Massive Questionnaire Analysis and Sleep Apnea

by

Adam Paul Wisniewski

A thesis submitted in conformity with the requirements for the degree of Master of Science
Graduate Department of Mathematics
University of Toronto

© Copyright 2015 by Adam Paul Wisniewski
Abstract

Massive Questionnaire Analysis and Sleep Apnea

Adam Paul Wisniewski
Master of Science
Graduate Department of Mathematics
University of Toronto
2015

Massive data analysis is a new and growing field in applied mathematics. Questionnaire datasets are ubiquitous in many different fields and they usually contain massive structures. This study surveys prominent supervised and unsupervised massive data analysis techniques for classification and clustering, including k-means clustering, spectral clustering, hierarchical clustering, support vector machine, and random forest. It also explores a novel new approach to questionnaire data analysis developed by Professor Ronald Coifman that incorporates the dual geometry metric. All of these techniques are then applied to an easily obtainable and noninvasive questionnaire dataset in order to predict the severity of a patient’s obstructive sleep apnea. The resulting analysis is found to have significant predicative power for female patients.
Acknowledgements

I would like to thank Professor Hau-tieng Wu for his guidance and supervision as well as Dr. Liu Wen-Te for providing the sleep apnea dataset. Additionally, this work would not have been possible without the love and support of my family and friends.
## Contents

1 Introduction 1

2 Clustering and Classification Algorithms 3
   2.0.1 Preprocessing Datasets: Reading, Normalization, and Imputation . . . . . . 4
   2.1 $k$-mean Clustering . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
   2.2 Spectral Clustering . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   2.3 Hierarchical Clustering . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
   2.4 Support Vector Machine (SVM) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
   2.5 CART, Random Forest, and Other Ensemble Methods . . . . . . . . . . . . . . . . 12
      2.5.1 Classification And Regression Trees (CART) . . . . . . . . . . . . . . . . . . . 12
      2.5.2 Random Forest . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

3 A Questionnaire Algorithm Using Dual Geometry 16
   3.1 Creating the Initial Tree Structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
   3.2 Deriving the Dual Geometric Metric . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
   3.3 Creating a Second Tree Structure Using Dual Affinity . . . . . . . . . . . . . . . . 18
   3.4 Iterating the Process to Refine the Tree Structures . . . . . . . . . . . . . . . . . . 19
   3.5 Wavelet Transforms and Other Representations (Optional) . . . . . . . . . . . . . . . 19
   3.6 Visualizing the Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

4 Results of the Sleep Apnea Questionnaire Analysis 21
   4.1 Clustering the Dataset . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
   4.2 Dataset Analysis Using the Questionnaire Algorithm and Dual Geometry Metric . . 27
   4.3 Predicting Subject Classifications Using Supervised Learning Techniques . . . . . . . 31

5 Discussion and Conclusions 33

A The Ultrametric 35

Bibliography 36
Chapter 1

Introduction

Massive data analysis is an inherently interdisciplinary field that spans computer science, mathematics and statistics while also incorporating field specific knowledge that is dependent on the type of data being analyzed. In this study we survey some of the recent developments and applications of massive data analysis techniques to analyze datasets containing massive structures. For example, datasets containing answers to questionnaires often lend themselves to such massive data analysis. In particular, we apply these techniques to sleep apnea questionnaires that are used for patient screening purposes.

Obstructive Sleep Apnea (OSA)\textsuperscript{1} is a medical condition that causes periods of paused or shallow breathing during sleep, thereby reducing sleep quality. It is inherently hard to screen for because, in the initial approach, a doctor is forced to rely on subjective self reported symptoms before deciding what tests, if any, the patient requires. This process makes screening difficult because the main symptom of sleep apnea is poor sleep quality which can be caused by a large number of other factors. Additionally, the prevalence of sleep apnea is very high in the general population causing doctors to be overloaded with patients requiring medical care. Therefore any improvement in the efficiency of the screening process will have a large impact on health care.\textsuperscript{2}

To standardize the self reporting of symptoms and reduce some of the subjectivity, some doctors have begun to have patients fill out questionnaires to describe their symptoms. The dataset used in this thesis includes the answers to one such questionnaire combined with other patient information such as sex, weight, age, etc.\textsuperscript{3} Finding any sort of relationships between patient answers/information and their likelihood of having sleep apnea would allow doctors to prioritize patients and tests more effectively. This thesis will use massive data techniques to find such relationships within this dataset.

The main techniques applied in this thesis include the well known random forest method of supervised learning as well as the methods for questionnaire analysis developed by Professor Ronald Coifman especially the methodology presented in [14]. Other prominent techniques like $k$-mean clustering, spectral clustering, hierarchical clustering, and support vector machines will also be applied to provide a more thorough analysis as well as to allow us to compare the strengths and weaknesses of different approaches.

In Chapter 2 a literature survey will be presented that will introduce the reader to the field of massive data analysis and it will include descriptions of some prominent techniques. This will be followed by a detailed explanation of the questionnaire algorithm and the dual geometry metric in Chapter 3. This

\textsuperscript{1}Simply referred to as “sleep apnea” throughout the majority of this thesis.

\textsuperscript{2}A general overview of sleep apnea and its epidemiology can be found in [42].

\textsuperscript{3}A full list and description of the parameters in the questionnaire can be found in Table 4.1.
will then be followed by the results from applying these methods to the sleep apnea dataset in Chapter 4. These results will then be discussed in Chapter 5 alongside a brief discussion of the ultrametric as a potential mathematical justification for the usefulness of some of the techniques.
Chapter 2

Clustering and Classification Algorithms

The goal of this chapter is to introduce the reader to the field of massive data analysis as well as to present and explain some of common and powerful techniques in the field. In order to accomplish this we must begin with a more detailed definition of the field.

To understand “massive data analysis” it is helpful to understand what each of the three words mean in this context. The term ‘massive’ is different from the commonly encountered adjective “big” which usually just refers to datasets that are too large to analyze effectively without significant computational power. Instead, massive means that there is a non-linear, and possibly non-stationary, structure present in the dataset. “Data” is simply the information that is being analyzed, but it is important to note that this data does not necessarily have to be quantitative and it can come in many forms such as images or text. Lastly, the term “analysis” in this case refers to applying techniques that attempt to uncover the underlying non-linear structure of the data.

In practice, it is rare that a dataset can be analyzed in its initial form. Therefore it is usually necessary to convert it. Thus, the first step in the process is to select an “appropriate” representation of the data such as converting images to functions which extract patterns from the data or converting a set of points to a graph. In different fields, different terms might be applied to this process of representation such as “feature extraction” or “dimension reduction”. Once the data is represented in an appropriate way, it is necessary to select a metric to measure the distance/similarity/affinity between two data points. Despite these steps taking place before finding any structure, they are arguably the two most important steps in the process of effective massive data analysis. If the selected representation and metric are poor, then even the most sophisticated techniques will provide nearly useless results. It is well accepted that there is no universal approach to select the best representation and metric for all possible datasets. Instead, we have to use background knowledge in order to design the entire data analysis procedure.

Once the representation and metric are selected, a technique to carry out the analysis can be chosen. Although, it is important to note that most techniques require a specific representation to be carried out. These techniques can be separated into the categories of supervised learning and unsupervised learning. **Supervised learning** refers to techniques that employ the use of labeled training data to create functions that predict the label for a new subject. **Unsupervised learning** refers to techniques where there is no labeled training data or predetermined classes available and the algorithm attempts...
to uncover a hidden structure by finding potential classes or relationships within the data. Clustering is an example of a set of unsupervised learning techniques that aim to find clusters which are defined as naturally occurring subgroups of a population. A third category known as semi-supervised learning also exists and it refers to analysis that is done on a large dataset that only has a small subset which is labeled, but such techniques are not applied in this study.

In this thesis we will focus on how such techniques can be applied to the sleep apnea dataset. This dataset is in the form of a database where subjects are represented by the rows and sensors are represented by the columns. We will now make our terminology more precise. A sensor or parameter (both terms are used interchangeably in this thesis) is a value that corresponds to some attribute of an object. Many other names for sensors are used in literature including variable, attribute, feature, or property. In the case of the sleep apnea data, the sensors are the measurements of traits of sleep apnea patients like height, weight, age, questionnaire answers, etc.\(^1\) A subject is an object represented by a set of values corresponding to different sensors with respect to the object. Many other names for subjects are used in literature including cases, examples and data points. In the case of the sleep apnea data, the subjects are the patients.

The following sections of this chapter will provide explanations of various techniques that can be applied to such a dataset. This will include the \(k\)-means and spectral clustering algorithms that separate subjects into clusters based on their similarity. Hierarchical clustering which can be used to illustrate the different levels of similarity between subjects through the use of a tree. Which will be followed by the Support Vector Machine (SVM) method that aims to derive a hyperplane that can be used to predict which class a subject will belong to. Finally, the Random Forest method of decision tree learning will then be explained along with its ability to predict classifications of subjects through the use of a forest of decision trees.

Out of the methods mentioned above, \(k\)-means, spectral clustering, and hierarchical clustering are all unsupervised while SVM and Random Forest are supervised. It should also be noted that these techniques take advantage of five different types of data representation since the data is represented as points in Euclidean space for \(k\)-means, graphs for spectral clustering, trees for hierarchical clustering, vectors for SVM, and decision trees for Random Forest.

These algorithms should be sufficient to provide a general sense of the various techniques in massive data analysis and they will also help the reader understand the differences and similarities of the questionnaire algorithm approach with respect to other methods. We begin with clustering as it is the most simple approach in massive data analysis. However, before this is done, we must first discuss how the data is preprocessed for analysis.

### 2.0.1 Preprocessing Datasets: Reading, Normalization, and Imputation

Let \(Q\) be an input dataset that contains questionnaire data where \(Q\) has \(m\) subjects represented in its rows with information about each subject with respect to \(n\) sensors (properties) that are represented by \(Q\)’s columns. We turn this data into a matrix \(M \in \mathbb{R}^{m \times n}\). Then every cell containing no data is filled with the average for its respective column \((m'_{ji} = \mu_i \forall m_{ji} = \emptyset)\). The data is then “normalized” with respect to every column (sensor). There are various normalization techniques that can be applied, but

\(^1\)A full list and description of the sensors in the sleep apnea data can be found in Table 4.1.
in this study the standard score \((z\text{-score})\) normalization will be used.\(^2\) By normalizing \(M\) we attain a new normalized matrix \(\overline{M}\) which will be used in the analysis.

**Notation:** In all the techniques below, \(d(x, y)\) is the distance function between subjects \(x\) and \(y\) in the dataset. This is always assumed to be the Euclidean metric unless otherwise specified.

### 2.1 \(k\)-mean Clustering

The basis of the \(k\)-means algorithm (also known as Lloyd’s algorithm) is to create clusters based around \(k\) number of means. There are different ways to select the initial \(k\) means, but the method applied in this thesis is to just randomly select \(k\) points from the dataset to use as the initial means.\(^3\) Once the initial means are selected, the distance from every data point to every mean is calculated. In this case the distance is calculated using the standard *Euclidean metric*, but it is possible to use a variety of other metrics to calculate this distance as long as the notion of mean can be well defined.

Recall that the *Euclidean metric*, \(d: \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}\), is a distance measure between any two points \(u, v\) in \(m\) dimensional space. These points can be written as \(u = [u_0, u_1, ..., u_m]^{\top}\) and \(v = [v_0, v_1, ..., v_m]^{\top}\) and their distance is equal to the square root of the sum of the square of the differences between each component. Written formally:

\[
d(u, v) = \sqrt{\sum_{i=0}^{m} (u_i - v_i)^2}.
\]

Once these distances are calculated, each point is put in the cluster corresponding to the mean that has the smallest distance to that point. Once every point is placed in a cluster, new means are created by taking the average of all the points in the cluster for each component and assigning these averages as the new means. This process is then run again and all points are clustered based on their distance to these new means. The process is repeated until the assignment of points to clusters doesn’t change.\(^4\) More formally the algorithm can be described like so:

Given the dataset \(\chi = \{x_i\}_{i=1}^{n} \subset \mathbb{R}^n\) and an initial set of means \(\mu_0^{(0)}, \ldots, \mu_K^{(0)} \in \mathbb{R}^n\) with \(K > 1\). We begin by partitioning \(\chi\) into the initial set of clusters \(S^{(0)}_\ell\) where \(\ell = 1, \ldots, K\) by setting

\[
S^{(0)}_\ell := \{x_i : ||x_i - \mu^{(0)}_\ell|| < ||x_i - \mu^{(0)}_j|| \forall j \neq \ell\}.
\]

After \(S^{(0)}_\ell\) is calculated, we run the following iteration starting from \(T = 0, 1, \ldots\). Each new iteration is equal to

\[
\mu^{(T+1)}_\ell := \frac{1}{|S^{(T)}_\ell|} \sum_{j \in S^{(T)}_\ell} x_j.
\]

The new partition, \(S^{(T+1)}_\ell\), is then created by

\[
S^{(T+1)}_\ell := \{x_i : ||x_i - \mu^{(T+1)}_\ell|| < ||x_i - \mu^{(T+1)}_j|| \forall j \neq \ell\}.
\]

This process is repeated until \(S^{(T)}_\ell\) is stabilized for all \(\ell = 1, \ldots, K\). Once this is done we are left with

\(\overline{M}\) as the normalized matrix.

---

\(^2\)This is done by calculating the standard deviation, \(\delta_i \forall 0 \leq i \leq n\), of every column in the matrix. Then a new matrix, \(\overline{M}_{m \times n} \in \mathbb{R}^{m \times n}\), is created where the value of every cell, \(\overline{m}_{ji}\), in the matrix is equal to \(z_{ji} = \frac{m_{ji} - \mu_i}{\delta_i}\) where \(m_{ji}\) is the value of the corresponding cell from \(M\).

\(^3\)This is called the ‘Forgy’ method.

\(^4\)It has been proven that \(k\)-means converge to a local minimum [5].
the set of partitions \( \{S^{(T+1)}_k\}_{k=1}^K \) and the set of means \( \{\mu^{(T+1)}_k\}_{k=1}^K \).

This can also be explained through the use of pseudocode as seen in Algorithm 1.

**Algorithm 1** The k-means algorithm

Let \( X \) be the dataset being clustered
Let \( U \) be the set of means and set \( U \) equal to \( k \) randomly selected subjects from \( X \)
Let \( S \) be the set of clusters and set it equal to \( k \) empty clusters each of which is associated with one of the means in \( U \)

for \( x \) in \( X \) do
- Place \( x \) in the cluster in \( S \) associated with \( u_* \) where \( d(x, u_*) = \min_{u_i \in U} d(x, u_i) \)
end for

\( S_O = null \)

while \( S \neq S_O \) do
- Make \( U \) equal to the means of the clusters of \( S \)
- Make \( S \) equal to \( k \) empty clusters each of which is associated with one of the means in \( U \)

for \( x \) in \( X \) do
- Place \( x \) in the cluster in \( S \) associated with \( u_* \) where \( d(x, u_*) = \min_{u_i \in U} d(x, u_i) \)
end for

end while

Although k-means is a widely used method that is applicable in many situations, there are datasets where k-mean clustering fails to provide useful information and alternative methods must be used. For example consider the set of points in Figure 2.1.

Figure 2.1: A point cloud where k-means clustering is not very effective

In this example, it is clear that k-means clustering is not very effective and it is better to apply Spectral Clustering which is another very prominent clustering algorithm.

### 2.2 Spectral Clustering

The process of spectral clustering is one of the most common ways to take advantage of the graph theoretical representation of a dataset. This section will begin by introducing the notion that a dataset can also be represented as a graph and as well as points in \( n \)-dimensional space. This will involve brief overview of the relevant parts of graph theory.
Definition 1. An affinity graph is a triple \( G = (V; E; \omega) \), where \( (V; E) \) is a graph and \( \omega : E \rightarrow \mathbb{R}_+ = \{x > 0\} \) is referred to as the affinity function. The graph \( G = (V; E) \) is a pair where \( V \) is the set of vertices of the graph, and \( E \subseteq V \times V \) is the set of edges.

It is important to note that spectral clustering is applied to undirected graphs which means that if \((v_1, v_2) \in E\) then \((v_2, v_1) \in E\) and \(w_{ij} = w_{ji}\) for \(\forall v_1, v_2 \in V\).

To be able to analyze the dataset using graph theoretic approaches we must understand how to construct a graph from a given dataset or point cloud.\(^5\) To do this, we interpret each subject as a point in multidimensional space that acts as a vertex of the graph. The edges of the graph can then be derived in multiple ways, but the most common methods are:

1. **k-nearest neighbor graph** draws edges between a vertex and the \( k \geq 1 \) points nearest to it.
   
   Formally, \((v_i, v_j) \in E\) if and only if the number of vertices \( v_q \) such that \( d(v_i, v_q) < d(v_i, v_j) \) is less than \( k \) or the number of vertices \( v_p \) such that \( d(v_j, v_p) < d(v_i, v_j) \) is less than \( k \).

2. **r-neighborhood graph** draws edges between a vertex and all vertices contained within a sphere of radius \( r > 0 \) centered at the vertex. Formally, \((v_i, v_j) \in E\) if and only if \( d(v_i, v_j) \leq r \).

3. **Fully connected graph** draws edges such that every vertex is connected to all other vertices.
   
   Formally, \((v_i, v_j) \in E\) \(\forall v_i, v_j \in G\).

In general, the construction of a graph from the dataset is an important aspect of massive data analysis especially when the dataset is noisy [29]. Once a graph is derived from the dataset using a construction technique, we can better analyze its structure by using its graph Laplacian. Defining the graph Laplacian requires us to first define the affinity matrix and degree matrix for an affinity graph.

Definition 2. Given an affinity graph \( G = (V; E; \omega) \) where \(|V| = n\), the affinity matrix of \( G \) is defined as the matrix \( W \in \mathbb{R}^{n \times n} \) defined by:

\[
W_{i,j} = \begin{cases} 
\omega(i, j) & : (i, j) \in E \\
0 & : (i, j) \notin E 
\end{cases}
\]

The degree matrix of \( G \) is the matrix \( D \in \mathbb{R}^{n \times n} \) defined by:

\[
D_{i,j} = \begin{cases} 
\sum_{(i,j) \in E} W_{i,j} & : i = j \\
0 & : i \neq j 
\end{cases}
\]

Where \( d(i) \) is the degree function, \( d : V \rightarrow \mathbb{R}_+ \), of \( G \) defined by \( d(i) = \sum_{(i,j) \in E} W_{i,j} \).

Now we can define the transition matrix, \( A \), of an affinity graph \( G \) to be the inverse of the degree matrix times the affinity matrix meaning \( A = D^{-1}W \). It is important to note that the matrix \( D \) is invertible if and only if \( d(i) \neq 0 \forall i \).

Observe that the matrix \( A \in \mathbb{R}^{n \times n} \) is not symmetric, but it is similar to \( D^{-1/2}WD^{1/2} \), which is symmetric. By the spectral theorem, we know that there is some \( V \in O(n) \) and \( \Lambda = \text{diag}(\lambda_1, ..., \lambda_n) \) such that \( D^{-1/2}WD^{1/2} = VAV^T \). We assume that the \( \lambda_i \)'s are ordered so that \( |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \).

If we define \( \Phi = D^{-1/2}V \) and \( \Psi = D^{-1/2}V \) then we see that \( A = D^{-1}W = \Phi \Lambda \Psi^T \). Now we are able to define the Graph Laplacian as it applies to affinity graphs.

---

\(^5\)In our case the dataset must first be converted into a graph, but the spectral clustering algorithm can also be used to analyze graphs that already exist.
**Definition 3.** The *Normalized Graph Laplacian* of an affinity graph $G$ is defined as $L = I_n - A = I_n - D^{-1}W$, where $I_n$ is the $n \times n$ identity matrix. The *Unnormalized Graph Laplacian* is defined as $\tilde{L} = D - W$. The relationship between the two is simply $\tilde{L} = DL$.

The spectral clustering process begins by creating the affinity graph, $G$, from the data using a method like those specified above. Then we derive the normalized graph Laplacian, $L$, of the affinity graph and calculate the eigenvectors and eigenvalues of $L$.

We create a new matrix $\mathcal{L}$ where the columns of $\mathcal{L}$ are the eigenvectors of $L$ with the eigenvectors ordered according to the magnitude of their corresponding eigenvalue. The eigenvector corresponding to the smallest eigenvalue appears at the far right of the matrix and the one corresponding to the largest appears at the far left. The spectral clustering process is now described in Algorithm 2.

**Algorithm 2** The spectral clustering algorithm

**Require:** Dataset $\mathcal{X} = \{x_i\}_{i=1}^m \subset \mathbb{R}^n$, $K > 1$.

[Step 0] Construct a directed affinity graph from the point cloud $\mathcal{X}$, and establish the normalized GL $L$. Order the eigenvalues of $L$ from small to large.

[Step 1] Find the first $K$ right eigenvectors $\lambda_1, \ldots, \lambda_K$ of $L$;

[Step 2] Embed $i \in V$ into $\mathbb{R}^K$ by $\Phi : i \mapsto (\lambda_1(i), \ldots, \lambda_K(i))$;

[Step 3] Run $K$-mean or any other clustering algorithms on $\{\Phi(i)\}_{i=1}^m$ with $K$ clusters.

Output $\{S_i\}_{i=1}^K$ as the $K$ clusters.

It is important to realize that spectral embedding is closely related to diffusion maps [15]. Recall that the diffusion map of an affinity graph $G$ with diffusion time $t \geq 0$ can be defined as $\Phi_t^m : V \rightarrow \mathbb{R}^{m-1}$, $\Phi_t^m(i) = (\lambda_2^t \phi_2(i), \lambda_3^t \phi_3(i), \ldots, \lambda_m^t \phi_m(i))^T$. Where $\Phi$ is the matrix from $A = \Phi \Delta \Psi^T$ as defined above. Note that each $\phi_q$ is the $q$-th column of $\Phi$ and $\phi_1$ is excluded in diffusion maps because it is always equal to 1, but it is included in spectral clustering. In spectral clustering, the embedding can be thought of as similar to the diffusion map with $t = 0$ which is understood as an eigenmap [3]. In practice, we could consider a variation of the embedding before running the $k$-mean clustering by taking diffusion into account like diffusion maps with $t > 0$.

There is a fruitful theoretical understanding of spectral clustering and we will now mention some key aspects of it. Understanding spectral clustering requires us to be able to measure the properties of the separation of a graph into subgraphs. This is done is through the notion of cut and volume. Cut is the measure of how strong the links are between two sets of vertices. It can also be thought of as how much ‘energy’ it would take to separate, or cut, all the edges connecting the two sets.

**Definition 4.** If $A \subset V$ then the Cut is defined by

$$\text{Cut}(A) = \sum_{i \in A, j \in A^C} W_{i,j}$$

where $A^C$ is the compliment of $A$ and $W$ is the affinity matrix of $G$.

However, this definition means that cut would be biased to favor smaller subsets so we need another measure to correct this. For that purpose we use the volume measurement.

**Definition 5.** If $A \subset V$ is a subset of vertices then the volume is defined by

$$\text{Vol}(A) = \sum_{i \in A} D_{i,i}$$
where $W$ is the affinity matrix of $G$ and $n$ is the total number of vertices in $G$.

Using these two definitions we can now define a measure that determines the effectiveness of a division of a graph into subgraphs.

**Definition 6.** For a non-null subset $A \subset V$, so that $A^C \neq \emptyset$, the **Normalized cut (Ncut)** is defined as

$$Ncut(A) = \text{Cut}(A) \left( \frac{1}{\text{Vol}(A)} + \frac{1}{\text{Vol}(A^C)} \right)$$

Minimizing $Ncut$ can be approximated by computing the second smallest eigenvector of the matrix $I - D^{-1/2}WD^{1/2}$ [41]. The relationship between the graph Laplacian and Ncut is related by Cheeger’s inequality [13]. This theory also holds when $k = 2$. For cases where $k > 2$, we refer the reader to [30].

### 2.3 Hierarchical Clustering

The goal of hierarchical clustering, as the name suggests, is to build a hierarchy of clusters between the cluster containing all subjects and a set of clusters where each cluster is a singleton only containing one subject. The goal of hierarchical clustering is similar to that of $k$-mean clustering, but the aim is to find a “natural hierarchical structure”. Also, in $k$-mean clustering you have to choose the desired number of clusters, but in hierarchical clustering you choose how to measure the distance between clusters. For some theoretical studies of hierarchical clustering see [11] and [17].

**Definition 7.** **Trees** are graphs that are defined as graphs without self loops where any two vertices are connected by exactly one path and the edges are undirected.

The results of hierarchical clustering are usually represented by a dendrogram, which is a special type of tree, where the cluster containing all subjects acts as the ‘top’ node of the tree and clusters only containing a single subject act as the ‘leaves’ of the tree. This hierarchy can be built by either splitting the cluster containing all subjects and working your way down, this is referred to as the *divisive* method, or by combining the clusters containing single subjects and working your way up which is referred to as the *agglomerative* method. In this section only the agglomerative process is explained.

The goal of the hierarchical clustering process is to create a tree structure where each node of the tree represents a subset of subjects. This is done by using the concept of *linkage* to combine clusters based on the “distance”, $D(X,Y)$, between any two clusters $X,Y$.

**Linkage** refers to linking or combining two clusters together based on which two clusters have the smallest ‘distance’ between them. There are different types of linkage and they differ based on how the distance $D(X,Y)$ is calculated. Some of the most common types of linkage are:

1. **Single-linkage (Minimum) clustering** is based on the distance function between two clusters $X,Y$ being defined as

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

Another measure known as **Ratio cut (Rcut)** can also be used. For a non-null subset $A \subset V$, so that $A^C \neq \emptyset$, the Ratio cut is defined as

$$Rcut(A) = \text{Cut}(A) \times \left( \frac{1}{|A|} + \frac{1}{|A^C|} \right).$$
2. **Complete-linkage (Maximum) clustering** is based on the distance function between two clusters $X, Y$ being defined as

$$D(X, Y) = \max_{x \in X, y \in Y} d(x, y)$$

3. **Centroid linkage clustering** is based on the distance function between two clusters $X, Y$ being defined as

$$D(X, Y) = ||c_X - c_Y||$$

where $c_X$ and $c_Y$ are the centroids of $X$ and $Y$ respectively defined as $c_S = \frac{s_1, s_2, ..., s_{|S|}}{|S|}$ for a cluster $S$ where $|S|$ is the number of subjects contained in $S$.

4. **Average linkage clustering** is based on the distance function between two clusters $X, Y$ being defined as

$$D(X, Y) = \frac{1}{|X| \cdot |Y|} \sum_{x \in X} \sum_{y \in Y} d(x, y)$$

where $|X|$ and $|Y|$ are the number of subjects in $X$ and $Y$ respectively.

This is a non-exhaustive list of linkages and there are other more general distance measurements between sets that can be considered. Note that hierarchical clustering in this thesis calculates the distance, $d(x, y)$, between subjects $x$ and $y$ using the Euclidean metric. However, in different applications any number of other metrics such as Manhattan distance or maximum distance may be used.

In summary, all of these four types of linkage allow you to quantify the dissimilarity between clusters based on different geometric features. In single-linkage, clusters are combined based on the distance of their closest points to each other. In complete-linkage, clusters are combined based on the distance of their furthest points from each other. In centroid linkage, clusters are combined based on which clusters have the smallest distance between their centers. Finally, in average linkage, clusters are combined based which clusters have the smallest average distance between all of their points.

Linkage can now be used to construct a hierarchical tree by first making each subject in the dataset its own cluster. This gives us a set of clusters, $N$, where each cluster only contains a single subject. Each of these clusters is then represented as a node on a tree $T$ which, at this point, is completely disconnected. Now we calculate the distance $D(X, Y)$ between all clusters $X, Y \in N$. We then find $X', Y' \in N$ such that $D(X', Y') = \min_{X, Y \in N} (D(X, Y))$.

We then create a new cluster $C_1 = X' \cup Y'$ and represent it as a new node on $T$ that is only connected to the nodes representing $X'$ and $Y'$. Now we create an updated set of clusters $N_1 = N - X' - Y' + C_1$. We then repeat the process by finding the clusters with the minimum distance in $N_1$ and combining them to form a new cluster $C_2$. The cluster $C_2$ is then represented as a new node on $T$ that is only connected to the nodes representing the clusters used to form it. Once this is done, another updated set of clusters $N_2$ is created by taking $N_1$ and removing the two clusters that were combined to form $C_2$ and adding the cluster $C_2$. This process is repeated until there is a set of clusters $N_\ell$ created that only contains one cluster $C_\ell$ which is a cluster consisting of all subjects in the dataset. Cluster $C_\ell$ is then represented as the ‘top’ node on tree $T$. See Algorithm 3 for a pseudocode illustration of hierarchical clustering.

The result of this process is the creation of a hierarchical dendogram which allows a researcher to see which subjects are the most similar to each other and which subjects are the most different from all
others in the data set. The nodes closer to the leaves of the tree show the clusters with points that are
the most similar while the clusters closest to the top of the tree show clusters where subjects only have
small similarity on average.

Algorithm 3 Creating a hierarchical clustering tree

Create an array of $N$ clusters where each cluster is initially equal to one subject in the dataset
Create a tree, $T$, that initially contains a set of disconnected nodes where each node is a cluster in $N$
while $1 < |N|$ do
    $\text{temparray} = \text{null}$
    for $w$ in $\{1, \ldots, |N|\}$ do
        for $s$ in $\{1, \ldots, |N|\}$ do
            if $w \neq s$ then
                $\text{temparray} += D(n_w, n_s)$
            end if
        end for
    end for
    Find $w^*, s^*$ such that $\text{argmin}(\text{temparray}) = D(n_{w^*}, n_{s^*})$
    Remove $n_{w^*}$ and $n_{s^*}$ from $N$
    Add a cluster that combines $n_{w^*}$ and $n_{s^*}$ to $N$
    Renumber all clusters in $N$
    Add a node representing the cluster that combines $n_{w^*}$ and $n_{s^*}$ to the tree $T$
    Connect the new node to the nodes on the tree that represent $n_{w^*}$ and $n_{s^*}$
end while

2.4 Support Vector Machine (SVM)

Support Vector Machine is a technique in supervised learning that provides a method for binary classification. Binary in this context means that it predicts which one of two classes a subject will belong to [4]. Techniques to classify subjects between more than two classes also exist and are known as Multiclass SVM [19] [25]. To simplify our discussion, we only explain binary classification in this section.

The goal of the SVM process is to represent the subjects as vectors in $n$ dimensional space (where all vectors begin at the origin) and derive an $n-1$ dimensional hyperplane that accurately divides the subjects between two classes. This technique allows us to predict which class a subject belongs to by taking the vector representation of the subject and calculating its projection on the normal vector, $\vec{w}$, of the hyperplane.

To demonstrate how SVM works, the following explanation assumes that the two classes are linearly separable. If there are two classes $A, B$ that a subject $x_i$ can belong to then we can predict which class it belongs to using the following equations:

\[
\vec{w} \cdot \vec{x}_i + b \geq 1 \text{ for } x_i \in A
\]
\[
\vec{w} \cdot \vec{x}_i + b \leq -1 \text{ for } x_i \in B
\]

The purpose of the SVM process is to find a hyperplane and associated $\vec{w}$ that accurately divides the subjects between $A$ and $B$. As there are likely multiple hyperplanes that accomplish this, the SVM process seeks to find the hyperplane that has the greatest distance between itself and the hyperplanes that act as boundaries for the subjects belonging to the two classes. These two boundary hyperplanes are referred to as margins. The vector $\vec{w}$ can be calculated by finding set of vectors from each class
that are on the margins of the hyperplane. These margin vectors are the “support vectors” that SVM is named after. The process of calculating $\vec{w}$ is as follows:

The two previous equations can be rewritten as one equation in the form

$$y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1$$

through the addition the condition $y_i$ which is defined as

$$y_i = \begin{cases} 1 & x_i \in A \\ -1 & x_i \in B \end{cases}$$

From this it follows that $y_i(\vec{w} \cdot \vec{x}_i + b) - 1 \geq 0 \ \forall i$ and $y_i(\vec{w} \cdot \vec{x}_i + b) - 1 = 0$ when $x_i$ is on one of the margins.

We can also see that the width of the margins is equal to $(x_a - x_b) \cdot \frac{\vec{w}}{\|\vec{w}\|} = (\vec{w} \cdot x_a - \vec{w} \cdot x_b) \cdot \frac{1}{\|\vec{w}\|}$ substituting in $y_i(\vec{w} \cdot \vec{x}_i + b) - 1 = 0 \Rightarrow \vec{w} \cdot \vec{x}_i = y_i - b$ (since $x_a$, $x_b$ are on the margins) we arrive at $((1 - b) - (-1 - b)) \cdot \frac{1}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|}$. Therefore, this problem is equivalent to maximizing $\frac{2}{\|\vec{w}\|}$ which is equivalent to minimizing $\frac{1}{2}\|\vec{w}\|^2$ and it can be solved using Lagrange multipliers by writing it as

$$L(\vec{w}, b, a) = \frac{1}{2}\|\vec{w}\|^2 - \sum_i a_i(y_i(\vec{w} \cdot \vec{x}_i + b) - 1).$$

Then by taking the partial derivatives with respect to $\vec{w}$ and $b$ we arrive at

$$\frac{\partial L}{\partial \vec{w}} = \vec{w} - \sum_i a_i y_i \vec{x}_i \Rightarrow \vec{w} = \sum_i a_i y_i \vec{x}_i \quad \text{and} \quad \frac{\partial L}{\partial b} = -\sum_i a_i y_i \Rightarrow \sum_i a_i y_i = 0.$$

Which means $L$ can be reformulated as

$$L = \sum_i a_i - \sum_{i,j} a_i a_j y_i y_j \vec{x}_i \cdot \vec{x}_j \quad \text{subject to} \quad \sum_i a_i y_i = 0 \quad \text{and} \quad a_i \geq 0.$$

This resulting equation can then be solved using techniques from numerical analysis and it will provide us with our $\vec{w}$ vector.

If the dataset is not linearly separable, then a similar approach can still be used, but the data must first be mapped to a higher dimensional space where it becomes linearly separable. This task is accomplished using the kernel method which is the process of mapping points to a higher dimensional space through the use of kernel functions [24].

### 2.5 CART, Random Forest, and Other Ensemble Methods

This section will explain CART, Random Forest, and other relevant ensemble methods of supervised learning such as those presented in [8].

#### 2.5.1 Classification And Regression Trees (CART)

Classification And Regression Trees (CART) is a method outlined by Breiman et al. in [9] to create predictive decision trees based on a dataset. While CART is the method that is used in this thesis, there are various other prominent methods such as ID3 and C4.5 that can be used to create decision trees from a dataset [24]. To understand CART, the concept of decision trees must first be explained.

A decision tree is a tree graph where each node represents a set of subjects and each edge represents a split of those subjects into two different nodes based on a certain property. The goal of decision tree
**learning** is to create a decision tree that can be used to predict the class that a subject will belong to based on the subject’s parameters. This concept is made more clear by example.

**Example 1.** Consider the following five subjects \( \{t_1, t_2, t_3, t_4, t_5\} \) each of which have three properties \( \{a, b, c\} \) as shown in Table 2.1. This data can be used to construct the decision tree shown in Figure 2.2.

Table 2.1: Example Dataset

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_1</td>
<td>0.1</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>t_2</td>
<td>17</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>t_3</td>
<td>-5</td>
<td>0</td>
<td>(\sqrt{2})</td>
</tr>
<tr>
<td>t_4</td>
<td>(\frac{7}{4})</td>
<td>'c'</td>
<td>-60</td>
</tr>
<tr>
<td>t_5</td>
<td>1000</td>
<td>'dog'</td>
<td>-7.7</td>
</tr>
</tbody>
</table>

Figure 2.2: An example of a decision tree that can be created using the data in Table 2.1

In this tree we can see that nodes contain the following subjects: \( N_1 = \{t_1, t_2, t_3, t_4, t_5\} \), \( N_2 = \{t_1, t_3, t_4\} \), \( N_3 = \{t_2, t_5\} \), \( N_4 = \{t_3\} \), \( N_5 = \{t_1, t_4\} \), \( N_6 = \{t_2\} \), \( N_7 = \{t_5\} \).

To create a decision tree CART begins by taking a single node that contains all subjects in the dataset. This node is then split into two nodes with subjects allocated between them based on some discrimination criteria. The discrimination criteria that is used to split the node is chosen based on whether it results in the ‘best’ split. There are multiple ways to measure the effectiveness of a split the most prominent of which are **information gain** and **Gini impurity**. The CART methodology uses Gini impurity.

**Definition 8.** **Gini impurity** is a measure of the effectiveness of a split. If there are \( p \) classes and \( f_i \) is the fraction of subjects in a node that belong to class \( i \) then the Gini impurity is equal to \( 1 - \sum_{i=1}^{p} f_i^2 \).

Therefore the node is split by attempting to separate the subjects using every possible range within all parameters and then selecting the discrimination criteria that results in the lowest Gini impurity. Once this split is found the resulting two nodes are again split according to the same methodology unless a node fulfills one of the following stopping conditions:

1. All subjects in the node belong to the same class.
2. No further split can be made among the subjects (i.e. the subjects in the node have identical or nearly identical properties, but belong to different classes).

3. (Optional) A custom stopping condition is met. For example stopping after a certain percentage of subjects (ex. 95%) in a node belong to the same class.

The CART process continues until every unsplit node has met one of the stopping conditions at which point the construction of the decision tree is complete. This construction process is illustrated through pseudocode in Algorithm 2.

The constructed tree can now be used to predict the classification of a new subject $g_1$ whose class is unknown. This is done by placing $g_1$ at the top node of the tree and then determining which leaf node it would belong to by making the decisions based on its parameter values. Once $g_1$ is allocated to a leaf node, we can determine a set of probabilities based on the class composition of the subjects from the training data that were allocated to that leaf node during the trees construction. If we suppose that there are $Q = \{q_1, \ldots, q_m\}$ possible classes that a subject $g_1$ can belong to, then the leaf node gives us a set of probabilities $\{P(g_1 \in q_1), \ldots, P(g_1 \in q_m)\}$ where $P(g_1 \in q_i)$ is equal to the fraction of subjects in the leaf that belong to class $q_i$. In other words, if $L$ is the leaf node and $\{l_1, \ldots, l_{|L|}\}$ are the subjects belonging to it from the training data then

$$P(g_1 \in q_i) = \frac{|\{l_p \in q_i\}|}{|L|}$$

where $|\{l_p \in q_i\}|$ is the number of subjects in the node belonging class $q_i$.

**Algorithm 4 Creating a classification tree using CART**

```plaintext
while There exists some unsplit node on the tree that has not fulfilled a stopping condition do
    Select an unsplit node, $n^*$, on the tree that hasn’t fulfilled the stopping condition
    Find the best split condition for $n^*$
    Using the best split as the decision and create two new nodes on the tree with an edge from each of them to $n^*$
    Allocate the subjects of $n^*$ to the two nodes based on the best split condition
end while
```

### 2.5.2 Random Forest

The premise of random forest is that it uses the results of multiple decision trees (a ‘forest’ of trees) to predict how a subject will be classified. To create a forest of nonidentical trees random forest uses a concept called *bootstrap aggregating* (*bagging*) where subsets of the data are used to create weak classifiers which are later combined into a strong classifier. Bagging is used by random forest to create a set of decision trees where each tree is derived from a different subset that contains $s$ subjects where the $s$ subjects are selected randomly with replacement from the set of all subjects [6]. In contrast to bagging, the process of *boosting* tries to reduce bias by using more stringent criteria (such as stopping the decision tree construction process after 3 levels) to create the set of weak learners [7] [21].

Bagging subjects is a common practice in many techniques in decision tree learning, but random forest differentiates itself by also applying a similar principle to the set of parameters that a tree is allowed to consider. For the purpose of reducing the correlation between trees, the random forest methodology creates each tree in the forest using a different subset of $s$ subjects as well as a different subset of $m$
parameters\textsuperscript{7} from the set of all parameters. While the sets of subjects and parameters used to create each tree are different, the standard approach is to keep the size of these sets constant across all trees meaning that numbers $s$ and $m$ never change throughout the entire forest.

Before a forest can be created a number of trees, $n$, for the forest must be decided as well as the values of $s$ and $m$. These values are dependent on the nature of the analysis being done and considerations such as available computational resources. Once these values are chosen, the forest is created by randomly selecting $s$ subjects and $m$ parameters and then using them to create a decision tree with the CART methodology and doing this $n$ times. This process creates the forest $F = \{t_1, \ldots, t_n\}$ where each $t_j$ represents decision tree.

After the forest is created, it can be used to predict what class a subject, $g_1$, belongs to. This is done by running $g_1$ through every decision tree in the forest. Each decision tree, denoted $t_j$, outputs the set of probabilities $\{P_{t_j}(g_1 \in q_1), \ldots, P_{t_j}(g_1 \in q_m)\}$ as explained in Section 2.4.1. Based on these probabilities every tree ‘votes’ for the class that it believes the subject has the highest chance of belonging to.

In our example with $Q$ classes and $n$ trees, the number of votes that a subject, $g_1$, belongs to class $q_i \in Q$ is equal to

$$V(g_1 \in q_i) = \sum_{j=1}^{n} v_{t_j}(g_1 \in q_i) \quad \text{where} \quad v_{t_j}(g_1 \in q_i) = \begin{cases} 1 & \text{if } P_{t_j}(g_1 \in q_i) = \max_{q_l \in Q} P_{t_j}(g_1 \in q_l) \\ 0 & \text{otherwise} \end{cases}.$$ 

The class that receives the highest number of votes from the forest is the predicted classification. It should be noted that this ‘vote’ method is the original method proposed in [8]. However, there is an alternative method to predict classification that we have chosen to implement in our analysis.

This alternative method works by averaging the probability that $g_1 \in q_i$ from all trees in the forest and it can be written as

$$A(g_1 \in q_i) = \frac{1}{n} \sum_{j=1}^{n} P_{t_j}(g_1 \in q_i).$$

The class with the highest average probability is the predicted classification for the subject.

\textsuperscript{7}Usually $m = \sqrt{\text{number of total parameters}}$. 
Chapter 3

A Questionnaire Algorithm Using Dual Geometry

Using the tools and methods described in Chapter 2, this chapter will discuss a novel approach to database analysis proposed by Professor Ronald Coifman in [14]. This analysis creates two tree structures from the dataset, one for the sensors and one for the subjects, that can be used to create a new metric to measure the distance/similarity between subjects. The intuition behind this approach is that a metric created using information from the dataset may be able to take more hidden structures into account than other metrics. The final result of the analysis is the generation of two refined tree structures that show which subsets of subjects are similar as well as which subsets of sensors are similar.

3.1 Creating the Initial Tree Structure

We apply the tree construction techniques to the preprocessed normalized matrix, $\overline{M}$, by first selecting a “measurement” that can be used to build up the tree. We have chosen to use the $k$-means algorithm to create a tree structure for the sleep apnea data. We start by creating a tree structure for the sensors, $\{s_1, \ldots, s_n\} \subset \mathbb{R}^m$, of the dataset.

Denote the constructed tree $T$ and it will be built agglomeratively. Each column of $\overline{M}$ is taken to be an $m$-dimensional point in euclidean space. Let $\{s_1, \ldots, s_n\}$ such that $s_i \in \mathbb{R}^m$ for $1 \leq i \leq n$ be the set of all columns in $\overline{M}$ which represent the sensors while the rows represent subjects. Our goal is to create a set of levels, denoted $\{T^0, \ldots, T^L\}$, of the tree where each level $0 \leq \ell \leq L$ is composed of $|T^\ell|$ mutually disjoint sets which are associated with the nodes of the tree. At this point it is important to note that in our notation we use $|T^\ell|$ to refer to the number of nodes on level $\ell$ of the tree.

The construction of a tree consisting of $L$ levels will begin as $T^0$, where there are $|T^0| = n$ disconnected nodes which is referred to as level 0. Denote $T^0_k$ as a singleton such that $T^0_k = \{s_k\} \subset T^0$ for all $0 \leq k \leq n$. Therefore, the nodes $T^0_k$ comprise the leaves of the tree. The $\ell$-th level of the tree is denoted as $T^\ell$. The $k$-th node of $T^\ell$ is denoted $T^\ell_k$ which is a union of the nodes on the level below it while $|T^\ell_k|$ refers to the

---

1 Preprocessed using the methods mentioned in Section 2.0.1.
number of nodes on level \((\ell - 1)\) that are contained in node \(T^\ell_k\). Note that,

\[
\{s_1, \ldots, s_n\} = \bigcup_{k=1}^{\lceil T^\ell \rceil} T^\ell_k \quad \forall \ 0 \leq \ell \leq L.
\]

Also, the ‘top’ level of the tree is level \(L\) and it consists of only one node \(T^1_1\) which contains all of the columns in \(M\) meaning that \(T^1_1 = \{s_1, \ldots, s_n\}\).

To construct level \(\ell\) of the tree (where \(0 < \ell < L\)) we cluster the nodes on level \((\ell - 1)\) of the tree into \(\lceil \frac{|T^{\ell-1}|}{5} \rceil\) clusters\(^2\) using the \(k\)-means algorithm with the Euclidean metric on points in \(m\)-dimensional space. To cluster these nodes we must be able to represent every node on the tree as a point in \(m\)-dimensional space. We will accomplish this by defining

\[
val(T^\ell_k) = \frac{1}{|T^\ell_k|} \sum_{T^{\ell-1}_j \subset T^\ell_k} val(T^{\ell-1}_j) \quad \forall \ \ell > 0.
\]

In other words, \(val(T^\ell_k)\) of node \(k\) on level \(\ell\) of the tree is equal to the average of \(val(T^{\ell-1}_j)\) on nodes, \(T^{\ell-1}_j\), on level \((\ell - 1)\) that are contained in \(T^\ell_k\). For nodes on level 0 of the tree we simply define \(val(T^0_k) = s_k\) as they are singletons that are already points in \(m\)-dimensional space.

In summary, we already have the nodes on level 0 as they are simply singletons of the set \(\{s_1, \ldots, s_n\}\). To construct level 1 of the tree we run \(k\)-means clustering on the set consisting of \(val(T^0_k) = s_k \ \forall \ 1 \leq k \leq |T^0| = n\) to attain \(\lceil \frac{|T^{0-1}|}{5} \rceil = \lceil \frac{|T^{1-1}|}{5} \rceil = \lceil \frac{n}{5} \rceil\) clusters. A new node on level 1 is then created for each cluster and this node contains all of the level 0 nodes in that cluster. This means that we now have constructed level 1 of the tree and it consists of \(\lceil \frac{n}{5} \rceil\) nodes. Now we can construct level 2 of the tree by running \(k\)-means clustering on the set consisting of

\[
val(T^1_k) = \sum_{T^0_y \in T^1_k} \frac{1}{|T^1_k|} val(T^0_y) \quad \forall \ 1 \leq |T^1| = \lceil \frac{n}{5} \rceil.
\]

A new node on level 2 is created for each of these clusters and this node contains all of the level 1 nodes in that cluster. The same process is used to construct level 3, level 4, and so on. The process continues until a level \(\ell\) is constructed such that \(|T^{\ell}| \leq 7\) at which point all the nodes on level \(\ell\) are combined into one node \(T^\ell\) which is the top node of the tree.\(^3\) This completes the construction of \(T\).

A similar process can be used to construct a tree structure on the subjects.

### 3.2 Deriving the Dual Geometric Metric

The constructed sensor tree can now be used to define a *dual geometry metric*\(^4\), \(\mathcal{D} : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}\), in order to measure the distance between two subjects \(a_i, a_j\) in the dataset. The utilization of this tree structure in the creation of the metric is what allows the dual geometry metric to be adaptive to the dataset. In theory, this should provide more accurate results than just using a universal metric, such

---

\(^2\)The denominator of 5 is specific to this dataset and different datasets might be better analyzed with a different proportion of clusters relative to the subjects.

\(^3\)The number 7 is specific to this dataset and different datasets might be better analyzed with a different minimum number of nodes on the level below the top node.

\(^4\)The definition of the dual geometry metric (know as the “dual affinity metric” in [14]) that follows is specific to its implementation within our analysis. A more formal and broad definition of dual affinity is presented as Definition 1 in [14].
as the Euclidean metric, because it captures more structures hidden in the data. It is called the dual geometry metric because uses a tree structure derived from the geometry of the sensors to create a measure for the geometry of the subjects or vice versa. The method used to define the dual geometry metric on subjects using the structure of the sensor tree will now be explained, but before this can be done we must first explain how the tree structure, \( T \), can be applied to a single subject in the dataset.

In the previous section we constructed a tree \( T \) using the set of sensors, \( \{s_1, \ldots, s_n\} \), as the leaves of the tree and defined \( \text{val}(T^\ell_k) \) for all nodes on the tree. For the \( i \)-th subject, we have a vector \( a_i = (a_{i1}, \ldots, a_{in}) \in \mathbb{R}^n \) representing its sensored status. To define the dual geometry for each subject, we create a new tree \( A_i \) that has the same structure as \( T \). We define a function, \( F : \bigcup_{\ell=1}^L T^\ell \rightarrow \mathbb{R} \), on the tree \( A_i \) in the following way:

\[
F^0_{i,k} = a_k \quad \text{and} \quad F^\ell_{i,k} = \frac{1}{|T^\ell_k|} \sum_{q \text{ s.t. } T^\ell_{q-1} \subset T^\ell_k} F^{\ell-1}_{i,q} \quad \text{for } \forall \ell > 0.
\]

Note that we are not constructing another tree structure or doing any clustering. Based on this understanding, we are now able to define the dual geometry metric between subjects \( a_i \) and \( a_j \) using the trees \( A_i \) and \( A_j \).

**Definition 9.** The **dual geometry metric** is the calculation of distance between two subjects \( a_i, a_j \) defined as

\[
D(a_i, a_j) = \sum_{\ell,k} |F^\ell_{i,k} - F^{\ell-1}_{j,k}|
\]

In other words, the metric calculates the distance between \( a_i \) and \( a_j \) by summing the difference between all nodes in \( A_i \) and \( A_j \) that are in the same position on the tree. We can also use similar ideas to define the **dual geometry affinity** between the subjects \( a_i \) and \( a_j \).

**Definition 10.** To define the **dual geometry affinity**, take a kernel \( K : [0, \infty) \rightarrow \mathbb{R} \) decaying fast enough. Then the affinity between two subjects \( a_i, a_j \) is defined as

\[
K\left( \frac{D(a_i, a_j)}{\sqrt{\epsilon}} \right)
\]

where \( \epsilon > 0 \) is chosen by the researcher.

Now the dual geometry metric derived from the sensor tree can be used to construct a tree structure for the subjects.

### 3.3 Creating a Second Tree Structure Using Dual Affinity

A tree structure must now be created to show the relationships between the subjects to define a dual geometry metric on the sensors. The process used to construct this tree is identical to the method discussed in Section 3.1, except for two important differences, which are the following:

1. The set of subjects \( \{x_1, \ldots, x_m\} \) such that \( x_i \in \mathbb{R}^n \), represented by the rows of \( \overline{M} \), is used as the leaves of the tree instead of \( \{s_1, \ldots, s_n\} \).
2. The $k$-means algorithm that is run to cluster the nodes on the tree uses the dual geometry metric defined in Section 3.2 to measure distance instead of the Euclidean metric.

Once this new tree structure is created we have two tree structures. One which shows the relationships between the sensors and the other which shows the relationships between the subjects. Now the constructed tree structures can be refined and improved by iterating the tree construction process.

### 3.4 Iterating the Process to Refine the Tree Structures

Refining the results by iteration is accomplished by using the most recent dual affinity metric that is derived from the most recent tree structure. If it is the sensor tree that is being refined then the most recent tree structure for the subjects, along with the dual affinity derived from it, is used to create a new tree structure from the sensors. The idea is that the most recent dual affinity will ideally contain more accurate information than the previous dual affinities or the Euclidean metric. This more accurate metric will result in a more accurate $k$-mean clustering and a more accurate tree structure.

As there currently is no proof for the convergence of this iterative process, selecting the number of iterations run on the dataset is up to the researcher.

### 3.5 Wavelet Transforms and Other Representations (Optional)

At this stage in the analysis, the resulting tree structures can be treated as final results and be examined to attain information about the dataset. However, it is also possible to use the final tree structures for both the sensors and subjects to do further analysis on the dataset. One of the main ways to do this is to use the resulting hierarchical structure to represent the data using multiscale wavelets as explained in [22]. There are also other methods that can be used to attain a different representation of the data. Once such a representation is applied, there can be any number of techniques used to analyze it including running the questionnaire algorithm again. Detailed explanations of such methods are outside of the scope of this thesis, but the application of such methods is a significant component of the overall approach to database analysis presented in [14].

### 3.6 Visualizing the Results

Now that we have derived the relevant tree structures, we must visualize the results so that we are able to understand them. In order to accomplish this, the matrix representing the dataset will be rearranged and then visualized as a heat map. A heat map of a matrix $M$ is a graphical representation of the matrix where each cell is represented by a color corresponding to the numerical value of the cell. A common convention for this coloring is that higher numbers are represented by the red spectrum and lower numbers are represented by the blue spectrum.

Rearranging the data refers to changing the indexes of the values in the matrix. First the sensors are arranged by placing the groups of columns contained in nodes on level 1 of the sensor tree in order of magnitude where the magnitude of node $T_k^1$ is equal to

$$\|\text{val}(T_k^1)\| := \| \sum_{j \in T_k^1} s_j \|.$$
This will be done in accordance with the following convention: the cluster associated with the highest magnitude is placed at the far right of the matrix. Then the columns (level 0 nodes) within each cluster are arranged from left to right according their magnitude using the same convention as before, but they are only compared to other columns in the same cluster and are only free to move within the portion of matrix taken up by their cluster. Once this is done, the subjects represented by the rows of the matrix are also rearranged in an analogous way.\(^5\)

After the matrix is fully rearranged, a heat map is applied to it and the resulting image is displayed alongside visual representations of the relevant tree structures for the sensors and subjects.

\(^5\)Note this entire process can also be done by beginning the rearranging process at Level \((n - 1)\) of the tree and then placing the node with the highest magnitude on that level at the far right and then arranging the clusters on level \((n - 2)\) within the \((n - 1)\) clusters and then arranging the level \((n - 3)\) clusters within the \((n - 2)\) clusters and so on.
Chapter 4

Results of the Sleep Apnea Questionnaire Analysis

This chapter will present the results generated by analyzing the sleep apnea dataset using the questionnaire algorithm as well as the methods discussed in the Chapter 2.

Sleep apnea is a common disorder with a prevalence rate of approximately 14% in adult men and 5% in adult women [35]. Evidence shows that sleep apnea is associated with several different diseases, ranging from hypertension [34] to heart disease [23] to stroke [1] and it is responsible for several public disasters [16] meaning that it is clearly an important public health issue [36]. The current gold standard to diagnose sleep apnea is reading the multichannel signals recorded through polysomnography (PSG) [26]. However, it is a labor intensive process that requires a special environment as well as special equipment, which limits its use for the purpose of screening. Although sleep apnea has raised a great amount of attention in the past decades, a convenient screening tool for sleep apnea is, unfortunately, still lacking.

To be able to interpret the results of the analysis we must first understand how to measure the severity of a patient’s sleep apnea. In this study the measure that will be used is the Respiratory Disturbance Index (RDI) of a patient.

**Definition 11.** The *Respiratory Disturbance Index (RDI)* is a measure of respiratory events during sleep that can be used to diagnose the severity of sleep apnea. It calculated according to the formula:

\[
RDI = \frac{Apneas + Hypopneas + RERAs}{TST}
\]

Where the variables are defined as:

1. Apneas is the number of suspensions of breathing, which is identified when the airflow breathing amplitude reduces more than 90% for a period ranging from 10 to 120 seconds [26]

2. Hypopneas is the number of instances of abnormally shallow or slow breathing, which is identified when the airflow breathing amplitude decreases over 30% of the baseline with 4% or greater oxygen desaturation for a period ranging from 10 to 120 seconds, but does not meet the criteria for apnea.

3. Respiratory-Effort Related Arousals (RERAs) is the number of changes in breathing pattern that disrupt sleep, but do not meet the definition of hypopnea or apnea.

4. TST is the total sleep time measured in hours.
The RDI measure allows patients to be classified into three groups defined as mild ($RDI < 15$), moderate ($15 \leq RDI \leq 30$) or severe ($RDI > 30$) degrees of sleep apnea.

RDI is different from the AHI (Apnea-Hypopnea Index) due to the addition of RERA events. Indeed, we have

$$AHI = \frac{\text{Apneas} + \text{Hypopneas}}{TST}.$$ 

There have already been multiple studies done that compare the correlation of parameters such as neck size or Epworth Sleepiness Scale (ESS) with sleep apnea diagnoses (see [20], [39], [12], [40]). However, the approach taken in this thesis will differ from previous studies by applying massive data analysis techniques, such as random forest and the questionnaire algorithm, to the dataset instead of solely relying on statistical methods such as covariance and regression.

The sleep apnea dataset that is analyzed in this section is the result of a study led by Dr. Chuang Kai-Jen and Dr. Liu Wen-Te. The institutional review board of the Taipei Medical University Hospital, TMU-Shuang-Ho Hospital approved the study protocol (TMU-JIRB No. 201412036). It includes measurements for 1154 patients, 816 of which are men and 338 of which are women. The proportion of RDI classifications in the dataset is: Mild = 46.19%, Moderate = 18.63%, and Severe = 35.18%. This data contains values for 34 parameters about each subject one of which is RDI. It is important to note that even though RDI is included in the dataset, it will be taken out of prior to some types of analysis because we will want to predict it using other parameters. A full list and description of the parameters can be found in Table 4.1. Not every parameter value was able to be attained for every subject so approximately 5% of the values in the dataset are missing.

The goal of our analysis is to predict patients’ sleep apnea classification (as according to RDI) based on questionnaire responses. A secondary goal of the analysis is to find which sensors seem to be the most correlated with the classification and which sensors are the most closely related to each other.

Aside from section 4.1 and the hierarchical clustering in 4.2, the code that was used to generate the results in this chapter called the functions for each machine learning algorithm from the Scikit-learn library for Python [33].
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>Age of the subject in years</td>
</tr>
<tr>
<td>RDI</td>
<td>Respiratory Disturbance Index</td>
</tr>
<tr>
<td>Sex</td>
<td>male = 0 and female = 1</td>
</tr>
</tbody>
</table>
| Neckratio | The neckratio is equal to: \[
\frac{\text{neck circumference in cm} \times \text{waist circumference in cm}}{\text{height in cm}}
\] |
| Education | No education = 0, elementary = 1, junior high = 2, senior high = 3, university = 4, and graduate school or higher = 5 |
| Marriage  | single = 0, common law marriage = 1, married = 2, divorced = 3, and widowed = 4 |
| Economics | Poor = 0, fair = 1, good = 2, and wealthy = 3 |
| Smoke     | non-smoker = 0 and smoker = 1 |
| Alcohol   | non-drinker = 0 and drinker = 1 |
| Sedation  | no sleeping pill = 0 and sleeping pill = 1 |
| Other Drug| No other drugs = 0 and any other drug = 1 |
| BMI       | Body Mass Index which is equal to \[
\frac{\text{mass in kg}}{\text{height in m}}^2
\] |
| PSQI-5a, PSQI-5b, PSQI-5c, PSQI-5d, PSQI-5e, PSQI-5f, PSQI-5g, PSQI-5h, PSQI-5i | Answers to questions contained in the Pittsburgh Sleep Quality Index (PSQI), where the number after the hyphen represents the question number from the Appendix of [10] |
| EX-1, EX-2, EX-3, EX-4 | Restless leg syndrome rating scale |
| ESS-1, ESS-2, ESS-3, ESS-4, ESS-5, ESS-6, ESS-7, ESS-8 | Epworth Sleepiness Scale (ESS) values as according to [27] |
| BDI       | Answers to questions that appear on the Beck Depression Inventory (BDI) [2] |
4.1 Clustering the Dataset

Note that RDI has been kept in the dataset for all the clustering results in this section.

The results of $k$-mean clustering with 3 clusters can be seen in figure 4.1 and are visualized in 3 dimensions using principal component analysis (PCA) [28]. The average RDI value in each of the three clusters is: Pink = 33.22, Black = 33.63, and Green = 14.43. This provides us with valuable insight into the natural clusters of subjects and explains why it’s easier for the learning algorithms in Section 4.3 to predict classifications of patients into groups of $RDI < 30$ and $RDI \geq 30$ compared to the groups $RDI < 15$ and $RDI \geq 15$.

Figure 4.1: $k$-means clustering applied to the sleep apnea data with RDI included. The average RDI value in each of the three clusters is: Pink = 33.22, Black = 33.63, and Green = 14.43.

A hierarchical clustering tree is presented in Figure 4.2 and it was created with Cluster3 software [18] using average-linkage and Euclidean distance. It can be observed that BMI, RDI and Neckratio are all closely related.

Spectral clustering results with 3 clusters are presented in Figure 4.3, but they failed to provide useful results as $> 99\%$ of all the subjects ended up in a single cluster and the remaining $< 1\%$ of subjects were split among the other two clusters. This is likely due to the method of imputation that was used which caused subjects with few known parameter values to have identical values for most of their parameters making them very similar. This resulted two extremely small clusters of near identical points.
Figure 4.2: Hierarchical clustering tree of the sleep apnea data with RDI included
Figure 4.3: Spectral clustering applied to the sleep apnea data with RDI included
4.2 Dataset Analysis Using the Questionnaire Algorithm and Dual Geometry Metric

The questionnaire algorithm has also been applied to the sleep apnea dataset using 15 iterations of the process because this is when the results seemed to stabilize. The resulting heat map and sensor tree structure are presented below in Figures 4.4 and 4.5 (which show analysis with RDI) as well as Figures 4.6 and 4.7 (which show analysis without RDI).

It is clear from the results that neckratio and BMI are clustered with RDI when it is included and are still clustered together when RDI is taken out. This suggests that these two parameters have a strong relationship with each other and with RDI. Based on this observation we will apply supervised learning techniques to the dataset using only those two parameters as well as using the set of all parameters. Also, notice that age and marriage are clustered together in both cases which is consistent with expectations.
Chapter 4. Results of the Sleep Apnea Questionnaire Analysis

Figure 4.4: The sensor tree for the sleep apnea data with RDI included

Figure 4.5: Arranged heat map of the values in the sleep apnea data with RDI included
Figure 4.6: The sensor tree for the sleep apnea data without RDI included

Figure 4.7: Arranged heat map of the values in the sleep apnea data without RDI included
Figure 4.8: A comparison of the heat maps for RDI included (left) and RDI excluded (right). Notice that the inclusion/exclusion of RDI has a significant affect on the clustering and organization of the subjects.
4.3 Predicting Subject Classifications Using Supervised Learning Techniques

Random forest, SVM, gradient boosting and linear regression were the four methods of supervised learning applied to the dataset in this section. Each method was applied to the dataset as a whole as well as the subsets consisting of only men or women.

For each method, 50 subjects were randomly selected to be taken out of the dataset so that they could be used for cross-validation. The remaining subjects were used as the training data. Each method was run 100 times taking out a different random set of 50 subjects each time. For every run, all of the 50 subjects that were removed had their classes predicted and compared to their actual classification after which the percent error was recorded. Based on the findings in Section 4.2, we will also analyze the dataset with only the BMI and neckratio ratio parameters included. The average percent error and standard deviation for the methods are presented in Table 4.2 and Table 4.3.

Table 4.2: Analysis results using normalized versions of all parameters in the dataset

<table>
<thead>
<tr>
<th>Classes</th>
<th>Classification Method</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>All</td>
</tr>
<tr>
<td>Not Severe: RDI &lt; 30</td>
<td>Random Forest⁴</td>
<td>23.86% ± 5.37%</td>
</tr>
<tr>
<td>Severe: RDI ≥ 30</td>
<td>SVM</td>
<td>23.88% ± 5.93%</td>
</tr>
<tr>
<td></td>
<td>Gradient Boosting²</td>
<td>26.04% ± 5.56%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>25.26% ± 5.97%</td>
</tr>
<tr>
<td>Mild: RDI &lt; 15</td>
<td>Random Forest⁴</td>
<td>28.78% ± 6.95%</td>
</tr>
<tr>
<td>Not Mild: RDI ≥ 15</td>
<td>SVM</td>
<td>29.34% ± 6.48%</td>
</tr>
<tr>
<td></td>
<td>Gradient Boosting²</td>
<td>28.22% ± 6.08%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>30.46% ± 7.09%</td>
</tr>
<tr>
<td>Mild: RDI &lt; 15</td>
<td>Random Forest⁴</td>
<td>36.72% ± 6.27%</td>
</tr>
<tr>
<td>Moderate: 15 ≤ RDI &lt; 30</td>
<td>SVM</td>
<td>38.12% ± 7.22%</td>
</tr>
<tr>
<td>Severe: RDI ≥ 30</td>
<td>Gradient Boosting²</td>
<td>42.08% ± 6.69%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>46.6% ± 6.88%</td>
</tr>
</tbody>
</table>

Table 4.3: Analysis using only the normalized BMI and normalized neckratio data as the parameters

<table>
<thead>
<tr>
<th>Classes</th>
<th>Classification Method</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>All</td>
</tr>
<tr>
<td>Not Severe: RDI &lt; 30</td>
<td>Random Forest⁴</td>
<td>28% ± 6.14%</td>
</tr>
<tr>
<td>Severe: RDI ≥ 30</td>
<td>SVM</td>
<td>26.16% ± 6.2%</td>
</tr>
<tr>
<td></td>
<td>Gradient Boosting²</td>
<td>36.36% ± 6.17%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>26.4% ± 5.5%</td>
</tr>
<tr>
<td>Mild: RDI &lt; 15</td>
<td>Random Forest⁴</td>
<td>35.12% ± 7.07%</td>
</tr>
<tr>
<td>Not Mild: RDI ≥ 15</td>
<td>SVM</td>
<td>30.8% ± 6.29%</td>
</tr>
<tr>
<td></td>
<td>Gradient Boosting²</td>
<td>48.26% ± 7.28%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>30.24% ± 6.61%</td>
</tr>
<tr>
<td>Mild: RDI &lt; 15</td>
<td>Random Forest⁴</td>
<td>44.9% ± 6.37%</td>
</tr>
<tr>
<td>Moderate: 15 ≤ RDI &lt; 30</td>
<td>SVM</td>
<td>37.54% ± 6.29%</td>
</tr>
<tr>
<td>Severe: RDI ≥ 30</td>
<td>Gradient Boosting²</td>
<td>56.54% ± 6.44%</td>
</tr>
<tr>
<td></td>
<td>Linear Regression</td>
<td>48.96% ± 6.51%</td>
</tr>
</tbody>
</table>

¹Random forest was run with \( n = 100 \) trees.
²Gradient Boosting was run using \( n = 100 \) estimators.
From these results we can see that predicting the classification of females into severe and not serve can be done accurately using our methods. Most of the techniques are also comparably accurate to their results from the set of all parameters when they are applied solely to the BMI and neckratio parameters. This result corroborates the findings from the analysis of the dataset using the questionnaire algorithm. Also, linear regression performs comparably or even better in some cases to the other more sophisticated techniques which suggests that the data has a linear structure. For the classification of females into mild/non-mild or into 3 different classes, SVM seems to be the best performing technique for females.

For male subjects the results are not accurate, but the reason for this is not clear at this time. A larger study is needed to confirm these preliminary findings and the result of that study will be reported in a future paper.
Chapter 5

Discussion and Conclusions

The results of the sleep apnea dataset analysis can be summarized in Table 5.1 which is consistent with what would usually be expected when these techniques are applied.

<table>
<thead>
<tr>
<th></th>
<th>Boosting</th>
<th>Random Forest</th>
<th>CART</th>
<th>SVM</th>
<th>Linear Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity to noise</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>low</td>
<td>low</td>
</tr>
<tr>
<td>Sensitivity to outliers</td>
<td>depends</td>
<td>depends</td>
<td>poor</td>
<td>depends</td>
<td>depends</td>
</tr>
<tr>
<td>Computational Scalability</td>
<td>good</td>
<td>good</td>
<td>very good</td>
<td>poor</td>
<td>good</td>
</tr>
<tr>
<td>Interpretability</td>
<td>poor</td>
<td>poor</td>
<td>good</td>
<td>good</td>
<td>good</td>
</tr>
<tr>
<td>Dependent on Metric</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Predictive Power</td>
<td>good</td>
<td>good</td>
<td>poor</td>
<td>good</td>
<td>depends</td>
</tr>
</tbody>
</table>

Fortunately, when analyzing the sleep apnea dataset we did not have to deal with many problems that commonly arise when analyzing other datasets. Examples of such problems are:

1. **Normalization**: Normalization is an important issue in data analysis which depends heavily on the type of research being done. The dataset and goals of the research determine what normalization method, if any, needs to be applied to the dataset. In our case, the z-score normalization was sufficient to improve the results.

2. **Imputation (Missing Data)**: Imputation is required when information is missing about about a subject with respect to one or more of the sensors. In the case of sleep apnea the missing data was a small portion of the data was missing and only the most basic approach to imputation was taken which involved filling missing cells with the average value for that sensor. In the analysis of other datasets this can be a much larger problem as upwards of 95% of values could be missing. In such cases, far more sophisticated methods of imputation are required to allow for accurate analysis.

3. **Noise**: Noise refers to the values in the dataset including some degree of randomness due to factors such as limitations in measurement, human error, external factors, and so on. The sleep apnea dataset is noisy and we do not apply any preprocessing techniques to address it in this study. Instead, we count on the stability of the analysis techniques we applied to cope with the noise in an adaptive way. Nevertheless, noise reduction techniques are a large and important part of the field of massive data analysis and are often required when dealing with datasets.
4. **Outliers:** Depending on the type of data, outliers can cause significant problems and skew data analysis significantly. In the case of the sleep apnea dataset, the amount of outliers in the dataset is marginal and the ones that did exist weren’t extreme. This allowed us to deal with the outlier issue simply through standard score normalization. In other datasets the outliers can be more prevalent and more extreme resulting in significant problems when applying analysis techniques and models to the data. Such cases require the application of sophisticated outlier reduction techniques.

5. **Visualization:** Another significant area of massive data analysis is visualizing the results so that they can be interpreted as easily as possible. While the sleep apnea dataset and analysis only required basic visualization tools such as tree diagrams and heat maps, there are many cases where such methods are insufficient to accurately and comprehensively convey the results.

6. **Metric Selection:** While this study only applied the Euclidean metric and the dual geometry metric, there are several other metrics that are considered in data analysis such as cosine affinity, correlation. Applying some of these different metrics is a potential further direction for research on the sleep apnea dataset.

The results in this study were accurate with respect to classifying females into classes of severe and not severe sleep apnea. However, further research using a larger dataset and additional techniques is needed to confirm these preliminary results. Another possible direction for further research is the potential theoretical justification for the dual geometry metric through the concept of the *ultramerttic*. A brief introduction the ultrametric can be found in Appendix A.
Appendix A

The Ultrametric

The basic property of an ultrametric space is that it satisfies a different triangle inequality. The familiar triangle inequality is:

\[ d(x, z) \leq d(x, y) + d(y, z) \]

for any triplet \( x, y, z \) while in an ultrametric space this inequality is replaced by

\[ d(x, z) \leq \max\{d(x, y), d(y, z)\} \]

which can be referred to as the ultrametric property. One of the most significant examples of an ultrametric space from pure math is the set of \( p \)-adic numbers, but research in recent years has shown that many large datasets can be approximated by an ultrametric space. It has also been suggested that the sparsely populated cortexes of humans and animals might be ultrametric spaces and that is the reason why they are able to compute so efficiently.

Due to the ultrametric property, points in ultrametric space are usually able to be organized into a hierarchical structure. The dendrogram tree\(^1\) is the most prominent example of a hierarchical structure that can be used to represent data in many ultrametric spaces. This structure makes many computations such as nearest neighbour search much more computationally efficient since the main calculation that has to be done is simply tree traversal.

Large datasets lend themselves to being represented as ultrametric spaces because it has been shown that the ultrametricity of a space increases when the dimension of the data increases. The ultrametricity of a space also increases when it becomes more sparse and decreases when it becomes more dense, although these changes are not as significant as those caused by an increase in dimension. This means that highly sparse data sets that have a very high dimension are very well represented by the ultrametric. Since most large datasets have both those properties, the properties of ultrametric spaces can be used to reduce the computational cost of analyzing them.

Some further insight might be given if we determine how to measure the extent to which a given dataset can be considered to be an ultrametric space. Two early measures for the ultrametricity of a space are Lerman’s \( H \) measure [31] and Rammal’s measure [37] [38]. The measurement that will be explained in this section is the one proposed by Murtagh [32].

The goal of this measure is to find the ultrametricity of a space which is a value \( 0 \leq \alpha \leq 1 \). On this

\(^1\)It is otherwise known as a binary tree where each node is connected to at most 2 nodes on the level below it.
scale 1 means that a space completely satisfies the ultrametric property and 0 means that no portion of the space satisfies it.

To find the exact ultrametricity of a space $\chi$ with $q$ points in $\mathbb{R}^n$ space we would have to consider all triples in the space. This amounts to $\frac{q(q-1)(q-2)}{6}$ which may be incomputable in practice for large $q$ and large $n$. In most cases, large data sets require that we only select a subset of $m$ points $\{v_1, \ldots, v_m\} \subset \chi$.

Once we have this subset of points we calculate the side lengths and angles for each triple in the subset. At this point we discard all triples that form a degenerate triangle\(^2\) from our analysis.\(^3\) For the remaining triples we select the smallest angle $a$ such that $a \leq 60^\circ$. Denote the other angles of the triplet as $b, c$ and these form the base of the triangle. A triangle is considered as approximately satisfying the ultrametric property if $|b - c| < 2^\circ$.

Let $U$ be the count of triplets that satisfy the property $|b - c| < 2^\circ$. Then

$$\alpha = \frac{U}{\frac{m(m-1)(m-2)}{6} - D}$$

where $D$ is the number of triplets that form a degenerate triangle.

\(^2\)A triangle where all three points are approximately collinear.

\(^3\)There are various ways that can be used to determine whether a triangle is degenerate. For a tolerance $\epsilon > 0$ some examples include triangles with an area $< \epsilon$, triangles that have some interior angle $a$ such that $180^\circ - \epsilon < a$, or triangles where the ratio of the area to the sum of the side lengths is less than $\epsilon$. 
Bibliography


