DEGREE PROJECT IN MATHEMATICS,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2019

Cluster analysis on sparse customer data on purchase of insurance products

MICHEL POSTIGO SMURA

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

Cluster analysis on sparse customer data on purchase of insurance products

MICHEL POSTIGO SMURA

Master's programme in Applied and Computational Mathematics
Date: April 5, 2019
Supervisor at Länsförsäkringar: Blerta Capri
Supervisor at KTH: Timo Koski
Examiner at KTH: Timo Koski
School of Engineering Sciences
Host company: Länsförsäkringar AB
Swedish title: Klusteranalys på gles kunddata på köp av försäkringsprodukter

Abstract

This thesis work aims at performing a cluster analysis on customer data of insurance products. Three different clustering algorithms are investigated. These are K-means (center-based clustering), Two-Level clustering (SOM and Hierarchical clustering) and HDBSCAN (density-based clustering). The input to the algorithms is a high-dimensional and sparse data set. It contains information about the customers previous purchases, how many of a product they have bought and how much they have paid. The data set is partitioned in four different subsets done with domain knowledge and also preprocessed by normalizing respectively scaling before running the three different cluster algorithms on it. A parameter search is performed for each of the cluster algorithms and the best clustering is compared with the other results. The best is measured by the highest average silhouette index.

The results indicates that all of the three algorithms performs approximately equally good, with single exceptions. However, it can be stated that the algorithm showing best general results is K-means on scaled data sets. The different preprocessings and partitions of the data impacts the results in different ways and this shows that it is important to preprocess the input data in several ways when performing a cluster analysis.

Sammanfattning

Målet med detta examensarbete är att utföra en klusteranalys på kunddata av försäkringsprodukter. Tre olika klusteralgoritmer undersöks. Dessa är K-means (center-based clustering), Two-Level clustering (SOM och Hierarchical clustering) och HDBSCAN (density-based clustering). Input till algoritmerna är ett högdimensionellt och glest dataset. Det innhåller information om kundernas tidigare köp, hur många produkter de har köpt och hur mycket de har betalat. Datasetet delas upp i fyra delmängder med kunskap inom området och förarbetas också genom att normaliseras respektive skalas innan klustringsalgoritmerna körs på det. En parametersökning utförs för dem tre olika algoritmerna och den bästa klustringen jämförs med de andra resultaten. Den bästa algoritmen bestäms genom att beräkna the högsta silhouette index-medelvärdet.

Resultaten indikerar att alla tre algoritmerna levererar ungefärligt lika bra resultat, med enstaka undantag. Dock, kan det bekräftas att algoritmen som visar bäst resultat överlag är K-means på skalade dataset. De olika förberedelserna och uppdelningarna av datasetet påverkar resultaten på olika sätt och detta tyder på vikten av att förbereda input datat på flera sätt när en klusteranalys utförs.

Contents

Chapter 1

Introduction

Machine learning is defined as developing computer systems that automatically improve their performance through experience [1]. The learning from experience (data) can be categorized as either supervised or unsupervised. In supervised learning, the goal is to predict the value of an outcome measure based on a number of input measures. Unsupervised learning, does not have an outcome measure and instead it aims to describe the associations and patterns among a set of input measures [2]. Machine learning is an area that has evolved in a high pace and new research keeps pushing it further each year. Two reasons for this fast development in machine learning is due to the amount of data being gathered and stored, which also has increased, and the improvements of computational power which has given the opportunity to process and use large amounts of data in reasonable time. All of this has led to machine learning successfully being applied and giving business value in many different fields such as such as business, medicine, astrophysics and public policy, to name a few [3].

Companies has always wanted to understand their customers and trying to segment them to better make decisions giving value to their business and improving their products. This can be done by performing a cluster analysis on information (data) of the customers. A cluster analysis is a part of unsupervised machine learning and aims at grouping the input data in similar subgroups and this is also what will be done in this thesis.

1.1 Problem Statement

The main objective of this work is to perform a cluster analysis on a subset of the customers of the Swedish insurance company Länsförsäkringar, to get

a better understanding of the customers behaviour. This cluster analysis will be performed on previous purchases of the customers to see if there are any interesting patterns in the data. This thesis work aims at answering the main question which is formulated below.

- Which of the clustering algorithms *K-means* (center-based clustering), Two-Level clustering with *Self-organizing map* and *Hierarchical clustering*, and *HDBSCAN* (density-based clustering) performs best on a sparse data set with respect to average silhouette value?

Also of interest is to investigate how different preprocessings of the data in combination with trying to cluster different subsets of the data chosen with domain knowledge, impacts the results. Further interest for this thesis and for the company is to look for interesting patterns in resulting clusterings, i.e. is there any specific customer behaviour to be identified when the customers are clustered their on their previous purchases.

The complete provided data set contains information of the customers of Länsförsäkringar and the data sets used as input to the clustering algorithms contains information about previous purchases, more precisely which and how many products were bought and how much was paid for those products. For more details about the data set see Section 3.1.

1.1.1 Limitations and Scope

The limitations of this work are set to two preprocessings of the input data and to three different ways of performing a cluster analysis. The validation to measure how good the cluster results are is limited to one validation index.

Performing a cluster analysis can be summarized in the following six steps [4].

1. Problem definition.
2. Data acquisition.
3. Data preprocessing and survey.
4. Data modeling.
5. Evaluation
6. Knowledge deployment

Step 1 is presented above and the data (step 2) is provided by the host company Länsförsäkringar. The other steps (3-6) will be accomplished as good as possible in the rest of this work.

1.2 Previous Work

There is similar research trying to cluster customers like [5][6] but they either do not have a real-world data set or have a small data set. They do not try with different combinations of preprocessings and algorithms from different cluster paradigms. Therefore, this thesis work contributes by trying to cluster a large, real-world data set in several different ways.

Chapter 2

Background

2.1 Cluster Analysis

Cluster analysis is the task of grouping a set of objects such that objects in the same cluster are more similar to each other than to objects in other clusters [7]. This is a main task of exploratory data mining and used in unsupervised machine learning. Unsupervised refers to the fact that there are no predefined categories to place the data in, i.e. the correct answer is unknown. This differs from its counter-part called supervised learning where you know the correct data labels.

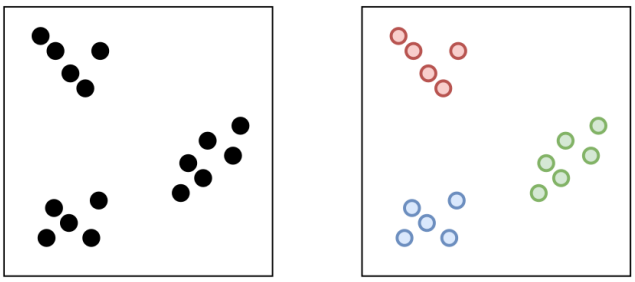

Figure 2.1: An example of data to the left that is clustered to the right.

To perform a cluster analysis it is necessary to first perform a feature selection, data preprocessing, cluster the data and to validate the clusters, this is what will be presented further in this section.

2.2 Data Preparation

Feature selection is an important step in cluster analysis. The primary objective of feature selection is to select relevant features from a data set. This is done for reasons to reduce the dimension of the input data set to the algorithm so that the algorithms can run faster and more efficient, i.e. to remove features that actually does not tell you a lot about the patterns in the data [8]. The importance of reducing dimensionality is also motivated by the curse of dimensionality [9], which mainly states that data becomes sparse with increased dimensions. Not only is it important to choose features but also to consider how to handle outliers [10]. An outlier is a data point that is distant from other points [11]. In this case referring to customers that has bought markedly many insurances and/or paid more than others and therefore deviates from the other customers. When features and data points are chosen the decision to transform the data or not also has to be taken. This is done because the raw data usually varies widely in range. In this work two transformations are used. A linear transformation, in this work referred to as normalization, to make all the columns belong to the same interval $[0, 1]$. See Equation 2.1.

$$x_i' = \frac{x_i - \max(x)}{\max(x) - \min(x)} \tag{2.1}$$

Where x_i is the data point and x is the feature vector it belongs to. Mean/variance normalization, in this work referred to as scaling, centers the data to roughly equating the dynamic ranges along each dimension. It is done by Equation 2.2.

$$x_i' = \frac{x_i - \bar{x}_i}{\sigma_{x_i}} \tag{2.2}$$

Where x_i is the data point and $\bar{x}_i$ is the sample mean and σ_{x_i} is the sample standard deviation [12].

Finally, a distance measure has to be chosen. The goal of clustering is to create similarity in clusters and dissimilarity between clusters, to quantify this a distance function has to be chosen. The type of data and variables affects the choice of distance function. In this work the variables are quantitative, i.e. they are all exact amounts measured by a numeric value and therefore the distance function chosen in this work is the Euclidean distance [11].

2.3 Clustering Algorithms

There are different kinds of paradigms for clustering a data set. Centroid-based clustering [13] aims at finding k centroids such that each point is close to one of the centroids, in this work applied with K-means. The two-level clustering in this work is inspired by Vesanto and Alhoniemi [4]. First the data points are clustered by a self-organizing map and then the self-organizing map is clustered by hierarchical clustering. Hierarchical clustering is by itself another paradigm that creates a tree-like representation of the data and the clusters are extracted by selecting different branches of the tree [14]. The last paradigm used in this work is density-based clustering [15] which assumes that clusters are contiguous dense regions in the data space, separated by areas of low point density, in this work applied with HDBSCAN.

2.3.1 K-means

The K-means clustering partitions a data set into *K* distinct (non-overlapping) clusters. Formally this can be defined as following, let $C_1, \ldots, C_K$ denote sets containing indices of observations in each cluster. These sets satisfies the condition that each observation belongs to exactly one distinct C_i, i.e. the C_i's are disjoint sets.

For a *good* K-means clustering the variance within each cluster should be as small as possible and the variation between clusters should be clear. This is an optimization problem, to minimize the following function:

$$\sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \tag{2.3}$$

where p is the amount of points in the cluster. As seen the function to be optimized is the square Euclidean distance [3].

The K-means clustering is a NP-hard problem [16] and therefore heuristic algorithms has been developed. The one used in this work is the one proposed by Hartigan and Wong [17], which finds a local optimum. It is good to run the algorithm several times since it does not converge to a global optimum.

The Hartigan and Wong algorithm will be briefly explained below, for a more detailed description see [17].

- n is the number of d-dimensional vectors to be clustered
- k is the number of clusters

- μ_j is the center of the cluster j
- Let $m_1^{(t)}, m_2^{(t)}, \ldots, m_k^{(t)}$ be the k cluster centers at iteration t.
- Let $S_i = \left\{ x_p : ||x_p - m_i^{(t)}||^2 \leq ||x_p - m_j^{(t)}||^2, \forall j \in [1, k] \right\}$
- Let $\Phi(S_j) = \sum_{x \in S_j} (x - \mu_j)^2$ be the individual cost of S_j

Then the algorithm is as follows:

- **Assignment step:** Partition the data points into random clusters $\{S_j\}_{j \in \{1,\ldots,k\}}$.
- **Update step:** Determine $n, m \in \{1, \ldots, k\}$ and x_n for which the following function reaches a minimum $\Delta(m, n, x) = \Phi(S_n) + \Phi(S_m) - \Phi(S_n \setminus \{x\}) - \Phi(S_m \cup \{x\})$. For the minimum values move x from S_n to S_m.
- **Termination:** Once $\Delta(m, n, x) > 0$ for all x, n, m the algorithm determines.

To illustrate an example of the update steps for computing the new cluster centers see Figure 2.2. [1]

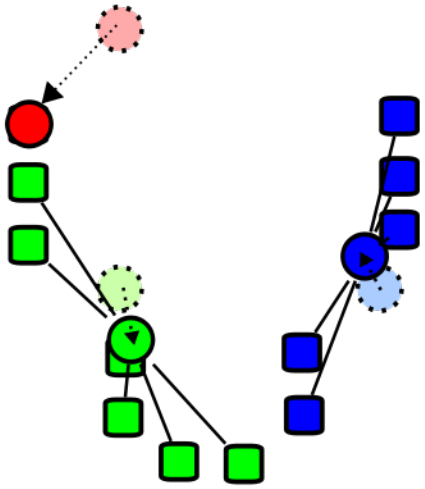

Figure 2.2: An example of the update steps where the centroids are moved to the average of within cluster data points.

[1] https://commons.wikimedia.org/wiki/File:K_Means_Example_Step_3.svg, 2019-03-27

K-means is a clustering algorithm that is reasonably efficient with respect to within-class variance. It is also easily programmed and computationally economical so it scales good with large data sets [18]. K-means is a partitional clustering algorithm, i.e. the algorithm aims at finding a single partition decided by the input k. This is one of the drawbacks with K-means, that is why you usually run it for different k and compare the results [19].

2.3.2 Self-Organizing Map

The self-organzing map (SOM) by Kohonen [20] is an artificial neural network (ANN) that is used to classify the input space based on similarity. It does so by mapping each data point to a two (sometimes one) dimensional network topology and thus reducing the dimension of the data. The network learns to detect regularities and correlations in their inputs. A SOM learns to recognize in a way such that neurons physically near each other in the neuron layer respond to similar input vectors [21].

The two-dimensional grid of map units represents the SOM. Each unit i is represented by a prototype vector $m_i = [m_{i1} \ldots, m_{id}]$, where d is the dimensions of the input vector. These prototype vectors are connected to adjacent ones through a neighbourhood relation and the number of map units determines the accuracy and generalization capability of the SOM. Elements of the prototype vectors of each unit are initialized randomly.

The SOM learns by competitive learning where all neurons that represents the output layer compete for each input vector under a winner-takes-it-all condition. It is trained iteratively and at each training step a sample vector x is randomly chosen from the input data set. Then the Euclidean distance between x and all the prototype vectors are computed. The best matching unit (BMU), which is denoted by b is the map unit prototype closest to x (Equation 2.4).

$$||x - m_b|| = \min_i \{||x - m_i||\} \tag{2.4}$$

When the BMU is found the next step is to update the prototype vectors. The BMU and its topological neighbours are moved closer to the input vector that was randomly selected from the input data. The update rule is seen in Equation 2.5.

$$m_i(t+1) = m_i(t) + \alpha(t)h_{bi}(t)(x - m_i(t)) \tag{2.5}$$

Where

- t is the time (iteration)

- $\alpha(t)$ is the adaption coefficient (learning rate)
- $h_{bi}(t) = exp\left(-\frac{||r_b - r_i||^2}{2\sigma^2(t)}\right)$ is the neighbourhood kernel centered on the winning prototype vector (topological neighbourhood).

Where r_b and r_i are positions of neurons b and i on the SOM grid and $\sigma(t)$ is the radius of the neighbourhood of the kernel. Both $\alpha(t)$ and $\sigma(t)$ decreases monotonically.

Figure 2.3[2] illustrates input vectors training the SOM grid, with the winning node as dark and the neighbourhood nodes as darker.

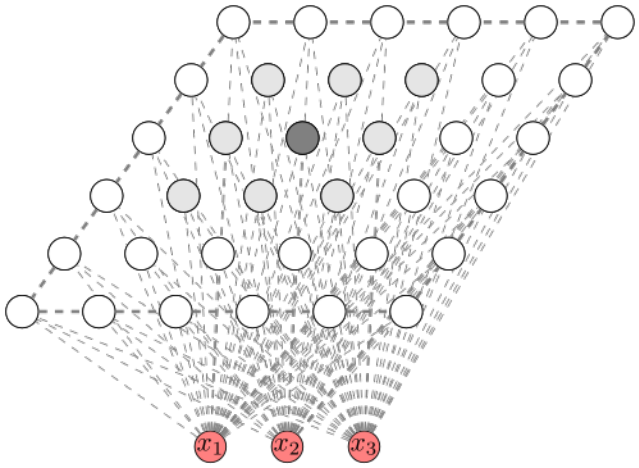

Figure 2.3: An example of the training of the SOM.

2.3.3 Hierarchical clustering

Hierarchical clustering seeks to step by step build a hierarchy of clusters. There are two main ways to achieve this. The first one is called *agglomerative* approach where each observation starts in its own cluster, and in each step merge pairs of clusters as moving up the hierarchy. The other one is called *divisive* approach which starts with all observations in one cluster and step by step splits the clusters when moving down the hierarchy [7]. The one used in this work is the agglomerative approach. The hierarchical clustering also requires a defined distance and an agglomeratieration criterion. The ones used are euclidean distance and complete-linkage. Complete linkage [22] is defined as following: the distance D_{ij} between two clusters C_i and C_j is the

[2]https://commons.wikimedia.org/wiki/File:Self-organizing-map.svg, 2019-03-24

maximum distance between two points $x \in C_i$ and $y \in C_j$, i.e. $D_{ij} = \max_{x \in C_i, y \in C_j} d(x, y)$, where $d(x, y)$ is the distance function (in this case euclidean distance) [23]. The steps in the algorithm is described in James et al. [3]:

1. Begin with n observations and a distance measure of the observations.
2. For $i = n, n-1, \ldots, 2$
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the most similar one and merge those clusters.
   (b) Compute the new pairwise inter-cluster dissimlarities among the $i - 1$ remaining clusters.

In Figure 2.4 [3] an example is illustrated of agglomerative hierarchical clustering.

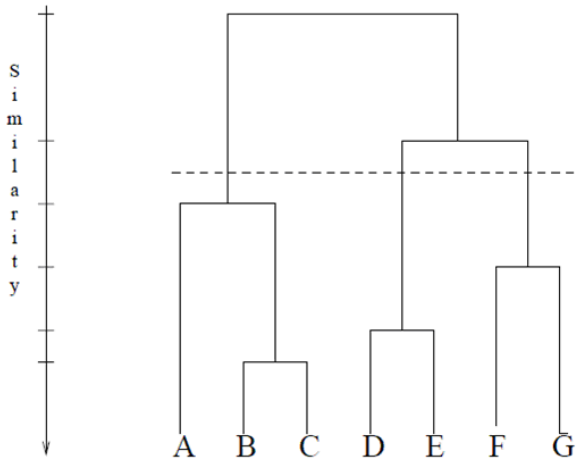

Figure 2.4: An example of agglomerative hierarchical clustering.

2.3.4 HDBSCAN

Hierarchical Density-Based- Spatial Clustering of Applications with Noise (HDBSCAN) [24] is a hierarchical density-based clustering algorithm. As

[3] Source: https://commons.wikimedia.org/wiki/File:Agglomerative_clustering_dendogram.png, 2019-03-27

the name indicates it is a hierarchical DBSCAN [25]. One of the most significant differences is that HDBSCAN can find clusters of varying density (unlike DBSCAN). HDBSCAN is a powerful algorithm that generates a complete density-based clustering hierarchy from which a simplified cluster tree composed only of the most significant clusters can easily be extracted, so it can be used for visualization of data. It also detects outliers in the data set. The HDBSCAN has a single input parameter m_{pts} which can be interpreted as the minimum amount of points being close to each other to be considered a cluster and not an outlier [24]. An overview of the algorithm will be presented below (for more details see the work of Campello et al. [24]). Define first a core object, an object x_p is called a core object if its ϵ-neighbourhood contains at least m_{pts}, i.e. if $|N_\epsilon(x_p)| \geq m_{\text{pts}}$, where $N_\epsilon(x_p) = \{x | d(x, x_p) \leq \epsilon\}$, if a point is not a core point it is noise. Core distance, $d_{\text{core}}(x_p)$, is defined as the distance of an object $x_p \in X$ to its m_{pts}-nearest neighbour and mutual reachability distance as $d_{\text{mreach}}(x_p, x_q) = \max(d_{\text{core}}(x_p), d_{\text{core}}(x_q)), d(x_p, x_q))$. The mutucal reachability graph is defined as a complete graph $G_{m_{\text{pts}}}$, in which the objects of X are vertices and the weight of each edge is the mutual reachability distance between the respective pair of objects. Then the HDBSCAN algorithm can be explained in the steps below.

1. Compute core distance for all data objects in X.
2. Compute a minimum spanning tree (MST [26]) of $G_{m_{\text{pts}}}$.
3. Extend the MST to obtain MST_{ext} by for each vertex adding a "self edge" with the core distance of the corresponding object as weight.
4. Extract the HDBSCAN hierarchy as a dendrogram from MST_{ext}:
   (a) For root of tree assign all objects as single cluster
   (b) Iteratively remove all edges from MST_{ext} in decreasing order of weights
      i. Before removal, set dendrogram scale value of current hierarchical level as the weight of edge to be removed
      ii. After removal, assign labels to connected components that contains end vertices of removed edges. To obtain next hierarchical level assign a new cluster label to component if it still has at least one edge, else assign it as noise.

Figure 2.5[4] illustrates core distance for three of the points based on min-pts parameter and the mutual reachability between green and red point.

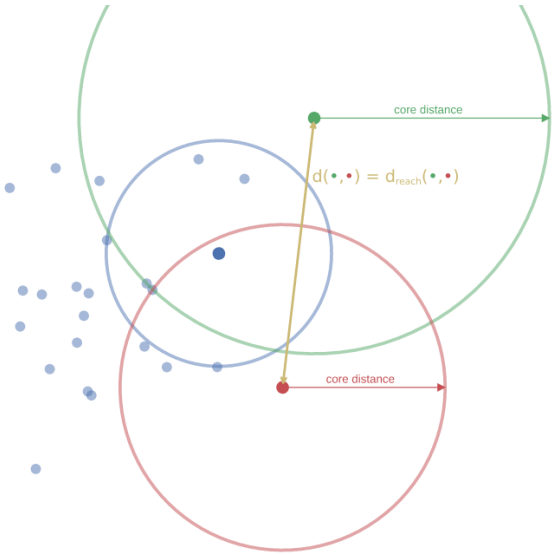

Figure 2.5: An example of how HDBSCAN clusters data points.

2.4 Validating Clustering Algorithms

As mentioned, cluster analysis is a part of unsupervised machine learning, i.e. the correct answer is unknown. This means that it is more difficult to know if the results are good or not. Therefore, measures to validate the clustering partitions goodness has been developed. These measures should be used to get an indication of best model parameters and insight of how many true clusters are hidden in the data. The clustering validity indices combine information about intracluster compactness and intercluster isolation, as welll as other factors [23]. Some example of clustering validity indices are total within sum of square (WSS), the Dunn index [27] and the Silhouette index [28].

2.4.1 Silhouette Index

The Silhouette index [28] is a measurement for validating clusters, it is used as an indication of how good a clustering is. It is a internal validation index that measures the degree of confidence in the clustering assignment of

[4]Source: https://hdbscan.readthedocs.io/en/latest/$_images/distance5.svg$, 2019 − 03 − 27

a particular observation [29]. The idea of the algorithm is to for a clustering partition $C_1, C_2, \ldots, C_k$ of a data set for each cluster compute the average distance (e.g. Euclidean distance) between its data points and compare that to the average distance it has to the data points in the nearest cluster. Therefore, needed to compute silhouette values is a clustering obtained by running a clustering algorithm on the data set and the collection of al distances between objects. For a data point i let $a(i)$ be the average distance from i to all other data points within the same cluster and $b(i)$ be the average distance from i to all data points in its nearest cluster. Then the silhouette index is computed as seen in Equation 2.6.

$$s(i) = \begin{cases} 1 - a(i)/b(i) & \text{if } a(i) < b(i) \\ 0 & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1 & \text{if } a(i) > b(i) \end{cases} \tag{2.6}$$

As seen in Equation 2.6 it holds that $-1 \leq s(i) \leq 1$ and a high silhouette value indicates a good clustering, the total silhouette score for clustering is obtained by taking the mean of all the $s(i)$ [28]. The advantages is that the only thing you need to compute the average silhouette value is the resulting clustering and the distances between all the points. It also considers not only internal goodness of the clusters but measures it to a neighbouring cluster. The disadvantage is that it requires a lot of memory, i.e. a distance matrix of size $\mathcal{O}(N^2)$, where N is the total amount of data points.

Chapter 3

Methodology

In this chapter the methods of the work performed is presented. The preprocessing of the data, the parameter search for the clustering algorithms, the validation of the results and the investigation of the cluster distributions. The data set used will also be presented in more details in Section 3.1.

The three different cluster algorithms are performed on four different subsets of the data (see Figure 3.1) and all the data sets are preprocessed in two different ways, this results in four different clusterings of each data set and a total of 24 clustering results. All of the code and every analysis was done using the programming language R, version 3.5.1. For every random step the same seed was set.

The steps of the cluster analysis can be summarized in the following steps.

1. Preprocessing of data
2. Parameter search and clustering
3. Analyzing results

3.1 Data

The data set clustered in this work contains N data points (customers) and 68 variables (features). The features are information about previous purchases the customers have done, i.e. how many items of a specific product they have bought and how much they have paid for those products. All of the values in the data set are numerical values that lies in the following interval $[0, \infty)$, recall that this interval is valid before any preprocessing is done. These type

of features are called quantitative variables [11], which means they indicate an exact amount measured as a value.

The variables will in the report be referred to as *Var 1*, . . . , *Var 68*. These variables has with knowledge about the company and the business also been divided into different subgroups as seen in Figure 3.1. The partition of the data set is done with domain knowledge. This is motivated by the curse of dimensionality [9] that states that when the dimension of the data increases it becomes sparse and more difficult to analyze. Reducing dimensions also decreases computational time and needs less memory.

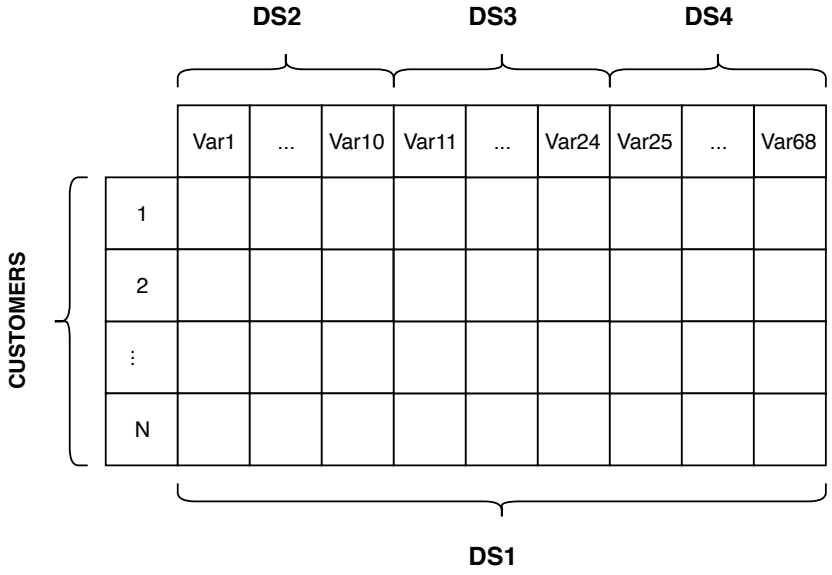

Figure 3.1: The different partitions of the data set to be clustered.

In Table 3.1 some information about the data is presented to get an overview of it. The whole data set (DS1 in Figure 3.1) is sparse and there are 95,4% zeros in it (Table 3.1). It could also be interesting to look at other attributes of the data, for example as maximum value but to not expose the companies secrecy this information is not presented in this report.

The amount of data points (customers) in the data set might be reduced for the different clustering algorithms. This is due to memory limitations. If the data sets were reduced it will be mentioned under the corresponding section on how many data points and which data points was removed.

Furthermore, there are no missing data for the customers in these variables and also that extreme values exists but no action has been taken towards them because they are true values and not a result of incorrect measurements, but it

Data Set	Amount of Zero values	Mean
1	95,4 %	14 158
2	88,4 %	66 780
3	98,9 %	1 442
4	96,9 %	327

Table 3.1: Overview information of the data sets.

is good to mention that some of the values for some customers, relatively, are very high.

Also, each customer belongs to a group and of interest for the company is to look at the distribution of this variable in the resulting clusters and this will also be done and presented in this report.

3.2 Preprocessing of Data

Before the running the clustering algorithms on the data it was preprocessed. All the columns of the data set were separetly scaled or normalized. Also the data set was divided in different subsets as seen in Figure 3.1. So the data sets clustered by all of the algorithms were normalized/scaled DS1, DS2, DS3 and DS4.

3.3 Parameter Search and Clustering

As mentioned all the clustering algorithms were run on each of the differently preproccesed data sets. In this step the methodology differs but not in the preprocessing step and the analysis of the results, which is the same. Depending on what algorithm was chosen the parameter search is different here the details are explained on what was done for the different algorithms.

3.3.1 K-means

The K-means clustering algorithm is performed with the package "stats" and the silhoutte values are computed with the package "cluster" in R.

Because of computational and memory limitations the amount of customers in the data set was reduced by $0,13\%$. The data points removed were chosen at random (with a set seed).

The parameter search for K-means is to determine an optimal number for k, which determines the number of clusters. The K-means cluster analysis was performed as following.

1. Running the K-means algorithm for $k \in [2, 3, \ldots, 25]$. Other relevant input data arguments to the algorithm was letting the max number of iterations being set to 100 and to run the algorithm with 25 different starting states (since the K-means converges to a local optimum).

2. Compute the average silhouette value for each of the of the clusterings.

3. Choosing the K-means clustering with the highest average silhouette value for further analysis.

As mentioned, the above steps were performed on all the data sets as seen in Figure 3.1 and on normalized and scaled data.

3.3.2 Two-Level Clustering

The two-level approach used in this work is inspired by Vesanto and Alhoniemi [4]. The two steps are to first train a SOM on the data and then to cluster the resulting self-organizing map with hierarchical clustering using complete linkage. The SOM algorithm is performed with the package "kohonen" [30], the hierarchical clustering is performed with the package "stats" and the silhoutte values are computed with the package "cluster" in R.

Computational and memory limitations were not a restriction for this algorithm and therefore all of the N data points were used.

The parameter search for this algorithm was to decide where to cut the dendrogram produced by the hierarchical clustering, i.e. decide the optimal number of clusters.

1. Create a quadratic hexagonal SOM-grid, i.e. the sides are of the same size and each node has 6 immediate neighbours. The size of the sides were decided by $x = y = \lfloor \sqrt{5 * \sqrt{N}} \rfloor$. This grid size is suggested byVesanto and Alhoniemi [4].

2. Train a SOM on the data by presenting the whole data set a 1000 times (1000 iterations) and let the learning rate linearly decrease in the following interval $\alpha \in [0.05, 0.01]$.

3. Cluster the trained self-organizing map with hierarchical clustering by merging on complete linkage.

4. Compute the average silhouette index on clusters (cuts) from $[2, 3, \ldots, 25]$.
5. Choosing the cut which gives the highest average silhouette value for further analysis.

As mentioned, the above steps were performed on all the data sets as seen in Figure 3.1 and on normalized and scaled data.

3.3.3 HDBSCAN

The HDBSCAN clustering algorithm is performed with the package "dbscan" and the silhoutte values are computed with the package "cluster" in R.

Because of computational and memory limitations the amount of customers in the data set was reduced by $1,34\%$. The data points removed were chosen at random (with a set seed).

The parameter search for HDBSCAN is to determine an optimal number for the parameter "min-pts", which determines which points are noise and which are not. The HDBSCAN clustering was performed as seen below.

1. Run the HDBSCAN clustering algorithm for min-pts $\in [10, 20, \ldots, 100]$.
2. Compute the average silhouette value for each of the clusterings (remember to remove the points that HDBSCAN classifies as noise).
3. Choose the HDBSCAN clustering with the highest average silhouette value for further analysis.

As mentioned, the above steps were performed on all the data sets as seen in Figure 3.1 and on normalized and scaled data.

3.4 Analyzing the Clusters

Also of interest for this thesis and the company was to look at the results of the clusters trying to identify customer patterns. For each of the 24 models the clusters that are larger than 5% are presented in tables in Chapter 4. Then the within cluster distribution is also presented in tables looking at which group the customers within the cluster belongs to, with the restriction that only groups that has more than 5% of the customers are presented.

Chapter 4

Results

In this chapter the results of the different runs of the clustering algorithms are presented. With two different preprocessings, four different partitions of the original data set and three different clustering algorithms a total of 24 different clusterings were performed trying to find some patterns of customer behaviour in the given data set with information about the purchases the customers of the company have done. As described in Chapter 3 (Methodology) a parameter search was performed for each of these 24 clusterings and in this section only the results of the best clustering (with respect to the average silhouette value) are presented.

4.1 Best clusterings

In Table 4.1 the algorithms that performed the best clustering are presented, i.e. three clusterings for each data set and preprocessing combination. The table presents the average silhouette value for the clustering, how many clusters were found and what parameter produced this result. In the case of K-means and Two-Level clustering the number of clusters and parameter are the same value. For K-means k is equal to number of clusters and for Two-Level clustering number of clusters is equal to where the the cut in the dendrogram is made. As seen in Table 4.1, K-means performed best on scaled DS1 with a average silhouette value of 0.936 (second highest 0.872). K-means also performed best on normalized DS2 and scaled DS3. For DS4, Two-Level clustering performed best when the data was scaled.

K-means								
	DS1		DS2		DS3		DS4	
	Norm	Scaled	Norm	Scaled	Norm	Scaled	Norm	Scaled
Number of Clusters	2	2	2	24	4	2	2	2
Silhouette Value	0.720	0.936	0.989	0.986	0.970	0.987	0.752	0.824
Two-Level Clustering								
	DS1		DS2		DS3		DS4	
	Norm	Scaled	Norm	Scaled	Norm	Scaled	Norm	Scaled
Number of Clusters	2	3	3	3	2	2	2	2
Silhouette Value	0.704	0.872	0.807	0.855	0.900	0.952	0.478	0.917
HDBSCAN								
	DS1		DS2		DS3		DS4	
	Norm	Scaled	Norm	Scaled	Norm	Scaled	Norm	Scaled
Min Points	70	90	90	40	80	70	10	10
Number of Clusters	41	2	4	2	8	8	352	341
Silhouette Value	0.715	0.746	0.979	0.975	0.652	0.611	0.834	0.805

Table 4.1: The best clustering by the algorithms, number of clusters, parameter value and the average silhouette value.

4.2 Clusters size

Area of interest for this thesis (and for the company) was also to look at the distribution of the resulting clusters to see if there is any interesting behaviour of the customers, these results are presented here. Only the clusters that contain more than 5% of the total amount of the data points N are presented. This is due to the fact that some of the cluster algorithms found many and really small clusters and these are considered insignificant for finding any customer behavioural patterns, also it would not be any feasible way to present all the clusters in the report in a user friendly way. None of the best clusterings resulted in more than two clusters containing more than 5% of the total amount of customers. In Table 4.2 and Table 4.3 the clusters and their size for normalized respectively scaled data are presented.

DS1			
K-means	Cluster:	1	2
	Size:	90,56%	9,44%
Two-Level	Cluster:	1	-
	Size:	99,99%	-
HDBSCAN	Cluster:	4	40
	Size:	7,12%	50,72%

DS2			
K-means	Cluster:	1	-
	Size:	100,00%	-
Two-Level	Cluster:	1	-
	Size:	99,67%	-
HDBSCAN	Cluster:	4	-
	Size:	96,82%	-

DS3			
K-means	Cluster:	3	-
	Size:	98,27%	-
Two-Level	Cluster:	1	-
	Size:	99,97%	-
HDBSCAN	Cluster:	7	8
	Size:	5,76%	73,68%

DS4			
K-means	Cluster:	1	2
	Size:	91,00%	9,00%
Two-Level	Cluster:	1	-
	Size:	99,78	-
HDBSCAN	Cluster:	349	-
	Size:	51,09%	-

Table 4.2: Cluster sizes found in the different data sets, only 5% and higher are presented, this is for normalized data.

4.3 Distribution within clusters

Furthermore, the within cluster distribution of which group the customer belongs to will also be limited to only looking at groups that contains at least 5% of the data points belonging to the specific cluster, with the same motivation as

DS1			
K-means	Cluster:	1	-
	Size:	99,94%	-
Two-Level	Cluster:	1	-
	Size:	99,99%	-
HDBSCAN	Cluster:	2	-
	Size:	98,20%	-

DS2			
K-means	Cluster:	9	-
	Size:	96,00%	-
Two-Level	Cluster:	1	-
	Size:	99,99%	-
HDBSCAN	Cluster:	2	-
	Size:	99,58%	-

DS3			
K-means	Cluster:	1	-
	Size:	100,00%	-
Two-Level	Cluster:	1	-
	Size:	100,00%	-
HDBSCAN	Cluster:	7	8
	Size:	6,83%	73,89%

DS4			
K-means	Cluster:	2	-
	Size:	98,53%	-
Two-Level	Cluster:	1	-
	Size:	100,00%	-
HDBSCAN	Cluster:	338	-
	Size:	50,99%	-

Table 4.3: Cluster sizes found in the different data sets, only 5% and higher are presented, this is for scaled data.

for previous constraint. The large clusters were presented above and here the distribution within every cluster is presented for the variable group which were of interest for the company. These are seen in Table 4.4 to Table 4.11 and are divided into datasets and preprocessing so that they easily can be compared.

DS1									
Group:		1	2	C	F	G	J	L	M
K-means	Cl 1		20,07%	5,03%	12,62%	12,89%	6,47%	5,49%	16,46%
	Cl 2						25,48%		52,37%
Two-Level	Cl 1		18,25%		11,58%	12,03%	8,27%	5,20%	19,85%
HDBSCAN	Cl 4						25,88%		53,01%
	Cl 40	5,24%	32,88%		6,71%	10,91%	6,78%		16,76%

Table 4.4: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for normalized data.

DS2							
Group:		2	F	G	J	L	M
K-means	Cl 1	18,30%	11,58%	12,07%	8,24%	5,18%	19,80%
Two-Level	Cl 1	18,30%	11,59%	12,06%	8,24%	5,18%	19,79%
HDBSCAN	Cl 4	18,39%	11,62%	12,20%	8,14%	5,17%	19,65%

Table 4.5: Cluster distritubtion for variable group. Only the groups that are larger than 5% and higher are presented, this is for normalized data.

DS3								
Group:		1	2	F	G	J	L	M
K-means	Cl 3		18,36%	11,53%	12,02%	8,27%	5,23%	19,79%
Two-Level	Cl 1		18,25%	11,59%	12,04%	8,25%	5,20%	19,85%
HDBSCAN	Cl 7	12,15%	38,77%	5,90%	8,56%			14,70%
	Cl 8		14,86%	12,82%	11,83%	8,89%	6,16%	20,46%

Table 4.6: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for normalized data.

DS4									
Group:		1	2	C	F	G	J	L	M
K-means	Cl 1		20,07%	5,03%	12,62%	12,89%	6,47%	5,49%	16,46%
	Cl 2						25,48%		52,37%
Two-Level	Cl 1		18,28%		11,48%	12,05%	8,28%	5,20%	19,89%
HDBSCAN	Cl 349	5,43%	33,76%		6,27%	10,46%	7,11%		16,95%

Table 4.7: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for normalized data.

DS1							
Group:		2	F	G	J	L	M
K-means	Cl 1	18,26%	11,58%	12,04%	8,25%	5,20%	19,85%
Two-Level	Cl 1	18,25%	11,59%	12,03%	8,27%	5,20%	19,85%
HDBSCAN	Cl 2	18,54%	11,44%	12,05%	8,30%	5,13%	19,88%

Table 4.8: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for scaled data.

DS2							
Group:		2	F	G	J	L	M
K-means	Cl 9	18,38%	11,62%	12,24%	8,13%	5,15%	19,62%
Two-Level	Cl 1	18,25%	11,58%	12,03%	8,27%	5,20%	19,85%
HDBSCAN	Cl 2	18,31%	11,58%	12,08%	8,25%	5,15%	19,80%

Table 4.9: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for scaled data.

DS3								
Group:		1	2	F	G	J	L	M
K-means	Cl 1		18,26%	11,58%	12,04%	8,25%	5,20%	19,85%
Two-Level	Cl 1		18,25%	11,59%	12,03%	8,27%	5,20%	19,85%
HDBSCAN	Cl 7	11,71%	38,47%	6,21%	8,65%			14,76%
	Cl 8		14,84%	12,81%	11,85%	8,91%	6,16%	20,48%

Table 4.10: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for scaled data.

DS4								
Group:		1	2	F	G	J	L	M
K-means	Cl 2		18,52%	11,19%	12,12%	8,39%	5,17%	20,08%
Two-Level	Cl 1		18,25%	11,57%	12,05%	8,27%	5,19%	19,86%
HDBSCAN	Cl 338	5,44%	33,81%	6,28%	10,48%	7,12%		16,96%

Table 4.11: Cluster distribution for variable group. Only the groups that are larger than 5% and higher are presented, this is for scaled data.

Chapter 5

Discussion

In this Chapter the results of this thesis work will be discussed. How did the different clustering algorithms perform, how did the different preprocssings and the partitioning of the data set impact the results.

First of all the results indicates that sparse data sets can be clustered well with respect to the average silhouette index by different clustering algorithms. All of the algorithms performed well with only three experiments resulting in noticeable lower silhouette values, these were Two-Level clustering on normalized DS4 and HDBSCAN on normalized and scaled DS3. The K-means clustering algorithm performed well and was only outperformed on 2 out of 16 cases as seen in Figure 5.1. The K-means algorithm performs better in scaled data in 3 out of 4 cases and it performs best on DS2 which has the most values larger than zero and is of lowest dimension of all the data sets. The Two-Level clustering algorithm does not perform as well as the K-means, but it is not that far behind. It performs best on scaled data in 4 out of 4 cases. Noticeable is that unlike the two other algorithms it does not perform best on DS2 and that on DS4 there is a large difference on the performance for normalized and scaled data. HDBSCAN performs similar as K-means for DS1, DS2, and DS4 but a lot worse on DS3 which is the most sparse data set containing 98,9% zeroes.

As mentioned the sizes of the clusters and the within distribution also were of interest for this thesis and the company. This is something that is much more difficult to quantify and measure. Since the comparison will be more difficult this will not be used to answer the question of performance but a discussion will be done on the trade-off between high average silhouette values and finding patterns in the clusters. When observing the sizes of the clusters (Table 4.2 and Table 4.3) we see that for scaled data only HDBSCAN finds clusters

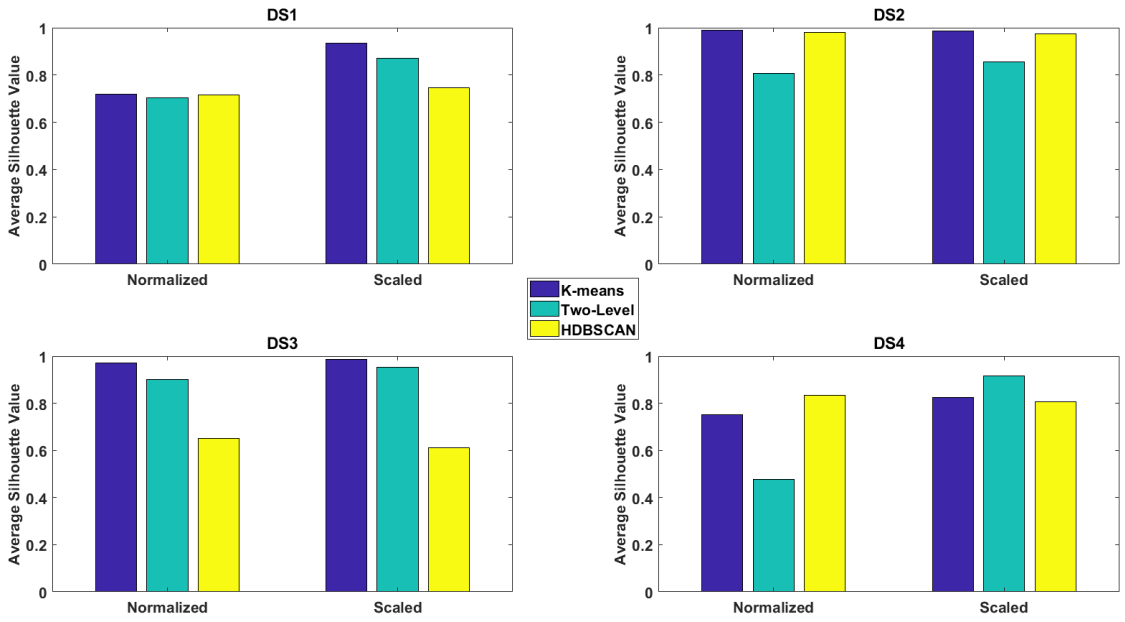

Figure 5.1: Comparing plots of the performance of the clustering algorithms.

in DS3 and DS4 that does not result in putting at least 98,20% of all the data points into one single cluster while still getting a high average silhouette value for DS4. For normalized data K-means also finds clusters where all of the data points does not go into one big cluster for DS1 and DS4. HDBSCAN for normalized data manages to achieve the same for DS1, DS3 and DS4 and getting good silhouette score for DS1 and the best for DS4. Two-Level clustering basically created one big cluster for all the data points on all of the experiments and scored high on silhouette values. K-means could find some clusters for normalized data while HDBSCAN could find clusters and get a good silhouette score, considering this trade-off between "finding" clusters and silhouette value one could argue that HDBSCAN performed best and that the Two-Level clustering the worst. However, when looking at the distribution of the group the customers belong to within the clusters there are no dominating patterns such that to say in this cluster most people belongs to this group.

To summarize the Two-Level clustering performs good compared to the methods used in this thesis on both preprocessings and all of the data sets (the exception being normalized DS4). The advantage of this method is that it also succeeds to cluster all the data points with the constraints on memory and computational time. However, noticeable is that the clusters it created was

by putting at least 99,67% of all the data points in one single cluster for each of the runs. Another advantage of this clustering is that SOM also presents other powerful tools in exploratory data mining (that were not used in this thesis). A disadvantage is that when adding steps to the clustering you also add complexity and it is never a good practice to add unnecessary complexity to data mining [31]. K-means clustering performed best with respect to average silhouette values in most of the cases. It is one of the most used clustering algorithms and is therefore a good measure for comparison, it also performs quite well. Its simplicity also is good because it is easy to understand. The disadvantage is that for it to cluster data perfectly all clusters should be spherical which might not be the case, but it still performs competitively relative the methods used in this work. Another drawback is that you have to choose the number clusters k as input to the algorithm. The HDBSCAN has the disadvantage of performing not as high compared to the others. It also required a reduction of the data set due to memory limitations. If you weigh in the fact that it does not put all the data points in to a single cluster and still gets good scores on average silhoutte value, i.e. a trade-off with silhouette values and cluster sizes it performs good, one could say the best, depending on how you measure this trade-off.

Performing a cluster analysis on a sparse data is not an easy task. It is necessary to have domain knowledge and it is crucial to try different clustering algorithms and different ways of processing the input data. Trying to workaround obstacles as memory constraints and trying to measure how good a clustering is when the real answer is unknown is also difficult. It is not possible to beforehand know what will work, but with experience at least it is possible to obtain an intuition on what steps are reasonable to implement.

Chapter 6

Conclusions

This thesis tried three different clustering algorithms:

- K-means (center-based clustering)
- SOM and Hierarchical clustering (two-level clustering)
- HDBSCAN (density-based clustering)

in combination with different preprocessings and with different feature selections of the customer data set (provided by the host company Länsförsäkringar) trying to find clusters in it and to answer the research questions formulated in Section 1.1.

To select the best clustering a measure is required and the one used in this work is:

- The Average Silhouette Index.

Based on the average silhouette value K-means performed best on scaled DS1 and DS3, and on normalized DS2. Two-level clustering performed best on scaled DS4. Experimenting with preprocessings and feature selection shows also impacts on the results. Scaling the data gives better average silhouette values 8 out of 12 cases and reducing the data set (DS1) gives better 9 out of 16 cases (Table 4.1).

However, the patterns found in clusters are also of interest for this thesis and to measure the performance of patterns found in the clusters is not as straightforward. Considering this it is harder to answer the question on which clustering was best. The results presented in the paragraph above indicates that scaling data usually gives better average silhouette values but by looking at Table 4.3 (scaled data) only HDBSCAN for DS3 and DS4 could find

clusters that does not contain almost all of the data points while in Table 4.2 (normalized data) HDBSCAN and K-means clustered the data in a way such that not almost all the points are in one cluster. As said this is much harder to quantify and to measure so this point will not be a basis to answer the research question.

To summarize, the results indicate that sparse data sets can be clustered well with K-means on scaled data without having to reduce the dimensions of the data. Also, to be mentioned is that no patterns in the clusters of the group belonging of the customers were found. The algorithms that did find clusters that did not cover all of the data points did unfortunately not have any dominating within cluster distribution belonging to a specific customer group. However, even though K-means performed best the others were not that far behind and the results also supports that it is a good practice to preprocess data in different ways and to cluster it with different algorithms when performing a cluster analysis.

6.1 Critique

Aspects of this thesis will be criticized here. One big drawback is that this thesis only compare the results based on one measure. Since we do not possess the correct answer in unsupervised machine learning it is much more difficult to verify the results. One could argue that because of that the importance to have several measures of the performance of the algorithm is even more important. However, in this thesis only the average silhouette value was used to determine the performance of the algorithms. This was due to the fact that there was an interest of experimenting with different preprocessings in combination with feature selections and if the validation indices would not have agreed on the best model, the 24 models presented in Section 4 could instead have been 48 only by increasing the number of validation indices by one. This is not feasible with the scope of this thesis work.

Also when comparing the results it should be remembered that the data set had to be reduced to perform K-means and HDBSCAN due to computational time and memory limitations.

6.2 Further work

A lesson from this thesis is that you have to try several different paths when trying to cluster data, so a natural extension would be to try other algorithms in

combination with other preprocessings. Also to focus more on the fact that the data set is very sparse, insert algorithms such as principal component analysis (PCA) [32] or search for other literature to deal with sparse data in clustering.

To add more business value to the company it could be interesting in developing recommendation systems based on clustering as has been done in [33] [34] [35], to name a few.

Bibliography

[1] Tom Mitchell et al. “Machine learning”. In: *Annual review of computer science* 4.1 (1990), pp. 417–433.

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. eng. Springer Series in Statistics. New York, NY: Springer New York, 2009. ISBN: 978-0-387-84857-0.

[3] G. James et al. *An Introduction to Statistical Learning: with Applications in R*. Springer New York, 2014. ISBN: 9781461471370.

[4] J. Vesanto and E. Alhoniemi. “Clustering of the self-organizing map”. In: *IEEE Transactions on Neural Networks* 11.3 (May 2000), pp. 586–600. ISSN: 1045-9227. DOI: 10.1109/72.846731.

[5] Morteza Namvar, Mohammad R Gholamian, and Sahand KhakAbi. “A two phase clustering method for intelligent customer segmentation”. In: *2010 International Conference on Intelligent Systems, Modelling and Simulation*. IEEE. 2010, pp. 215–219.

[6] Stephen Haben, Colin Singleton, and Peter Grindrod. “Analysis and clustering of residential customers energy behavioral demand using smart meter data”. In: *IEEE transactions on smart grid* 7.1 (2016), pp. 136–144.

[7] Brian S. Everitt et al. *Cluster Analysis*. eng. Chichester, UK: John Wiley Sons, Ltd, 2011. ISBN: 9780470749913.

[8] Pierre A Devijver and Josef Kittler. *Pattern recognition theory and applications*. Vol. 30. Springer Science & Business Media, 2012.

[9] R. Bellman, Rand Corporation, and Karreman Mathematics Research Collection. *Dynamic Programming*. Rand Corporation research study. Princeton University Press, 1957. ISBN: 9780691079516. URL: https://books.google.it/books?id=wdtoPwAACAAJ.

[10] Frank E. Grubbs. “Procedures for Detecting Outlying Observations in Samples”. In: *Technometrics* 11.1 (1969), pp. 1–21. DOI: 10.1080/00401706.1969.10490657. eprint: https://www.tandfonline.com/doi/pdf/10.1080/00401706.1969.10490657. URL: https://www.tandfonline.com/doi/abs/10.1080/00401706.1969.10490657.

[11] Sanghamitra Bandyopadhyay and Sriparna Saha. *Unsupervised classification. Similarity measures, classical and metaheuristic approaches, and applications*. Aug. 2013. DOI: 10.1007/978-3-642-32451-2.

[12] A. Stolcke, S. Kajarekar, and L. Ferrer. “Nonparametric feature normalization for SVM-based speaker verification”. In: *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. Mar. 2008, pp. 1577–1580. DOI: 10.1109/ICASSP.2008.4517925.

[13] Jon M Kleinberg. “An impossibility theorem for clustering”. In: *Advances in neural information processing systems*. 2003, pp. 463–470.

[14] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*. Vol. 20. Siam, 2007.

[15] Michael Hahsler, Matthew Piekenbrock, and Derek Doran. “dbscan: Fast Density-based Clustering with R”. In: *Journal of Statistical Software* 25 (), pp. 409–416.

[16] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. “The Planar k-Means Problem is NP-Hard”. In: *WALCOM: Algorithms and Computation*. Ed. by Sandip Das and Ryuhei Uehara. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 274–285.

[17] J Hartigan and M Wong. “AS136 A K-means clustering algorithm”. eng. In: *Applied Statistics* 28.1 (1979). ISSN: 0035-9254. URL: http://search.proquest.com/docview/1299671140/.

[18] J. MacQueen. “Some methods for classification and analysis of multivariate observations”. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 1967, pp. 281–297. URL: https://projecteuclid.org/euclid.bsmsp/1200512992.

[19] Sobia Zahra et al. “Novel centroid selection approaches for KMeans-clustering based recommender systems”. In: 320.C (2015). ISSN: 0020-0255. URL: http://roar.uel.ac.uk/4298/1/Novel%5C%20centroid%5C%20selection%5C%20approaches%5C%20for%5C%20KMeans-clustering%5C%20based%5C%20recommender%5C%20systems.pdf.

[20] Teuvo Kohonen. “Self-organized formation of topologically correct feature maps”. In: *Biological Cybernetics* 43.1 (Jan. 1982), pp. 59–69. ISSN: 1432-0770. DOI: 10.1007/BF00337288. URL: https://doi.org/10.1007/BF00337288.

[21] Mark Hudson Beale, Martin T Hagan, and Howard B Demuth. “Neural network toolbox™ user’s guide”. In: *The Mathworks Inc* (1992).

[22] T. A. Sorensen. “A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons”. In: *Biol. Skar.* 5 (1948), pp. 1–34. URL: https://ci.nii.ac.jp/naid/10008878962/en/.

[23] Malika Charrad et al. “NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set”. In: *Journal of Statistical Software, Articles* 61.6 (2014), pp. 1–36. ISSN: 1548-7660. DOI: 10.18637/jss.v061.i06. URL: https://www.jstatsoft.org/v061/i06.

[24] Ricardo J. G. B. Campello et al. “Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection”. In: *ACM Trans. Knowl. Discov. Data* 10.1 (July 2015), 5:1–5:51. ISSN: 1556-4681. DOI: 10.1145/2733381. URL: http://doi.acm.org/10.1145/2733381.

[25] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: AAAI Press, 1996, pp. 226–231.

[26] Md. Saidur Rahman. *Basic Graph Theory*. eng. Undergraduate Topics in Computer Science. 2017. ISBN: 3-319-49475-9.

[27] J. C. Dunn†. “Well-Separated Clusters and Optimal Fuzzy Partitions”. In: *Journal of Cybernetics* 4.1 (1974), pp. 95–104. DOI: 10.1080/01969727408546059. eprint: https://doi.org/10.1080/01969727408546059. URL: https://doi.org/10.1080/01969727408546059.

[28] Peter J. Rousseeuw. “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65. ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(87)90125-7. URL: http://www.sciencedirect.com/science/article/pii/0377042787901257.

[29] Guy Brock et al. “clValid, an R package for cluster validation”. In: *Journal of Statistical Software (Brock et al., March 2008)* (2011).

[30] Ron Wehrens, Lutgarde MC Buydens, et al. “Self-and super-organizing maps in R: the Kohonen package”. In: *Journal of Statistical Software* 21.5 (2007), pp. 1–19.

[31] Stephen P Luttrell. “Hierarchical self-organizing networks”. In: *Proc. 1st IEE Conf. Artificial Neural Networks*. British Neural Network Soc. 1989, pp. 2–6.

[32] Karl Pearson F.R.S. “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572. DOI: 10.1080/14786440109462720. eprint: https://doi.org/10.1080/14786440109462720. URL: https://doi.org/10.1080/14786440109462720.

[33] Kyoung-jae Kim and Hyunchul Ahn. “A recommender system using GA K-means clustering in an online shopping market”. In: *Expert systems with applications* 34.2 (2008), pp. 1200–1209.

[34] Qing Li and Byeong Man Kim. “Clustering approach for hybrid recommender system”. In: *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*. IEEE. 2003, pp. 33–38.

[35] Andriy Shepitsen et al. “Personalized recommendation in social tagging systems using hierarchical clustering”. In: *Proceedings of the 2008 ACM conference on Recommender systems*. ACM. 2008, pp. 259–266.

TRITA xx
ISSN xx
ISRN xx