACT

The success of a software project is a crucial factor in the information technology industry, but it is often difficult to predict due to its complexity and high dynamics. This research aims to develop a model for predicting the success of software projects, particularly B2C e-business software in Indonesia, utilizing a machine learning approach. This study involved 28 variables that affect the success of software projects obtained from previous research. The dataset was compiled from the historical records of software projects from various software development companies in Indonesia. The predictive model was developed using Support Vector Machine and Artificial Neural Network algorithms, with hyperparameter tuning performed via Grid Search. The modelling process includes the pre-processing stage of data, which involves synthetic data generation due to inadequate data collection, as well as the application of several dataset mining techniques (SMOTE, ADASYN, SMOTE Tomek Links, and ADASYN Tomek Links). Additionally, model training and performance evaluation are conducted using a confusion matrix. The search for important features using the Shapley Additive Explanations method is also conducted to develop an automated recommendation system based on key factors that require improvement. The results showed that the SVM model with Grid Search tuning of hyperparameters in the SMOTE Tomek Links data test yielded the best performance, with an accuracy of 87.8%, demonstrating the significant potential of machine learning in identifying project success factors from the early stages. This study contributes to the development of decision-support tools for B2C project managers in Indonesia by providing accurate early predictions and interpretable recommendations.

1. Introduction

In the rapidly growing field of software development, the success of software projects is critical for organizations because it provides broad strategic benefits, including operational efficiency and driving product innovation in a competitive market [1], [2]. However, research shows that most projects do not meet their objectives [3], [4] most software projects exceed their established time and budget constraints [5], which often leads to financial losses and waste of resources [6]. The Chaos report from Standish Group highlights that around 65% of software projects face challenges of exceeding budget, being late, or failing due to not meeting quality standards [7]. The alarming trend of software project failures makes it imperative to develop predictive models [8], to be able to assess the likelihood of project success or failure early on [9], [10], thus enabling proactive intervention against the factors that cause failure.

Many factors affect the success of software projects [11], [12], but the Project Planning and Requirement factor is a common factor used by several researchers, including [13]–[15]. Other researches [16], [17] have explored various factors and methodologies to predict the success of a software project. Similarly, [18] conducted a comprehensive analysis of software development factors globally, using machine learning techniques such as logistic regression, Decision Trees, and support vector machines. The results showed that logistic regression and decision tree models achieved the highest level of accuracy, demonstrating the effectiveness of machine learning in risk prediction.

Despite advances in predictive modeling for software project success, some gaps remain in the existing literature, including limited coverage of factors such as human, organizational, and environmental factors, as noted in studies [19]. Many studies focus on a narrow set of factors that affect the success of a project, often overlooking other factors of the software project. The existence of these diverse success factors for software projects is what makes the research in this field a challenge [20]. Another gap is related to the development of models that predict the success of software projects based on specific datasets or case studies, which limits their application in a variety of contexts, making the existing model not universally applicable to all software development environments. The need for an effective software project success prediction model is critical in today's competitive landscape. In fact, there is no consensus on which method is best to predict the success or failure of a software project [21], given that the different characteristics of the data collected, the very different types of software projects, and the different factors of technology acceptance in each country also influence the success of a software project [22]. In the case of Indonesia, research on the success of software projects is still limited to the factors that affect them, so to help prevent software project failures, more research is needed on how to assess or predict the success of software projects based on the type of software developed to avoid project failure early on, considering the consequences of software project failure can have an impact in terms of cost and reputation [23], [24]. The purpose of this study is to:

- Construct a machine learning model to predict the success of Software Projects, especially B2C e-business software in Indonesia.
- Offers an automated recommendation system for performance improvement based on the most significant determinants of software project success.

This study will not only contribute to the academic literature but also provide a decision-support tool for project managers, ultimately leading to increased project success and improved resource allocation.

2. Method

2.1. Research Design

This research is a quantitative experimental study that employs a data-driven approach to develop a machine learning-based model predicting the success of B2C e-business software projects in Indonesia, utilizing artificial neural network algorithms and support vector machines. Parameter optimization using grid search, dataset resampling, and cross-validation techniques is also applied to help reduce overfitting and provide more reliable model performance estimation. The evaluation of the prediction model's performance is conducted using a confusion matrix, which yields accuracy, recall, precision, and the F1 score. Moreover, the data of new software projects that have never been seen before can be predicted with predictive results in the form of "Successful", "Failed", or "Experience Challenges". The model can also provide automatic recommendations using the Shapley Additive Explanations method to enhance the performance of software projects by addressing identified weaknesses. A more detailed research design is shown in Fig. 1.

A machine learning pipeline for improving software development project performance through predictive modeling and feature analysis. The process begins with data collection, followed by handling data shortages using GANs and addressing class imbalance with resampling methods such as SMOTE, ADASYN, and Tomek Links. The dataset is split into training and test sets, with the training set undergoing K-fold cross-validation to tune ANN and SVM models via Grid Search optimization of parameters such as hidden layers, activation functions, learning rates, kernel type, C, and gamma. The best-performing model, selected based on its accuracy, is validated using a confusion matrix that includes metrics such as accuracy, recall, precision, and F1-score. SHAP analysis is then applied to identify important features, which, combined with expert advice, guide performance improvement recommendations. Finally, the optimized model can be used to predict outcomes for new input data.

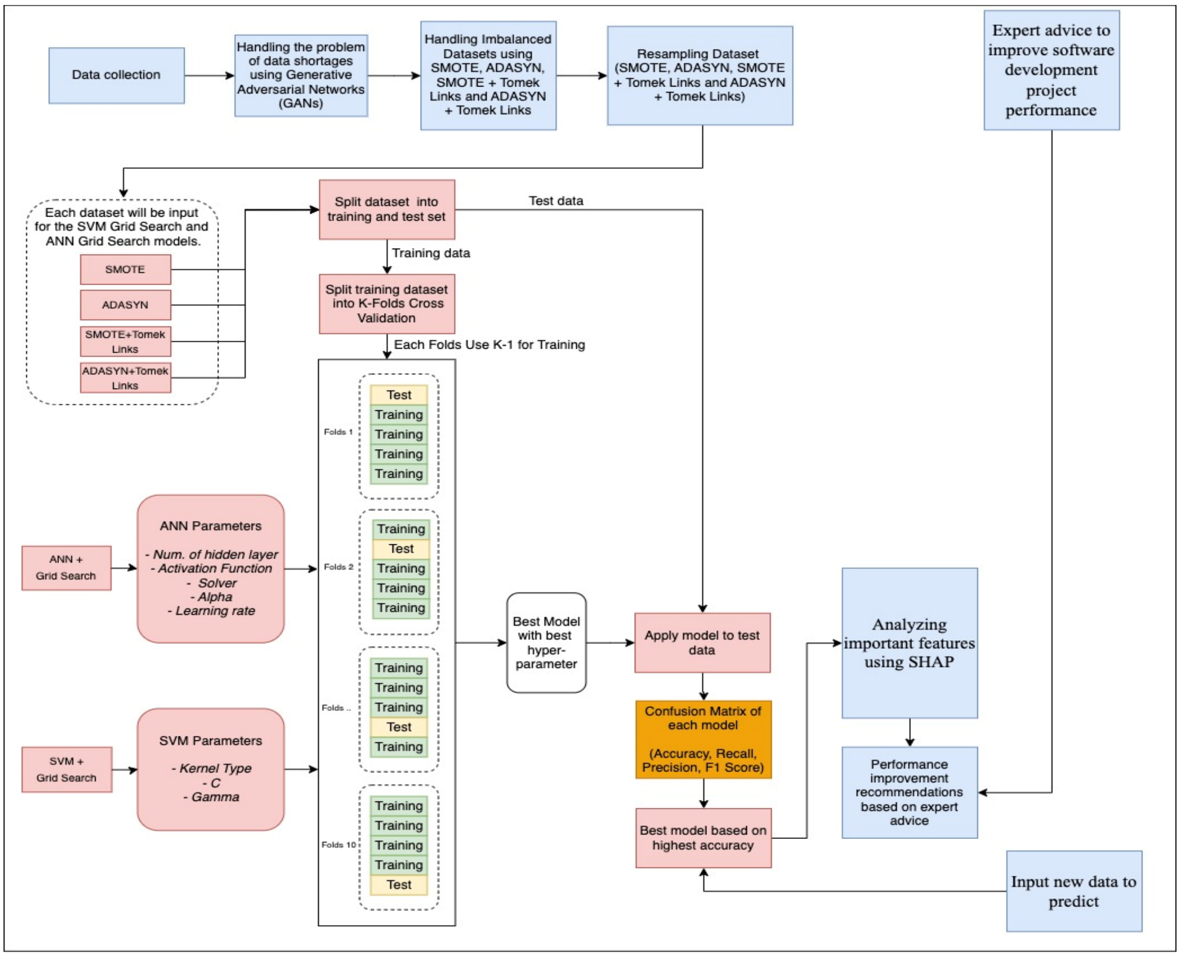

Data collection
Handling the problem of data shortages using Generative Adversarial Networks (GANs)
Handling Imbalanced Datasets using SMOTE, ADASYN, SMOTE + Tomek Links and ADASYN + Tomek Links
Resampling Dataset (SMOTE, ADASYN, SMOTE + Tomek Links and ADASYN + Tomek Links)
Expert advice to improve software development project performance
Each dataset will be input for the SVM Grid Search and ANN Grid Search models.
SMOTE
ADASYN
SMOTE+Tomek Links
ADASYN+Tomek Links
Split dataset into training and test set
Test data
Training data
Split training dataset into K-Folds Cross Validation
Each Folds Use K-1 for Training
Folds 1
Folds 2
Folds ..
Folds 10
Test
Training
ANN + Grid Search
ANN Parameters
- Num. of hidden layer
- Activation Function
- Solver
- Alpha
- Learning rate
SVM + Grid Search
SVM Parameters
- Kernel Type
- C
- Gamma
Best Model with best hyper-parameter
Apply model to test data
Confusion Matrix of each model (Accuracy, Recall, Precision, F1 Score)
Best model based on highest accuracy
Analyzing important features using SHAP
Performance improvement recommendations based on expert advice
Input new data to predict

Fig. 1. Research Design

2.1.1. Dataset and Sampling

The data in this study were obtained from several software development companies operating in Indonesia. Data collection was conducted through the dissemination of a survey questionnaire that was systematically designed and distributed using the snowball sampling technique, a non-probabilistic sampling method in which initial respondents recommend other relevant participants for inclusion [25]. This technique was chosen given the dispersed nature of the target population and its difficulty in being accessed directly. However, the use of snowball sampling methods also raises concerns regarding potential bias, as early respondents tend to recommend individuals with similar characteristics, which can reduce the diversity of the sample and influence the generalization of findings. However, the limitations in identifying and reaching the population at large make this method the most feasible alternative to obtain relevant data in the context of this study.

To determine the minimum number of samples needed, the Lemeshow formula, commonly used in population surveys, is used to calculate the sample size based on established statistical parameters. Assuming a margin of error of 5%, a confidence level of 95%, and a proportion of the population of 50% ($p = 0.5$), the minimum number of respondents needed to produce a statistically generalizable estimate is 385. This number is considered sufficiently representative to reflect the characteristics of the population as a whole in the context of this study. In this study, a sample of 157 data points was successfully collected, which has a very unbalanced target class. Therefore, data augmentation is necessary to meet the minimum data sample requirements.

2.1.2. Data Augmentation

2.1.2.1. Generative Adversarial Networks (GAN)

GAN is based on game theory, consisting of two machine learning models with neural networks trained simultaneously [26]. The generator, typically implemented as a convolutional neural network, aims to produce a sample of synthetic data that is indistinguishable from real data by analyzing the underlying data distribution. In contrast, discriminators are binary classifiers whose job is to distinguish between the original and the generated samples, perfecting the limits of their decisions through backpropagation. This conflicting arrangement encourages an evolutionary arms race, which pushes both networks towards improved performance and realism in the output produced [27]. The extreme situation of class imbalance in the minority data in the "successfully" class, which is only around 11.5%, is the reason for implementing GAN before class balancing with SMOTE and ADASYN. The Generative Adversarial Networks architecture is presented in Fig. 2.

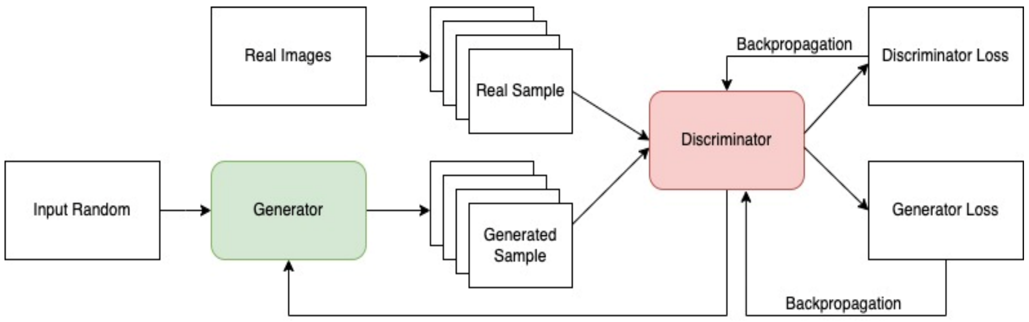


Fig. 2. GAN Architecture

2.1.2.2. Synthetic Minority Oversampling Technique (SMOTE)

SMOTE is an oversampling technique designed to address the problem of class imbalance in training data. When the amount of data in the minority class is much smaller than in the majority class, the machine learning model tends to provide biased predictions for the majority class. To address this, SMOTE generates new synthetic samples from the minority class [28], rather than simply duplicating existing data, as in conventional oversampling methods [29]. The SMOTE process begins by determining the minority class and then identifying the k-nearest neighbors from each minority sample based on Euclidean distance [30]. Once the nearest neighbor is obtained, SMOTE generates new synthetic data by selecting random points between the lines that connect the original minority sample to one of its neighbors. In this way, SMOTE effectively expands the representation of minority data, avoiding the overfitting problems that are common in data duplication. The formulas used in the synthetic data generation process are described in equations 1 and 2.

$$X_{new} = X_i + (\hat{X}_k - X_i) \times \delta \tag{1}$$

$$d(i,j) = \sqrt{\sum_{i=k}^{n} x_{ik} - x_{jk})^2} \tag{2}$$

Where X_{new} is new synthetic data. X_i is the original sample of the minority class, $\widehat{X}_k$ is the closest neighbor of X_i. δ is a random number between 0 and 1. $d(i,j)$ is the Euclidean distance between the data point and, calculated using the square root of the square of the difference of each feature i,j.

2.1.2.3. ADASYN

ADASYN oversampling technique aims to balance the class distribution by adaptively generating new synthetic data, especially in areas where minority data is difficult for the model to learn [31]. Unlike conventional oversampling methods, ADASYN focuses more on generating new data based on data density around minority class points that are difficult to classify [32]. ADASYN's steps in creating synthetic data include calculating the degree of class imbalance using equations 3 to 8.

$$d = \frac{m_s}{m_l} \tag{3}$$

Where M_S is the number of examples of minority classes and M_L is the number of examples of majority classes. Calculate the amount of synthetic data needed for a minority class.

$$G = (m_l - m_s)\beta \tag{4}$$

Where β is the desired minority: majority data ratio after ADASYN. Search for k-Nearest Neighbours from each minority example and calculate the r_i value. Each example of a minority is associated with a different neighborhood.

$$r_i = \frac{\#majority}{k} \tag{5}$$

r_i shows the dominance of the majority class. High r_i scores contain more examples of majority classes and are more difficult to study. Normalize the r_i values.

$$\hat{r}_i = \frac{r_i}{\Sigma r_i} \tag{6}$$

Calculate the number of synthetic examples to be generated.

$$G_i = G_{\hat{r}_i} \tag{7}$$

Since r_i is higher for neighbourhoods dominated by majority class examples, more synthetic minority class examples will be generated for those neighbourhoods.

Generate G_i data for each neighbourhoods. First, take the minority example for the neighbourhoods, x_i. Then, randomly select another minority example within that neighbourhood, x_{zi}. The new synthetic example can be calculated using

$$S_i = x_i + (x_{zi} - x_i)\lambda \tag{8}$$

In the above equation, λ is a random number between 0–1, S_i is the new synthetic example, x_i and x_{zi} are two minority examples within the same neighborhoods.

2.1.2.4. SMOTE Tomek Links

SMOTE Tomek Link is a combined technique in handling unbalanced data that combines oversampling and undersampling techniques [33]. SMOTE aims to increase the number of samples in minority classes through a synthetic oversampling process, while Tomek Links is used to clear the boundaries between classes by removing data pairs that are considered overlapping [34]. The SMOTE Tomek Link algorithm is described in the Pseudocode in Fig. 3.

The SMOTE–Tomek algorithm is a hybrid approach designed to tackle the problem of imbalanced datasets by not only oversampling the minority class but also cleaning the resulting dataset to improve model performance. In the first phase, the SMOTE generates artificial samples for the minority class. This is achieved by selecting random nearest neighbors of existing minority instances and creating new synthetic samples through interpolation of feature values. By doing so, SMOTE reduces the risk of overfitting that occurs when simply duplicating minority samples and helps balance the dataset distribution. In the second phase, the Tomek Links method is applied to the oversampled dataset. Tomek links are pairs of samples, one from the minority class and one from the majority class, that are each other's nearest neighbors. These pairs often exist at the decision boundary, where class overlap or noise is common. Removing them eliminates ambiguous and borderline cases, thereby clarifying class boundaries and enhancing the separability of the data. As a result, the final dataset produced by SMOTE–

Tomek is not only balanced but also cleaner, as it reduces noise and overlap. This makes it particularly suitable for enhancing the performance of machine learning models in classification tasks, thereby ensuring improved accuracy, recall, and generalization.

Algorithm 1 SMOTE-Tomek Algorithm

```
Require: Training set $D$ with minority class $M$ and majority class $M'$, SMOTE ratio $N$, Tomek links threshold $T$
Ensure: Oversampled training set
1:  $D_{SMOTE} \leftarrow D$                                    ▷ Initialize SMOTE dataset
2:  for all $m \in M$ do
3:    for $i = 1$ to $N$ do
4:        $nn \leftarrow$ select a random instance from $k$ nearest neighbors of $m$    ▷ Select a nearest neighbor
5:        for all $f \in$ features do
6:            $m_f \leftarrow m_f + rand(0,1) \times nn_f - m_f)$        ▷ Generate synthetic instance
7:        end for
8:        $D_{SMOTE} \leftarrow D_{SMOTE} \cup \{m\}$          ▷ Add synthetic instance to SMOTE dataset
9:    end for
10: end for
11: $D_{TOMEK} \leftarrow D_{SMOTE}$                          ▷ Initialize Tomek links dataset
12: for all $m \in D_{SMOTE}$ do
13:    for all $m' \in M'$ do
14:        if $distance(m, m') \leq T$ then
15:            $D_{TOMEK} \leftarrow D_{TOMEK} \setminus \{m, m'\}$          ▷ Remove Tomek links
16:        end if
17:    end for
18: end for
19: return $D_{TOMEK}$                     ▷ Return the final oversampled dataset
```

Fig. 3. Pseudocode SMOTE Tomek Links

2.1.2.5. ADASYN Tomek Links

ADASYN is an oversampling method that generates synthetic data adaptively based on the learning difficulty of the minority sample, thereby focusing more on the most challenging areas to classify. Meanwhile, Tomek Links serves as an undersampling method that removes data at class boundaries that could potentially cause ambiguity [35]. The ADASYN Tomek Link algorithm is described in the Pseudocode in Fig. 4.

Algorithm 2 ADASYN + TomekLinks

```
(a) Apply ADASYN until the number of minority samples increases with the desired proportion as shown in Algorithm 1;
(b) Loop over minority class samples;
(c) Find the nearest neighbor to the sample;
(d) Remove the nearest neighbor if it belongs to a majority class;
```

Fig. 4. Pseudocode ADASYN Tomek Links

2.1.2.6. Removing Outliers

Data cleansing from outliers is performed by first reducing the features to 2 dimensions using Principal Component Analysis (PCA), which projects the data into a low-dimensional space to detect deviations based on the distance of the data from the average distribution. The first step is to calculate the Euclidean distance between each synthetic data point and the original data point using equation 9. The next step is to determine the threshold for identifying synthetic data that is considered deviant, using the mean and standard deviation of the distance. Synthetic data with a distance greater than the mean plus 2 times the standard deviation is considered deviant.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \quad (9)$$

where q_i represents a synthetic data vector, and p_i denotes the original data vector.

2.2. Model Used

Selecting the correct machine learning algorithm is crucial in building a predictive model of software project success. In this study, two popular and representative algorithms from different machine learning approaches, namely Support Vector Machine (SVM) and Artificial Neural Networks (ANN), were used, considering the advantages of their respective characteristics. SVM fundamentally works with margin-based learning approaches and representation-based learning ANN. With these two different approaches, it can produce a comprehensive evaluation of algorithm performance in the context of software project prediction. The combination of these two models also provides deep insights into the trade-offs between model accuracy in data-driven decision-making.

2.2.1. Support Vector Machine

The Support Vector Machine algorithm was first introduced by [36], carrying the concept of optimal hyperplane search that maximizes the margin between two classes. SVM was chosen for its ability to handle small to medium-sized datasets with excellent generalization performance [37]. In the case of data that cannot be separated linearly, SVM leverages kernel functions to map data to higher-dimensional spaces so that they can be separated linearly together [38], [39]. The concept of a hyperplane in SVM is described in Fig. 5.

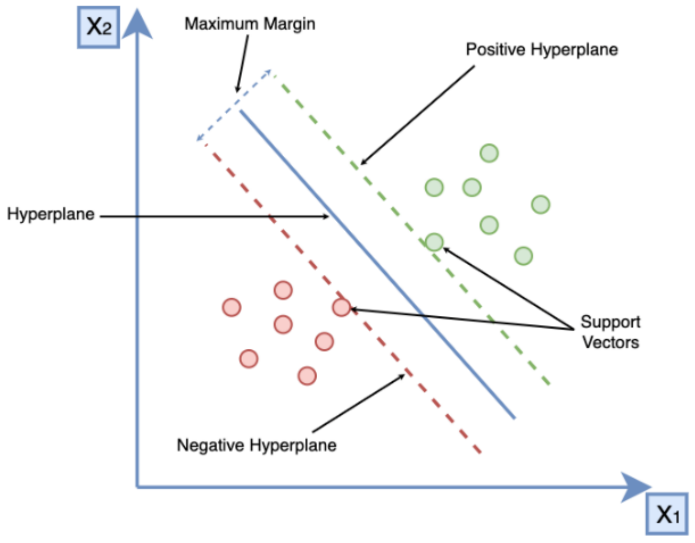


Fig. 5. SVM Hyperplane Concept

In the case of class 2 classification, the SVM formula is described in equation 10, where is the classification decision function used in the positive or negative class.

$$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + b \tag{10}$$

Where N is sum of vector. α_i is a value from data i, and $K(x, x_i)$ is kernel function and b is bias

2.2.2. Artificial Neural Networks

An Artificial Neural Network is one of the machine learning algorithms inspired by how biological neural networks work in processing information [40], [41]. ANN are made up of a number of layers of artificial neurons that are interconnected and work gradually to extract complex patterns in the data [42]. Through a training process based on error propagation (backpropagation), an ANN can adjust the weight of connections between neurons to minimize prediction errors [43]–[45]. In this study, the ANN algorithm was employed due to its ability to handle nonlinear relationships and detect hidden relationships between variables. The Neural Network function is defined in equation 11.

$$f(x) = K(\sum_i w_i . x_i + b) \tag{11}$$

where, (K) is a special function which is often called the activation function, (w) is the weight of the relationship between neurons and (b) is the bias value. The relationship between the weight of each element and the input and output of the ANN system is shown in Fig. 6.

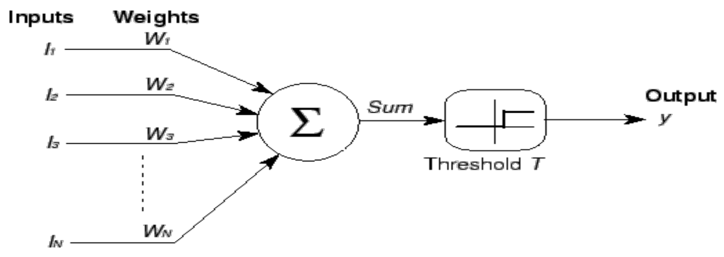


Fig. 6. Weight of each element and input and output of the ANN system

2.2.3. Hyperparameter Tuning Using Grid Search

Hyperparameter tuning with the Grid Search method is a process of systematically exploring parameter space by trying all possible combinations, model performance evaluation is carried out using a confusion matrix to obtain accuracy, precision, recall and F1-score values, through cross-validation. With this approach, the resulting model has the most appropriate parameter configuration to obtain the best accuracy. The steps of hyperparameter tuning with grid search are described in the study [46] as define the hyperparameter list and its value range, define grid combinations, evaluate each combination, and choose the best combination. In this study, several parameters that will be set in ANN are hidden layers, activation functions, solvers, alpha, and learning rates. While for the SVM model, the parameters to be set are Gamma, C and Kernel. The initialization of the values for each hyperparameter to be set is presented in Table 1.

Table 1. Initialization of Values for Each Hyperparameter in ANN and SVM

ANN Hyperparameter	Initial Value	SVM Hyperparameter	Initial Value
Hidden Layer	(50,), (100,), (50, 50)	Gamma	1, 0.1, 0.01, 0.001
Activation Function	Relu, Tanh, Logistic	C	0.1, 1, 10, 100
Solver	Adam, Sigmoid	Kernel	Linear, Poly, RBF , Sigmoid
Alpha	0.0001, 0.001, 0.01		
Learning Rate	Constant, Adaptive		

Based on the number of initialized values for each parameter, Grid Search will systematically explore all possible combinations of the initialized hyperparameters using cross-validation with a total of 10 folds.

2.2.4. Evaluation Metrics

In this study, the model evaluation was carried out using a confusion matrix to obtain True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), which were then calculated into the Accuracy, Precision, Recall, and F1 Score metrics. Accuracy measures the proportion of correct predictions to the total data, but accuracy tends to give the illusion of high performance if the model relies only on the majority class, without regard to the minority class [47]. Precision indicates the accuracy of positive predictions, while Recall measures the ability to detect all positive cases. F1 Score as a harmonized average of Precision and Recall provides a balanced evaluation when the two metrics are not equivalent, complemented by statistical analysis to ensure significant performance differences, as well as standardization of implementation to make evaluation results more consistent and reliable, especially on classification issues with class imbalances.

2.2.5. Feature Importance Using Shapley Additive Explanations (SHAP)

SHAP is an interpretability method developed to explain the contribution of each feature to the output of a machine learning model quantitatively. This method adopts the Shapley value concept from cooperative game theory to calculate the average contribution of features in various combinations, resulting in a fair, consistent, and mathematically accountable explanation [48]. In this study, SHAP was used to evaluate the influence of features on model predictions and support transparent interpretation of results. SHAP calculates the contribution value of feature i using equation 12.

$$\phi_i = \sum_{S \subseteq F\{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!}(f(S \cup \{i\} - f(S)) \tag{12}$$

where ϕ_i is the SHAP values for features i. S is subset of features without features i. F is all features in the model. $f(S)$ is Model prediction when only the subset S used. $(f(S \cup \{i\}$ is model prediction when feature i is added to subset S. and $|S|!\,(|F|-|S|-1)!/|F|!$ is weights that ensure fair calculation for all possible subsets.

3. Results and Discussion

3.1. Results of Handling Data Shortage Using Generative Adversarial Networks (GAN)

The dataset collection process is a very crucial stage. In this study, the data collection process encountered obstacles due to the difficulty of reaching IT companies in Indonesia, resulting in the minimum dataset requirements not being met. This experience aligns with the findings of a study conducted by [49]. To meet the minimum dataset requirements, a synthetic data generation technique was employed using the GAN method. A comparison of the initial dataset and the synthetic dataset created is presented in Table 2.

Table 2. Comparison of the Number of Original and Synthetic Sample Datasets Using GAN

Class Target	Original Dataset	Synthetics Dataset	Number of Dataset
Failed	38	53	91
Experience Challenges	101	142	243
Successful	18	33	51
	157	228	385

Based on the comparison table of the number of original and synthetic sample datasets using GAN, it appears that the target class “Experience Challenges” has more than the target classes "Failed" and "Successfully”. For the target class with the least number of targets, there is a target class "Successfully”. The proportion of data by target class in the synthetic data results appears to follow the original data pattern, indicating that the synthetic datasets generated by GAN successfully replicate the category distribution pattern of the original dataset. However, this cannot replace the true diversity of data.

3.2. Results of Handling Imbalanced Dataset and Removing Outliers

To address the problem of class imbalance in datasets, several data aggregation techniques were applied, including the use of GAN combined with classic oversampling strategies such as SMOTE and ADASYN, as well as a hybrid approach involving Tomek Links. The distribution of synthetic samples in each class before and after outlier removal was analyzed to evaluate the effectiveness of each method.

The use of GAN alone resulted in a highly unbalanced distribution, with only 91 samples produced for the “Failed” class, 243 for the “Experience Challenges” class, and 51 for the “Successfully” class, totaling 385 synthetic samples. These results reflect a known tendency for GAN to prioritize the majority class when trained on unbalanced data, leading to inadequate coverage of minority classes.

When the GAN was combined with SMOTE, the class distribution was successfully balanced, with 243 samples for each class (729 in total). After eliminating the outliers, the final sample distribution decreased slightly to 232, 240, and 235 for the classes "Failed", "Experience Challenges", and "Successfully", with a total final result of 707 samples. A similar pattern was observed with the combination of GAN and ADASYN, where balanced initial generation (241, 243, 245 samples) also resulted in post-outlier distributions of 232, 240, and 235, indicating resilience in maintaining class balance even after data cleansing.

The hybrid method, which integrates oversampling with Tomek Links to reduce class overlap and eliminate boundary line interference, shows a slightly reduced sample size after cleaning. The GAN-SMOTE-Tomek approach resulted in 240, 239, and 242 samples in three classes, which were then reduced to 229, 235, and 235, for a total of 699 samples that were free of outliers. Meanwhile, the GAN-ADASYN-Tomek hybrid initially produced 241, 241, and 245 samples, resulting in 231, 235, and 232 after the total outlier removal of the entire data set to 698 data samples.

From these results, it is clear that combining GAN with traditional oversampling techniques effectively reduces class imbalances, while hybridization with Tomek Links further improves sample quality by reducing the number of unclear or potentially biased synthetic samples. Although this hybrid approach slightly reduces the total number of samples that can be used, it compensates for improvements in class segregation and data integrity, which are critical for classification tasks. Table 3 shows the distribution of data for each class on the various resampling techniques applied.

Table 3. Data Distribution for Each Class

Dataset	Class			Number of Dataset
	Failed	*Experience Challenges*	*Successfully*	
Synthetics GAN + SMOTE	243	243	243	729
After Removing Outliers	232	240	235	707
Synthetics GAN + ADASYN	241	243	245	729
After Removing Outliers	232	240	235	707
Synthetics GAN hybrid SMOTE and Tomek Links	240	239	242	721
After Removing Outliers	229	235	235	699
Synthetics GAN Hybrid ADASYN and Tomek Links	241	241	245	727
After Removing Outliers	231	235	232	698

Fig. 7 presents a visualization of the outlier detection results, where the blue dots represent the main data, while the red dots marked with a cross indicate data identified as outliers. Visually, the majority of the data is evenly distributed around the center coordinates, forming a pattern resembling a two-dimensional normal distribution. The outlier points are spread in areas relatively far from the center of this distribution, indicating that the applied method successfully identified observations that have a much lower local density compared to their neighbors. This reflects the effectiveness in revealing anomalies in two-dimensional datasets that have a dense distribution in the center and are more sparse at the edges.

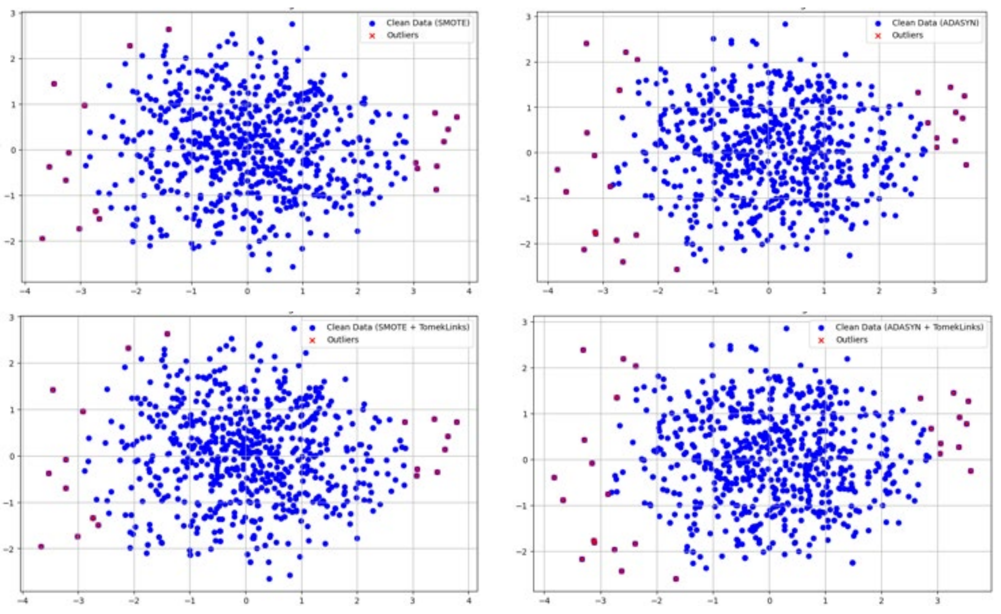


Fig. 7. Outlier Detection Visualization

3.3. Evaluation Model

The results of the experiment showed that overall, the SVM model performed consistently better than ANN on all evaluation metrics, namely accuracy, precision, recall, and F1-score. The best combination is shown by SVM with hyperparameter tuning using Grid Search combined with SMOTE + Tomek Links, with an accuracy value of 0.88, precision 0.90, recall 0.88, and F1-score of 0.88. Meanwhile, the ANN model shows a relatively lower performance, with the best results achieved when combined with SMOTE (accuracy and F1-score of 0.70). These findings confirm that the selection of the right algorithm and data balancing techniques has a significant impact on prediction accuracy, and that the SVM approach with oversampling and data cleaning techniques such as SMOTE + Tomek Links is highly recommended in the context of predicting the success of B2C E-Business software projects in Indonesia. Table 4 shows the results of the model evaluation comparison.

Table 4. Model Evaluation Comparison

Model	Imbalanced Dataset Technique	Accuracy	Precision	Recall	F1-Score
ANN Grid Search	SMOTE	0.70	0.71	0.70	0.70
	ADASYN	0.54	0.55	0.54	0.54
	SMOTE + Tomek Links	0.59	0.60	0.59	0.59
	ADASYN + Tomek Links	0.61	0.62	0.61	0.61
SVM Grid Search	SMOTE	0.84	0.86	0.84	0.85
	ADASYN	0.87	0.89	0.87	0.87
	SMOTE + Tomek Links	0.88	0.90	0.88	0.88
	ADASYN + Tomek Links	0.79	0.84	0.79	0.80

Based on the results of the evaluation carried out on all prediction models that have been developed, it was found that the best model performance based on the highest accuracy value was owned by SVM with hyperparameter tuning using Grid Search combined with SMOTE + Tomek Links. To find out how well the model predicts the target class, it can be done by comparing the model's prediction results with its actual value which can be seen in Fig. 8.

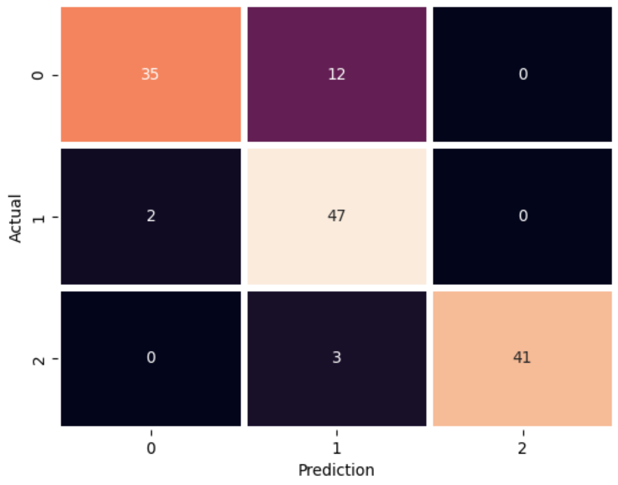


Fig. 8. Confusion Matrix SVM + Grid Search and SMOTE + Tomek Links Model

Based on the confusion matrix, the model shows an overall accuracy of 88%, which is calculated from the ratio of the correct predictions to the total data. Further analysis of the evaluation metrics per class revealed that the model achieved the highest accuracy in Class 2 (Successful Project), with a score of 1.00, indicating that all predictions for this class were correct. The recall for Class Successfully is also high, at 0.932, indicating that most successful projects are well recognizable by the model. For Class 1 (Challenging Projects), the accuracy was recorded at 0.758, while the recall reached 0.959. This indicates that the model performs well in identifying projects that are experiencing challenges, although there are some errors in the predictions. Meanwhile, Class 0 (Failed Project) has a precision of 0.897 and a recall of 0.878. While the high precision indicates that the forecasts for failed projects are fairly accurate, the lower recall indicates that the model still struggles to recognize all the projects that actually failed, suggesting that there are challenges in detecting failed projects. This can be due to the similarity in characteristics between failed and challenging projects, so the model has difficulty distinguishing the two.

In this study, SVM excelled due to its ability to handle high-dimensional data from a limited dataset through the implementation of kernel functions, without requiring explicit feature transformations. This makes SVM more efficient and effective in building reliable predictive models. In addition, SVM produces global solutions through a convex optimization approach, while ANN relies on a non-convex training process, so it has the potential to be stuck at local minima.

3.4. Statistical Testing for Classifier Comparison

Statistical testing is carried out to formally validate the differences between the performance of the model that has been developed. The MANOVA test was carried out with 5 repeated measurements per combination and followed by the Post-Hoc test (Tukey HSD) for Accuracy in two factors (Model and Imbalanced Dataset) to identify where the differences lie. The results of the Tukey HSD test are presented in Table 5.

Table 5. Tukey HSD Test Results For Accuracy

Group1	Group2	Mean Diff	P-Adj	Lower	Upper	Reject
ANN + ADASYN	ANN + ADASYN + Tomek Links	0.0705	0	0.0555	0.0854	True
ANN + ADASYN	ANN + SMOTE	0.157	0	0.1421	0.172	True
ANN + ADASYN	ANN + SMOTE + Tomek Links	0.0451	0	0.0302	0.0601	True
ANN + ADASYN	SVM + ADASYN	0.3264	0	0.3114	0.3413	True
ANN + ADASYN	SVM + ADASYN + Tomek Links	0.2534	0	0.2384	0.2683	True
ANN + ADASYN	SVM + SMOTE	0.295	0	0.2801	0.31	True
ANN + ADASYN	SVM + SMOTE + Tomek Links	0.3382	0	0.3233	0.3532	True
ANN + ADASYN + Tomek Links	ANN + SMOTE	0.0866	0	0.0716	0.1015	True
ANN + ADASYN + Tomek Links	ANN + SMOTE + Tomek Links	-0.0253	0.0001	-0.0403	-0.0104	True
ANN + ADASYN + Tomek Links	SVM + ADASYN	0.2559	0	0.241	0.2709	True
ANN + ADASYN + Tomek Links	SVM + ADASYN + Tomek Links	0.1829	0	0.168	0.1978	True
ANN + ADASYN + Tomek Links	SVM + SMOTE	0.2246	0	0.2096	0.2395	True
ANN + ADASYN + Tomek Links	SVM + SMOTE + Tomek Links	0.2678	0	0.2528	0.2827	True
ANN + SMOTE	ANN + SMOTE + Tomek Links	-0.1119	0	-0.1268	-0.0969	True
ANN + SMOTE	SVM + ADASYN	0.1694	0	0.1544	0.1843	True
ANN + SMOTE	SVM + ADASYN + Tomek Links	0.0963	0	0.0814	0.1113	True
ANN + SMOTE	SVM + SMOTE	0.138	0	0.1231	0.153	True
ANN + SMOTE	SVM + SMOTE + Tomek Links	0.1812	0	0.1663	0.1962	True
ANN + SMOTE + Tomek Links	SVM + ADASYN	0.2812	0	0.2663	0.2962	True
ANN + SMOTE + Tomek Links	SVM + ADASYN + Tomek Links	0.2082	0	0.1933	0.2232	True
ANN + SMOTE + Tomek Links	SVM + SMOTE	0.2499	0	0.235	0.2648	True
ANN + SMOTE + Tomek Links	SVM + SMOTE + Tomek Links	0.2931	0	0.2781	0.308	True
SVM + ADASYN	SVM + ADASYN + Tomek Links	-0.073	0	-0.088	-0.0581	True
SVM + ADASYN	SVM + SMOTE	-0.0313	0	-0.0463	-0.0164	True
SVM + ADASYN	SVM + SMOTE + Tomek Links	0.0119	0.2047	-0.0031	0.0268	False
SVM + ADASYN + Tomek Links	SVM + SMOTE	0.0417	0	0.0267	0.0566	True
SVM + ADASYN + Tomek Links	SVM + SMOTE + Tomek Links	0.0849	0	0.0699	0.0998	True
SVM + SMOTE	SVM + SMOTE + Tomek Links	0.0432	0	0.0282	0.0581	True

In Table 5 of the Tukey HSD test results. The "Reject" column with a value of True/False indicates a hypothesis result of zero (there is no difference between group1 and group 2). If "True", it indicates a statistically significant difference in average accuracy between the two groups at a significance level of 0.05. Of all the pairs compared, there was 1 pair of SVM + ADASYN and SVM + SMOTE + Tomek Links groups that did not have significant differences in accuracy.

3.5. Feature Importance Based on the Best Model (SVM+Grid Search using resampling dataset SMOTE + Tomek Links) in data testing

Fig. 9 shows a SHAP summary plot visualization that shows the contribution of each feature to the output of the software project's prediction model at the individual instance level of the testing data. The model used is multi-classification, with three output classes: project failed, project experience challenges, and project successfully. Based on the SHAP results from the analysis of important features in data testing, the Efficient Management feature has the most dominant influence on the prediction of the success of software projects. The ability to optimally manage resources, time, costs, and project activities reflects efficient management which is a key indicator of success in a software project. The Organizational Culture and Management Style feature also shows a high contribution, which indicates that a good Organizational Culture and Management Style can push projects out of the risk zone. The Team Commitment and Participation factor, which refers to the level of involvement, loyalty, and dedication of team members to the goals and implementation of the project, has a very crucial role because the project is collaborative and highly dependent on the active contribution of all team members. Highly committed teams are typically better able to maintain productivity, avoid destructive conflicts, and complete projects on time.

Technical factors such as "Technical Complexity" and "Customer Skill, Training, and Education in IT" also show considerable influence, indicating the importance of technological readiness and user capacity in the success of the project. In contrast, factors such as "Urgency", "Scope", and "Project Performance" had relatively small SHAP values, suggesting that their influence on the model was lower than that of other factors. Thus, this analysis provides a deeper understanding of the key factors that drive model decisions and can be a basis for strategic decision-making in the context of software project management. Overall, the results of SHAP's interpretation provide a more transparent understanding of the machine learning model's decision-making process and can be used as a strategic reference for stakeholders to focus attention on the key factors that determine the success of a software project.

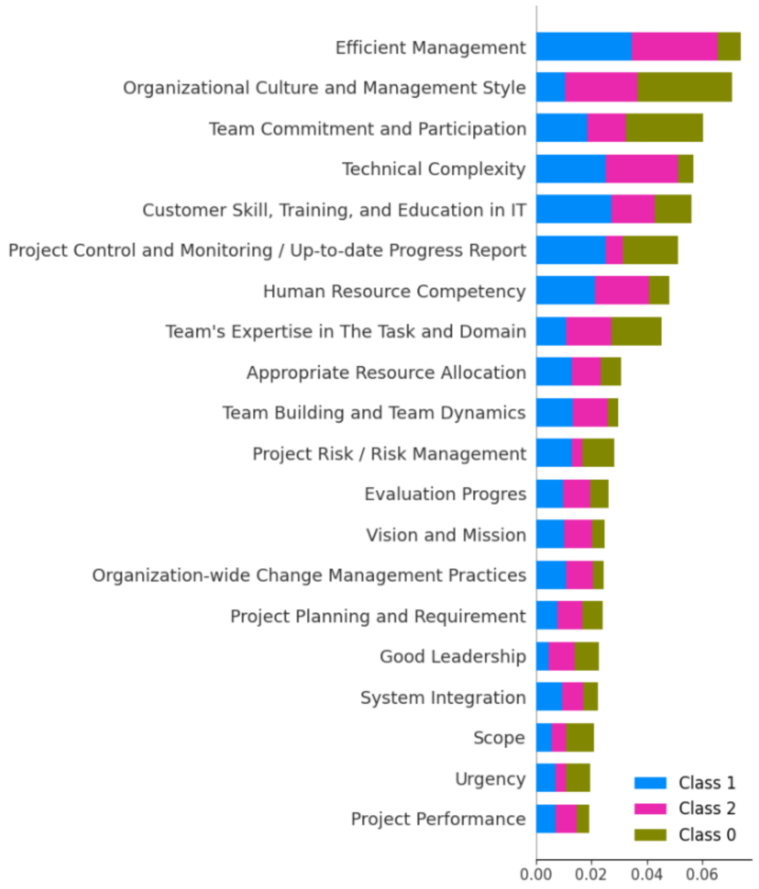
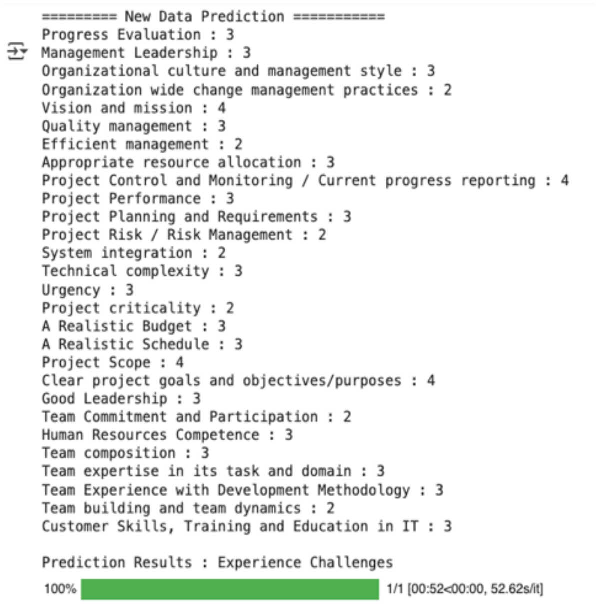


Fig. 9. The average SHAP value of each feature in the test data against the model output

3.6. Testing With New Data

Once the development phase of the software project's success prediction model is complete, the next important step is to test the new data. This process aims to evaluate the model's generalization ability in predicting outcomes on data that was not used during the training process. In this test, the new data used is dummy data, while the model used for prediction is an SVM machine learning model with hyperparameter tuning using Grid Search. The model is trained on a dataset that has been balanced with the SMOTE + Tomek Links technique, using the best parameters obtained from the tuning process. A total of 28 questions, in the form of factors that affect the success of the software project, have been identified. These questions will be answered using a 1-4 Likert scale, which indicates the performance of each factor. Fig 10 shows the process of new data input to be predicted to produce a prediction result in the form of a project success category that "has challenges.

The prediction process begins with the pre-processing of new data, which is performed in the same manner as the pre-processing pipeline on the training data. Next, the new data is fed into the trained model to obtain predictions. The prediction output can be labeled "successful", "experience challenges" and "failed". The analysis of the contribution of the most influential features is carried out using SHAP, the feature with the highest SHAP value is the basis for evaluating performance improvements in features that determine the success of a software project. Thus, the new data testing stage not only ensures that the model works well on training and validation data, but also demonstrates the model's potential to be used as a decision support system in helping project managers and other stakeholders mitigate the risk of project failure early on. Fig. 10 shows the test results with the new data.



```
========= New Data Prediction ===========
Progress Evaluation : 3
Management Leadership : 3
Organizational culture and management style : 3
Organization wide change management practices : 2
Vision and mission : 4
Quality management : 3
Efficient management : 2
Appropriate resource allocation : 3
Project Control and Monitoring / Current progress reporting : 4
Project Performance : 3
Project Planning and Requirements : 3
Project Risk / Risk Management : 2
System integration : 2
Technical complexity : 3
Urgency : 3
Project criticality : 2
A Realistic Budget : 3
A Realistic Schedule : 3
Project Scope : 4
Clear project goals and objectives/purposes : 4
Good Leadership : 3
Team Commitment and Participation : 2
Human Resources Competence : 3
Team composition : 3
Team expertise in its task and domain : 3
Team Experience with Development Methodology : 3
Team building and team dynamics : 2
Customer Skills, Training and Education in IT : 3

Prediction Results : Experience Challenges
100%                                        1/1 [00:52<00:00, 52.62s/it]
```

Fig. 10. Prediction On New Data

Based on the results of the new data predictions shown in Fig. 10, various challenges related to experience in the implementation of information technology projects can be identified. Aspects such as Organizational-wide change management practices (2), Efficient management (2), Team Commitment and Participation (2), Team building and team dynamics (2), and Project criticality (2) showed relatively low scores. This indicates that organizations face obstacles in managing change broadly, maintaining team commitment, and building effective teamwork dynamics. Meanwhile, several other factors showed relatively high scores, such as Vision and mission (4), Project Scope (4), and Project Control and Monitoring (4), indicating strength in project planning and supervision. These results are then classified by the prediction model into the category of Experience Challenges. These findings imply the need to increase organizational capacity in the aspects of change management and human resource management so that project implementation can run more effectively and efficiently.

In this research, automatic recommendations are given based on the highest SHAP value which is identified as feature importance in the newly input data. The recommendations presented in Table 6 are practical in nature by involving software development experts who have project management certification with the aim of improving the effectiveness of project management through the use of appropriate managerial tools, methods, and approaches. These recommendations can be used as an initial guide in developing project performance improvement strategies based on the priority of influential factors.

Table 6. Feature Importance and Recommendation

Factors	SHAP Value	Recommendation
Project Control and Monitoring / Current progress reporting	0.0561	Use automation tools for data-driven project monitoring, such as using JIRA tools or similar software.
Team composition	0.0532	The team composition can be adjusted to the complexity of the project being worked on.
Project Performance	0.0453	Use measurable metrics, such as function points or velocity.
Urgency	0.0427	Use methods like the Critical Path Method to manage urgent tasks.
Efficient management	0.0269	Use a proven project management methodology, such as Lean Project Management.
Good Leadership	0.0203	Good leadership must understand both technical and managerial aspects to handle challenges that arise in a project.
Customer Skills, Training and Education in IT	0.0133	Provide role-based training to improve the effectiveness of system use.

3.7. Discussion

Although the results of this study show that the SVM algorithm included with the application of data balancing techniques through the combination of SMOTE and Tomek Link shows superior performance compared to other models in predicting the success of software projects in the context of B2C projects in Indonesia, it should be noted that the generalization of these results to different project types, such as B2B software projects, It cannot be done directly without further validation, given that there are different success factors in each type of software project. From the perspective of geographical and organizational context, the software development ecosystem in Indonesia has differences in terms of project management maturity, process standards, and work culture compared to countries with more established digital infrastructure, so model adaptability needs to be considered when implementing prediction models to different regions or domains. Although the structure of the SVM algorithm is universal, its performance also depends on the representation of local data that is appropriate to the context, so retraining with local data and adapting features through techniques such as domain adaptation are important steps to ensure that the predictive models developed can be more accurate, adaptive, and applicable in different project domains.

4. Conclusion

This study has succeeded in developing a model of predicting the success of software projects, especially for B2C e-business software with datasets collected from the history of software development from various companies in Indonesia, by involving 28 success factors of software projects that have been identified from previous research, a model developed with the Support Vector Machine algorithm by resampling the results of the imbalanced dataset SMOTE Tomek Links has the highest accuracy compared to other models. In addition, the application of Shapley Additive Explanations to identify important features that affect the results of the prediction model based on the highest SHAP value can be a guide in providing an automatic recommendation system, so that it can be used to improve factors that need attention. The results of this study open up various directions for further development to improve the accuracy and generalization of predictive models for the success of software projects. Some of the proposed future jobs that can be done are 1) Larger-scale dataset collection from various project domains (B2B, B2G) and across countries is highly recommended to obtain more reliable findings for specific project domains; 2) Further studies can be directed towards real-time predictive model testing on dashboard-based project management platforms (Jira, Trello, or Microsoft Project). This integration will enable the evaluation of model performance in a dynamic context as well as support direct data-driven decision-making by project managers; and 3) The incorporation of natural language features from project documents, such as status reports, meeting minutes, and risk notes, can enrich the input model with semantic dimensions that have not been utilized so far. By applying Natural Language Processing, prediction models can capture the nuances of communication, issue escalation, and team dynamics that impact the final project outcome; 4) Further exploration can be carried out on the application of explainable AI techniques to improve the interpretability of the model, so that the prediction results are not only accurate but also can be understood and justified by the project stakeholders

Acknowledgment

With all humility, I express my deepest gratitude to all parties who have provided support and contributions in completing this research. I would like to express my gratitude to Trilogi University, which has fully supported the completion of this research. I would also like to express my sincere appreciation and respect to the supervisor who has patiently guided me, provided valuable input, and built my enthusiasm during the research process. I would also like to express my gratitude to the National Research and Innovation Agency (BRIN) for the support of the supercomputer facility (MAHAMERU HPC), which I really need in implementing and completing this research.

Declarations

Author contribution. All authors contributed equally to the main author of this paper except in the development of the AI model, the main author, who also serves as the corresponding author played a full role in this task, and all authors read and approved the final paper
Funding statement. There was no funding support for this research
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

[1] M. A. Akbar, A. A. Khan, S. Mahmood, and K. Smolander, "Successful management of cloud-based global software development projects: A multivocal study," *J. Softw. Evol. Process*, vol. 36, no. 4, p. e2527, Apr. 2024, doi: 10.1002/smr.2527.

[2] A. A. Vărzaru and C. G. Bocean, "Digital Transformation and Innovation: The Influence of Digital Technologies on Turnover from Innovation Activities and Types of Innovation," *Systems*, vol. 12, no. 9, p. 359, Sep. 2024, doi: 10.3390/systems12090359.

[3] E. Loureiro, B. Gomes, J. Varajão, and C. Silva, "Information systems project success surveys - Insights from the last 30 years," *Heliyon*, vol. 10, no. 23, p. e40619, Dec. 2024, doi: 10.1016/j.heliyon.2024.e40619.

[4] M. L. Prasetyo, R. A. Peranginangin, N. Martinovic, M. Ichsan, and H. Wicaksono, "Artificial intelligence in open innovation project management: A systematic literature review on technologies, applications, and integration requirements," *J. Open Innov. Technol. Mark. Complex.*, vol. 11, no. 1, p. 100445, Mar. 2025, doi: 10.1016/j.joitmc.2024.100445.

[5] A. Ardiansyah, M. I. Zulfa, A. Tarmuji, and F. H. Jabbar, "Optimization of use case point through the use of metaheuristic algorithm in estimating software effort," *Int. J. Adv. Intell. Informatics*, vol. 10, no. 1, p. 109, Feb. 2024, doi: 10.26555/ijain.v10i1.1298.

[6] J. Schmidt, "Mitigating risk of failure in information technology projects: Causes and mechanisms," *Proj. Leadersh. Soc.*, vol. 4, p. 100097, Dec. 2023, doi: 10.1016/j.plas.2023.100097.

[7] "CHAOS Report Beyond Infinity," *The Standish Group*, 2020. [Online]. Available at: https://www.standishgroup.com/.

[8] E. M. M. Alzeyani and C. Szabó, "Comparative Evaluation of Model Accuracy for Predicting Selected Attributes in Agile Project Management," *Mathematics*, vol. 12, no. 16, p. 2529, Aug. 2024, doi: 10.3390/math12162529.

[9] L. Madeyski and S. Stradowski, "Predicting test failures induced by software defects: A lightweight alternative to software defect prediction and its industrial application," *J. Syst. Softw.*, vol. 223, p. 112360, May 2025, doi: 10.1016/j.jss.2025.112360.

[10] M. E. Nenni, F. De Felice, C. De Luca, and A. Forcina, "How artificial intelligence will transform project management in the age of digitization: a systematic literature review," *Manag. Rev. Q.*, vol. 75, no. 2, pp. 1669–1716, Jun. 2025, doi: 10.1007/s11301-024-00418-z.

[11] J. Alqahtani, A. Siddique, A. M. Aseere, A. Alasiry, and Q. N. Naveed, "Evaluating Success Factors of Software Project Management in Global Software Development," *IEEE Access*, vol. 12, pp. 22345–22358, 2024, doi: 10.1109/ACCESS.2024.3360415.

[12] M. Monserrat, A. Mas, and A. Mesquida, "A Study of Factors That Influence the Software Project Success," *J. Softw. Evol. Process*, vol. 37, no. 2, p. e2735, Feb. 2025, doi: 10.1002/smr.2735.

[13] M. N. Mata, J. M. Martins, and P. L. Inácio, "Impact of absorptive capacity on project success through mediating role of strategic agility: Project complexity as a moderator," *J. Innov. Knowl.*, vol. 8, no. 1, p. 100327, Jan. 2023, doi: 10.1016/j.jik.2023.100327.

[14] A. C. P. Junior, V. R. da Silva, and P. T. A. Junior, "Critical Success Factors for Agile Software Development," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 14807–14823, 2024, doi: 10.1109/TEM.2024.3441829.

[15] Ahmad Abdullah N. Alghamdi, "Factors Affecting Success or Failure in Software Projects in the Kingdom of Saudi Arabia," *J. Electr. Syst.*, vol. 20, no. 9s, pp. 2895–2912, Jun. 2024, doi: 10.52783/jes.5051.

[16] L. Barros, C. Tam, and J. Varajão, "Agile software development projects–Unveiling the human-related critical success factors," *Inf. Softw. Technol.*, vol. 170, p. 107432, Jun. 2024, doi: 10.1016/j.infsof.2024.107432.

[17] T. Al-Rousan, "Success Factors for Conducting Software-Process Improvement in Web-Based Software Projects," *Cybern. Inf. Technol.*, vol. 24, no. 1, pp. 64–81, Mar. 2024, doi: 10.2478/cait-2024-0004.

[18] H. Hassan, M. A. Abdel-Fattah, and A. Ghoneim, "Risk Prediction Applied to Global Software Development using Machine Learning Methods," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 9, pp. 111–120, Dec. 2022, doi: 10.14569/IJACSA.2022.0130913.

[19] A. Kumar, M. Nadeem, and M. Shameem, "Metaheuristic-based cost-effective predictive modeling for DevOps project success," *Appl. Soft Comput.*, vol. 163, p. 111834, Sep. 2024, doi: 10.1016/J.ASOC.2024.111834.

[20] R. Hans and S. Marebane, "Are software projects evaluated using software teams' success criteria? A systematic literature review," *Procedia Comput. Sci.*, vol. 219, pp. 1599–1608, Jan. 2023, doi: 10.1016/j.procs.2023.01.452.

[21] R. K. B. N and Y. Suresh, "Effective ANN Model based on Neuro-Evolution Mechanism for Realistic Software Estimates in the Early Phase of Software Development," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 2, pp. 182–193, 2022, doi: 10.14569/IJACSA.2022.0130223.

[22] F. Zhang, N. A. S. Abdullah, and M. M. Rosli, "Critical Success Factors of Agile Software Projects: A Review," *Eng. Technol. Appl. Sci. Res.*, vol. 14, no. 5, pp. 16866–16873, Oct. 2024, doi: 10.48084/etasr.8358.

[23] R.-H. Pfeiffer and J. Aaen, "Tools for monitoring software quality in information systems development and maintenance: five key challenges and a design proposal," *Int. J. Inf. Syst. Proj. Manag.*, vol. 12, no. 1, pp. 19–40, Mar. 2024, doi: 10.12821/ijispm120102.

[24] T. Agrawal, G. S. Walia, and V. K. Anu, "Development of a Software Design Error Taxonomy: A Systematic Literature Review," *SN Comput. Sci.*, vol. 5, no. 5, p. 467, Apr. 2024, doi: 10.1007/s42979-024-02797-2.

[25] H. Ting, M. A. Memon, R. Thurasamy, and J.-H. Cheah, "Snowball Sampling: A Review and Guidelines for Survey Research," *Asian J. Bus. Res.*, vol. 15, no. 1, pp. 1–15, Mar. 2025, doi: 10.14707/ajbr.250186.

[26] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020, doi: 10.1145/3422622.

[27] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3313–3332, Apr. 2023, doi: 10.1109/TKDE.2021.3130191.

[28] D. Elreedy, A. F. Atiya, and F. Kamalov, "A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning," *Mach. Learn.*, vol. 113, no. 7, pp. 4903–4923, Jul. 2024, doi: 10.1007/s10994-022-06296-4.

[29] Z. Xu, D. Shen, Y. Kou, and T. Nie, "A Synthetic Minority Oversampling Technique Based on Gaussian Mixture Model Filtering for Imbalanced Data Classification," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 35, no. 3, pp. 3740–3753, Mar. 2024, doi: 10.1109/TNNLS.2022.3197156.

[30] M. Carvalho, A. J. Pinho, and S. Brás, "Resampling approaches to handle class imbalance: a review from a data perspective," *J. Big Data*, vol. 12, no. 1, p. 71, Mar. 2025, doi: 10.1186/s40537-025-01119-4.

[31] S. Feng, J. Keung, Y. Xiao, P. Zhang, X. Yu, and X. Cao, "Improving the undersampling technique by optimizing the termination condition for software defect prediction," *Expert Syst. Appl.*, vol. 235, p. 121084, Jan. 2024, doi: 10.1016/j.eswa.2023.121084.

[32] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J. Big Data*, vol. 11, no. 1, p. 87, Jun. 2024, doi: 10.1186/s40537-024-00943-4.

[33] D.-K. Kim and Y. K. Chung, "Addressing Class Imbalances in Software Defect Detection," *J. Comput. Inf. Syst.*, vol. 64, no. 2, pp. 219–231, Mar. 2024, doi: 10.1080/08874417.2023.2187483.

[34] Q. Leng, J. Guo, J. Tao, X. Meng, and C. Wang, "OBMI: oversampling borderline minority instances by a two-stage Tomek link-finding procedure for class imbalance problem," *Complex Intell. Syst.*, vol. 10, no. 4, pp. 4775–4792, Aug. 2024, doi: 10.1007/s40747-024-01399-y.

[35] F. Nhita, Adiwijaya, and I. Kurniawan, "Improvement of Imbalanced Data Handling: A Hybrid Sampling Approach by using Adaptive Synthetic Sampling and Tomek links," in *2023 Eighth International Conference on Informatics and Computing (ICIC)*, Dec. 2023, pp. 1–5, doi: 10.1109/ICIC60109.2023.10381929.

[36] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[37] M. Nadim, M. Hassan, A. K. Mandal, C. K. Roy, B. Roy, and K. A. Schneider, "Comparative analysis of quantum and classical support vector classifiers for software bug prediction: an exploratory study," *Quantum Mach. Intell.*, vol. 7, no. 1, p. 32, Jun. 2025, doi: 10.1007/s42484-025-00236-w.

[38] N. Amaya-Tejera, M. Gamarra, J. I. Vélez, and E. Zurek, "A distance-based kernel for classification via Support Vector Machines," *Front. Artif. Intell.*, vol. 7, p. 1287875, Feb. 2024, doi: 10.3389/frai.2024.1287875.

[39] K.-L. Du, B. Jiang, J. Lu, J. Hua, and M. N. S. Swamy, "Exploring Kernel Machines and Support Vector Machines: Principles, Techniques, and Future Directions," *Mathematics*, vol. 12, no. 24, p. 3935, Dec. 2024, doi: 10.3390/math12243935.

[40] M. Hofmann, M. F. P. Becker, C. Tetzlaff, and P. Mäder, "Concept transfer of synaptic diversity from biological to artificial neural networks," *Nat. Commun.*, vol. 16, no. 1, p. 5112, Jun. 2025, doi: 10.1038/s41467-025-60078-9.

[41] S. Chavlis and P. Poirazi, "Dendrites endow artificial neural networks with accurate, robust and parameter-efficient learning," *Nat. Commun.*, vol. 16, no. 1, p. 943, Jan. 2025, doi: 10.1038/s41467-025-56297-9.

[42] J. P. S. Rosa, D. J. D. Guerra, N. C. G. Horta, R. M. F. Martins, and N. C. C. Lourenço, "Overview of Artificial Neural Networks," in *SpringerBriefs in Applied Sciences and Technology*, Springer, Cham, 2020, pp. 21–44, doi: 10.1007/978-3-030-35743-6_3.

[43] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Efficient training of spiking neural networks with temporally-truncated local backpropagation through time," *Front. Neurosci.*, vol. 17, p. 1047008, Apr. 2023, doi: 10.3389/fnins.2023.1047008.

[44] Y. Song, B. Millidge, T. Salvatori, T. Lukasiewicz, Z. Xu, and R. Bogacz, "Inferring neural activity before plasticity as a foundation for learning beyond backpropagation," *Nat. Neurosci.*, vol. 27, no. 2, pp. 348–358, Feb. 2024, doi: 10.1038/s41593-023-01514-1.

[45] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton, "Loss of plasticity in deep continual learning," *Nature*, vol. 632, no. 8026, pp. 768–774, Aug. 2024, doi: 10.1038/s41586-024-07711-7.

[46] M. Ogunsanya, J. Isichei, and S. Desai, "Grid search hyperparameter tuning in additive manufacturing processes," *Manuf. Lett.*, vol. 35, pp. 1031–1042, Aug. 2023, doi: 10.1016/j.mfglet.2023.08.056.

[47] J. T. Hancock, T. M. Khoshgoftaar, and J. M. Johnson, "Evaluating classifier performance with highly imbalanced Big Data," *J. Big Data*, vol. 10, no. 1, p. 42, Apr. 2023, doi: 10.1186/s40537-023-00724-5.

[48] Y. Gebreyesus, D. Dalton, S. Nixon, D. De Chiara, and M. Chinnici, "Machine Learning for Data Center Optimizations: Feature Selection Using Shapley Additive exPlanation (SHAP)," *Futur. Internet*, vol. 15, no. 3, p. 88, Feb. 2023, doi: 10.3390/fi15030088.

[49] M. I. Hashfi and T. Raharjo, "Exploring the Challenges and Impacts of Artificial Intelligence Implementation in Project Management: A Systematic Literature Review," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 9, pp. 366–376, 2023, doi: 10.14569/IJACSA.2023.0140940.