Spring 2019

Data Patterns Discovery Using Unsupervised Learning

Rachel A. Lewis

DATA PATTERNS DISCOVERY USING UNSUPERVISED LEARNING

by

RACHEL ANTONETTE LEWIS

(Under the Direction of I. Emil Iacob)

ABSTRACT

Self-care activities classification poses significant challenges in identifying childrens unique functional abilities and needs within the exceptional children healthcare system. The accuracy of diagnosing a child's self-care problem, such as toileting or dressing, is highly influenced by an occupational therapists experience and time constraints. Thus, there is a need for objective means to detect and predict in advance the self-care problems of children with physical and motor disabilities. We use clustering to discover interesting information from self-care problems, perform automatic classification of binary data, and discover outliers. The advantages are twofold: the advancement of knowledge on identifying self-care problems in children and comprehensive experimental results on clustering binary healthcare data. By using various distances and linkage methods, resampling techniques of imbalanced data, and feature selection preprocessing in a clustering framework, we find associations among patients and an Adjusted Rand Index (ARI) of 76.26%

INDEX WORDS: Binary data, distance methods, hierarchical clustering, feature engineering, unsupervised learning

2009 Mathematics Subject Classification: 62H30,68T10

DATA PATTERNS DISCOVERY USING UNSUPERVISED LEARNING

by

RACHEL ANTONETTE LEWIS

B.S., Armstrong State University, 2016

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial

Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

STATESBORO, GEORGIA

DATA PATTERNS DISCOVERY USING UNSUPERVISED LEARNING

by

RACHEL ANTONETTE LEWIS

DEDICATION

I dedicate this thesis and all of my academic achievements to my mother, my grand-mother and all my extended family who has been a great source of inspiration and support.

ACKNOWLEDGMENTS

I wish to acknowledge Dr. Iacob for being a decent friend and mentor and also the awesome faculty and staff at Georgia Southerns Mathematical Sciences and Computer Science Departments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

LIST OF SYMBOLS

$\mathbb{R}$	Real Numbers
$\mathbb{C}$	Real Numbers
$\mathbb{Z}$	Integers
$\mathbb{N}$	Natural Numbers
$\mathbb{N}_0$	Natural Numbers including 0
$L_p(\mathbb{R})$	p-integrable functions over $\mathbb{R}$
$L(X,Y)$	Linear maps from X to Y
$\mathrm{rank}(T)$	Rank of a linear map

CHAPTER 1

INTRODUCTION TO CLUSTERING

1.1 MOTIVATION

Due to recent technological advances, large masses of medical data are being obtained. This data contains valuable information for diagnosing conditions and diseases and data mining techniques can be useful in extracting hidden patterns from this medical data. One major field where data mining medical data can be significant is the exceptional children health care systems where the self-care problems diagnosis and classification is an important challenge. Since, self-care problems classification is a time-consuming process and requires expert occupational therapists, using an expert system in classifying these problems can decrease cost and time, efficiently. Expert healthcare systems refer to the systems that are based on machine learning and artificial intelligence methods, which have the ability to learn, infer, and develop an automated tool that is capable of identifying relevant medical information. In this thesis, we use hierarchical clustering to discover interesting information from self-care problems of children with physical and motor disability We rely on unsupervised machine learning techniques to perform automatic classification of binary data and discover outliers and rare occurrences. With our proposed system, we seek insights among patients.

The advantages of using such a model are twofold. The first contribution is the advancement of knowledge about how to identify self care problems in children. The second contribution of this thesis is the presentation of comprehensive experiments to evaluate clustering binary healthcare data. By explicitly, exploring binary distances in hierarchical clustering, resampling techniques in clustering imbalanced data and feature selection preprocessing in a clustering framework, we find insights among patients and significant results.

1.2 What are Self Care problems?

The inability to look after and take care of one's self in a healthy way is referred to as self-care problems. These problems usually stem from physical and motor disabilities and disorders that restrict individual activities [8]. Disability diagnosis and classification is complex processes that require expert occupational therapists. Thus, the accuracy of identifying a child's self care problem is highly influenced by the expert's skills and experience. Moreover, given the fact that the data being analyzed is high dimensional, that same expert is inherently affected by inconsistencies in interpreting the data.

Therefore, in 2001, the World Health Organization (WHO) designed an approach that aims to identify and understand each child and family's unique functional abilities and needs. It is called the International Classification of Functioning, Disability and Health [28], or "the ICF" for short.

From the ICF, a branch called the International Classification of Functioning, Disability and Health-Children and Youth (ICF-CY) was created in 2007, which is a multipurpose classification conceptual framework. ICF and ICF-CY are used frequently as conceptual frameworks in disability evaluation, assessment, and classification [23].

The ICF-CY divides a health condition into 3 interconnected parts. These parts are connected in such a way that, if you influence one part, all the other parts change to adjust. First, the body structure and function, which explains how body parts work. Second, the activity and participation, which explain what people do, and how they engage with the world and third, the contextual factors, which include environmental and personal factors that help people function [5].

Thus, using the ICF approach to view health and help people to:

- Understand strengths and challenges, and see a broader picture of development.
- Communicate and advocate better about a child's functional needs to the health care

professionals and community

- Make informed decisions and engage actively in a child's health care.

1.3 Why Clustering Analysis?

Clustering is an unsupervised data mining technique used for grouping the data points without advance knowledge of the class labels. The objective of clustering is to find the intrinsic similar grouping in a set of unlabeled data. As shown in Figure 1.1, you can see a total of three clusters through the coloring of the data points. Formally, clustering can be defined as followed:

Definition 1. [Clustering] Clustering is a mathematical procedure for multi-dimensional analysis. Given the characteristics of a set of objects this procedure groups similar objects into clusters the resulting individual groups contain the objects that are most similar to each other when compared to those outside of their group.

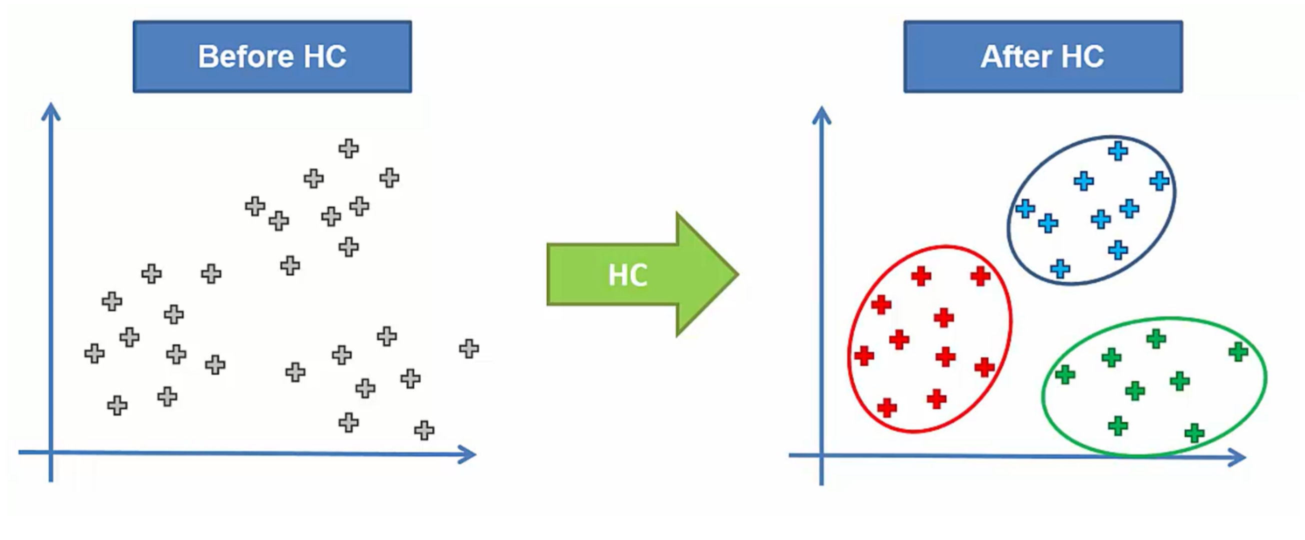


Figure 1.1: Three clusters

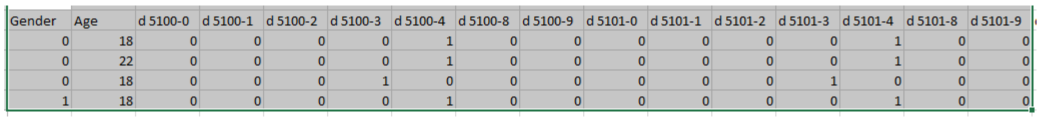
Unlike classification, which is a branch of supervised learning that approximates the relationship between input and output to predict the target class of given data points, clustering algorithms learn from the multiple variables supplied in the data, it then groups the data points without being told the logic by which to do it. Thus, clustering is often used for discovering hidden patterns and outliers in data.

Most importantly to this thesis, clustering has been proven to be an effective method for discovering structure in medical datasets [17]. As researched in many medical domains such as breast cancer [3], heart disease and heart attacks [2], and leukamia [29]. In particular to hierarchical clustering, Chen et al. [6] proposed an integrated approach for analyzing gene expression data, while Liu et al. [9] used hierarchical clustering and K-means to predict the severity of disease in patients using gene expression profile having Rheumatoid Arthritis.

The main objective of this work is to create a mathematical clustering system to predict the self care problems based on experimental data and contribute to the study of unlabeled binary healthcare data, as 80% of the world's data is unstructured and unlabeled. [25]. Also, our contribution to clustering self care problems may serve as the first step in creating a semi-supervised learning expert system, which would be able to use our proposed clustering analysis to learn reliable classification model from small number of labeled instances and large quantities of unlabeled data.

1.4 Experimental Data

Using a standard dataset is a critical factor in designing and creating an effective expert system. Thus, in this research we use a relatively new standard dataset called SCADI (Self-Care Activities Dataset based on ICF-CY), which was introduced in 2018 [20]. The SCADI dataset is the first and only dataset that has been created for the purpose of data mining self care problems based on ICF-CY to this date. Our model is created based on binary measurements performed on 70 children with physical and motor disabilities based on ICF-CY. Thus, our work additionally studies the problem of clustering binary data.



Gender	Age	d 5100-0	d 5100-1	d 5100-2	d 5100-3	d 5100-4	d 5100-8	d 5100-9	d 5101-0	d 5101-1	d 5101-2	d 5101-3	d 5101-4	d 5101-8	d 5101-9
0	18	0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	22	0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	18	0	0	0	1	0	0	0	0	0	0	1	0	0	0
1	18	0	0	0	0	1	0	0	0	0	0	0	1	0	0

Figure 1.2: Experimental data excerpt

The SCADI dataset contains 203 binary features (one feature per each activity deficiency). If the particular patient has an activity deficiency, the value of the activity's respective feature is set to 1 otherwise set to 0. Given the relatively large number of features, we also tackle the curse of dimensionality.

In addition to the 203 measurements, the last column is the target class, which refers to self-care problems classification that were determined by experts. Each child is classified to one of the 7 categories based on their self-care features by examinations of occupational therapists [30]. Also, the first two columns consist of patient gender and age, respectively. Consequently, the data set contains 70 samples and 205 features. A data excerpt is shown in Figure 1.2.

We give an in-depth description for the experimental data in Chapter 3.

1.5 Summary of Our Work

In this thesis, we first present some popular clustering algorithms. We then discuss the SCADI dataset in depth. Followed by a discussion of hierarchical clustering. More specifically, we point out the fundamental concepts of linkage methods and distance/ dissimilarity metrics in hierarchical clustering and how to apply them to our findings. Lastly, we present our results and conclusion.

CHAPTER 2
CLUSTERING ALGORITHMS

Clustering data is useful in many disciplines and has a wide range of applications. Because the use cases are plentiful, clustering is often subjective and the means that can be used for achieving a clustering solution are vast. In fact, there are more than 100 clustering algorithms [15]. However, in general, the differences in each clustering algorithms and the performance of the clustering can be traced back to these four basic components [19]:

1. the dataset used for clustering
2. the distance/dissimilarity measures between data points
3. the criterion/objective function which the clustering solutions should aim to optimize
4. the optimization procedure

In this chapter, we will go over a few most popular clustering algorithms to contrast the method we chose for our research: agglomerative hierarchical clustering.

2.1 Types of Clustering

In terms of how datapoints items are grouped into clusters, we distinguish the difference between hard and soft clustering [10].

Definition 2. [Hard clustering] In hard clustering, each data point either belongs to a cluster completely or not.

Definition 3. [Soft clustering] In soft clustering, instead of putting each data point into a separate cluster, a data points assignment a distribution over all clusters. In other words, a data point has fractional membership in several clusters.

In summary, hard clustering produces disjoint groups of dataset items, whereas soft clustering produces non-disjoint groups.

2.2 Clustering Methods

The algorithms for clustering can be organized as described in the following.

2.2.1 Centroid-based Clustering

In centroid-based clustering, the goal is to locate the center points of each cluster. The most common example of centroid based clustering is K-means. K-means starts with an initial partition with K clusters and assign patterns to clusters to reduce the squared error. Given the number of clusters thought to be in the dataset, it finds a local optimum, and is commonly run multiple times with different random initializations. Variations of k-means include: (k-medoids), choosing medians (k-medians clustering), choosing the initial centers less randomly (k-means++) or allowing a fuzzy cluster assignment (fuzzy c-means). As an example of centroid based algorithm, the main steps of K-means algorithm are as follows:

1. Chose the number K of clusters
2. Select at random K points, the centroids
3. Assign each data point to the closet centroid which forms K clusters
4. Compute and place the new centroid of each cluster
5. Reassign each data point to the new closest centroid. If any reassignment took place, go to step 4 until DONE

Also, another common algorithm called the mean shift clustering, which attempts to find dense areas of data points, fall in this category as well. In general, an advantage of centroid based clustering is that the algorithm can be fast, with the linear complexity of $O(n)$. A disadvantage is that the optimization functions for centriod based clustering are known to be NP-hard; thus the common approach is to search only for approximate solutions.

2.2.2 Distribution-based Clustering

The clustering model most closely related to probability theory is based on distribution models. It identifies the probability that a point belongs to a cluster, around each possible centroid. Clusters can easily be defined as the data points most likely belonging to the same distribution. One prominent method is known as Gaussian mixture models which uses the expectation-maximization (EM) algorithm to cluster data points. One can think of Gaussian mixture models as generalizing k-means clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussian. [22] The main steps are:

1. Randomly assign K cluster centers (K Gaussian)
2. Refine the clusters based on two steps for each data point: Expectation step and Maximization step
   - for each point, estimate the probability that each (Gaussian) cluster generates it
   - modify the parameters to maximize the likelihood of the data

Other distribution based clustering algorithm is Beta-binomial mixture model clustering. Often mixture models are used for soft clustering as they deal with probability; in order to obtain a hard clustering, objects are often then assigned to the distribution they most likely belong to. The disadvantages of distribution based clustering are that the clustering often suffers from over-fitting, which leads to poor generalization of data and centroid based clustering, mixture models converge to a local optimum. Lastly, assuming a certain distribution for data is a rather strong assumption, thus these type of algorithms can be hard to implement and interpret with much certainty.

2.2.3 Density-based Clustering

Density clustering clusters data points by how densely populated the clusters are. Therefore, data points in these sparse areas are usually considered to be noise. Similar to hierarchical based clustering, it is based on connecting points within certain distance thresholds. However, it only clusters points that satisfy a pre-defined density criterion. The most popular density based clustering method is DBSCAN. Also, OPTICS is a generalization of DBSCAN that falls within density based clustering. An example procedure for density based clustering as followed:

1. The algorithm begins with an arbitrary starting data point. The neighborhood of this point is extracted using a distance criterion

2. If there are a sufficient number of points according to the distance criterion within this neighborhood then the algorithm clusters the neighborhood. Otherwise, the point will be labeled as noise. In both cases that data point is marked as visited.

3. This process of steps 2 is repeated until all points in the cluster are determined i.e all points have been visited and labelled to a cluster or determined to be random noise.

Density based clustering is fairly low on time complexity and often, there is no need to run it multiple times. However, the key intuitive drawback of DBSCAN is that they expect some kind of density drop to detect cluster borders. For example: on datasets with overlapping clusters, the DBSCAN are impractical.

2.2.4 Hierarchical Clustering

Lastly, hierarchical clustering, also known as connectivity-based clustering, is based on the core idea that data points closely more related are nearby. These algorithms cluster data points together based on their distances. Thus, hierarchical clustering does not require a pre-specified number of clustering. The hierarchical methods can be divided into two approaches: the first approach is called the agglomerative or bottom-up approach and it is most popular. It starts the process with each data point in its own cluster and with each step, it will merge the two most similar clusters. The hierarchical process stops when all of the data points are aggregated into a single cluster. The second hierarchical approach is the divisive or top-down approach. It starts the process with all of the objects in a single cluster and then removes the outsiders from the least cohesive cluster. This process stops when each data point is in its own cluster. In the general case, the time complexity of agglomerative clustering is $O(n^3)$ and $O(2^{n-1})$ for divisive clustering, which makes them too slow for large data sets.

Hierarchical classifications produced by either the agglomerative or divisive clustering may be represented by a two-dimensional diagram known as a dendrogram, which illustrates the fusions or divisions made at each stage of the analysis.Dendrograms may be used in interpreting the results of hierarchical clustering techniques [10]. Given the dataset in Table 2.1 and the distance matrix in Table 2.2 constructed using agglomerative clustering, the example of a dendrogram is given in Figure 2.1.

2.3 HIERARCHICAL CLUSTERING FOR SCADI DATASET

For the purpose of our work we have chosen the agglomerative hierarchical clustering method. Our particular choice was based on two significant advantages. The first one is that our data set is relatively small, hence easy to process on a regular computer. The second one is that the hierarchical clustering results can be visualized in many meaningful ways, relative to the significance of the data.

Hierarchical clustering will be covered in more depth in Chapter 4.

	A	**B**	**C**	**D**	**E**	**F**	**G**
Alex	5	0	0	0	2	1	2
Bruce	2	1	2	0	0	0	0
Chris	0	0	1	4	0	0	1
Don	0	0	2	0	1	1	2
Emil	6	2	2	0	1	1	2
Fred	4	0	0	2	0	0	2

Table 2.1: Sample dataset

	Alex	**Bruce**	**Chris**	**Don**	**Emil**
Bruce	.83333				
Chris	.83333	.8			
Don	.4	.83333	.6	0	
Emil	.33333	.5	.71428	.33333	
Fred	.6	.8	.5	.83333	.71428

Table 2.2: Sample distance matrix given the sample dataset

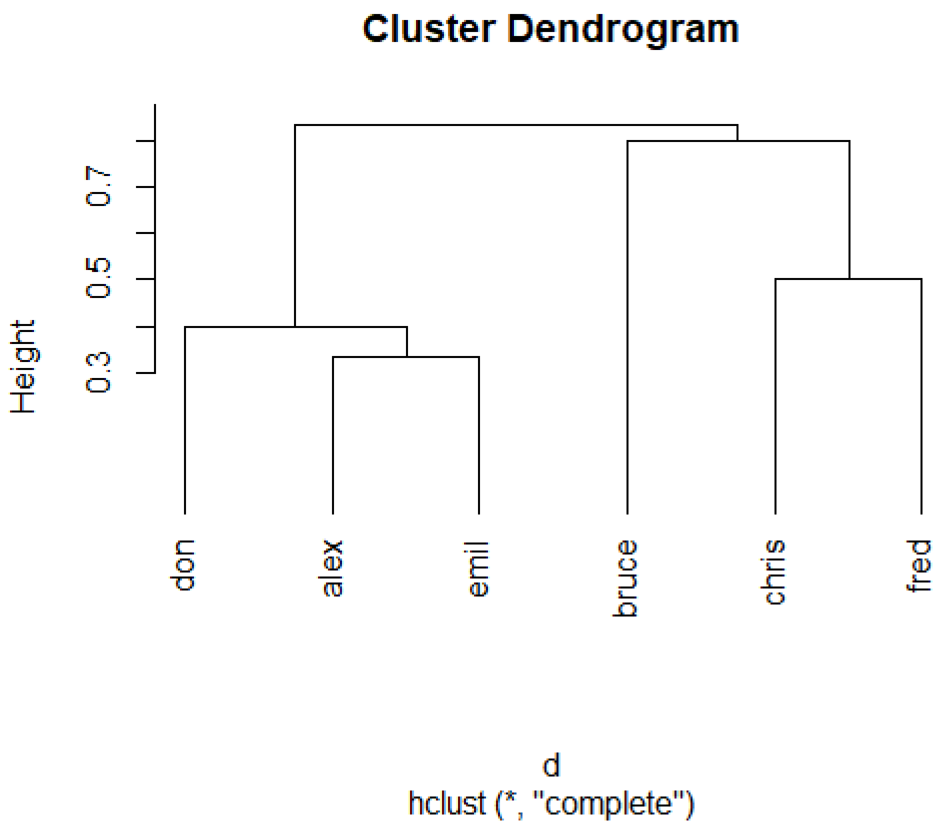


Figure 2.1: Dendrogram example

CHAPTER 3
SCADI DATASET

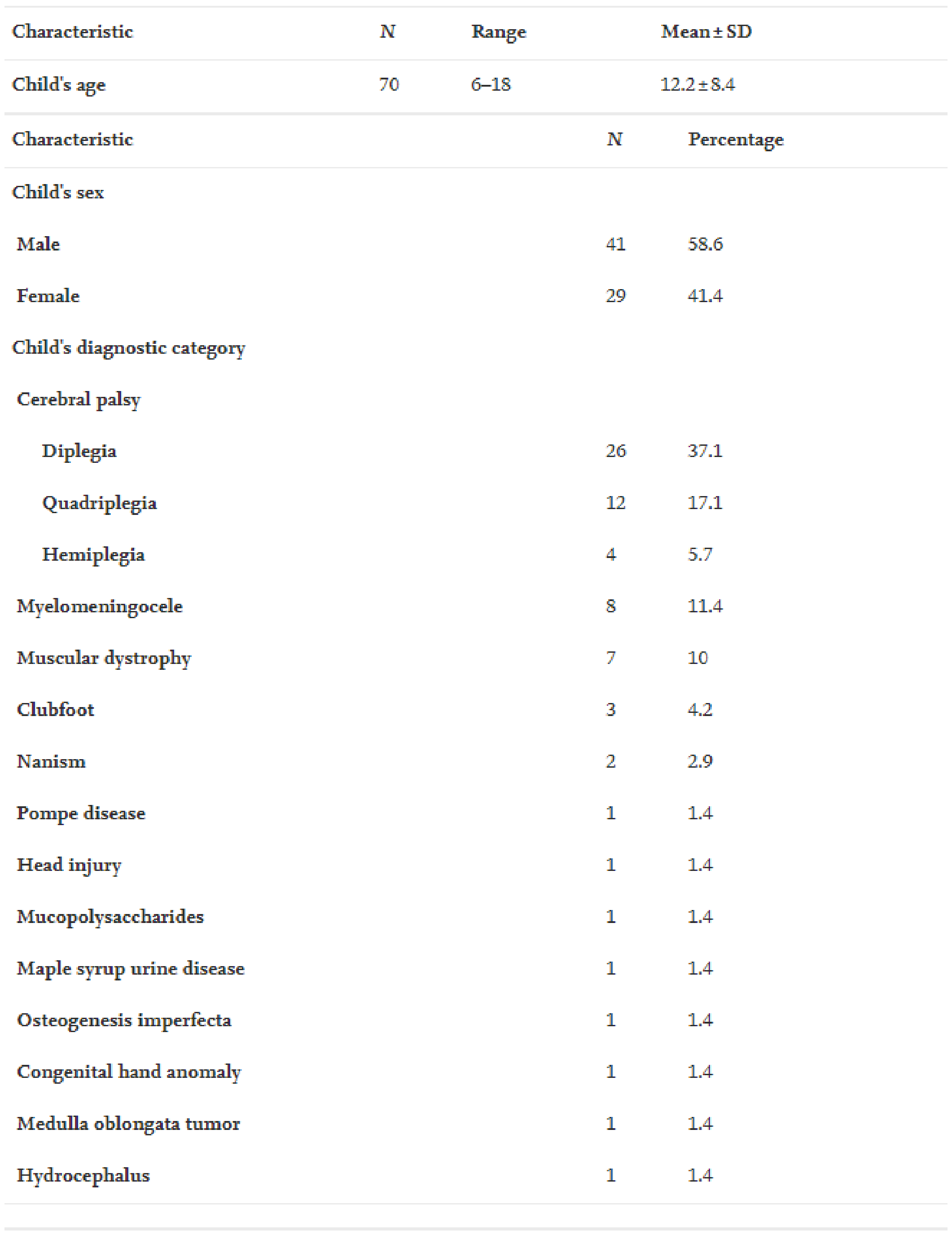
We have informally introduced self-care problems diagnosis and classification and briefly described our dataset in Chapter 1. However, in this chapter, we will formally explore the data, describe the existing literature works, and analyze two problems within the SCADI dataset.

3.1 Data Collection

Our SCADI dataset was collected by Zarchi et. al. in collaboration with two expert occupational therapists, who both had over 15 years in the field. To collect the records, 70 students (children) who studied in the three educational and health centers in Yazd, Iran from the period of from 2016 to 2017 were investigated [18]. The 70 children in this study were categorized into seven classes based on their self-care activities by the therapists as shown in Table 3.1, which is used as the target class in the dataset.

Since this data follows the ICF-CY framework for identifying the severity of each self care problem, Table 3.2 shows that each child can have one of the seven statues of impairment in the 29 self activities [31]. Also, a table of the 29 self care activities is found in the appendix.

The age of these children were from 6 to 18 years old. Figure 3.1 shows the ages, gender and disorder associated with each child. As inferred by this figure, genetic disorders, serious injuries are among the main causes of physical disabilities. Also, Cerebral Palsy, spinal cord injuries, and certain types of brain injuries worth mentioning [14]. However, each child and family’ s unique functional abilities and needs are distinct regardless of diagnosed disorder.



Characteristic	N	Range	Mean ± SD
Child's age	70	6–18	12.2 ± 8.4

Characteristic	N	Percentage
Child's sex		
Male	41	58.6
Female	29	41.4
Child's diagnostic category		
Cerebral palsy		
Diplegia	26	37.1
Quadriplegia	12	17.1
Hemiplegia	4	5.7
Myelomeningocele	8	11.4
Muscular dystrophy	7	10
Clubfoot	3	4.2
Nanism	2	2.9
Pompe disease	1	1.4
Head injury	1	1.4
Mucopolysaccharides	1	1.4
Maple syrup urine disease	1	1.4
Osteogenesis imperfecta	1	1.4
Congenital hand anomaly	1	1.4
Medulla oblongata tumor	1	1.4
Hydrocephalus	1	1.4

Figure 3.1: Disorders included in this dataset

Target classes	
No.	Description
0	Caring for body parts problem
1	Toileting problem
2	Dressing problem
3	Washing oneself and Caring for body parts and Dressing problem
4	Washing oneself, Caring for body parts, Toileting, and Dressing problem
5	Eating, Drinking, Washing oneself, Caring for body parts, toileting, Dressing, Looking after one's health and Looking after one's safety problem
6	No Problem

Table 3.1: Classes of activities

3.2 SCADI Literature Review

To this point, all of existing works on the SCADI dataset have employed as a supervised classifier to predict self-care problems. The original researchers [31] classified the self-care problems with 83.1% accuracy with an ANN and extracted self-care classification rules using C4.5, a popular decision tree algorithm.

While B. Islam et.al. (2018) proposed combining a PCA based approach with a host of different types of classification techniques, they suggested a final combined PCA and k-nearest neighbors algorithm (KNN) approach resulting in 84.29% accuracy [14]. Care2Vec [23] is a supervised learning algorithm approach where the researchers used autoencoders and deep neural networks as a two step modeling process to achieve an accuracy of 84.29%. Focusing on feature selection, Choudhury suggested Random Forest with the help of a feature reduction technique called Boruta algorithm, which minimizes the data dimensionality to advocate the minimal-optimal set of predictors. Their algorithm gives the classification accuracy of 84.75% [8].

Code	Value	Description
xxx.0	0-4%	NO impairment
xxx.1	5-24%	Mild impairment
xxx.2	25-49%	Moderate impairment
xxx.3	50-95%	Severe impairment
xxx.4	96-100%	Complete impairment
xxx.8		not specified
xxx.9		not applicable

Table 3.2: SCADI extent impairment codes

Attempting to take the target class imbalance into account, [14] proposes a robust framework using an over-sampling technique called Synthetic Minority Over-sampling Technique (SMOTE) and the extreme gradient boosting algorithm to improve the prediction performance for the SCADI dataset. The overall accuracy of the proposed framework reaches 85.4%. Lastly, M. K. Kele and . Kl used seven of 205 features selected by the artificial bee colony feature selection (ABC-FS) algorithm to achieved a 88.5714% accuracy rate.

3.3 Dimensionality Reduction and Preprocessing

Dimensionality reduction is one of the most critical areas of machine learning as it is used to reduce the redundancy of features and exploit hidden information in high-dimensional data and generally improve model performance. Dimensionality reduction can be categorized mainly into feature extraction and feature selection. In the feature extraction approach, features are projected into a new space with lower dimensionality. Examples of these techniques includes Principal Component Analysis (PCA) and Linear Discriminant

Analysis (LDA). On the other hand, the supervised feature selection approach aims to select a small subset of features that minimize redundancy and maximize relevance to the target class [1]. Feature selection is characterized into four categories: filter model, wrapper model, embedded model and hybrid model. The first category of approaches is the filter-based approaches, such as Mutual Information. This type of method assigns weight to each feature. The second category is a wrapper method, such as recursive feature elimination (RFS), which treats the selection of subsets as a search optimization problem that generates different combinations of features until the algorithm finds the best subset. The third method is the embedded method, such as regularization and random forests. Lastly, the hybrid is the combination of the filter and wrapper methods and attempts to take advantage of both the filter and wrapper methods to achieve the optimal performance.

Both dimensionality reduction approaches can improve learning performance in clustering. However, feature extraction is often criticized for its lack of interpretability since it maintains the original feature values in the reduced space. In addition, since supervised feature reduction selects features discriminate samples that belong to different classes, researchers often need labeled samples as training samples in order to select these features. In supervised learning, it is easy to define what relevant feature means. However, in unsupervised machine learning data like clustering, defining relevancy becomes unclear.

Studies have proven that dimensionality reduction may help improve unsupervised learning in a way similar to improving the supervised learning.[[27], [1]] Furthermore, different relevant features may produce different clustering, which could greatly help discovering different hidden patterns in the data. Motivated by these facts, different clustering techniques were proposed to utilize feature selection methods to improve clustering efficiency and quality. These methods are called unsupervised feature selection [1].

In this work, we will compare several dimensionality reduction techniques. Examples will be from feature extraction, supervised feature reduction and unsupervised feature selection for clustering.

3.4 Class Imbalance Problem

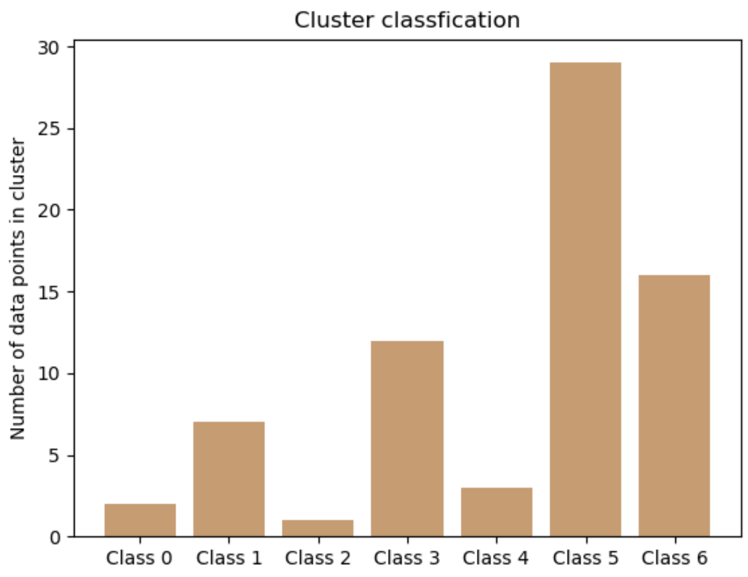


Figure 3.2: Distribution of target classes

While analyzing the SCADI dataset, one of our first things we noticed was the class imbalanced problem. As in many datasets, the class imbalance problem is the problem in machine learning where the total number of a class of data is far less than the total number of another class of data. In other words, the data has an uneven distribution between classes. For example: in the SCADI dataset, class 6 has 29 instances while class 2 has 1 instance as shown in Figure 3.2.

This problem is extremely common in practice and can be observed in various disciplines from fraud detection to facial recognition. In particular, identifying rare and significant conditions and diseases in imbalanced healthcare datasets is especially difficult ([24, 32]).

Although the effect of imbalanced classes is well researched in supervised learning techniques, very little is known about how imbalance affects clustering. Li Xuan et. al. studied this issue in 2013 [11]. Their experimental results indicated that the class-imbalance of the dataset can seriously influence the final performance and efficiency of the clustering algorithm. They also concluded that the higher the uneven ratio between classes, the higher the adverse effects of the clustering performance.

Among the few approaches for handling the class imbalance problem, resampling techniques are the most popular. Its goal is to get a better balance between the classes of the samples. These techniques can be categorized as oversampling techniques or undersampling techniques [18]. In oversampling, algorithms generally generate more synthetic examples of the minority class while in undersampling, the amount of examples of the majority class is reduced through various algorithms.

To deal with the class imbalanced problem occurring in the SCADI dataset, we will use a fitting algorithm from both undersampling and oversampling to view the effect of each on our clustering results.

CHAPTER 4

AGGLOMERATIVE HIERARCHICAL CLUSTERING OF SCADI

In this chapter, we will formally describe how we used hierarchical clustering to create models for clustering self care data. The chapter is organized as follows. Section 4.1 presents the concepts of distance and similarity measures on dataset items. In Section 4.2 we present a few methods for computing similarity/dissimilarity between subsets of a dataset. We then give an overview of the agglomerative hierarchical clustering algorithm in Section 4.3. Some measures for the quality of clusters are described in Section 4.4.

4.1 Distances and Similarities

Most efforts to produce a simple group structure from a complex data set require a measure of "closeness," or "similarity." There is often a great deal of subjectivity involved in the choice of a distance measure. Important considerations include the nature of the variables (discrete, continuous, binary). scales or measurement (nominal, ordinal, interval, ratio), and subject matter knowledge. The SCADI dataset we use in our work falls in the category of binary data. Hence, our main focus will be on measuring similarity/dissimilarity of binary data.

Quantitatively, the proximity of two elements of a set can be measured by a distance function, which computes the distance between every pair of elements in the set.

Definition 4. Let S be a dataset. A distance function on set S is a function

$$d_S : S \times S \to [0, \infty)$$

satisfying the following properties:

1. $d_S(x, y) \geq 0, \quad \forall x, y \in S$

2. $d_S(x, y) = 0 \Leftrightarrow x = y$

3. $d_S(x, y) = d_S(y, x)$

4. $d_S(x, y) \leq d_S(x, z) + d_S(z, y)$

The last property in Definition 4 (the triangle inequality) may be difficult to be satisfied in some practical situations, especially when dealing with noncontinuous dataset. A similarity function is used instead. Similarity functions are often defined as the negative of a distance function. However, in many practical situations similarity functions do not have a distance function correspondent, as the triangle inequality is not satisfied.

In the following section, we will give definitions for a few popular distance measures for binary data. A good survey of similarity/dissimilarity measures is given in [7].

4.1.1 Binary Data Similarity and Dissimilarity Measures

Unlike numerical distances, such as Euclidean and Manhattan, binary distances cannot be represented by meaningful p-dimensional measure because some pairs of items are often compared on the basis of the presence of absences of certain characteristics as represented in the SCADI dataset. The presence and absences of a characteristic can be described mathematically by introducing a binary variable, which assumes the value 1 if the characteristic is present and the value of 0 if characteristics is absent.

The binary similarity and dissimilarity (distance) measures play a critical role in pattern analysis problems such as classification, clustering, etc. Since the performance relies on the choice of an appropriate measure, many researchers have taken elaborate efforts to find the most meaningful binary similarity and distance measures over a hundred years. [19]. Numerous binary similarity measures and distance measures have been proposed in various fields. For example, the Jaccard similarity measure was used for clustering ecological species. Recently, they have been actively used to solve the identification problems in biometrics such as fingerprint, iris images, and handwritten character recognition.

Our goal is to test the performance of traditional hierarchial clustering methods for treating binary data and evaluate their characteristics.

To allow for differential treatment of the 1-1 matches and the 0-0 matches, several schemes for defining similarity coefficients are used. To introduce these schemes, we arrange the frequencies of matches and mismatches for items i and k in the form of a contingency table (Table 4.1).

		Item k		
		1	**0**	**Totals**
Item i	**1**	a	b	a+b
	0	c	d	c+d
	Totals	a+c	b+d	a+b+c+d

Table 4.1: Contingency table for binary data

Distances / Dissimilarity metrics		
Jaccard	$\frac{a}{a+b+c}$	No 0-0 matches (The 0-0 matches are treated as irrelevant)
Dice	$\frac{2a}{2a+b+c}$	Revised Jaccard
Hamming	$b+c$	the Manhattan distance for binary data
Sokal-Michener	$\frac{a+d}{a+b+c+d}$	Equal weights for 1-1 and 0-0 matches
Russell-Rao	$\frac{a}{a+b+c+d}$	No 0-0 matches in numerator
Sokal-Sneath	$\frac{a+d}{b+c}$	Ratio of matches to mismatches
Kulsinski	$\frac{a}{b+c}$	Ratio of matches to mismatches with 0-0 matches excluded

Table 4.2: Some similarity/dissimilarity metrics

4.2 Linkage Methods

In this section, we concentrate on agglomerative hierarchical procedures and in particular linkage methods. Linkage is the process that merges the two most similar clusters based on a similarity measure for any pair of two subsets of the dataset. The approaches we discuss are single linkage, complete linkage, average linkage, and Ward's linkage.

Definition 5. Let $S = \{x_1, x_2, \ldots, x_n\}$ be a data set and let $C_i, C_j \in 2^S$ be non-empty disjoint sets. Let $d_S : S \times S \to R^{\geq 0}$ be a dissimilarity measure on S. Then $d : 2^S \times 2^S \to \mathbb{R}^{\geq 0}$ is defined as follows:

1. Single linkage clustering:

$$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} d_s(x, y) \tag{4.1}$$

2. Complete linkage clustering:

$$d(C_i, C_j) = \max_{x \in C_i, y \in C_j} d_s(x, y) \tag{4.2}$$

3. Average linkage clustering:

$$d(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d_s(x, y) \tag{4.3}$$

4. Ward clustering:

$$d(C_i, C_j) = \sum_{x,y \in C_i \cup C_j} [d_S(x, y)]^2 - \left(\sum_{x,y \in C_i} [d_S(x, y)]^2 + \sum_{x,y \in C_j} [d_S(x, y)]^2 \right) \tag{4.4}$$

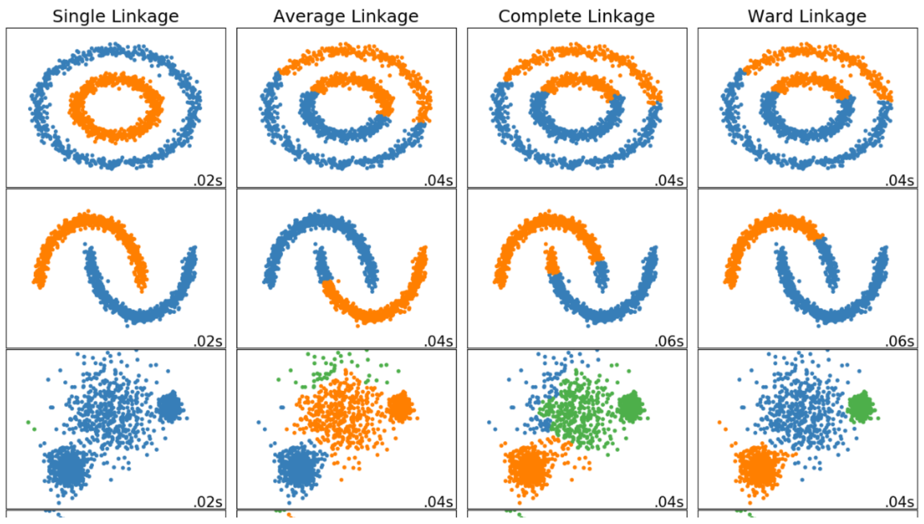


Figure 4.1: Comparing different hierarchical linkage methods

4.2.1 Single Linkage

The inputs to a single linkage algorithm are the distances or similarities between pairs of objects. Single linkage measures the distance between the closest pair of points. Since single linkage (Figure 4.1, the first column) joins clusters by the shortest link between them, the techniques cannot discern poorly separated clustering. On the other hand, single linkage is one of the few clustering methods that can delineate non-ellipsoidal clusters. The tendency of single linkage to pick out long stringlike clusters is know as chaining, which can be misleading if items at opposite ends of the chain are quite dissimilar.

4.2.2 Complete Linkage

Complete linkage (Figure 4.1, the second column) measures the distance between the farthest pair of points in the clusters at each stage. Thus, complete linkage ensured that all items in a cluster are within some maximum distances (minimum similarity) of each other.

4.2.3 Average Linkage

Average linkage (Figure 4.1, the third column) measures the average distance between all of the points. In other words, average linkage treats the distances between two clusters as the average distance between all pairs of items where one member of a pair belongs to each cluster. For average linkage clustering, changes in the assignment of distances (similarities) can affect the arrangement of final configuration of clusters even though the changes preserve relative orderings. Also, average linkage is less affected by outliers.

4.2.4 Ward's Method Linkage

Ward's minimum variance criterion minimizes the total within-cluster variance. Thus, this method does not directly define a measure of distance between two points or clusters. It is an ANOVA based approach. At each stage, two clusters merge that provide the smallest increase in the combined error sum of squares. An example of clusters produced using Ward's method is given in Figure 4.1, the last column.

4.3 Agglomerative Hierarchical Clustering

Informally, the agglomerative hierarchical clustering method creates groups of data items starting with individual data points and proceeding iteratively by joining pairs of data items/groups that are closest at each iteration step. The process of clustering a data set $S = \{x_1, x_2, \ldots, x_n\}$ produces a sequence $C^{(k)}$ of disjoint coverings of S, where:

$$\begin{aligned} C^{(k)} &= \{C_i^{(k)}\}, \quad k = 1...n,\ i = 1, \ldots, n-k-1 \\ S &= \bigcup_{i=0}^{n-k-1} C_i^{(k)} \\ \emptyset &= \bigcap_{i=0}^{n-k-1} C_i^{(k)} \end{aligned}$$

Formally, the process is described by Algorithm 1. The algorithm takes as input a dataset

Algorithm 1 Hierarchical clustering

```
1: procedure HierarchicalClust($S, d$)    ▷ agglomerative hierarchical clustering
2: Input: data set $S = \{x_1, x_2, \ldots, x_n\}, d : 2^S \times 2^S \to \mathbb{R}^{\geq 0}$
3: Output: sequence of partitions $C^{(k)} = \{C_i^{(k)}\}$ and distances $\{d_k\}$, $k = 1, \ldots, n$,
   $i = 1, \ldots, n - k - 1$
4:     $C^{(0)} = \{C_i^{(0)}\}, C_i^{(0)} = \{x_i\}$
5:     $D^{(0)} = \left[d_{ij}^{(0)}\right], d_{ij}^{(0)} = d(C_i^{(0)}, C_j^{(0)})$
6:     for $k = 1, \ldots, n$ do
7:         $p, q = \arg\min_{i,j}\{d_{ij}^{(k-1)}\}, p < q$
8:         $d_k = d_{p,q}^{(k-1)}$
9:         $C^{(k)} = \{C_i^{(k)}\}, C_i^{(k)} = C_i^{(k-1)}$ if $i \neq p, q$
10:        $C_p^{(k)} = C_p^{(k-1)} \cup C_q^{(k-1)}$
11:        $D^{(k)} = \left[d_{ij}^{(k)}\right], d_{ij}^{(k)} = d_{ij}^{(k-1)}$, if $i, j \neq p, q$
12:        $d_{pj}^{(k)} = d(C_p^{(k)}, C_j^{(k)}), d_{jp}^{(k)} = d_{mj}^{(k)}$
13:    end for
14:    return $C^{(k)}, \{d_k\}$    ▷ Returns the clusters, distances
15: end procedure
```

S (line 1) and produces a sequence of partitions $C^{(k)}$ (line 3), where each partition is a complete cover of dataset S. Also, a sequence of distances $\{d_k\}$ is produced, where each d_i in the sequence represents a distance between the two partitions that merge in step i. The algorithm completes in n steps (lines 6 – 13): at each step i, the two closest partitions from the current sequence of partitions $C^{(i)}$ are merged into one larger partition. After n steps, all partitions are merged, hence $C^{(n)} = S$. A dendrogram (as in Figure 2.1) presents visually the process described in Algorithm 1: a bottom-up scanning of the dendrogram reveals which partitions merge and at what distance (represented on the "Height" scale on the left hand side, which records the sequence of distances $\{d_k\}$). Clearly, only a monotonically increasing sequence of distances $\{d_k\}$ would produce an un-tangled dendrogram. Surprisingly, we have determined that the monotonicity of the sequence does not depend on the distance/similarity choice for the elements in the dataset S. It rather depends on the linkage method, that is, the distance/similarity between clusters. The following results will establish when such a monotonic sequence of distances (hence an un-tangled dendrogram) would be produced by Algorithm 1.

Lemma 4.3.1. *The sequence $\{d_k\}$ produced by the Algorithm 1 is monotonic if and only if $d_{pj}^{(k)} \geq d_k$.*

Proof. We prove by induction. Clearly, $d_k \leq d_{ij}^{(k-1)}$ (lines 7, 8). Hence $d_1 \leq d_{ij}^{(0)}$, $d_2 \leq d_{ij}^{(1)}$.

From line 9: $d_{ij}^{(0)} = d_{ij}^{(1)}, \;\; i, j \neq m, n$.

From lines 7 and 9:

$$\begin{aligned} d_2 &= min\left(\left\{d_{ij}^{(1)}\right\}_{i,j\neq m,n}, \left\{d_{mj}^{(1)}\right\}\right) \\ &= min\left(\left\{d_{ij}^{(0)}\right\}_{i,j\neq m,n}, \left\{d_{mj}^{(1)}\right\}\right) \\ &\geq min\left(\left\{d_{ij}^{(0)}\right\}_{i,j\neq m,n}, d_1\right) \\ &= d_1 \end{aligned}$$

Which proves the base case.

For the induction step, assume that $d_{k-1} \leq d_k$.

Similar as before:

$$\begin{aligned} d_2 &= min\left(\left\{d_{ij}^{(k)}\right\}_{i,j\neq m,n}, \left\{d_{mj}^{(k)}\right\}\right) \\ &= min\left(\left\{d_{ij}^{(k-1)}\right\}_{i,j\neq m,n}, \left\{d_{mj}^{(k)}\right\}\right) \\ &\geq min\left(\left\{d_{ij}^{(k-1)}\right\}_{i,j\neq m,n}, d_k\right) \\ &= d_k \end{aligned}$$

□

These results allow us to conclude that the hierarchical dendrogram (as the example in Figure 2.1) can be produced using any similarity or dissimilarity measure and without necessarily having the triangle identity satisfied. The hierarchical dendrogram is only influenced by the choice of linkage (similarity measure between clusters).

The following result can be established for the linkage methods discussed in Section 4.2.

Theorem 4.1. *The single, complete, average, and Ward linkages used with Algorithm 1 produce monotonically increasing sequences* $\{d_k\}$.

Proof. Using the result of Lemma 4.3.1, all we need to show is that each linkage method computes a distance between clusters (as defined in Definition 5) no less than distances between all pairs of elements within each cluster. We will treat each linkage method one at the time.

1. Single linkage clustering (4.1):

$$d(C_i, C_j) = \min_{x\in C_i, y\in C_j} d_s(x,y) \geq \max\left(\max_{x,y\in C_i} d_s(x,y), \max_{x,y\in C_j} d_s(x,y)\right)$$

   since clusters C_i, C_j were previously created with smaller distances.

2. Complete linkage clustering:

$$d(C_i, C_j) = \max_{x \in C_i, y \in C_j} d_s(x, y) \geq \min_{x \in C_i, y \in C_j} d_s(x, y)$$

   the the results for single linkage applies.

3. Average linkage clustering:

$$d(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d_s(x, y) \geq \min_{x \in C_i, y \in C_j} d_s(x, y)$$

   then the result from single linkage applies.

4. Ward clustering:

$$d(C_i, C_j) = \sum_{x,y \in C_i \cup C_j} [d_S(x, y)]^2 - \left(\sum_{x,y \in C_i} [d_S(x, y)]^2 + \sum_{x,y \in C_j} [d_S(x, y)]^2 \right) \geq \min_{x \in C_i, y \in C_j} d_s(x, y)$$

   then the result from single linkage applies.

□

A different linkage method, called k-link, is presented as a tool for statistical disclosure limitation. The k-link algorithm uses hierarchical clustering, however, it produces a non-monotonic sequence $\{d_k\}$, which is not appropriate for producing meaningful dendrogram representations.

4.4 Clustering Evaluation and Metrics

Unlike classification algorithms' evaluation, evaluating the performance of a clustering algorithm is not as trivial as counting the number of errors (accuracy) or precision and recall [22]. In general, there are two types of performance assessments used for clustering techniques:

- Supervised (external), which uses the target class ground truth for each sample's evaluation.

- Unsupervised (internal), which measures the quality of the model itself.

In clustering literature, it is common to measure the performance of a clustering solution based on how well it recovers the class labels. Thus, we will use Adjusted Rand Index for external evaluation. Additionally, we will use the Silhouette Coefficient score for internal evaluation. The larger the values of these metrics are, the better the clustering effect is.

4.4.1 Adjusted Rand index(Supervised)

Given the knowledge of the target class assignments and the clustering algorithm assignments of the same samples, the adjusted Rand index is a function that measures the similarity of the two assignments, penalizes both false positive and false negative decisions during clustering.

If C is a ground truth class assignment and K the clustering, we define a and b as:

- a, the number of pairs of elements that are in the same set in C and in the same set in K
- b, the number of pairs of elements that are in different sets in C and in different sets in K

The raw (unadjusted) Rand index is then given by:

$$\text{RI} = \frac{a+b}{C_2^{n_{samples}}}$$

where $C_2^{n_{samples}}$ is the total number of possible pairs in the dataset.

However, the RI score does not guarantee that random label assignments will get a value close to zero. To counter this effect we can discount the expected RI of random labelings by defining the adjusted Rand index as follows:

$$\text{ARI} = \frac{\text{RI} - E[\text{RI}]}{\max(\text{RI}) - E[\text{RI}]} \tag{4.5}$$

Formally defined as:

$$\overbrace{ARI}^{\text{Adjusted Index}} = \frac{\overbrace{\sum_{ij}\binom{n_{ij}}{2}}^{\text{Index}} - \overbrace{[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}^{\text{Expected Index}}}{\underbrace{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}]}_{\text{Max Index}} - \underbrace{[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}]/\binom{n}{2}}_{\text{Expected Index}}} \tag{4.6}$$

where n_{ij}, a_i, b_j are values from the contingency table.

4.4.2 Silhouette Coefficient(Unsupervised)

The Silhouette Coefficient is defined for each sample and is composed of two scores:

- a: The mean distance between a data point and all other points in the same cluster.
- b: The mean distance between a data point and all other points in the next nearest cluster.

The Silhouette Coefficient s for a single sample is then given as:

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

Moreover, the Silhouette Coefficient for a set of samples is given as the mean of the Silhouette Coefficient for each sample. Scores around zero indicate overlapping clusters. The score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster.

CHAPTER 5
EXPERIMENTS AND RESULTS

The following experiments were all implemented in Python v3.7.1, running on Windows 10, 64-bit Intel Core i5 CPU @3.40GHz, 16GB RAM.

5.1 Experiment Methods

We use the Python package, Scipy, to perform the hierarchical clustering on our data while using Scikit-learn for metrics and supervised feature selection, Scikit-feature for unsupervised feature reduction, and Imblearn for resampling techniques.

5.2 Results

Our experiments were organized in three major categories:

1. Baseline and interpretation of clustering of results
2. Dimensionality Reduction
3. Oversampling

These experimental results are reported in the subsequent subsections.

5.3 Baseline

Initially, all 203 self care activities features are selected and considered as a baseline for experimentation. We tried several different binary distance experiments including the most prominent distance metrics : Hamming, Jaccard, Russell-Rao, Sokal-Michener and Rogers-Tanimoto, and Dice. Although, these distances yielded distinctly different matrices, the hierarchical clustering resulted in the same clusters, ARI, and Silhouette index score for

	ARI	Silhouette
single	0.548375614	0.216576493
complete	0.734546335	0.214894955
average	0.695697749	0.272727374
ward	0.606364312	0.21899189

Table 5.1: Baseline results for all distances

all linkage methods (single, complete, average) with only slight variations using the Ward's method linkage.

As seen in 5.1, the complete linkage has the best clustering performance in terms of Adjusted Rand index while average linkage performs best when looking at the Silhouette index.

5.4 Association

Using the different linkages, we used visualization to explore the association between classes. Looking from the perspectives of how clusters are contained in each class and how children of a given class are assigned to a cluster, we are able to make some interesting observations.

As shown in Figure 5.4, single linkage does the worst job at clustering. Since single linkage joins clusters by the shortest link between them, the techniques cannot discern poorly separated clustering. With this method, at any step, two clusters are merged if their closest edges are close enough and no proximity between other parts of the two clusters are taken into consideration, producing the chaining pattern described in Chapter 4. Nevertheless, by grouping children from class 0, 1, 2, 3, 4 in cluster 2 and representing cluster

1 in class 5, single linkage shows that it is can identify the children with self care problems quite easily, but it cannot distinguish between the classes. Moreover, single linkage seems to group the children with no self care problem (class 6) as clusters 0, 2, 3, 4, and 5 are indicating the clustering may recognize the children with no problem as outliers. Also, Figure 5.2 shows that although single linkage does a bad job at classifying classes overall, single linkage clustering can still identify class 5 very well.

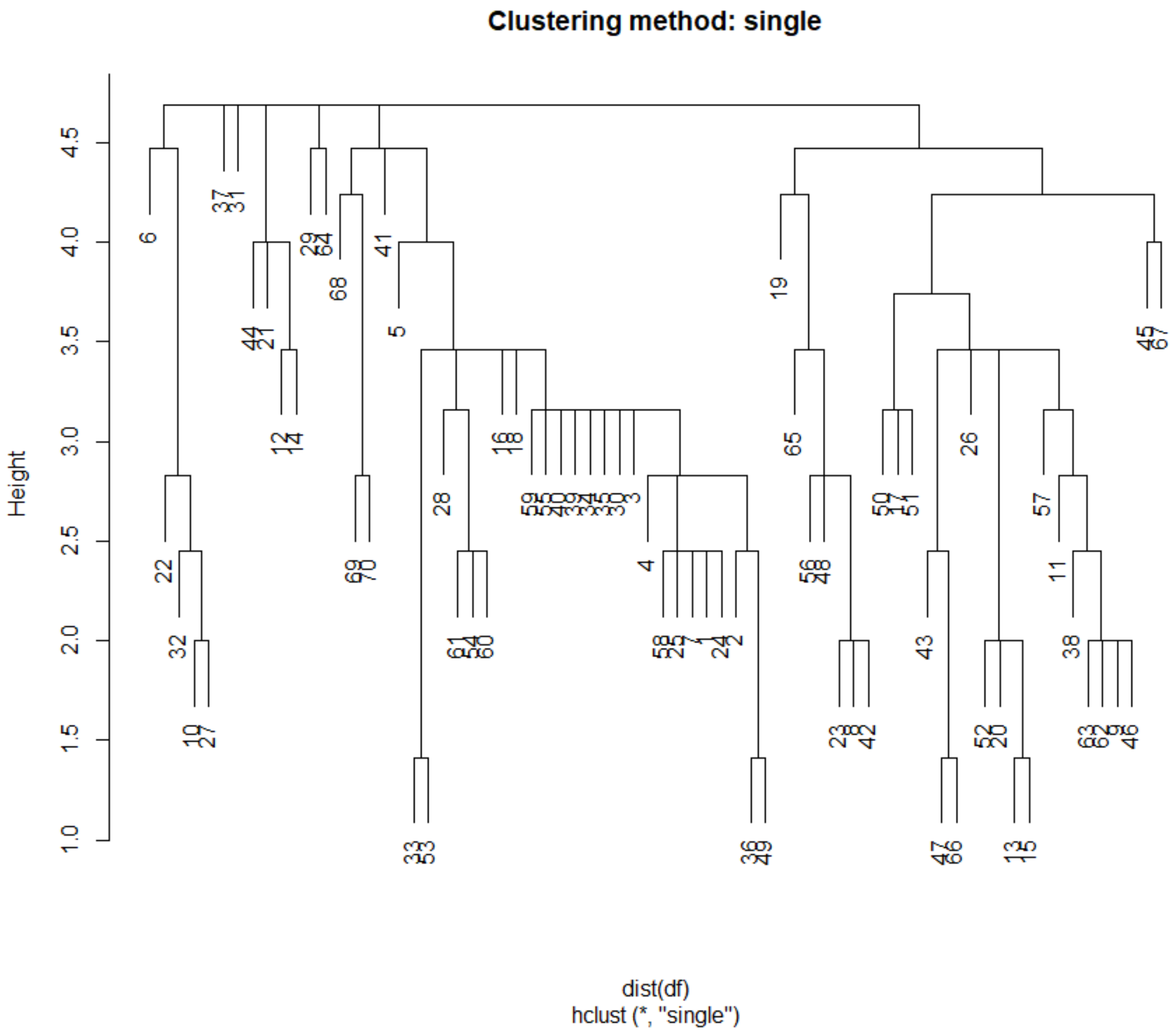


Figure 5.1: Single Linkage Dendrogram

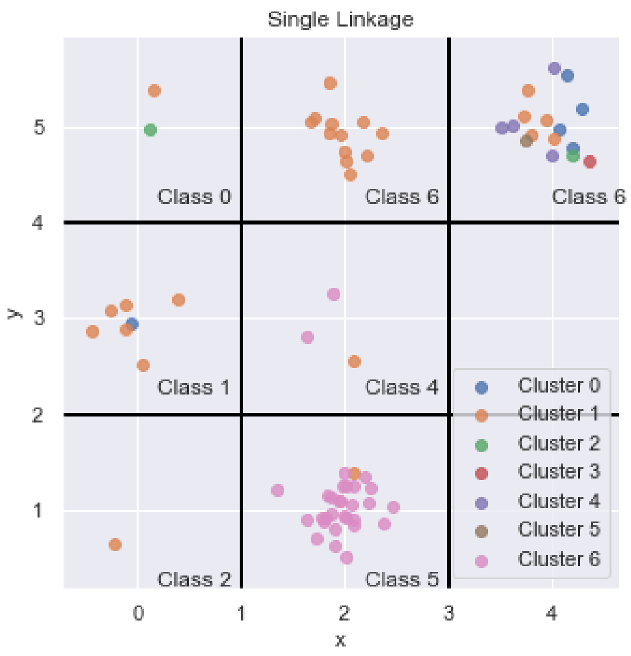


Figure 5.2: Single Classes

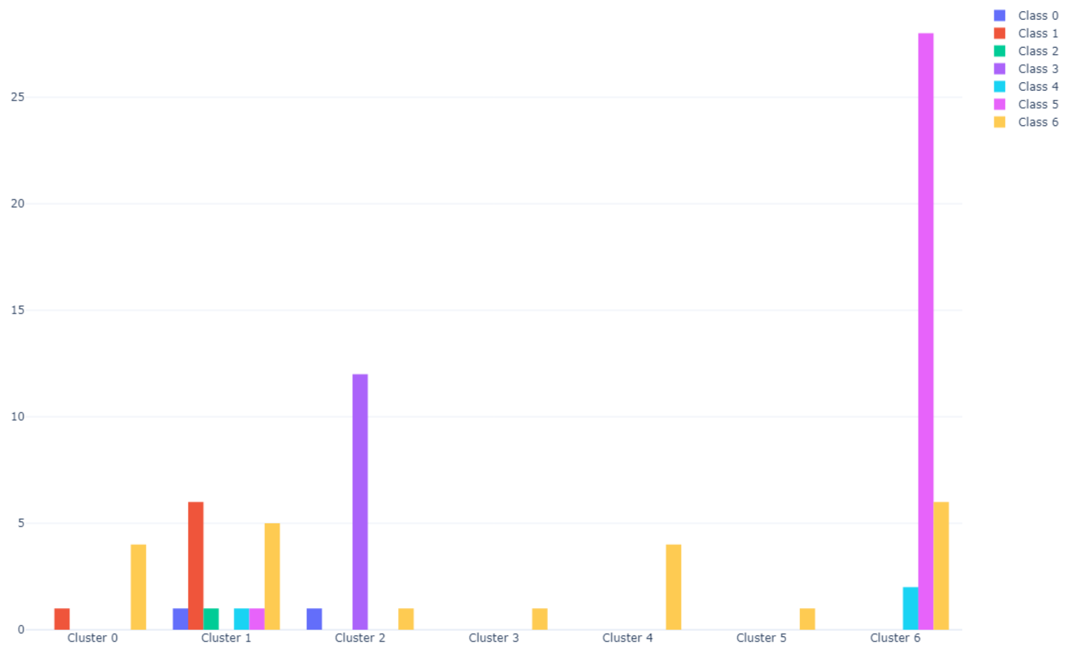


Figure 5.3: Single Clusters

Average linkage certainly does a better job at clustering than single linkage, often being seen as the middle ground between single and complete. As $\frac{28}{29}$ data points represented by cluster 1 is contained in class 5, this shows that hierarchical clustering does a very good job at identifying children that have problems with the most severe self care problems. Furthermore, it can be seen that cluster 4 is well represented in children who have problems with washing themselves, caring for body parts, and dressing themselves. These observations can be seen very well in Figure 5.2. However, as noted in Chapter 4, average linkage is less affected by outliers, thus resulting in a slightly lower ARI than complete linkage but performs best when looking at the Silhouette index.

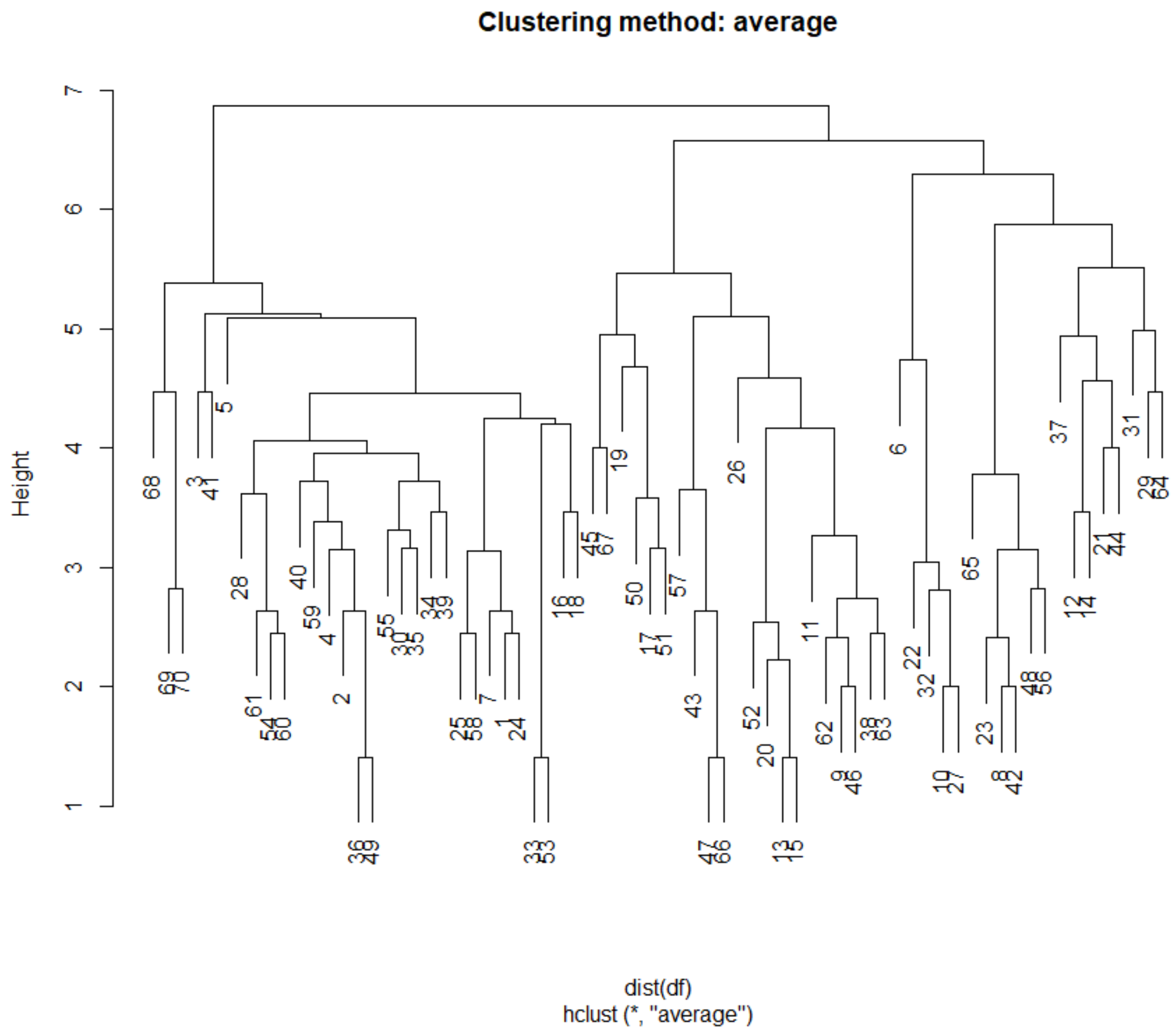


Figure 5.4: Average Linkage Dendrogram

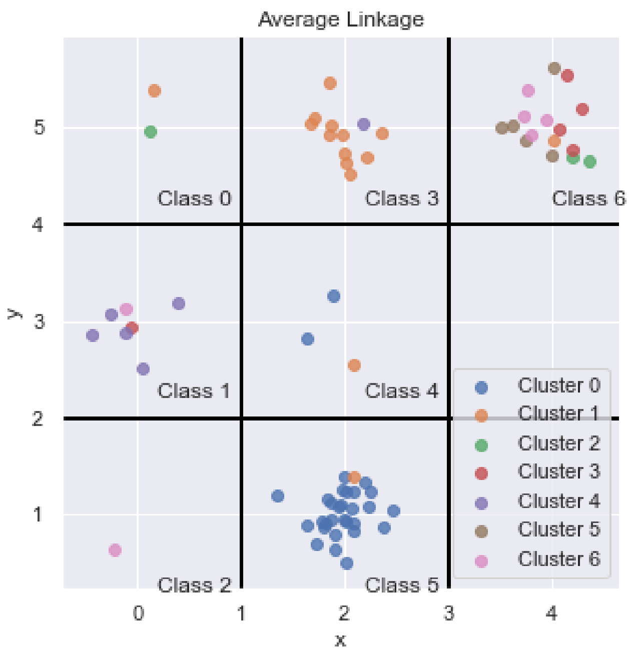


Figure 5.5: Average Classes

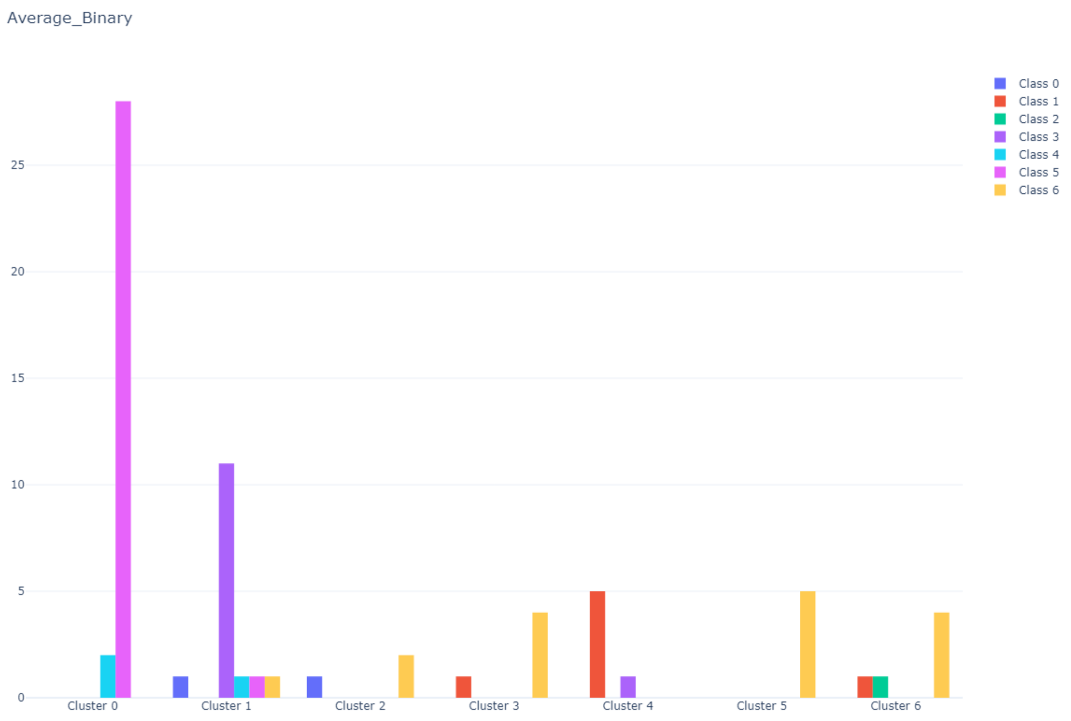


Figure 5.6: Average Clusters

According to ARI, complete linkage performs the best and outperforms average linkage by a small margin. Figure 5.2 shows that class 6 (the children with no problem) appears in less of the clusters in complete linkage than in average, which may explain the small difference. Also, this figure shows that cluster 2 contains children from several different classes except for class 1 which contain children with a toileting problem, which may indicate children in this class have distinctly different need than those of another classes. This can also be seen in average linkage clustering.

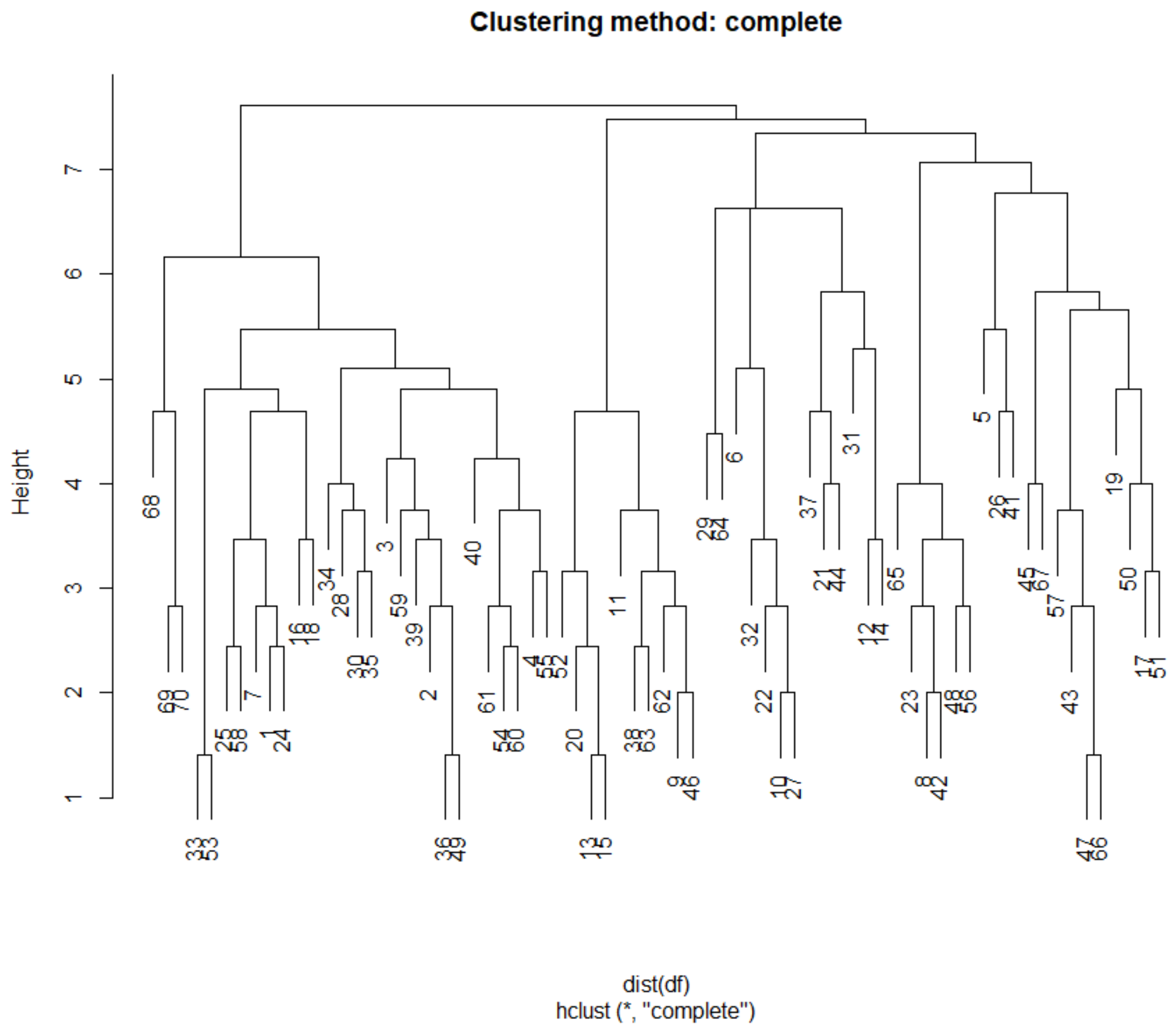


Figure 5.7: Complete Linkage Dendrogram

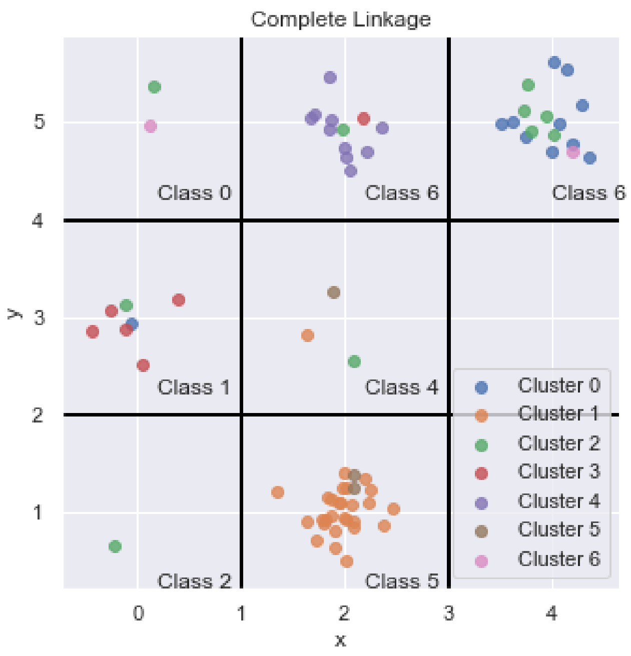


Figure 5.8: Complete Classes

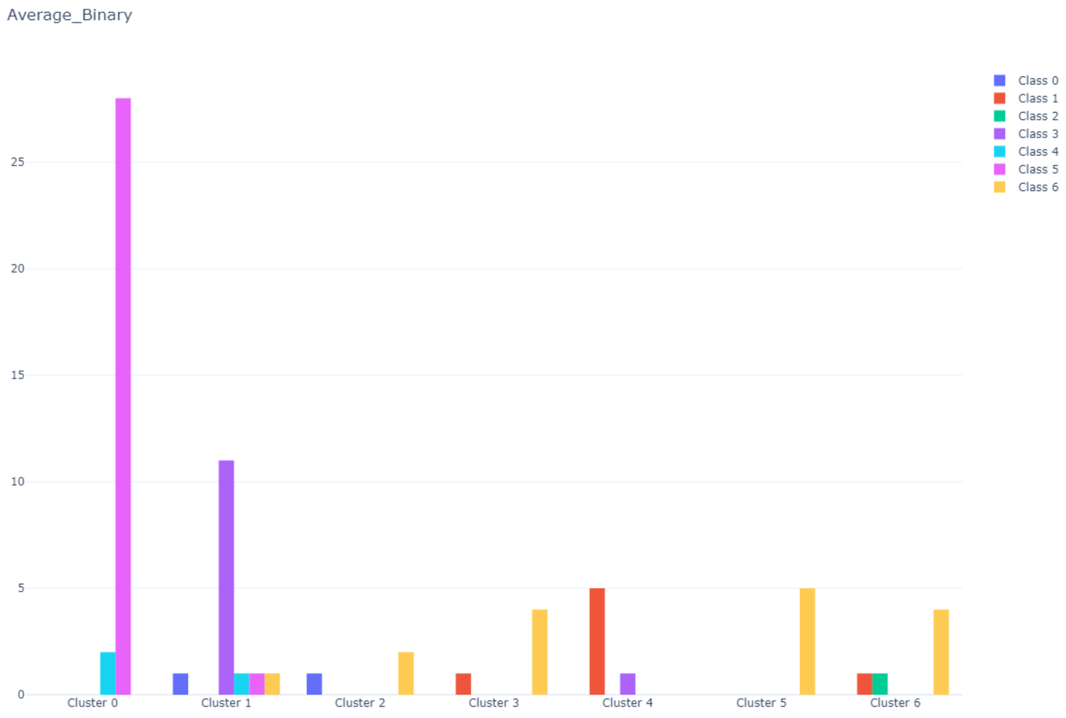


Figure 5.9: Compete Clusters

5.5 Dimensionality Reduction

Comparing several dimensionality reduction techniques, the following examples will be from feature extraction, supervised feature reduction and unsupervised feature selection algorithms.

5.5.1 Feature Extraction

As an example for feature extraction, we experimented with Principal Component Analysis (PCA) [14] to reduce the dimension from 203 features to k principal components. The steps of PCA are as follows:

- Normalizing the data
- Calculating the covariance matrix
- Calculating the eigenvectors, eigenvalues of the covariance matrix
- Choosing principal components and translating the data in terms of the components

Using Euclidean distance, we observed that the number of principal components greatly effect the performance of the clustering algorithm. The performance depends on distribution of data and correlation among various dimensions. In fact, after 10 principal components, which explain 76% of the data, the performance of the clustering worsens monotonically in the best performance linkage. This may indicate that at least 24% of the data acts as random noise. Also, we see an interesting decreasing pattern in the silhouette coefficient score. As this metric does not use the labeled data in its evaluation, we get a natural look at how the similarity of the data points in respect to the cluster decrease as the number of principal components increase.

Number of PCs	2	5	10	15	20	25
Explained Variance	0.417891897	0.604567818	0.761573849	0.84656986	0.901212466	0.93656631

Number of PCs	2	5	10	15	20	25
Linkage	Average	Average	Complete	Complete	Complete	Average
ARI	0.569244	0.627822	**0.748332**	0.672945	0.672945	0.627822
Silhouette Coefficient	**0.500275**	0.452096	0.405408	0.271755	0.248918	0.256006

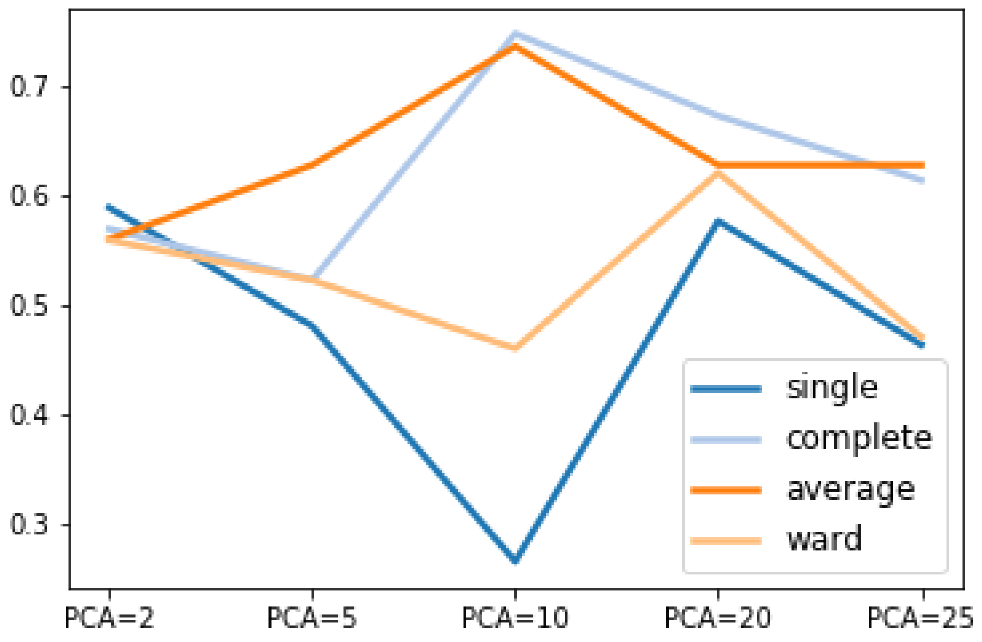


Figure 5.10: ARI for K principal components

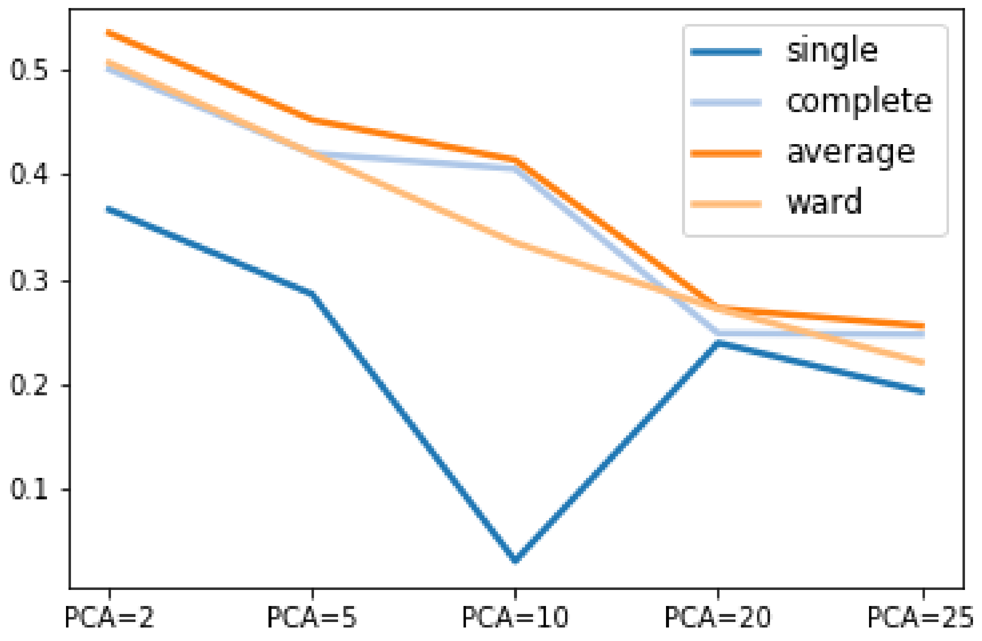


Figure 5.11: Silhouette Coefficient for K principal components

5.6 Feature Selection Methods

In this section, we compare the following supervised and unsupervised feature reduction techniques.

- Supervised
    - Decision tree based feature selection, which calculates feature importance based on the best performing features as close to the root of the decision tree.
    - Boruta algorithm, which is a wrapper algorithm that advocates the minimal-optimal set of predictors using the random forest classification algorithm.
- Unsupervised
    - Laplacian Score feature selection, which selects the features most consistent with the Gaussian Laplacian matrix
    - Multi-Cluster Feature Selection (MCFS), which selects features using spectral regression with ℓ_1 norm regularization

Since the tree based feature selection and Boruta algorithm selection only selects a subset of features that pass a certain criteria threshold, the number of features in their subsets are fixed at 7 and 55, respectively. To fairly compare different unsupervised feature selection algorithms, we set the number of selected features as $\{10, 50, 100, 150\}$ and report the best results of all the algorithms using those parameters. In our experiments, each feature selection algorithm is first performed on the selected features. Then agglomerative hierarchical clustering is performed based on the selected features. Because the results of hierarchical clustering depend on both binary distance and linkage, we report the best results corresponding distance and linkage as well.

Comparing the performance of different feature selection algorithms, the experiment results are shown in Table 5.6. We can see from the table that the clustering results of our

	Decision tree	**Laplacian Score**	**MCFS**	**Boruta algorithm**
Number of features	7	50	50	55
Linkage	Average	Ward	Ward	Complete
Distance metric	Dice	Dice, Sokalsneath, Jaccard	Dice , Sokal-Michener, Sokal-Sneath	Dice
ARI	.742981	0.728500	0.727157	**.755235**
Silhouette Coefficient	**.454532**	0.274813	0.259167	.364461

baseline clustering are better than the best results of both unsupervised methods. However, because the feature number is significantly reduced by performing unsupervised feature selection methods, clustering is more efficient and if we were clustering binary data points, which were truly unlabelled, the unsupervised method would perform fairly well. On the other hand, the results from the supervised feature selection algorithms are generally better than baseline and also more efficient. We can also see that the tree-based feature selection gains the second best performance with the least amount of features. Decision trees keep the most important features based on most impactful features (entropy), thus this feature selection captures such a high dimensional dataset with a small amount of features. Also, this method has the best silhouette coefficient score, which indicates that each data point is well matched to its own cluster and poorly matched to neighboring clusters.

Finally, we observed that the Boruta algorithm identifies 55 variables as significant variables. This technique reduced the dataset by approximately 74% while improving prediction accuracy. The Boruta algorithm decides whether a variable is relatively important and statistically significant, which is the main reason for its improved performance on ARI.

5.7 Resampling

In our last experiments, we looked at the imbalance in the SCADI dataset. Given that we performed the feature selection experiments first, we were able to use the Boruta algorithm results (best clustering results) to build our imbalanced data experiments. As mentioned in Chapter 3, we experimented with both an oversampling and undersampling resampling technique to view the effect of each on our clustering results.

5.7.1 Oversampling

Because the SCADI dataset is relatively small, we used the naive approach to oversampling called the Random Oversampler Technique. It is the simplest method of oversampling that picks sample data points of the underrepresented classes at random with replacement. Unlike other oversampling methods such as the Synthetic Minority Oversampling Technique (SMOTE) and its variants and the Adaptive Synthetic Sampling (ADASYN), Random Oversampler does not create new samples of data. Therefore, our experiments involving oversampling perform worse than expected. The Random Oversampler technique, sampled each underrepresented class until each of the 7 classes had 29 samples, increased the sample size from 70 to 203. Table 5.2 shows the poor results. All binary distances have very similar results.

	ARI	Silhouette
single	0.18921105	0.273848863
complete	0.347668253	0.258072703
average	0.285687288	0.361976726
ward	0.553352354	0.448533275

Table 5.2: Oversampling results

5.8 UNDERSAMPLING

When performing undersampling, we reduced the amount of examples of the majority class through algorithms. To experiment with undersampling, we choose the popular technique of Tomeks links, which is a technique that removes data points from the majority classes where its nearest neighbor in distance is a prominent data point from an underrepresented class. In our experiment, the algorithm only removes two data points: one data point from class 6 which represents the children with no self care problems and one data point from class 5 which represents the children with problems of eating, drinking, washing oneself, caring for body parts, toileting, dressing, looking after themselves, health, and safety. This resulted in the sample size decreasing from 70 to 68. Table 5.3 shows the results from this experiment with the Dice binary distance metric.

	ARI	Silhouette
single	0.693048	0.315539
complete	**0.762579**	**0.379568**
average	**0.762579**	**0.379568**
ward	0.698158	0.383298

Table 5.3: Undersampling results

These results indicate that our hierarchical clustering results would be slightly better with a more balanced SCADI dataset.

CHAPTER 6
CONCLUSION

In this thesis, we perform hierarchical clustering on binary data, using a few similarity measures to discover interesting information from self-care problems of children with physical and motor disability data. We showed how using different similarity measures and linkage methods affect one's ability to perform automatic classification of binary data, and, in general, discover outliers and rare occurrences in data. Moreover, we proved that hierarchical clustering using single, complete, average, and Ward linkages produce monotonic sequences of cluster distances when a general distance measure is being used.

Additionally, we proposed experimental models using dimensionality reduction methods and resampling techniques that improved accuracy and computational effectiveness.

Our experimental results showed that the final model we proposed produces satisfactory results and better prediction accuracy than baseline hierarchical clustering. The best results of experiments are obtained from hierarchical clustering using the Boruta algorithm for feature selection and Tomek's links undersampling for resampling. Experimental results revealed that the proposed expert system model can classify self-care problems with 76.25% ARI (accuracy).

REFERENCES

[1] Salem Alelyani, Jiliang Tang, and Huan Liu, *Feature selection for clustering: A review*, Data Clustering: Algorithms and Applications, 2013.

[2] Gulmohamed Banu, M Phil, J H Bousal, and Jamala Mca, *Predicting heart attack using fuzzy c means clustering algorithm*, International Journal of latest Trends in Engineering and Technology **5** (2015).

[3] Smaranda Belciug, Florin Gorunescu, Marina Gorunescu, and Abdel-Badeeh M.Salem, *Clustering-based approach for detecting breast cancer recurrence*, 12 2010.

[4] Roel Bertens, Jilles Vreeken, and Arno Siebes, *Efficiently discovering unexpected pattern-co-occurrences*, 2017 SIAM International Conference on Data Mining (Houston, TX), April 2017, pp. 126–134.

[5] J Case-Smith, *Self-care strategies for children with developmental disabilities.*, Ways of living: Self-care strategies for special needs (2000), 81–121.

[6] Hugh Chipman and Robert Tibshirani, *Hybrid hierarchical clustering with applications to microarray data*, Biostatistics (Oxford, England) **7** (2006), 286–301.

[7] Seung-Seok Choi, Sung-Hyuk Cha, and Charles C. Tappert, *A survey of binary similarity and distance measures*, 2009.

[8] Avishek Choudhury, *Classification of functioning, disability, and health: ICF-CY self care (SCADI dataset) using predictive analytics*, CoRR **abs/1901.00756** (2019).

[9] Javier Escudero, John P. Zajicek, and Emmanuel C. Ifeachor, *Early detection and characterization of alzheimer's disease in clinical scenarios using bioprofile concepts and k-means*, 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2011, Boston, MA, USA, August 30 - Sept. 3, 2011, 2011, pp. 6470–6473.

[10] Brian S. Everitt, Sabine Landau, and Morven Leese, *Cluster analysis*, 4th ed., Wiley Publishing, 2009.

[11] Li Xuan; Chen Zhigang; Yang Fan, *Exploring of clustering algorithm on class-imbalanced data*, 2013 8th International Conference on Computer Science Education, April 2013, pp. 89–93.

[12] Z. He, X. Xu, J. Z. Huang, and S. Deng, *Fp-outlier: Frequent pattern based outlier detection*, ComSIS **2** (2005), no. 1, 103118.

[13] Shuqing Huang, *A comparative study of clustering and classification algorithms*, Ph.D. thesis, New Orleans, LA, USA, 2007, AAI3258261.

[14] B. Islam, N. I. M. Ashafuddula, and F. Mahmud, *A machine learning approach to detect self-care problems of children with physical and motor disability*, 2018 21st International Conference of Computer and Information Technology (ICCIT), Dec 2018, pp. 1–4.

[15] Anil K. Jain, *Data clustering: 50 years beyond k-means*, 2008.

[16] M. K. Kele and . Kl, *Artificial bee colony algorithm for feature selection on scadi dataset*, 2018 3rd International Conference on Computer Science and Engineering (UBMK), Sep. 2018, pp. 463–466.

[17] Sina Khanmohammadi, Naiier Adibeig, and Samaneh Shanehbandy, *An improved overlapping k-means clustering method for medical applications*, Expert Systems with Applications **67** (2017), 12 – 18.

[18] Tuong Le and Sung Wook Baik, *A robust framework for self-care problem identification for children with disability*, Symmetry **11** (2019), no. 1.

[19] Tao Li, *A unified view on clustering binary data*, Machine Learning **62** (2006), 199–215.

[20] M. Lichman, *UCI machine learning repository*, 2013.

[21] K. Narita and H. Kitagawa, *Outlier detection for transaction databases using association rules*, The Ninth Conference on Web-Age Information Management (WAIM) (Zhangjiajie Hunan, China), July 2008, pp. 373–380.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau,

M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research **12** (2011), 2825–2830.

[23] Sayan Putatunda, *Care2vec: A deep learning approach for the classification of self-care problems in physically disabled children*, CoRR **abs/1812.00715** (2018).

[24] Md. Mostafizur Rahman and Darryl N. Davis, *Addressing the class imbalance problem in medical datasets*, 2013.

[25] C. Schneider, *The biggest data challenges that you might not even know you have*, 2016.

[26] Alexander Strehl and Joydeep Ghosh, *Cluster ensembles — a knowledge reuse framework for combining multiple partitions*, J. Mach. Learn. Res. **3** (2003), 583–617.

[27] Luis Talavera, *Feature selection as a preprocessing step for hierarchical clustering*, ICML, 1999.

[28] Jilda Vargus-Adams and Annette Majnemer, *International classification of functioning, disability and health (icf) as a framework for change: Revolutionizing rehabilitation*, Journal of child neurology **29** (2014).

[29] S Vijayarani and S Sudha, *An efficient clustering algorithm for predicting diseases from hemogram blood test samples*, Indian Journal of Science and Technology **8** (2015).

[30] Wikipedia, *Clustering*, 2017.

[31] MS Zarchi, SMM Fatemi Bushehri, and M Dehghanizadeh, *Scadi: A standard dataset for self-care problems classification of children with physical and motor disability*, International Journal of Medical Informatics (2018).

[32] Yang Zhao, Zoie Shui-Yee Wong, and Kwok Leung Tsui, *A framework of rebalancing imbalanced healthcare data for rare events classification: A case of look-alike sound-alike mix-up incident detection*, Journal of Healthcare Engineering **2018** (2018), 1–11.

APPENDIX A

APPENDIX: DATA DESCRIPTION

A.1 Experimental Data Set Source:

https://archive.ics.uci.edu/ml/datasets/SCADI

A.2 Data Set Information:

- This dataset contains 206 attributes of 70 children with physical and motor disability based on ICF-CY.
- The 'Class' field refers to the presence of the self-care problems of the children with physical and motor disabilities.The classes are determined by occupational therapists.
- The names and social security numbers of the children were recently removed from the dataset.

A.3 Experimental Binary Feature Information:

In SCADI, 29 activities are considered for self-care based on ICF-CY. The table below shows SCADI self-care activitiy features. The 29 activities multiplied by the 7 possible gives the SCADI dataset 203 binary features.

Category No.	Self-care category	Activity No.	Description	Codes
I.	Washing oneself	1	Washing body parts	d 5100
		2	Washing whole body	d 5101
		3	Drying oneself	d 5102
II.	Caring for body parts	4	Caring for skin	d 5200
		5	Caring for teeth	d 5201
		6	Caring for hair	d 5202
		7	Caring for fingernails	d 5203
		8	Caring for toenails	d 5204
		9	Caring for nose	d 5205
III.	Toileting	10	Indicating need for urination	d 53000
		11	Carrying out urination appropriately	d 53001
		12	Indicating need for defecation	d 530010
		13	Carrying out defecation appropriately	d 530011
		14	Menstrual care	d302
IV.	Dressing	15	Putting on clothes	d 5400
		16	Taking off clothes	d 5401
		17	Putting on footwear	d 5402
		18	Taking off footwear	d 5403
		19	Choosing appropriate clothing	d 5404
V.	Eating	20	Indicating need for eating	d 5500
		21	Carrying out eating appropriately	d 5501
VI.	Drinking	22	Indicating need for drinking	d 5600
		23	Indicating need for drinking	d 5600
VII.	Looking after one’s health	24	Ensuring one’s physical comfort	d 5700
		25	Managing diet and fitness	d 5701
		26	Managing medications and following health advice	d 57020
		27	Seeking advice or assistance from caregivers or professionals	d 57021
		28	Avoiding risks of abuse of drugs or alcohol	d 57022
		29	Looking after one’s safety	d 571

Table A.1: The self care activities

APPENDIX B

APPENDIX: PYTHON CODE

B.1 BASELINE EXPERIMENTS

```
import warnings
warnings.filterwarnings('ignore')
from sklearn import metrics
import pandas as pd
from sklearn.cluster import AgglomerativeClustering
import numpy as np

from scipy.spatial.distance import squareform
from scipy.spatial.distance import pdist

from scipy.cluster.hierarchy import dendrogram, single, complete, average,fcluster
from scipy.cluster.hierarchy import weighted, centroid, median, ward

import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import cophenet

data = pd.read_csv('SCADI.csv')

print(data.shape)

X= data.loc[:,'d␣5100-0':'d␣571-9']
X=X.values

y=data.loc[:, 'Classes']
y = y.str.extract('(\d+)').astype(int) #remove class from front
#Remove strings from a float number in a column
#y=y-1
```

```
y =y.values
print(y.shape)
y=y.flatten()
print(y.shape)

distances=["dice", "hamming", "jaccard", "kulsinski", "rogerstanimoto",
           "russellrao", "sokalmichener", "sokalmichener", 'sokalsneath','euclidean']

a=[]
for dist in distances:
    a.append(pdist(X, dist))

linkage = [single, complete, average, weighted, centroid, median, ward]
data = []

for l in linkage:
    ######NOTE THAT TO GET THE RESULTS FOR EACH distaces in distances. you must
    change #a[0], in the nextline to corresponding list item for distance#######
    clusters=(fcluster(l(a[0]), t=7, criterion='maxclust'))
    #print(clusters)

#print(clusters[0])
#print(len(clusters))
    data.append(({
        'ARI': metrics.adjusted_rand_score(y, clusters),
        'AMI': metrics.adjusted_mutual_info_score(y, clusters),
        'Homogenity': metrics.homogeneity_score(y, clusters),
        'Completeness': metrics.completeness_score(y, clusters),
        'V-measure': metrics.v_measure_score(y, clusters),
        'Silhouette': metrics.silhouette_score(X, clusters)}))
```

```
results = pd.DataFrame(data=data, columns=['ARI', 'AMI', 'Homogenity',
                                           'Completeness', 'V-measure',
                                           'Silhouette'],
    index=['single', 'complete', 'average', 'weighted', 'centroid', 'median', 'ward'])
print(results)
 results.to_csv('original.csv')
```

B.2 Feature Extraction-PCA

```
import warnings
warnings.filterwarnings('ignore')
from sklearn import metrics
import pandas as pd
from sklearn.cluster import AgglomerativeClustering
import numpy as np

from scipy.spatial.distance import squareform
from scipy.spatial.distance import pdist

from scipy.cluster.hierarchy import dendrogram, single, complete, average,fcluster
from scipy.cluster.hierarchy import weighted, centroid, median, ward

import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import cophenet

data = pd.read_csv('SCADI.csv')

print(data.shape)

X= data.loc[:,'d␣5100-4':'d␣571-1']
X=X.values
```

```
y=data.loc[:, 'Classes']
y = y.str.extract('(\d+)').astype(int) #remove class from front
#Remove strings from a float number in a column
y=y-1
y =y.values
print(y.shape)
y=y.flatten()
print(y.shape)
#############################################################################
# Applying PCA
from sklearn.decomposition import PCA
pca = PCA(n_components = 10)# CHANGE NUMBER OF PRINCIPAL COMPONENTS TO 2, 10, 15, 20, 2
X_pca = pca.fit_transform(X)

explained_variance = pca.explained_variance_ratio_

print(explained_variance)
print(sum(explained_variance))
print(X_pca)
##############################################################################
a=pdist(X_res, euclidean ))

linkage = [single, complete, average, weighted, centroid, median, ward] #DICE
data = []

for l in linkage:
    #T=l(a[0])
    clusters=(fcluster(l(a), t=7, criterion='maxclust'))
    #print(clusters)

#print(clusters[0])
```

```
#print(len(clusters))
    data.append(({
        'ARI': metrics.adjusted_rand_score(y_res, clusters),
        'AMI': metrics.adjusted_mutual_info_score(y_res, clusters),
        'Homogenity': metrics.homogeneity_score(y_res, clusters),
        'Completeness': metrics.completeness_score(y_res, clusters),
        'V-measure': metrics.v_measure_score(y_res, clusters),
        'Silhouette': metrics.silhouette_score(X_res, clusters)}))

results = pd.DataFrame(data=data, columns=['ARI', 'AMI', 'Homogenity',
                                       'Completeness', 'V-measure',
                                       'Silhouette'],
     index=['single', 'complete', 'average', 'weighted', 'centroid', 'median', 'ward'])
print(results)
```

B.3 Supervised Feature Selection Experiments

For Decision Tree and Boruta Algorithm experiements, the similar code to the baseline experiments with the exception of the corresponding .csv file bring changed.

B.4 Unsupervised Feature Selection Experiments

B.4.1 MCFS Feature Selection

```
# construct affinity matrix
    kwargs = {"metric": "euclidean", "neighborMode": "knn", "weightMode": "heatKernel",
    W = construct_W.construct_W(X, **kwargs)

    num_fea = 100    # specify the number of selected features
    # CHANGE NUMBER OF FEATURE TO  10, 50, 100, 150 ACCORDING
    num_cluster = 7    # specify the number of clusters, it is usually set as the numbe
    # obtain the feature weight matrix
    Weight = MCFS.mcfs(X, n_selected_features=num_fea, W=W, n_clusters=20)

    # sort the feature scores in an ascending order according to the feature scores
    idx = MCFS.feature_ranking(Weight)

    # obtain the dataset on the selected features
    selected_features = X[:, idx[0:num_fea]]
```

B.4.2 Laplacian Feature Selection

```
# construct affinity matrix
    kwargs_W = {"metric": "euclidean", "neighbor_mode": "knn", "weight_mode": "heat_ker
    W = construct_W.construct_W(X, **kwargs_W)

    # obtain the scores of features
    score = lap_score.lap_score(X, W=W)

    # sort the feature scores in an ascending order according to the feature scores
    idx = lap_score.feature_ranking(score)

    # perform evaluation on clustering task
    num_fea = 100    # number of selected features
     # CHANGE NUMBER OF FEATURE TO  10, 50, 100, 150 ACCORDING
    num_cluster = 7    # number of clusters, it is usually set as the number of classes

    # obtain the dataset on the selected features
    selected_features = X[:, idx[0:num_fea]]
```

B.5 Resampling Experiments

B.5.1 Oversampling

```
from imblearn.over_sampling import RandomOverSampler
ros = RandomOverSampler(random_state=0)
X_res, y_res = ros.fit_resample(X, y)
from collections import Counter
print(sorted(Counter(y_res).items()))
```

B.5.2 UNDERSAMPLING

B.6 VISUALIZATIONS EXAMPLE

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm

from sklearn.datasets import make_blobs

n_samples = 70
n_bins = 3  # use 3 bins for calibration_curve as we have 3 clusters here

# Generate 3 blobs with 2 classes where the second blob contains
# half positive samples and half negative samples. Probability in this
# blob is therefore 0.5.
centers = [(0, 5), (0, 3), (0, 1), (2,5), (2,3), (2, 1), (4, 5)]
X, y = make_blobs(n_samples=[2,7,1, 12, 3,29, 16], n_features=2, cluster_std=0.3,
                  centers=centers, shuffle=False, random_state=42)
#X, y = make_blobs(n_samples=[3, 3, 4], centers=None, n_features=2,
#                   random_state=0)
#centers=None

#y[:n_samples // 2] = 0
```

```
#y[n_samples // 2:] = 1
#sample_weight = np.random.RandomState(42).rand(y.shape[0])
np.savetxt("make_graphs_random.csv", X, delimiter=",")
# ##############################################################################

df = pd.read_csv('make_graphs_random(1).csv')
# Plot the data

import seaborn as sns
sns.set()
plt.figure()
#plt.scatter(X[:, 0], X[:, 1], c='red', # all rows first columns
                #alpha=0.5, edgecolor='k')#,
                #label="Class %s" % this_y)
sns.lmplot(x='x', y='y', data=df, fit_reg=False, hue='avg', legend=False)
plt.axhline(2, color='black', lw=2)
plt.axhline(4, color='black', lw=2)
plt.axvline(1, color='black', lw=2)
plt.axvline(3, color='black', lw=2)

#plt.fig.text(0.2, 0.2,'Class 0', fontsize=9)
#plt.fig.text(0.2, 0.2,'Class 0', fontsize=9)
plt.text(0.2, 0.2, "Class␣2")
plt.text(0.2, 2.2, "Class␣1")
plt.text(0.2, 4.2, "Class␣0")
plt.text(2.2, 0.2, "Class␣5")
plt.text(2.2, 2.2, "Class␣4")
plt.text(2.2, 4.2, "Class␣3")
plt.text(4, 4.2, "Class␣6")
L=plt.legend()
L.get_texts()[0].set_text('Cluster␣0')
```

```
L.get_texts()[1].set_text('Cluster␣1')
L.get_texts()[2].set_text('Cluster␣2')
L.get_texts()[3].set_text('Cluster␣3')
L.get_texts()[4].set_text('Cluster␣4')
L.get_texts()[5].set_text('Cluster␣5')
L.get_texts()[6].set_text('Cluster␣6')
#L(loc="best")
plt.title("Average␣Linkage")
plt.savefig('Average_Linkage.png')
```