Journal of Information Technology, computer science and Electrical Engineering (JITCSE)
Vol. 2, No. 1, February 2025, pp. 189~197
ISSN: 3046-4900

Data Mining: Building The Automatic Pipeline System for Clustering the Compliance Level of PBB-P2 Taxpayers

Sugeng Pranoto[1], Sri Wahyuni[2]
[1]Master of Information Technology Student, Universitas Pembangunan Panca Budi Medan

[2]Lecturer of Master of Information Technology, Universitas Pembangunan Panca Budi Medan

ABSTRACT

Taxpayer compliance analysis is crucial in regional revenue management, especially for Rural and Urban Land and Building Tax (PBB-P2). The large volume of taxpayer data poses a significant challenge in manual processing, so an automated and efficient approach is needed. This research develops a data mining pipeline to cluster the level of taxpayer compliance in the Badan Pengelola Keuangan dan Pendapatan Daerah (BPKPD) of Tebing Tinggi City. The proposed pipeline is implemented using the Java programming language and the SMILE library and includes three main procedures, namely Extraction, which is in charge of retrieving receivables and payment data from the SISMIOP database; Transformation, which is in charge of processing the extracted data to generate new insights through K-Means clustering, and Load is in charge of storing the transformation results into the MySQL database for further analysis and reporting. This pipeline is run every hour to ensure that data processing is carried out in real-time. By utilizing this automated system, this study aims to increase understanding of taxpayer compliance patterns and assist local governments in designing more effective policies to increase tax revenues and taxpayer compliance levels.

Keyword : Data Mining; **Pipeline**; **K-Means; Clustering; PBB-P2**

Corresponding Author:
Sugeng Pranoto,
Master of Information Technology Student
Universitas Pemmbangunan Panca Budi Medan
Jl. Gatot Subroto , 20222, Indonesia.
Email : stu.genk@gmai.com

Article history:
Received Jan 07 2025
Revised Jan 15 2025
Accepted Jan 20 2025

1. INTRODUCTION

Rural and Urban Land and Building Tax (PBB-P2) is one of the essential sources of regional revenue for local governments to support development and public services (S.Pranoto and D.Nasution, 2024)(Putera, Siahaan, Jabar, et al., 2024)(Putera, Siahaan, Sutiono, et al., 2024)(Sitorus et al., 2024). However, managing and analyzing taxpayer compliance in PBB-P2 payments is still a challenge, mainly due to the large volume of data and the complexity of determining the level of taxpayer compliance. The manual analysis process is not only time-consuming but also prone to human error, so a solution based on data mining technology that can process large-scale data efficiently and accurately is needed (Wahyuni, 2018)(Wahyuni & Marbun, 2020)(Wahyuni et al., 2019).

In this study, a data mining pipeline was developed that aims to optimize (Wahyuni et al., 2022)(Novelan et al., 2023)(Gallego et al., 2024) the clustering of the compliance level of PBB-P2 taxpayers in Badan Pengelola Keuangan dan Pendapatan Daerah (BPKPD) Tebing Tinggi City. These pipelines are designed to automate the data extraction, transformation, and loading (ETL) process (Raj et al., 2020)(Agostinelli et al., 2023) using the Java programming language and the SMILE library. The clustering method used in this process is the K-Means algorithm because it is fast and easy (Novelan et al., 2023)(Putra et al., 2021)(Zhao et al., 2018)(Iqbal et al., 2024). This algorithm allows the grouping of taxpayers based on their compliance patterns in paying taxes.

The pipeline developed consists of three main procedures, one of which is Extraction, which retrieves receivables and taxpayer payment data from the SISMIOP application based on the Oracle database. The transformation functions process the extracted data to produce new information in the form of clusters of taxpayer compliance levels using the K-Means algorithm. Load functions to enter the transformation results into the MySQL database for analysis and reporting purposes.

By building an automated and scheduled pipeline system, this study can provide an efficient solution in analyzing the level of taxpayer compliance in real-time and providing data-based recommendations to improve tax management performance at BPKPD Tebing Tinggi City.

2. RESEARCH METHOD

A. Research Stages

his study uses a descriptive and experimental approach that focuses on developing and implementing application solutions (Sri Wahyuni et al., 2023) to building automatic pipeline system. The stages of the research are explained as follows:

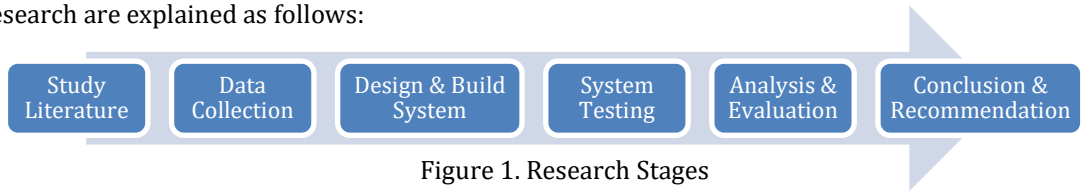

Figure 1. Research Stages

Table 1. Explanation of the Research Flow

Stage	Method/Tool	Output
Study Literatute	Conventional and online literature	This stage seeks references and theories about PBB-P2, Pipeline , Java Libraries using SMILE in clustering, and other supporting theories. In this study, it was carried out at BPKPD Tebing Tinggi City.
Data Collection	Collect PBB-P2 Receivables Data in Oracle Database using SQL Developer, and create a result table from clustering in MySQL with MySQL Administrator.	Generate Data View normalization of receivables and payment data. Datasets for clustering by having the fields of Arrears Frequency, Delinquency Rate, and Number of Days of Delay.
Design and Build Systems	Designing the flow of the system. Using the Java programming language to create an Automatic Pipeline System	The result of this stage is an application that clusters the compliance level of PBB-P2 taxpayers with the K-Means algorithm
System Testing	Test the app already built.	The results of clustering stored in the MySQL database
Analysis & Evaluation	Analyze and evaluate clustering results using the K-Means algorithm.	Knowing the centroid of each cluster at the level of compliance (compliant, less compliant, and non-compliant) and calculating the percentage of data for each cluster. Calculate the Silhouette Score to measure how well the data in each clustered cluster is performing.
Conclusion & Recommendation	Compile conclusions of research results and recommendations for system development.	Conclusion on improving the auto pipeline system from Java with SMILE library in clustering the compliance level of PBB-P2 taxpayers and suggestions in advanced system development.

B. Data Collection

BPKPD Tebing Tinggi City in managing PBB-P2 tax data using the Tax Object Information System (SISMIOP) application based on the Oracle database. The data needed to create this system is receivables data with a field for the tax object number, taxpayer name, sub-district, sub-district, tax year, amount of receivables, due date, payment status, and payment date if it has been paid. These fields are then made a dataset that is needed for grouping the compliance level of PBB-P2 taxpayers with variable tax object

numbers, taxpayer names, sub-districts, sub-districts, total arrears, frequency of arrears, arrears ratio and number of days in arrears. This dataset is created by creating a data view with the name DS_PIUTANG, which has the following structure:

Table 1. Structure View Dataset DS_PIUTANG

Colum Name	Data Type	Description
NOP	CHAR(18)	Tax Object Number
NM_WP	VARCHAR2(30)	Taxpayer Name
NM_KECAMATAN	VARCHAR2(30)	Name of District
NM_KELURAHAN	VARCHAR2(30)	Name of the Village
TOTAL_TUNGGAKAN	NUMBER	Total Arrears
FREQ_TUNGGAKAN	NUMBER	Frequency of Arrears
RASIO_TUNGGAKAN	NUMBER	Arrears Ratio
TOTAL_HARI_TERLAMBAT	NUMBER	Total late payment days

Apart from the Receivables Dataset, in building this system, a dataset is also needed to accommodate the results of the clustering of PBB-P2 taxpayers' compliance levels. This dataset will later be stored using a MySQL database with the name table DS_CLUSTER, which has the following structure:

Table 1. Structure Table Dataset DS_CLUSTER

Colum Name	Data Type	Description
NOP	CHAR(18)	Tax Object Number
NM_WP	VARCHAR2(30)	Taxpayer Name
NM_KECAMATAN	VARCHAR2(30)	Name of District
NM_KELURAHAN	VARCHAR2(30)	Name of the Village
TOTAL_TUNGGAKAN	NUMBER	Total Arrears
FREQ_TUNGGAKAN	NUMBER	Frequency of Arrears
RASIO_TUNGGAKAN	NUMBER	Arrears Ratio
TOTAL_HARI_TERLAMBAT	NUMBER	Total late payment days
KATEGORI	INTEGER	Clustered product categories

3. RESULTS AND DISCUSSION

A. Design & Build System

The Automatic Pipeline System architecture for grouping the compliance level of PBB-P2 taxpayers can be described as follows:

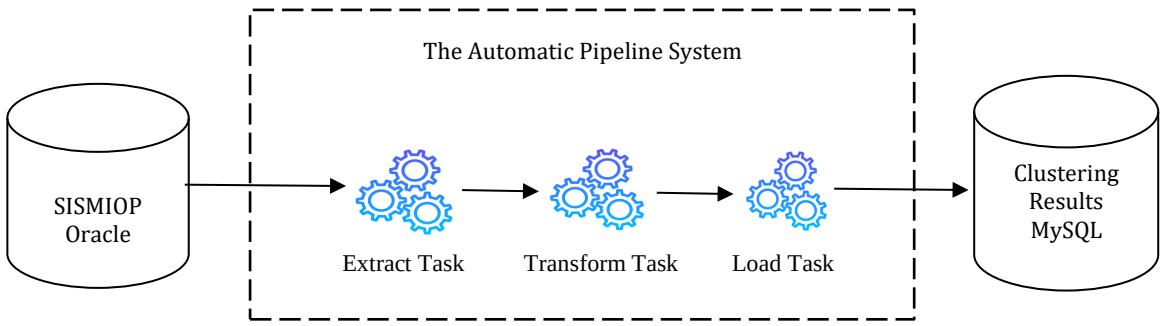

Figure 2. System Architecture

Figure 2 explains that the Automatic Pipeline System has three task functions, namely Extraction, which is in charge of retrieving receivables and payment data from the SISMIOP database;

Transformation is in charge of processing data that has been extracted to generate new insights through K-Means clustering, and Load has the task of storing the transformation results into the MySQL database The Automatic Pipeline System will be run in a batch process for 1 hour. The algorithm of the Automatic Pipeline System program is as follows:

```
BEGIN
    // Spring Scheduler configuration
    @Scheduled(fixedRate = 3600000) // Runs every 1 hour
    FUNCTION ScheduledTask()
        extracted_task <- ExtractTask()
        clustered_task, SilhouetteScore <- TransformTask(extracted_data)
        LoadTask(clustered_data)
    END FUNCTION

    // Extraction Procedure
    FUNCTION ExtractTask()
        CONNECT TO database_SISMIOP
        RETRIEVE piutang_data
        RETURN extracted_data
    END FUNCTION

    // Transformation Procedure
    FUNCTION TransformTask(extracted_data)
        CLEAN extracted_data
        APPLY K-Means Clustering on extracted_data
        clustered_data <- RESULT OF K-Means
        silhouetteScore <- CalculateSilhouetteScore(clustered_data)
        RETURN clustered_data, silhouetteScore
    END FUNCTION

    // SilhouetteScore Calculating Function
    FUNCTION CalculateSilhouetteScore(clustered_data)
       APPLY silhouetteScore on clustered_data
       RETURN silhouetteScore
    END FUNCTION

    // Load Procedure
    FUNCTION Loadtask(clustered_data)
        CONNECT TO database_MySQL
        STORE clustered_data
    END FUNCTION
END
```

Figure 3. Pseudocode Algorithm System

The next step is to build an Automatic Pipeline System program using the Java programming language and the SMILE (Statistical Machine Intelligence and Learning Engine) library. The list of programs that have been made is as follows:

```
package com.ml.task;
import com.ml.model.source.Pbb;
import com.ml.repository.source.PbbRepository;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class ExtractTask {
    private List<Pbb>  pbbList=new ArrayList<Pbb>();
    private ArrayList<double[]>dataList = new ArrayList<>();
    public ArrayList<double[]> doTask(PbbRepository pbbRepository){
       this.pbbList=pbbRepository.getAll();  // Dataset Acquisition from Oracle SISMIOP
       for (Iterator it = this.pbbList.iterator(); it.hasNext();) {
          Pbb pbb=(Pbb) it.next();
          this.dataList.add(new double[]{pbb.getFreqTunggakan(), pbb.getRasioTunggakan(),
          pbb.getTotalHariTerlambat()});
       }
        return dataList;
    }
    public List<Pbb> getPBBList(){
        return this.pbbList;
    }
}
```

Figure 4. Listing Program ExtractTask.java

```java
package com.ml.task;
import smile.clustering.KMeans;

public class TransformTask {
    private double[][] dataArray;
    private int cluster;
    public KMeans doTask(double[][] dataArray,int cluster){
        this.dataArray = dataArray;
        this.cluster = cluster;
        int maxIterasi=600;
        double tol=1e-4;
        KMeans kmeans = KMeans.fit(dataArray, cluster,maxIterasi,tol);
        return kmeans;
    }
}
```

Figure 5. Listing Program TranformTask.java

```java
package com.ml.task;
import com.ml.model.DSPbb;
import com.ml.model.source.Pbb;
import com.ml.repository.DSPbbRepository;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import smile.clustering.KMeans;

public class LoadTask {
    private DSPbbRepository dsPBBRepository;
    private List<Pbb> pbbList=new ArrayList<Pbb>();
    private KMeans kmeans;
    public void doTask( DSPbbRepository dsPBBRepository,List<Pbb> pbbList,KMeans kmeans){
        this.dsPBBRepository=dsPBBRepository;
        this.pbbList=pbbList;
        this.kmeans=kmeans;
        dsPBBRepository.deleteAll();
        int i=0;
        for (Iterator it2 = pbbList.iterator(); it2.hasNext();) {
            Pbb pbb=(Pbb) it2.next();
            DSPbb dsPBB=new DSPbb();
            dsPBB.setNop(pbb.getNop());
            dsPBB.setNama(pbb.getNama());
            dsPBB.setKelurahan(pbb.getKelurahan());
            dsPBB.setKecamatan(pbb.getKecamatan());
            dsPBB.setTotalTunggakan(pbb.getTotalTunggakan());
            dsPBB.setFreqTunggakan(pbb.getFreqTunggakan());
            dsPBB.setRasioTunggakan(pbb.getRasioTunggakan());
            dsPBB.setTotalHariTerlambat(pbb.getTotalHariTerlambat());
            dsPBB.setKategori(kmeans.y[i]);
            dsPBBRepository.save(dsPBB);
            i=i+1;
           }
    }
}
```

Figure 6. Listing Program LoadTask.java

```java
package com.ml.task;
import smile.math.distance.EuclideanDistance;

public class SilhouetteScore {
    public static double calculate(double[][] data, int[] clusterAssignments, int k) {
        EuclideanDistance distance = new EuclideanDistance();
        double[] silhouetteScores = new double[data.length];
        for (int i = 0; i < data.length; i++) {
            int currentCluster = clusterAssignments[i];
            double a = 0; // Cohesion
            double b = Double.MAX_VALUE; // Separation
            int sameClusterCount = 0;
            // Calc cohesion (a)
            for (int j = 0; j < data.length; j++) {
                if (i == j) continue; // Skip diri sendiri
                if (clusterAssignments[j] == currentCluster) {
                    a += distance.d(data[i], data[j]);
                    sameClusterCount++;
                }
            }
            a = sameClusterCount > 0 ? a / sameClusterCount : 0;
            // Calc separation (b)
            for (int cluster = 0; cluster < k; cluster++) {
                if (cluster == currentCluster) continue;
                double avgDist = 0;
                int clusterCount = 0;
                for (int j = 0; j < data.length; j++) {
                    if (clusterAssignments[j] == cluster) {
                        avgDist += distance.d(data[i], data[j]);
                        clusterCount++;
                    }
                }
                if (clusterCount > 0) {
                    avgDist /= clusterCount;
                    b = Math.min(b, avgDist);
                }
            }
            // Calc silhouette score untuk data i
            silhouetteScores[i] = (b - a) / Math.max(a, b);
        }
        // Avg Silhouette Score for All Data
        double totalScore = 0;
        for (double score : silhouetteScores) {
            totalScore += score;
        }
        return totalScore / data.length;
    }
}
```

Figure 7. Listing Program SilhouetteScore.java

```java
package com.ml.task;

import com.ml.model.source.Pbb;
import com.ml.repository.source.PbbRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.scheduling.annotation.Scheduled;
import org.springframework.stereotype.Component;
import com.ml.repository.DSPbbRepository;
import java.util.ArrayList;
import java.util.List;
import smile.clustering.KMeans;

@Component
public class ScheduledTasks {
    @Autowired
        private PbbRepository pbbRepository;
        @Autowired
        private DSPbbRepository dsPBBRepository;
        private ArrayList<double[]> dataList = new ArrayList<>();
        private List<Pbb> pbbList=new ArrayList<Pbb>();
        private double[][] dataArray;

        @Scheduled(fixedRate = 60*60*1000)
        public void PipelineTask() {

                ExtractTask extractTask=new ExtractTask();
                this.dataList=extractTask.doTask(pbbRepository);
                this.pbbList=extractTask.getPBBList();

                TransformTask transformTask=new TransformTask();
                int cluster=3;
                dataArray=dataList.toArray(new double[0][]);
                KMeans kmeans =transformTask.doTask(dataArray,cluster);

                LoadTask loadTask=new LoadTask();
                loadTask.doTask(dsPBBRepository, pbbList, kmeans);

                // Centroid cluster
                 System.out.println("\nCluster Centroids:");
                 for (int j = 0; j < kmeans.centroids.length; j++) {
                     System.out.printf("Cluster %d: %s%n", j, java.util.Arrays.toString(kmeans.centroids[j]));
                 }

                 //Silhouette Score
                 double silhouetteScore = SilhouetteScore.calculate(dataArray, kmeans.y, cluster);
                 System.out.println("Silhouette Score: " + silhouetteScore);

                 System.out.println("Finish!");
        }
}
```

Figure 8. Listing Program ScheduledTask.java

The way the program that has been built works is that the first one to be run is a SheduledTask.java program with a batch process every hour. The ScheduledTask.java program will run in stages, first ExtractTask.java, then TrasnformTask.java, then LoadTask.java. After running the ETL task program process, SheduledTask.java carries out the Silhoutte Score calculation procedure by running SilhouteScore.java.

B. System Testing

The system test was carried out using PBB-P2 receivables data at BPKPD Kota Tebing Tinggi for the tax period 2020 to 2024, with a record number of 55,359 taxpayer data. The results of the system test went well. To run a single ETL process takes approximately 80 seconds.

```
Cluster Centroids:
Cluster 0: [0.4579643676133357, 0.14478673487387547, 185.87412242017993]
Cluster 1: [4.616702355460386, 0.9219316251938272, 3875.7828398434617]
Cluster 2: [2.4164501521787543, 0.5111008833791107, 1908.483037636404]
Silhouette Score: 0.7504464864055562
```

Figure 9. Program Execution Results

C. Analysis & Evaluation

From the results of the first test, it can be concluded that the compliant taxpayer cluster is in cluster 0 with the centroid cluster at [0.4579643676133357, 0.14478673487387547, 185.87412242017993], while the non-compliant taxpayer cluster is in cluster 1 in the centroid cluster [4.616702355460386, 0.9219316251938272, 3875.7828398434617] and the less compliant cluster in cluster 2 at [2.4164501521787543, 0.5111008833791107, 1908.483037636404].

The Silhouette Score calculation measures how well the data in each clustered cluster on the K-Means algorithm is close to 1, indicating that the grouping of taxpayers' compliance levels using K-Means is very good.

4. CONCLUSION

This research successfully developed and implemented an automatic data mining pipeline to cluster the level of compliance of PBB-P2 taxpayers in BPKPD Tebing Tinggi City. Using the Java programming language and the SMILE library, the system can automatically extract, transform, and load data into a MySQL database every hour.

The results show that applying the K-Means algorithm in this pipeline can identify taxpayer compliance patterns more accurately and efficiently than manual approaches. With this system, local governments can gain deeper insights into tax compliance, which can be used to formulate more effective policies in improving tax revenues and taxpayer compliance.

Implementing this automated pipeline also proves that the data mining approach can be integrated into the tax system to support real-time data-driven decision-making. For further development, this study can be expanded by evaluating the performance of the clustering model and considering other algorithms to improve the accuracy and interpretability of clustering results.

REFERENCES

Agostinelli, S., Benvenuti, D., Luzi, F. De, & Marrella, A. (2023). Big Data Pipeline Discovery through Process Mining: Challenges and Research Directions⋆. *Catalogo Dei Prodotti Della Ricerca*, *101016835*.

Gallego, V., Lingan, J., Freixes, A., Juan, A. A., & Osorio, C. (2024). Applying Machine Learning in Marketing: An Analysis Using the NMF and k-Means Algorithms. *Information (Switzerland)*, *15*(7), 1–16. https://doi.org/10.3390/info15070368

Iqbal, M., Sipayung, S. P., Sinaga, A. R., & Hasugian, P. M. (2024). Analysis of Student Achievement with K-Means on Socioeconomic , Behavioral , and Psychological Factors. *Jurnal Info Sains : Informatika Dan Sains*, *14*(04), 715–728. https://doi.org/10.54209/infosains.v14i04

Novelan, M. S., Efendi, S., Sihombing, P., & Mawengkang, H. (2023). Optimation Cavacity Vehicle Routing Problem with K-Nearest Neighbor in Classification of Goods Ditribution Route. *2023 International Conference of Computer Science and Information Technology (ICOSNIKOM)*, 1–6.

Putera, A., Siahaan, U., Jabar, A. A., Pranoto, S., & Sutiono, S. (2024). Analysis of Property Tax Bill Classification Using the C4 . 5 Algorithm. *Journal of Information Technology, Computer Science and Electrical Engineering (JITCSE)*, *1*(3), 181–185. https://doi.org/10.30596/jitcse

Putera, A., Siahaan, U., Sutiono, S., Pranoto, S., & Mentari, R. S. (2024). Analysis of Property Tax Payment Compliance Classification in Tebing Tinggi City Using the C4 . 5 Decision Tree Algorithm. *Journal of Information Technology, Computer Science and Electrical Engineering (JITCSE)*, *1*(2), 134–138.

https://doi.org/10.61306/jitcse.v1i2
Putra, P. H., Syahputra, Z., Novelan, M. S., Budi, P., Utara, S., & Info, A. (2021). Application Of The K-Means Algorithm In Identifying Types Of Skin Disease. *Jurnal Infokum*, *9*(2), 281–286.
Raj, A., Bosch, J., Holmstr, H., & Wang, T. J. (2020). Modelling Data Pipelines. *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 13–20. https://doi.org/10.1109/SEAA51224.2020.00014
S.Pranoto and D.Nasution. (2024). Business Intelligence Menggunakan Apache Superset untuk Sistem Pendukung Keputusan Kebijakan Penagihan Pajak Bumi dan Bangunan : Studi Kasus BPKPD Kota Tebing Tinggi. *Indonesian Journal of Education*, *2*(3), 154–160.
Sitorus, Z., Pranoto, S., & Sutiono, S. (2024). Comparison of Accuracy between Naïve Bayes and Decision Tree Methods for Property Tax (PBB-P2) Compliance in Tebing Tinggi City. *Journal of Information Technology, Computer Science and Electrical Engineering (JITCSE)*, *1*(2), 121–128. https://doi.org/10.61306/jitcse.v1i2
Sri Wahyuni, Ahmad Akbar, Abdul Khaliq, & Aulia Akbar. (2023). Implementation of the Membership Method in Developing a Digital Marketing Website for Secanggan Village Sea Products. *International Journal Of Computer Sciences and Mathematics Engineering*, *2*(2), 115–123. https://doi.org/10.61306/ijecom.v2i2.29
Wahyuni, S. (2018). Implementation of Data Mining to Analyze Drug Cases Using C4.5 Decision Tree. *Journal of Physics: Conference Series*, *970*(1). https://doi.org/10.1088/1742-6596/970/1/012030
Wahyuni, S., Julia Sari, D., & Afifah, N. (2022). Implementation of the Ternakloka Application membership method in increasing livestock sales in Kota Pari Village. *Sciences Development and Technology*, *2*(1). http://creativecommons.org/licenses/by-sa/4.0/
Wahyuni, S., & Marbun, M. (2020). Implementation of Data Mining in Predicting the Study Period of Student Using the Naïve Bayes Algorithm. *IOP Conference Series: Materials Science and Engineering*, *769*(1). https://doi.org/10.1088/1757-899X/769/1/012039
Wahyuni, S., Zarlis, M., Solikhun, Jollyta, D., Safii, M., & Sulistianingsih, I. (2019). Implementation of MD Heuristic Method for Classifying Numerical Data in Data Preprocessing. *Journal of Physics: Conference Series*, *1255*(1). https://doi.org/10.1088/1742-6596/1255/1/012060
Zhao, W. L., Deng, C. H., & Ngo, C. W. (2018). k-means: A revisit. *Neurocomputing*, *291*, 195–206. https://doi.org/10.1016/j.neucom.2018.02.072