**MEMORY EFFICIENT ONLINE HIERARCHICAL CLUSTERING BASED ON OVERSAMPLING**

**Samed ARSLAN**

**APRIL 2022**

**MEMORY EFFICIENT ONLINE HIERARCHICAL CLUSTERING BASED ON OVERSAMPLING**

**A THESIS SUBMITTED TO THE**

**GRADUATE SCHOOL**

**OF**

**BAHÇEŞEHİR UNIVERSITY**

**BY**

**SAMED ARSLAN**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS**

**FOR**

**THE DEGREE OF MASTER OF SCIENCE**

**IN THE DEPARTMENT OF ARTIFICIAL INTELLIGENCE**

**APRIL 2022**



**T.C.**

**BAHÇEŞEHİR UNIVERSITY**

**GRADUATE SCHOOL**

...../....../.....

**MASTER THESIS APPROVAL FORM**

|  |  |
| --- | --- |
| **Program Name:** |  |
| **Student's Name and Surname:** |  |
| **Name of The Thesis:** |  |
| **Thesis Defense Date** |  |

This thesis has been approved by the Graduate School which has fulfilled the necessary conditions as Master thesis.

**Prof. Dr. Fatma ÖZKUL Institute Director**

This thesis was read by us, quality and content as a Master's thesis has been seen and accepted as sufficient.

|  |  |  |
| --- | --- | --- |
|  | **Title, Name** | **Signature** |
| **Thesis Advisor:** |  |  |
| **2. Member:** |  |  |
| **3. Member:** |  |  |

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Surname : Samed ARSLAN

Signature :

# ABSTRACT

MEMORY EFFICIENT ONLINE HIERARCHICAL CLUSTERING BASED ON OVERSAMPLING

Arslan, Samed

Artificial Intelligence Masters Program

Supervisor: Asst. Prof. Dr. Cemal Okan Şakar

April 2022, 52 pages

With the rapid development and spread of technology, it has become difficult for classical algorithms to respond to current needs. Known weaknesses of some algorithms have become more obstacles. It is a known situation that some of the clustering algorithms work dependent on the similarity matrix. However, this allows working within the limits of memory resources. Algorithms cannot work when the data size increases. Although the classical hierarchical clustering algorithm is one of the most well-known and successful algorithms, which has these limitations.

The proposed method will aim to overcome this weakness of the algorithm. This approach, which works in batches, will allow the algorithm to be updated online when a new data set arrives, it allows real-time cluster assignment of new observations. In short, the proposed method, processes the data in batches and eliminates the sudden load on the memory, unlike the classical algorithm. Each data batch is analyzed with classical algorithm, it is organized and reduced in size with the help of the SMOTE algorithm. Stacks are included in the master dataset, preserving a smaller volume of information. Thus, the cluster information of the main set is preserved and minimized. Assignments are made according to the clusters determined in the last stage.

**Keywords:** Hierarchical Clustering, SMOTE, Online Clustering, Sampling

# ÖZ

ÖRNEKLEMEYE DAYALI HAFIZA VERİMLİ ÇEVİRİMİÇİ HİYERARŞİK KÜMELEME

Arslan, Samed

Yapay Zeka Yüksek Lisans Programı

Tez Danışmanı: Yrd. Prof. Dr. Cemal Okan Şakar

Nisan 2022, 52 sayfa

Teknolojinin hızla gelişmesi ve yaygınlaşması ile birlikte klasik algoritmaların güncel ihtiyaçlara cevap vermesi zorlaştı. Bazı algoritmaların bilinen zaafları daha fazla engel oluşturur hale geldi. Kümeleme algoritmalarının bir kısmı benzerlik matrisine bağımlı çalıştığı bilinen bir durumdur. Ancak bu durum hafıza kaynaklarının sınırları dahilinde çalışmaya olanak sağlamaktadır. Veri boyutu arttığında algoritmalar çalışmamaktadır. Klasik hiyerarşik kümeleme algoritması da en bilinen ve başarılı algoritmalardan biri olmasına rağmen bu kısıtlara sahip bir algoritmadır.

Önerilen metod, algoritmanın bu zaafının üstesinden gelmeye yönelik olacaktır. Bununla birlikte batch çalışan bu algoritmanın yeni veri seti geldiğinde çevrimiçi olarak güncellemesine imkân sağlayacak olup, aynı şekilde, yeni gözlemlerin de gerçek zamanlı küme atamasının yapılmasına imkân vermektedir. Önerilen metot kısaca, klasik algoritmadan farklı olarak, veriyi yığınlar halinde işleyerek hafızaya ani yüklenme durumunu ortadan kaldırmaktadır. Her bir yığın, basit yöntem ile analiz edildikten sonra SMOTE algoritması yardımı ile düzenlenir ve küçültülür. Yığınlar daha küçük hacimde bilgilerini koruyarak ana veri setine dahil edilir. Böylelikle ana setin küme bilgisi korunarak küçültülmüş olur. En son aşamada tespit edilen kümelere göre atamalar gerçekleştirilir.

**Anahtar Kelimeler:** Hiyerarşik Kümeleme, SMOTE, Çevirimiçi Kümeleme, Örnekleme

# ACKNOWLEDGEMENTS

I would like to present my sincere gratitude to Assist. Prof. Dr. Cemal Okan Şakar for his continuous support, mentorship and motivations during the entire research effort and my studies. I am also thankful to my wife who has always been there supporting and motivating me in the challenges I encountered during the journey.

# TABLE OF CONTENTS

[ABSTRACT iv](#_Toc100581243)

[ÖZ v](#_Toc100581244)

[ACKNOWLEDGEMENTS vi](#_Toc100581245)

[TABLE OF CONTENTS vii](#_Toc100581246)

[1. Introduction 1](#_Toc100581247)

[2. Related Works 3](#_Toc100581248)

[2.1. Hierarchical Clustering on Very Large Datasets 3](#_Toc100581249)

[2.2. Synthetic Minority Over-Sampling Technique (Smote) 7](#_Toc100581250)

[3. Materials and Methods 8](#_Toc100581251)

[3.1. Dataset Description 8](#_Toc100581252)

[3.2. Hierarchical Clustering 16](#_Toc100581253)

[3.3. Synthetic Minority Over-Sampling Technique (Smote) 17](#_Toc100581254)

[3.4. Validation and Measuring Methods 17](#_Toc100581255)

[**Performance Metrics** 17](#_Toc100581256)

[3.5. Propose Method: Hierarchical Clustering with Chunks Via SMOTE (HCCS) 23](#_Toc100581257)

[4. Statistical Test 24](#_Toc100581258)

[5. Experiments 25](#_Toc100581259)

[Experimental Setup 25](#_Toc100581260)

[Results 27](#_Toc100581261)

[6. Conclusion and Discussion 50](#_Toc100581262)

[REFERENCES 51](#_Toc100581263)

**Chapter 1**

# Introduction

Clustering techniques are one of the most used machine learning subcategories in the business world. According to the KDnuggets 2018/2019 Survey [1]; “Clustering” takes the 3rd place after “regression” and “decision trees”. Today, they are actively used in different sectors such as, consumer analytics, finance, healthcare, fraud detection, etc.

The main idea behind clustering is the grouping of objects[2] and this approach may be necessary in different situations. Unsupervised methods are more useful to supervised methods in some cases. But they are harder to apply too.[3] For example, in the fraud detection area, detected fraudsters were tagged as a fraudulent asset group. Behavioral information and labels can be used as a training dataset by a supervised algorithm and classification can be made with this information. Labels are very useful in such cases, and some algorithms can produce very easy outputs. In supervised situations, information is hidden among the characteristics of labeled groups and can be inferred by analysis of similarities in group observations and differences between other groups.[4] In this example, most fraud scenarios can be deduced from fraudulent data using detailed analysis tasks or machine learning methods. Without the need for complex algorithms, the desired results can be achieved in a more understandable way with simple filtering methods. In the real world, labels are available in very few cases [5]. Where provided situations, the data is generated in line with the decisions taken by the business units which are responsible for investigating possible fraud cases. This causes some problems. First, the decisions taken by the business units that make the examination affect the results of that day's attention level[6]. Work experience is linked to the quality of decisions taken. The filtering methods that provide the review content also determine the framework of the diversity here. Apart from these, the way the data is kept and its consistency can also be considered as external factors. Therefore, the fraud area tends to have a high rate of false positive detected cases, and some fraudulent cases stay undetected. For this reason, labels may be questioned in some cases. The implementation of data quality and validation studies requires a long time and cost. In these cases, an unsupervised approach becomes best practice. Clustering algorithms do not need tags. In general, they work according to the distance of the observations from each other

Clustering algorithms are divided into hierarchical clustering and non-hierarchical clustering methods. [7]

Hierarchical clustering (HC) has a good place among clustering algorithms.[8] And there are many use cases in different businesses. Both hierarchical clustering methods (divisive and agglomerative) are very successful in detecting hierarchical data structures. In particular, visualization of outputs as dendrograms; makes it understandable and interpretable. This is the strongest part of the algorithm.

But the algorithm has the same gaps in big and big data due to memory dependencies. Today, data sets are growing in parallel with information technology. And work requirements are likewise increasing. Due to the dependency on the distance matrix, it makes the method inoperable on large datasets. At this point, there are two options; the first is to search for different algorithms, and the second is improving the hierarchical algorithm.

This thesis is about a heap method for algorithms. Most similar methods focus on the centers of the clusters or mini sample the pieces for a final clustering. After this combination, most of the time information about the cluster distribution is lost. But the proposed method focuses on transmitting information of each chunk. At the point of transmitting the information, the work to be done proceeds by protecting the information and reducing the data set. The hierarchical clustering method to be used in this approach will be used in the kernel while processing the data set. The information obtained as output from the algorithm will be preserved, and the data set will be under-sampled and included in the main data set for later reuse. With this approach, the data set will be reduced for the final study. After clustering in the last stage, the centers of the clusters will be determined and used to assign the main data set to the clusters. The related work will be tested on many synthetic datasets. The results obtained in different data patterns will be examined and the overall performance will be measured with the tests to be run on the whole set. The performance of the simple hierarchical clustering method and the proposed method will be examined comparatively. The results to be obtained and the improvements that can be made afterwards will be evaluated.

Since the method to be used in the study will ensure that the data is processed in chunks, the step sizes of the work to be done will be balanced with the available memory resources. An alternative method has been designed that can be used when it is not possible to work with big data in working environments with limited memory.

**Chapter 2**

# Related Works

## **Hierarchical Clustering on Very Large Datasets**

In Machine Learning, hierarchical clustering is one of the popular clustering techniques which seeks to build a hierarchy of clusters. Hierarchical clustering is generally divided into two as agglomerative and divisive. Agglomerative clustering uses the bottom-up approach. In other words, it treats each data as a cluster and combines clusters that are close to each other until k clusters are formed. On the other hand, Divisive Clustering is the opposite of agglomerative clustering. This uses the top-down approach. In this clustering method, there is only one cluster at the beginning. It assigns dissimilar clusters to different clusters until K clusters are formed. Hierarchical Clustering and its types are discussed in more detail under the title of "2. Methods".

In one study [9], a hybrid hierarchical clustering algorithm is proposed to capture both far and near views of the hierarchical structure for a large dataset. In this approach, firstly, the data is segmented in the most appropriate way with the k-means algorithm. The exact k-means algorithm was optimized, which utilizes a triangle inequality to avoid unnecessary distance computations.[10] Clusters are randomly started from the data points. Secondly, at the first level, AHC is applied to cluster each Pi component obtained by k-means into a separate detailed Ti tree. In the second level, each Ti is processed as a leaf. A rough tree is then clustered in a tree T by the AHC. For this reason, T shows both the detailed relationship between the members of each component and the rough relationship between the components. In the study, three typical real data used for biomedical research applications were selected. One of them is the activity matrix consisting of 2117 compounds (D1), which can be considered as an example of a medium-sized clustering problem. The other data is a larger activity matrix of 45000 compounds across 178 assays (d2). This is a subset of the larger matrix described in a published HTS frequent hit study [11]. The last one consists of 1 million compounds randomly selected from their in-house compound collection (D3). This data was chosen to enable us to examine the robustness of the hybrid algorithm. For the result of running time and memory analysis, the experimental running time of the hybrid algorithm was measured with different k values for both the partitioning stage and the hierarchical clustering stage (Figure 1). The result reached in this study is "as the data set gets bigger, the running speed of the hybrid algorithm gets shorter." has been. An example is given as “when the parameters are optimized, the hybrid algorithm is only 5 times faster in D1 and 370 times faster in D2, running in 22 seconds compared to 8117 seconds for the full algorithm.” according to the graphic.

Chart

Description automatically generated

*Figure 2.1: “Running time of the hybrid algorithm for datasets D1 and D2. (A) Dataset D1, for all values of k. (B) Dataset D1, zoomed in for k < 500. (C) Dataset D2. (D) Dataset D2, zoomed in for k < 1000.”*

It is specified as "We estimated the probability that a pair of compounds will cluster together in consecutive runs to be 37.1% with a standard deviation of 0.9%15for the hybrid method, and 27.8% with a standard deviation of 1.6% for the Sphere Exclusion (SE) [12] methods (p-value is 1×e-10). Similarly, we also estimated the probability that a pair of compounds will not cluster together in consecutive runs to be 99.8% and 99.9%, respectively. These results indicate the superior robustness of the hybrid algorithms across multiple runs." for the performance result.

In other study [13], the memory problem in hierarchical clustering is tackled by presenting a new general-purpose online hierarchical clustering algorithm called SparseHC. The focus is on reducing the primary memory consumption of the AHC stored matrix approach with SparseHC, a general-purpose memory-eﬃcient AHC algorithm for single-, complete- and average-linkage schemes. SparseHC uses a similar strategy as in MCUPGMA [14] and hcluster [15] where the input distance matrix is ﬁrst sorted and then processed in a chunk-by-chunk manner.

1. Compressing the information in the currently loaded part of the input matrix into the most compact format.

 2. The use of an efficient graphical representation to store unassembled cluster links provides continuous access to these links for faster access.

SparseHC takes a sorted distance matrix D as input. And it iteratively build a dendrogram from taking only a part of D in each iteration step. A sequence of values is built instantly depending on the available memory. Starting from a tree of only N leaves, where one leaf node represents a single cluster, a bottom-up binary tree is constructed. In SparceHC, an undirected weighted graph, which consists of set of V vertices (the nodes of the binary tree) and a set of E edges, is used to model connections between pairs of active clusters for this study.

To streamline the cluster merging process, SparseHC prefers for a graph that provides a constant time to perform edge insertion, edge deletion and edge update with minimal storage for the graph. To assist these operations, the standard neighbor list has been changed and this graph is called an adjacency (hash) map. The adjacency map is a collection of unordered hash maps, one for each vertex of the graph. Each hash map uses the neighboring vertex's identification number as the key and records the set of neighbors of its vertex. SparseHC can use the O(|V|+|E|) space to store all the clusters and their active connections, because of this adjacent map representation. More importantly, these connections can be accessed and updated in O(1) time with better performance than other approaches. In this study, four sparse matrices computed from DNA sequence datasets of size 10000 40000 sequences were used (%50 sparsity). Matrices are calculated using the sequence embedding approach used in the popular Clustal-Omega multiple sequence alignment tool [16]. For the comparing the runtime and memory size, SparseHC, average-linkage clustering (aveclust) and MCUPGMA were used. According to the results, SparceHC was more efficient than other sparse clustering tools.

In one study [17], the main contribution is to introduce an online agglomerative hierarchical technique based on the concept of single-linkage method for clustering symbolic and numeric data. The proposed algorithm also has lower memory size which makes it easier to deal with large datasets. Objects are introduced to the algorithm one by one in online operation. The new object updates the membership matrix at each step. The goal here is to improve the results achieved every time. After the last object, the algorithm must be ready to build the final hierarchical dendrogram. For each object, the nearest k objects are calculated by ordering from the closest to the furthest according to the similarity measure. And the nearest k objects are constantly updated after a new object arrives. As the final object, since there already exists a group of objects and their nearest k objects, they are stored in ascending order according to their dissimilarity, or in descending order when the similarity measure is used to create a set of pairs.

According to the results, the memory required by the proposed algorithm is much lower than any traditional hierarchical technique. For 1000 objects, the memory required for proposed algorithm is 8000 memory cells assuming k=8, while the memory cells required by any traditional hierarchical technique is at least 499,500.

## **Synthetic Minority Over-Sampling Technique (Smote)**

In one study [18], a discretization method called Optimal Equal Width Binning and an oversampling technique known as Synthetic Minority Oversampling (SMOTE) were used to improve the accuracy of students' final grade prediction model. The number of students enrolled in a course was used for the related study and the total number of students was 181. The final grade was considered the class label. The same dataset was used to create the oversampled dataset where the sample count is 360. In the data, there are 5 classes which are called A, B, C, D and F. There were 28 students in Class A, 76 students in B, 48 students in C, 18 students in D and 10 students in F. The imbalance problem was solved with SMOTE and the counts for each class label were almost equalized (A:70, B:76, C:72, D:72, F:70). Naive Bayes, Decision Tree and Neural Network Algorithms were used in this study. These algorithms were applied for both original data and SMOTE data and for all classes separately. According to the results, Naive Bayes provided 61% accuracy when using data without Smote, while this accuracy increased to 75% with data using Smote. On the other hand, while the neural network algorithm provided 66% accuracy on data without smote, this rate increased to 75% on data using smote. This means that there is a greater accuracy gain when using Naive Bayes classification.

In other study [19], it is aimed to find and compare the classification in analysis sentiments from the perspective of XL customers. For the data, tweets from XL customers posted on myXLCare Twitter account were used. As stated in the related study, “transform case”, “tokenize”, “token filters by length”, “n-gram”, “stemming” methods were used for classification and sentiment analysis. The data taken is only tweets in Indonesian, which is 2000 tweets with the words MyXLCare. Data are taken randomly from either ordinary user accounts or online media accounts on Twitter. According to the results, with the Naive Bayes algorithm, 79% accuracy, 74% sensitivity and 77% recall were achieved when using the original data. On the other hand, the accuracy increased to 86%, sensitivity to 82% and recall to 92% in the data which used the SMOTE method.

**Chapter 3**

# Materials and Methods

## **Dataset Description**

All of the data sets to be used in the study were artificially produced. They have been widely used in literature. And it is a compilation of labeled datasets prepared for different clustering tests. The data sets were previously collected for other studies and were obtained from the "clustering-benchmark" repository of the user named "deric" on github.[20]

The repo contains one hundred and eight different data sets. The number of observations of each data set varies between one hundred and ten thousand. It usually consists of two- sets so that the visualization is also understandable. They contain between two and twenty different classes, as they will be used to measure the performance of different clustering and classification algorithms. Each of them has different pattern properties. Some of the datasets have an unbalanced class problem. Since examining all of the data sets used here one by one will distract attention, ten of datasets will be examined in detail in this study. While selecting these data sets, attention will be paid to ensure diversity according to the performance status of the hierarchical clustering method. Thus, the outputs will be evaluated in the most diverse and comprehensive way.

The data list and features to be examined in this thesis will be as follows

|  |  |  |
| --- | --- | --- |
| **Set Number** | **Dataset Name** | **Row Size** |
| 1 | cluto-t7-10k | 10000 |
| 2 | complex9 | 3031 |
| 3 | dartboard2 | 1000 |
| 4 | dense-disk-5000 | 5000 |
| 5 | donut2 | 1000 |
| 6 | smile3 | 1000 |
| 7 | square1 | 1000 |
| 8 | square2 | 1000 |
| 9 | square3 | 1000 |
| 10 | square4 | 1000 |

*Table 3.1 : Ten artificial test datalist out of ninety-eight which are examined in this work and row numbers*

At this point, the characteristics of the data sets will be briefly examined. It can be evaluated together with its two dimensional visuals as follows.

Data Set number 1: cluto-t7-10k

A picture containing text, vector graphics

Description automatically generated

*Figure 3.1: cluto-t7-10k dataset distribution*

This dataset consists of ten different clusters in two-dimensional space. Each cluster has its own unique distribution pattern. Although clusters can be transitive to each other, a cluster can be defined as noise unlike the others. The number of observations of the clusters may vary. Distribution of clusters are not a fully balanced.

The data set consists of ten thousand observations in total. And it is two-dimensional.

Data Set number 2: complex9

Chart, scatter chart

Description automatically generated

*Figure 3.2: complex9 dataset distribution*

This dataset consists of nine different clusters in two-dimensional space. each cluster has its own unique distribution pattern. sets are not transitive. The number of observations of the clusters may vary. We cannot talk about a fully balanced data set.

The data set consists of three thousand and thirty-one observations in total. And it is two-dimensional.

Data Set number 3: dartboard2

Diagram

Description automatically generated

*Figure 3.3: dartboard2 dataset distribution*

In this data set, it consists of four rings located inside each other in two-dimensional space. Although the rings do not touch each other's region, the inner two rings and the outer two rings have closer values to each other. The observation numbers of each cluster seem close to each other. The dataset has a total of one thousand observations. The dataset is two-dimensional.

Data Set number 4: disk-5000n

Chart, scatter chart

Description automatically generated

*Figure 3.4: disk-5000n dataset distribution*

This data set consists of two different sets in two-dimensional space. The observations are distributed in a circular region in the two-dimensional plane. One of the clusters is in a central region of these observations. The other is located in the form of a ring in the area outside the center. sets are not transitive. The number of observations is evenly distributed among the clusters. The data set consists of five thousand observations in total. And it is two-dimensional.

Data Set number 5: donut2

Chart, scatter chart

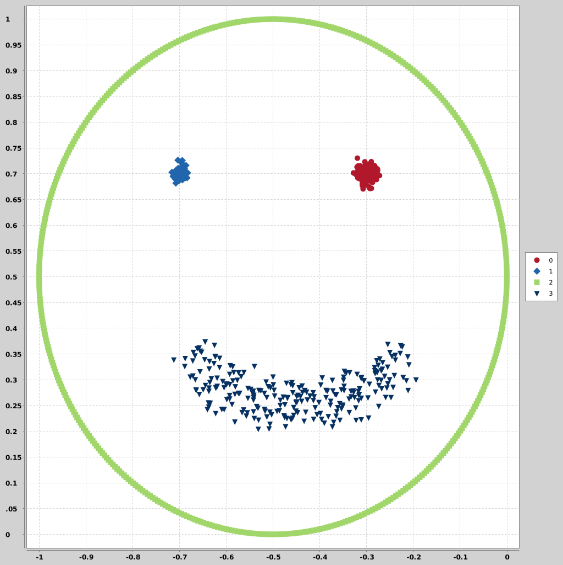
Description automatically generated

*Figure 3.5: donut2 dataset distribution*

This dataset consists of two different sets in two-dimensional space. One of the clusters is distributed in the form of a ring in the two-dimensional plane. The other has a random distribution close to the center in three of the rings. sets are not transitive. The number of observations is evenly distributed among the clusters.

The data set consists of a thousand observations in total. And it is two-dimensional.

Data Set number 6: smile3



*Figure 3.6: smile3 dataset distribution*

This dataset consists of four different clusters in two-dimensional space. One of the clusters is distributed in the form of a ring in the two-dimensional plane. Others have a regional density distribution in three of the rings. sets are not transitive. The number of observations is not equally distributed among the clusters.

The data set consists of a thousand observations in total. And it is two-dimensional.

Data Set number 7: square1

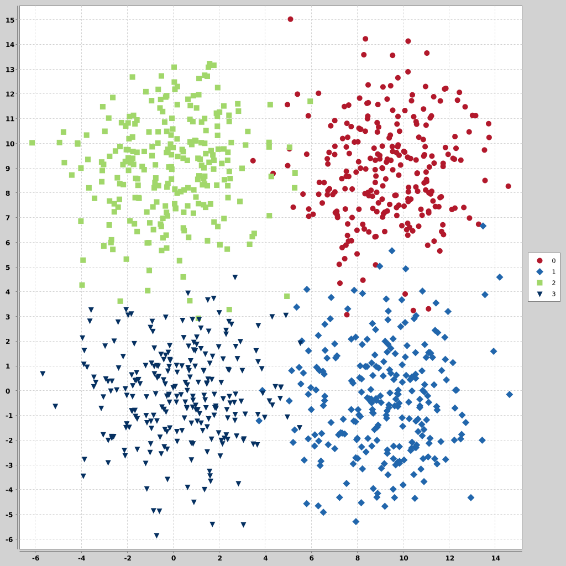
Chart, scatter chart

Description automatically generated

*Figure 3.7: square1 dataset distribution*

In this data set, there are four clusters that are transitive to each other. Although they are in the two-dimensional plane with their densities at the corners, there are also observations that can be included in each other's regions. However, the distance between the classes in the previous sets are lower than in this set and they are more involved in each other's spaces. The numbers of observations seem balanced to each other. It contains a thousand observations in total. As seen, it is a two-dimensional data set.

Data Set number 8: square2



*Figure 3.8: square2 dataset distribution*

In this data set, there are four clusters that are transitive to each other. Although they are in the two-dimensional plane with their densities at the corners, there are also observations that can be included in each other's regions. This set looks similar to the “square1 ” set before. however, the distance between the classes in the previous set is lower than in this set and they are more involved in each other's spaces. The numbers of observations seem balanced to each other. It contains a thousand observations in total. As seen, it is a two-dimensional data set.

Data Set number 9: square3

Chart, scatter chart

Description automatically generated

*Figure 3.9: square3 dataset distribution*

In this data set, there are four clusters that are transitive to each other. Although they are in the two-dimensional plane with their densities at the corners, there are also observations that can be included in each other's regions. This set looks similar to the “square1” and “square2” set before. however, the distance between the classes in the previous set is lower than in this set and they are more involved in each other's spaces. The numbers of observations seem balanced to each other. It contains a thousand observations in total. As seen, it is a two-dimensional data set.

Data Set number 10: square4

Chart, scatter chart

Description automatically generated

*Figure 3.10: square4 dataset distribution*

In this data set, there are four clusters that are transitive to each other. Although they are located in the two-dimensional plane with their densities at the corners, there are also observations that can be included in each other's regions. This set looks similar to the “square1”, “square2” and “square3” set before. These datasets were specifically selected for the examination of the transition effect. The numbers of observations seem balanced to each other. It contains a thousand observations in total. As seen, it is a two-dimensional data set.

## **Hierarchical Clustering**

The hierarchical clustering algorithm is one of the clustering methods. It is an iterative approach. As in every clustering method, it is used to group the observations that are most similar to each other in the data set within the determined constraints. It performs clustering using two different streams. The first method is agglomerative clustering. In this method, each element in the data set is considered as a set. Among these sets, the most similar sets are combined. The other method is divisive clustering. In this method, we start with a single set and have two elements that are least similar to each other. The whole dataset is divided into two clusters according to their proximity to these two elements. These processes are repeated for each subset that is formed. The processes are repeated until the desired number of clusters or a predetermined total similarity value is reached.

## **Synthetic Minority Over-Sampling Technique (Smote)**

SMOTE (Synthetic Minority Over-Sampling Technique) is an oversampling process that produces synthetic data. It is one of the most commonly used methods in data science projects.

The main idea of the method is to create new instances of the minority class by performing certain operations between instances of the minority class.

Synthetic samples are produced as follows:

The difference between the examined feature vector (𝐸𝑖) and its nearest neighbor is taken,

Multiply this difference by a random number (𝛿) between 0 and 1,

The result is added to the examined feature vector and a new sample is created.

Depending on the amount of oversampling required, neighbors from the k nearest neighbors are randomly selected. This process avoids the over-learning problem and presents it with good classification performance.

## **Validation and Measuring Methods**

### **Performance Metrics**

#### **Accuracy**

Accuracy value is calculated by the ratio of those predicted correctly in the model to the total data set. It is one of the simplest measures of a model. Confusion matrix is used to calculate accuracy. In a confusion matrix there are 4 cases in the confusion matrix:

|  |  |  |
| --- | --- | --- |
| **Predicted**  **Actual** | ***Negative*** | ***Positive*** |
| ***Negative*** | TN | FP |
| ***Positive*** | FN | TP |

*Table 3.2 : confusion matrix*

* True Positive (TP): Shows the number of positive examples classified accurately.
* False Positive (FP): Shows the number of actual negative examples classified as positive.
* False Negative (FN): Shows the number of actual positive examples classified as negative.
* True Negative (TN): Shows the number of negative examples classified accurately.

The Accuracy is determined using the equation from a confusion matrix:

If there is an imbalanced data, accuracy can be misleading. Therefore, the other metrics (Precision, Recall, etc.) can be useful for evaluating performance.

#### **Kappa**

Kappa number is a ratio calculated from symmetric crosstabs with row = column. Kappa number determines the agreement of evaluations between two observers who are evaluating a situation or phenomenon at the same time. Kappa number varies between 0 and +1. A value of 0 indicates inconsistency and a value of +1 indicates a positive full fit.

If Pr(a) is the sum ratio of the observed matches for two raters, and if Pr(e) is the probability of this match occurring by chance, the formula to be used to find Cohen's kappa coefficient is:

Kappa coefficient (κ);

– No compatibility if 0 ≤ κ ≤ 0.20

– If 0.20 ≤ κ ≤ 0.40, there is poor compatibility.

– If 0.40 ≤ κ ≤ 0.60, there is moderate (sufficient) compatibility.

– If 0.60 ≤ κ ≤ 0.80 there is a very good (high) level of compatibility.

– If 0.80 ≤ κ ≤ 1.00, there is excellent compatibility.

#### **Silhouette**

It is the average of the distances of each element in the cluster from other clusters. It is expected to be maximized. Thus, each element will not be similar to clusters other than the cluster it belongs to.

In the method used to calculate the silhouette index value; first for any value of k

After dividing into clusters, the Silhouette value is calculated for each data in the data set. The average of these values

Returns the Silhouette index value for the da k value. The Silhouette value for any i data is the following equation.

calculated with the help of

Here a(i), i. the average of the distances of the data from all the data in its own set, b(i) is i. give your data is the minimum value of the mean of their distance from all data in other clusters.

#### **Dunn index**

It is a measure of the distance between the clusters formed. It is expected to be maximized. Thus, the differences between the clusters will be the most. It is a value between 0 and positive infinity.

#### **Purity**

To compute purity , each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by N.

where

N = number of objects(data points),

k = number of clusters,

ci is a cluster in ci

tj is the classification which has the max count for cluster ci

#### **Normalized mutual information (NMI)**

It is a method usually used in feature selection. It is used to compare two different sets of classes.

where,

Y = class labels

C = cluster labels

H(.) = Entropy

I(Y;C) = Mutual Information b/w Y and C

#### **Variation of information**

The 'variation of information' distance presented by Meila (2007) between two clusterings/partitions of the same objects. [21]

It measures informational significance between two sets of classes. The formula is as follows.

Suppose we have two partitions X and Y of a set A into disjoint subsets, namely

X = {, ,…, } and Y = {, ,…,

Let:



 and 



Then the variation of information between the two partitions is:



**Cluster Matching**

Linear sum assignment problem using the Hungarian method [22]

Since many data sets were used in the study here, it also brought some problems.

Hence clustering algorithms are unsupervised, its assignment outputs can produce different categorical results each time. In order for these results to be accurate at the reporting stage, the previously given cluster classes in the data sets and the class groups obtained after the estimation should represent the same regions. However, since the algorithm outputs are different, it will be necessary to arrange the results before reporting. At this point, an optimization problem arises. In order to make the assignments with the highest performance, the Hungarian method will be used in accordance with the one-to-one relationship.

One of the commonly used methods in solving assignment problems is the Hungarian Algorithm. In this algorithm, the number of sources must be equal to the number of destinations. So the problem must be balanced. Hungarian algorithm steps are like below:

Step 1: Find the minimum of each row in the original dimensional cost matrix and subtract this value from the values in its row to create a new matrix. Find the minimum of each column in the new matrix and subtract this value from the values in its own column to form the new reduced cost matrix.

Step 2: Create a minimum number of horizontal and vertical lines to pass through all zero value cells in the reduced cost matrix. If a number of lines are drawn to cover all zero values, the optimum solution exists between the covered zeros. Otherwise, go to Step 3.

Step 3: Find the smallest non-zero element. Subtract this value from the uncovered elements in the reduced cost matrix and add it to the covered elements with two lines.

Then go to Step 2.

## **Propose Method: Hierarchical Clustering with Chunks Via SMOTE (HCCS)**

Basically, our approach is containing a continuous information transfer for next clustering and data shrinks itself that way. After a chunk clustering every cluster goes for the SMOTE process and under-sampled data is replaced with input data. And the main data sets size decreases. The transferred data contains all cluster information with under-sampled distribution. That loop continues until all data size falls below the chunks size.

Behind this approach is a structure in which hierarchical clustering is at the center. The most challenging aspect of working with chunks in large data sets is ensuring the continuity of information transfer. An innovative approach to the methods used to overcome this problem has been tried to find a solution. In theory, information is the class information they receive after each chunk clustering. When each chunk is clustered within itself, a distribution density information emerges. It is not possible to carry the entire data set so that this information can be transferred to the next processes and the process is finite. Instead, the obtained information should be made smaller and easier to calculate and understand. For this reason, it was deemed appropriate to use an under-sampling method.

Another difficulty in working with chunk sets will be that the selected sample groups cannot fully represent the main data set. In this case, each chunk only has to provide its own representation. The cluster distribution will be different in each small data set to be studied compared to the main data set. And the results and distributions will differ from each other in each chunk generation process. In order to prevent this situation or to reduce its effect, it seems appropriate to balance the output distributions. Hence this approach will also create a misdirection situation in the datasets that least fit the distribution of the main dataset. However, it is expected to show the expected effect in the majority of the data set. At this point, the SMOTE method will come into play to help us. will be applied after each clustering, providing a balanced, scaled-down dataset with each chunk best represented as output.

The proposed structure is generally as follows.

* The master data is divided into parts.
* Each part in order
* Clustering is done
* A sample group is formed from each cluster obtained
* The next part of the sample group is added to the data.
* Cluster centers are determined after all parts are completed.
* The master data is assigned according to the cluster centers obtained.

Thus, the information obtained from each part is preserved and transferred to the next step, and it is ensured that the important information is preserved and comes to the last stage. It is envisaged to use the SMOTE method in sample selection.

# Statistical Test

In this study, statistical analysis will not be made on the data set, and the correlation between the performance of the proposed method and the factors will be examined. In the study, a total of one hundred and eight data sets were tested with both the proposed method and simple hierarchical clustering. At the end of each test, the performance metrics of the study were calculated for both methods. The bilateral relations between the outputs obtained and the calculated metrics and the properties of the data set will be tested with "pearsons correlation". The relationality that seems significant here will give information about the cases where this study is successful. And it will create ideas to be evaluated in future studies.

Pearson's correlation is a statistical method used to determine whether there is a linear relationship between two numerical measurements, and if so, what is the direction and severity of this relationship. Pearson correlation coefficient is preferred if the data has a normal distribution, and Spearman Rank correlation coefficient is preferred if the data is not normally distributed. For a correlation coefficient to be interpreted, the p value must be less than 0.05.

If the correlation coefficient is negative, there is an inverse relationship between the two variables, that is, it is said that "as one variable increases, the other decreases". If the correlation coefficient is positive, it is interpreted that "as one of the variables increases, the other increases".

Interpretation of the correlation coefficient (r);

* If r<0.2, very weak correlation or no correlation
* Between 0.2-0.4 weak correlation
* Moderate correlation between 0.4-0.6
* High correlation between 0.6-0.8
* If it is 0.8>, it is interpreted that there is a very high correlation.

Formula is like below;

**Chapter 5**

# Experiments

## **Experimental Setup**

Our work will generally consist of seven stages. The first is the supply of data. Secondly, it will be in the form of preparing the requirement functions that will be needed, then preparing the data for testing, performing the clustering operations with the simple algorithm, clustering with the proposed method, making the result performance measurements, and analyzing the outputs.

The selected data sets were especially preferred to contain different patterns. Here, in addition to the data sets in which hierarchical clustering works well, data structures that are difficult to detect that are not suitable for the way HC works are also included in the tests. Thus, the working properties of the proposed structure will be observed in cases where the simple structure is disadvantageous.

First of all, the process will need to be prepared automatically as it will be working with a large number of data sets. For this purpose, a catalog of the data set to be studied will be prepared. The number of rows and columns, the column name containing the class information, and the number of classes will be kept for later use in the catalog content.

In the second stage, two function definitions will be made to be used later. One of these functions will be used to match the calculated and actual cluster numbers. This function will work for applying the “Hungarian Method” and then assigning the relevant cluster information. The other function is the function to be assigned so that the cluster information obtained after smote can be applied to the whole set. This function works to assign the nearest cluster centroid according to mean squared error calculations for newly assigned observations.

The next stage includes reading the data sets sequentially and checking and editing the data for use in the next stage.

In the other step, there is the division of the data set into clusters equal to the actual number of clusters with the help of an algorithm using the simple hierarchical clustering method. The clustering method used at this stage was a centroid-based approach known as "Unweighted Pair Group Method using Centroids (UPGMC)"[23]. The main reason for preferring this approach is that in the last step of the proposed method, a centroid-based assignment is made in the same way. By using the output matching function, which we defined earlier, the output sets will be matched with the actual class values.

At another stage, the clustering process will be started with the same data sets and the proposed method. the outputs will be matched with the original sets in the same way.

In the conclusion phase, the detected simple clustering output and the outputs of the proposed method will be compared with the actual cluster values. To see the effectiveness of batch size, each dataset will calculate with different batch sizes which are 50,100 and 200. Here, in addition to the silhouette, dunn index, purity, normalized mutual information, variation of information distance metrics used in the performance measurements of the clustering algorithms that are normally applied, the accuracy and kappa measurements used in the measurements of the supervised models will also be calculated in comparison with the actual cluster values. In addition to the stability information to be obtained from clustering performance metrics, these accuracy and kappa values ​​will be main metrics and more explanatory at the decision point. The correlation matrix will then assist in the comparative analysis to evaluate the outputs.

## **Results**

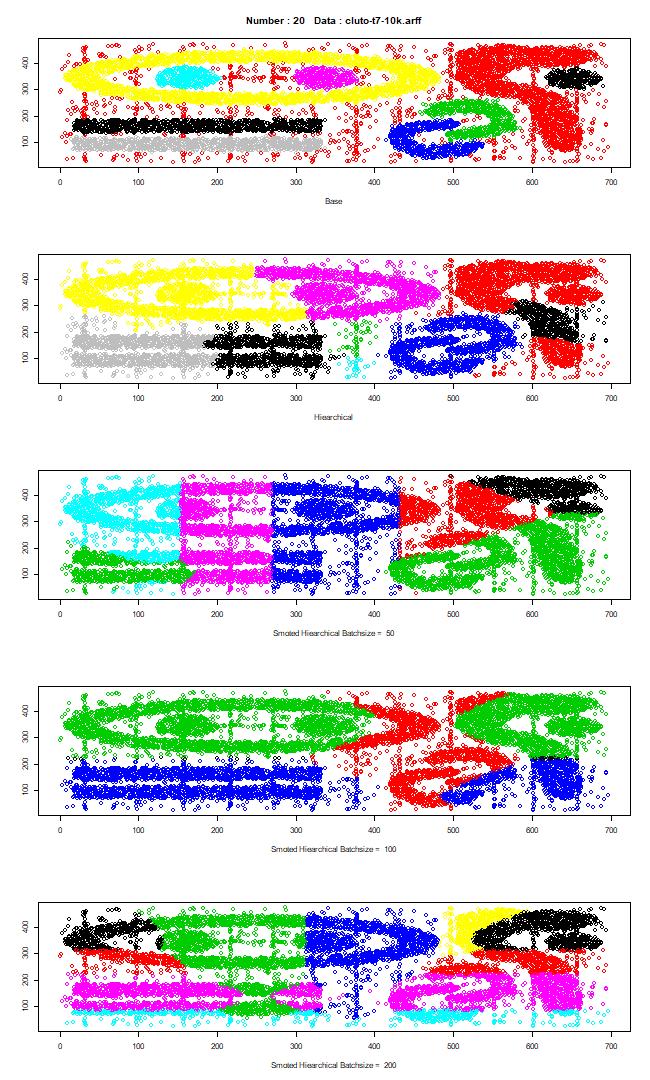
In this chapter, the outputs will be analyzed in two stages. First, the results of the ten sample data sets that we have created to represent the entire data set will be evaluated within themselves. The datasets were selected from among the most widely used synthetic datasets in the literature. Apart from the data distributions where hierarchical clustering is advantageous in terms of analysis, data structures that are difficult to analyze are also included in the study to see the difference of approaches under hard conditions. These data were included in the ten sets selected from the study set. Points that may have affected the performance and success of each will be commented on. The results for both simple hierarchical clustering and the proposed method will be compared. In the next stage, the results of the ninety-eight data sets that will be tested will be examined. The factors that can affect each other in the outputs will be tried to be determined and interpreted. And a general post-analysis interpretation will be made on the results.

Result sets will be shown under these performance parameters explained below.

|  |  |
| --- | --- |
| **accuracy hc** | accuracy of hieararchical clustering |
| **accuracy hc smote** | accuracy of proposed method (smoted hc) |
| **accuracy difference** | accuracy difference of base and proposed method |
| **kappa hc** | kappa of hieararchical clustering |
| **kappa hc smote** | kappa of proposed method (smoted hc) |
| **sill base** | sillhouette value of real clusters |
| **sill hc** | sillhouette value of hieararchical clustering |
| **sill hc smote** | sillhouette value of proposed method (smoted hc) |
| **dunn base** | dunn index value of real clusters |
| **dunn hc** | dunn index value of hieararchical clustering |
| **dunn hc smote** | dunn index value of proposed method (smoted hc) |
| **purity hc** | purity value of hieararchical clustering |
| **purity hc smote** | purity value of proposed method (smoted hc) |
| **mut inf hc** | mutual information value of hieararchical clustering with real clusters |
| **mut inf hc smote** | mutual information value of proposed method (smoted hc) with real clusters |
| **mut inf hc smote and hc** | mutual information value of proposed method (smoted hc) with hierarchical clustering result |
| **var inf hc** | variance of information value of hieararchical clustering |
| **var inf hc smote** | variance of information value of proposed method (smoted hc) |

*Table 5.1 : explanation of performance result list metrics detail*

Data Set number 1: cluto-t7-10k



*Figure 5.1: cluto-t7-10k dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,50 | 0,50 | 0,50 |
| **accuracy hc smote** | 0,30 | 0,37 | 0,34 |
| **accuracy difference** | 0,19 | 0,13 | 0,16 |
| **kappa hc** | 0,42 | 0,42 | 0,42 |
| **kappa hc smote** | 0,22 | 0,23 | 0,24 |
| **sill base** | -0,07 | -0,07 | -0,07 |
| **sill hc** | 0,32 | 0,32 | 0,32 |
| **sill hc smote** | 0,12 | 0,06 | -0,03 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,01 | 0,01 | 0,01 |
| **dunn hc smote** | 0,00 | 0,00 | 0,00 |
| **purity hc** | 0,50 | 0,50 | 0,50 |
| **purity hc smote** | 0,27 | 0,37 | 0,34 |
| **mut inf hc** | 0,65 | 0,65 | 0,65 |
| **mut inf hc smote** | 0,32 | 0,27 | 0,36 |
| **mut inf hc smote and hc** | 0,50 | 0,36 | 0,51 |
| **var inf hc** | 2,06 | 2,06 | 2,06 |
| **var inf hc smote** | 3,55 | 2,86 | 3,46 |

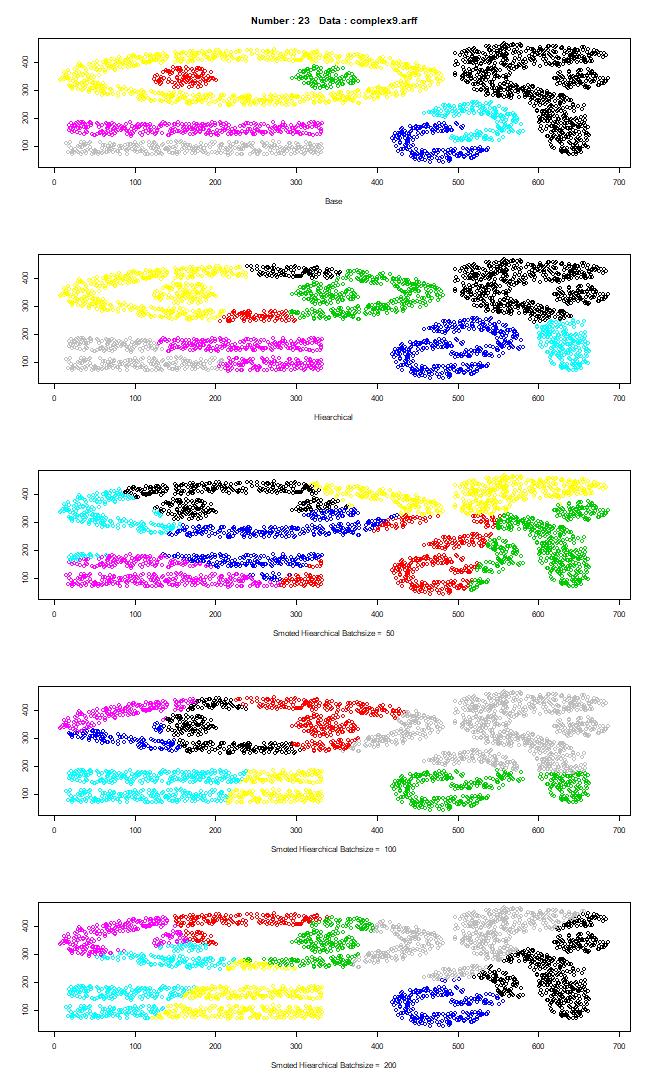
*Table 5.2 : application test metric results of cluto-t7-10k dataset with different batch sizes*

This dataset is a noisy dataset. Besides, decentralized patterns are difficult to detect by hierarchical clustering. Although some of the data can be separated regionally around cluster centers, the transitive distribution caused the error to grow.

Considering the performance values, the simple method achieved an accuracy rate of 50%, while the proposed method detected clusters correctly at a rate about 34%. There are similar differences in other metrics as well. Since there is no distance between the clusters, the dunn index was calculated close to zero. Both methods seem to fail. Here, the factor can be considered as a high cluster number and not localized distribution. Attention should be paid to this situation.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 100 compared to the others. The difference of ten points is not too small to be considered. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 2: complex9



*Figure 5.2:* complex9 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,55 | 0,55 | 0,55 |
| **accuracy hc smote** | 0,45 | 0,52 | 0,45 |
| **accuracy difference** | 0,09 | 0,03 | 0,09 |
| **kappa hc** | 0,47 | 0,47 | 0,47 |
| **kappa hc smote** | 0,38 | 0,44 | 0,38 |
| **sill base** | -0,02 | -0,02 | -0,02 |
| **sill hc** | 0,37 | 0,37 | 0,37 |
| **sill hc smote** | 0,31 | 0,34 | 0,31 |
| **dunn base** | 0,03 | 0,03 | 0,03 |
| **dunn hc** | 0,02 | 0,02 | 0,02 |
| **dunn hc smote** | 0,00 | 0,00 | 0,00 |
| **purity hc** | 0,55 | 0,55 | 0,55 |
| **purity hc smote** | 0,42 | 0,50 | 0,45 |
| **mut inf hc** | 0,68 | 0,68 | 0,68 |
| **mut inf hc smote** | 0,50 | 0,58 | 0,53 |
| **mut inf hc smote and hc** | 0,57 | 0,68 | 0,65 |
| **var inf hc** | 1,65 | 1,65 | 1,65 |
| **var inf hc smote** | 2,72 | 2,28 | 2,57 |

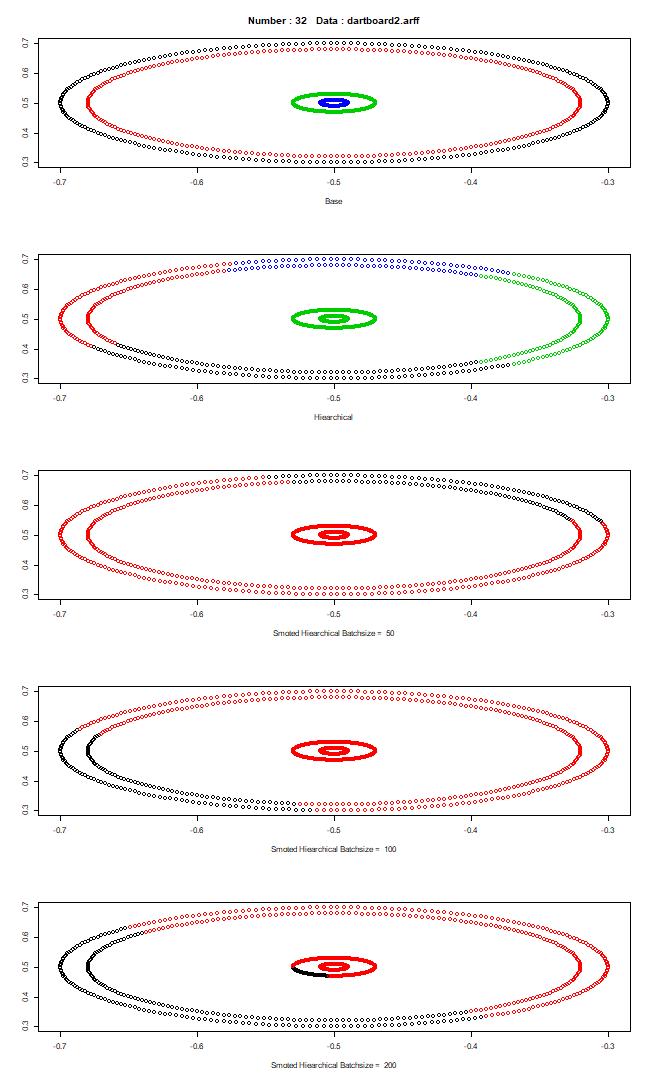
*Table 5.3 : application test metric results of complex9 dataset with different batch sizes*

This dataset is not a noisy dataset like the previous one. is the noise-free version of the previous set. However, decentralized patterns are difficult to detect by hierarchical clustering. Although some of the data can be separated regionally around cluster centers, the transitive distribution caused the error to grow.

Considering the performance values, the simple method achieved an accuracy rate of 55%, while the proposed method detected about 50% correct cluster. As can be seen in these two examples, the success of the simple method increases as noise decrease. It appears to be similar in other metrics as well. And proposal method also acts accordingly.

Batch sizes applied in this dataset seem to be not effective. However, a higher performance was achieved in size 100 compared to the others as the same with dataset number 1.

Data Set number 3: dartboard2



*Figure 5.3:* dartboard2 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,39 | 0,39 | 0,39 |
| **accuracy hc smote** | 0,31 | 0,32 | 0,36 |
| **accuracy difference** | 0,07 | 0,06 | 0,02 |
| **kappa hc** | 0,18 | 0,18 | 0,18 |
| **kappa hc smote** | 0,09 | 0,10 | 0,15 |
| **sill base** | -0,01 | -0,01 | -0,01 |
| **sill hc** | 0,50 | 0,50 | 0,50 |
| **sill hc smote** | 0,40 | 0,42 | 0,34 |
| **dunn base** | 0,05 | 0,05 | 0,05 |
| **dunn hc** | 0,01 | 0,01 | 0,01 |
| **dunn hc smote** | 0,01 | 0,01 | 0,00 |
| **purity hc** | 0,39 | 0,39 | 0,39 |
| **purity hc smote** | 0,31 | 0,32 | 0,36 |
| **mut inf hc** | 0,25 | 0,25 | 0,25 |
| **mut inf hc smote** | 0,07 | 0,08 | 0,09 |
| **mut inf hc smote and hc** | 0,14 | 0,22 | 0,29 |
| **var inf hc** | 2,48 | 2,48 | 2,48 |
| **var inf hc smote** | 2,27 | 2,27 | 2,52 |

*Table 5.4 : application test metric results of dartboard2 dataset with different batch sizes*

The hierarchical clustering method is disadvantageous in this data set. Cluster centers intersect at a single point as can be seen. In this distribution, where normally a density-based algorithm will work best, the simple method divides the outer rings into four parts. The proposed method, on the other hand, distributed the inner clusters based on the clusters of the two outer rings as the density.

Considering the performance values, the simple method achieved an accuracy rate of 39%, while the proposed method detected about 32% of clusters correctly. There are similar differences in other metrics as well.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 200 compared to the others. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 4: disk-5000n

ok içeren bir resim

Açıklama otomatik olarak oluşturuldu

*Figure 5.4:* disk-5000n *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,61 | 0,61 | 0,61 |
| **accuracy hc smote** | 0,50 | 0,59 | 0,57 |
| **accuracy difference** | 0,11 | 0,03 | 0,04 |
| **kappa hc** | -0,24 | -0,24 | -0,24 |
| **kappa hc smote** | -0,18 | 0,14 | 0,15 |
| **sill base** | -0,06 | -0,06 | -0,06 |
| **sill hc** | 0,24 | 0,24 | 0,24 |
| **sill hc smote** | 0,23 | 0,26 | 0,21 |
| **dunn base** | 0,02 | 0,02 | 0,02 |
| **dunn hc** | 0,01 | 0,01 | 0,01 |
| **dunn hc smote** | 0,00 | 0,00 | 0,00 |
| **purity hc** | 0,61 | 0,61 | 0,61 |
| **purity hc smote** | 0,50 | 0,59 | 0,57 |
| **mut inf hc** | 0,09 | 0,09 | 0,09 |
| **mut inf hc smote** | 0,03 | 0,02 | 0,03 |
| **mut inf hc smote and hc** | 0,00 | 0,04 | 0,21 |
| **var inf hc** | 1,28 | 1,28 | 1,28 |
| **var inf hc smote** | 1,61 | 1,68 | 1,67 |

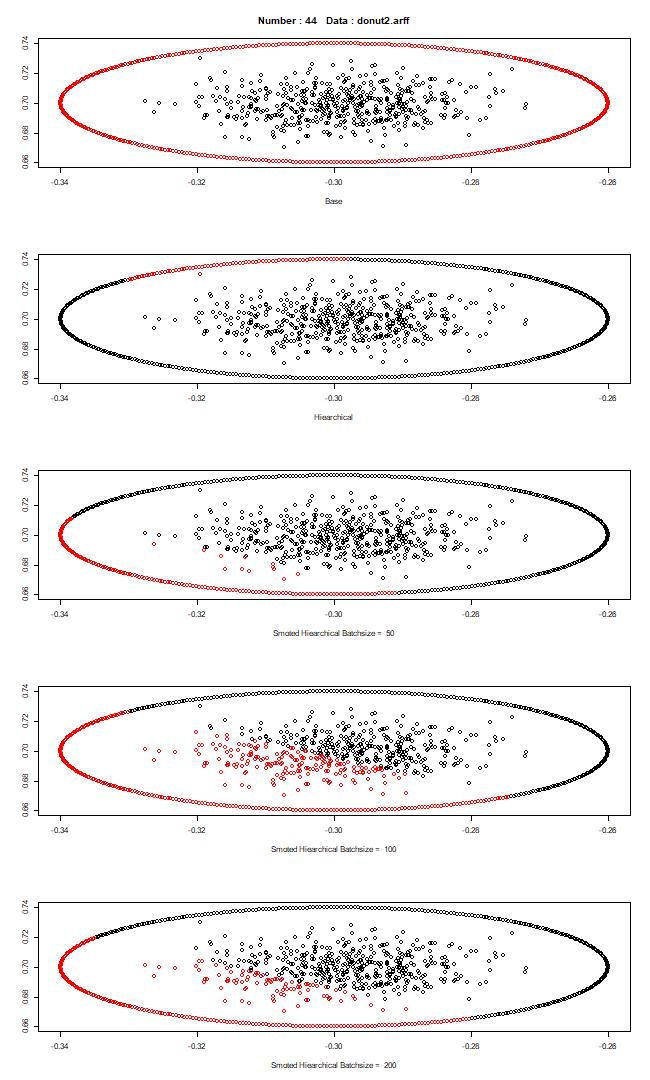
*Table 5.5 : application test metric results of disk-5000n dataset with different batch sizes*

This data set again has a concentric distribution. Neither algorithm could detect the middle cluster.

Considering the performance values, the simple method achieved an accuracy rate of 61%, while the proposed method detected about 56% of clusters correctly. It appears to be similar in other metrics as well. But at the batch size 100 application, success is %59 and is about to catch basic algorithms success.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 100 compared to the others. The difference of nine points is not too small to be considered. Since the overall application performance is not high, it should be evaluated together with other results.

Data Set number 5: donut2



*Figure 5.5:* donut2 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,57 | 0,57 | 0,57 |
| **accuracy hc smote** | 0,66 | 0,55 | 0,62 |
| **accuracy difference** | -0,08 | 0,02 | -0,05 |
| **kappa hc** | 0,14 | 0,14 | 0,14 |
| **kappa hc smote** | 0,31 | 0,10 | 0,23 |
| **sill base** | 0,18 | 0,18 | 0,18 |
| **sill hc** | 0,26 | 0,26 | 0,26 |
| **sill hc smote** | 0,32 | 0,29 | 0,31 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,01 | 0,01 | 0,01 |
| **dunn hc smote** | 0,01 | 0,01 | 0,00 |
| **purity hc** | 0,57 | 0,57 | 0,57 |
| **purity hc smote** | 0,66 | 0,55 | 0,62 |
| **mut inf hc** | 0,07 | 0,07 | 0,07 |
| **mut inf hc smote** | 0,14 | 0,01 | 0,05 |
| **mut inf hc smote and hc** | 0,03 | 0,06 | 0,04 |
| **var inf hc** | 1,24 | 1,24 | 1,24 |
| **var inf hc smote** | 1,40 | 1,97 | 1,78 |

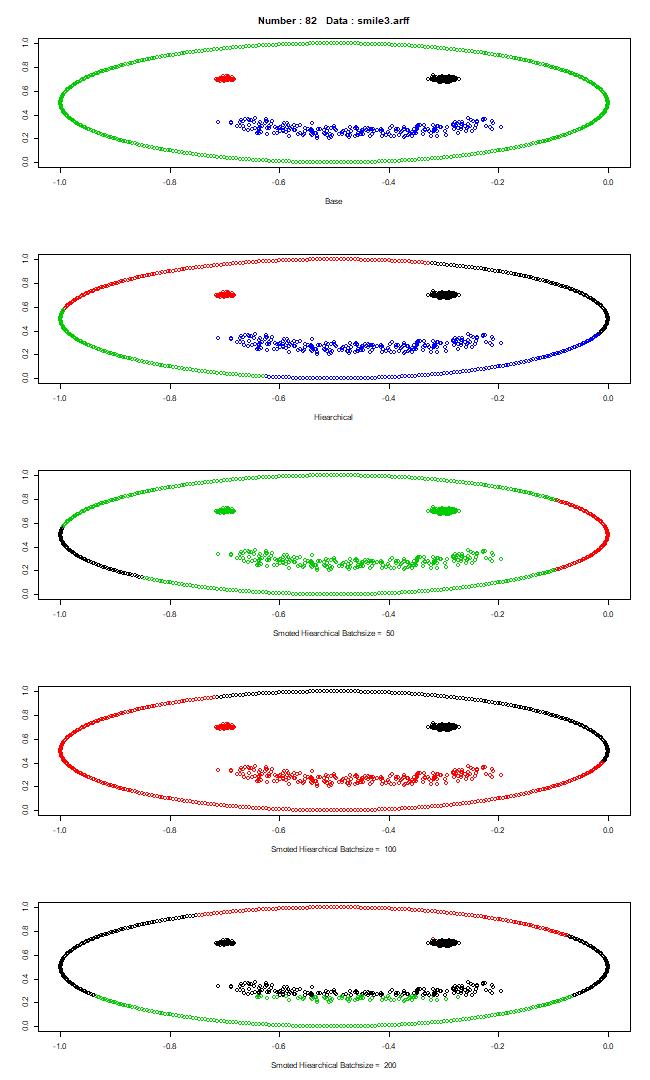
*Table 5.6 : application test metric results of donut2 dataset with different batch sizes*

This data again includes two clusters with close center values. This is a disadvantageous situation in terms of hierarchical clustering. While the simple method separated part of the outer ring from the inner region, the proposed method found the centroid value closer to the center.

Considering the performance values, the simple method has an accuracy rate of 57%, while the proposed method detects clusters correctly at a rate about 60%. It appears to be similar in other metrics as well. Although the performances seem close to each other, the similarity between the outputs is around 5%.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 50 compared to the others. The difference of nine points is not too small to be considered. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 6: smile3



*Figure 5.6:* smile3 *dataset experimental results both HC and Proposal method with different batch sizes*

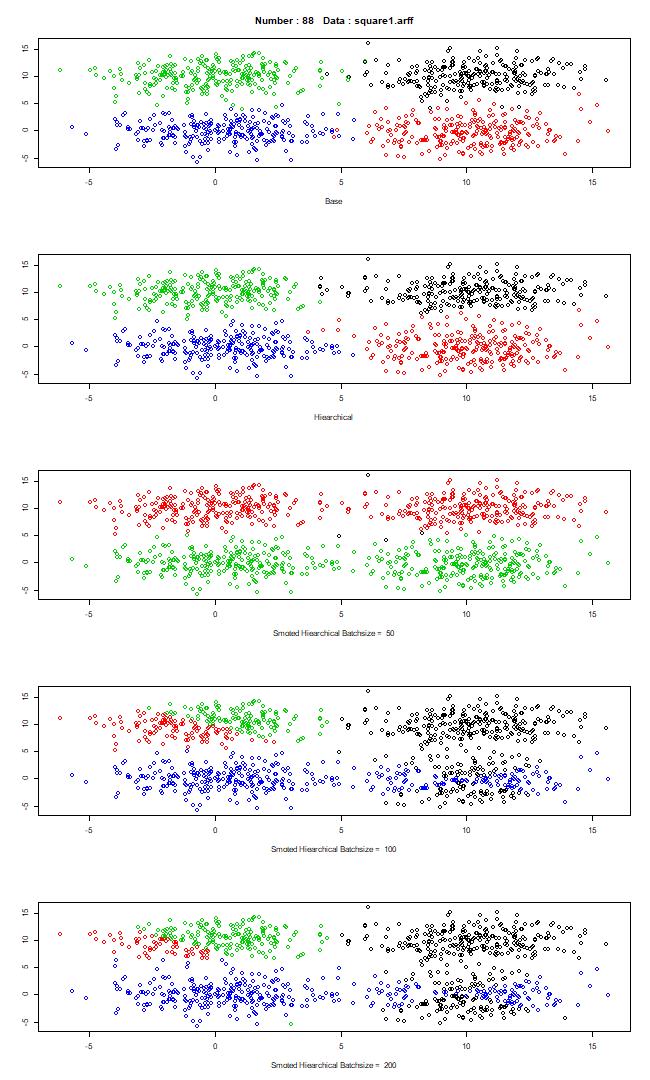
|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,62 | 0,62 | 0,62 |
| **accuracy hc smote** | 0,35 | 0,53 | 0,39 |
| **accuracy difference** | 0,27 | 0,09 | 0,23 |
| **kappa hc** | 0,51 | 0,51 | 0,51 |
| **kappa hc smote** | 0,11 | 0,22 | 0,19 |
| **sill base** | 0,14 | 0,14 | 0,14 |
| **sill hc** | 0,49 | 0,49 | 0,49 |
| **sill hc smote** | 0,06 | 0,40 | 0,10 |
| **dunn base** | 0,13 | 0,13 | 0,13 |
| **dunn hc** | 0,01 | 0,01 | 0,01 |
| **dunn hc smote** | 0,01 | 0,01 | 0,00 |
| **purity hc** | 0,59 | 0,59 | 0,59 |
| **purity hc smote** | 0,33 | 0,53 | 0,28 |
| **mut inf hc** | 0,47 | 0,47 | 0,47 |
| **mut inf hc smote** | 0,12 | 0,29 | 0,17 |
| **mut inf hc smote and hc** | 0,18 | 0,39 | 0,21 |
| **var inf hc** | 1,80 | 1,80 | 1,80 |
| **var inf hc smote** | 2,12 | 1,66 | 2,40 |

*Table 5.7 : application test metric results of smile3 dataset with different batch sizes*

This data set is fixed to a distribution where density-based algorithms will be more successful. Although HC methods are not very successful, the simple method seems more successful.

Considering the performance values, the simple method achieved an accuracy rate of 62%, while the proposed method detected 35,39,53% at each try of clusters correctly. It appears to be similar in other metrics as well.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 100 compared to the others. The difference of eighteen points is not too small to be considered. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 7: square1

*Figure 5.7:* square1 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,97 | 0,97 | 0,97 |
| **accuracy hc smote** | 0,50 | 0,63 | 0,67 |
| **accuracy difference** | 0,48 | 0,35 | 0,31 |
| **kappa hc** | 0,97 | 0,97 | 0,97 |
| **kappa hc smote** | 0,33 | 0,50 | 0,56 |
| **sill base** | 0,59 | 0,59 | 0,59 |
| **sill hc** | 0,59 | 0,59 | 0,59 |
| **sill hc smote** | 0,18 | 0,27 | 0,25 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,06 | 0,06 | 0,06 |
| **dunn hc smote** | 0,03 | 0,01 | 0,01 |
| **purity hc** | 0,97 | 0,97 | 0,97 |
| **purity hc smote** | 0,50 | 0,63 | 0,67 |
| **mut inf hc** | 0,91 | 0,91 | 0,91 |
| **mut inf hc smote** | 0,46 | 0,61 | 0,59 |
| **mut inf hc smote and hc** | 0,48 | 0,63 | 0,60 |
| **var inf hc** | 0,37 | 0,37 | 0,37 |
| **var inf hc smote** | 1,20 | 1,34 | 1,38 |

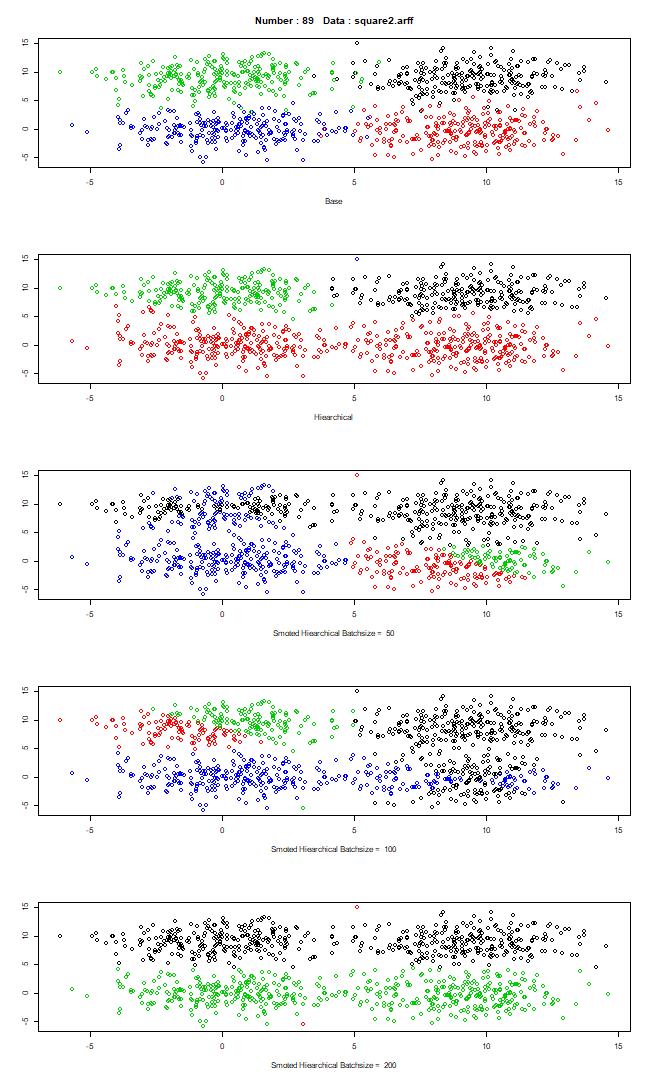
*Table 5.8 : application test metric results of square1 dataset with different batch sizes*

In the first square data set, the regions close to each other in which the clusters are located were determined at a high rate by the hierarchical method. The proposed method was not as successful as the simple method. As can be seen, the source of the problem is that the red cluster center is detected in the wrong place. This may be due to the scarcity of the studied data set or the weighted sample groups included in the mini-batch data. Apart from this situation, the centroid points of the other two clusters seem to be correct.

Considering the performance values, the simple method achieved an accuracy rate of 97%, while the proposed method detected about 63% of clusters correctly. There are similar differences in other metrics as well.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 200 compared to the others. The difference of ten points is not too small to be considered. Since the overall application performance is low, it should be evaluated together with other results. But bigger batch sizes had more success in this dataset.

Data Set number 8: square2



*Figure 5.8:* square2 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,71 | 0,71 | 0,71 |
| **accuracy hc smote** | 0,64 | 0,63 | 0,49 |
| **accuracy difference** | 0,08 | 0,09 | 0,22 |
| **kappa hc** | 0,62 | 0,62 | 0,62 |
| **kappa hc smote** | 0,52 | 0,50 | 0,32 |
| **sill base** | 0,54 | 0,54 | 0,54 |
| **sill hc** | 0,39 | 0,39 | 0,39 |
| **sill hc smote** | 0,23 | 0,27 | 0,39 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,03 | 0,03 | 0,03 |
| **dunn hc smote** | 0,01 | 0,01 | 0,03 |
| **purity hc** | 0,71 | 0,71 | 0,71 |
| **purity hc smote** | 0,64 | 0,63 | 0,49 |
| **mut inf hc** | 0,63 | 0,63 | 0,63 |
| **mut inf hc smote** | 0,57 | 0,59 | 0,44 |
| **mut inf hc smote and hc** | 0,38 | 0,56 | 0,61 |
| **var inf hc** | 0,97 | 0,97 | 0,97 |
| **var inf hc smote** | 1,50 | 1,43 | 1,27 |

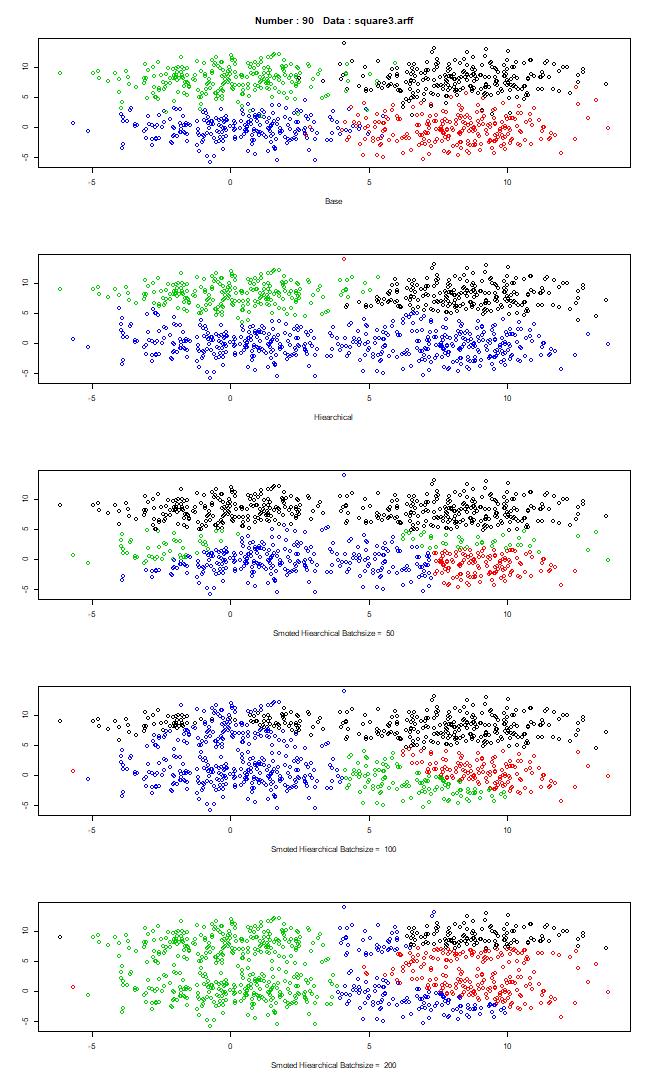
*Table 5.9 : application test metric results of square2 dataset with different batch sizes*

In this set, regions close to each other in which clusters are located have been detected at a high rate by the hierarchical method. The proposed method has not been as successful as the simple method. However, the difference was not as much as the previous data set. Although this data set has similar features with the previous data set, the transitivity of the clusters is more intense.

Considering the performance values, the simple method achieved an accuracy rate of 71%, while the proposed method detected about 63% of clusters correctly. There are similar differences in other metrics as well.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 50 compared to the others. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 9: square3



*Figure 5.9:* square3 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,70 | 0,70 | 0,70 |
| **accuracy hc smote** | 0,60 | 0,61 | 0,52 |
| **accuracy difference** | 0,11 | 0,10 | 0,19 |
| **kappa hc** | 0,60 | 0,60 | 0,60 |
| **kappa hc smote** | 0,46 | 0,48 | 0,35 |
| **sill base** | 0,48 | 0,48 | 0,48 |
| **sill hc** | 0,37 | 0,37 | 0,37 |
| **sill hc smote** | 0,28 | 0,24 | 0,23 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,02 | 0,02 | 0,02 |
| **dunn hc smote** | 0,01 | 0,01 | 0,01 |
| **purity hc** | 0,70 | 0,70 | 0,70 |
| **purity hc smote** | 0,60 | 0,61 | 0,52 |
| **mut inf hc** | 0,58 | 0,58 | 0,58 |
| **mut inf hc smote** | 0,50 | 0,57 | 0,51 |
| **mut inf hc smote and hc** | 0,49 | 0,41 | 0,34 |
| **var inf hc** | 1,15 | 1,15 | 1,15 |
| **var inf hc smote** | 1,73 | 1,56 | 1,78 |

*Table 5.10 : application test metric results of square3 dataset with different batch sizes*

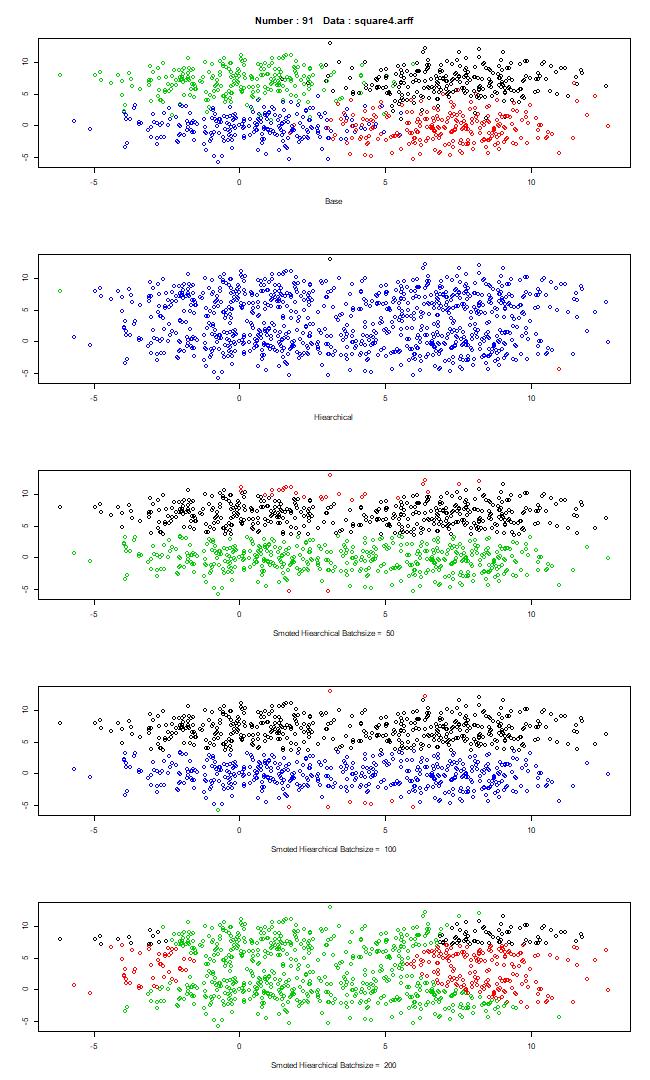
In this set, regions close to each other in which clusters are located have been detected at a high rate by the hierarchical method. The proposed method has not been as successful as the simple method. However, the difference was not as much as the previous data set. Although this data set has similar features with the previous data set, the transitivity of the clusters is more intense. In this case, hierarchical clustering is a more disadvantageous method.[24]

The problems that may occur in the first set are also valid for this set.

Considering the performance values, the simple method achieved an accuracy rate of 70%, while the proposed method detected a cluster correctly at a rate about 60%. There are similar differences in other metrics as well.

Batch sizes applied in this dataset seem to be effective. However, a higher performance was achieved in size 100 compared to the others. The difference of ten points is not too small to be considered. Since the overall application performance is low, it should be evaluated together with other results.

Data Set number 10: square4



*Figure 5.10:* square4 *dataset experimental results both HC and Proposal method with different batch sizes*

|  |  |  |  |
| --- | --- | --- | --- |
|  | batchsize 50 | batchsize 100 | batchsize 200 |
| **accuracy hc** | 0,25 | 0,25 | 0,25 |
| **accuracy hc smote** | 0,49 | 0,49 | 0,40 |
| **accuracy difference** | -0,24 | -0,23 | -0,15 |
| **kappa hc** | 0,00 | 0,00 | 0,00 |
| **kappa hc smote** | 0,32 | 0,31 | 0,20 |
| **sill base** | 0,41 | 0,41 | 0,41 |
| **sill hc** | -0,05 | -0,05 | -0,05 |
| **sill hc smote** | 0,24 | 0,26 | 0,10 |
| **dunn base** | 0,00 | 0,00 | 0,00 |
| **dunn hc** | 0,06 | 0,06 | 0,06 |
| **dunn hc smote** | 0,01 | 0,01 | 0,01 |
| **purity hc** | 0,25 | 0,25 | 0,25 |
| **purity hc smote** | 0,49 | 0,49 | 0,40 |
| **mut inf hc** | 0,00 | 0,00 | 0,00 |
| **mut inf hc smote** | 0,38 | 0,38 | 0,12 |
| **mut inf hc smote and hc** | 0,01 | 0,01 | 0,00 |
| **var inf hc** | 2,02 | 2,02 | 2,02 |
| **var inf hc smote** | 1,63 | 1,55 | 2,70 |

*Table 5.11 : application test metric results of square4 dataset with different batch sizes*

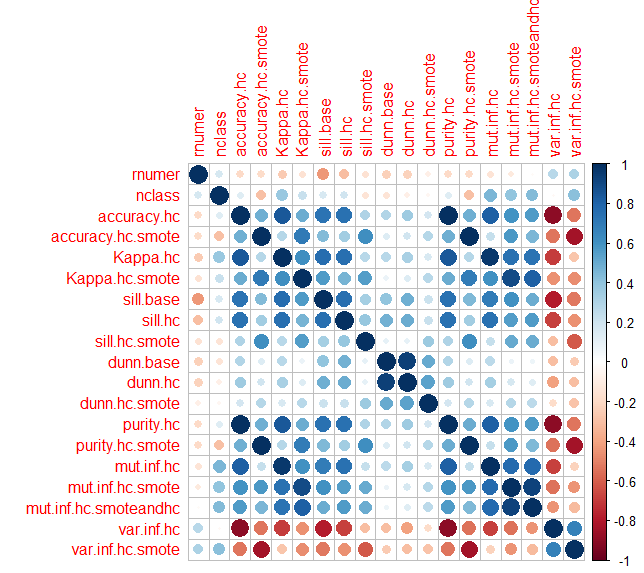
In this data set, the regions close to each other in which the clusters are located could not be determined by the hierarchical method. The proposed method has achieved similar success as before “square” datasets. The reason the simple method stumbles so much is because this set is much more transitive than its predecessors. For this reason, the value determined as the cluster center seems to be determined at the most central point. The problem of the proposed method may be the same as in the previous data.

Considering the performance values, the simple method achieved an accuracy rate of 25%, while the proposed method detected 49% of clusters correctly. Since there is no distance between the clusters, the dunn index was calculated close to zero. There are similar differences in other metrics as well. The proposed method is more successful. Looking at these three data sets, it is seen that there is a relationship between the transition in the data set and hierarchical clustering success, but there is no visible interaction of the proposed method. In these conditions, transition of clusters are effective on success of base hc algorithm. But proposed method is less affected.

Individual case studies were as above. As for the general evaluation and statistics; First of all, since we have the actual values ​​of the clusters here, accuracy will be the most important performance metric for us. In addition, the kappa will be included in the assessment. After that, silhouette and dunn index values ​​from clustering metrics were used as informative in algorithm performances specific to datasets. In the significance measurement, purity, normalized mutual information, variation of information values ​​will be considered.

To see the effectiveness of batch size, average accuracy of each number of batch size groups can be examined. There were three groups tested in this work: 50,100 and 200. The average accuracy of 50 batches is 58 percent; 100 is 59 and 200 is 59. As seen, there are no meaningful difference between results. So can say there are no relation between batch size between success rates.

The average of the accuracy values ​​calculated as a result of all tests for the simple algorithm was calculated as 74%. Besides, the average value of the proposed method is 59%. Since the deviation values ​​of each calculation made will not be meaningful (as the data sets are different from each other, it is not correct to evaluate these values ​​in their own way. It does not provide information in terms of comparing the two methods.) It is necessary to examine the difference in accuracy values ​​obtained in each dataset. According to this calculation, the deviation of the differences seems to be 19%. Kappa value was calculated as 57%. The average NMI value of the proposed method with a simple structure is 38%.



*Figure 5.11: Relational Correlation Matrix of Test Results*

According to the figure, if we evaluate the relationships between the metrics calculated here. There is a weak inverse correlation between the number of classes and the accuracy of the proposed method. There is a positive correlation between the calculated silhouette values of the base cluster values and the accuracy values of other clustering methods. That is, more central distribution of clusters in data sets increases the detection success of clustering methods. There is also a positive relationship between the accuracy values of the two methods.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset number** | **Distance Matrix Size - HC (MB)** | **Distance Matrix Size - Smote (MB)** | **Worktime - HC (sec)** | **Worktime - smote (sec)** |
| 1 | 3,81 | 0,17 | 1,70 | 1,34 |
| 2 | 1,27 | 0,17 | 0,35 | 1,64 |
| 3 | 2,75 | 0,17 | 1,06 | 1,44 |
| 4 | 1,32 | 0,17 | 0,44 | 1,54 |
| 5 | 3,12 | 0,17 | 1,25 | 1,43 |
| 6 | 0,90 | 0,17 | 0,22 | 1,66 |
| 7 | 0,95 | 0,17 | 0,24 | 1,65 |
| 8 | 29,81 | 0,17 | 1,65 | 2,50 |
| 9 | 0,34 | 0,17 | 0,08 | 1,79 |
| 10 | 1,37 | 0,17 | 0,42 | 1,59 |
| 11 | 3,81 | 0,17 | 2,00 | 1,41 |
| 12 | 3,81 | 0,17 | 1,92 | 1,28 |
| 13 | 3,81 | 0,17 | 1,93 | 1,30 |
| 14 | 3,81 | 0,17 | 1,90 | 1,33 |
| 15 | 3,81 | 0,17 | 1,94 | 1,33 |
| 16 | 3,81 | 0,17 | 1,99 | 1,28 |
| 17 | 3,81 | 0,17 | 2,00 | 1,35 |
| 18 | 3,81 | 0,17 | 2,00 | 1,31 |
| 19 | 3,81 | 0,17 | 2,00 | 1,30 |
| 20 | 3,81 | 0,17 | 2,27 | 1,21 |
| 21 | 0,24 | 0,17 | 0,06 | 1,81 |

*Table 5.12 : application test memory usage and work time details of different test sets*

As shown in Table 5.12, there are memory and running time measurements of 21 different data sets using the hierarchical method and the recommended method. In the memory part, regardless of the size of the data set, since the size of the batch to be used in the proposed method can be managed, it is possible to work in the desired range. However, the traditional method must use a distance matrix that grows exponentially with the size of the data. Since the measurement made with smote in the table works with a fixed batch size, it seems fixed.

When the running times are compared, the results are closer to each other. Because the number of cycles is low in small data sets, while higher differences emerge as the data set gets larger. In general, there is an average of 15% runtime difference in small datasets, while this value will be higher in large datasets.

**Chapter 6**

# Conclusion and Discussion

The method proposed here is presented as an alternative to the simple hierarchical clustering method. After the examinations, it was seen that similar results could be produced with an average of twelve percent information loss, but some points should be considered. These are briefly; heavily transitive datasets are disadvantageous for hierarchical clustering, but the proposed method can produce more successful results in these cases. And as seen on test results batch size used in proposal method is not effective on clustering performance. Datasets that are small and contain many clusters may not allow the proposed method to converge itself due to the low number of observations. Datasets containing high noise can be a problem for HC. The same is true for the proposed method. Although the performances for both methods seem similar, sometimes the outputs may not be the same. This makes the proposed method more fragile. In cases where the number of observations is low, the number of chunks created can be important to fit the main set distribution.

In the light of all this information, if the data set to be studied with hierarchical clustering is too large to work with the available resources, the proposed method can be used, considering some information loss. However, the above-mentioned items should be considered when using it. If there is suitability for these situations, the results can sometimes be more satisfactory. The tradeoff here should be well appreciated.

In future studies, the disadvantageous situations mentioned here can be better scrutinized. Algorithms with distance matrix dependency can also be evaluated in accordance with this algorithm flow. Proposal method can be tested on real-world high-volume data and tested on different batch sizes, to see very large data size success is similar with these small synthetic data test results.

# REFERENCES

[1] https://www.kdnuggets.com/2019/04/top-data-science-machine-learning-methods-2018-2019.html

[2] Madhulatha, T. Soni. "An overview on clustering methods." arXiv preprint arXiv:1205.1117 (2012).

[3] Love, B.C. Comparing supervised and unsupervised category learning. Psychonomic Bulletin & Review 9, 829–835 (2002).

[4] Learned-Miller, Erik G. "Introduction to supervised learning." I: Department of Computer Science, University of Massachusetts (2014): 3.

[5] F. Lin and W. W. Cohen, "Semi-Supervised Classification of Network Data Using Very Few Labels," 2010 International Conference on Advances in Social Networks Analysis and Mining, 2010, pp. 192-199, doi: 10.1109/ASONAM.2010.19.

[6] Barua, Anitesh, Charles H. Kriebel, and Tridas Mukhopadhyay. "Information technologies and business value: An analytic and empirical investigation." Information systems research 6.1 (1995): 3-23.

[7] Lorr, Maurice. Cluster analysis for social scientists. Jossey-Bass Incorporated Pub, 1983.

[8] Mollineda, Ramon A., and Enrique Vidal. "A relative approach to hierarchical clustering." Pattern Recognition and Applications 56 (2000): 19-28.

[9] Austin J Proteomics Bioinform & Genomics. 2015; 2(1): 1008 En Efficient Hierarchical Clustering Algorithm for Large Dataset

[10] Elkan C. Using the triangle inequality to accelerate k Means. In: Proceedings of the Twentieth International Conference on Machine Learning 2003 (ICML-2003)

[11] Che J1, King FJ, Zhou B, Zhou Y. Chemical and biological properties of frequent screening hits. J Chem Inf Model 2012; 52: 913-926

[12] Gobbi A, Lee ML. DISE: Directed Sphere Exclusion. J Chem Inf Comput Sci 2003; 43: 317-323

[13] Nguyen, Thuy-Diem, Bertil Schmidt, and Chee-Keong Kwoh. "SparseHC: a memory-efficient online hierarchical clustering algorithm." Procedia Computer Science 29 (2014): 8-19.

[14] Yaniv Loewenstein, Elon Portugaly, Menachem Fromer, and Michal Linial. Eﬃcient algorithms for accurate hierarchical clustering of huge datasets. Bioinformatics, 24(13):i41–9, July 2008.

[15] Yijun Sun, Yunpeng Cai, Li Liu, Fahong Yu, Michael L Farrell, William McKendree, and William Farmerie. ESPRIT: estimating species richness using large collections of 16S rRNA pyrosequences. Nucleic Acids Research, 37(10):e76, June 2009.

[16] Gordon Blackshields, Fabian Sievers, Weifeng Shi, Andreas Wilm, and Desmond G Higgins. Sequence embedding for fast construction of guide trees for multiple sequence alignment. AMB, 5(1):21, January 2010.

[17] El-Sonbaty, Yasser, and Mohamed A. Ismail. "On-line hierarchical clustering." Pattern Recognition Letters 19.14 (1998): 1285-1291.

[18] Syed Tanveer Jishan, Raisul Islam Rashu, Naheena Haque & Rashedur M Rahman Improving accuracy of students’ final grade prediction model using optimal equal width binning and synthetic minority over-sampling technique

[19] Dedi Dwi Saputra et al 2020 J. Phys.: Conf. Ser. 1471 012014 Optimization Sentiments of Analysis from Tweets in myXLCare using Naïve Bayes Algorithm and Synthetic Minority Over Sampling Technique Method

[20] https://github.com/deric/clustering-benchmark

[21] Meila, M. (2007) Comparing Clusterings - an Information Based Distance. Journal of Multivariate Analysis, 98, 873 – 895.

[22] Kuhn, Harold W.. “The Hungarian Method for the Assignment Problem.” 50 Years of Integer Programming (2010).

[23] Bouguettaya, Athman, et al. "Efficient agglomerative hierarchical clustering." Expert Systems with Applications 42.5 (2015): 2785-2797.

[24] Preud’homme, G., Duarte, K., Dalleau, K. et al. Head-to-head comparison of clustering methods for heterogeneous data: a simulation-driven benchmark. Sci Rep 11, 4202 (2021). https://doi.org/10.1038/s41598-021-83340-8