Journal of Electronics, Electromedical Engineering, and Medical Informatics
Multidisciplinary: Rapid Review: Open Access Journal **Vol. 6, No. 2, April 2024, pp: 219-230; eISSN: 2656-8632**

RESEARCH ARTICLE **OPEN ACCESS**

Manuscript received February 10, 2024; revised February 26, 2024; accepted April 10, 2024; date of publication April 20, 2024
Digital Object Identifier (**DOI**): https://doi.org/10.35882/jeeemi.v6i2.409

How to cite: Putri Nabella , Rudy Herteno, Setyo Wahyu Saputro, Mohammad Reza Faisal, and Friska Abadi, Impact of a Synthetic Data Vault for Imbalanced Class in Cross-Project Defect Prediction, Journal of Electronics, Electromedical Engineering, and Medical Informatics, vol. 6, no. 2, pp. 219-230, April 2024.

Impact of a Synthetic Data Vault for Imbalanced Class in Cross-Project Defect Prediction

Putri Nabella, Rudy Herteno, Setyo Wahyu Saputro, Mohammad Reza Faisal, and Friska Abadi

Computer Science Department, Lambung Mangkurat University, Banjarbaru, South Kalimantan, Indonesia
Corresponding author: rudy.herteno@ulm.ac.id

ABSTRACT Software Defect Prediction (SDP) is crucial for ensuring software quality. However, class imbalance (CI) poses a significant challenge in predictive modeling. This study introduces a novel approach by employing the Synthetic Data Vault (SDV) to tackle CI within Cross-Project Defect Prediction (CPDP). Methodologically, the study addresses CI across multiple datasets (ReLink, MDP, and PROMISE) by leveraging SDV to augment minority classes. Classification utilizing Decision Tree (DT), Logistic Regression (LR), K-Nearest Neighbors (KNN), Naive Bayes (NB), and Random Forest (RF), also model performance is evaluated using AUC and t-Test. The results consistently show that SDV performs better than SMOTE and other techniques in various projects. This superiority is evident through statistically significant improvements. KNN dominance in average AUC results, with values 0.695, 0.704, and 0.750. On ReLink, KNN show 16.06% improvement over the imbalanced and 12.84% over SMOTE. Similarly, on MDP, KNN 20.71% improvement over the imbalanced and a 10.16% over SMOTE. Moreover, on PROMISE, KNN 13.55% improvement over the imbalanced and 7.01% over SMOTE. RF displays moderate performance, closely followed by LR and DT, while NB lags behind. Overall, SDV got an improvement of 10.10% from imbalanced, and 7.54% from SMOTE. The statistical significance of these findings is confirmed by t-Test, all below the 0.05 threshold. The practical implication of adopting SDV for defect detection and CI mitigation lies in its demonstrated effectiveness, particularly with KNN as the best classification algorithm, showcasing promising potential to enhance software quality by addressing CI and improving predictive modeling outcomes.

INDEX TERMS Class Imbalance, Cross Project Defect Prediction, Machine Learning, Software Defect Prediction, Synthetic Data Vault

I. INTRODUCTION

Modern software development has undergone a profound Software development has evolved significantly, marked by increasing complexities in coding and implementation processes, necessitating meticulous attention to ensure defect-free outcomes [1]. Despite substantial advancements in software engineering, challenges persist, particularly in the identification and rectification of software defects, which are vital for businesses to mitigate unforeseeable financial losses [2] [3]. To address these challenges, preemptive measures are essential, underscoring the importance of defect prediction methodologies in software engineering [4].

Software Defect Prediction (SDP) has emerged as a critical focus within software engineering, dedicated to systematically identifying flawed components within software projects [5]. These predictive models play a pivotal role in discerning segments of the software system with elevated probabilities of harboring defects, thereby facilitating efficient allocation of testing resources [6]. Among the various SDP methodologies, Within-Project Defect Predictions (WPDP) stand out, integrating models within the broader framework of SDP [7].

However, traditional SDP approaches encounter limitations, particularly in scenarios where historical data from locally accessible projects is lacking, rendering WPDP nonviable [8],[9] Consequently, researchers have shifted their attention towards emerging methodologies, prominently including Cross Project Defect Prediction (CPDP) [4].

CPDP represents a paradigm shift in SDP, leveraging historical data from previous software projects to tailor predictive models to specific project objectives [10], [11]. While promising, CPDP presents a common challenge: class imbalance (CI) [12], [13]. This imbalance significantly impacts the effectiveness of prediction models [14], [15], with fewer defective modules observed compared to non-defective ones [12].

To tackle the challenge of CI in software defect prediction, numerous studies have explored over-sampling techniques, among which the Synthetic Minority Over-sampling Technique (SMOTE) has emerged as a widely adopted approach [16]. In this study [17], compares two techniques for handling imbalanced data: oversampling with SMOTE and undersampling with Random Undersampling (RUS), using Gradient Boosting (GB) and RF as classification algorithms. Initially, on the original unbalanced dataset, the AUC values were 0.635 for GB and 0.644 for RF. However, after applying SMOTE, the AUC values increased to 0.649 for GB and 0.667 for RF. Conversely, by using RUS, AUC values of 0.644 for GB and 0.650 for RF were obtained.

These findings demonstrate that employing SMOTE in both classification algorithms resulted in a significant enhancement in model performance, while the use of RUS yielded insignificant changes. Therefore, SMOTE can be considered an effective method for addressing CI in the PROMISE dataset. However, it's important to note that this study only utilizes CK metrics, incorporating a subset of six attributes out of a total of 20 available attributes. This approach was adopted to focus on revealing the relationship between defects in object-oriented projects and CM metrics.

Other research has also investigated the efficacy of SMOTE as a method for addressing CI in CPDP. In this study [18], SMOTE combined with AdaBoost (AD-SMOTE) was utilized to mitigate misclassification, resulting in an AUC of 0.664. Another study [19] employed SMOTE in conjunction with Deep Canonical Correlation Analysis (S-DCCA) to calculate correlations and selectively utilize subsets characterized only by features with high correlation, leading to an AUC of 0.632.

SMOTE's popularity stems from its ability to enhance class balance without sacrificing valuable minority samples, showcasing its efficacy across various studies. While originally devised for classification tasks, SMOTE's adaptability has extended to addressing regression challenges as well [20]. Over the past decade, SMOTE has proven its utility across diverse domains, yielding significant contributions to various applications [21].

However, it is crucial to acknowledge that SMOTE is not without limitations. Despite its effectiveness, SMOTE may oversimplify the minority class, potentially resulting in instances that fail to capture the complexity of real-world data. Furthermore, the introduction of noise or bias into synthetic data poses challenges to the performance of defect prediction models and potentially leading to overfitting [22], [23].

In response, this study proposes the adoption of synthetic data from the Synthetic Data Vault (SDV) as an alternative approach. Synthetic data generated by data synthesizers have been shown to better represent the original data distribution, offering potential advantages over traditional methods [24], [25]. SDV features a number of approaches that each offer their unique advantages. GAN have proven to be powerful, generating high-quality, diverse synthetic data that closely resembles the original dataset. GAN improve model performance through data augmentation [26]. Conditional GAN (CT-GAN) enhance this innovation by generating data with certainty of discrete values, overcoming CI, and enriching the dataset with specialized information [26]. Copula GAN differentiates itself by utilizing copula functions in the generative process, offering greater interpretability and flexibility in capturing relationships between variables [27]. The Gaussian Copula is distinguished by its remarkable capacity to generate synthetic data effectively calibrate noise, attributed to its flexibility in describing dependencies between random variables [28]. Variational Autoencoders (VAE) capture the underlying data distribution using nonparametric approaches, providing a powerful alternative in tabular data generation [29]. Overall, the SDV approach offers a diverse set of tools with specific advantages for addressing challenges in synthetic data generation across various applications.

This study endeavors to assess the efficacy of SDV techniques in mitigating CI within CPDP. This involves utilizing five frequently used classification algorithms [30], including Decision Tree (DT), Logistic Regression (LR), K-Nearest Neighbors (KNN), Naive Bayes (NB), and Random Forest (RF), with the evaluation metric being the AUC. The research focuses on leveraging original samples from the minority class within CPDP datasets to create new synthetic instances. This approach directly addresses CI between the majority and minority classes, thereby enhancing the overall effectiveness and fairness of CPDP models. The contribution of this study is as follows:

a. Introduction of SDV as an alternative approach to traditional oversampling techniques like SMOTE for mitigating class imbalance in CPDP.
b. Identification of the optimal classification method among the five most commonly utilized algorithms in CPDP.

II. METHOD

The proposed methodology presents a meticulously structured approach to designing and implementing trials by harnessing the ReLink, NASA MDP, and PROMISE, within a computational framework, specifically leveraging Google Colab and Python programming. FIGURE 1 shows a flowchart that we used in this study. Within this methodology, one dataset is designated as the target project, while the others serve as source projects. To effectively tackle CI issues inherent in the datasets, synthetic data is generated using the SDV, a proficient tool in developing generative models within relational databases. SDV facilitates data synthesis by selectively sampling across database components post-model formulation, ensuring adherence to underlying structural constraints [31]. Moreover, the study incorporates the utilization of five classification algorithms, namely DT, LR, KNN, NB, and RF, to conduct a comprehensive assessment of defect prediction effectiveness across multiple projects. This evaluation employs the 10-fold cross-validation technique and utilizes metrics such as the AUC to measure the performance of each algorithm.

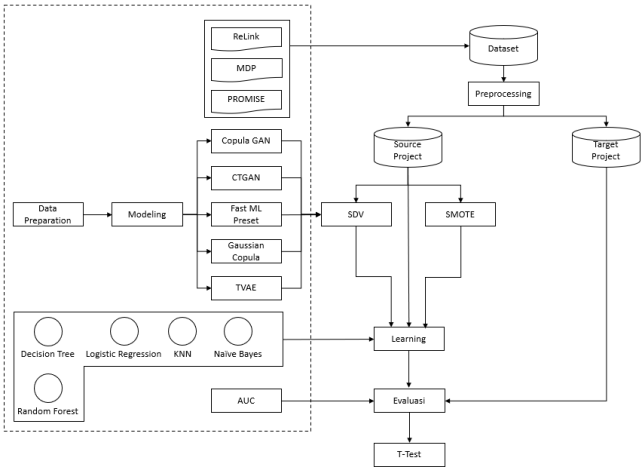

FIGURE 1. Research Flow

A. DATA COLLECTION

The study employs three datasets: ReLink, NASA MDP, and PROMISE datasets which is a publicly available dataset that widely applied in various domains [4], [19]. Within the ReLink dataset, three projects are featured: Apache, Safe, and Zxing. The NASA MDP dataset is focused on five specific projects out of twelve, namely CM1, MW1, PC1, PC3, and PC5, chosen due to their shared attributes, thereby eliminating the necessity for attribute selection for CPDP [18]. In the PROMISE dataset, 11 projects are integrated, including ant-1.7, camel-1.4, ivy-1.1, jedit-4.2, log4j-1.0, lucene-2.4, poi-3.0, synapse-1.2, velocity-1.6, xalan-2.4, and xerces-1.3.

TABLE 1
Summary of Dataset Statistics Employed in This Study

Source	Project	Instances	Non-Defects	Defects	Defective (%)
ReLink	apache	194	96	98	50.52%
	safe	56	34	22	39.29%
	zxing	399	281	118	29.57%
NASA MDP	CM1	327	285	42	12.84%
	MW1	253	226	27	10.67%
	PC1	705	644	61	8.65%
	PC3	1077	943	134	12.44%
	PC4	1287	1110	177	13.75%
PROMISE	ant-1.7	745	579	166	22.28%
	camel-1.4	872	727	145	16.63%
	ivy-1.1	241	225	16	6.64%
	jedit-4.2	367	319	48	13.08%
	log4j-1.0	135	101	34	25.19%
	lucene-2.4	340	137	203	59.71%
	poi-3.0	442	161	281	63.57%
	synapse-1.2	256	170	86	33.59%
	velocity-1.6	229	151	78	34.06%
	xalan-2.4	723	613	110	15.21%
	xerces-1.3	453	384	69	15.23%

This selection rationale is driven by the utilization of multi-version datasets, where a singular version is chosen as the distributions of two versions within a project may exhibit high similarity, potentially even identical [32]. Access to the ReLink and NASA MDP datasets is available through the following link: https://github.com/bharlow058/AEEEM-and-other-SDP-datasets [33], whereas the PROMISE dataset can be obtained from: https://github.com/feiwww/PROMISE-backup [34]. TABLE 1 is shows, which contains information and some general statistics about each of the datasets used.

B. PREPROCESSING

In the data preprocessing phase, attributes containing categorical values are converted to nominal values, specifically 0 and 1. For instance, within the ReLink dataset, the attribute 'isDefective' represents 'bug' as 1 and 'clean' as 0. Similarly, in the NASA MDP dataset, the 'Defective' attribute denotes 'Y' as 1 and 'N' as 0. Likewise, within the PROMISE dataset, the 'bug' attribute designates values other than 0 as 1.

C. OVERSAMPLING WITH SYNTHETIC DATA VAULT

Within the software defect dataset, most of the data exhibits a significantly larger proportion of non-defective samples compared to defective ones [31]. CI often results in bias within machine learning models towards the majority class [35]. Given the critical importance of accurately predicting the defective class, addressing CI becomes imperative prior to constructing CPDP models [36], [37].

Synthetic oversampling techniques, such as SMOTE, are employed to address the imbalance by generating artificial minority instances and rebalancing the dataset [38], [39]. However, concerns arise regarding the fidelity of replicating the original dataset with conventional oversampling techniques [22]. Synthetic data, intentionally manufactured to resemble real-world data, presents a promising strategy to overcome such issues, potentially offering higher quality than directly obtained or measured data [40]. Synthetic data retains a robust set of variables essential for supporting relevant multivariate analyses [41]. In a previous investigation utilizing fMRI images from an open-access database, the efficacy of CI mitigation through synthetic data shaping techniques was found to surpass that of SMOTE [42]. Therefore, the SDV, an end-to-end framework for modeling and generating synthetic sequential data tailored for tabular datasets [43], will be utilized to create minority data and address the CI problem in CPDP. Constructed with precision, these models aim to capture and estimate the correlations and distributions among different variables found within the original dataset [44].

During the initial phase of the SDV redesign process, the system was augmented with two additional libraries to optimize its functionality. Reversible Data Transforms (RDT) were employed by SDV to preprocess tables, which underwent iterative processing facilitated by Copulas for modeling purposes [45]. Presently, SDV offers various options for modeling single tables, including Copula GAN, CTGAN, Fast ML Preset, Gaussian Copula, and TVAE [46]. Furthermore, SDV consists of interconnected modules, each serving distinct functionalities. Here are some APIs for the modules within SDV.

1. META FILE

The primary phase of the operation encompasses the acquisition of the dataset. Following this, SDV mandates access to metadata concerning the dataset, encompassing attributes such as column data types. This requisite

information is encapsulated within a JSON structure denoted as the meta file, serving as a foundational element essential for the execution of SDV procedures on the dataset in question [45].

2. DATA LOADER

The CSVDataLoader class is initialized with a meta file supplied as input parameter. Upon instantiation, this file is stored internally as an attribute named 'meta'. Subsequent to this initialization, the DataNavigator class is instantiated utilizing the details provided within the meta file to identify and load the corresponding CSV files as pandas DataFrames. A dictionary structure is then created, associating each table's name with an instance of the Table class. Each Table instance encapsulates both the metadata and DataFrame specific to its corresponding table. This amalgamation of information serves as the foundation for the instantiation of a DataNavigator instance. The DataNavigator, thus created, encapsulates the necessary information and functionalities required for navigating through the dataset effectively. Finally, this instantiated DataNavigator is returned by the 'loadData' method for further utilization [45].

3. DATA NAVIGATOR

DataNavigator serves as a crucial component for both data navigation and modeling, housing pertinent information regarding the dataset's structure. Its primary functionalities encompass accessing child or parent tables, retrieving data from tables, obtaining table metadata, applying data transformations, and discerning relationships between tables. A key operation performed by DataNavigator is the get_relationships method, wherein it meticulously traces the dataset's structure, storing essential details regarding inter-table relationships, including parent-child associations and primary-foreign key mappings. Such insights are fundamental for the subsequent data modeling endeavors [45].

4. MODELER

The SDV modeling technique utilizes Conditional Parameter Aggregation (CPA) and Recursive Conditional Parameter Aggregation (RCPA) to characterize relationships among tables in a dataset. CPA consolidates conditional parameters within individual tables, while RCPA extends these parameters recursively to all descendant tables, starting from leaf nodes and progressing towards the root node. The modelDatabase function identifies dataset roots, initiates RCPA, and stores the resultant models in the Modeler attribute, enabling efficient modeling of intricate relational structures. The Modeler class possesses the capacity to store numerous models and is adaptable to various types of models utilized, such as Copula or others [45].

Let D represent a database comprising numerous tables, denoted as T. The interconnections among these tables are established, thus $C(T)$ signifies the set of children of T, while $P(T)$ denotes the set of parents of T.

Since the CPA method returns the extended table, line 4 of the algorithm stores the extended tables as T. Subsequently, line 5 preprocesses T to convert the values into numerical data. The base case of this algorithm is for leaf tables, where $C(T)=\emptyset$. During the creation of the overall model by SDV, it applies RCPA and uses the result to calculate the database model. The SDV's modeling algorithm invokes the RCPA method on all tables without parents. Due to the recursive nature of RCPA, this ensures that all tables in the database ultimately undergo the CPA method [47].

TABLE 2
Recursive Application of CPA to add Derived Columns to T

No.	Algorithm
1.	function RCPA(T)
2.	for all $C \in C(T)$ do
3.	RCPA(C)
4.	$T \leftarrow$ CPA(T)
5.	$T \leftarrow$ PreProcess(T)

5. SAMPLER

Following the completion of modeling, the ultimate phase in data synthesis entails the sampling of new data. This task is executed by the Sampler class, which is initialized with an instance of the Modeler class. Utilizing the insights gleaned from the Modeler, the Sampler orchestrates the generation of synthetic data. Its core objective is to offer a spectrum of sampling methods catering to diverse user requisites. Thus, users merely need to furnish a Modeler instance and a DataNavigator instance to the Sampler, facilitating the invocation of relevant sampling methods and subsequent data sampling. The Sampler maintains all sampled data within a dictionary structure, wherein each table's name is correlated with the respective sampled rows [45].

From the user's standpoint, SDV entails discrete stages, namely data preparation, modeling, sampling, and evaluation.

1. DATA PREPARATION

In the data preparation phase, the initial step involves loading the data as a pandas DataFrame object. Subsequently, the data undergoes conversion into metadata using the SingleTableMetadata approach, which meticulously describes each table. This metadata encompasses details such as the data type for each column, the primary key, and other pertinent identifiers [46].

2. MODELING

During the modeling phase, synthetic data is generated based on the prepared metadata. This process involves employing a synthesizer that utilizes the original data as a foundation. Throughout this stage, the synthesizer discerns the underlying patterns within the original dataset. Various synthesizers are utilized in this modeling phase, including Copula GAN, CTGAN, Fast ML Preset, Gaussian Copula, and TVAE [46].

Due to the limited elucidation provided for each modeling aspect within the documentation or paper concerning SDV, the following is a little explanation that can be summarized from various sources.

a) Copula GAN: This hybrid synthesizer integrates classical statistics with GAN-based deep learning techniques,

offering a comprehensive approach to data modeling [46]. In the realm of GAN, there are two main components: the discriminator (D) and the generator (G). The discriminator aims to distinguish real data from fake, while the generator tries to produce data that looks real. The equation represents a game where the generator minimizes its likelihood of being detected by the discriminator, while the discriminator maximizes its ability to differentiate real from fake. At equilibrium, the generator creates data indistinguishable from real, and the discriminator can't reliably tell real from fake, achieving a balance where the generated data distribution matches the real data distribution [48].

$$min_G min_D V(D,G) = E_{x\sim p\ data(x)}\left[log\left(D(x)\right)\right] + E_{z\sim pz(z)}\left[log\left(1-D\left(G(z)\right)\right)\right] \quad (1)$$

b) Conditional Tabular GAN (CTGAN): Employs a GAN-based method to model the distribution of tabular data and sample rows from it [49]. CTGAN assesses the dissimilarity between the acquired conditional distribution and the real data's conditional distribution. The conditional vector is denoted as $D_t^* = k^*$, where $k = 1, \dots, |D_t|$, and $d_{i,j}$, a discrete variable in D_t, is initially represented as a one-hot vector $d_{i,j}$ with dimension $|D_i|$. During training, the conditional generator is permitted to generate any set of one-hot discrete vectors. Additionally, to penalize generator losses, cross-entropy between D_i^* and m_i^* is incorporated. A suggested procedure for producing $D_1^* = m_1^*$ is proposed, enabling the generator to replicate m_1^* accurately into D_1^* [50].

c) Fast ML Preset: This synthesizer leverages machine learning (ML) techniques to efficiently model the data [46]. It introduces an innovative approach known as indel-coding methodology, where each indel in the input sequence is represented as either present ('1') or absent ('0'). This binary representation is then utilized in a machine learning-based algorithm to estimate the likelihood of gap characters in ancestral sequences. Initially, Fast ML employs a simple coding scheme to convert all indels into binary format, indicating their presence or absence. The resulting binary data matrix serves as input to an ML-based ancestral indel reconstruction algorithm [51]. However, it's important to note that there isn't a single equation that encapsulates the entirety of a machine learning model [52].

d) Gaussian Copula: Copula models provide an efficient approach to capturing both inter-variable dependencies and individual behaviors. They prove especially valuable for synthesizing datasets from complex, smaller real datasets [53]. Each column in the table is indexed from 0, 1, ..., *n*, with each column having its Cumulative Distribution Function (CDF) denoted as $F_0\ to\ F_n$ respectively. Subsequently, each row of the table is treated as a vector $X = (x_0, x_1, \dots, x_n)$. The Gaussian Copula is then applied to transform the row vector. Mathematically, this transformation can be expressed as:

$$Y = \left[\varphi^{-1}\left(F_0(x_0)\right), \varphi^{-1}\left(F_1(x_1)\right), \dots, \varphi^{-1}\left(F_n(x_n)\right)\right] \quad (2)$$

In this equation, $\varphi^{-1}\left(F_n(x_n)\right)$ represents the inverse cumulative distribution function of the Gaussian distribution applied to the original distribution [54].

e) Tabular Variational Autoencoder (TVAE): Implementing the Variational Autoencoder (VAE) approach, this synthesizer consists of an encoder for compressing input data into a low-dimensional latent space and a decoder for reconstructing output data based on the learned representation from the encoder [24].

$$\log p(x) \geq E_{z\sim q(z|x)}[\textstyle\sum_t \log p(x_t|z_t) + \log p(z_t|z_{t-1}) - \log q\left(z_t|z_{t-1}, \emptyset_t(x)\right) \quad (3)$$

This equation delineates a constraint derived from VAE methodology, which elucidates the interplay between latent variable *z* and observed variable *x*. Within the VAE framework, *z* adheres to a predetermined prior distribution *p*(*z*), typically a standard normal distribution. The choice of likelihood distribution *p*(*x*|*z*) varies depending on the task, being either Normal or Bernoulli. The fundamental objective is to derive the posterior distribution of the latent variable, denoted as *p*(*z*|*x*). However, the true posterior is challenging to compute for continuous latent spaces like *z* [55], [56].

3. SAMPLING

Following the conclusion of the modeling process, the synthesizer possesses the capability to produce synthetic data. In this context, the generated synthetic data specifically targets the minority class, addressing the issue of data imbalance [46].

4. DIAGNOSTIC

The Diagnostic Report performs fundamental checks on data format and validity. Specifically, it applies the TableStructure metric to each table in the dataset to ensure consistency. This metric compares the column names between the synthetic and real data. By identifying all column names in both datasets, it calculates a score based on the overlap between the columns.

$$score = \frac{r \cap s}{r \cup s} \quad (4)$$

A score of 100% indicates perfect alignment, meaning the synthetic data shares identical column names with the real data [46].

C. Synthetic Minority Oversampling Technique

The Synthetic Minority Over-Sampling Technique (SMOTE) is employed as an oversampling method to mitigate CI in datasets [57]. This technique leverages original samples from the minority class to generate new synthetic instances. Unlike traditional data space approaches, SMOTE operates in feature space for synthesizing instances [26]. In this study, the assessment outcomes derived from the SMOTE will be juxtaposed with those obtained from SDV and unbalanced datasets. This comparative strategy enables a comprehensive evaluation of SDV's efficacy in addressing dataset imbalance by contrasting it with alternative methodologies such as SMOTE. The equation of SMOTE, represented as follows:

$$x_{new} = x + rand(0,1) \times (y[i] - x) \quad (5)$$

This equation generates a new synthetic sample, denoted as $xnew$, by linearly blending between an original sample, *x*, and another sample, $y[i]$. The degree of blending is determined by a random factor, rand (0,1), which adjusts the difference between *x* and $y[i]$. This random factor introduces variability into the process of generating the synthetic sample [58].

D. CLASSIFICATION ALGORITHM

In recent years, researchers have increasingly focused on the classification stage, which represents the final phase of prediction models. This stage has been the subject of intense scrutiny aimed at enhancing the efficiency of CPDP models and improving classifier performance. As such, this study adopts the five most prevalent classification methods utilized in CPDP [30].

1. K-NEAREST NEIGHBORS

The K-Nearest Neighbors (KNN) algorithm is highly regarded for its versatility, as it refrains from imposing stringent assumptions regarding the underlying data distribution. KNN achieves remarkable classification accuracy by leveraging the proximity of data points and making decisions based on the majority class among the nearest neighbors, a methodology that frequently yields favorable outcomes across diverse datasets [59]. Upon deployment, KNN classifies new data points by scrutinizing the predominant class among their nearest neighbors within a predefined neighborhood size, denoted as the *K* value. This approach ensures both adaptability and efficacy in classification tasks. The Euclidean distance stands as the fundamental formulation utilized in the KNN, represented as:

$$dis(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2} \quad (6)$$

In this equation, $\boldsymbol{x_i}$ and $\boldsymbol{y_i}$ represent elements of the feature vectors *x* and *y* from sets *A* and *B*, respectively. The variable *n* denotes the dimensionality of the feature space, encompassing the number of features considered in the comparison [60].

2. NAIVE BAYES

Naive Bayes (NB) is a probabilistic machine learning technique employed for classification tasks [18]. It determines the highest probability value and assigns the test data to the most suitable category based on this calculation [61]. The classifier derives its name from the "naive" assumption that all features are independent of each other given a class label. While this assumption is often violated in real-world contexts, Naive Bayes classifiers can still yield satisfactory outcomes in numerous scenarios [62]. This simplicity and robust classification performance contribute to NB being widely adopted as a classification algorithm [63]. The equation of NB, represented as:

$$P(Y|X) = \frac{P(X|H)\,P(H)}{P(X)} \quad (7)$$

In this equation, *X* represents data with an unknown class, while *H* stands for the hypothesis that *X* belongs to a specific class. The term $P(Y|X)$ denotes the probability of hypothesis HH given the data *X*, known as the posterior probability. $P(H)$ represents the prior probability of hypothesis *H*, while $P(X|H)$ signifies the probability of observing data *X* given hypothesis *H*. Finally, $P(X)$ represents the overall probability of observing data *X* [64].

3. DECISION TREE

Decision Tree (DT) classifier stands out as a computational model revered for its multistage decision-making process, adept at handling both numerical and nominal data types. Its hierarchical structure comprises decision nodes and leaf nodes, facilitating the creation of efficient decision rules [65]. In essence, there exist two primary types of nodes within this structure: decision nodes and leaf nodes. Decision nodes play a crucial role in establishing decision rules by segmenting the data into different sections based on specific criteria. Conversely, leaf nodes represent the ultimate outcomes or conclusions derived from these decision rules and do not lead to further subdivisions or branches. Thus, while decision nodes steer the tree's structure, leaf nodes furnish the final decisions or predictions [66]. The entropy equation serves as a pivotal tool in DT analysis, particularly when calculating the impurity at a node, represented as:

$$Entropy\,(P) = -\sum p(i).\log_2\big(p(i)\big) \quad (8)$$

In this equation, $Entropy\,(P)$ represents the entropy of the dataset *P*, where $p(i)$ denotes the probability that an instance in dataset *P* belongs to class *i* [67].

4. RANDOM FOREST

Random Forest (RF) algorithm is a supervised classification technique utilized in creating a forest through a randomized procedure [68]. Initially, it identifies the root node employing the most effective splitting technique. This process is then replicated for the child nodes, utilizing the same optimal splitting method. The iterative nature of this cycle results in the construction of a complete tree, with the desired outcome at the leaf nodes. Subsequently, the algorithm repeats these steps to generate multiple trees, each with its random selection of features and splits [69]. RF execution involves a structured process. It begins with bootstrapping, where samples of size *n* are drawn randomly with replacement from dataset clusters. DT are then grown without pruning until reaching maximum size, using these bootstrap samples. At each node, a split is chosen by randomly selecting a subset of m predictors from the total *p* predictors (where $m << p$), known as the random feature selection phase. This process repeats *k* times, creating a forest of *k* trees [70].

5. LOGISTIC REGRESSION

Logistic Regression (LR) is a versatile predictive modeling technique extensively utilized to assess the relationship between dependent (target) variables, typically categorical data with nominal or ordinal scales, and independent (predictor) variables [71]. It stands out as a prominent statistical method employed in constructing predictive models, particularly for estimating the probability of an event [72]. LR is specifically tailored for making categorical predictions, handling binary or multinomial outcomes by modeling the probability of belonging to a specific category. It achieves this by employing a logistic function to transform the output of a LR model into probabilities, ensuring

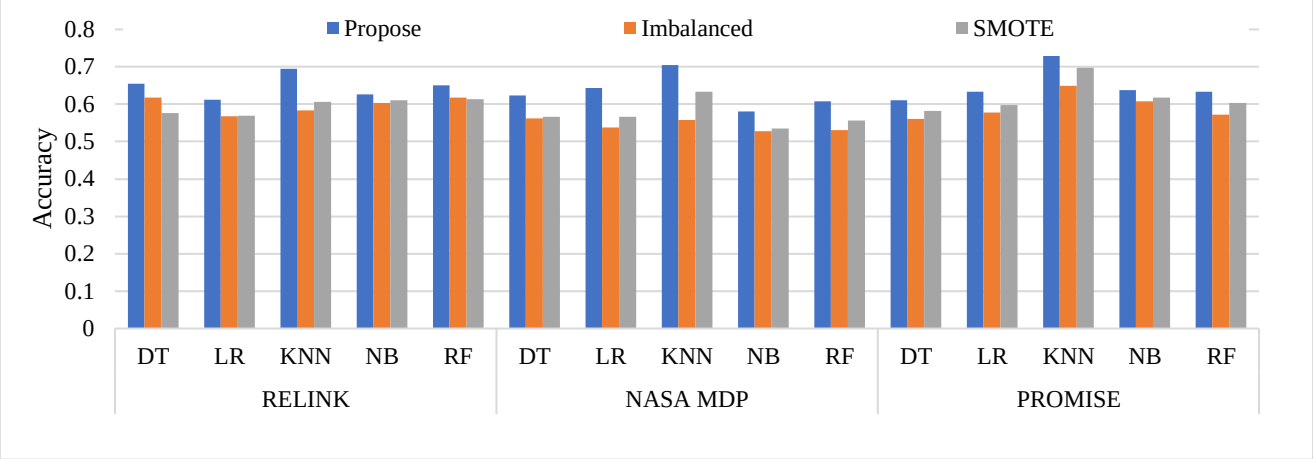

FIGURE 2. Performance Comparison of the Proposed Method and Others

predictions fall within the range of 0 to 1 [73]. The equation of LR, represented as:

$$\log\left(\frac{Pbj}{1-Pbj}\right) + \beta_1 X_{1j} + \beta_2 X_{2j} + \cdots + \beta_n X_{nj} \quad (9)$$

In this equation, β_n represents the slope of independent attributes, and X_{nj}signifies an independent attribute in record j. The variable nn denotes the number of independent attributes, and j signifies the number of records in the dataset [70].

E. PERFORMANCE EVALUATION

Model performance evaluation is a crucial aspect of this study [59], primarily focusing on the AUC, which holds significant importance in evaluating the effectiveness of data categorization [74]. AUC provides a quantitative measure of the model's ability to distinguish between different classes, with values ranging from 0 to 1. A value of 1 indicates perfect separation between classes, while a value of 0.5 suggests random categorization [75]. Analyzing AUC values provides valuable insights into the discriminatory power of the model and its performance in accurately classifying instances. The equation of AUC, represented as:

$$AUC = \int_0^1 TPR\left(FPR^{-1}(t)\right)dt \quad (10)$$

In this equation, the AUC represents the integral of the True Positive Rate (TPR) plotted against the False Positive Rate (FPR), where t signifies various classification thresholds [76].

F. T-TEST

This test focused on the difference in AUC values to evaluate the average performance of the model and determine its significance [78]. Setting the alpha (α) value at 0.05, a common significance level, provides a confident basis for rejecting the null hypothesis with 95% certainty in statistical testing. A t-Test result below this threshold indicates strong statistical significance. While alpha levels can be adjusted, 0.05 is generally accepted as a practical compromise [79]. The equation of T-Test, represented as:

$$T = \frac{y_1 - y_2}{\sqrt{s_p^2\left(\frac{1}{n_1} + \frac{1}{n_{q2}}\right)}} \quad (11)$$

In this equation, $\boldsymbol{y_1}$ and $\boldsymbol{y_2}$ represent the mean values from groups 1 and 2, respectively. $\boldsymbol{s_p}$ stands for an estimate of the pooled standard deviation of the measurements. Additionally, $\boldsymbol{n_1}$ and $\boldsymbol{n_2}$ denote the number of observations for each group [58].

III. RESULT

This study embarks on a comprehensive assessment aimed at gauging the efficacy of synthetic data generated through SDV in tackling the persistent challenge of CI within the domain of CPDP. Through a meticulous and comparative investigation, we delve into the performance analysis of SDV-generated synthetic datasets in contrast with those fashioned by the widely adopted SMOTE technique.

Drawing upon a diverse array of data gleaned from 19 projects, our research endeavors to unveil the nuanced intricacies of synthetic data's efficacy in addressing CI challenges within CPDP. TABLE 3 – 5 show the empirical evidence meticulously gathered and analyzed throughout our study firmly establishes the superiority of SDV-generated synthetic data over both the original imbalanced dataset and those artificially balanced by the SMOTE technique.

Moreover, the study displays the results of the evaluation using five evaluation algorithms, enhancing the robustness of the findings. These evaluation algorithms likely encompass a range of metrics such AUC, among others. The use of multiple evaluation algorithms helps provide a comprehensive understanding of the performance of synthetic data generated through SDV and SMOTE across various dimensions.

These robust findings not only underscore the substantial potential of SDV synthetic data in rectifying CI issues but also shed light on its transformative impact on predictive modeling paradigms within CPDP. By offering novel insights and statistically superior outcomes compared to conventional methods like SMOTE, our study heralds a new era in the realm of CI strategies within CPDP.

During the initial validation stage, one project was designated as the testing dataset, while the others projects were utilized as the training datasets. Subsequently, in the

subsequent validation stages, the dataset from the next project was chosen as the test data, with datasets from the remaining projects employed as the training data. This iterative process continued until all projects had been utilized as testing datasets.

TABLE 3
AUC Performance on ReLink Dataset

Classification	Result	Propose	Imbalanced	SMOTE
Decision Tree	Average	**0.655**	0.618	0.576
	Improvement	-	5.53%	12.06%
Logistic Regression	Average	**0.612**	0.568	0.569
	Improvement	-	7.15%	7.00%
KNN	Average	**0.695**	0.584	0.606
	Improvement	-	16.05%	12.84%
Naïve Bayes	Average	**0.626**	0.604	0.611
	Improvement	-	3.61%	2.44%
Random Forest	Average	**0.651**	0.617	0.613
	Improvement	-	5.28%	5.82%

TABLE 4
AUC Performance on NASA MDP Dataset

Classification	Result	Propose	Imbalanced	SMOTE
Decision Tree	Average	**0.623**	0.562	0.566
	Improvement	-	9.89%	9.22%
Logistic Regression	Average	**0.644**	0.538	0.566
	Improvement	-	16.56%	12.18%
KNN	Average	**0.704**	0.558	0.633
	Improvement	-	20.71%	10.16%
Naïve Bayes	Average	**0.581**	0.528	0.535
	Improvement	-	9.11%	7.97%
Random Forest	Average	**0.607**	0.530	0.556
	Improvement	-	12.78%	8.50%

TABLE 5
AUC Performance on PROMISE Dataset

Classification	Result	Propose	Imbalanced	SMOTE
Decision Tree	Average	**0.610**	0.561	0.582
	Improvement	-	7.96%	4.49%
Logistic Regression	Average	**0.633**	0.578	0.598
	Improvement	-	8.69%	5.63%
KNN	Average	**0.750**	0.649	0.698
	Improvement	-	13.55%	7.01%
Naïve Bayes	Average	**0.638**	0.607	0.618
	Improvement	-	4.80%	3.01%
Random Forest	Average	**0.634**	0.572	0.604
	Improvement	-	9.76%	4.79%

SDV exhibits superior performance compared to SMOTE in handling imbalanced datasets, consistently outperforming both original imbalanced datasets and those balanced using SMOTE across various classification algorithms and datasets. However, SDV techniques demand substantial computational resources and expertise to generate high-quality synthetic data accurately representing the underlying distribution. In contrast, SMOTE is simpler and less resource-intensive but may produce synthetic samples sensitive to noise and outliers, potentially leading to overfitting or decreased model performance. Imbalanced datasets reflect real-world scenarios, yet their inherent bias can cause classifiers to favor the majority class, resulting in suboptimal predictive performance for minority classes. Therefore, while imbalanced datasets remain representative of practical applications, employing SDV or SMOTE techniques requires careful consideration of computational requirements and potential impacts on model generalization.

FIGURE 2 depicts a graph comparing the average results of the proposed method with those of other methods across different datasets. Each dataset is represented as a cluster of bars along the x-axis, with each bar providing a visual representation of the mean outcomes of the method examined within the corresponding dataset.

Following the attainment of average AUC results for each project, we conducted a significance test utilizing the t-Test to ascertain whether our proposed method exhibited statistical significance compared to others.

Table 6
The Result of the T-Test

Dataset	Method Comparison	T-Test ($\propto = 0.05$)	Significance
ReLink	DT – Imbalance	0.013048389	Significance
	DT – SMOTE	0.021842223	Significance
	LR – Imbalance	0.027067072	Significance
	LR – SMOTE	0.019639803	Significance
	KNN – Imbalance	0.038361514	Significance
	KNN – SMOTE	0.001148641	Significance
	NB – Imbalance	0.034557446	Significance
	NB – SMOTE	0.036961263	Significance
	RF – Imbalance	0.021797694	Significance
	RF – SMOTE	0.038530614	Significance
MDP	DT – Imbalance	0.000004295	Significance
	DT – SMOTE	0.000000513	Significance
	LR – Imbalance	0.000000277	Significance
	LR – SMOTE	0.000000037	Significance
	KNN – Imbalance	0.000284700	Significance
	KNN – SMOTE	0.007529326	Significance
	NB – Imbalance	0.000252743	Significance
	NB – SMOTE	0.000180900	Significance
	RF – Imbalance	0.000000163	Significance
	RF – SMOTE	0.000001129	Significance
PROMISE	DT – Imbalance	0.000262698	Significance
	DT – SMOTE	0.004458561	Significance
	LR – Imbalance	0.000048935	Significance
	LR – SMOTE	0.000036173	Significance
	KNN – Imbalance	0.000000186	Significance
	KNN – SMOTE	0.000000755	Significance
	NB – Imbalance	0.001251084	Significance
	NB – SMOTE	0.009402305	Significance
	RF – Imbalance	0.000078728	Significance
	RF – SMOTE	0.000130464	Significance

TABLE 6 presents the test results utilizing the t-Test to evaluate the significance of differences between the various test methods. If the obtained t-Test value is below the predetermined alpha threshold of 0.050, it indicates a statistically significant performance improvement. Conversely, if the t-Test value exceeds the alpha threshold, the observed performance improvement is considered statistically non-significant. In this study, the t-Test results suggest that there are significant differences between the performance of different methods, particularly in the context of addressing CI in datasets. For instance, comparing the proposed method against SMOTE and imbalance approaches with five classifiers, across various datasets, the p-values are consistently low. This indicates that the proposed method yields statistically significant improvements over others.

Moreover, the significance levels vary across different datasets and algorithms. For instance, in the MDP dataset, the p-values for all method comparisons are extremely low, suggesting highly significant differences. On the other hand,

in the PROMISE and RELINK dataset, while most comparisons still yield low p-values, indicating significance, there are instances where the significance levels are slightly higher. This variability underscores the importance of considering dataset-specific characteristics when evaluating the effectiveness of different methods.

Overall, the t-Test results, coupled with the alpha value, provide strong evidence to support the superiority of the proposed method in addressing CI compared to traditional approaches across multiple datasets.

IV. DISCUSSION

The study substantiates the remarkable efficacy of our proposed methodology in effectively addressing the intricate challenge of CI within the realm of CPDP. Through the judicious utilization of Synthetic Data Generation via SDV to rectify CI, our approach distinctly demonstrates superior performance when juxtaposed against five utilized classifiers. Notably, it surpasses conventional methodologies such as SMOTE and imbalance data scenarios, thereby underscoring its robustness and effectiveness.

A meticulous and comprehensive comparative analysis reveals the consistent outperformance of our approach over SMOTE across a myriad of datasets, as meticulously delineated in TABLE 3 – 5. Furthermore, leveraging the rigorous statistical tool of t-Test, as outlined in TABLE 6, we establish statistical significance, thereby unequivocally showcasing the superiority of our SDV approach over SMOTE across all evaluated datasets.

The synthetic data generated through SDV consistently exhibits superior performance across diverse evaluation metrics, with particular prominence observed in the realm of the AUC metric. Notably, the steadfast superiority of the KNN algorithm underscores the pivotal role of algorithmic selection in effectively mitigating the challenges associated with CPDP.

While antecedent studies have explored an array of techniques to grapple with CI, our investigation empirically substantiates that SDV presents a more efficacious resolution within this domain. This assertion gains further credence through the comparative analysis presented in TABLE 7, which unequivocally underscores the supremacy of our method over alternative techniques.

TABLE 7
Comparison of AUC Values of the Proposed Method with Other Studies

Dataset	Method	AUC
ReLink	**Propose Method with KNN**	**0.695**
	PF-SMOTE [13]	0.633
	S-DCCA [19]	0.632
NASA MDP	**Propose Method with KNN**	**0.704**
	AD-SMOTE [18]	0.664
PROMISE	**Propose Method with KNN**	**0.750**
	GSMOTE-NFM [80]	0.715
	SMOTE-GB [17]	0.649
	SMOTE-RF [17]	0.667

However, we conscientiously acknowledge the inherent constraints in our study. The reliance on specific datasets inevitably curtails the generalizability of our findings, while the focus on the AUC metric and select classification algorithms may inadvertently overshadow other salient facets of model performance assessment. Furthermore, the computational intricacies attendant to the SDV technique pose pragmatic challenges in real-world deployment, warranting further exploration and refinement. The study also does not rule out the possibility that further exploration may lead to overfitting when employing data generated by SDV. This consideration underscores the need for caution in extending the application of SDV-generated data beyond the scope of this study.

Nevertheless, our study yields pivotal findings that carry profound implications for the field of CPDP. Academically, we offer invaluable insights into enhancing the reliability and precision of defect prediction models by showcasing the efficacy of synthetic data generation through SDV. The implementation of SDV techniques stands poised to usher in more precise and reliable forecasts of software defects, thereby bolstering the quality and dependability of software products.

Moreover, the seamless integration of SDV holds promise for streamlining the development lifecycle, curtailing maintenance expenditures, and ultimately elevating customer satisfaction levels. Additionally, synthetic data serves as an indispensable tool for safeguarding sensitive personal information that cannot be divulged, thereby ensuring compliance with stringent data privacy regulations.

Furthermore, we ardently advocate for the exploration of alternative methodologies within SDP frameworks to mitigate CI, surpassing traditional techniques such as SMOTE. In essence, our research underscores the transformative potential of our proposed methodology in reshaping the landscape of CPDP. By effectively mitigating CI through SDV, our approach engenders robust predictive models that surpass existing methodologies, thereby offering a compelling roadmap for future research endeavors aimed at augmenting the efficacy and applicability of defect prediction models.

V. CONCLUSION

This research aims to tackle a common challenge in CPDP, namely CI, by leveraging synthetic data generated by SDV. SDV works to balance the data by creating minority classes, thereby achieving a balanced distribution of instances across classes. Using five different classification algorithms and the AUC metric, this study thoroughly investigated the performance of synthetic data generated by SDV compared to traditional methods like SMOTE across 19 selected projects.

Our study unequivocally demonstrates the superiority of synthetic data generated by SDV in addressing CI within CPDP. Across all analyzed datasets, SDV consistently outperformed both the original unbalanced datasets and those balanced using SMOTE, as evidenced by higher AUC scores. Specifically, various methods, including Relink, Nasa MDP, and PROMISE, showed sequential improvements. KNN achieved AUC scores of 0.695, 0.704, and 0.750 for the respective datasets, while DT attained scores of 0.655, 0.623, and 0.610. LR yielded AUC scores of 0.612, 0.644, and 0.633, whereas NB obtained scores of

0.626, 0.581, and 0.638. RF received AUC scores of 0.651, 0.607, and 0.634. These results confirm that the utilization of synthetic data from SDV significantly enhances model performance in addressing CI in CPDP.

To address the limitations identified in this study, future research could explore the application of SDV techniques across a broader range of datasets and project contexts to enhance generalizability. Additionally, investigating the performance of SDV in conjunction with other machine learning techniques and performance measures could provide a more comprehensive understanding of its capabilities. Moreover, efforts to mitigate the computational overhead associated with SDV implementation could facilitate its adoption in real-world CPDP scenarios, as well as further exploration for addressing the overfitting problem.

REFERENCES

[1] S. Amasaki, "Cross-version defect prediction: use historical data, cross-project data, or both?," *Empir Softw Eng*, vol. 25, no. 2, pp. 1573–1595, Mar. 2020, doi: 10.1007/s10664-019-09777-8.

[2] S. Noreen, R. Bin Faiz, S. Alyahya, and M. Maddeh, "Performance Evaluation of Convolutional Neural Network for Multi-Class in Cross Project Defect Prediction," *Applied Sciences (Switzerland)*, vol. 12, no. 23, Dec. 2022, doi: 10.3390/app122312269.

[3] S. Tang, S. Huang, C. Zheng, E. Liu, C. Zong, and Y. Ding, "A Novel Cross-Project Software Defect Prediction Algorithm Based on Transfer Learning," *Tsinghua Sci Technol*, vol. 27, no. 1, pp. 41–57, 2022, doi: 10.26599/TST.2020.9010040.

[4] S. Pal and A. Sillitti, "Cross-Project Defect Prediction: A Literature Review," *IEEE Access*, vol. 10. Institute of Electrical and Electronics Engineers Inc., pp. 118697–118717, 2022. doi: 10.1109/ACCESS.2022.3221184.

[5] Y. Zhao, Y. Zhu, Q. Yu, and X. Chen, "Cross-project defect prediction method based on manifold feature transformation," *Future Internet*, vol. 13, no. 8, Aug. 2021, doi: 10.3390/fi13080216.

[6] Y. Z. Bala, P. A. Samat, K. Y. Sharif, and N. Manshor, "Improving Cross-Project Software Defect Prediction Method Through Transformation and Feature Selection Approach," *IEEE Access*, vol. 11, pp. 2318–2326, 2023, doi: 10.1109/ACCESS.2022.3231456.

[7] U. S. Bhutamapuram and R. Sadam, "With-in-project defect prediction using bootstrap aggregation based diverse ensemble learning technique," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10. King Saud bin Abdulaziz University, pp. 8675–8691, Nov. 01, 2022. doi: 10.1016/j.jksuci.2021.09.010.

[8] Z. Sun, J. Li, H. Sun, and L. He, "CFPS: Collaborative filtering based source projects selection for cross-project defect prediction," *Appl Soft Comput*, vol. 99, Feb. 2021, doi: 10.1016/j.asoc.2020.106940.

[9] R. Vashisht and S. A. M. Rizvi, "Addressing Noise and Class Imbalance Problems in Heterogeneous Cross-Project Defect Prediction: An Empirical Study," *International Journal of e-Collaboration*, vol. 19, no. 1, 2023, doi: 10.4018/IJeC.315777.

[10] M. Nevendra and P. Singh, "Cross-Project Defect Prediction with Metrics Selection and Balancing Approach," *Applied Computer Systems*, vol. 27, no. 2, pp. 137–148, Dec. 2022, doi: 10.2478/acss-2022-0015.

[11] Y. Zhao, Y. Zhu, Q. Yu, and X. Chen, "Cross-Project Defect Prediction Considering Multiple Data Distribution Simultaneously," *Symmetry (Basel)*, vol. 14, no. 2, Feb. 2022, doi: 10.3390/sym14020401.

[12] S. Hosseini, B. Turhan, and D. Gunarathna, "A systematic literature review and meta-analysis on cross project defect prediction," *IEEE Transactions on Software Engineering*, vol. 45, no. 2. Institute of Electrical and Electronics Engineers Inc., pp. 111–147, Feb. 01, 2019. doi: 10.1109/TSE.2017.2770124.

[13] K. K. Bejjanki, S. P. Kanchanapally, and M. K. Thota, "Class Imbalance Reduction and Centroid based Relevant Project Selection for Cross Project Defect Prediction," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 11, no. 6 s, pp. 293–302, Jun. 2023, doi: 10.17762/ijritcc.v11i6s.6933.

[14] Z. Li, X. Zhang, J. Guo, and Y. Shang, "Class Imbalance Data-Generation for Software Defect Prediction," in *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, IEEE Computer Society, Dec. 2019, pp. 276–283. doi: 10.1109/APSEC48747.2019.00045.

[15] S. Kumar Pandey and A. Kumar Tripathi, "An Empirical Study towards dealing with Noise and Class Imbalance issues in Software Defect Prediction," *Soft Computing*, vol. 25, pp. 13465–13492, 2021, doi: 10.21203/rs.3.rs-549406/v1.

[16] X. Yi, Y. Xu, Q. Hu, S. Krishnamoorthy, W. Li, and Z. Tang, "ASN-SMOTE: a synthetic minority oversampling method with adaptive qualified synthesizer selection," *Complex and Intelligent Systems*, vol. 8, no. 3, pp. 2247–2272, Jun. 2022, doi: 10.1007/s40747-021-00638-w.

[17] L. Goel, M. Sharma, S. K. Khatri, and D. Damodaran, "Cross-project defect prediction using data sampling for class imbalance learning: an empirical study," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 36, no. 2, pp. 130–143, 2021, doi: 10.1080/17445760.2019.1650039.

[18] A. Saifudin, S. W. H. L. Hendric, B. Soewito, F. L. Gaol, E. Abdurachman, and Y. Heryadi, "Tackling Imbalanced Class on Cross-Project Defect Prediction Using Ensemble SMOTE," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Nov. 2019. doi: 10.1088/1757-899X/662/6/062011.

[19] X. Fan, S. Zhang, K. Wu, W. Zheng, and Y. Ge, "Cross-Project Software Defect Prediction Based on SMOTE and Deep Canonical Correlation Analysis," *Computers, Materials & Continua*, vol. 0, no. 0, pp. 1–10, 2023, doi: 10.32604/cmc.2023.046187.

[20] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "SMOTE for regression," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, pp. 378–389. doi: 10.1007/978-3-642-40669-0_33.

[21] A. Fernández, S. García, F. Herrera, and N. V Chawla, "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary," 2018.

[22] W. Wang and T. W. Pai, "Enhancing Small Tabular Clinical Trial Dataset through Hybrid Data Augmentation: Combining SMOTE and WCGAN-GP," *Data (Basel)*, vol. 8, no. 9, Sep. 2023, doi: 10.3390/data8090135.

[23] D. Elreedy and A. F. Atiya, "A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance," *Inf Sci (N Y)*, vol. 505, pp. 32–64, Dec. 2019, doi: 10.1016/j.ins.2019.07.070.

[24] J. Lu *et al.*, "Deep learning model to predict exercise stress test results: Optimizing the diagnostic test selection strategy and reduce wastage in suspected coronary artery disease patients," *Comput Methods Programs Biomed*, vol. 240, Oct. 2023, doi: 10.1016/j.cmpb.2023.107717.

[25] A. X. Wang, S. S. Chukova, A. Sporle, B. J. Milne, C. R. Simpson, and B. P. Nguyen, "Enhancing public research on citizen data: An empirical investigation of data synthesis using Statistics New Zealand's Integrated Data Infrastructure," *Inf Process Manag*, vol. 61, no. 1, Jan. 2024, doi: 10.1016/j.ipm.2023.103558.

[26] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: combination, implementation and evaluation," *Expert Syst Appl*, vol. 244, Apr. 2024, doi: 10.1016/j.eswa.2023.122778.

[27] S. Kamthe, S. Assefa, M. Chase, A. I. Research, and M. Deisenroth, "COPULA FLOWS FOR SYNTHETIC DATA GENERATION," *arXiv e-print*, pp. 1–15, 2021, doi: arXiv:2101.00598v1.

[28] H. J. Asghar, M. Ding, T. Rakotoarivelo, S. Mrabet, and D. Kaafar, "Differentially private release of high-dimensional datasets using the gaussian copula," *Journal of Privacy and Confidentiality*, vol. 10, no. 2, pp. 1–38, 2020, doi: 10.29012/jpc.686.

[29] S. An and J.-J. Jeon, "Distributional Learning of Variational AutoEncoder: Application to Synthetic Data Generation," in *Neural Information Processing System* , New Orleans, USA, Dec. 2023.

[30] S. Pal, "Generative Adversarial Network-based Cross-Project Fault Prediction," May 2021, [Online]. Available: http://arxiv.org/abs/2105.07207

[31] Y. Zhong, K. Song, S. K. Lv, and P. He, "An Empirical Study of Software Metrics Diversity for Cross-Project Defect Prediction," *Math Probl Eng*, vol. 2021, 2021, doi: 10.1155/2021/3135702.
[32] H. Tong, B. Liu, S. Wang, and Q. Li, "Transfer-Learning Oriented Class Imbalance Learning for Cross-Project Defect Prediction," Jan. 2019, [Online]. Available: http://arxiv.org/abs/1901.08429
[33] "AEEEM and Other SDP Datasets," Github. Accessed: Apr. 16, 2024. [Online]. Available: https://github.com/bharlow058/AEEEM-and-other-SDP-datasets
[34] "Promise Repository," GitHub. Accessed: Apr. 16, 2024. [Online]. Available: https://github.com/feiwww/PROMISE-backup
[35] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: combination, implementation and evaluation," *Expert System With Applications*, vol. 244, Apr. 2023, doi: 10.1016/j.eswa.2023.122778.
[36] A. O. Balogun *et al.*, "SMOTE-Based Homogeneous Ensemble Methods for Software Defect Prediction," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Science and Business Media Deutschland GmbH, 2020, pp. 615–631. doi: 10.1007/978-3-030-58817-5_45.
[37] Y. Khatri and S. K. Singh, "Cross project defect prediction: a comprehensive survey with its SWOT analysis," *Innov Syst Softw Eng*, vol. 18, no. 2, pp. 263–281, Jun. 2022, doi: 10.1007/s11334-020-00380-5.
[38] S. DEMİR and E. K. ŞAHİN, "Evaluation of Oversampling Methods (OVER, SMOTE, and ROSE) in Classifying Soil Liquefaction Dataset based on SVM, RF, and Naïve Bayes," *European Journal of Science and Technology*, Feb. 2022, doi: 10.31590/ejosat.1077867.
[39] S. Mehta and K. S. Patnaik, "Improved prediction of software defects using ensemble machine learning techniques," *Neural Comput Appl*, vol. 33, no. 16, pp. 10551–10562, Aug. 2021, doi: 10.1007/s00521-021-05811-3.
[40] A. Figueira and B. Vaz, "Survey on Synthetic Data Generation, Evaluation Methods and GANs," *Mathematics*, vol. 10, no. 15. MDPI, Aug. 01, 2022. doi: 10.3390/math10152733.
[41] A. Gonzales, G. Guruswamy, and S. R. Smith, "Synthetic data in health care: A narrative review," *PLOS Digital Health*, vol. 2, no. 1, p. e0000082, Jan. 2023, doi: 10.1371/journal.pdig.0000082.
[42] A. Salazar, L. Vergara, and G. Safont, "Generative Adversarial Networks and Markov Random Fields for oversampling very small training sets," *Expert Syst Appl*, vol. 163, Jan. 2021, doi: 10.1016/j.eswa.2020.113819.
[43] K. Zhang, N. Patki, and K. Veeramachaneni, "Sequential Models in the Synthetic Data Vault," Jul. 2022, doi: 10.48550/ARXIV.2207.14406.
[44] T. Kokosi and K. Harron, "Synthetic data in medical research," *BMJ Medicine*, vol. 1, no. 1, p. e000167, Sep. 2022, doi: 10.1136/bmjmed-2022-000167.
[45] A. Montanez, "SDV: An Open Source Library for Synthetic Data Generation," M.Eng. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2018.
[46] DataCebo Team, "Synthetic Data Vault." Accessed: Apr. 03, 2024. [Online]. Available: https://docs.sdv.dev/sdv
[47] N. Patki, R. Wedge, and K. Veeramachaneni, "The synthetic data vault," in *Proceedings - 3rd IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016*, Institute of Electrical and Electronics Engineers Inc., Dec. 2016, pp. 399–410. doi: 10.1109/DSAA.2016.49.
[48] M. Fallahian, M. Dorodchi, and K. Kreth, "GAN-Based Tabular Data Generator for Constructing Synopsis in Approximate Query Processing: Challenges and Solutions," *Mach Learn Knowl Extr*, vol. 6, no. 1, pp. 171–198, Jan. 2024, doi: 10.3390/make6010010.
[49] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling Tabular data using Conditional GAN," *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 7335–7345, Jun. 2019, [Online]. Available: http://arxiv.org/abs/1907.00503
[50] O. Lee, M. Jong Cheon, D. Hee Lee, J. Woong Park, H. Jin Choi, and J. Seuck Lee, "CTGAN VS TGAN? WHICH ONE IS MORE SUITABLE FOR GENERATING SYNTHETIC EEG DATA," *J Theor Appl Inf Technol*, vol. 31, no. 10, 2021, [Online]. Available: https://www.researchgate.net/publication/352007070
[51] H. Ashkenazy *et al.*, "FastML: A web server for probabilistic reconstruction of ancestral sequences," *Nucleic Acids Res*, vol. 40, no. W1, Jul. 2012, doi: 10.1093/nar/gks498.
[52] K. Muandet, "Impossibility of Collective Intelligence," Jun. 2022, [Online]. Available: http://arxiv.org/abs/2206.02786
[53] F. Benali, D. Bodénès, N. Labroche, and C. De Runz, "MTCopula: Synthetic Complex Data Generation Using Copula," 2021. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/AutoUniv
[54] N. Patki, "The Synthetic Data Vault: Generative Modeling for Relational Databases," 2016.
[55] K. Gregor, G. Papamakarios, F. Besse, L. Buesing, and T. W. Deepmind, "Temporal Difference Variational Auto-Encoder," in *International Conference on Learning Representations*, New Orleans, Louisiana, United States, May 2019, pp. 1–17.
[56] T. A. Pham, J. H. Lee, and C. S. Park, "MST-VAE: Multi-Scale Temporal Variational Autoencoder for Anomaly Detection in Multivariate Time Series," *Applied Sciences (Switzerland)*, vol. 12, no. 19, Oct. 2022, doi: 10.3390/app121910078.
[57] C. Pak, T. T. Wang, and X. H. Su, "An Empirical Study on Software Defect Prediction Using Over-Sampling by SMOTE," *International Journal of Software Engineering and Knowledge Engineering*, vol. 28, no. 6, pp. 811–830, Jun. 2018, doi: 10.1142/S0218194018500237.
[58] R. Herteno, S. W. Saputro, M. R. Faisal, R. A. Nugroho, and A. Maulana Akbar, "Enhancing Software Defect Prediction through Hybrid Optimization for Feature Selection and Gradient Boosting Classification," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 169–181, 2024, doi: 10.35882/jeeemi.v6i2.388.
[59] W. Li, Y. Chen, and Y. Song, "Boosted K-nearest neighbor classifiers based on fuzzy granules," *Knowl Based Syst*, vol. 195, May 2020, doi: 10.1016/j.knosys.2020.105606.
[60] P. Pietrzak and M. Wolkiewicz, "Online Detection and Classification of PMSM Stator Winding Faults Based on Stator Current Symmetrical Components Analysis and the KNN Algorithm," *Electronics (Switzerland)*, vol. 10, no. 15, Aug. 2021, doi: 10.3390/electronics10151786.
[61] W. B. Zulfikar, A. R. Atmadja, and S. F. Pratama, "Sentiment Analysis on Social Media Against Public Policy Using Multinomial Naive Bayes," *Scientific Journal of Informatics*, vol. 10, no. 1, pp. 25–34, Jan. 2023, doi: 10.15294/sji.v10i1.39952.
[62] A. V. D. Sano, A. A. Stefanus, E. D. Madyatmadja, H. Nindito, A. Purnomo, and C. P. M. Sianipar, "Proposing a visualized comparative review analysis model on tourism domain using Naïve Bayes classifier," in *Procedia Computer Science*, Elsevier B.V., 2023, pp. 482–489. doi: 10.1016/j.procs.2023.10.549.
[63] G. Zhu *et al.*, "Naive Bayes Classifiers for Music Emotion Classification Based on Lyrics," in *EEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, Wuhan, China, May 2017, pp. 635–638. doi: 10.1109/ICIS.2017.7960070.
[64] P. L. Kumalasari, R. Arifudin, and Alamsyah, "Decision Making System to Determine Childbirth Process with Naïve Bayes and Forward Chaining Methods," *Scientific Journal of Informatics*, vol. 7, no. 2, pp. 2407–7658, 2020, doi: 10.15294/sji.v7i2.25352.
[65] K. A. Dhanya, S. Vajipayajula, K. Srinivasan, A. Tibrewal, T. S. Kumar, and T. G. Kumar, "Detection of Network Attacks using Machine Learning and Deep Learning Models," in *Procedia Computer Science*, Elsevier B.V., 2023, pp. 57–66. doi: 10.1016/j.procs.2022.12.401.
[66] N. Saran and N. Kesswani, "A comparative study of supervised Machine Learning classifiers for Intrusion Detection in Internet of Things," in *Procedia Computer Science*, Elsevier B.V., 2022, pp. 2049–2057. doi: 10.1016/j.procs.2023.01.181.
[67] L. Alfaris, R. C. Siagian, A. C. Muhammad, U. I. Nyuswantoro, N. Laeiq, and F. D. Mobo, "Classification of Spiral and Non-Spiral Galaxies using Decision Tree Analysis and Random Forest Model: A Study on the Zoo Galaxy Dataset," *Scientific Journal of Informatics*, vol. 10, no. 2, pp. 139–150, May 2023, doi: 10.15294/sji.v10i2.44027.
[68] V. Maulida, R. Herteno, M. R. Faisal, D. Kartini, and F. Abadi, "Feature Selection Using Firefly Algorithm with Tree-Based Classification in Software Defect Prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 5, no. 4, pp. 223–230, 2023, doi: 10.35882/jeemi.v5i4.315.
[69] M. K. Suryadi, R. Herteno, S. W. Saputro, M. R. Faisal, and R. A. Nugroho, "A Comparative Study of Various Hyperparameter Tuning

on Random Forest Classification with SMOTE and Feature Selection Using Genetic Algorithm in Software Defect Prediction," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 6, no. 2, pp. 137–147, Apr. 2024, doi: 10.35882/jeeemi.v6i2.375.

[70] M. R. Ansyari, M. I. Mazdadi, F. Indriani, D. Kartini, and T. H. Saragih, "Implementation of Random Forest and Extreme Gradient Boosting in the Classification of Heart Disease Using Particle Swarm Optimization," *Journal of Electronics, Electromedical Engineering, and Medical Informatics*, vol. 5, no. 4, pp. 250–260, 2023, doi: 10.35882/jeemi.v5i4.322.

[71] T. Ciu and R. S. Oetama, "Logistic Regression Prediction Model for Cardiovascular Disease," *International Journal of New Media Technology*, vol. VII, no. 1, p. 33, 2020, doi: 10.31937/ijnmt.v7i1.1340.

[72] R. T. Yunardi, R. Apsari, and M. Yasin, "Comparison of Machine Learning Algorithm For Urine Glucose Level Classification Using Side-Polished Fiber Sensor," 2020. [Online]. Available: http://jeeemi.org/index.php/jeeemi/index

[73] A. Balboa, A. Cuesta, J. González-Villa, G. Ortiz, and D. Alvear, "Logistic regression vs machine learning to predict evacuation decisions in fire alarm situations," *Saf Sci*, vol. 174, Jun. 2024, doi: 10.1016/j.ssci.2024.106485.

[74] R. Malhotra, R. Kapoor, P. Saxena, and P. Sharma, "SAGA: A Hybrid Technique to handle Imbalance Data in Software Defect Prediction," in *ISCAIE 2021 - IEEE 11th Symposium on Computer Applications and Industrial Electronics*, Institute of Electrical and Electronics Engineers Inc., Apr. 2021, pp. 331–336. doi: 10.1109/ISCAIE51753.2021.9431842.

[75] M. H. Murad, A. K. Balla, M. S. Khan, A. Shaikh, S. Saadi, and Z. Wang, "Thresholds for interpreting the fragility index derived from sample of randomised controlled trials in cardiology: a meta-epidemiologic study," *BMJ Evid Based Med*, vol. 28, no. 2, pp. 133–136, 2023, doi: 10.1136/bmjebm-2021-111858.

[76] S. Dubey, G. Tiwari, S. Singh, S. Goldberg, and E. Pinsky, "Using machine learning for healthcare treatment planning," *Front Artif Intell*, vol. 6, 2023, doi: 10.3389/frai.2023.1124182.

[77] N. A. A. Khleel and K. Nehéz, "A novel approach for software defect prediction using CNN and GRU based on SMOTE Tomek method," *J Intell Inf Syst*, vol. 60, no. 3, pp. 673–707, Jun. 2023, doi: 10.1007/s10844-023-00793-1.

[78] J. L. Ortega, "The presence of academic journals on Twitter The presence of academic journals on Twitter and its relationship with dissemination (tweets) and research impact (citations)," *Aslib Journal of Information Management*, vol. 69, no. 6, pp. 674–687, 2017, doi: 10.1108/AJIM-02-2017-0055.

[79] M. Nabil, M. Rahman, R. A. Nugroho, M. R. Faisal, F. Abadi, and R. Herteno, "Optimized multi correlation-based feature selection in software defect prediction," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 22, no. 3, pp. 598–605, 2024, doi: 10.12928/TELKOMNIKA.v22i3.25793.

[80] S. Zheng, J. Gai, H. Yu, H. Zou, and S. Gao, "Training data selection for imbalanced cross-project defect prediction" *Computers and Electrical Engineering*, vol. 94, Sep. 2021, doi: 10.1016/j.compeleceng.2021.107370.

BIBLIOGRAPHY

Putri Nabella is currently a bachelor's degree student in computer science from Lambung Mangkurat University. Her current area of research interest is centered on software defect prediction. Additionally, her final assignment involves research centered on predicting defects in software. The goal of her research is to improving defect prediction in software.

Rudy Herteno is currently a lecturer in the Faculty of Mathematics and Natural Science, Lambung Mangkurat University. He received his bachelor's degree in Computer Science from Lambung Mangkurat University and a master's degree in Informatics from STMIK Amikom University. His research interests include software engineering, software defect prediction and deep learning. He can be contacted at email: rudy.herteno@ulm.ac.id.

Setyo Wahyu Saputro is a lecturer in Computer Science Department, Faculty of Mathematics and Natural Science, Lambung Mangkurat University in Banjarbaru. He received bachelor's degree also in Computer Science from Lambung Mangkurat Univesity, and received his master's degree in Informatics from STMIK Amikom University. His research interests include software engineering and artificial intelligence applications. He can be contacted at email: setyo.saputro@ulm.ac.id

Mohammad Reza Faisal received the B.Sc. and M.Eng. degrees in physics and informatics from Bandung Institute of Technology, Bandung, Indonesia. He also received a B.Eng. degree in informatics from Pasundan University, Bandung, Indonesia, and a Ph.D. in computer science from Kanazawa University, Ishikawa, Japan. He is currently a lecturer in the Computer Science Department, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University in Banjarbaru, Indonesia. His research interests include artificial intelligence applications, text mining, and software engineering. He can be contacted at email: reza.faisal@ulm.ac.id.

Friska Abadi received his bachelor's degree in computer science from Lambung Mangkurat University, Banjarbaru, Indonesia, in 2011. He also received a master's degree in informatics from STMIK Amikom, Yogyakarta, in 2016. He is currently the lecturer in the Computer Science Department, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University, Banjarbaru, Indonesia. His research interests include data mining and software engineering. He can be contacted at email: friska.abadi@ulm.ac.id