Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometerie & Operationele Research

Optimal Combination of Clustering and Dimensionality Reduction for Complex Datasets: A Case Study on Joke Ratings

Niek Hendriks (484171)

Supervisor:	J. Durieux
Second assessor:	M. Khismatullina
Date final version:	29th July 2023

The views stated in this thesis are those of the author and not necessarily those of the supervisor, second assessor, Erasmus School of Economics or Erasmus University Rotterdam.

Abstract

The last decades, data usage has grown rapidly, resulting in complex and big datasets. This research tries to find the optimal combination of clustering algorithms and dimensionality reduction techniques for effectively clustering complex datasets. The study uses a subset of the Jester dataset, which contains 4.1 million anonymous ratings of 100 jokes, ranging from -10 to 10, provided by 73,421 readers between April 1999 and May 2003. Four dimensionality reduction techniques (PCA, ICA, t-SNE, and LLE) and three clustering algorithms (K-means, AGNES, and DBSCAN) are performed. The research question is, "What combination of clustering algorithms and dimensionality reduction techniques yields the most accurate results in clustering complex datasets and does DBSCAN outperform K-means and AGNES in clustering data?". The results suggest that the combination of t-SNE with K-means clustering provides the best performance for the specific dataset. This combination outperforms the other combinations of clustering methods (AGNES and DBSCAN) and dimensionality reduction techniques (PCA, ICA, and LLE). Moreover, DBSCAN performs the weakest among the clustering algorithms, but ICA, examined.

1 Introduction

In the modern world, the usage of big data has grown exponentially. According to McAfee, Brynjolfsson, Davenport, Patil and Barton (2012), in 2012, each day 2.5 Exabytes were generated from multiple sources, such as smartphones or product codes. Kune, Konugurthi, Agarwal, Chillarige and Buyya (2016) stated that this amount would grow to 40 Zettabytes by 2020, which is a tremendous growth. To express this growth in numbers, one Exabyte is equivalent to one billion Gigabytes and one Zettabyte is equivalent to 1000 Exabytes. As a result of this growth, analyzing and understanding large and complex datasets has become a big challenge. Companies and individuals get enormous amounts of data from various sources such as customer data, social media, and scientific measurements. Extracting value and gaining insights from these datasets has become essential for making wise decisions (Ji, 2017).

The combination of dimensionality reduction and clustering offers a lot of advantages for different domains in the modern world. Data reduction makes it possible to reduce the data size and, therefore, can significantly reduce the processing time and storage needs. In R. Sembiring (2011) for example, the results show that dimension reduction significantly shorten processing time and also increased performance of clustering in several health datasets. This is very useful in situations where the storage capacity is limited or when there is little time. Clustering helps to identify groups and patterns within the data. Companies in different industries and individuals could gain a better understanding of the structure and characteristics of the data. This could lead to improved decision-making for companies, for example, targeted marketing campaigns based on customer segmentation or understanding health patterns for medical diagnosis. In (Noviantoro & Huang, 2022), for example, they examine which features are most important for different groups of airline passengers in obtaining the highest passenger satisfaction. The airline could use these results for performing targeted marketing and improving passenger satisfaction. So, the combination of data reduction and clustering can help companies and individuals to extract valuable information from big datasets and enable them to improve decision-making.

In this research, the focus lies on investigating and evaluating different methods and techniques for data reduction in combination with clustering. By investigating these different approaches, the goal is to improve the understanding of the capabilities and limitations of these techniques and how they perform on various datasets. The research question is: **"What combination of clustering algorithms and dimensionality reduction techniques yields the most accurate results in clustering complex datasets, and does DBSCAN outperform K-means and AGNES in clustering data?"**.

The data used in this paper is a subset of the Jester dataset, which contains around 4.1 million anonymous ratings of 100 jokes. The ratings vary in the range of -10 to 10 and are rated by around 73,421 readers in the time window from April 1999 to May 2003. In the original paper, Renjith, Sreekumar and Jathavedan (2019), four dimensionality reduction techniques are used, two linear and two non-linear. For the linear techniques, Principal Component Analysis (PCA) and Independent Component Analysis (ICA) are performed, while for the non-linear techniques, t-Distributed Stochastic Neighbor Embedding (t-SNE) and Locally Linear Embedding (LLE) are applied. The clustering algorithms used in the original paper are K-means and Agglomerative Hierarchical Clustering (AGNES). In this research, the paper of Renjith et al. (2019) is replicated, and the Density-based spatial clustering of applications with noise (DBSCAN) is added to extend the original paper and investigate if this clustering method performs better than the methods used in the original paper. After applying both the different dimensionality reduction techniques and clustering methods, the combination of t-SNE with K-means performs the best for the dataset used in this research. The contribution of this research is that it provides insights into the optimal combination of clustering algorithms and dimensionality reduction techniques for complex datasets. The paper is structured into six sections. In Section 2, important context and knowledge about the problem at hand are provided, along with its corresponding literature. Section 3 describes the data and the appropriate data preparation for the analysis. In Section 4, the Methodology discusses the methods used for clustering, dimensional reduction techniques, and internal clustering validation. Section 5 discusses the results of the used combinations of methods and compares the performance of these methods. Finally, the main results and conclusions of this study are highlighted in Section 6.

2 Literature Review

In the field of comparing clustering methods and dimension reduction techniques, a lot of work has been done, as well as in analyzing social media data. This literature review aims to provide an overview of existing research on data reduction and clustering methods, specifically focusing on PCA, ICA, t-SNE, LLE, K-means, and AGNES. Moreover, this review explores the potential extension of incorporating DBSCAN as an alternative clustering method.

In Tang, Shepherd, Heywood and Luo (2005), they compare four dimension reduction techniques, including ICA, in the context of document clustering. The techniques evaluated are ICA, Latent Semantic Indexing (LSI), Document Frequency (DF), and Random Projection (RP). The research tries to measure the relative effectiveness and robustness of these techniques

when combined with the k-means clustering algorithm on five benchmark datasets. The paper concludes that ICA is the most effective technique used in the research, followed by LSI, DF, and RP.

Renjith, Sreekumar and Jathavedan (2020) focuses on evaluating the performance of clustering algorithms when applied to datasets with different sizes and dimensions. The study specifically analyzes dimensionally reduced social media data and compares the clustering quality using internal validation indices. The paper evaluates nine key algorithms among partitioning, hierarchical, density-based, and model-based clustering approaches using different social media datasets. By varying the cardinality and dimensionality parameters of the datasets, the performance trends of these algorithms in terms of turnaround time are captured. Based on their experiments, CLARA, CLARANS, and k-means algorithms are the best performing algorithms with varying cardinality. Another result in the paper is that changes in dimensionality do not impact hierarchical clustering approaches, whereas there is a positive influence on the execution time for partitioning, density-based, and model-based clustering approaches.

In the paper Liu et al. (2020), the authors investigate the use of the K-means++ algorithm in combination with t-SNE dimension reduction technique for clustering transformer districts. The objective of this research is to effectively group transformer districts based on their similarities and patterns. The K-means++ algorithm is used to divide the transformer districts into clusters by minimizing the distance between data points and cluster centroids. To make the algorithm work even better, the authors used the t-SNE dimension reduction technique. By reducing the dimensionality of the data, t-SNE improves the clustering results. The results of the research demonstrate that the application of the K-means++ algorithm with t-SNE dimension reduction successfully clusters transformer districts, revealing important insights into their similarities and patterns.

The paper Hai, Nhung and Jasek (2022) investigates how AGNES can be used to improve the estimation of software development effort. The authors propose a method that includes AGNES in the estimation process by analyzing past project data and grouping similar projects together. The research shows that by incorporating AGNES, estimations become more accurate, leading to better planning of resources and project management. The utilization of AGNES improves the clustering process, resulting in overall enhanced accuracy in effort estimation. Another paper, by Nath and Nema (n.d.), conducted a comparative and predictive analysis utilizing three different datasets, namely IRIS, Wine, and Seed from the UCI benchmark in Machine learning on four distinctive techniques. The goal of the study was to form a good clustering pattern for each dataset for given techniques, and it showed that the performance of clustering algorithms depends on the dataset and the dimensionality reduction technique used.

While PCA, ICA, t-SNE, LLE, K-means, and AGNES have been widely used in big data analysis, there is an opportunity to further enhance the clustering capabilities by incorporating DBSCAN. DBSCAN is a density-based clustering algorithm that identifies clusters based on density, allow-

ing for clusters of arbitrary shapes. This extension could be helpful to address some limitations of the other clustering methods, such as the requirement of specifying the number of clusters. The paper R. W. Sembiring, Zain and Embong (2011) presents an alternative model to extract multidimensional data through dimension reduction. When dealing with datasets that have a lot of dimensions, it can be challenging to process them efficiently. This paper suggests a new method that reduces the number of dimensions while still capturing important information. Five dimension reduction techniques are evaluated, such as ISOMAP (Isometric Feature Mapping), KernelPCA, LLE (Local Linear Embedded), Maximum Variance Unfolded (MVU), and Principal Component Analysis (PCA). They found that by using these techniques, the processing time became shorter and the quality of the results improved. The best performance was seen when using DBSCAN with KernelPCA and Super Vector with KernelPCA. As in the previous paper, DBSCAN could probably outperform the other techniques in this research as well. By introducing DBSCAN to the analysis of joke datasets, the effectiveness in identifying clusters of jokes with similar characteristics or humor styles is investigated.

3 Data

In this section, the data used in the paper and the processing process will be discussed. The dataset used is a subset of the Jester dataset, which contains around 4.1 million anonymous ratings of 100 jokes. The ratings vary in the range of -10 to 10 and are rated by around 73,421 readers in the time window from April 1999 to May 2003. The first column contains the user's ID and the rest of the columns contains the ratings of the users for each joke. If a user did not rate a joke, the value for this joke is set to 99. Due to computer performance limitations, a sample with a cardinality of 3500 is used. Since the specific subsample is not mentioned in the original paper, it is difficult to find the exact same subsample. The Jester set is divided into different zip files, where the subsample for this research is taken from the first zip file called "jester-data-1.zip", which contains 24,983 users and the ratings of 100 jokes. For this research the users need to have rated all the 100 jokes, so the file chosen for this research contains candidates that are suitable, since the users in this file have rated 36 or more jokes.

Pre-processing: In order to make the dataset applicable, a couple of pre-processing steps need to be done. First, it is important to create a subsample from the "jester-data-1.zip" file where the users have rated all the jokes, and thus the rows do not contain any value of 99, as these values would influence the clustering process due to their high value in comparison with the other data (values ranging from -10 to 10). If a user has rated all the 100 jokes, the user's ID is set to 100 (i.e., the value in the first column of the dataset is set to 100). Now, from the users that rated all the 100 jokes, a random subsample of 3500 rows is selected. After creating the subsample, the first column is deleted since the user ID should not be involved in the clustering process or dimension reduction of the rating of jokes.

4 Methodology

In this section all the different types of clustering, dimension reduction techniques and internal clustering validation measurement which are used in the original paper and which will be used in my extension will be explained.

4.1 Clustering methods

K-means: The K-means clustering algorithm is a centroid-based method. This method uses centroids (i.e. the center of each cluster) to minimize the sum of the distances between the data-points and their corresponding cluster centroids. K-means clustering divides the data-set into k distinct clusters with the help of the iteration process explained above. The iterations have the goal of minimizing the total within cluster variation (TWCV) for the clusters being generated. The TWCV formula is as follows:

$$TWCV = \sum_{k=1}^{K} \sum_{E_i \in C_k} (E_i - \mu_k)^2 \tag{1}$$

where K is the cluster count and E_i is an entity in cluster C_k with centroid μ_k.

AGNES: AGNES is a hierarchical clustering method, which means that clusters are represented in a hierarchical tree called a dendogram. The algorithm merges in each iteration the nearest clusters in subsequent steps to result in a single cluster with all the entities of the data-set. The merging of clusters can be done by several linkage criterion. In the research, Ward's method, which can be found in the paper of Tibshirani (2012), is used for AGNES which aims to minimize the variance within each cluster as clusters are merged together. Ward's method is often used as a linkage method since the clustering results in clusters with similar size and a good balance between compactness and separation. The procedure is as follows:

1. Start from n clusters, with each cluster containing only one data-point.
2. Suppose $D_{nxn} = [d_{rs}]$ is a proximity matrix (i.e. a square matrix containing the distances, taken pairwise between the data-points of a matrix). Find an inequality matrix D for the most similar pair rs, with r and s the objects.
3. Merge r and s into one new cluster (rs) ans decrease the number of clusters by one through deleting row and column of object r and s. Then calculate the inequality between cluster rs and all the other remaining objects and add row and column to the new inequality matrix.
4. Repeat steps 2 and 3 $n - 1$ times in order to merge all object into one cluster. In each step, identify the cluster union and value of inequality where the clusters are united. (Wijuniamurti, Nugroho & Rachmawati, 2022)

The steps to form clusters in each iteration are irreversible, though it lacks global distribution details of the data.

DBSCAN: The DBSCAN algorithm is a density-based clustering method, which groups data points based on their density. The algorithm requires two parameters, namely eps and MinPts. The key idea of DBSCAN is that for each object of a cluster the neighborhood of a given radius (Eps) has to contain at least a minimum number of objects (MinPts), which means that the cardinality of the neighborhood has to exceed some threshold (Khan, Rehman, Aziz, Fong & Sarasvady, 2014). The ϵ-neighborhood of an arbitrary point 'p' is formulated as follows:

$$N_{Eps} = q\epsilon D/dist(p,q) < EPS \tag{2}$$

Where D is the set of data-points. If the ϵ-neighborhood of a point 'p' contains more data-points than the required MinPts, this point is called a core point.

$$N_{Eps}(P) > MinPts \tag{3}$$

If equation 3 is not specified, this data-point is not a core point (Khan et al., 2014). In order to find the optimal eps, the K-Nearest Neighbors (KNN) plot , which sorts all data points by their k-nearest neighbor distance is used. The Euclidean distance (Liberti, Lavor, Maculan & Mucherino, 2014) is used by the KNN distance plot for DBSCAN and the optimal value for eps is where the "elbow" appears in the plot, i.e. where the k-th nearest distance suddenly increases steeply. KNN-distance plot is used because of its ease of implementation and because it can be effective with a complex data structure.

4.2 Linear dimension reduction techniques

Principal Component Analysis (PCA): PCA is a feature projection approach and extract a new set of features, called principal components, as linear combination of the original features. Orthogonal transformation is used to convert a set of correlated features into a set of linearly uncorrelated variables, called principal components. Suppose the random vector $X = [X_1, X_2, ..., X_N]$ has covariance matrix V with eigenvalues $\beta_1 \geq \beta_2 \geq ...\beta_n \geq 0$ and normalized eigenvectors $l_1, l_2, ..l_n$ (Assi, 2020). Then the following linear combination of the principal components is as follows:

$$X_{PC_t} = l_i'X = l_{1i}X_1 + l_{2i}X_2 + ...l_{ni}X_n \tag{4}$$

$$Var(X_{PC_T}) = l_i'Vl_i, i = 1, 2, ..., n \tag{5}$$

$$Cov(X_{PC_T}, X_{PC_T}) = l_i'Vl_k \tag{6}$$

The uncorrelated linear combinations $X_{PC_1}, X_{PC_2}, ..., X_{PC_n}$ ere the principal components ranked in descending order based on their variances (Assi, 2020).

Independent Component Analysis (ICA): ICA is another linear dimensionality reduction technique and quite similar to PCA, but is different in some ways. The goal of ICA is to find a linear combination based on independent features, whereas PCA focuses on uncorrelated features. In the context of ICA, we assume that the observed data is a linear mixture of inde-

pendent components. The procedure is as follows, based on the paper of Hyvärinen and Oja (2000): Assume there are n linear mixtures denoted by $x_1, x_2, ..., x_n$. These mixtures x_j are composed of n independent components denoted by $s_1, s_2, ..., s_n$. The linear mixture model can be expressed as follows:

$$x_j = a_{j1}s_1 + a_{j2}s_2 + \ldots + a_{jn}s_n \quad \text{for all } j$$

Where x_j represents the jth mixture signal, and s_i represents the ith independent component. The coefficients a_{ji} represent the mixing weights. By using vector-matrix notation, the entire set of mixtures x can be expressed as a random vector, and the set of independent components s can also be expressed as a vector. The mixing weights a_{ji} can be expressed as elements of a mixing matrix A. The random vector of mixtures x is given by:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The random vector of independent components s is given by:

$$s = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix}$$

The mixing matrix A is given by:

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}$$

The mixing model in vector-matrix notation is represented as:

$$x = As$$

The ICA model describes how the observed data (mixtures) are generated by mixing the independent components. The goal of ICA is to estimate both the mixing matrix A and the independent components s from the observed data x. This must be done under the assumptions that the components are statistically independent and that the mixing matrix is unknown (Hyvärinen & Oja, 2000).Once the mixing matrix A is estimated, we can compute its inverse W and we can obtain the independent components s using the following equation:

$$s = Wx$$

.

4.3 Non-linear dimension reduction techniques

t-Distributed stochastic Neighbor Embedding (t-SNE): For a given (high) d-dimensional data set $X = \{x_1, x_2, ..., x_N\} \subset \mathbb{R}^d$, the goal of t-SNE is to compute the low dimensional embedding $Y = \{y_1, y_2, ..., y_N\} \subset \mathbb{R}^s$ with $s << d$ (s usually 2 or 3). This means that if two points x_i and x_j are close in the input space, then y_i and y_j are also close (Linderman, Rachh, Hoskins, Steinerberger & Kluger, 2017). The similarity between x_i and x_j in the input space (i.e. p_{ij}) are defined as:

$$p_{i|j} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)} \tag{7}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \tag{8}$$

In the above equation, the bandwidth of the Gaussian kernels, σ_i, is chosen so the perplexity of the conditional distribution P_i equals a certain value, where P_i is the conditional distribution of all the other points given x_i (Linderman et al., 2017). For the low-dimensional embedding, the similarity between points y_i and y_j is measured according to the Cauchy kernel:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}} \tag{9}$$

The t-SNE approach finds the points $\{y_1, ..., y_n\}$ that minimize the divergence between the points in the input space P and the embedding space Q, called the Kullback-Leiber divergence. The formula of Kullback-Leiber is as follows:

$$C(\gamma) = \sum_{i \neq j} p_{ij} log \frac{p_{ij}}{qij} \tag{10}$$

(Linderman et al., 2017)

The t-SNE approach can keep the local structure of the data even after performing dimensionality reduction, which causes that the overall geometry of the data-set is preserved.

Locally Linear Embedding (LLE): LLE is a non-linear approach with the following procedure: First, LLE finds the k-nearest neighbors of the points using the Euclidean distance. Let $x_{ij} \in \mathbb{R}^d$ denote the j-th neighbor of x_i and $X_i := [x_{i1}, x_{i2}, ..., x_{ik}] \in \mathbb{R}^{dxk}$ are the k neighbors of x_i. Secondly, it approximates each data vector as a weighted linear combination of its k-nearest neighbors. The optimization of the linear combination is as follows:

$$\begin{aligned} &\min_{\tilde{W}} \epsilon(\tilde{W}) := \sum_{i=1}^{n} \|x_i - \sum_{j=1}^{k} \tilde{w}_{ij} x_{ij}\|_2^2, \\ &subject to \sum_{j=1}^{k} \tilde{w}_{ij} = 1, \forall i \in \{1, ..., n\} \end{aligned} \tag{11}$$

Where the weights are included in $\tilde{W} := [\tilde{w}_1, \ldots, \tilde{w}_n]' \in \mathbb{R}^{n \times k}$, $\tilde{w}_i := [\tilde{w}_{i1}, \ldots, \tilde{w}_{ik}]' \in \mathbb{R}^k$ includes the weights of the i-th data-point using the k neighbors of this point, and $x_{ij} \in \mathbb{R}^d$ is the j-th neighbor of the i-th data-point (Ghojogh, Ghodsi, Karray & Crowley, 2020). The constraint ensures that the weights of the linear construction sum up to one.

Finally, it computes the weights that best reconstruct the vectors from its neighbors, then produce the low-dimensional vectors best reconstructed by these weights. This linear embedding is formulated as follows:

$$\begin{aligned} &\min_{\mathbf{Y}} \sum_{i=1}^{n} \|\mathbf{y}_i - \sum_{j=1}^{n} w_{ij}\mathbf{y}_j\|_2^2, \\ &\text{subject to } \frac{1}{n}\sum_{i=1}^{n} \mathbf{y}_i\mathbf{y}_i' = \mathbf{I}, \\ &\qquad\qquad \sum_{i=1}^{n} \mathbf{y}_i = \mathbf{0} \end{aligned} \tag{12}$$

Where the embedded data-points are the rows of $\mathbb{R}^{nxp}$, i.e., $Y := [y_1, \ldots, y_n]'$, $y_i \in \mathbb{R}^p$ is the i-th embedded data-point, I is the identity matrix, and w_{ij} is $\tilde{w}_{ij}$ if x_j is in the k-nearest neighbors of x_i and zero otherwise. LLE expects the data-set to be non-linear, smooth and it is sensitive to outliers and noise. The LLE approach ensures that the neighborhood structure will be preserved in the low-dimensional space.

4.4 Internal clustering validation

In the internal clustering validation, the cluster quality is measured in terms of a set indices. In this paper, the following indices are used: The Silhouette Index, Dunn Index, Calinski-Harabasz Index and Davies-Bouldin Index.

The Silhouette index measures the difference between average distance within the cluster and the minimum distance between the clusters, where a high value means a better cluster quality (Wang & Xu, 2019). It measure the cohesion of data-points in each cluster and the dissimilarity between clusters. The formula is as follows:

$$s(i) = \frac{\max(a(i), b(i))}{b(i) - a(i)} \tag{13}$$

Where $a(i)$ represents the average distance of data point i to all other data points within the same cluster (intra-cluster distance) and b(i) represents the average distance of data point i to all the data points in the nearest neighboring cluster (inter-cluster distance). The value of the Silhouette index ranges from -1 to 1, where 1 means a good fit and -1 a bad fit (Rousseeuw, 1987).

The Dunn index measures which evaluates the degree of compactness of clusters and the separation between clusters, where a high value indicate more goodness. The formula is as follows:

$$\text{Dunn Index} = \frac{\min \text{inter-cluster distance}}{\max \text{intra-cluster distance}} \tag{14}$$

Where "min inter-cluster" represents the minimum distance between any two clusters, i.e. it is the smallest distance between data points belonging to different clusters. "Max-intra-cluster" represents the maximum distance between data points within the same cluster, i.e. it is the largest distance between any two data points in the same cluster. The value of the Dunn index ranges from 0 to infinity (Ncir, Hamza & Bouaguel, 2021).

The Calinski-Harabasz index measures the degree of dispersion between clusters and within clusters, where a high value means better cluster quality (Wang & Xu, 2019). The formula of the Calinski-Harabasz index is as follows:

$$\text{Calinski-Harabasz Index} = \frac{B}{W} \times \frac{N-k}{k-1} \tag{15}$$

Where B is the between-cluster variance, i.e. the variance between cluster centroids, W is the within-cluster variance, N is the total number of data points and k is the number of clusters (Wang & Xu, 2019).

The Davies-Bouldin Index measures the compactness of clusters and the separation between clusters based on the cluster centroids, where a lower value means a better cluster quality (Thomas, Santos & Cofre, 2014). The value ranges from 0 to 1 and the formula is as follows:

$$\text{DBI} = \frac{1}{n} \sum_{i=1}^{n} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \tag{16}$$

Where n is the number of clusters, σ_i is the average distance of each point in cluster i to the centroid c_i of that cluster and $d(c_i, c_j)$ is the distance between the centroids c_i and c_j of clusters i and j (Tempola & Assagaf, 2018).

4.5 Flowchart

The procedure in analyzing the data is following the flowchart which is shown in the paper of Renjith et al. (2019).

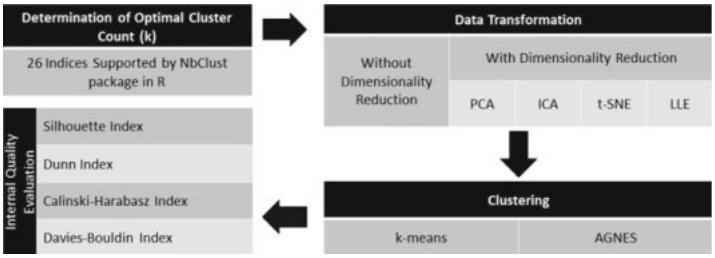


Figure 1: Flowchart of the analyzing proces.

The number of clusters is determined before performing the data reduction, except for DBSCAN as this method does not need an initialization for the number of clusters. Data reduction can result in some loss of information as certain variations in the data are eliminated. This can impact clustering performance and therefore we determine the number of clusters before performing data reduction methods (Jolliffe, 2002).

5 Results

In this section, the results from both the original clustering methods, K-means and AGNES, and the extension (DBSCAN), will be presented. First, the clustering methods in combination with linear dimension reduction techniques, PCA and ICA, will be presented, and the non-linear techniques, t-SNE and LLE, will be presented afterwards. A general note that applies to all results is that the original research is not reproducible because the exact subsample used in the original research has not been mentioned.

Before performing data reduction or clustering with K-means and AGNES, the number of clusters needs to be determined first. To do this, the R-package NbClust is used (Charrad, Ghazzali, Boiteau & Niknafs, 2014). The function is as follows: NbClust(data, diss = NULL, distance = "euclidean", min.nc = 2, max.nc = 10, method = "kmeans", index = "all"), where as the distance measure for the dissimilarity matrix the Euclidean distance is chosen, which can be found in the paper of Liberti et al. (2014), the minimal number of clusters is 2 and the maximal number of clusters is 10. For the method, kmeans is chosen for K-means and Ward is chosen for AGNES. NbClust determines the number of clusters by the majority voting of 26 indices. In Table 1, an overview of all indices is given for K-means, combined with the number of clusters each index advises. In Table 2, the same is done for AGNES. After majority voting, the number of clusters in our subsample for both K-means and AGNES is 2.

Index	Scott	Cindex	Ball	McClain	Dindex	KL	Silhouette	Duda	PseudoT2	SDbw	Hartigan	Marriot	TrCovW	TraceW	Friedman	Beale	Ratkowsky	CH	CCC	Rubin	DB	Frey	Hubert	PtBiserial	Dunn	SDindex
Nb of clusters	-Inf	2	3	4	10	2	2	2	2	2	3	5	3	3	3	2	3	2	2	2	2	2	2	1	0	0

Table 1: The number of clusters for each index after performing NbClust for K-means.

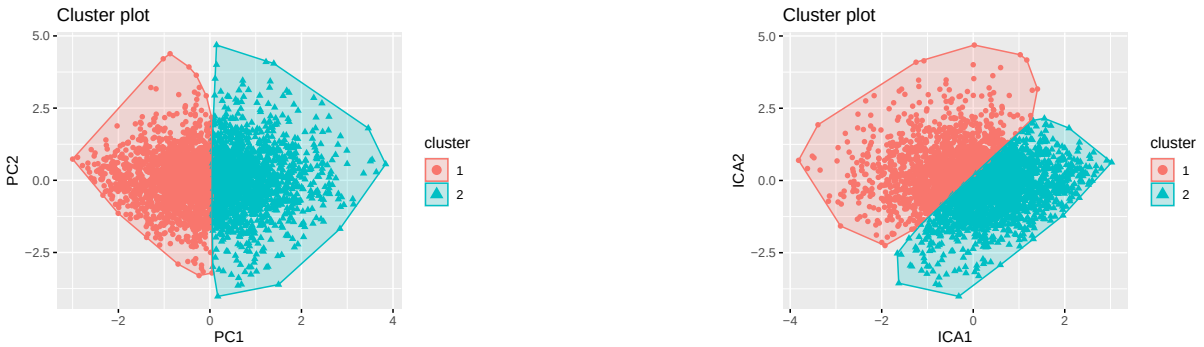
Index	Scott	Cindex	Ball	McClain	Dindex	KL	Silhouette	Duda	PseudoT2	SDbw	Hartigan	Marriot	TrCovW	TraceW	Friedman	Beale	Ratkowsky	CH	CCC	Rubin	DB	Frey	Hubert	PtBiserial	Dunn	SDindex
Nb of clusters	-Inf	2	3	10	8	6	6	6	NA	6	3	5	3	3	3	2	3	2	2	2	2	2	2	1	0	0

Table 2: The number of clusters for each index after performing NbClust for AGNES.

5.1 Dimensionality reduction with K-means

The first method applied in this research is the K-means clustering in combination with PCA dimension reduction. Before applying the K-means clustering, PCA is performed on the subsample taken from the Jester dataset. For performing PCA, the function "prcomp" of the package "Stats" is used (R Core Team, 2013). The dimensionality will be reduced from 100 to 2. The reduction from 100 to 2 is done because t-SNE only allows a dimension of 1, 2, or 3. For dimension 2, we preserve information from the original components and can also plot the clusters, so visual results are available. After applying both methods, the components are used

to plot the 2D graph. The result of the clustering is shown in Figure 2a.
After applying PCA, ICA is performed in combination with K-means clustering. To do so, the package 'caret' is used (Kuhn & Max, 2008), and especially the functions preProcess() and predict(). First, the subsample consisting of 3500 rows and 100 features is pre-processed for ICA, after which predictions are made and the independent components are extracted. The first two independent components are used to perform K-means, and the result of the clustering can be seen in Figure 2b. The first two independent components are extracted to be able to show a two-dimensional graph.



(a) The clusters after performing PCA and K-means

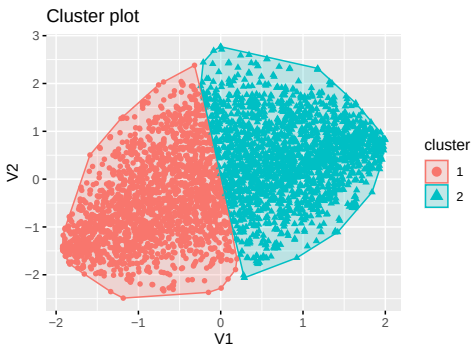
(b) The clusters after performing ICA and K-means

Figure 2: Linear dimension reduction with K-means

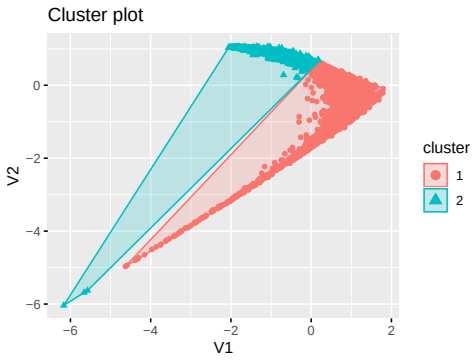
Now both the linear dimensional reduction techniques have been applied, and the non-linear techniques will be used next. First, t-SNE is applied. The package "Rtsne" is used to get the low-dimensional embeddings of the subsample, and then K-means is performed to cluster the t-SNE data (Krijthe, 2015). The function "Rtsne" requires a few hyperparameters to be specified. The function is as follows: Rtsne(subsample, dims, perplexity, theta, max iteration). For the dimension, at most 3 can be selected, but due to the dimension of the graphs, 2 is chosen. The perplexity, with a default value of 30, typically ranges from 5 to 50, where a higher value is assigned for more complex datasets. As the subsample used in this research is not that big, the default value is used. Theta is set to 0.5 to create a balance between accuracy and running time. The maximum iterations, which often range between 1000 and 5000, are set at 3000, so that the method performs many iterations while keeping the running time acceptable. The result can be found in Figure 3a.

The last dimensional reduction technique used in the paper is LLE. The "lle" package is used to get the embedded coordinates before applying K-means with two clusters (De Ridder & Duin, 2002). The formulation of the LLE function is as follows: lle(subsample, m, k, reg, ss, p, id, nnk, eps, iLLE, v), where "subsample" is the dataset used for LLE. "m" is the dimension, which in this research is 2, and "k" is the number of nearest neighbors, which often ranges from 5 to 20, and in this research, it is set at 5 because the data in the subsample is only two-dimensional due to the dimension reduction techniques. "reg" is the regularization parameter that controls the level of regularization in local linear reconstruction and ranges from 0 to 1. In this paper, "reg" is set to 0.1, as this value is often used. "ss" is for scaling, which is not important in our

case. "p" is for the distance we use in our clustering methods, which is the Euclidean distance. The value for the Euclidean distance is 2. "id" is set to FALSE to emphasize the local structure, and "nnk" is set to TRUE to preserve local characteristics. "eps" and "iLLE" do not need to be specified, and "v" is a variation value which ranges from 0.8 to 0.99. In this research, the value is set to 0.99 to retain as much variation as possible. The cluster results are projected in Figure 3b. This method acquires the most different clusters of the 4 methods applied. In the section of internal cluster validation, we will see if this is reflected in the values of the indices used to compare the different techniques.



(a) The clusters after performing t-SNE and K-means

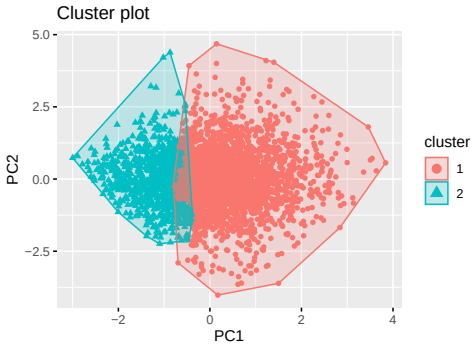


(b) The clusters after performing LLE and K-means

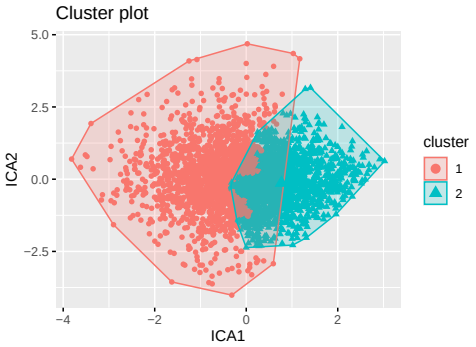
Figure 3: non-linear dimension reduction with K-means

5.2 Dimensionality Reduction with AGNES

The second clustering method applied in this research, is the AGNES clustering. This hierarchical clustering method merges clusters with linkage criteria, in this research Ward's linkage method is used. This method is chosen because it results in clusters with similar size and a good balance between compactness and separation. For all the dimension reduction techniques, the same procedure is followed as by the K-means clustering, with the same parameters for the functions used in R. To perform AGNES, the R package 'cluster' is used, which contains the agnes() function (Maechler, Rousseeuw, Struyf, Hubert & Hornik, 2022). After performing this function, we extract the desired number of clusters, in this case 2, and plot the clusters. The resulting clusters for both PCA and ICA can be found in respectively Figure 4a and Figure 4b.



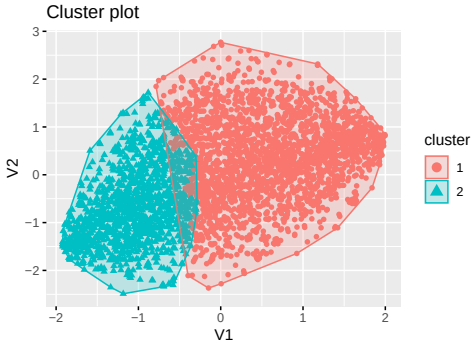
(a) The clusters after performing PCA and AGNES



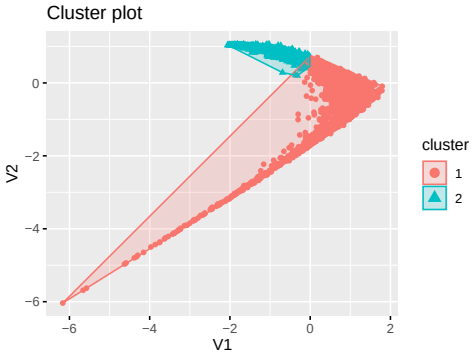
(b) The clusters after performing ICA and AGNES

Figure 4: Linear dimension reduction with AGNES

For the non-linear dimensional reduction techniques, the results are shown in Figure 5a and Figure 5b. For LLe, the clustering gives just like K-means a strange clustering.



(a) The clusters after performing PCA and AGNES

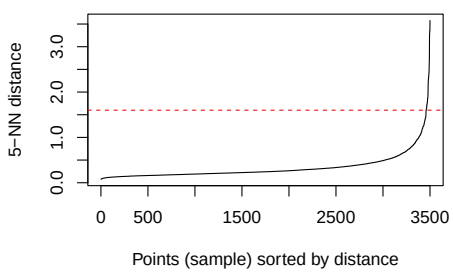


(b) The clusters after performing ICA and AGNES

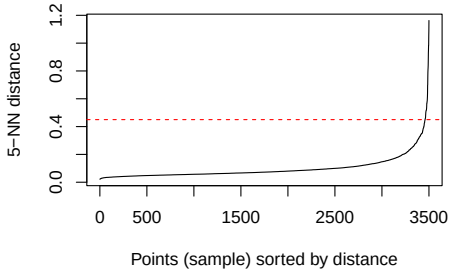
Figure 5: Non-linear dimension reduction with AGNES

5.3 Dimensionality Reduction with DBSCAN

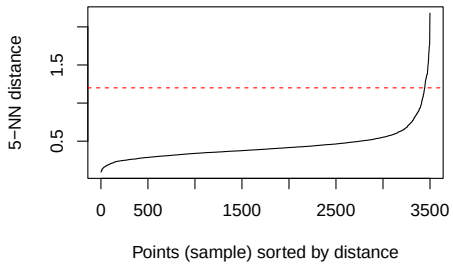
The last clustering method used in this research is DBSCAN. DBSCAN is a density-based method which, unlike K-means and AGNES, does not require a specific number of clusters as input. The package "DBSCAN" is used (Hahsler, Piekenbrock & Doran, 2019), and the function "dbscan" contains two parameters that need to be specified: "eps" and "MinPts". To determine the optimal "eps", the K-nearest neighbors method is used. The function "KNNdistplot()" requires the number of the k-nearest neighbors as input, and in this paper, we use 5. The optimal value for "eps" is where the curve is the steepest. For "MinPts", we use the value 5, which is the same value we used to determine the optimal "eps". The optimal "eps" values for each dimension reduction model are 1.60, 0.45, 1.20, and 0.15, as shown in Figure 6.



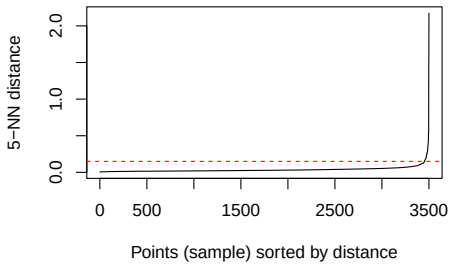
(a) KNN for PCA



(b) KNN for ICA



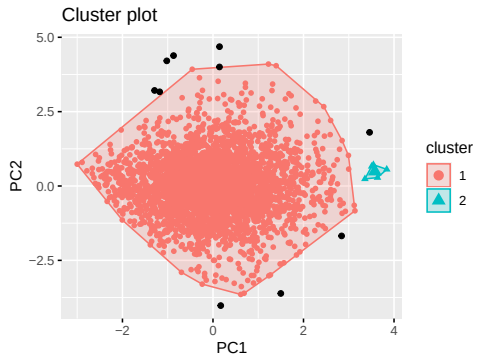
(c) KNN for t-SNE



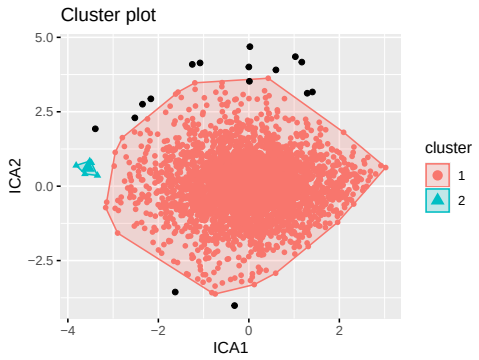
(d) KNN for LLE

Figure 6: The KNN distant graphs for each dimension reduction technique. The red line gives the point wheren the steepest curve appears.

Now the parameters are determined, the clustering can be performed. The results of the clustering for each different dimension reduction technique is presented in the Figures 7 and 8.

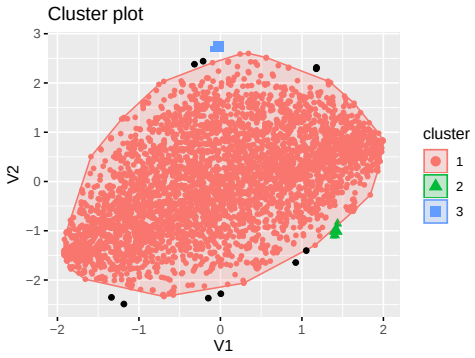


(a) The clusters after performing PCA and DBSCAN

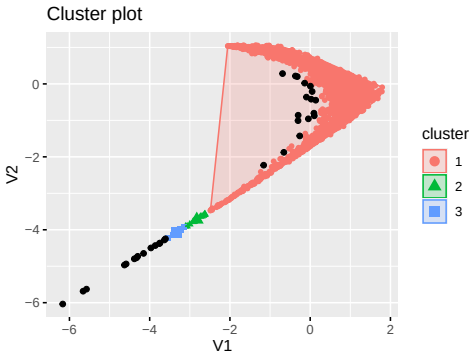


(b) The clusters after performing ICA and DBSCAN

Figure 7: Linear dimension reduction with DBSCAN



(a) The clusters after performing t-SNE and DBSCAN



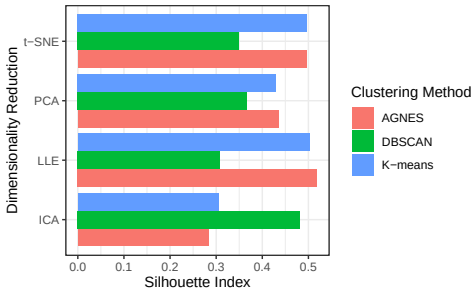
(b) The clusters after performing LLE and DBSCAN

Figure 8: Non-linear dimension reduction with DBSCAN

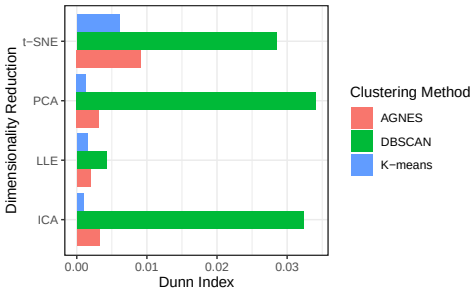
In each of the Figures above, the black points are noise points (i.e. outliers). Where in performing K-means and AGNES the clusters where determined before, DBSCAN makes clusters during the clustering process. This results in 3 clusters instead of 2 for the non-linear dimension reduction techniques.

5.4 Internal clustering validation

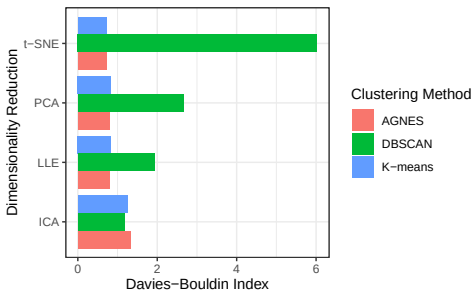
In this subsection, the performance of the clustering methods and dimensional reduction methods will be analysed. Based on several different indices the different methods will be compared to each other. In Figure 9, the indices for each cluster method are shown.



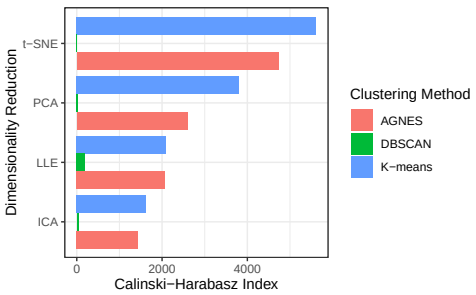
(a) Silhouette Index, where a higher value means better prediction



(b) Dunn Index, where a higher value means better clustering.



(c) Davies-Bouldin Index, where a lower value means better clustering.



(d) Calinski-Harabasz Index, where a higher value means better clustering.

Figure 9: The different indices for cluster validation

Because for each clustering method in combination with a dimensionality reduction technique there is only one value per index, a statistical test is not available for comparing the methods. Therefore the methods are compared via visual and procentual comparison in this research. Assumed in this research is that two methods differ from each other when the values differ at least 15.00%.

The procedure in evaluating which combination is most suitable for this specific subsample from the specific dataset is as follows: Per dimensional reduction technique, the best combination will be sorted out. After that, the best combinations for each technique will be compared to conclude which combination is the best for this research. All the procentual differences are stated in Tables 3, 4, 5 and 6.

Dimension Reduction	Clustering Method	Comparison	Than
PCA	K-Means	1.21% smaller	AGNES
	K-means	17.16% larger	DBSCAN
	AGNES	18.56% larger	DBSCAN
ICA	K-means	7.94% larger	AGNES
	K-means	36.64% smaller	DBSCAN
	AGNES	41.28% smaller	DBSCAN
t-SNE	K-means	0.24% larger	AGNES
	K-means	42.03% larger	DBSCAN
	AGNES	41.62% larger	DBSCAN
LLE	K-means	3.16% smaller	AGNES
	K-means	63.01% larger	DBSCAN
	AGNES	68.26% larger	DBSCAN

Table 3: Comparison of Clustering Methods for Dimension Reduction Techniques for the Silhouette Index, where a higher value means better prediction.

Dimension Reduction	Clustering Method	Comparison	Than
PCA	K-Means	57.28% smaller	AGNES
	K-means	97.10% smaller	DBSCAN
	AGNES	90.73% smaller	DBSCAN
ICA	K-means	68.44% smaller	AGNES
	K-means	96.93% smaller	DBSCAN
	AGNES	90.08% smaller	DBSCAN
t-SNE	K-means	33.54% smaller	AGNES
	K-means	78.71% smaller	DBSCAN
	AGNES	67.93% smaller	DBSCAN
LLE	K-means	22.40% smaller	AGNES
	K-means	64.71% smaller	DBSCAN
	AGNES	54.82% smaller	DBSCAN

Table 4: Comparison of Clustering Methods for Dimension Reduction Techniques for the Dunn Index, where a higher value means better prediction.

Dimension Reduction	Clustering Method	Comparison	Than
PCA	K-means	3.35% larger	AGNES
	K-means	68.80% smaller	DBSCAN
	AGNES	69.77% smaller	DBSCAN
ICA	K-means	5.16% smaller	AGNES
	K-means	6.66% larger	DBSCAN
	AGNES	12.50% larger	DBSCAN
t-SNE	K-means	1.28% larger	AGNES
	K-means	88.02% smaller	DBSCAN
	AGNES	88.16% smaller	DBSCAN
LLE	K-means	4.68% larger	AGNES
	K-means	57.00% smaller	DBSCAN
	AGNES	58.91% smaller	DBSCAN

Table 5: Comparison of Clustering Methods for Dimension Reduction Techniques for the Davies-Bouldin Index, where a lower value means better prediction

Dimension Reduction	Clustering Method	Comparison	Than
PCA	K-means	46.51% larger	AGNES
ICA	K-means	13.12% larger	AGNES
t-SNE	K-means	18.59% larger	AGNES
LLE	K-means	1.06% larger	AGNES

Table 6: Comparison of Clustering Methods for Dimension Reduction Techniques for the Calinski-Harabasz Index, where a higher value means better prediction.

5.4.1 PCA

For the Silhouette Index, a higher value means a better clustering. If we look at Figure 3, the AGNES outperforms the other clustering methods. Although the AGNES value is bigger than the other values, it does not differ K-means enough, since K-means is 1.21% smaller. Both AGNES and K-means differ from DBSCAN for this index, where DBSCAN performs worst. Next is the Dunn Index, where a higher value means a better clustering. Both AGNES and K-means are way smaller for each dimensional reduction technique in comparison with DBSCAN, namely at least 90% smaller. Besides that, the K-means value is 57.28% smaller than AGNES, meaning that K-means performs worst for the Dunn Index. The Davies-Bouldin Index associates good clustering with a lower value. As can be seen in Figure 5, the DBSCAN value is way higher than the other values. K-means and AGNES are close to each other, where K-means is 3.35% larger than AGNES. AGNES performs best, but not better than K-means.The last Index is the Calinski-Harabasz, where a higher value means a better clustering. As can be seen in Figure 6, the DBSCAN bar is not even visible. The K-means value is 46.51% larger than the AGNES value and therefore higher. The K-means performs best for the Calinski-Harabasz Index.

Combining all the results of the different indices, K-means performs best when performing

PCA. Although DBSCAN performs best for the Dunn index, it performs worst for all the other indices. K-means performs better for the Calinski-Harabasz indices and does not differ from AGNES for the other indices.

5.4.2 ICA

For the Silhouette Index, a higher value means a better clustering. If we look at Figure 3, DBSCAN performs better than both AGNES and K-means as they are way smaller and so DBSCAN is the best method according to this index. K-means and AGNES do not differ for this index. The Dunn Index, where a higher value means a better clustering, is for DBSCAN a lot higher than for the other two methods which can be seen in Figure 4. On top of that, K-means has a smaller index value than AGNES. K-means performs worst for this index and DBSCAN best. The Davies-Bouldin Index associates good clustering with a lower value. As can be seen in Figure 5, all values are close to each other for this index. Looking at table 9c, the DBSCAN value is the smallest but no clustering method differs from another and performs best. The last Index is the Calinski-Harabasz, where a higher value means a better clustering. As can be seen in Figure 6, the DBSCAN bar is barely visible. The K-means value is 13.12% larger than the AGNES value which means they don't differ. No method performs best for this index.

Combining all the results of the different indices, DBSCAN performs best for ICA.

5.4.3 t-SNE

For the Silhouette Index, a higher value means a better clustering. In Figure 3, the DBSCAN is outperformed by the other clustering methods. Both K-means and AGNES have higher values for the Silhouette index. Although, the difference between those two values is not enough to say they differ for eachother(0.24%). The Dunn index, where a higher value means a better clustering, is for DBSCAN higher than for the other two methods. K-means is 33.54% smaller than AGNES, which means that the DBSCAN method performs best for t-SNE and K-means worst, according to the Dunn index. The Davies-Bouldin Index associates good clustering with a lower value. The DBSCAN is a lot higher in value than both K-means and AGNES, which is illustrated in Figure 5. K-means and AGNES are respectively 88.02% and 88.15% smaller. Between K-means and AGNES, the difference is a lot smaller, namely K-means is 1.28% higher than AGNES. So DBSCAN performs worst and there is no difference between K-means and AGNES. Finally the Calinski-Harabasz, where a higher value means a better clustering. As can be seen in Figure 6, the DBSCAN bar is, again, barely visible. The K-means value is 18.59% larger than the AGNES value, making it higher. The K-means performs best for the Calinski-Harabasz Index.

Combining all the results of the different indices, K-means performs best when performing t-SNE. It performs better for the Calinski-Harabasz indices and outperforms DBSCAN for all indices but the Dunn Index.

5.4.4 LLE

The Silhouette Index, for which a higher value means a better clustering, has a higher value for K-means and AGNES in comparison with DBSCAN. The value between themselves is although not significant: K-means is 3.16% smaller than AGNES. For this index, DBSCAN performs the worst. Next is the Dunn Index, where a higher value means a better clustering. K-means is worse than both AGNES and DBSCAN, the value is respectively 22.40% and 64.71% smaller. K-means performs worst for this index and DBSCAN best.The Davies-Bouldin Index associates good clustering with a lower value. As can be seen in Figure 5, the DBSCAN value is way higher than the other values and K-means and AGNES do not differ a lot. The value of K-means is 4.68% higher than AGNES, which means that AGNES performs best, but it is not enough to assume they differ. The last Index is the Calinski-Harabasz, where a higher value means a better clustering. As can be seen in Figure 6, the DBSCAN bar just a little bit visible. Moreover, the K-means value is just 1.06% higher than AGNES, which means they do not differ. DBSCAN performs worst for this index.

Combining all the results of the different indices, no method performs best. Since DBSCAN performs best for the Dunn index and worst for the Calinski-Harabasz and K-means and AGNES do not differ.

5.4.5 Compare best combinations

For each dimensional reduction technique the best clustering method is determined, if there was a significant best combination. The next step is to compare these combinations to determine the best combination of dimensionality reduction and clustering for the data set used in this research.

Combination	Silhouette	Dunn	Calinski-Harabasz	Davies-Bouldin
t-SNE VS PCA	15.49% larger	355.64% larger	47.49% larger	13.04% smaller
t-SNE VS ICA	3.14% smaller	81.28% smaller	12100% larger	38.43% smaller
ICA VS PCA	11.97% larger	95.89% smaller	98.79% smaller	41.24% larger

Table 7: Comparison of the best combinations, where t-SNE and PCA are in combination with K-means and ICA with DBSCAN.

In Table 7 the results of the comparison of the best performing combinations per dimension reduction technique are showed. For the Silhouette index t-SNE with K-means performs better in comparison with the other combinations. For the Dunn index, ICA with DBSCAN performs best and PCA with K-means worst. Moreover, for the Calinski-Harabasz t-SNE with K-means perfors way better than the other combinations. Finally, the Davies-Bouldin does not appear to have a best combination. All together, t-SNE in combination with K-means performs best for the dataset used in this research.

6 Conclusion

In the paper of Renjith et al. (2019), four different data reduction methods are used to reduce the dimension of a dataset, before clustering the reduced data with K-means and AGNES. The goal of the research is to get the combination which generates the best results in clustering complex datasets. In this paper, the paper of Renjith et al. (2019) is replicated and extended by adding DBSCAN to the clustering methods options and analyze if DBSCAN outperforms the other two clustering methods. The research question addressed is: **"What combination of clustering algorithms and dimensionality reduction techniques yields the most accurate results in clustering complex datasets and does DBSCAN outperform K-means and AGNES in clustering data?".**

The results of this research show that the combination of t-SNE with K-means clustering performs the best for the specific dataset used in this research, which consisted of jokes rated on a scale from -10 to 10. This combination showed the best performance compared to other clustering methods, including AGNES and DBSCAN, and dimensionality reduction techniques, including PCA (Principal Component Analysis), ICA (Independent Component Analysis), and LLE (Locally Linear Embedding). To come to this conclusion, first the clustering method that performs best for each dimension reduction technique is chosen, based on the graphs and percentual differences between the clustering methods of four indices (Silhouette, Dunn, Calinski-Harabasz and Davies-Bouldin). For PCA and t-SNE is K-means the best clustering method, for ICA this is DBSCAN and LLE does not appear to have a clustering method that outperforms the other methods. Then the best combinations for each dimension reduction technique are compared and the t-SNE in combination with K-means appeared to perform the best.

The results of this research contradict the results found in the paper of R. W. Sembiring et al. (2011), where five dimension reduction techniques in combination with different clustering methods are evaluated, including DBSCAN. The best performance was seen when using DBSCAN as a clustering method in combination with KernelPCA. In this particular research, DBSCAN does not perform that good, which is a interesting result compared to the research of R. W. Sembiring et al. (2011). The results of this research do confirm the results of Liu et al. (2020) that K-means in combination with t-SNE is a good combination. In Liu et al. (2020), the use of a K-means version in combination with t-SNE improves the clustering results. The results of that research demonstrate that the application of the K-means++ algorithm with t-SNE dimension reduction successfully clusters transformer districts, revealing important insights into their similarities and patterns. The combination of K-means with t-SNE is also the best combination of a dimensional reduction technique in combination with a clustering method in this research, based on the different methods used in this research. By using t-SNE and K-means clustering, valuable information can be extracted from this big dataset, which results in an improved decision-making process. The dimensionality reduction offered by t-SNE ensures effective processing of data and the clustering performed by K-means identifies different groups and patterns within the dataset, based on different characteristics. For further research investigating the performance of the same methods and techniques on different types of datasets

should be done. By doing this, valuable insights into their generalizability and applicability could be provided. On top of that, investigating more indices can boost the understanding of the strengths and weaknesses of different clustering approaches.

References

Assi, K. (2020). Traffic crash severity prediction—a synergy by hybrid principal component analysis and machine learning models. *International journal of environmental research and public health*, *17*(20), 7598.

Charrad, M., Ghazzali, N., Boiteau, V. & Niknafs, A. (2014). Nbclust: an r package for determining the relevant number of clusters in a data set. *Journal of statistical software*, *61*, 1–36.

De Ridder, D. & Duin, R. P. (2002). Locally linear embedding for classification. *Pattern Recognition Group, Dept. of Imaging Science & Technology, Delft University of Technology, Delft, The Netherlands, Tech. Rep. PH-2002-01*, 1–12.

Ghojogh, B., Ghodsi, A., Karray, F. & Crowley, M. (2020). Locally linear embedding and its variants: Tutorial and survey. *arXiv preprint arXiv:2011.10925*.

Hahsler, M., Piekenbrock, M. & Doran, D. (2019). dbscan: Fast density-based clustering with r. *Journal of Statistical Software*, *91*, 1–30.

Hai, V. V., Nhung, H. L. L. L. & Jasek, R. (2022). Toward applying agglomerative hierarchical clustering in improving the software development effort estimation. In *Software engineering perspectives in systems: Proceedings of 11th computer science on-line conference 2022, vol. 1* (pp. 353–371).

Hyvärinen, A. & Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural networks*, *13*(4-5), 411–430.

Ji, G. (2017). A big data decision-making mechanism for food supply chain. , *100*, 02048. doi: 10.1051/MATECCONF/201710002048

Jolliffe, I. (2002). Principal component analysis, 2nd edn new york. *NY: Springer.[Google Scholar]*.

Khan, K., Rehman, S. U., Aziz, K., Fong, S. & Sarasvady, S. (2014). Dbscan: Past, present and future. In *The fifth international conference on the applications of digital information and web technologies (icadiwt 2014)* (pp. 232–238).

Krijthe, J. H. (2015). Rtsne: T-distributed stochastic neighbor embedding using barnes-hut implementation [Computer software manual]. Retrieved from `https://github.com/jkrijthe/Rtsne` (R package version 0.16)

Kuhn & Max. (2008). Building predictive models in r using the caret package. *Journal of Statistical Software*, *28*(5), 1–26. Retrieved from `https://www.jstatsoft.org/index.php/jss/article/view/v028i05` doi: 10.18637/jss.v028.i05

Kune, R., Konugurthi, P. K., Agarwal, A., Chillarige, R. R. & Buyya, R. (2016). The anatomy of big data computing. *Software: Practice and Experience*, *46*(1), 79–105.

Liberti, L., Lavor, C., Maculan, N. & Mucherino, A. (2014). Euclidean distance geometry and applications. *SIAM review*, *56*(1), 3–69.

Linderman, G. C., Rachh, M., Hoskins, J. G., Steinerberger, S. & Kluger, Y. (2017). Efficient algorithms for t-distributed stochastic neighborhood embedding. *arXiv preprint arXiv:1712.09005*.

Liu, L.-p., Wang, Q., Dong, M.-n., Zhang, Z.-y., Li, Y., Wang, Z.-z. & Wang, S.-q. (2020). Application of k-means++ algorithm based on t-sne dimension reduction in transformer district clustering. In *2020 asia energy and electrical engineering symposium (aeees)* (pp. 74–78).

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M. & Hornik, K. (2022). cluster: Cluster analysis basics and extensions [Computer software manual]. Retrieved from `https://CRAN.R-project.org/package=cluster` (R package version 2.1.4 — For new features, see the 'Changelog' file (in the package source))

McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. & Barton, D. (2012). Big data: the management revolution. *Harvard business review*, *90*(10), 60–68.

Nath, A. & Nema, S. (n.d.). Clustering visualization and class prediction using flask of benchmark dataset for unsupervised techniques in machine learning.

Ncir, C.-E. B., Hamza, A. & Bouaguel, W. (2021). Parallel and scalable dunn index for the validation of big data clusters. *Parallel Computing*, *102*, 102751.

Noviantoro, T. & Huang, J.-P. (2022). Investigating airline passenger satisfaction: Data mining method. *Research in Transportation Business & Management*, *43*, 100726.

R Core Team. (2013). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from `http://www.R-project.org/` (ISBN 3-900051-07-0)

Renjith, S., Sreekumar, A. & Jathavedan, M. (2019). A comparative analysis of clustering quality based on internal validation indices for dimensionally reduced social media data. In *International conference on artificial intelligence and data engineering* (pp. 1047–1065).

Renjith, S., Sreekumar, A. & Jathavedan, M. (2020). Performance evaluation of clustering algorithms for varying cardinality and dimensionality of data sets. *Materials Today: Proceedings*, *27*, 627–633.

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.

Sembiring, R. (2011). Dimension reduction of health data clustering. *ArXiv*, *abs/1110.3569*.

Sembiring, R. W., Zain, J. M. & Embong, A. (2011). Alternative model for extracting multidimensional data based-on comparative dimension reduction. In *Icsecs (2)* (pp. 28–42).

Tang, B., Shepherd, M., Heywood, M. I. & Luo, X. (2005). Comparing dimension reduction techniques for document clustering. In *Advances in artificial intelligence: 18th conference of the canadian society for computational studies of intelligence, canadian ai 2005, victoria, canada, may 9-11, 2005. proceedings 18* (pp. 292–296).

Tempola, F. & Assagaf, A. F. (2018). Clustering of potency of shrimp in indonesia with k-means algorithm and validation of davies-bouldin index. In *International conference on science and technology (icst 2018)* (pp. 730–733).

Thomas, J. C. R., Santos, M. & Cofre, M. M. (2014). New version of davies-bouldin index for clustering validation based on hyperrectangles. *vol*, *1*, 2014.

Tibshirani, R. (2012). *Distances between clustering, hierarchical clustering.* Department of Statistics and Machine Learning Department Carnegie Mellon ....

Wang, X. & Xu, Y. (2019). An improved index for clustering validation based on silhouette index

and calinski-harabasz index. In *Iop conference series: Materials science and engineering* (Vol. 569, p. 052024).

Wijuniamurti, S., Nugroho, S. & Rachmawati, R. (2022). Agglomerative nesting (agnes) method and divisive analysis (diana) method for hierarchical clustering on some distance measurement concepts. *Journal of Statistics and Data Science*, *1*(1), 7–11.

A Programming code

```
library(poLCA)
library(readr)
library(dplyr)
library(readxl)
library(NbClust)
library(stats)
library(cluster)
library(caret)
library(fastICA)
library(ggpubr)
library(factoextra)
library(Rtsne)
library(snowfall)
library(snow)
library(lle)
library(dbscan)
library(clValid)
library(fpc)
library(clusterCrit)


# select the cases where there are no missing values
id <- which(Jester[,1] == 100)

# make a subset of the complete cases and delete the first
column with user values
Jester_subset <- Jester[id, 2:101]

# Set seed for reproducibility
set.seed(123)

#create a subsample of 3500 rows and the 100 remaining variables
subsample <- Jester_subset[sample(nrow(Jester_subset), 3500), ]
```

```r
#perform NbClust with all indices
result <- NbClust(subsample, diss = NULL, distance = "euclidean"
, min.nc = 2, max.nc = 10, method = "kmeans", index = "all")
#for the Hubert plot, change "All" with "Hubert"
#same for Dindex

#get the number of clusters for each index
best_nc <- result$Best.nc
print(best_nc)

#perform k-means and plot
clustering_result_kmeans <- kmeans(subsample,
centers = 2, nstart = 15)

fviz_cluster(clustering_result_kmeans,
data = subsample, geom = "point")

print(clustering_result_kmeans)



#-----------------------------------------------------------------------

# Perform agglomerative hierarchical clustering (AGNES)

clustering_result_agnes <- agnes(subsample, method = "ward")

# Extract the desired number of clusters
k <- 2
clusters_agnes <- cutree(clustering_result_agnes, k)

#plot the clusters
fviz_cluster(list(data = subsample, cluster = clusters_agnes)
, geom = "point")

print(clustering_result_agnes)

#-----------------------------------------------------------------------
#perform dbscan and plot
#choose optimal eps
kNNdistplot(subsample, k = 5)
```

```r
clustering_result_dbscan <- dbscan(subsample, eps = 55, MinPts = 5)

num_clusters <- length(unique(clustering_result_dbscan$cluster)) -1

fviz_cluster(clustering_result_dbscan, data = subsample, geom = "point")



distance_matrix <- dist(subsample)


#-----------------------------------------------------------------------
#PCA

pca_result <- prcomp(subsample, center= TRUE, scale = TRUE)

#access the principal components and take the first 50
reduced_data <- pca_result$x[, 1:2]

#KMEANS WITH PCA
clustering_result_kmeans_pca <- kmeans(reduced_data, centers = 2)

fviz_cluster(clustering_result_kmeans_pca,
data = reduced_data, geom = "point")

print(reduced_data)


#AGNES WITH PCA
clustering_result_agnes_pca <- agnes(reduced_data,
method = "ward")

# Extract the desired number of clusters
k <- 2
clusters_agnes_pca <- cutree(clustering_result_agnes_pca, k)


#plot the clusters
fviz_cluster(list(data = reduced_data,
cluster = clusters_agnes_pca), geom = "point")
```

```r
#DBSCAN with PCA
kNNdistplot(reduced_data, k = 5)
abline(h = 1.60, col = "red", lty = 2)
clustering_result_dbscan_pca <- dbscan(reduced_data,
eps = 1.60, MinPts = 5)

num_clusters <-
length(unique(clustering_result_dbscan_pca$cluster)) -1

fviz_cluster(clustering_result_dbscan_pca,
data = reduced_data, geom = "point")


#-------------------------------------------------------------------------
#KMEANS WITH ICA

# Perform ICA on the data
ica_model <- preProcess(subsample, method = "ica", n.comp = 2)
ica_data <- predict(ica_model, subsample)

# Perform k-means clustering on the transformed data
clustering_result_kmeans_ica <- kmeans(ica_data,
centers = 2, iter.max = 20, nstart = 5)
#plot the clusters
fviz_cluster(clustering_result_kmeans_ica,
data = ica_data, geom = "point")



#AGNES with ICA

clustering_result_agnes_ica <- agnes(ica_data, method = "ward")
# Extract the desired number of clusters
k <- 2
clusters_agnes_ica <- cutree(clustering_result_agnes_ica, k)

fviz_cluster(list(data = ica_data, c
luster = clusters_agnes_ica), geom = "point")
```

```
#DBSCAN with ICA
kNNdistplot(ica_data, k = 5)
abline(h = 0.45, col = "red", lty = 2)
clustering_result_dbscan_ica <- dbscan(ica_data, eps = 0.45, MinPts = 5)

num_clusters <- length(unique(clustering_result_dbscan_ica$cluster)) -1

fviz_cluster(clustering_result_dbscan_ica,
data = ica_data, geom = "point")


#-------------------------------------------------------------------------


tsne_result <- Rtsne(subsample, dims = 2, perplexity = 50,
theta = 0.5, max_iter = 3000)
tsne_coordinates <- tsne_result$Y

#make a frame with column names so we can apply fviz_cluster
tsne_data <- data.frame(V1 = tsne_coordinates[, 1],
V2 = tsne_coordinates[, 2])

# Perform k-means clustering on the t-SNE coordinates
clustering_result_kmeans_tsne <- kmeans(tsne_data,
centers = 2, iter.max = 15, nstart = 10)

#plot cluster
fviz_cluster(clustering_result_kmeans_tsne,
data = tsne_data, geom = "point")




#t-SNE with AGNES

clustering_result_agnes_tsne <- agnes(tsne_data, method = "ward")
# Extract the desired number of clusters
k <- 2
clusters_agnes_tsne <- cutree(clustering_result_agnes_tsne, k)

fviz_cluster(list(data = tsne_data,
```

```
cluster = clusters_agnes_tsne), geom = "point")

#DBSCAN with tsne
kNNdistplot(tsne_data, k = 5)
abline(h = 1.20, col = "red", lty = 2)
clustering_result_dbscan_tsne <- dbscan(tsne_data,
eps = 1.20, MinPts = 5)

num_clusters <- length(unique(clustering_result_dbscan_tsne$cluster)) -1

fviz_cluster(clustering_result_dbscan_tsne,
data = tsne_data, geom = "point")

#LLE
lle_result <- lle(subsample, m = 2, k = 10, reg = 0.1,
ss = FALSE, p = 2, id = FALSE, nnk = TRUE, eps = 0.0001,
iLLE = FALSE, v = 0.99)


# Get the embedded coordinates
lle_coordinates <- lle_result$Y

lle_data <- data.frame(V1 = lle_coordinates[, 1],
V2 = lle_coordinates[, 2])


#LLE with KMEANs
clustering_result_kmeans_lle <- kmeans(lle_data, centers = 2)

# Plot the clusters
fviz_cluster(clustering_result_kmeans_lle,
data = lle_data, geom = "point")



#LLE with AGNES
clustering_result_agnes_lle <- agnes(lle_data, method = "ward")
# Extract the desired number of clusters
k <- 2
clusters_agnes_lle <- cutree(clustering_result_agnes_lle, k)

fviz_cluster(list(data = lle_data, cluster = clusters_agnes_lle)
```

```
, geom = "point")

#DBSCAN with lle
kNNdistplot(lle_data, k = 5)
abline(h = 0.15, col = "red", lty = 2)
clustering_result_dbscan_lle <- dbscan(lle_data,
eps = 0.15, MinPts = 5)

num_clusters <-
length(unique(clustering_result_dbscan_lle$cluster)) -1

fviz_cluster(clustering_result_dbscan_lle,
data = lle_data, geom = "point")

#CLUSTERING VALIDATION VOOR ELKE CLUSTERMETHODE


#clusterCrit KMEANs
subsample_matrix <- as.matrix(subsample)
indices_kmeans <- intCriteria(subsample_matrix,
clustering_result_kmeans$cluster, crit = "all")

reduced_data_matrix <- as.matrix(reduced_data)
indices_kmeans_pca <- intCriteria(reduced_data_matrix,
clustering_result_kmeans_pca$cluster, crit = "all")

ica_data_matrix <- as.matrix(ica_data)
indices_kmeans_ica <- intCriteria(ica_data_matrix,
clustering_result_kmeans_ica$cluster, crit = "all")

tsne_data_matrix <- as.matrix(tsne_data)
indices_kmeans_tsne <- intCriteria(tsne_data_matrix,
clustering_result_kmeans_tsne$cluster, crit = "all")

lle_data_matrix <- as.matrix(lle_data)
indices_kmeans_lle <- intCriteria(lle_data_matrix,
clustering_result_kmeans_lle$cluster, crit = "all")


#clustercrit AGNES

indices_agnes_pca <- intCriteria(reduced_data_matrix,
```

```
clusters_agnes_pca, crit = "all")

indices_agnes_ica <- intCriteria(ica_data_matrix,
clusters_agnes_ica, crit = "all")

indices_agnes_tsne <- intCriteria(tsne_data_matrix,
clusters_agnes_tsne, crit = "all")

indices_agnes_lle <- intCriteria(lle_data_matrix,
clusters_agnes_lle, crit = "all")

#clustercrit DBSCAN

#change to vectors
clustering_dbscan_pca <- as.integer(clustering_result_dbscan_pca$cluster)
clustering_dbscan_ica <- as.integer(clustering_result_dbscan_ica$cluster)
clustering_dbscan_tsne <- as.integer(clustering_result_dbscan_tsne$cluster)
clustering_dbscan_lle <- as.integer(clustering_result_dbscan_lle$cluster)

indices_dbscan_pca <- intCriteria(reduced_data_matrix,
clustering_dbscan_pca, crit = "all")

indices_dbscan_ica <- intCriteria(ica_data_matrix,
clustering_dbscan_ica, crit = "all")

indices_dbscan_tsne <- intCriteria(tsne_data_matrix,
clustering_dbscan_tsne, crit = "all")

indices_dbscan_lle <- intCriteria(lle_data_matrix,
clustering_dbscan_lle, crit = "all")



#maak grafieken om de indices te vergelijken

dunn_index <- data.frame(
  Dimensionality_Reduction = rep(c("PCA", "ICA",
  "t-SNE", "LLE"), each = 3),
  Method = rep(c("K-means", "AGNES", "DBSCAN"), times = 4),
  Dunn_Index = c(0.00133, 0.00316, 0.03413, 0.00099,
  0.00320, 0.03237, 0.00606, 0.00913, 0.02847, 0.00149, 0.00192, 0.00425)
)
```

```
# Maak het staafdiagram met horizontale staven
ggplot(dunn_index, aes(x = Dunn_Index,
y = Dimensionality_Reduction, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Dunn Index", y = "Dimensionality Reduction",
  fill = "Clustering Method") +
  theme_bw()

 silhouette_index <- data.frame(
  Dimensionality_Reduction = rep(c("PCA", "ICA",
  "t-SNE", "LLE"), each = 3),
  Method = rep(c("K-means", "AGNES", "DBSCAN"),
  times = 4),
  Silhouette_Index = c(0.42997, 0.43526, 0.36705,
  0.30505, 0.28270, 0.48144, 0.49658, 0.49536, 0.34964,
  0.50260, 0.51896, 0.30837)
)

# Maak het staafdiagram met horizontale staven
  ggplot(silhouette_index, aes(x = Silhouette_Index,
  y = Dimensionality_Reduction, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Silhouette Index", y = "Dimensionality Reduction",
  fill = "Clustering Method") +
  theme_bw()

  calinski_index <- data.frame(
  Dimensionality_Reduction = rep(c("PCA", "ICA",
  "t-SNE", "LLE"), each = 3),
  Method = rep(c("K-means", "AGNES", "DBSCAN"),
  times = 4),
  Calinski_Index = c(3805, 2597, 30, 1628, 1439, 46,
  5612, 4732, 5, 2090, 2068, 192)
)

# Maak het staafdiagram met horizontale staven
ggplot(calinski_index, aes(x = Calinski_Index,
y = Dimensionality_Reduction, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Calinski-Harabasz Index", y = "Dimensionality Reduction",
  fill = "Clustering Method") +
```

```
  theme_bw()


davies_index <- data.frame(
  Dimensionality_Reduction = rep(c("PCA", "ICA",
  "t-SNE", "LLE"), each = 3),
  Method = rep(c("K-means", "AGNES", "DBSCAN"), times = 4),
  Davies_Index = c(0.83211,0.80516, 2.66480, 1.25371, 1.32186
  ,     1.17535, 0.72362, 0.71444, 6.02692      , 0.83190, 0.79463, 1.93398)
)

# Maak het staafdiagram met horizontale staven
ggplot(davies_index, aes(x = Davies_Index,
y = Dimensionality_Reduction, fill = Method)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Davies-Bouldin Index", y = "Dimensionality Reduction",
  fill = "Clustering Method") +
  theme_bw()
```