IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.19, No.2, April 2025, pp. 211~222
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: 10.22146/ijccs.105290

Preprocessing Algorithm for K-Means Anomaly Detection on Payment Logs

Reka Alamsyah[1], **Nur Rokhman***[2]
[1]Master Program of Computer Science, FMIPA UGM, Yogyakarta, Indonesia
[2]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]nurrokhman@ymail.com**, [2]rekaalamsyah1997@mail.ugm.ac.id

Abstrak

Sistem payment aggregator dengan fitur single settlement meningkatkan efisiensi transaksi. Namun demikian hal ini berisiko terhadap serangan siber dan kesalahan system. Resiko ini akan memunculkan kejadian abnormal atau anomali. Middleware service mencatat aktivitas transaksi dalam bentuk log. Data log dapat dianalisis untuk deteksi anomaly sebagai akibat dari serangan siber ataupun kesalahan system.

K-Means clustering kurang efektif untuk mendeteksi anomaly pada data log karena data log transaksi sering kali tidak terstruktur, tidak konsisten, dan memiliki skala fitur yang bervariasi.

Penelitian ini mengembangkan algoritma praproses untuk meningkatkan kualitas data sebelum klasterisasi. Data log transaksi dari Juli hingga Desember 2023 digunakan, dengan tahapan praproses mencakup normalisasi, standarisasi, dan Principal Component Analysis (PCA). K-Means diterapkan dengan inisialisasi K-Means++, dan jumlah klaster ditentukan menggunakan kneedle algorithm. Hasil menunjukkan bahwa standarisasi meningkatkan segmentasi, dan PCA meningkatkan efektifitas deteksi anomaly

Kata kunci: *Praproses Data, K-Means, Deteksi Anomali, Log Middleware, Kneedle Algorithm*

Abstract

The payment aggregator system with the single settlement feature enhances transaction efficiency. However, this also poses risks of cyberattacks and system errors. These risks can lead to abnormal events or anomalies. The middleware service records transaction activities in the form of logs. Log data can be analyzed for anomaly detection resulting from cyberattacks or system errors.

K-Means clustering is less effective in detecting anomalies in log data because transaction log data is often unstructured, inconsistent, and has varying feature scales.

This study develops a preprocessing algorithm to improve data quality before clustering. Transaction log data from July to December 2023 is used, with preprocessing stages including normalization, standardization, and Principal Component Analysis (PCA). K-Means is applied with K-Means++ initialization, and the number of clusters is determined using the kneedle algorithm. The results show that standardization improves segmentation, and PCA enhances anomaly detection effectiveness.

Keywords: *Data Preprocessing, K-Means, Anomaly Detection, Middleware Logs, Kneedle Algorithm*

1. INTRODUCTION

In the digital era, the demand for fast, secure, and efficient payment systems has become

Received March 10th,2025; Revised Aprtil 28th, 2025; Accepted April 30th, 2025

increasingly significant. Financial service providers continuously innovate by introducing digital solutions, one of which is middleware service to support the single settlement transaction process. Middleware acts as a bridge between system components while recording transaction activities in structured logs. These logs are crucial for real-time transaction status monitoring. However, the growing volume of log data poses new challenges, particularly in identifying patterns of unusual or abnormal activity (anomalies) that could disrupt the system [1][2].

Anomalies in middleware activities can reflect various issues, such as technical disruptions, recording failures, or even illegal activities that harm the system. Early detection of anomaly patterns is therefore essential to maintain system performance and security. One effective approach is clustering, which groups data based on shared characteristics. This technique facilitates the identification of anomalous patterns more effectively. However, the success of clustering largely depends on the quality of the dataset, which is often in raw form and contains issues such as noise, duplication, or inconsistency [3][4]. Comprehensive preprocessing is thus required to clean and prepare the data for analysis.

Data preprocessing is a critical step in producing high-quality datasets for subsequent analysis. This process involves various techniques, including feature selection to identify relevant data attributes [2] feature extraction to derive additional information from existing data [5], normalization to standardize data scales [1][6], and dimensional reduction to simplify data complexity [7]. Previous studies have demonstrated that effective preprocessing significantly enhances the accuracy of clustering models, leading to more optimal analysis outcomes.

In this study, the K-Means algorithm is employed for its reliability in handling large datasets and its ability to detect anomalies by measuring the distance of data points from cluster centroids. This algorithm enables efficient identification of unusual activity patterns and supports the grouping of middleware log data based on activity characteristics [8].

To address this limitation, this study proposes the development of an optimized data preprocessing algorithm tailored for middleware log data. The objective is to evaluate how preprocessing influences the effectiveness of K-Means-based anomaly detection while ensuring that the proposed approach can be generalized to other systems with complex log structures.

2. METHODS

This study was conducted through a series of systematically designed stages to achieve its objectives. The process began can see in Figure 1 with the first step is collection of raw data from the payment aggregator system, which served as the research object. The raw data underwent preprocessing to ensure their quality and suitability as a research dataset. This preprocessing included data cleaning, transformation, and feature selection or dimensionality reduction to produce an optimal dataset tailored for analysis.

Once the dataset was prepared, the study progressed to the modeling and evaluation phase, where the K-Means algorithm was applied to detect anomalies in transaction data. The evaluation process utilized relevant performance metrics to assess the accuracy and effectiveness of the developed model. The evaluated model was then implemented in anomaly detection scenarios to identify deviations from normal transaction patterns.

The results of the experiments and anomaly detection were thoroughly analyzed to interpret the identified patterns and assess their relevance to the payment aggregator domain. This analysis formed the foundation for drawing conclusions, which were presented as the primary contribution of this study to the development of preprocessing algorithms supporting K-Means-based anomaly detection. Through this approach, the research aims to provide significant contributions to transaction data management and the improvement of payment system security.

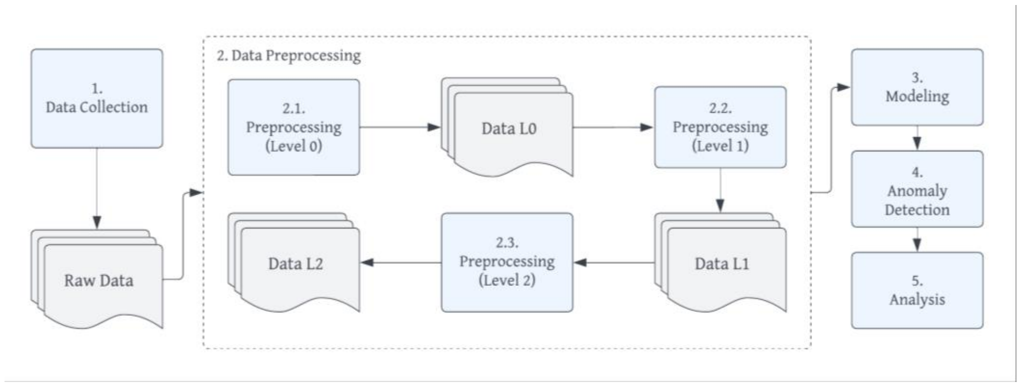

Figure 1 Methodology Research

2.1 *Data Collection*

The data collection process involves gathering the necessary data for this research. The data used in this study comes from three entities: logs single settlement, transactions settlement, and invoices. The data collected covers transaction records from July to December 2023. To retrieve this data, tools or programs are required to extract it from the data source, which is the *Google BigQuery* datalake. The extracted data is stored in CSV format. During the data collection process, it is ensured that the structure of the collected data does not contain any Personal Identifiable Information (PII).

2.2 *Data Preprocessing*

The objective of this stage is to transform raw data into a dataset that is ready for modeling. This step is critical to ensure that the modeling process achieves optimal performance. Based on this objective, the preprocessing tasks are divided into three levels, as illustrated in Figure 2, each with its specific focus. The following outlines the stages of this process.

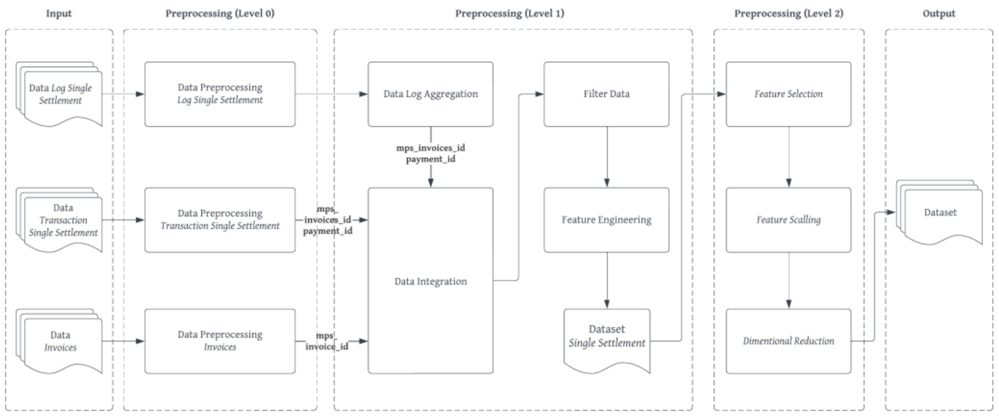

Figure 2 Data Preprocessing Flow

2.2.1 *Preprocessing Level 0*

Data preprocessing at Level 0 focuses on each individual data entity. The collected data may contain duplicates, inconsistent data types, or ordinal data requiring further processing. Key activities at this stage include removing duplicate records, encoding ordinal data into numerical representations, and standardizing data types to ensure consistency. While preprocessing is applied to all entities, the single settlement log data requires an additional preprocessing step, as outlined in Algorithm 1.

ALGORITHM 1: PREPROCESSING LEVEL 0

```
Input:
log: Data Raw Log Transaction Single Settlement
trx: Data Raw Transaction Single Settlement
inv: Data Raw Invoices
Output: Processed log, transaction, and invoice data at Level 0
1  Function PreprocessingLogLevelZero(log, trx, inv):
2      $L' \leftarrow$ RemoveDuplication($log$)
3      $T_0 \leftarrow$ RemoveDuplication($trx$)
4      $I_0 \leftarrow$ RemoveDuplication($inv$)
4      $L_{extracted} \leftarrow$ ExtractLog($L'.message.acc\_token, L'.message.refresh\_token$)
5      $L_0 \leftarrow$ FeatureOrdinal($L_{extracted}.message\_type$)
6      return $L_0, T_0, I_0$
7  End Function
```

2.2.2 *Preprocessing Level 1*

Preprocessing at Level 1 builds upon the results of the Level 0 preprocessing stage. This stage focuses on integrating the single settlement transaction log data with the settlement transaction and invoice data to create a unified dataset.

ALGORITMA 2: PREPROCESSING LEVEL 1

```
Input:
$L_{processed}$ : Set of Data Log Transaction Single Settlement Level 0
$T_0$: Set of Data Transaction Single Settlement Level 0
$I_0$: Set of Data Invoices Level 0
Output:
$D_1$: Unified dataset Single Settlement
1  Function PreprocessingLevelOne($L_{processed}, T_0, I_0$)
2      $L_{agg} \leftarrow$ AggregationLog($L_{processed}$)
3      $D_{integrated} \leftarrow$ IntegrationData($L_{processed}, T_0, I_0$)
4      $D_{filtered} \leftarrow$ FilterDataSettled($D_{integrated}$)
5      $D_1 \leftarrow$ FeatureCreation($D_{filtered}$)
6      Return $D_1$
7  End Function
```

In Algorithm 2, which pertains to preprocessing level 1, several functions are utilized for data transformation and processing. The AggregationLog function, executed in the second step, is responsible for transforming and aggregating log data that was initially stored in a row-wise format. After being processed by this function, the log data is grouped based on a key representing the transaction, ensuring that each transaction is consolidated into a single row. This process generates derived attributes, such as the number of log occurrences, success ratio, and sequence characteristics of the transaction.

Next, in the third step, the IntegrationData function is used to merge relevant entities, such as single settlement transaction data and invoice data, resulting in a more comprehensive dataset. The primary goal of this step is to obtain information regarding the transaction's settlement status, which is recorded within the single settlement entity. In the fourth step, the FilterDataSettled function is applied to filter and retain only transactions that have been successfully settled.

Finally, in the fifth step, the feature creation process is performed to generate a new attribute representing the time difference between the initial settlement request and its completion.

This feature is not created at the beginning of the process, as the necessary comparative data is only available within the invoice entity. The output of preprocessing level 1 consists of the derived data and the essential information required for the subsequent analysis phase.

2.2.3 Preprocessing Level 2

At Level 2, data preprocessing focuses on selecting relevant features to be used in the modeling process based on the case study analysis show at Table 1.

Table 1 Case Study for Preprocessing

Code	Case Study	Description
S001	Segmentation Activity and Performances	Measuring the number of transaction activities, requests, responses, successes, and failures occurring within a transaction session
S002	Segmentation Success Rate Transaction	Using the success and error ratios in middleware service stages as indicators of process quality
S003	Segmentation Pattern Log	Analyzing patterns and variations in transaction logs using statistical and sequential features

This stage is followed by applying feature scaling techniques to standardize the data using MinMax Scaler and Standardization Scaler.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad \textbf{(1)}$$

Where:
- X' The normalized value
- X The original or raw value
- X_{min} The minimum value across the entire dataset
- X_{max} The maximum value across the entire dataset

$$z = \frac{x - \mu}{\sigma} \quad \textbf{(2)}$$

Where:
- z The standardized value (z-score)
- x The original value of a feature
- μ The mean of the feature
- σ The standard deviation of the feature

Than, reducing dimensionality using the Principal Component Analysis (PCA) technique. Additionally, this study involves creating multiple dataset combinations based on case study analysis, incorporating both feature scaling techniques and PCA to explore their impact on the modeling process.

This phase also contributes to the formulation of the experimental situations to be executed in this investigation. Eighteen experimental scenarios were executed, integrating case studies, preprocessing scaling methods, and PCA. The objective of these studies is to assess the influence of each scenario on clustering quality and the efficacy of anomaly detection. An illustration of the scenario formation can be seen in Figure 3.

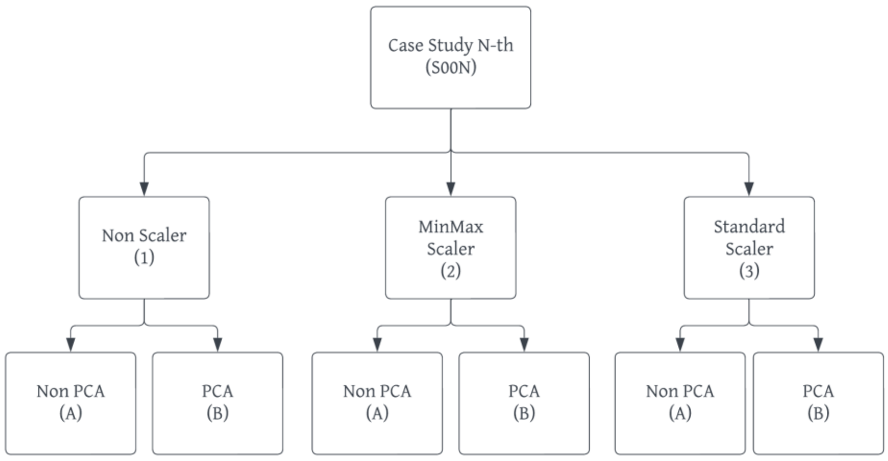

Figure 3 Ilustration formation of scenario experiment

2.3 Modeling and Evaluation

Following the data preprocessing stage, the next step involves modeling the dataset obtained. The modeling approach employed in this research is K-Means Clustering, chosen for its effectiveness in grouping data based on patterns and feature similarities. However, the implementation of the K-Means method requires two key parameters to be determined: the initialization of centroids and the optimal number of clusters (k).

In this research, the k-means++ algorithm is used for centroid initialization in the K-Means modeling process. This algorithm is selected for its ability to improve clustering efficiency and accuracy compared to random centroid initialization [9].

ALGORITHM 3: INITIALIZATION CENTROID K-MEANS++

```
Input: X: Parameter Dataset; k: Total Cluster
Output: μ: Set of Data
1    Function InitializeCentroids(X, k):
2        $\mu[1] \leftarrow$ SelectRandom(X)
4        for $j \leftarrow 2$ to k do
5            $D \leftarrow$ Empty Array with length by $|X|$
7            for $i \leftarrow 1$ to $|X|$ do
8                $D[i] \leftarrow \min($EuclideanDistance$(X[i], \mu[j]))$
9            end for
10           $P \leftarrow$ ProbabilityDistribution$(D)$
11           $\mu[j] \leftarrow$ SelectRandomWeighted$(X,P)$
12       end for
13       return $\mu$
14   End Function
```

With k-means++, the initial centroids are selected probabilistically in equation (**3**, considering the maximum distance between data points, thus expediting the clustering process. The use of k-means++ is a constraint in this study, as the primary focus is directed towards the data preprocessing stage. Therefore, k-means++ is employed as a fixed initialization method for the K-Means algorithm.

$$P(x) = \frac{D(x)^2}{\sum D(x)^2} \quad \textbf{(3)}$$

Where:
- $P(x)$ The probability of selecting data point x as the next centroid
- $D(x)$ The Euclidean Distance
- $\sum D(x)^2$ The Sum of squared values minimum distances all data points in dataset

The overall architecture of the modeling and evaluation stages is depicted in Figure 4, which provides a detailed representation of the workflow, from centroid initialization to the determination of the optimal number of clusters.

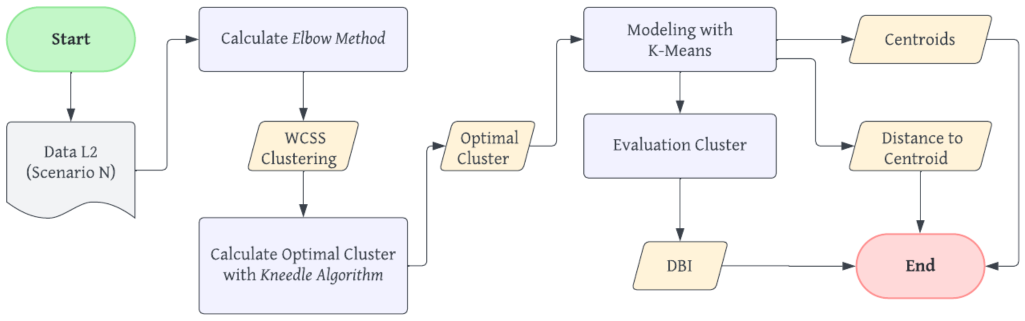

Figure 4 Modeling Flow

To determine the optimal number of clusters (k), a combination of the Elbow Method and the Kneedle Algorithm is utilized. The Elbow Method provides an initial visualization through a graph representing the relationship between the number of clusters (k) and the Within-Cluster Sum of Squares (WCSS) (**4**). The point at which the WCSS reduction slows down is identified as the "elbow point," indicating the optimal number of clusters.

$$WCSS = \sum_{i=1}^{k} \sum_{x \in C_i} || x - \mu_i ||^2 \tag{4}$$

Where:
- k The total cluster will create with k-means
- C The set of cluster include with point every cluster
- $|| x - \mu_i ||^2$ The squared distance between data point x and its corresponding centroid

In this research, the Elbow Method is executed by performing 11 iterations(k), starting from 2 clusters up to 12 clusters. The WCSS value is calculated for each iteration, and the results are plotted on an Elbow Method graph. Subsequently, the Kneedle Algorithm is employed to precisely [10] identify the optimal number of clusters based on the WCSS values obtained.

Evaluation of the formed clusters is performed using the Davies-Bouldin Index (DBI), a metric that measures cluster quality. DBI evaluates how well clusters are separated and how compact they are, with a lower DBI value indicating better clustering performance.

$$DBI = \frac{1}{K} \sum_{i=1}^{K} DB_i \tag{5}$$

Where:
- K The total number of clusters formed in the clustering algorithm
- DB_i Davies-Bouldin score for each cluster, how similar that cluster is to other clusters

The DBI is calculated using equation (**5**), which considers both the intra-cluster distance and the inter-cluster separation. This evaluation is essential to validate the quality of the clusters and to ensure that the clustering results align with the research objectives.

2.4 Anomaly Detection

Anomaly detection is performed after clusters with their respective centroids are formed. In this research, a centroid-based method is employed to detect anomalies by leveraging the concept of clustering [11]. This technique relies on the ability to calculate and determine the centroid of each cluster, which represents normal data.

The method analyzes not only data within a single cluster (intra-cluster) but also considers anomaly patterns between clusters (inter-cluster) [12]. Consequently, this study is designed to identify anomalies both within clusters and across clusters.

This subsection provides a detailed explanation of the steps involved in detecting anomalies. One critical aspect to address before developing intra-cluster and inter-cluster anomaly detection algorithms is determining the threshold value equation **(6)**. The threshold significantly impacts the sensitivity of anomaly detection; therefore, its selection must be carefully tailored to the needs and characteristics of the analyzed data.

$$T = \mu + (k \times \sigma) \tag{6}$$

Where:
- T The threshold for identifying anomalies
- μ Mean of distance points to centroid
- k The standard factor with value set is 3
- σ Standard deviation of distance points to centroid

To evaluate the performance of anomaly detection, this research uses the recall metric, which measures the proportion of actual anomalies correctly identified by the algorithm. Recall is calculated using equation (**7)**

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

Where:
- True Positive (TP): The number of unique anomalies correctly detected in each scenario
- False Negative (FN): The number of anomalies that were not detected

A higher recall value indicates better sensitivity of the anomaly detection method, ensuring that most anomalies in the dataset are identified. By defining TP as unique anomalies for each scenario, the metric accurately reflects the method's ability to identify a diverse range of abnormal behaviors, minimizing the risk of overlooking critical issues.

3. RESULTS AND DISCUSSION

The primary objective of this study is to detect anomalies in middleware service transaction logs using K-Means clustering. By leveraging preprocessing and clustering techniques, this study evaluates the effectiveness of anomaly detection in identifying abnormal patterns within and across clusters from 18 scenarios. Key metrics such as Davies-Bouldin Index (DBI) and Recall are used to evaluate the quality of clustering and anomaly detection performance.

3.1 Clustering Result

The analysis results indicate that scaling techniques significantly influence the Davies-Bouldin Index (DBI) values obtained. As shown in Figure 5, the Standardization scaling technique consistently delivers the lowest DBI values for both Non-PCA and PCA approaches.

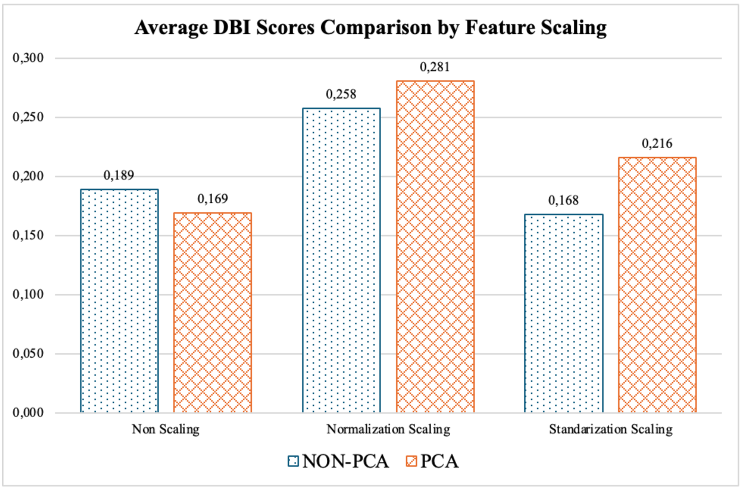

Figure 5 Result Avarage DBI Score Scaling in Non-PCA or PCA

This demonstrates that standardization is the most effective scaling method for improving clustering quality, as it balances the variance across features, preventing any single feature from dominating the clustering process. In contrast, the Normalization technique produces moderately good results but performs less effectively than standardization, especially in the PCA approach. Meanwhile, the No Scaling technique consistently results in the highest DBI values, indicating poor clustering quality due to unadjusted feature scales.

Furthermore, the comparison of average DBI values between PCA and Non-PCA approaches, as depicted in Figure 6, reveals that the Non-PCA approach consistently achieves lower DBI values than PCA. Although PCA can reduce data dimensionality, this transformation process may lead to the loss of critical information from the original features, especially when the data has already been well-preprocessed using scaling techniques. The Non-PCA approach, which retains the original feature information, fully benefits from preprocessing methods like standardization, resulting in better clustering in terms of compactness and separation.

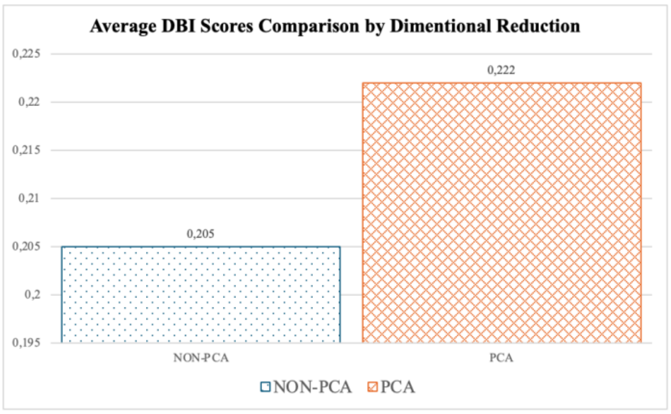

Figure 6 Result Avarage DBI Score in Non-PCA or PCA

Overall, these findings highlight the importance of preprocessing algorithms in the clustering process, particularly for K-Means. The Standardization technique proves to be the most effective scaling method for enhancing clustering quality in both PCA and Non-PCA approaches. However, in this study, the Non-PCA approach demonstrates superiority over PCA as it leverages the original feature information optimized through preprocessing. These results emphasize that selecting the appropriate preprocessing methods, including scaling and the decision to use PCA, plays a crucial role in achieving high-quality clustering outcomes.

3.2 Anomaly Detection Result

Scaling data is an essential step in preprocessing to ensure that each feature contributes equally to the anomaly detection process. Based on the detection results, a total of 84 unique anomalies were successfully identified across all scenarios and approaches. The presented graphs demonstrate that scaling techniques such as Normalization and Standardization consistently yield better results compared to No Scaling. In the Recall graph (Figure 1), these techniques achieve remarkably high Recall values of 0.996 for Non-PCA and 0.992 and 0.956 for PCA.

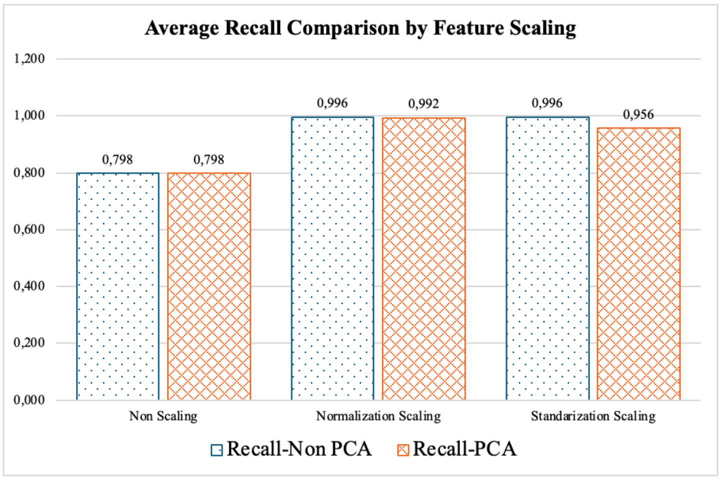

Figure 7 Result Avarage Recall by Scaling Non-PCA or PCA

In contrast, No Scaling produces significantly lower Recall values of 0.798, indicating that without scaling, certain features dominate the clustering process, which reduces the ability to detect anomalies. This is further supported by the results in the total anomalies detected graph, where Normalization and Standardization allow for the detection of 83 anomalies in both approaches, whereas No Scaling identifies only 67 anomalies.

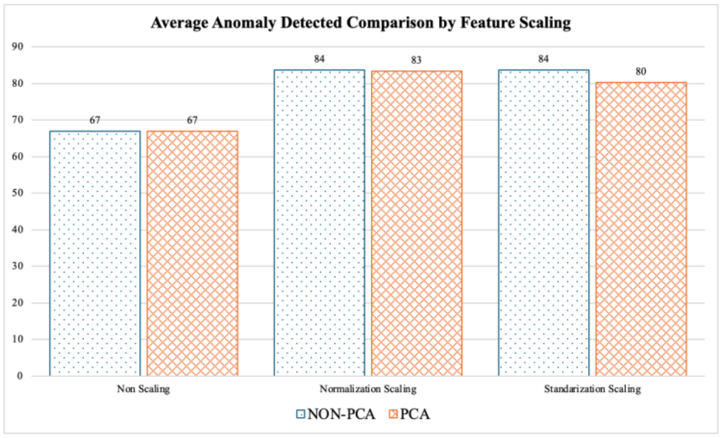

Figure 8 Result Avarage Total Anomalies Detected by Scaling Non-PCA or PCA

Additionally, the lower Recall observed in PCA with Standardization (0.956 compared to 0.996 in Non-PCA) suggests that the PCA transformation may discard critical information from the original features, particularly when the data has been adequately preprocessed using scaling. Therefore, Normalization and Standardization emerge as the most effective scaling techniques for improving Recall and the total number of anomalies detected, regardless of whether PCA or Non-PCA is used. These findings underscore the importance of selecting the appropriate scaling techniques to achieve optimal anomaly detection outcomes.

3.3 Discussion

The experiment demonstrates that preprocessing, particularly data scaling, plays a significant role in improving the quality of clustering and anomaly detection. Techniques such as Standardization and Normalization consistently produced lower DBI values, indicating more compact and well-separated clusters. These scaling methods also enhanced anomaly detection by increasing the number of detected anomalies and achieving higher Recall compared to the No Scaling approach. Furthermore, the Non-PCA approach showed more consistent results than PCA, especially when combined with Standardization, as PCA often resulted in the loss of critical information during dimensionality reduction. By retaining the original features, Non-PCA delivered more optimal anomaly detection performance across various scenarios. Therefore, combining appropriate preprocessing techniques, such as Normalization or Standardization, with the Non-PCA approach is an effective strategy for enhancing clustering quality and anomaly detection performance.

4. CONCLUSIONS

This study successfully developed a preprocessing algorithm to enhance the quality of clustering and the effectiveness of anomaly detection in middleware service activity data for single settlement transactions. Preprocessing techniques such as Normalization and Standardization proved effective in aligning data scales and distributions, resulting in improved clustering outcomes. By employing the K-Means Clustering model, the data was effectively grouped based on specific patterns and characteristics, enabling more accurate anomaly detection.

The centroid-based anomaly detection process identified 84 unique anomalies, with the combination of Non-PCA and Standardization demonstrating the best performance, including achieving perfect Recall in several scenarios. While PCA was beneficial for dimensionality reduction, the Non-PCA approach was more optimal in preserving essential feature information critical for anomaly detection.

The preprocessing algorithm developed in this study also accounted for the unique characteristics of middleware log data, such as activity distribution, process efficiency, and temporal transaction patterns. This approach not only improved clustering quality but also facilitated more effective anomaly detection. It contributed to enhancing the security and performance of the single settlement transaction system while supporting better management and monitoring of the system.

Overall, this research highlights that selecting appropriate preprocessing techniques, such as Normalization and Standardization, and applying suitable clustering models are crucial steps in supporting anomaly detection in digital transaction systems. These findings provide a strong foundation for further advancements in handling complex activity log data.

REFERENCES

[1] C. H. Saputra, "Integrasi Audit dan Teknik Clustering untuk Segmentasi dan Kategorisasi Aktivitas Log," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 11, no. 1, 2024, doi: 10.25126/jtiik.20241118071.

[2] K. DeMedeiros, C. Y. Koh, and A. Hendawi, “Clustering on the Chicago Array of Things: Spotting Anomalies in the Internet of Things Records,” *Future Internet*, vol. 16, no. 1, Jan. 2024, doi: 10.3390/fi16010028.

[3] N. Basha and A. K. P.S., “Distance-based K-Means Clustering Algorithm for Anomaly Detection in Categorical Datasets,” *Int J Comput Appl*, vol. 183, no. 11, 2021, doi: 10.5120/ijca2021921415.

[4] L. Fan, J. Ma, J. Tian, T. Li, and H. Wang, “Comparative Study of Isolation Forest and LOF algorithm in anomaly detection of data mining,” in *Proceedings - 2021 International Conference on Big Data, Artificial Intelligence and Risk Management, ICBAR 2021*, 2021. doi: 10.1109/ICBAR55169.2021.00008.

[5] H. K. Prakosa and N. Rokhman, “Anomaly Detection in Hospital Claims Using K-Means and Linear Regression,” *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 15, no. 4, p. 391, Oct. 2021, doi: 10.22146/ijccs.68160.

[6] A. A. Ma’ali, Girinoto, M. N. Ghiffari, and R. B. Hadiprakoso, “Analisis Log Web Server dengan Pendekatan Algoritme K-Means Clustering dan Feature Importance,” *Info Kripto*, vol. 16, no. 3, 2022, doi: 10.56706/ik.v16i3.60.

[7] S. Naeem, A. Ali, S. Anam, and M. M. Ahmed, “An Unsupervised Machine Learning Algorithms: Comprehensive Review,” *International Journal of Computing and Digital Systems*, vol. 13, no. 1, 2023, doi: 10.12785/ijcds/130172.

[8] C. Pradana, S. S. Kusumawardani, and A. E. Permanasari, “Comparison Clustering Performance Based on Moodle Log Mining,” in *IOP Conference Series: Materials Science and Engineering*, 2020. doi: 10.1088/1757-899X/722/1/012012.

[9] M. Gul and M. A. Rehman, “Big data: an optimized approach for cluster initialization,” *J Big Data*, vol. 10, no. 1, 2023, doi: 10.1186/s40537-023-00798-1.

[10] E. M. Qumsiyeh and M. N. Sabha, “Classification of Leaf Disease via Deep Neural Network combined with Clustering Algorithm,” *Computational Mathematics*, 2022.

[11] N. Almusallam, “An Unsupervised Feature Selection Method for Data-Driven Anomaly Detection Systems,” in *Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE*, 2020. doi: 10.1109/WETICE49692.2020.00016.

[12] M. Kherbache, D. Espes, and K. Amroun, “An Enhanced approach of the K-means clustering for Anomaly-based intrusion detection systems,” in *2021 International Conference on Computing, Computational Modelling and Applications (ICCMA)*, 2021, pp. 78–83. doi: 10.1109/ICCMA53594.2021.00021.