Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

June 2026. Vol. 5. No. 3; e-ISSN: 2808-4519

Application of K-Means Clustering: Bot Activity and Sybill Attack Detection on the Solana Blockchain

Bryant Tinambunan[1] , Hafizam Mufti Siregar[2], Ahmad Zulfan Hafiz Hrp[3*], Guez Rade Nainggolan[4]

[1,2,3,4]*Ilmu Komputer, FMIPA, Universitas Negeri Medan, Indonesia*
bryanttinambunan12@gmail.com[1], hafizhammufti123@gmail.com[2], ahmadzulfann2021@gmail.com[3*], radenainggolan05@gmail.com[4]

Abstract

With the development of Blockchain technology, for example, the Solana Blockchain has generated enormous amounts of data and possesses the 5Vs of Big Data: volume, velocity, value, veracity, and variety. This has brought challenges, for example, in distinguishing transactions carried out by humans from automated bots that often carry out market manipulation or Sybil attacks. Therefore, this research aims to detect bot activity on the Solana network by applying data mining techniques, namely the K-Means Clustering algorithm. From the large transaction data that will be extracted only a portion from the public Solana dataset in BigQuery, it will then be processed through a preprocessing stage to normalize the data and simplify complex data into simpler variables before being grouped. Because the extracted data is in the form of unlabeled data groups (unsupervised data), the Clustering Method is used because of its ability to recognize data groups based on behavioral or characteristic similarities without requiring initial data labels (unsupervised learning). The main variables used for the grouping process include transaction frequency, inter-arrival time (inter-transaction), and the number of unique program interactions. The results of this analysis are expected to map transaction accounts into several clusters based on their transaction patterns, allowing for the classification of bots and humans. This research is expected to demonstrate that Big Data infrastructure such as Google Cloud, using data mining techniques (Clustering), can be used to maintain the security and integrity of the blockchain ecosystem.

Keywords: *Big Data; Solana Blockchain; Data Mining; Clustering; K-Means; Bot Detection.*

1. Introduction

The Solana Blockchain is one of the largest blockchains, processing up to thousands of data points per second, which is characteristic of big data: Volume, Velocity, and Variety [2]. However, this high data processing speed creates a loophole for bots to exploit to carry out Sybil attacks and manipulate transactions that damage the blockchain ecosystem. The sheer volume of transactions per second makes it impossible to manually identify these bots.

Therefore, this study adopted a solution, namely using a data mining method. This data mining method, namely K-means Clustering, is considered effective in classifying transaction patterns carried out by humans and those carried out by bots [5]. By using variables such as transaction frequency and time patterns, this algorithm can separate clusters of transactions carried out by bots from those carried out by humans maintaining the Blockchain.

2. Theoretical Review

2.1. Data mining and big data

Data mining is the process of extracting valuable information from large amounts of data that are hidden or not directly visible using certain methods [3]. This science functions to find patterns, trends, and relationships in data that are impossible to detect using manual methods [4]. One of the main methods of data mining is Clustering, a method that groups data into certain groups (clusters) based on their similarities [5].
The main characteristic of the K-means algorithm that makes it suitable for this Blockchain case is its ability to process large amounts of data quickly and is able to identify and group based on pattern similarities to distinguish bot transactions from human ones [5]. The concept of big data has 5V characteristics [2]:

1. Volume: Big data is incredibly large, reaching petabytes. In the Solana Blockchain, this massive data is reflected in transaction data, which can amount to millions per day.

2. Velocity: Describes how fast the system processes data, such as the Solana Blockchain which can process incoming transaction data every second in real-time.
3. Variety: Showing data variations or different data formats. In the Solana Blockchain, this is reflected in the numerous tables, such as transaction tables.
4. Veracity: Describes how accurate or reliable data is. In the Solana Blockchain, data comes from Google BigQuery, which is guaranteed to be accurate.
5. Value: Reflecting how much valuable information can be extracted from such large amounts of data.

2.2. Clustering

In the world of data processing, clustering is known as a data mining tool that functions to map data into groups based on certain shared characteristics. The essence of this technique is to ensure that data within a group (cluster) is as similar as possible, while minimizing differences between groups.
Because it works without requiring pre-existing labels or categories, clustering is classified as an unsupervised learning method. On a big data scale, this technique is crucial for uncovering initially invisible patterns in complex data sets. Its applications are broad, ranging from analyzing consumer behavior and market segmentation to strengthening the security of computer networks and blockchain systems.
For example, in the blockchain ecosystem, clustering allows us to group digital wallet addresses based on their activity trends. This makes it easier to identify whether a transaction was made by a regular user or is an indication of a cyberattack such as a bot or Sybil attack [12].
One of the most popular approaches in this study is K-Means Clustering. This algorithm operates by dividing the data into a predetermined number of clusters (K). The process begins by determining the center point, or centroid, and then each data point is drawn to the closest centroid.

2.3. K-Means

The K-Means algorithm is one of the most widely used clustering techniques due to its efficiency and simplicity in handling large-scale data. Technically, the clustering procedure follows these systematic steps:

1. Centroid Initialization: Randomly selects an initial center point called the centroid.
2. Data: Each data point is assigned to the group that has the closest distance to its centroid.
3. Centroid Update: The centroid position is recalculated from the average position of all data in the group.
4. Convergence Iteration: The steps are repeated until the centroid position is stable or no longer experiences significant shifts.
   One technical obstacle is the vulnerability to outliers, which can significantly shift the centroid position. To address this, the use of validation methods such as the Silhouette Score is crucial.

2.4. Silhouette Score

Clustering results cannot be immediately considered valid without quality testing. This is where the Silhouette Score comes into play as an evaluation parameter to measure how precisely a piece of data fits into its cluster compared to other clusters. This method, introduced by Peter J. Rousseeuw in 1987, works by calculating the ratio of the internal distance of a cluster to the distance to its nearest neighboring cluster.

Simply put, the Silhouette Score compares two values: the average distance of a data point from other data points in the same cluster and the average distance of that data point from the nearest data point in the cluster. This calculation produces a value ranging from −1 to 1.

The interpretation of the Silhouette Score value is as follows:
1. Values close to 1 indicate that the data is in the correct cluster and has a clear separation from other clusters.
2. Values close to 0 indicate that the data is between two clusters so the separation is less clear.
3. Values approaching −1 indicate that the data is likely to be in an inappropriate cluster.

In practice, the Silhouette Score serves as a primary guide for researchers in determining the ideal number of clusters (K) in the K-Means algorithm. This evaluation technique is highly effective in network behavior research, such as separating organic human transactions from suspicious anomalous activity within the blockchain.

2.5. Sybil Attack

Wireless Sensor Networks (WSNs) have significantly improved the quality of human life. This technology allows individuals to communicate with distant devices without requiring a direct physical connection. This technological advancement allows various devices to interact with each other remotely, allowing for more efficient and practical activities.
However, this ease of interaction is not without its challenges. WSNs are vulnerable to various security threats, such as hacking and virus attacks. These vulnerabilities can be exploited by malicious parties to compromise the network's functionality and integrity.
One such malicious activity is a Sybil attack. In this attack, a single node in the network impersonates several different nodes. By creating these false identities, the attacker attempts to infiltrate the network and establish connections with legitimate nodes. This intrusion allows the attacker to manipulate network routing data, potentially using it for malicious purposes.

2.6. Blockchain

a technique to improve data security on a Blockchain-based Wireless Sensor Network. They use a microcontroller to connect various sensor nodes, and the security of the system is improved by utilizing blockchain technology.
In this system, the proposed method functions as a private cloud data center that visualizes data uploaded by sensors in the form of graphs and tables. The integration of blockchain technology with wireless sensor networks is detailed in the study.
To simplify blockchain security, the encryption method is changed from asymmetric to symmetric. The proposed method is divided into several stages to integrate Wireless Sensor Networks with Blockchain. The main stages are described as follows:

1. Data Collection and Processing: In the methodology to improve the security and reliability of Wireless Sensor Network by using Blockchain, the process begins with the data collection and processing stage.
2. Wireless Sensor Network Configuration: Wireless Sensor Network Configuration is another important step in the proposed methodology. In this stage, each node in the network is randomly assigned coordinates to ensure a diverse and expansive network layout.
3. Integration with Blockchain: The next step is integration with Blockchain. The process is as follows: Routing data that has been collected and stored in the form of a file. Then uploaded to the blockchain. This step is important because the blockchain provides an additional layer of security for the stored data.
4. Sybil Attack Simulation: In simulating a Sybil attack on a Blockchain network, a condition is created where an attacker (adversary)

3. Methodology

3.1. Research Flow

This research uses a quantitative approach based on data mining with the K-Means Clustering algorithm. The research flow is systematically designed through six main stages that are interconnected. The first stage is data acquisition, where Solana transaction data is extracted from Google BigQuery using SQL queries while simultaneously performing initial normalization using the MinMax method. The second stage is data pre-processing in the Python environment, including logarithmic transformation (log1p) to address skewness of the data distribution, followed by standardization using StandardScaler so that each feature has the same scale and is suitable for the clustering algorithm. The third stage is determining the optimal number of clusters by running the K-Means algorithm for a range of K from two to ten, then evaluating it using the Elbow method based on the WCSS value and the Silhouette Score method. The fourth stage is the implementation of K-Means clustering with two different scenarios, namely K = 4, which refers to the results of the Elbow Method and K = 3, which refers to the results of the Silhouette Score, to compare the resulting interpretations. The fifth stage is cluster interpretation, where each cluster is analyzed based on its centroid value and labeled according to dominant characteristics such as transaction frequency, gas fees, or number of interactions with other accounts. The sixth stage involves visualizing the results in graphical form (a combination of Elbow and Silhouette, as well as a two-dimensional distribution) and creating sample tables for manual verification. This entire flow is designed to ensure that each step can be explained transparently and reproducibly.

3.2. Data Acquisition from Google BigQuery

Data is sourced from the Solana public dataset on Google BigQuery. The data collection period was limited to one day, March 1, 2026, and only included wallets with more than five transactions to avoid passive accounts. The sample size was limited to 1,000 rows for methodology development purposes.

3.2.1. SQL Queries and Initial Normalization

```
WITH RawData AS (
    SELECT
        (SELECT pubkey FROM UNNEST(accounts) WHERE signer = TRUE LIMIT 1) AS wallet_address,
        COUNT(signature) AS tx_frequency,
        AVG(fee) AS avg_gas_fee,
        AVG(ARRAY_LENGTH(accounts)) AS avg_interacted_accounts,
        VARIANCE(fee) AS fee_variance
    FROM
        `bigquery-public-data.crypto_solana_mainnet_us.Transactions`
    WHERE
        block_timestamp >= TIMESTAMP("2026-03-01")
        AND block_timestamp < TIMESTAMP("2026-03-02")
    GROUP BY
        wallet_address
    HAVING
        tx_frequency > 5
)
SELECT
    wallet_address,
    (tx_frequency - MIN(tx_frequency) OVER()) / (MAX(tx_frequency) OVER() - MIN(tx_frequency) OVER
()) AS norm_frequency,
    (avg_gas_fee - MIN(avg_gas_fee) OVER()) / (MAX(avg_gas_fee) OVER() - MIN(avg_gas_fee) OVER()) AS
norm_fee,
    (avg_interacted_accounts - MIN(avg_interacted_accounts) OVER()) / (MAX(avg_interacted_accounts)
OVER() - MIN(avg_interacted_accounts) OVER()) AS norm_interacted,
    (fee_variance - MIN(fee_variance) OVER()) / (MAX(fee_variance) OVER() - MIN(fee_variance) OVER
()) AS norm_fee_variance
FROM RawData
LIMIT 1000
```

Fig. 1: SQL Query Code

In the `RawData` subquery, transactions are grouped by wallet identified from the signer account (`signer`), then for each wallet four key metrics are calculated: `tx_frequency` as the total number of transactions, `avg_gas_fee` as the average gas fee in lampords, `avg_interacted_accounts` as the average number of accounts involved per transaction obtained from the length of the `accounts` array, and `fee_variance` which measures the consistency of fees paid through the variance of gas fees. The filter `HAVING tx_frequency > 5` is applied to eliminate wallets with very low activity that could potentially be noise. After that, in the main query, the four metrics are normalized using the Min-Max method via the window function (`OVER()`), so that each feature is in the range [0,1] where the value 0 represents the minimum value of the dataset and the value 1 represents the maximum value of the dataset. The results of this normalization produce four main features that serve as input for the next stage, namely `norm_frequency` (normalized transaction frequency), `norm_fee` (normalized average gas fee), `norm_interacted` (normalized average number of interacted accounts), and `norm_fee_variance` (normalized gas fee variance). The use of `LIMIT 1000` is only to limit the number of rows in the methodology development and can be removed in the full implementation.

3.3. Data Preprocessing with Python

The extracted data is saved as a CSV file. All subsequent processing is done using Python with the pandas, numpy, scikitlearn, matplotlib, and seaborn libraries.

3.3.1. Creating Data and Features

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_samples, silhouette_score

df = pd.read_csv('bquxjob_793bb0ff_19cf4cf739c.csv')
features_raw = ['norm_frequency', 'norm_fee', 'norm_interacted', 'norm_fee_variance']
```

Fig. 2: Data Loading Code

3.3.2. Logarithmic Transform (Log 1p)

Blockchain data generally has a heavy-tailed distribution (some wallets are highly active, others are less active). This results in some extreme values that can dominate the distance calculation in KMeans. The logarithmic transformation log(1+x) is used to compress the range of values and make the distribution more symmetric, without losing relative information.

Formula: $x' = ln(1 + x)$ (1)

Definition 3.1 : This formula of a logarithmic transfrom
The value 0 remains 0 after the transformation, so it is safe for data that might have a value of 0.

3.3.3. Standardization with StandardScaler

KMeans uses Euclidean distance, which is highly sensitive to feature scale. Although the data has been normalized to [0,1] in BigQuery, after log transformation, the scales between features can differ. Therefore, standardization (Zscore normalization) is performed so that each feature has a mean of 0 and a standard deviation of 1.

Formula:$Z = \frac{x^1 - u}{\sigma}$ (2)

Definition 3.2 : This formula of a Standardization with StandardScaler

3.4. Determining the Optimal Number of Clusters

To determine the most appropriate K value, experiments were conducted for K = 2 to 10 with two criteria: Elbow Method (using WCSS) and Silhouette Score.

3.4.1. K Iteration Code

```
wcss = []
sil_scores = []
k_range = range(2, 11)

for k in k_range:
    km = KMeans(n_clusters=k, random_state=42, n_init=100)
    km_labels = km.fit_predict(X_scaled)
    wcss.append(km.inertia_)
    sil_scores.append(silhouette_score(X_scaled, km_labels))
```

Fig 3: Iteration Code

In the process of determining the optimal number of clusters, two main metrics are used to evaluate the clustering results: inertia_ (WCSS) and silhouette_score. Inertia_, or within-cluster sum of squares, measures the sum of the squares of the distances between each data point and the centroid of its cluster, so a smaller value indicates that the points in a cluster are more compact and closer to its center. However, the inertia_ value will always decrease as the number of clusters K increases, because more clusters mean each cluster can have fewer members and be closer to its respective centroid. Therefore, the elbow method is used to find the point where the decrease in inertia_ begins to plateau, indicating that additional clusters no longer provide a significant increase in internal compactness. Meanwhile, silhouette_score measures how similar a point is to its own cluster compared to its nearest neighboring cluster, with a value ranging from -1 to 1. Values closer to 1 indicate that the point is in the right cluster and is well separated from other clusters, while values closer to -1 indicate a possible clustering error. By combining these two metrics, researchers can obtain a more comprehensive picture of the data structure and determine the most appropriate number of clusters, both in terms of internal compactness and the quality of separation between clusters**.**

3.4.2. Visualization and Interpretation

```
optimal_k_silhouette = list(k_range)[np.argmax(sil_scores)]
max_silhouette_score = np.max(sil_scores)

print(f"K terbaik berdasarkan Silhouette Score: {optimal_k_silhouette} (score: {max_silhouette_scor
e:.4f})")
```

Fig. 4: Visualization Code

Based on the evaluation results, the Elbow method shows a point at K=4, indicating that adding clusters after that point no longer provides a significant decrease in WCSS, while the Silhouette Score reaches its highest value at K=3, indicating the most optimal cluster quality. Since both metrics provide different indications of the number of clusters, this study decided to run two clustering scenarios, namely K=4, which follows the data structure according to Elbow, and K=3, which follows the cluster quality according to Silhouette. The two scenarios will then be compared to determine which is more meaningful in the context of bot detection and Sybil attacks. The combined Elbow and Silhouette plots are constructed as follows:

```
fig, ax1 = plt.subplots(figsize=(12,6))
ax1.set_xlabel('Jumlah Cluster (K)')
ax1.set_ylabel('WCSS (Inertia)', color='tab:blue')
ax1.plot(k_range, wcss, marker='o', color='tab:blue', label='WCSS (Elbow)')
ax1.tick_params(axis='y', labelcolor='tab:blue')
ax1.axvline(x=4, color='red', linestyle='--', label='K=4 (Elbow)')

ax2 = ax1.twinx()
ax2.set_ylabel('Silhouette Score', color='tab:green')
ax2.plot(k_range, sil_scores, marker='s', linestyle='--', color='tab:green', label='Silhouette')
ax2.tick_params(axis='y', labelcolor='tab:green')
ax2.axvline(x=3, color='blue', linestyle='--', label='K=3 (Silhouette)')

plt.title('Validasi Jumlah Cluster')
fig.tight_layout()
ax1.legend(loc='upper left')
ax2.legend(loc='upper right')
plt.grid(alpha=0.3)
plt.show()
```

Fig. 5: Elbow and Silhouette Code

3.5. Implementation of K-Means Clustering

Once the optimal K was determined, the KMeans algorithm was run for both scenarios. The random_state=42 parameter was used to ensure reproducibility, and n_init=100 to avoid local minima.

3.5.1. Scenario A (K=4)

```
kmeans_k4 = KMeans(n_clusters=4, random_state=42, n_init=100)
df['cluster_k4'] = kmeans_k4.fit_predict(X_scaled)

# Hitung silhouette score per wallet dan rata-rata
df['silhouette_k4'] = silhouette_samples(X_scaled, df['cluster_k4'])
silhouette_avg_k4 = silhouette_score(X_scaled, df['cluster_k4'])

# Simpan centroid dalam ruang terstandarisasi
centroids_k4_scaled = kmeans_k4.cluster_centers_
```

Fig. 6: Code A(K=4)

3.5.2. Scenario B (K=3)

```
kmeans_k3 = KMeans(n_clusters=3, random_state=42, n_init=100)
df['cluster_k3'] = kmeans_k3.fit_predict(X_scaled)

df['silhouette_k3'] = silhouette_samples(X_scaled, df['cluster_k3'])
silhouette_avg_k3 = silhouette_score(X_scaled, df['cluster_k3'])

centroids_k3_scaled = kmeans_k3.cluster_centers_
```

Fig. 7: Code B (K=3)

3.5.3. Convergence Information

```
print(f"Iterasi hingga konvergen (K=4): {kmeans_k4.n_iter_}")
print(f"Inertia akhir (K=4): {kmeans_k4.inertia_:.2f}")
print(f"Iterasi hingga konvergen (K=3): {kmeans_k3.n_iter_}")
print(f"Inertia akhir (K=3): {kmeans_k3.inertia_:.2f}")
```

Fig. 8: Convergence Information Code

3.6. Cluster Interpretation and Labeling

After the clustering process is complete, the next step is to interpret the results by labeling each cluster based on its dominant characteristics. This begins by returning the centroid values to their original scale through an inverse transformation, so that the feature values can be understood in a more intuitive context. Then, the average of each feature per cluster is calculated to determine the profile of each group. Labeling is done with tiered priorities: the cluster with the highest `norm_fee` value is identified as Whale or High-Value Investor, representing large-value transactions. Of the remaining clusters, the group with the highest `norm_frequency` is labeled Systemic Bot or Spam due to its very high transaction frequency. Furthermore, of the remaining clusters, the cluster with the highest `norm_interacted` and above the global average is categorized as Sybil or Airdrop Farmer, characterized by numerous interactions with different accounts. The remaining cluster after these three priorities is then labeled as Retail or Human, representing normal users.

3.6.1. Centroid Back Transformation

```
# Untuk K=4
centroids_k4_original = scaler.inverse_transform(centroids_k4_scaled)
centroids_k4_original = np.expm1(centroids_k4_original)  # inverse log1p
centers_k4 = pd.DataFrame(centroids_k4_original, columns=features_raw)
centers_k4.index.name = 'Cluster'

# Untuk K=3
centroids_k3_original = scaler.inverse_transform(centroids_k3_scaled)
centroids_k3_original = np.expm1(centroids_k3_original)
centers_k3 = pd.DataFrame(centroids_k3_original, columns=features_raw)
centers_k3.index.name = 'Cluster'
```

Fig. 9: Centroid Return Code

3.6.2. Labeling Function

```
def label_clusters(centers_df, df_ref):
    labels = {}
    sorted_by_fee = centers_df['norm_fee'].sort_values(ascending=False).index
    sorted_by_freq = centers_df['norm_frequency'].sort_values(ascending=False).index
    sorted_by_inter = centers_df['norm_interacted'].sort_values(ascending=False).index

    # 1. Whale
    whale = sorted_by_fee[0]
    labels[whale] = "Whale / High-Value"

    # 2. Bot (dari sisa)
    remaining = [c for c in centers_df.index if c not in labels]
    if remaining:
        bot = max(remaining, key=lambda c: centers_df.loc[c, 'norm_frequency'])
        labels[bot] = "Systemic Bot / Spam"

    # 3. Sybil (dari sisa, jika interaksi di atas rata-rata)
    remaining = [c for c in centers_df.index if c not in labels]
    if remaining:
        if len(remaining) == 1:
            last = remaining[0]
            if centers_df.loc[last, 'norm_interacted'] > df_ref['norm_interacted'].mean():
                labels[last] = "Sybil / Airdrop Farmer"
            else:
                labels[last] = "Retail / Human"
        else:
            sybil = max(remaining, key=lambda c: centers_df.loc[c, 'norm_interacted'])
            labels[sybil] = "Sybil / Airdrop Farmer"
            remaining.remove(sybil)
            for r in remaining:
                labels[r] = "Retail / Human"
    return labels

labels_k4 = label_clusters(centers_k4, df)
df['label_k4'] = df['cluster_k4'].map(labels_k4)

labels_k3 = label_clusters(centers_k3, df)
df['label_k3'] = df['cluster_k3'].map(labels_k3)
```

Fig. 10: Labeling Function Code

3.7. Visualization of Results

3.7.1. Cluster Distribution

A two-dimensional visualization between norm_frequency (X-axis) and norm_interacted (Y-axis) was chosen because these two features are most relevant for distinguishing bots (high frequency) and Sybils (high interaction). The dot size represents norm_fee and the color indicates the cluster label.

```
plt.figure(figsize=(14,8))
scatter = sns.scatterplot(
    data=df,
    x='norm_frequency',
    y='norm_interacted',
    hue='label_k4',
    size='norm_fee',
    sizes=(50,500),
    palette='Set1',
    alpha=0.7
)
plt.title('Segmentasi Wallet Solana (K=4)')
plt.xlabel('Frequency (normalized)')
plt.ylabel('Interacted Accounts (normalized)')
plt.legend(bbox_to_anchor=(1.05,1), loc='upper left')
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```

Fig. 11: Cluster Visualization Code

3.7.2. Member Distribution per Label

```
print("\n=== Distribusi Anggota K=4 ===")
counts_k4 = df['label_k4'].value_counts()
for label, cnt in counts_k4.items():
    print(f"{label:25}: {cnt:5} wallet")
print(f"Rata-rata Silhouette K=4: {silhouette_avg_k4:.4f}")

print("\n=== Distribusi Anggota K=3 ===")
counts_k3 = df['label_k3'].value_counts()
for label, cnt in counts_k3.items():
    print(f"{label:25}: {cnt:5} wallet")
print(f"Rata-rata Silhouette K=3: {silhouette_avg_k3:.4f}")
```

Fig. 12: Member Distribution Code

3.7.3. Investigation Sample Table

```
samples_k4 = df.groupby('label_k4').apply(lambda x: x.head(5)).reset_index(drop=True)
cols_show = ['wallet_address', 'label_k4', 'norm_frequency', 'norm_interacted', 'norm_fee', 'norm_fe
e_variance', 'silhouette_k4']
print(samples_k4[cols_show].to_string(index=False))
```

Fig. 13: Sample Table Code

4. Results and Discussion

4.1. Determining the optimal number of clusters (K)

To determine the optimal number of clusters (K) in this study so that the clusters are meaningful and robust, two methods are used, namely the Elbow Method and the Silhouette score.

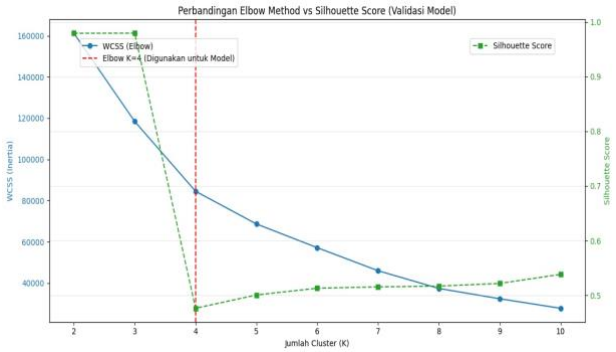

Fig. 14: Comparison Chart

Based on the elbow method in the graph above, the results of the K = 4 fault point were obtained. This result was obtained because the program identified the most significant decrease in WCSS values and began to level off after K = 4. This shows that before K = 4, the addition of new clusters, for example at K = 2 and K = 3, showed a fairly drastic decrease in WCSS values, indicating that the data could be grouped better. However, after K = 4, the addition of new clusters such as K = 5, K = 6, and so on showed that the decrease in WCSS values began to decrease or tended to level off, indicating that most of the variation had been explained by K = 4 and adding more clusters than that would only create noise or be uninformative.
Therefore, based on the elbow method, the optimal K=4 result is the most informative for K-means clustering, as it provides the most informative grouping without too many clusters, which can introduce noise. However, the Silhouette score method yields different results than the Elbow Method, which determined the optimal K at K=3.

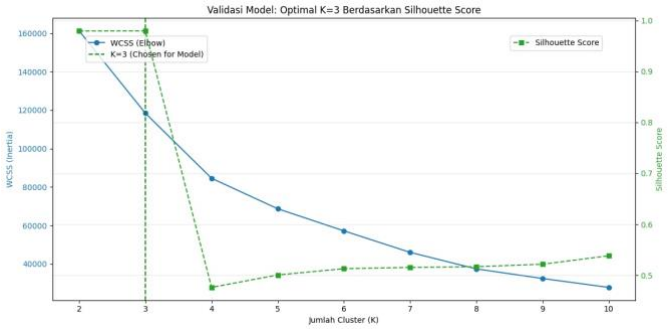

Fig. 15: Optimization Graph K=3

Based on the Silhouette score in the graph above, the most optimal (K) is determined at K = 3. Based on the program run, K = 3 is the most optimal because it has the highest Silhouette score value, namely 0.9795, almost close to 1, which indicates a near-perfect separation between each cluster. In other words, K = 3 has the least noise.

4.2. Cluster Identification

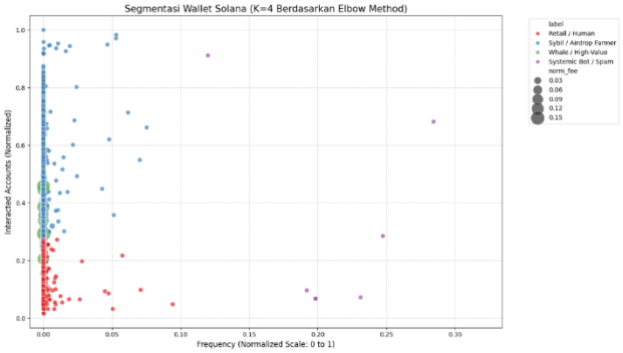

Fig. 16: K=4 Segmentation Graph

KONVERGENSI CENTROID (dalam ruang fitur terukur)

Cluster ID	norm_frequency	norm_fee	norm_interacted	norm_fee_variance
0	-0.009370	-0.145129	-0.910552	-0.055718
1	-0.010533	0.087361	0.665994	0.018266
2	-0.000497	39.538575	0.223099	47.217950
3	78.204101	-0.171420	-0.176691	-0.019580

Catatan: Centroid di atas berada di ruang fitur yang telah di-scaling (StandardScaler applied pada logip data).

Fig. 17: Centroid Convergence Table K=4

Based on the data in the table above, it displays the final centeroid value (convergence) for each cluster (c0, c1, c2, c3) which is then used in this study to identify each cluster based on the pattern and characteristics of its centeroid data.
Cluster (c0) is identified as human/retail because its centeroid value is low and tends to approach 0. This indicates random transaction activity, low costs and limited interaction with other accounts which are characteristics of humans/retail who make moderate transactions.
Cluster 1 (c1) was identified as a Sybil attack because its centeroid value showed an extreme value compared to the other values. Norm_interacted (0.665994) indicates interactions with many different accounts, but with a low transaction frequency. This is one of the characteristics of a Sybil attack, which involves transactions with many accounts to obtain rewards in a campaign or activity.
Cluster 2 (c2) is identified as a whale/high value because its centeroid values show extreme values in norm_fee and norm_fee_variance, indicating high transaction costs with high variance. This indicates the characteristics of institutions/whales that consistently conduct high-cost transactions with low frequency. Cluster 3 (c3) is identified as a systemic bot/spam because its centeroid values show extreme values in norm_frequency, indicating high transaction frequency but with low norm_fee or fees in a short period of time with the aim of flooding the network, which is a characteristic of bots/spam.

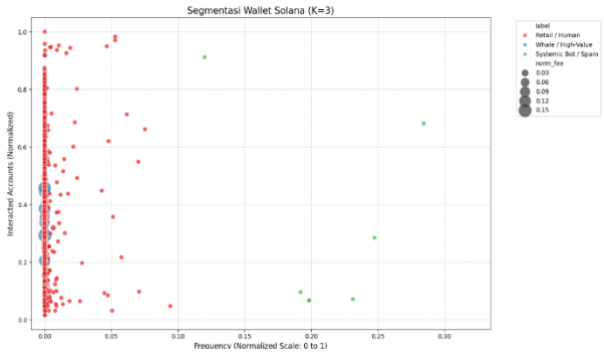

Fig. 18: Segmentation Graph K=3

```
============================================================
          KONVERGENSI CENTROID (dalam ruang fitur terukur)
============================================================
            norm_frequency  norm_fee  norm_interacted  norm_fee_variance
Cluster ID
0              -0.010041 -0.010857        -0.000039         -0.012989
1              -0.000497 39.538575         0.223099         47.217950
2              78.204101 -0.171420        -0.176691         -0.019580
============================================================
Catatan: Centroid di atas berada di ruang fitur yang telah di-scaling (StandardScaler applied pada log1p data).
```

Fig. 19: Centroid Convergence Table K=3

Based on the data in the table above, it displays the final centroid value (convergence) for each cluster (c0, c1, c2) which is then used in this study to identify each cluster based on the pattern and characteristics of its centeroid data.
Cluster 0 (c0) is identified as a human/retail cluster because its centeroid value is close to 0 or lacks large extreme values. This indicates that members of this cluster have low transaction activity, minimal fees, and only interact with a few accounts.
Cluster 1 (c1) is identified as a whale/high-value cluster because its centeroid values are extreme or tend to be larger than other clusters in the norm_fee and norm_fee_variance values, indicating high transaction costs and significant variations in transaction costs. This indicates trading activity or the transfer of high-value assets typically carried out by institutions/whales.
Cluster 2 (c2) is identified as a systemic bot / spam cluster because its centeroid value shows an extreme value in the norm_frequency value compared to other values and norm_frequency in other clusters. This indicates that members of the cluster carry out transactions with high frequency, automatically, and consistently with the aim of flooding the network or carrying out automated schemes with minimal costs which are characteristics of bots, this is because humans have physical and mental limits so it is impossible to carry out so many transaction activities in a short time.

4.3. Comparison of K-Means Silhoutte Score (K=3) vs Elbow Method (K=4)

```
================================================
  DISTRIBUSI ANGGOTA KLASTER (Berdasarkan Label)
================================================
Sybil / Airdrop Farmer              :      31467 wallet
Retail / Human                      :      23049 wallet
Whale / High-Value                  :         15 wallet
Systemic Bot / Spam                 :          7 wallet
------------------------------------------------
TOTAL RATA-RATA SILHOUETTE (untuk K=4):     0.4763
================================================
```

Fig. 20: Silhoutte Table K=3

```
================================================
  DISTRIBUSI ANGGOTA KLASTER (Berdasarkan Label)
================================================
Retail / Human                      :      54516 wallet
Whale / High-Value                  :         15 wallet
Systemic Bot / Spam                 :          7 wallet
------------------------------------------------
TOTAL RATA-RATA SILHOUETTE          :     0.9795
================================================
```

Fig.21: Elbow Method K=4

Based on the comparison of the two tables above. Whale and bot clusters are clearly identified in both tables with 15 whales and 7 Systematic bots / spam. While the main difference between the two methods based on the data in the table is in human / retail and Sybil attacks, where in the Elbow Method (K = 4) human and Sybil clusters are separated based on their norm_interacted. Where Sybil has a high norm_ingeracted value. However, when viewed from the Silhouette score value, K = 4 (0.4763) is lower than K = 3 (0.9795) which shows that the separation of Sybil from humans by K = 4 is not as clear as the separation made by K = 3 into only 3 clusters of humans, whales and bots. However, this is understandable because Sybil attacks are designed to imitate humans so that Sybil attacks tend to be similar in transaction patterns to humans / retail.

4.4. Validation with Wallet Sample

TABEL INVESTIGASI (4D + SILHOUETTE SCORE) - MODEL K=3

wallet_address	label	norm_frequency	norm_interacted	norm_fee	norm_fee_variance	silhouette_score
nWU4yRJTWTHsPPGH7fDgHEwW5zTa32j	Retail / Human	0.000001	0.095662	0.000010	0.000001	0.970271
.6AhPqvE9v562vBUs3JLZ19XoVFpkvD	Retail / Human	0.000002	0.408157	0.000053	0.000011	0.981703
.PEcsHNr6yZ53F3PvBZTQ86PBJcdxca	Retail / Human	0.000003	0.331273	0.000003	0.000001	0.986290
:UWjBRZ4GP7Aciv84ZrTGuvSvnmkjMx	Retail / Human	0.000001	0.338359	0.000275	0.000169	0.985756
'Cjx4Ze18bgoZCXrZMgvzYzVfU8YKCG	Retail / Human	0.000002	0.284993	0.000068	0.000006	0.986173
Kau1vP4jXrkqtN8Dmh8yuEqyn6GMYh	Systemic Bot / Spam	0.119947	0.911378	0.000007	0.000004	0.202536
Rc4hdb2ZKq2JmWHD8UKJp7guEdyT9j	Systemic Bot / Spam	0.198504	0.067597	0.000001	0.000003	0.804343
:FrMVivXRW5eMDVYADnZgrJpPC3fMp3	Systemic Bot / Spam	0.231233	0.071899	0.000004	0.000006	0.807569
8oz26AgK8x8m5ePsdeRawjqvojbjnQ	Systemic Bot / Spam	0.192013	0.096261	0.000003	0.000001	0.783385
tXTb79PJ99bbxkTipgo3GJ4EJZ1YpB	Systemic Bot / Spam	0.284343	0.681757	0.000003	0.000011	0.715733
oTq2WV8AqpZsD9vfH6kGKhHMfx2xie	Whale / High-Value	0.000019	0.443856	0.113335	0.018633	0.158834
YgD5sCmLDriawJxQPp6TG1PKci8wMS	Whale / High-Value	0.000010	0.286986	0.066115	0.109982	0.481756
SVFNxPBb9RoqroLWi6kgCN6qTCL4LL	Whale / High-Value	0.000003	0.338173	0.096602	0.023319	0.143412
.wycXApooMDnXrxgJ464HuJFkGSzqAj	Whale / High-Value	0.000434	0.297424	0.046668	0.051187	0.134007
f3oS4J7mrwrFc42AzW1Zr7ULH6tcr	Whale / High-Value	0.000006	0.339165	0.053048	0.055777	0.262788

Fig. 22: Swallow Sample Table K=3

For further verification (K=3), wallet samples were taken for each cluster, as shown in the table above. The human/retail cluster showed a high Silhouette score, indicating near-perfect separation. Although the whale, bot, and Sybil clusters tended to vary, overall they were still positive, indicating fairly good separation, although some members fell into the "grey area" and overlapped with characteristics of other clusters. However, their extreme scores provided a clear separation from the other clusters.

TABEL INVESTIGASI (4D + SILHOUETTE SCORE) - MODEL K=4 (Elbow Method)

wallet_address	label	norm_frequency	norm_interacted	norm_fee	norm_fee_variance	silhouette_score
9uh5WBg2FfpEjaWU4yRJTWTHsPPGH7fDgHEwW5zTa32j	Retail / Human	0.000001	0.095662	0.000010	0.000001	0.619659
CdnSy3FDg1VnMVCjx4Ze18bgoZCXrZMgvzYzVfU8YKCG	Retail / Human	0.000002	0.284993	0.000068	0.000006	0.135172
7BSoZE1SDccNDLSsznhS8fhSV8vEUhEC9pwEndRY7iUf	Retail / Human	0.000002	0.195575	0.000001	0.000001	0.630936
AdZ8jgSQPpbFiaNM2i41X2eQAkKXsZMEZhdoN1q4SkzW	Retail / Human	0.000005	0.239745	0.000035	0.000013	0.539377
FvtXk299NR3FhtkpT1VjtxSPGK8WLetzN86VyP25qLER	Retail / Human	0.000001	0.095662	0.000011	0.000001	0.619658
5Ykixs9iquKQx16AhPqvE9v562vBUs3JLZ19XoVFpkvD	Sybil / Airdrop Farmer	0.000002	0.408157	0.000053	0.000011	0.594784
2mQ37Z4wZ57VP1PEcsHNr6yZ53F3PvBZTQ86PBJcdxca	Sybil / Airdrop Farmer	0.000003	0.331273	0.000003	0.000001	0.449822
4tDEyR1CZkEiBRUWjBRZ4GP7Aciv84ZrTGuvSvnmkjMx	Sybil / Airdrop Farmer	0.000001	0.338359	0.000275	0.000169	0.474274
GJsg4hkEvQmnj5SE9dqX5vCHGwzD31QQdkLb9VSMerj3	Sybil / Airdrop Farmer	0.000002	0.329911	0.000004	0.000001	0.440033
2jDUUkMh71peFZSDFCAEBPs6M4NMjPrrzJ5WUrbAGiSD	Sybil / Airdrop Farmer	0.000001	0.429150	0.000132	0.000014	0.589415
zeroDbresTUpHKau1vP4jXrkqtN8Dmh8yuEqyn6GMYh	Systemic Bot / Spam	0.119947	0.911378	0.000007	0.000004	0.201565
4KSLE7EU1P7PQ8Rc4hdb2ZKq2JmWHD8UKJp7guEdyT9j	Systemic Bot / Spam	0.198504	0.067597	0.000001	0.000003	0.804267
H5cATk3vp37wJ2FrMVivXRW5eMDVYADnZgrJpPC3fMp3	Systemic Bot / Spam	0.231233	0.071899	0.000004	0.000006	0.807514
7rhxnLV8C77o6d8oz26AgK8x8m5ePsdeRawjqvojbjnQ	Systemic Bot / Spam	0.192013	0.096261	0.000003	0.000001	0.783307
roUteHjDohtkatXTb79PJ99bbxkTipgo3GJ4EJZ1YpB	Systemic Bot / Spam	0.284343	0.681757	0.000003	0.000011	0.715687
4A4oxAcW4pE9XaoTq2WV8AqpZsD9vfH6kGKhHMfx2xie	Whale / High-Value	0.000019	0.443856	0.113335	0.018633	0.157097
EfdtcUmuYjHQBDYgD5sCmLDriawJxQPp6TG1PKci8wMS	Whale / High-Value	0.000010	0.286986	0.066115	0.109982	0.481253
9FvQzySbDqiMzXSVFNxPBb9RoqroLWi6kgCN6qTCL4LL	Whale / High-Value	0.000003	0.338173	0.096602	0.023319	0.141541
6s8Z55Qtdks3EiwycXApooMDnXrxgJ464HuJFkGSzqAj	Whale / High-Value	0.000434	0.297424	0.046668	0.051187	0.132141
B8A6B5GdVf3c4bf3oS4J7mrwrFc42AzW1Zr7ULH6tcr	Whale / High-Value	0.000006	0.339165	0.053048	0.055777	0.261238

Fig. 23: Swallow Sample Table K=4)

For further proof (K = 4), wallet samples were taken for each cluster as shown in the table above. Where for the human / retail cluster at K = 4 shows a lower Silhouette score compared to K = 3 but still in the good or reasonable category with an average sample wallet showing a value of 0.5 and above although some enter the 'grey area'. Likewise with other clusters, Sybil bots, spam bots, and whales. With spam bots showing a high Silhouette score (0.804267) helping to identify them as bots with high transaction frequency but there are also low ones (0.201565) as well as whales that tend to have low Silhouette scores. This is due to the large overlap of characteristics between clusters but the presence of extreme values at certain values helps identify members of each cluster such as whales with high norm_fee and norm_fee_variance and norm_frequency in spam bots.

4.5. Discussion Results

Anomaly detection: in both K-means (K=3) and (K=4), it was found that the Systematic bot/spam cluster and the whale/high value cluster had the same grouping in terms of the number of members. This shows that their extreme values of norm_frequency for bots and norm_fee and norm_fee_variance for whales have a very large influence so that they can be identified by each cluster. This can be used to detect anomalies in the blockchain to detect fraud, market manipulation, or other malicious activities.
Risk management: by being able to identify whales, it can be used to assess changes in asset concentration and changes in market direction.
Network management: by identifying bots, network owners can implement preparatory measures to maintain the ecosystem and performance of the Blockchain network.

5. Closing

5.1. Conclusion

This study has identified and grouped wallet user behavior in Solan Blockchain using the K-means Clustering method using the Elbow Method and Silhouette score validation methods, each of which provides an optimal number of clusters of K = 4 for the Elbow Method and K = 3 for the Silhouette score. With each providing certain advantages. Where K = 3 for the Silhouette score has a value (0.9795) which is close to 1 indicating almost perfect clustering and forms a cluster that is denser internally and well separated from other clusters. While K = 4 for the Elbow Method has a Silhouette score value (0.4763) lower than K = 3 and is included in the weak structure, but the advantage of K = 4 is that it identifies Sybil attacks which indeed imitate human wallet transaction patterns. Simply put, K = 3 provides a more structured and informative clustering while K = 4 provides a more varied clustering.

5.2. Suggestion

For future research:

1. Additional features: adding new features such as asset ownership period, transaction patterns with certain smart contracts and fund flows between wallets that can help identify Sybil attacks.
2. Using other Clustering algorithms: Try using other algorithms such as DBSCAN or Hierarchical Clustering to see if it is possible to get different clustering results.
3. Comparing with other Blockchains: conducting the same research on other Blockchains to see the differences and similarities in user behavior patterns.
4. Temporal Cluster Change Analysis: Examines whether cluster characteristics change over time, such as whether wallets move from one cluster to another. For example, human/retail wallets may become whales or bots at a certain time.

References

[1] Hartawan, M. S., dkk. (2022). Big Data (Informasi dan Kasus). Kun Fayakun.
[2] Kurniawan, S. D., dkk. (2024). Big Data: Mengenal Big Data & Implementasinya di Berbagai Bidang. Sonpedia Publishing Indonesia.
[3] Maulani, G., dkk. (2024). Penerapan Data Mining di Berbagai Bidang. HEI Publishing.
[4] Wibowo, A. (2025). Pengantar AI, Big Data dan Ilmu Data. Yayasan Prima Agus Teknik.
[5] Yulia, & Silalahi, M. (2021). Penerapan Data Mining Clustering Dalam Mengelompokan Buku Dengan Metode K-Means. Indonesian Journal of Computer Science.
[6] Singh, S., & Chander, S. (2024). Prevention of Sybil Attack on Blockchain to Ensure Security of Wireless Sensor Network. International Journal of Intelligent Systems and Applications in Engineering, 12(8s), 14–24.
[7] Hsiao, S.-J., et al. (2024). Enhancing the Security and Reliability of Wireless Sensor Networks Using Blockchain Technology. International Journal of Intelligent Systems and Applications in Engineering (IJISAE), 12(8s), 14–24.
[8] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from www.bitcoin.org
[9] Drescher, D. (2017). Blockchain basics: A non-technical introduction in 25 steps. New York: Apress
[10] Rousseeuw, P. J. (1987). Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. Journal of Computational and Applied Mathematics, 20, 53–65.
[11] Leskovec, J., Rajaraman, A., & Ullman J. D. (2020). Mining of Massive Datasets (3rded.). Cambridge University Press.
[12] Pham, T., & Lee, S. (2016). Anomaly Detection in Bitcoin Network Using Unsupervised Learning Methods.
[13] Amelia, V. R., Pratama, P., Ramadhan, M. A. N., Lampang, M. A., Pratama, M. H., Mentari, T., & Widyaningsih, D. S. (2024). A systematic literature review of blockchain-based triple-entry accounting in crypto assets. Jurnal Bisnis Mahasiswa, 4(4).
[14] Pradana, I. G. M. T., Djatna, T., Hermadi, I., & Yuliashih, I. (2024). Model of integrated assessment layer for implementation readiness of blockchain-based traceability system. Jurnal Teknologi Industri Pertanian, 34(2), 127–139.
[15] Raghav, N., & Bhola, A. K. (2023). Detecting Sybil attack in blockchain and preventing through universal unique identifier in health care sector for privacy preservation. International Journal on Recent and Innovation Trends in Computing and Communication, 11(8).
[16] Supriyanto, D., Desembrianita, E., Prihadi, D. J., Suryaningrum, D. A., & Alhakim, B. A. (2025). The urgency of blockchain in developing the tourism industry in Indonesia. Jurnal Teknologi dan Manajemen Industri Terapan (JTMIT), 4(1), 75–80.