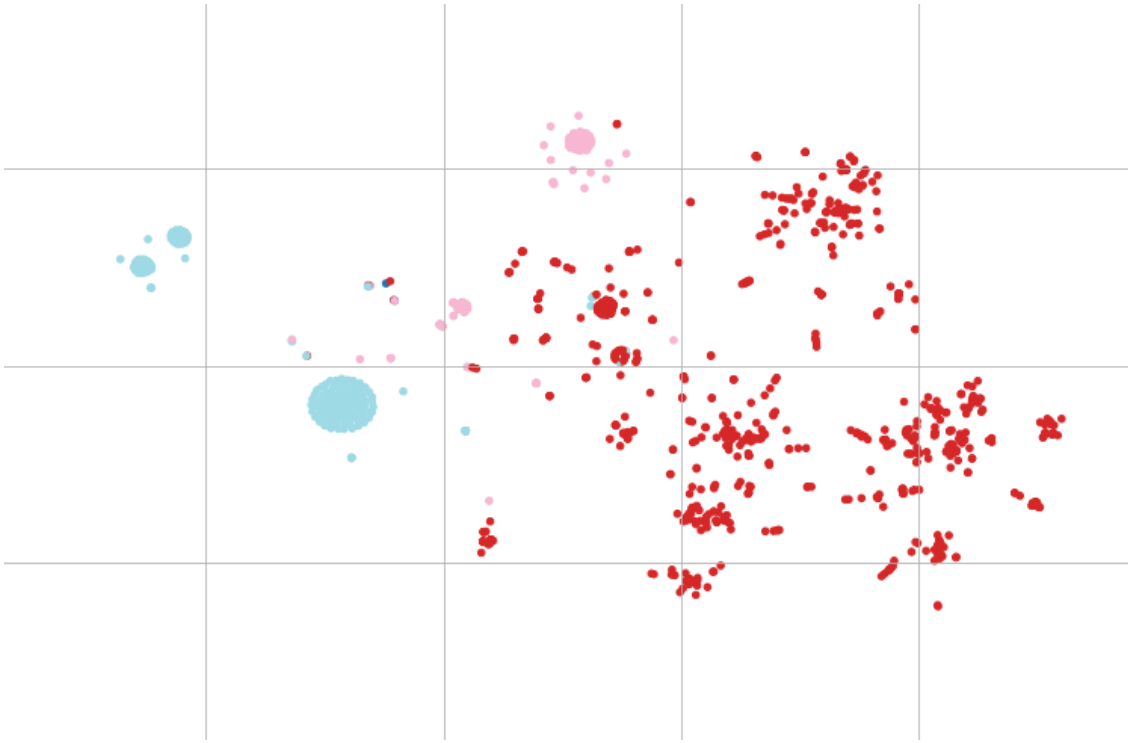

Capturing the Base Station by Feature Engineering

Visualisation and Clustering of Feature Vectors Representing Base Station States

Master's thesis in Mathematical Sciences

Rikard Helgegren

Department of Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2020

Master's thesis 2020

Capturing the Base Station by Feature Engineering

Visualisation and Clustering of Feature Vectors Representing Base Station States

Rikard Helgegren

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2020

Capturing the Base Station by Feature Engineering
Visualisation and Clustering of Feature Vectors Representing Base Station States
Rikard Helgegren

Supervisor: Sergei Zuyev, Mathematical Sciences
Advisor: David Andersson, Ericsson
Advisor: Erik Eklund, Sigma Technology Consultant
Examiner: Serik Sagitov, Mathematical Sciences

Master's Thesis 2020
Department of Mathematical Sciences
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover Image: An example of an interesting attribute which the feature vector captures in a clear pattern.

Typeset in LaTeX
Gothenburg, Sweden 2020

Capturing the Base Station by Feature Engineering
Visualisation and Clustering of Feature Vectors Representing Base Station States
Rikard Helgegren
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

A feature vector is a compact representation of an object or system, that contains the most important or informative aspects of the entity. A feature vector is often used in the context of machine learning due to the effectiveness and noise reduction, but can also be used for data exploration.

In this thesis we want to investigate if a complex system such as a base station can be represented with a feature vector in an useful manner. To create the feature vector, we ask subject matter experts for the most relevant attributes of the base station. The feature vector is evaluated by partitioning all the base stations, either with an unsupervised clustering algorithm, or by an interesting attribute of the base station. The partitions are then visualized and presented to experts, who determined if the partitions brings forth interesting patterns that can be useful, or if the partitions are useful in themselves.

The result is a feature vector containing 494 features, based of 22 attributes that are recommended by subject matter experts. The feature vector brings forth interesting and useful patterns, and we can thus conclude that a feature vector can be used to represent a complex system such as a base station in an useful manner.

Keywords: Feature vector, Visualisation, Clustering, Evaluation metrics, Base station.

Acknowledgements

We would like to thank Ericsson for providing the data and tools to make this thesis possible. A special thanks to our supervisors at Ericsson, David Andersson and Erik Eklund, for giving us guidance and helpful feedback along the way.

Further, we would like to thank our supervisor at Chalmers, Sergei Zuyev at the department of Mathematical Sciences. Thank you for helping us require the scientific level needed, and for valuable feedback on the thesis. We also want to thank our examiner at Chalmers Serik Sagitov, for his input on the thesis.

Lastly, thanks to the opponent for valuable feedback on the thesis. We would also like to thank everybody at Ericsson for creating an enjoyable time with fikas, Monday runs, and much more.

Rikard Helgegren, Gothenburg, July 2020

Contents

1

Introduction

Many large companies automatically generate huge amount of data. Lot of big-data applications have been made and many companies and institutions wants to join the bandwagon of machine learning and AI, in order to find out to what extent they can use their data. With vast amount of data however, it can be hard to get an understanding of what information the data contains, how to find certain information and what the data can be used for. This results in that only people with lot of experience from working with the data are able to use the data in an efficient manner. For a new employee looking at gigabytes of data he or she does not know what is important and it is hard to spot interesting patterns. However if only the most relevant information is extracted and presented, then it is much easier to compare data and find patterns. When the pattern is found then it might be suitable to go back to the full data set [1, 2].

To extract only the most relevant information it is common to create what is known as a feature vector, which is a dense representation of an object. The first approach for creating a feature vector is often to use machine learning methods [3]. This approach is easy to implement and works well on large data sets and can capture complex structures.

However, deep learning algorithms have low interpretability and are often viewed upon as black boxes [4, 5]. This is not desirable if understanding of the results is highly valued, or the patterns found should be interpreted. Therefore some companies would rather continue to apply statistical methods or use domain expertise for creating feature vectors [3].

The question that this project aims to answer is if a complex entity such as a base station state can be represented in a feature vector so that visualising feature vectors for multiple base stations the visualisation is useful or forms distinct clusters. If distinct clusters are found, it could be of great use when exploring the vast data. Moreover, if the feature vector captures distinct underlying properties of a base station, this could be used as a baseline comparison of base stations.

The data in this project is provided by the telecommunication company Ericsson, and therefore the data is gathered from base stations. In telecommunication the base station have a central role. The base station transmits and receives radio signals and is the link between a wired network and the wireless network, e.g. between a mobile phone and a radio tower.

1.1 Thesis outline

In this thesis we engineer a feature vector based on features recommended by subject matter experts. The data used can roughly be categorised in to the configurational data, log message data, and data regarding alarms, however more details regarding the data will not be possible to provide due to confidentiality. Due to missing values, subsets of the data is imputated or omitted depending of what is suitable. In order to make the feature vector suitable for clustering, all features are made comparable by scaling and translation. Then the feature vectors for all base stations are clustered by k-means, Gausian Mixture Models (GMM), Baysian Gausian Mixture Models (BGMM) and Density-based spatial clustering of applications with noise (DBSCAN). The clusterings are visualised by t-SNE and by a heat map of the feature vectors. Based on these two visualisations the usefulness of the feature vector is determined, in combination with clustering evaluation scores. However, what these use cases are is also classified. The conclusion of this study is that it is possible to create a feature vector such that it is useful for representing a base station.

1.2 Related work

As far as we know, no paper has investigated visualisation and clustering of feature vectors in the context of base stations. However, both the papers P. Bodik et al. [6], which is about error classification, and H. Shilin et al. [7], which is about anomaly detection, uses feature vectors for describing complex systems.

Similar to both of these paper, we have in this thesis used a time window. In these papers the time window is used for selecting recent and relevant data. In our case it is mainly a way to reduce the amount of data to make the computations faster, but also for selecting relevant data. Bodik et al. represent their data based on metric quantiles which proves to be efficient for error comparisons, however, for representing a complex object, not focusing on errors, we believe that such a representation would not be suitable. Shilin et al. uses a frequency vector as feature vector for each time window, which is the same approach for handling text strings as in this thesis.

1.2.1 Visualisation

Principal component analysis (PCA) is often used for reducing the number of dimensions before using a scatter plot to visualise the data [8]. This approach was used in the beginning of this project but started to fail when the dimensionality of the data passed 50 dimensions, and almost all data points is presented as one compact cluster. Then t-SNE was implemented instead, which in J. Tang et al. [9] is described as one of the most popular ways to make two dimensional visualisations of high dimensional data. As a complement to t-SNE the data was visualised with heat maps as well, which is also implemented in [6], and [8].

1.2.2 Clustering methods

There exists lots of clustering algorithms based on various approaches. In P. Berkhin [10], the approaches of k-means methods, probabilistic clustering with mixture models, and density based clustering are evaluated in the context of data mining. This includes the problem of clustering high dimensional data and different methods of dimensionality reduction.

1.2.3 Evaluating clusterings

As well as many clustering algorithms there are many clustering validation algorithms of different categories and approaches. M. Halkidi et al. [11] compare the most used validation indexes, and presents strengths and drawbacks with the different indexes. The same is done in J. Handl et al. [12], which evaluates many of the same evaluation indexes as M. Halkidi et al., and points out biases of the different evaluating indexes.

2
Theory

2.1 Clustering

Clustering is the technique of grouping data points together that are similar to each other, or that are probable to be generated by the same phenomenon or object. Typical examples of clustering have well separated groups of data points. A good clustering algorithm would then assign these groups to different clusters, where data points within the same group would be assigned to the same cluster.

Even if it is an easy task for a human to cluster the data points in two dimensions it is hard to write an algorithm that works in all situations, and humans definitely need algorithms when surpassing three dimensions. Clustering algorithms have not one and the same approach but uses different presumptions of what defines a cluster. It is usually a presumption regarding shape of the clusters, the density of the clusters, the distance between clusters, or regarding the number of clusters.

2.1.1 K-means

K-means is a well known clustering algorithm using a simple and fast algorithm for clustering the data into K clusters, where each data point is assigned to the cluster with the nearest cluster mean.

The algorithm works as follows: K data points are selected at random, and are used in the first iteration as cluster means. Then each data point is assigned to the closest cluster mean. Now the new means are calculated as the centers of masses, and the process repeats until the cluster means remain static. [13]

2.1.2 Gaussian Mixture Model

Gaussian mixture models (GMM) is a probabilistic model which uses an expectation maximisation (EM) algorithm to fit its parameters to the data. The model is a weighted sum of K Gaussians, where each Gaussian can be seen as representing a cluster [14]. The GMM is defined as

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{K} w_i g(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2.1}$$

where w_i is the weight associated to the ith Gaussian. The Gaussian is calculated as

$$g(\mathbf{x}|\mu_\mathbf{i}, \mathbf{\Sigma_i}) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}_i|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\}, \tag{2.2}$$

where $\boldsymbol{\mu}_i$ and $\mathbf{\Sigma}_i$ are the mean and the covariance matrix of the ith Gaussian, respectively, and D is the dimensionality of the data vector $\mathbf{x}$.
The Gaussian mixture model have three parameters to fit, that is w_i, $\boldsymbol{\mu}_i$, and $\mathbf{\Sigma}_i$ which will collectively be represented by $\lambda = \{w_i, \boldsymbol{\mu}_i, \mathbf{\Sigma}_i\}$.

One method for estimating the parameters of GMM so that it matches the data is to use maximum likelihood in combination with EM. The idea behind EM is to initiate λ and then calculate new parameters λ' which improves the model. This is done in repeat until convergence according to some set threshold. The equations to improve the parameters are as follows: For mixture weights

$$w_i = \frac{1}{T}\sum_{t=1}^{T}\Pr(i|\mathbf{x}_t, \lambda), \tag{2.3}$$

for means

$$\boldsymbol{\mu}_i = \frac{\sum_{t=1}^{T}\Pr(i|\mathbf{x}_t, \lambda)\mathbf{x}_t}{\sum_{t=1}^{T}\Pr(i|\mathbf{x}_t, \lambda)}, \tag{2.4}$$

and for variances

$$\sigma_i^2 = \frac{\sum_{t=1}^{T}\Pr(i|\mathbf{x}_t, \lambda)x_t^2}{\sum_{t=1}^{T}\Pr(i|\mathbf{x}_t, \lambda)} - \mu_i^2. \tag{2.5}$$

The probability for the ith component is given by

$$\Pr(i|\mathbf{x}_t, \lambda) = \frac{w_i g(\mathbf{x}|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i)}{\sum_{k=1}^{K} w_k g(\mathbf{x}|\boldsymbol{\mu}_k, \mathbf{\Sigma}_k)}. \tag{2.6}$$

2.1.3 Bayesian Gaussian Mixture Model

The Bayesian Gaussian Mixture model (BGMM) is closely related to the GMM. However BGMM also adds regularisation by computing and using the full posteriors. This makes the algorithm less prone to overfitting but makes the model a bit biased [15, 16]. For full details, read H. Attias [16].

2.1.4 DBSCAN

Density-based spatial clustering of applications with noise (DBSCAN), is a clustering algorithm that clusters data points that are close to each other to the same cluster and data points that are in sparse regions are classified as noise [17].

DBSCAN have two important parameters, ϵ which is the radius that determines if something is regarded as close or not, and m, which is the smallest number of data points that will be considered a cluster. The algorithm classifies each data point into one of the following.

- A **core point**, if there is at least m other data points within a radius of ϵ.

- A **border point** if there is a core point within the radius of ϵ, but the point is not a core point itself.
- An outlier if it is none of the above.

A core point form a cluster with all core points and border points that is reachable by moving between points with steps of maximum length of ϵ.

2.1.5 Spectral clustering

Spectral clustering have some similarities with DBSCAN in that it finds high density areas [18]. The algorithm will not be explained in detail since it does not make out a substantial part of this thesis. In short the clustering works as follow:

First create a graph representation of the data by using k-nearest neighbour. Then compute the Laplacian matrix of the graph. Calculate the K first eigenvectors of the matrix, these will span a K-dimensional space. This results in a $N \times K$ matrix where each row is a base station in this K-dimensional space. At this stage K-means is used for partitioning the data points in to K clusters.

2.2 Data Visualisation

Data visualisation is the technique of graphically present information in a explanatory way, in order to make it easier to grasp the information of the data. To present the information the graphical object often uses colours, various shapes, and various sizes, as to help the viewer find patterns and relations within the data.

2.2.1 t-SNE

t-distributed stochastic neighbor embedding (t-SNE) is a visualisation technique for visualising high dimensional data in two and three dimensions. The technique preserves local distances so that data points that are similar are embedded close to each other, and dissimilar points are distant [19].
The best mapping is defined as finding the minimum of the sum of Kullback-Leibler divergences, Equation 2.7, which is found by using gradient descent.

$$C = \sum_{i}^{N} \sum_{j}^{N} p_{ji} \log \frac{p_{ji}}{q_{ji}} \tag{2.7}$$

Given the perplexity *Perp*, number of iterations T, learning rate η, and momentum $\alpha(t)$, first compute the pairwise affinities $p_{j|i}$ by

$$p_{j|i} = \frac{\exp(- \parallel x_i - x_j \parallel^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(- \parallel x_i - x_k \parallel^2 / 2\sigma_i^2)}, \tag{2.8}$$

where σ_i is the variance of the Gaussian, determined with a binary search so that it matches the perplexity according to Equation 2.9.

$$Perp = 2^{-\sum_j^N p_{j|i} \log_2 p_{j|i}} \tag{2.9}$$

The pairwise similarities are then defined by using the pairwise affinities as

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}. \tag{2.10}$$

Sample an initial data representation $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from a normal distribution with small variance. Next, for T iterations the following is preformed:
(i) Calculate the low-dimensional similarities

$$q_{ij} = \frac{(1+ \parallel y_i - y_j \parallel^2)^{-1}}{\sum_{k \neq l}(1+ \parallel y_k - y_l \parallel^2)^{-1}}. \tag{2.11}$$

(ii) Calculate the gradient of the sum of the Kullback-Leibler divergences

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j) \tag{2.12}$$

(iii) Calculate the updated mapping

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\partial C}{\partial \mathcal{Y}} + \alpha(t)(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}). \tag{2.13}$$

After the iterations the low-dimensional data representation $\mathcal{Y}^(T)$ is complete.

To summarise, the goal is to find a mapping $\mathcal{Y}$ which minimises the cost function C, which is the same as making the similarities in the original space p_{ji} and the similarities in the mapping q_{ji}, as similar as possible. This is done by incrementally updating the mapping with gradient decent.

2.2.2 PCA

Principal component analysis is a dimension reduction method that finds an orthogonal coordinate system such that the variance is maximised along each axis. That is, given data points $\mathbf{x}_1, ..., \mathbf{x}_N$ it finds the unit vector $\mathbf{r}$ that maximises

$$\sum_{i=1}^{N} (\mathbf{r}^T(\mathbf{x_i} - \bar{\mathbf{x}}))^2. \tag{2.14}$$

Then to find the other axis the data is projected to the orthogonal complement of all $\mathbf{r}$s' that have been calculated and repeat the above again until the desired dimensionality is accomplished.

2.2.3 Heat map

A heat map is a visualisation tool for showing the magnitude of matrix's elements. The advantage with this visualisation tool compared to both PCA and t-SNE is that the data do not have to be mapped or projected but can be viewed in the feature space. A problem in t-SNE is that a pattern in the visualisation can be hard to interpret, and might not be meaningful in the feature space. This is not the case with a heat map. However, it can be harder to find patterns in a heat map, since the patterns might not be as visible.

2.3 Preprocessing data

The aim of preprocessing data is to have the similar magnitude and scale for each attribute. This is of uttermost importance when using data exploration algorithms as for example clustering. The reason is that without preprocessing, attributes with larger values would dominate the attributes that have low values [20, 21].

2.3.1 Standardization

Standardisation, also known as the z-score, is a commonly used preprocessing method which centralises each feature around 0 and scales it by dividing by the standard deviation of each feature. The standardization is defined as

$$X' = \frac{X - \mu}{\sigma},$$

where μ is the mean and σ is the standard deviation.

2.3.2 Normalization

Normalization of data is done by subtracting the smallest value for each feature, and dividing by the range of the feature. The Normalization is defined as

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}. \tag{2.15}$$

2.3.3 Robust scaling

The Robust scaling is defined as

$$X' = \frac{X - X_{median}}{X_{IQR}}, \tag{2.16}$$

where X_{median} is the median for each feature, and X_{IQR} is interquartile range, that is the difference between 75th and 25th percentiles. This preprocessing method is not as influenced by outliers as when the mean or the variance are used.

2.4 Missing values

When using gathered data there is sometimes the problem of missing values. This can be caused by an error in the object inspected, that not all data is allowed to be gathered from an object, or that there are different sampling frequencies for different data. To deal with the missing values there are three methods [22].

The first one is to simply remove the observation with missing values. This could bias the data if there are a lot of missing values or if the values are not missing at random [23]. But it is appropriate if the data set is large enough.

Secondly, the missing values can be imputed with values that are reasonable. For example if no values are found for the number of errors related to well working objects, it could be reasonable to impute with the value 0. This imputed value would then be used as if it was observed. However a risk with imputation is that it can lead to artificial similarities.

Lastly, based on the other observations an unbiased value can be calculated by using maximum likelihood.

2.5 Mixed types of data

There are two approaches for clustering data with both categorical and numerical data. First there are clustering algorithms that handles mixed types [24]. The second approach is to convert the data so that all data is either numerical or categorical [21].

One common way to convert categorical data to numerical is to use one-hot-vectors [21]. That is, a vector with the same size as the number of categories is created; an observation is then converted so that if, for example, the third categorical value is observed this would result in a vector with all zeroes except for at the third place of the vector where there would be one. The same method can be used for converting text to numerical values. Then every unique word in the data is considered a category, and a string consisting of the 4^{th}, 7^{th}, and 8th word would result in a one-hot-vector with only ones at the 4^{th}, 7^{th}, and 8^{th} position. Both when converting categories and strings to one-hot-vectors, the result is usually spars.

2.6 Evaluation indices

There are two major ways to evaluate the quality of a clustering. There is the External evaluation, which measures how well the clustering matches some external labels which is provided as a ground truth. And then there is Internal evaluation, which without external information uses the structure of the clustering to evaluate the goodness of the clustering.

2.6.1 External evaluation

The external indices are designed to estimate the similarity of two partitions, they do not however give any indications of how good the partitions are in themselves. These indices are determined by the confusion matrix describing the count of pairs of data points that are in the same cluster according to two partitions P_1 and P_2. Thus, there are four possible classifications of a pair of data points.

- If the pair are in the same clustering according to P_1 and P_2, this is denoted *yy*.
- If the pair belongs to the same clustering according to P_1 but **not** P_2, this is denoted *yn*.

- If the pair does **not** belong to the same clustering according to P_1 but does so according to P_2, this is denoted ny.
- And lastly, If the pair does **not** belong to the same clustering according to P_1 **nor** according to P_2, this is denoted nn.

The total amount of parings is thus $N_T = \frac{N(N-1)}{2} = yy + yn + ny + nn$, where N is the total number of data points.

2.6.1.1 Adjusted Rand index

The Adjusted Rand index is an improvement of the Rand index which is defined as $\frac{yy+nn}{N_T}$. The rand index calculates the ratio of parings which P_1 and P_2 agrees on relative to the total amount of pairings. The rand index however have the flaw that if the two partitions where randomly generated, the index would increase when increasing the number of clusters, due to the increase of nn. Therefore the adjusted Rand index was proposed by Hubert and Arabie, which is a more advanced index that do not suffer from the same problem [25]. The adjusted Rand index is defined as:

$$S = \frac{\binom{N}{2}(yy + nn) - [(yy + yn)(yy + ny) + (ny + nn)(yn + nn)]}{\binom{N}{2}^2 - [(yy + yn)(yy + ny) + (ny + nn)(yn + nn)]} \tag{2.17}$$

where $[(yy + yn)(yy + ny) + (ny + nn)(yn + nn)]$ is the expected index [26].

2.6.1.2 Jaccard index

The Jaccard index is a combination of the two frequently used terms

$$\text{Recall} = \frac{yy}{yy + yn} \tag{2.18}$$

and

$$\text{Precision} = \frac{yy}{yy + ny}. \tag{2.19}$$

The Jaccard index is defined as the intersection of P_1 and P_2, divided by the union. Which in our notation is the same as

$$S = \frac{yy}{yy + yn + ny}. \tag{2.20}$$

2.6.1.3 Hubert Γ index

Define a variable X_k associated to partition P_k. $X_k(i, j) = 1$ if two data points i and j are in the same cluster according to P_k, otherwise $X_k(i, j) = 0$. The Hubert Γ index is defined with this indicative variable X_k, as the average correlation between X_1 and X_2 [27].
Also define the mean μ_{X_k} and the standard deviation σ_{X_k}

$$\mu_{X_k} = \frac{1}{N_T} \sum_{i<j} X_k(i, j) \tag{2.21}$$

$$\sigma_{X_k}^2 = \frac{1}{N_T} \sum_{i<j} X_k(i,j)^2 - \mu_{X_k}^2 \tag{2.22}$$

Then the Hubert Γ index is calculated as

$$S = \frac{\sum_{i<j}(X_1(i,j) - \mu_{X_1})(X_2(i,j) - \mu_{X_2})}{N_T \sigma_{X_1} \sigma_{X_2}} \tag{2.23}$$

2.6.2 Internal evaluation

Internal evaluation are clustering measures that are based on mainly three traits:
Compactness, which increases with small variation within a cluster. This measure works well for spherical clusters and when the clusters are well separated.
Connectedness, a local method that is based on the principle that neighbouring data points should belong to the same cluster. This measure works good for clusters of all shapes that have a high density.
Spatial separation, improves with increased separation between clusters, however this can result in trivial solutions especially if there are outliers in the data [12].

2.6.2.1 Dunn index

The Dunn index combines compactness and spatial separation and is defined as

$$S = \frac{d_{min}}{d_{max}}, \tag{2.24}$$

where d_{min} is the smallest distance between points belonging to different clusters, and d_{max} is the largest distance between points belonging to the same cluster [27]. The distances are defined as follows:

$$d_{min} = \min_{k \neq k'} (\min_{\substack{x_i \in C_k \\ x_j \in C_{k'}}} \| x_i - x_j \|), \tag{2.25}$$

and

$$d_{\max} = \max_{1 \leq k \leq K} (\max_{\substack{x_i, x_j \in C_k \\ i \neq j}} \| x_i - x_j \|), \tag{2.26}$$

where C_k is a cluster and K is the total number of clusters.

2.6.2.2 Davies–Bouldin index

Davies–Bouldin index is measuring the within cluster scatter for each cluster and the cluster separation between all clusters. The index almost calculates the average of the fraction between the within cluster scatter and the distance between a cluster and the cluster closest to it in space [27]. For the true definition is as follows.

Define the center of a cluster C_k as

$$\mu^{\{k\}} = \frac{1}{n_k} \sum_{x_i \in C_k} x_i, \tag{2.27}$$

where n_k is the number of data points in C_k. Then the mean distance for the points $x_i^{\{k\}}$ in the cluster C_k to the center $\mu^{\{k\}}$ is defined as

$$\delta_k = \frac{1}{n_k} \sum_{x_i \in C_k} \parallel x_i^{\{k\}} - \mu^{\{k\}} \parallel, \tag{2.28}$$

Which makes it possible to define the Davies–Bouldin index as

$$S = \frac{1}{K} \sum_{k=1}^{K} \max_{k' \neq k} \left(\frac{\delta_k + \delta_{k'}}{\mu^{\{k'\}} - \mu^{\{k\}}} \right), \tag{2.29}$$

where K is the total number of clusters.

2.6.2.3 Silhouette width

The silhouette width index is roughly formulated as an averaged value of how suitable it would be to reassign a data point to another cluster. This is based on how close it is to data points in its current cluster, and how close it is to the data points in the other clusters respectively.

Define the mean distance a data point have to all other data points belonging to the same cluster C_k as

$$a_i = \frac{1}{n_k - 1} \sum_{\substack{x_i, x_i' \in C_k \\ i' \neq i}} d(x_i, x_{i'}), \tag{2.30}$$

and define the mean distance for this point to each of the other clusters $C_{k'}$ as

$$\partial(x_i, C_{k'}) = \frac{1}{n_{k'}} \sum_{\substack{x_i \in C_k \\ x_i' \in C_{k'}}} d(x_i, x_{i'}). \tag{2.31}$$

Now the cluster resulting in the minimum of all $\partial(x_i, C_{k'})$ would be the best choice for changing the cluster point x_i belongs to. Therefore we calculate what is called the silhouette width of a point as

$$s_i = \frac{\min_{k' \neq k} \partial(x_i, C_{k'}) - a_i}{\max(a_i, \min_{k' \neq k} \partial(x_i, C_{k'}))}. \tag{2.32}$$

s_i can assume values between -1 and 1, where 1 indicates that the point x_i is assigned to the right cluster and -1 indicates that x_i should be assigned to another cluster. The silhouette width for a cluster is defined as

$$S_k = \frac{1}{n_k} \sum_{x_i \in C_k} s_i, \tag{2.33}$$

and finally the silhouette index is then the mean of the silhouette widths of all clusters, that is

$$S = \frac{1}{K} \sum_{k=1}^{K} S_k. \tag{2.34}$$

2.6.2.4 S-Dbw index

The S-Dbw index is defined as the sum of average scattering of clusters $\mathcal{S}$ based of the variance for each feature and the inter-cluster density D-bw denoted as $\mathcal{G}$ [11,12]. Define a vector $\mathcal{V}$ of variances $\mathrm{Var}(V_i)$ for each of the p variables $V_1, ..., V_p$

$$\mathcal{V} = [\mathrm{Var}(V_1), ..., \mathrm{Var}(V_p)], \tag{2.35}$$

and likewise the variances for each cluster C_k

$$\mathcal{V}^{\{k\}} = [\mathrm{Var}(V_1^{\{k\}}), ..., \mathrm{Var}(V_p^{\{k\}})]. \tag{2.36}$$

Then $\mathcal{S}$ is given as

$$\mathcal{S} = \frac{\frac{1}{K}\sum_{k=1}^{K} \parallel \mathcal{V}^{\{k\}} \parallel}{\parallel \mathcal{V} \parallel}. \tag{2.37}$$

In order to define $\mathcal{G}$, first define σ as

$$\sigma = \frac{1}{K}\sqrt{\sum_{k=1}^{K} \parallel \mathcal{V}^{\{k\}} \parallel}. \tag{2.38}$$

Then the density $\gamma_{kk'}$ for a given point is defined as the number of data points belonging to the clusters C_k and $C_{k'}$ within radius σ of this point in space.
The inter-cluster density $\mathcal{G}$ is then defined as

$$\mathcal{G} = \frac{2}{K(K-1)} \sum_{k<k'} \frac{\gamma_{kk'}((\mu^{\{k'\}} + \mu^{\{k'\}})/2)}{\max(\gamma_{kk'}(\mu^{\{k\}})\gamma_{kk'}(\mu^{\{k'\}}))} \tag{2.39}$$

Now the S-Dbw index can be calculated as

$$S = \mathcal{S} + \mathcal{G} \tag{2.40}$$

3
Method

3.1 Data

The data used in this report is provided by the telecommunication company Ericsson. The data is pre-extracted from base stations and stored in tabular form as text strings, categorical values and numerical values. The details of the data are classified but the data consists of information regarding the configuration of the base stations and their software, alarms, and log data. Three data sets have been used, containing data from years back,with millions of recorded events per day. Each event contain information regarding the 10 000 attributes that are recorded by these three files.

When managing the huge size of data that been used in this project we have used the analytics engine Apache Spark in Python (PySpark), and the data analysis and manipulation tool Pandas, which also is for Python. We used PySpark for filtering and for the first steps of managing the data. The filtering results in a massive reduction of data so that it was possible to continue with Pandas which is more compatible with Python libraries used in this thesis.

3.1.1 Filtering and Merging

Due to huge amount of data we decided in the beginning to only analyse data from one day, otherwise the compiling time would slow down the project and the data would require too much computer memory. There are three files that we extract data from, these contain the attributes that subject matter experts (SME) have classified as extra relevant. When merging the data from different data files, we matched the base station name, timestamp and date, and customer, as was recommended by experts.

3.1.2 Missing data

In this thesis we have two situations of missing data. There are some time frames where there is no configuration data for some base stations, and then there are some time frames where we have no alarm data or log data for some base stations.

Base stations with no configuration data is removed from the data set. This is motivated by the fact that all base stations have a configuration and if the information about the configuration is absent in the data we have no interest in that base station before that error is corrected. The numbers of base stations after omission are 1079.

The reason for missing alarm and log data is that there were no alarms for that base station in that time period, so it is reasonable to impute with zeroes where no alarms nor log messages was recorded.

3.1.3 Preprocessing data

All categorical data and text data have been transformed to one-hot-vectors in order to be compatible with the numerical algorithms that are used. If a word or a category appears multiple times the number of occurrences is stored instead of a one. After the transformation the data was standardised in order for the clustering algorithms to perform well.

3.2 Visualisation

In the beginning of the project, PCA was used for visualisation to evaluate new parameters and as a way to gain insight how a certain parameter changed the distances and how the data is grouped. However with increasing number of parameters and also the problem of outliers contributing to the variance, a change from PCA to t-SNE was made to easier asses if the feature vector would be suitable for clustering.

t-SNE was also used for inspecting how the clustering algorithms clustered the base stations together. Along with t-SNE, heat maps of the feature vector was also used for inspecting the result of the clustering. The heat map was used by first sorting the feature vectors based on their cluster and then plot the heat map. This made it possible to look for patterns directly, without mapping the feature vectors down to two dimensions. For an illustration of patterns in a heat map see Figure 4.2.

3.3 Clustering

When clustering the data we have used well known clustering methods from the python library Scikit-learn [28]. This include K-means, GMM, BGMM, DBSCAN, and Spectral Clustering. The choices was based on what others use [11, 12, 24, 29], and what have been used in machine learning courses at Chalmers. The clustering algorithms have also been chosen so that they cover different approaches of clustering algorithm. The clustering of the data is done in the original space before the result is mapped down to two dimensions for visualisation.

Each clustering algorithm was executed several times with different hyper-parameters in order to find optimal settings. For k-means, GMM, and BGMM, the number of clusters was altered to find the best clustering, and for DBSCAN the distance of what was considered a neighbour was altered as well as the limit of what is considered a cluster.

The parameters that is most suitable for the clustering was determined by doing 30 runs of calculating the different index scores for each setting and then taking the

average score to represent the score for that setting. Then by inspecting the plot of scores, the parameter values that resulted in the highest score and parameters that resulted in extra high peaks, was selected for extra inspection.

The scores used during the inspection was for internal indexes the Silhouette width, the S-Dbw index, the Dunn index, and the Davies-Bouldin index, and the external indexes used was the Adjusted rand index, the Jaccard index, and the Huberts Γ index. The label used as the truth for the external evaluation scores, was attributes selected by SMEs. The goal with this was to see if a clustering matches some previous known attribute well.

3.4 Evaluating results

The main way to evaluate if the feature vector is useful was to ask the SMEs of their opinions. This is due to that they have the best understanding of the base stations and what is useful to know, or extract. The experts were given a document explaining the context, which data that have been used, and the visualisations with explanatory text of which features that was extra important for the different clusters, or which known attribute that the clustering was made with. The experts was then asked to rate the clusterings of the feature vector from 1 (not useful) to 5 (useful), where 3 means *might be useful*.

The second way of evaluating the results was by using evaluation indexes. We decided to use common and well known evaluation indexes [12, 27]. In hope that these will be more general and useful than evaluation metrics that are not as well tested or tailored for specific problems. The signs of the S-Dbw index and the Davies–Bouldin index were inverted, so that a higher index would indicate a better clustering, same as the other indexes. The indexes used in the evaluation were only the internal indexes, since they are a more objective quality measure then the external, and the external labels are not the actually aim of the clustering.

3.5 Label matching

When implementing the Jaccard index for evaluation, the clustering algorithm can sometimes find the same clusters as the labels without receiving a good score. This is due to that the clustering might have labelled the clusters differently. For example, if we have two clusters and the clustering algorithm names them 2 and 1 but our reference labels are the other way around, that is 1 and 2. Then this would result in a low score when in fact is spot on. Therefore we have written a relabeling algorithm

that tries to match the labels in a good way.

Algorithm 1: Re labelling

Create confusion matrix
while *there exists matrix element not equal to -1* **do**
    Select largest value and remap so that labelling matches the reference label.
    Set all values on same row and column in the confusion matrix to -1, in order to avoid mapping multiple labels to same reference label, and to avoid remapping a already managed label.
end

4
Results

4.1 Feature vector

The feature vector was created by using 22 attributes, resulting in 494 features. 84 of the features describe configuration data. These features are based on 14 attributes, and it is 2 categorical attributes that make up most of the 84 features. 73 features describe log data, which are based on 3 attributes. These are all strings that is converted to an one-hot-vector. 377 features describe alarm data. These features are based on 5 attributes which of 2 are strings, the strings make up most of the features.

4.2 Figures and Findings

Figure 4.1 shows the result of a how t-SNE have mapped the feature vector of each base station to two-dimensions. In this particular figure the data is first preprocessed by normalization and then clustered by BGMM, which can be extracted from the figures title. The legend of the figure shows that the clustering algorithm found two clusters, ”0” coloured dark blue, and ”1” coloured light blue. This is a very similar clustering to one of the labels recommended by a subject matter expert (SME), used for testing if the feature vector could capture an interesting attribute. The clustering is presented in Figure A.12

Figure 4.2 is the same clustering of the feature vectors as in Figure 4.1 but now presented in a heat map. The heat map have feature vectors as rows, and the features as columns. To clarify, traversing horizontally the different features for a base station is changing, and traversing vertically the base station is changing but the feature is the same. The rows of the heat map are sorted by cluster number so cluster "0" is at the top and the highest cluster number is at the bottom. In Figure 4.2 the two clusters can be identified based on the horizontal shift in the heat map. White pixels represent high values, gray pixels is the value zero, and black pixels are negative values. This makes it clear that the feature vectors are sparse since most of the heat map is gray.

Figure 4.3 is also a t-SNE plot, same as Figure 4.1, but they uses Standardisation for preprocessing instead of Normalization which results in a different mapping. One clear distinction between the preprocessing methods is that standardisation seem to be more prone to outliers, which can be seen on the left side in Figure 4.3. Another

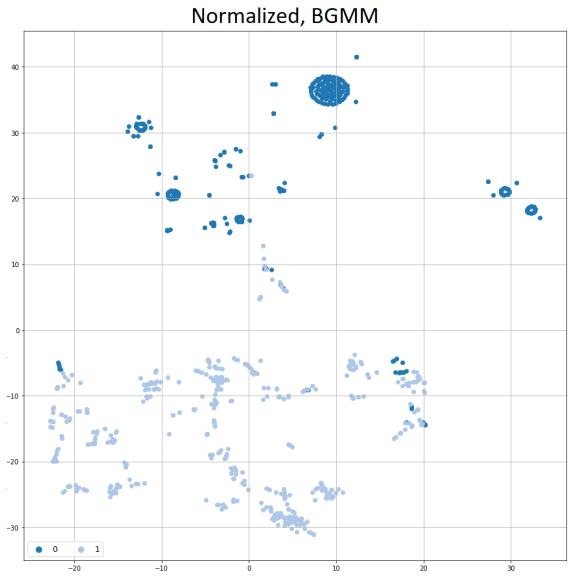

Figure 4.1: This BGMM clustering of the normalized feature vectors separates the feature vectors quite well. The clustering is similar to a label classified as interesting by a SME, See Figure A.12.

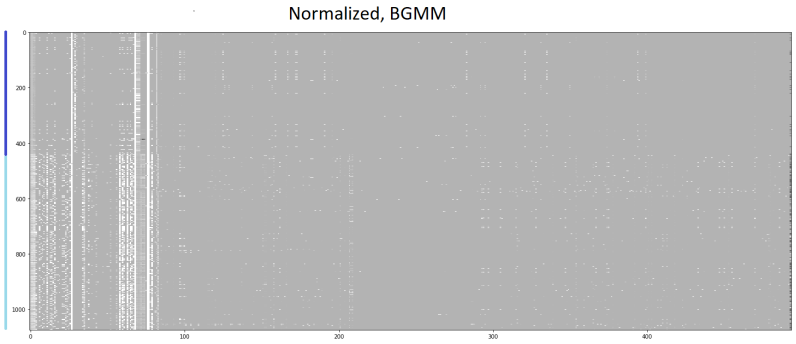

Figure 4.2: This BGMM clustering of the normalized feature vectors shows a quite clear change of feature vectors close to feature vector 400. This shift makes it probable that the clustering is good.

big distinction between Figure 4.3 and Figure 4.1 is that the latter is coloured by a clustering algorithm, whereas Figure 4.3 is coloured by an attribute considered interesting by a SME. The colouring fits the mapping well, forming distinct clusters as can be seen in Figure 4.3.

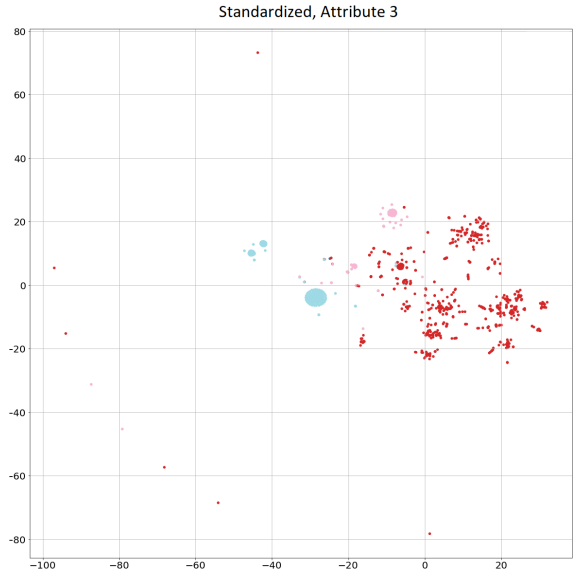

Figure 4.3: Clustering by an attribute considered interesting by a SME. The attribute presents clear patterns in the visualised feature vectors, which indicates that the feature vector is capable of capturing attributes of this kind.

4.2.1 Clustering by attributes

For evaluating the feature vector, one of the approaches was to colour the t-SNE mapping with attributes that the SMEs thought was interesting. The reasoning was that if the feature vector was able to capture these attributes, by presenting clear patterns, that would indicate usefulness. Six of these attributes are shown in Figure 4.4, and all attributes are in the Appendix A, where they are in full size. The attributes 1 to 4 show fairly clear patterns whereas attribute 5 is unclear, and attribute 6 present no clear pattern.

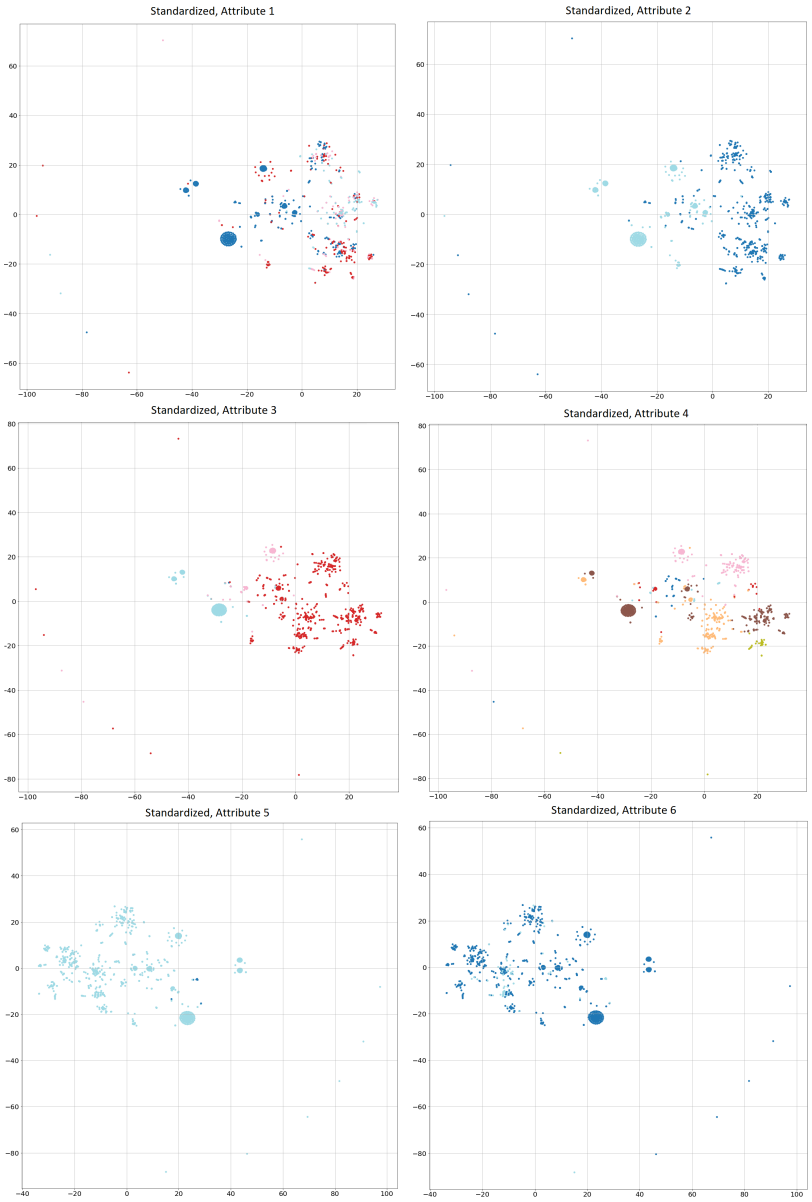

Figure 4.4: t-SNE mapping of standardized feature vectors coloured by selected attributes 1 to 6, counting left to right, top to bottom. The colouring of the t-SNE mappings shows quite clear patterns for attributes 1 to 4, 5 is debatable whether it is a good cluster or not and attribute 6 display no obvious pattern. For full scale mappings, see Appendix A.

4.2.2 Clustering by algorithms

The second approach for evaluating the feature vectors was to use clustering algorithms and see if they created interesting clustering. The clusterings where visualised with t-SNE plots and heat maps before they were presented to SMEs for evaluation; a selection of the most interesting of these clusterings are presented in Figure 4.5. Among the clusterings in Figure 4.5, there is a variation of clusterings with 2, 14, and 15 clusters. The reason why the clustering algorithms performed better with so many clusters as 15, which was the maximum numbers of clusters tested, is presumably that the cluster scoring methods resulted in higher scores if outliers was classified as a cluster of its own. In Figure 4.5 image 1 and 2, the t-SNE mapping creates distinct well separated clusters. In both of these cases does GMM assign the majority of the data to cluster 0 and only 20 to 50 data points in to cluster 1. The images 7, 8 and 9 uses standardisation as preprocessing instead of Robust scaling as in images 1 and 2, which does not result in equally well separated clusters. The heat map images 11 and 12 show clearly which features that are characteristic for the different clusters. This is presented as a white or gray vertical line that ends when the cluster switches, see Figure 4.5.

4.3 Evaluation by Subject matter experts

Ten Subject matter experts (SMEs) was emailed a form, with a preface explaining the context of the form, an explanation of what the figures displayed, along with the figures presented in Section **??**, and what feature or attribute that is unique for each cluster. Due to classification restrictions the features and attributes can not be presented in this report, but are referred to by aliases. The experts was asked to rate the figures usefulness from 1 (not useful) to 5 (useful), where 3 should be considered as *might be useful*.

From these ten emails seven replied in total. Two filled in the form, three replied only with an answer without filling in the form, and two asked instead for a quick meeting where they gave their verbal feedback.

All experts rated one or more images with 4, see Table 4.1, or replied with ”The impact and usefulness of this method and this information is great.”, ”I think the work you’ve done is interesting.”, ”I could see where it might be useful.”, ”I think your work is interesting.”, or ”Your ideas are very interesting, and I’m sure this is very useful.”. Some experts who did not fill in the form mentioned examples of images they thought of as most relevant. These images are attribute 1, 2, and 3 in Figure 4.4, and images 3 and 9 in Figure 4.5. The opinion of how useful an image is and which images are the most useful varies between the SMEs.

4.4 Choosing parameters and clustering methods

The evaluation of how many clusters gave the best result, resulted in using 2, 14

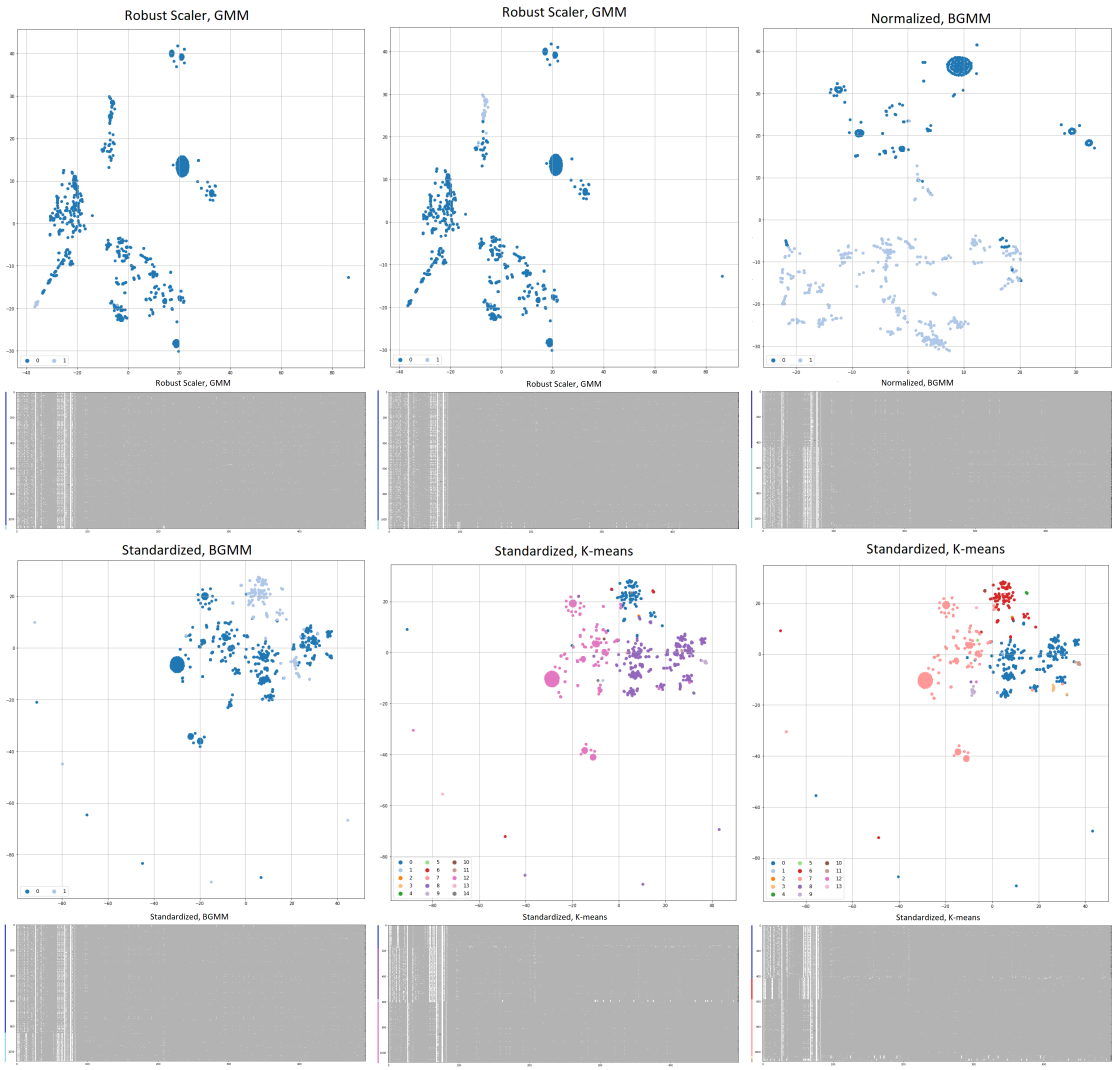

Figure 4.5: A selection of interesting clusterings, the images will be referred to as 1 to 12, counting left to right and top to bottom. Each clustering is shown with a t-SNE plot and its associated heat map directly below. For full scale images, see Appendix A. In image 1 and 2 the t-SNE mapping creates distinct well separated clusters. In both of these cases does GMM assign the majority of the data to cluster 0 and only 20 to 50 data points in to cluster 1. The images 7, 8 and 9 uses standardisation as preprocessing instead of Robustscaling as in images 1 and 2, which does not result in equally well separated clusters. The heat map images 11 and 12 show clearly which features that are characteristic for the different clusters. This is presented as a white or gray vertical line that ends when the cluster switches.

or 15 clusters. This is shown in Figure 4.6, where the highest score of the cluster indices is achieved at the endpoints. There are some strong indications that 3, 4, 5 or 7 clusters might yield good results as well, but this was not the case, as shown in Figure 4.7.

Due to that some of the best evaluation scores was yielded for the maximum numbers of clusters tried, a few runs with more clusters was tried as well, see Figure 4.8, but

Table 4.1: The usefulness of each image according to the two experts who answered the form. The rating is from 1 (not useful) to 5 (useful), where 3 should be considered as *might be useful*, and "-" marks where no answer where given. Both experts rate at least a few of the images with 4. The opinion of how useful an image is and which images are the most useful varies between the SMEs

Figure 4.4 \ SME	0	1
Attribute 1	4	3
Attribute 2	3	2
Attribute 3	3	3
Attribute 4	3	5
Attribute 5	2	4
Attribute 6	2	4
Figure A.2	-	3
Figure A.4	3	2
Figure A.10	3	1
Figure A.11	3	4
Figure A.7	4	1

Figure 4.5 \ SME	0	1
Image 1	2	4
Image 2	4	4
Image 3	-	1
Image 4	-	-
Image 5	-	-
Image 6	-	1
Image 7	2	1
Image 8	-	2
Image 9	3	2
Image 10	-	1
Image 11	-	1
Image 12	-	1

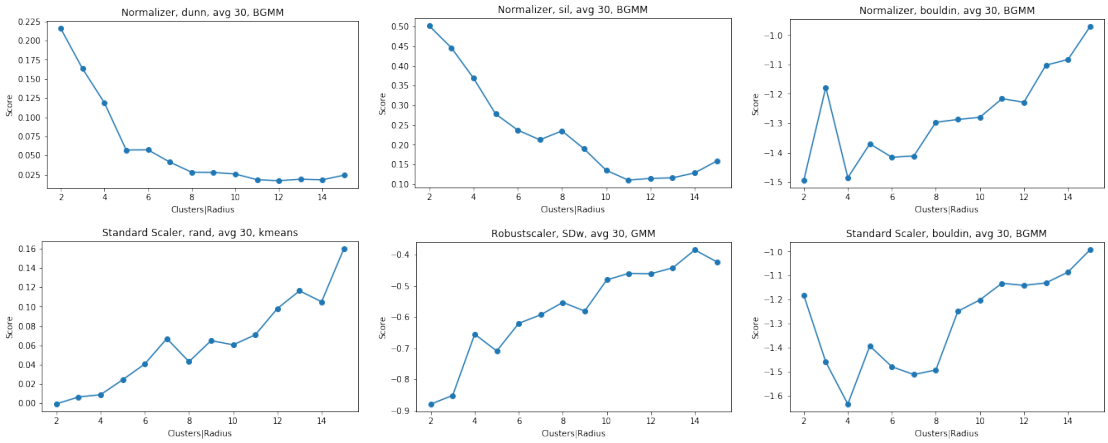

Figure 4.6: The images will be referred to as 1 to 6 counting left to right and top to bottom. Evaluation scores from different scoring functions used for selecting the number of clusters for the clustering algorithms. The number of clusters that received the highest scores are 2, 14 and 15, where 15 is the highest number of clusters tested. There are also some peaks for 3, 4, 5 and 7 clusters.

this yielded similar clusterings as with 15 clusters.

In addition to K-means, GMM, and BGMM the clustering algorithms Spectral clustering and DBSCAN was also tested in the thesis but did not perform well, and was therefore not used in the later parts. Spectral clustering have the problem of entangling clusters, and DBSCAN either classified too many data points as noise, or resulted in bad clusters, see Figure 4.9.

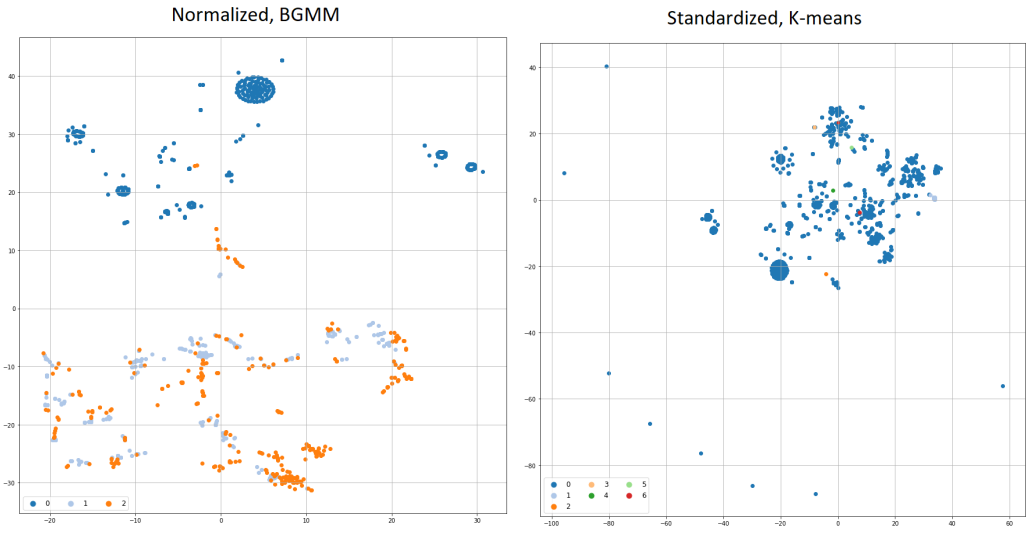

Figure 4.7: Two examples of bad clusterings due to selection of hyperparameters. The left image have cluster 1 and 2 entangled, and the right image have one large cluster, that is cluster 0, and six clusters only including a very few numbers of data points.

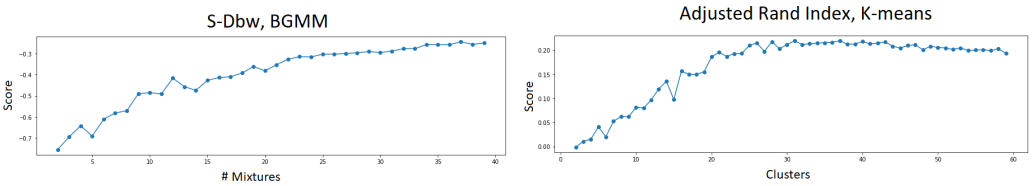

Figure 4.8: Evaluation of clusterings with more than 15 clusters. No clear peak is visible, which makes it unclear how many clusters would result in the best clustering.

4.5 Evaluation by metrics

Evaluating the clusterings with internal clustering metrics resulted in that attribute 5 recived comparatively high scores compared to the other attributes, as shown in Table 4.2. However the attributes receives generally lower scores than the clusterings created by algorithm. The clustering that over all gets the best scores is the clustering in Figure 4.5, Image 1, which receives the best scores in two out of four indexes and is close to the highest score on a third, see Table 4.2.

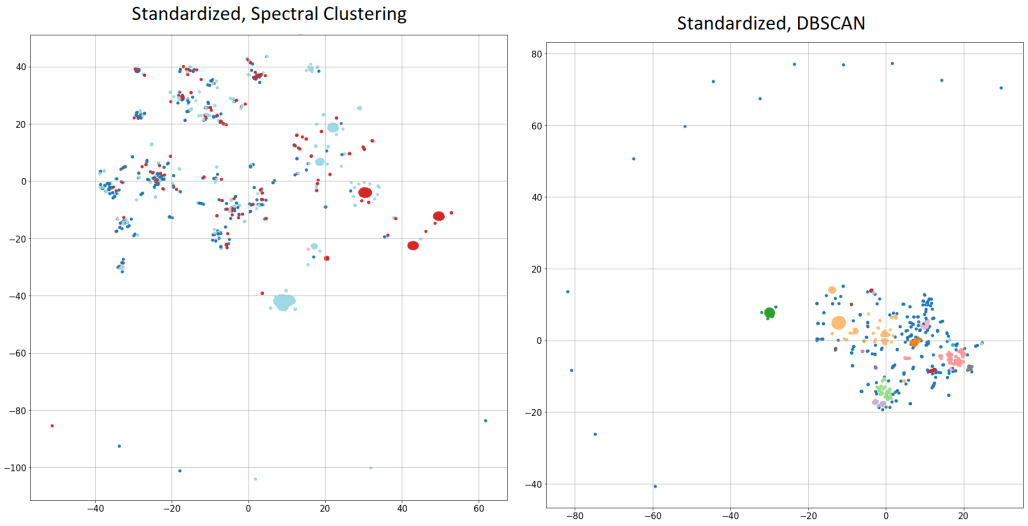

Figure 4.9: Two examples of bad clusterings due to clustering algorithm. The left image using Spectral clustering have entangled the clusters. The right image using DBSCAN have a bit to many data points clustered as noise. The noise clas is clored with blue.

Table 4.2: Clusterings that received the highest/lowest score by an index is coloured in green/blue. Attribute 5 receives the highest scores in 3 out of 4 clustering indexes, when comparing to other attributes. The clustering that seems to perform best from Figure 4.5 is image 1, which receives the highest scores in 2 out of 4 clustering indexes, and is close to the top score in a third index as well. The signs of the S-Dbw index and the Davies–Bouldin index are inverted, so that a higher index indicates a better clustering, same as the other indexes.

Figure 4.4 \ Index	Silhouette	- S-Dbw	Dunn	- Davies Bouldin
Attribute 1	0.14	-1.42	0.002	-6.94
Attribute 2	0.06	-0.91	0.024	-3.35
Attribute 3	-0.15	-0.84	0.0	-3.87
Attribute 4	-0.15	-1.51	0.002	-3.9
Attribute 5	0.27	-0.97	0.053	-1.81
Attribute 6	0.2	-1.16	0.018	-5.08
Figure 4.5 \ Index	Silhouette	- S-Dbw	Dunn	- Davies Bouldin
Image 1	0.85	-0.67	0.08	-0.23
Image 2	0.88	-1.22	0.01	-0.47
Image 3	0.58	-2.03	0.006	-0.6
Image 7	0.28	-1.15	0.018	-3.88
Image 8	0.14	-0.27	0.024	-1.07
Image 9	0.16	-0.38	0.025	-1.07

5
Conclusion

5.1 Discussion

5.1.1 Feature vector

The feature vector is constructed by only including features recommended by SMEs. This resulted in that only 22 attributes was used from a pool of 10 000 attributes. It is highly probable that some attributes, now left out, would be improving the feature vector and making it more useful. However, the approach used have high interpretability and was quick to implement, and is therefore suitable as a first evaluation of whether a useful feature vector can be created.

Based on the images in Figure 4.4, it is clear due to the distinguishable clusters that the feature vector is intricate enough to capture some of the most important attributes of a base station. The sufficient intricacy is further confirmed by the clusterings in Figure 4.5, image 1, 2, 3 and 9, which captures traits of the base stations that some SMEs consider *probably useful*. This does however not mean that a more intricate feature vector would be worse. I believe that a feature vector created with deep learning methods would outperform the feature vector presented in this thesis, due to the ability of deep learning methods to capture complex structures [3].

5.1.2 Usefulness

For determining the usefulness of the feature vector, SMEs where asked to rate the usefulness of the different visualisations of the feature vector. From their replies, see Section 4.3 it is clear that what is considered useful varies. This can be due to that the SMEs are from different areas within Ericsson, have different knowledge, and think of different purposes the feature vector can be used for. Another factor that might affect the answers is how well the different SMEs understood the images. An example of this is Figure A.7 which takes some time to analyse but are according to some SMEs interesting and probably useful, but was according to others considered as noise. Most of the SMEs had little time to spare which resulted in that not all SMEs took the time to inspect every image carefully.

When determining if the feature vector is useful, it is okay that it is not useful for all departments. It must how ever be useful for some persons, in some departments, to be considered useful. This might not be the SME, even if the SME have the best knowledge of what is important. For example, it is mentioned in the introduction

that a feature vector might be of use for the not so experienced worker. He or she could for example cluster by an attribute and see which patterns appear, to learn the relations that the SME already know.

The highest ratings from the two SMEs that filled in the form are 4 respectively 5, and our interpretation of the replies presented in Section 4.3, is that they range from 3 to 5. Thus, it is reasonable to conclude that the feature vector is useful. It might seem strange to use subjective opinions to determine whether the feature vector is useful or not, but according to J. Handl et al. [12], it is common with manual evaluation of clusterings. This is because it is hard or impossible to mathematically describe what is useful even though a expert knows what is useful.

5.1.2.1 Evaluation metrics

To make it easier to compare the feature vector with future feature vectors, and as a way to evaluate the feature vector more objectively; the clusterings were evaluated with internal clustering metrics. The metrics are well defined and makes it easier to say which clustering is the better and by how much. The problem is that what we as humans consider the best cluster or the most useful cluster, do not have to receive the highest scores. An example of this is the clustering of attribute 3, which is a relatively clean pattern, receive the lowest scores from both the Silhouette index and the Dunn index.

However, if a new feature vector where created, and it received generally higher scores, and found interesting and useful patterns, that would indicate that the data points in the new feature space are more clustered and that the new feature vector probably is better. So the clustering evaluation scores are interesting to present for comparative purposes, even if they favors simplicity, which can result in that useless clusterings gets high scores.

5.2 Conclusion

In this thesis we created a useful feature vector, and thereby proven that a base station can be represented by a feature vector in a useful manner. The feature vector, containing 494 features, captures interesting and important attributes of the base station in useful patterns. The visualisation of the feature vectors results in distinct clusters, and well separated clusters for some preprocessing methods. However, most of the clusterings made with the feature vector in this thesis are not considered useful, which indicates that further improvements of the feature vector would be desirable.

Further, we evaluated the feature vector with internal clustering indexes. Those results can not be used for determining the usefulness of the feature vector, as is argued in the discussion, but the scores can be useful for future works for comparative purposes.

Future work could include regularisation of the feature vector to only use the most relevant features in the current feature vector. This reduction could be made by implementing a dimension reduction technique with correlation filters and low variance filters. It would also be interesting to allow all the attributes in the data set to be used and apply machine learning algorithms for creating the feature vector. Lastly, it could also be interesting to try hierarchical clustering methods such as Agglomerative clustering, which was done in the very end of the project, but too late in the project to include it in this report.

To conclude, it is possible to capture a complex system such as a base station in a useful manner with a feature vector. However, there are still room for improvements on such a vector.

Bibliography

[1] S. Idreos, O. Papaemmanouil, and S. Chaudhuri, "Overview of data exploration techniques," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 277–281, 2015.

[2] M. F. De Oliveira and H. Levkowitz, "From visual data exploration to visual data mining: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 378–394, 2003.

[3] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[4] M. Wu, M. C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez, "Beyond sparsity: Tree regularization of deep models for interpretability," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[5] S. Chakraborty, R. Tomsett, R. Raghavendra, D. Harborne, M. Alzantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, R. M. Rao, *et al.*, "Interpretability of deep learning models: a survey of results," in *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pp. 1–6, IEEE, 2017.

[6] P. Bodik, M. Goldszmidt, A. Fox, D. B. Woodard, and H. Andersen, "Fingerprinting the datacenter: automated classification of performance crises," in *Proceedings of the 5th European conference on Computer systems*, pp. 111–124, 2010.

[7] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," in *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*, pp. 207–218, IEEE, 2016.

[8] M. Fontes and C. Soneson, "The projection score-an evaluation criterion for variable subset selection in pca visualization," *BMC bioinformatics*, vol. 12, no. 1, p. 307, 2011.

[9] J. Tang, J. Liu, M. Zhang, and Q. Mei, "Visualizing large-scale and high-dimensional data," in *Proceedings of the 25th international conference on world wide web*, pp. 287–297, 2016.

[10] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*, pp. 25–71, Springer, 2006.

[11] M. Halkidi and M. Vazirgiannis, "Clustering validity assessment: Finding the optimal partitioning of a data set," in *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 187–194, IEEE, 2001.

[12] J. Handl, J. Knowles, and D. B. Kell, "Computational cluster validation in post-genomic data analysis," *Bioinformatics*, vol. 21, no. 15, pp. 3201–3212, 2005.

[13] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[14] D. A. Reynolds, "Gaussian mixture models.," *Encyclopedia of biometrics*, vol. 741, 2009.

[15] "Scikit-learn: Variational bayesian gaussian mixture." `https://scikit-learn.org/stable/modules/mixture.html#bgmm`. Accessed: 2020-05-23.

[16] H. Attias, "Inferring parameters and structure of latent variable models by variational bayes," *arXiv preprint arXiv:1301.6676*, 2013.

[17] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *Kdd*, vol. 96, pp. 226–231, 1996.

[18] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[19] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[20] I. B. Mohamad and D. Usman, "Standardization and its effects on k-means clustering algorithm," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 6, no. 17, pp. 3299–3303, 2013.

[21] M. van de Velden, A. Iodice D'Enza, and A. Markos, "Distance-based clustering of mixed data," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 11, no. 3, p. e1456, 2019.

[22] S. E. Wilson, "Methods for clustering data with missing values," *url: https://www. math. leidenuniv. nl/scripties/MasterWilson. pdf (visited on 11/02/2016)*, 2015.

[23] A. N. Baraldi and C. K. Enders, "An introduction to modern missing data analyses," *Journal of school psychology*, vol. 48, no. 1, pp. 5–37, 2010.

[24] D. Lam, M. Wei, and D. Wunsch, "Clustering data of mixed categorical and numerical type with unsupervised feature learning," *IEEE Access*, vol. 3, pp. 1605–1613, 2015.

[25] M. J. Warrens, "On the equivalence of cohen's kappa and the hubert-arabie adjusted rand index," *Journal of classification*, vol. 25, no. 2, pp. 177–183, 2008.

[26] J. M. Santos and M. Embrechts, "On the use of the adjusted rand index as a metric for evaluating supervised classification," in *International conference on artificial neural networks*, pp. 175–184, Springer, 2009.

[27] B. Desgraupes, "Clustering indices," *University of Paris Ouest-Lab Modal'X*, vol. 1, p. 34, 2013.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn:

Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] M. Z. Rodriguez, C. H. Comin, D. Casanova, O. M. Bruno, D. R. Amancio, L. d. F. Costa, and F. A. Rodrigues, "Clustering algorithms: A comparative approach," *PloS one*, vol. 14, no. 1, 2019.

A

Appendix

A.1 Figures clustered by attributes

In this section we present figures clustered by attributes.

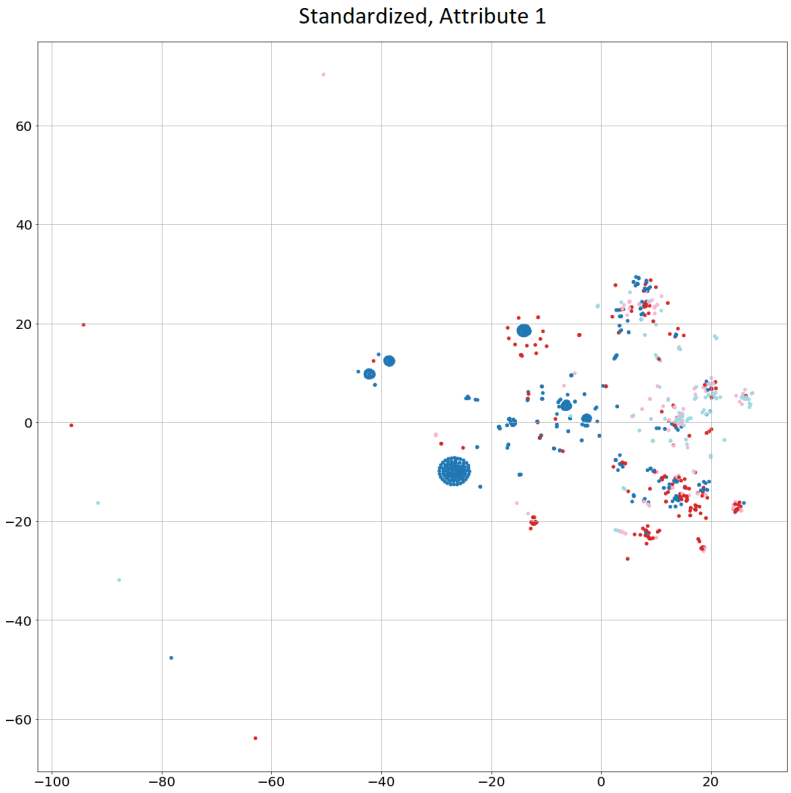

Figure A.1: t-SNE plot of attribute 1.

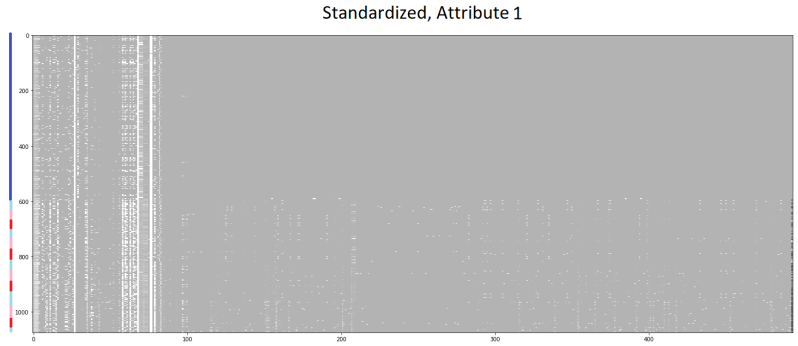

Figure A.2: Due to no shifts in the lower parts of this figure, we can not easily say where one cluster ends and the other begins. Therefor have we choosen to stripe the bar on the left side of the heat map as an indication of this.

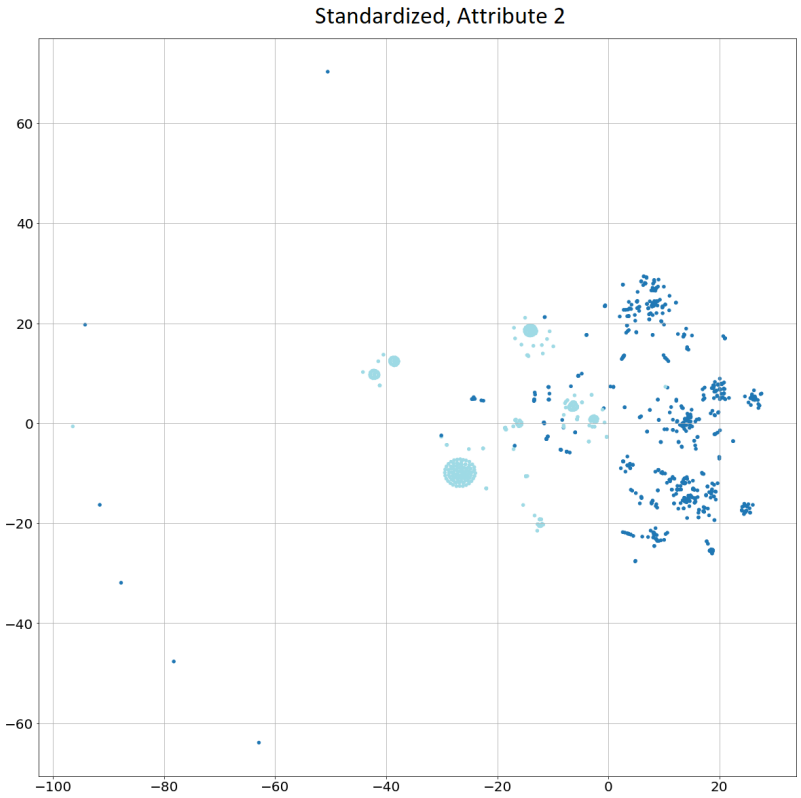

Figure A.3: t-SNE plot of attribute 2.

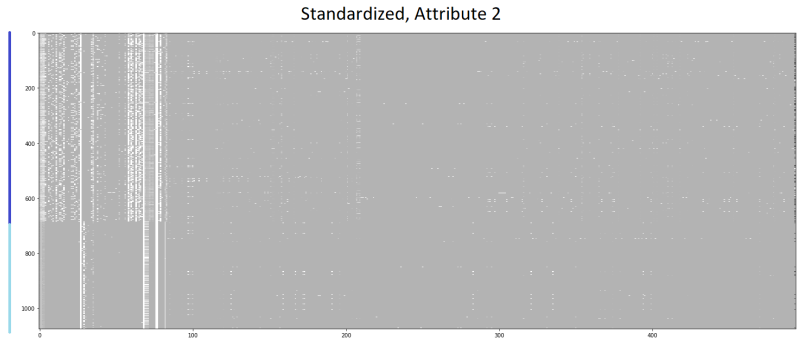

Figure A.4: Heat map of attribute 2.

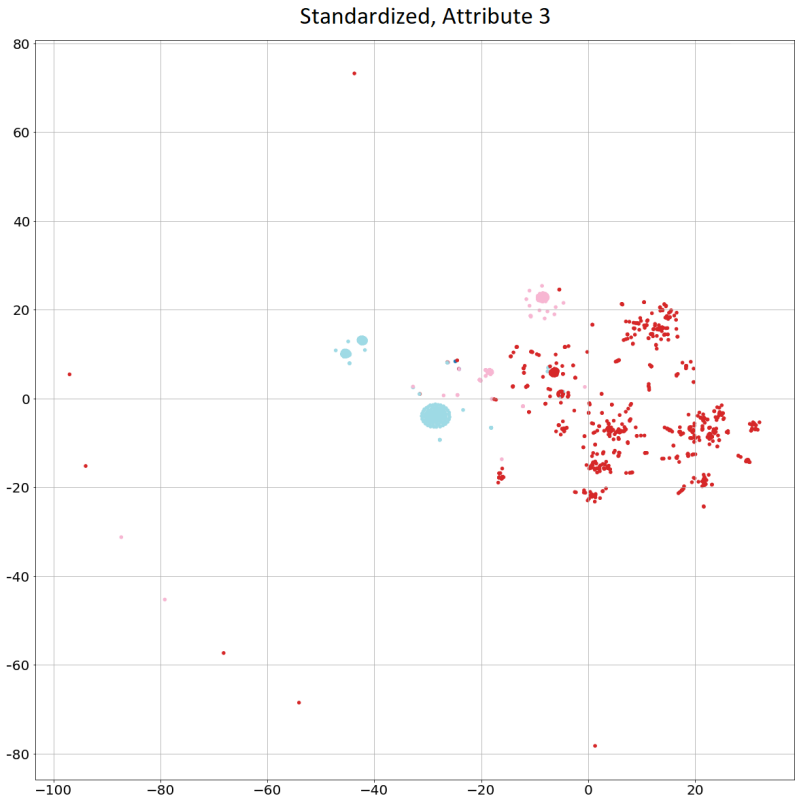

Figure A.5: t-SNE plot of attribute 3.

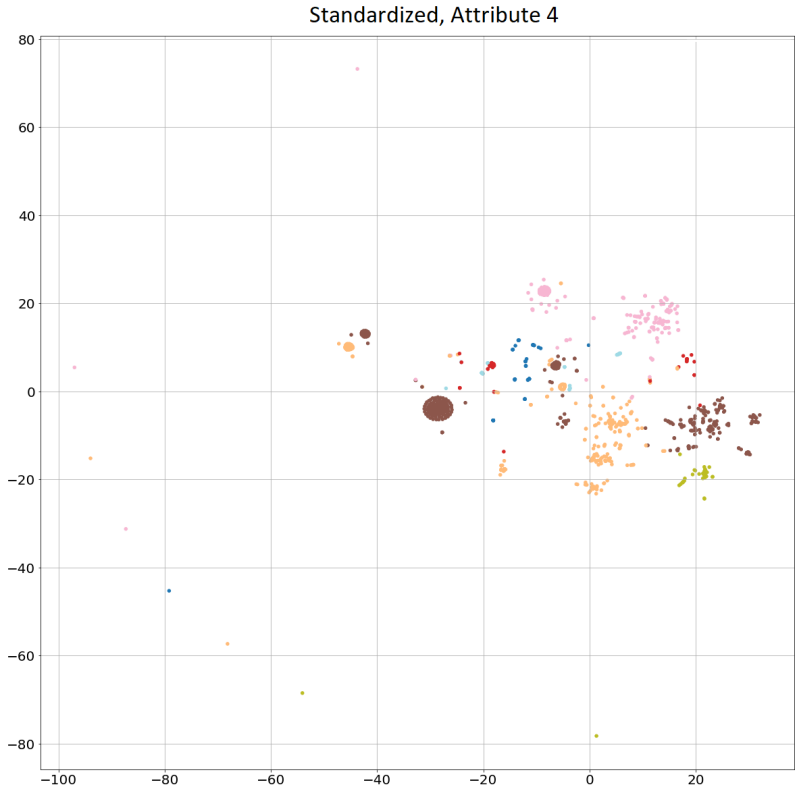

Figure A.6: t-SNE plot of attribute 4.

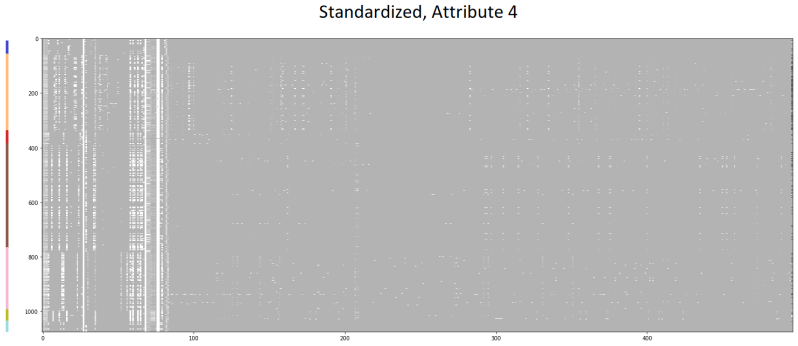

Figure A.7: Heat map of attribute 4.

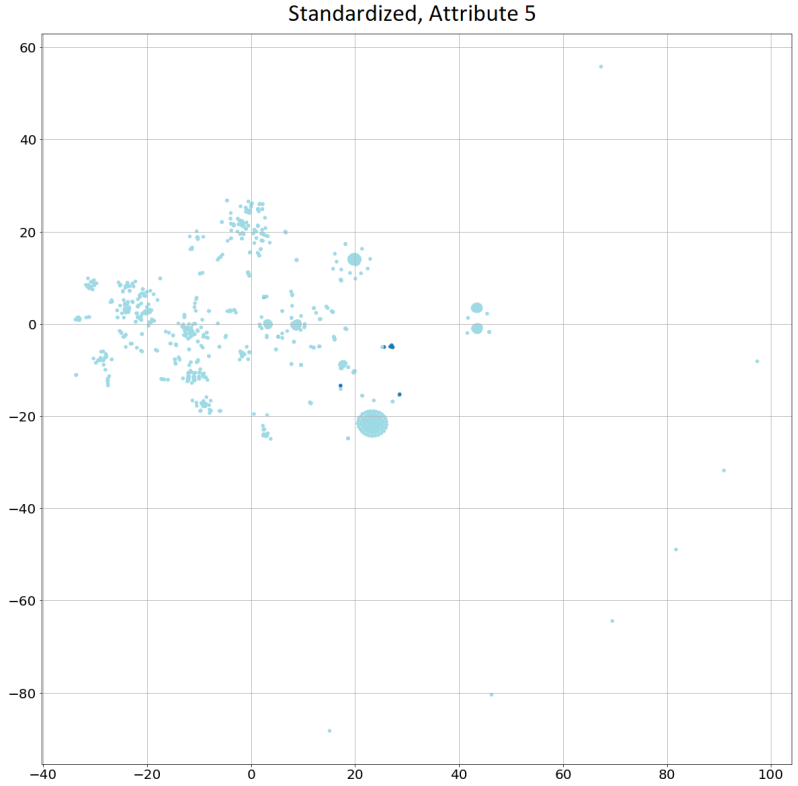

Figure A.8: t-SNE plot of attribute 5.

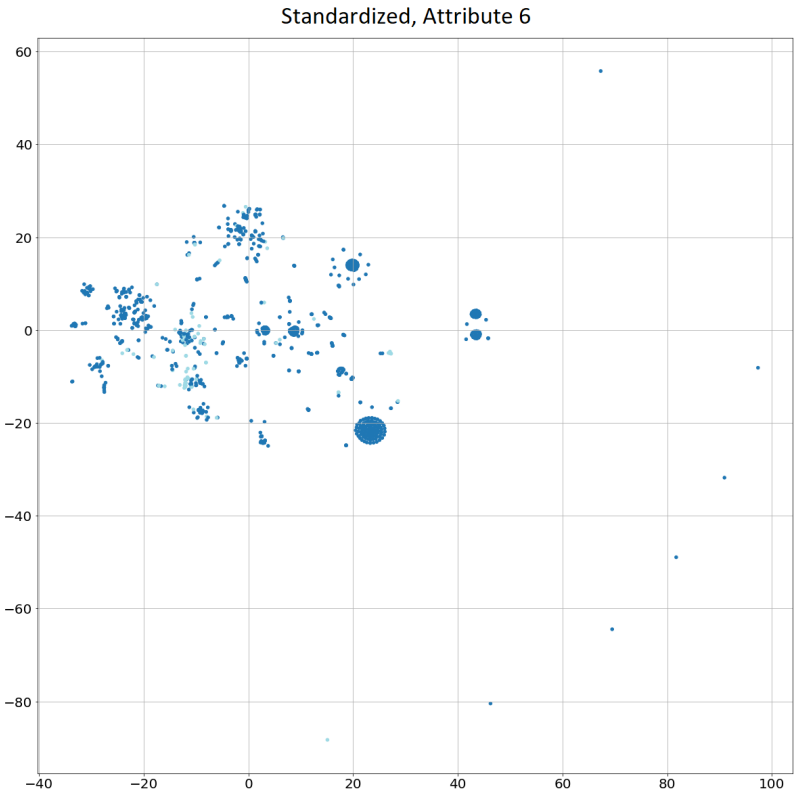

Figure A.9: t-SNE plot of attribute 6.

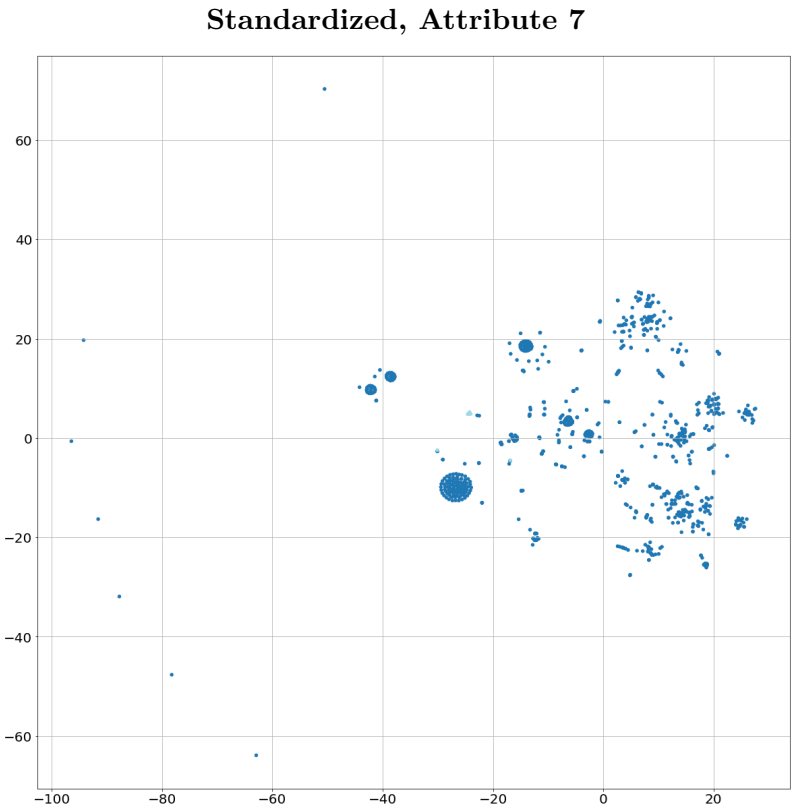

Figure A.10: t-SNE plot of attribute 7.

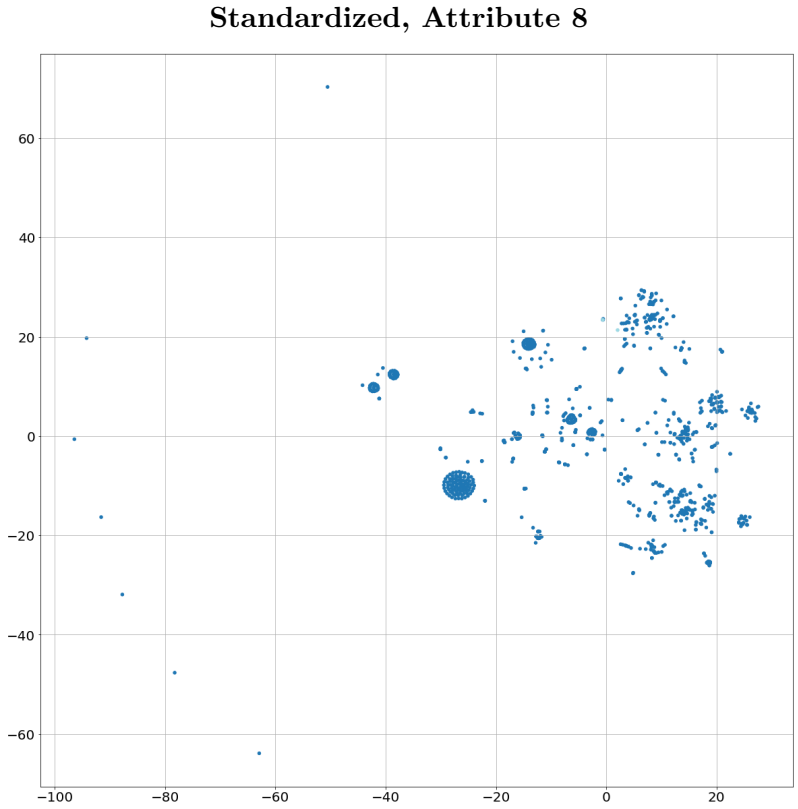

Figure A.11: t-SNE plot of attribute 8.

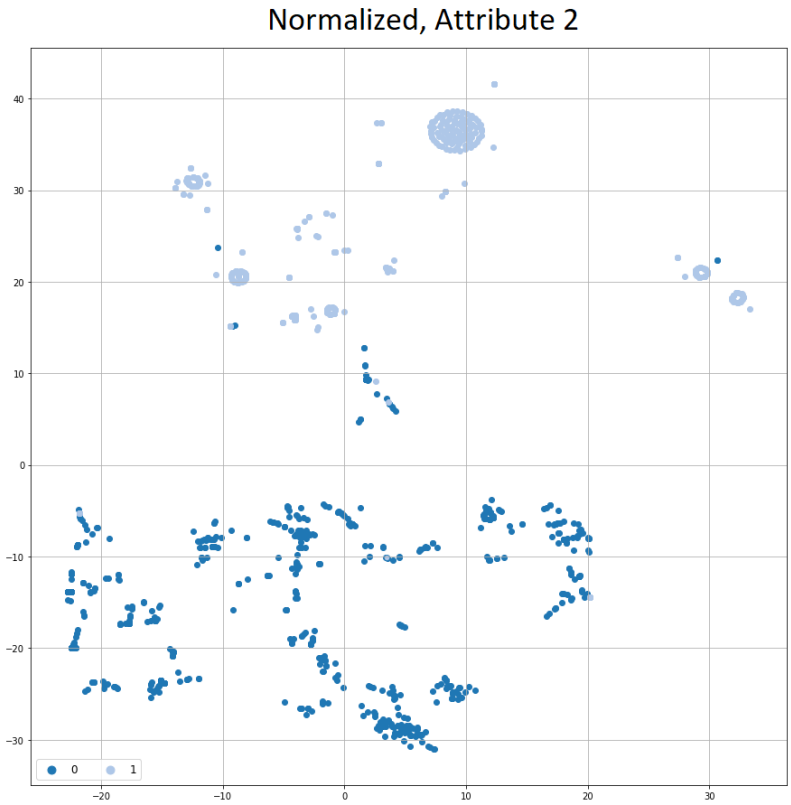

Figure A.12: t-SNE plot of attribute 2.

A.2 Figures clustered by clustering algorithm

In this section we present figures clustered by clustering algorithms.

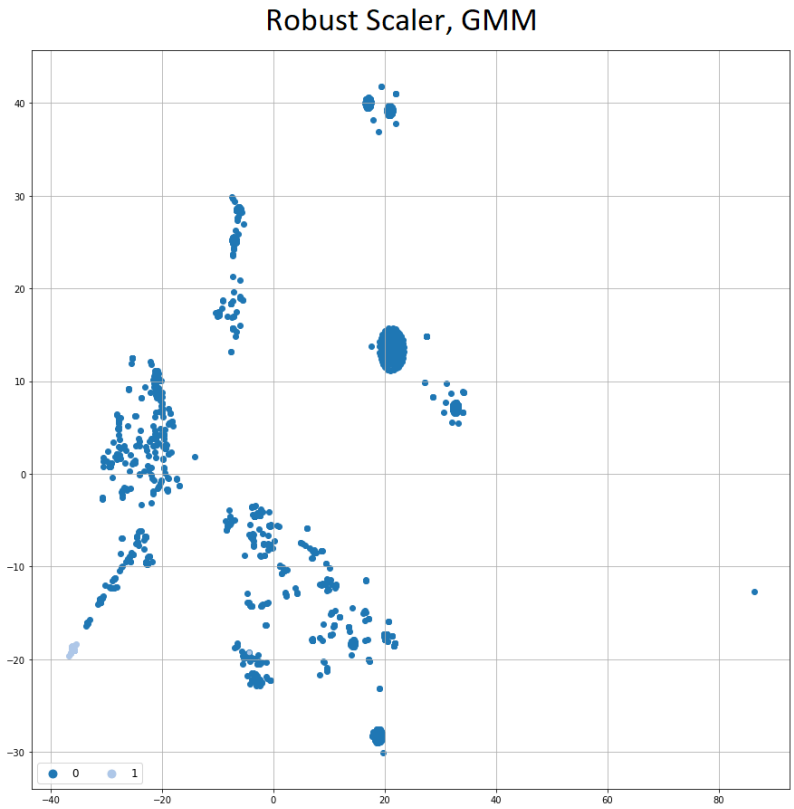

Figure A.13: t-SNE plot of the result from GMM.

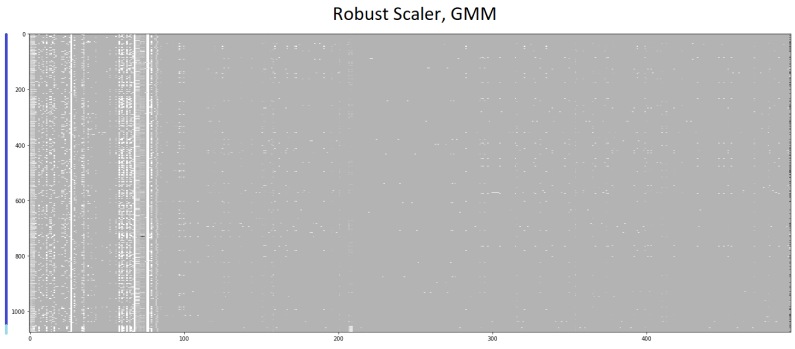

Figure A.14: Heat map of GMM clustering.

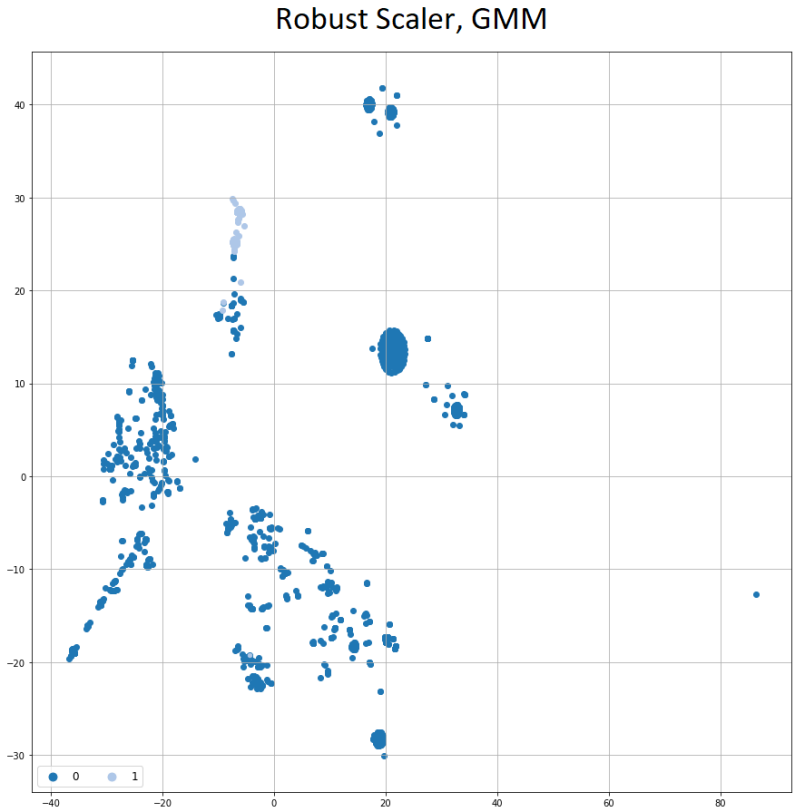

Figure A.15: t-SNE plot of the result from GMM.

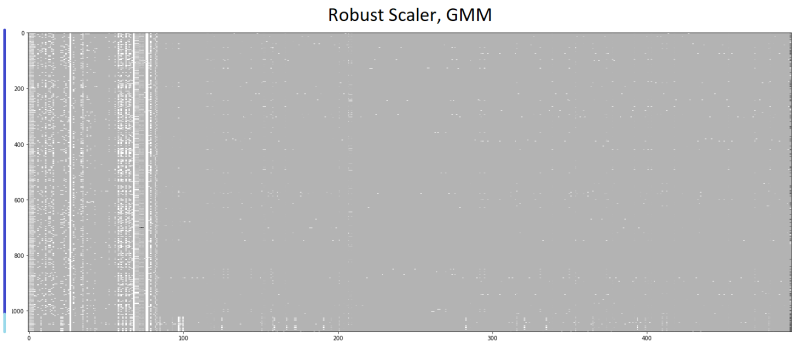

Figure A.16: Heat map of GMM clustering.

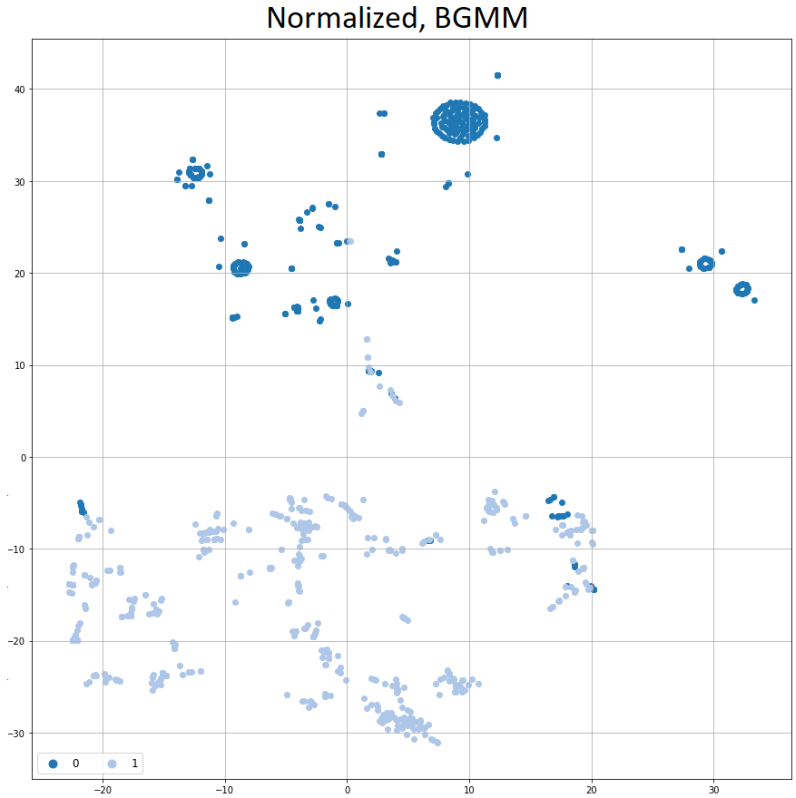

Figure A.17: t-SNE plot of the result from BGMM.

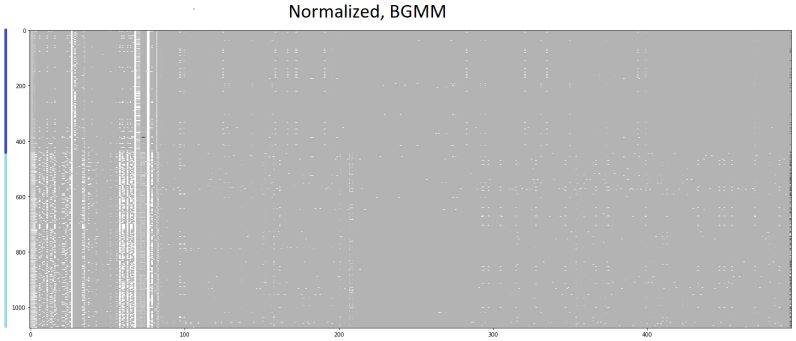

Figure A.18: Heat map of BGMM clustering.

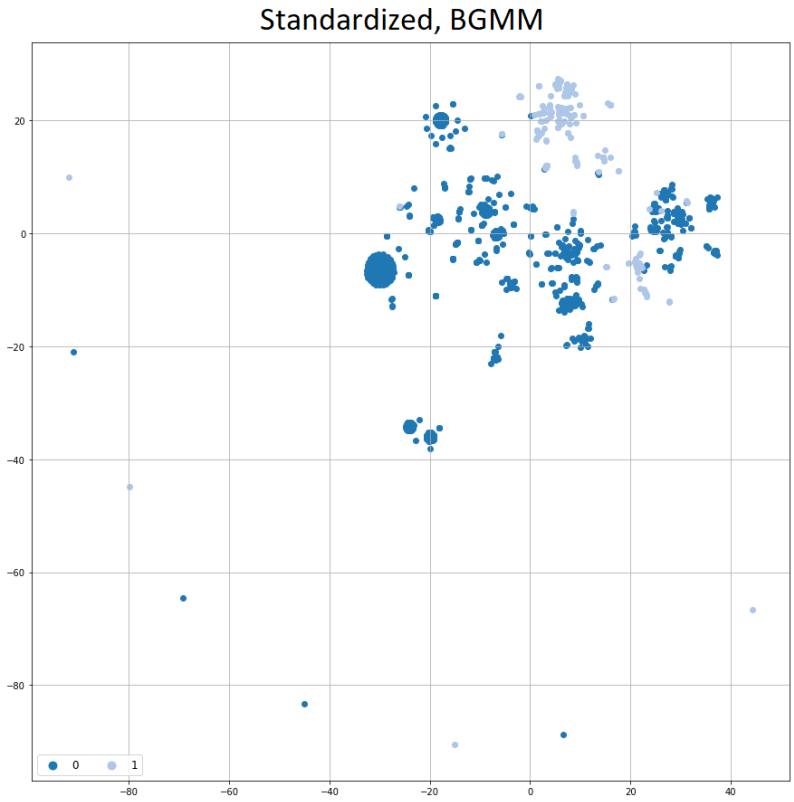

Figure A.19: t-SNE plot of the result from BGMM.

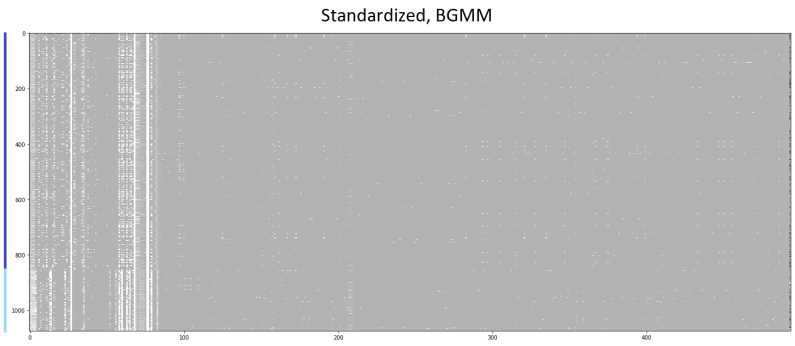

Figure A.20: Heat map of BGMM clustering.

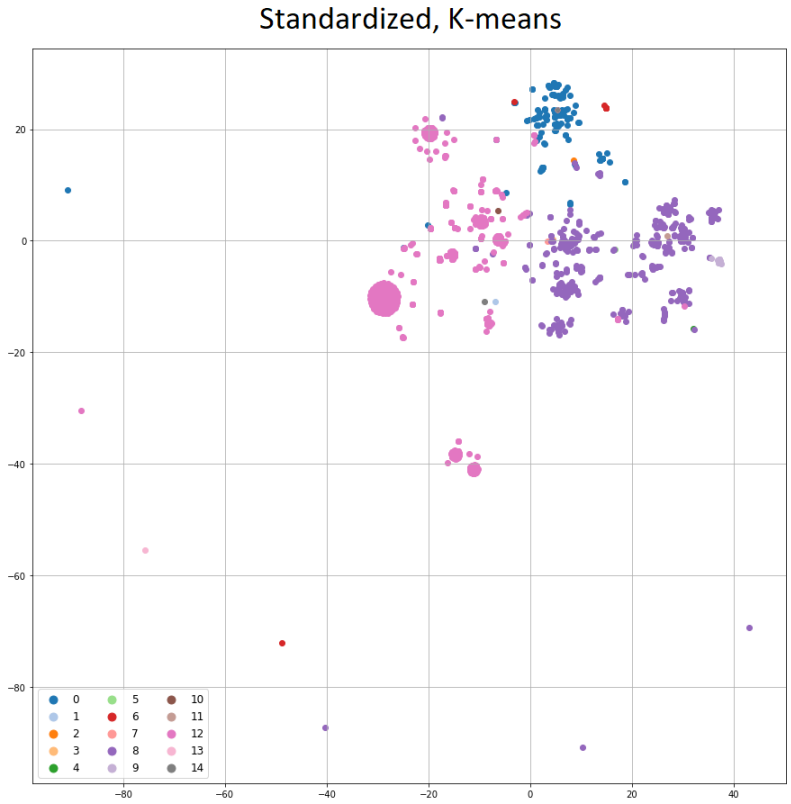

Figure A.21: t-SNE plot of the result from k-means.

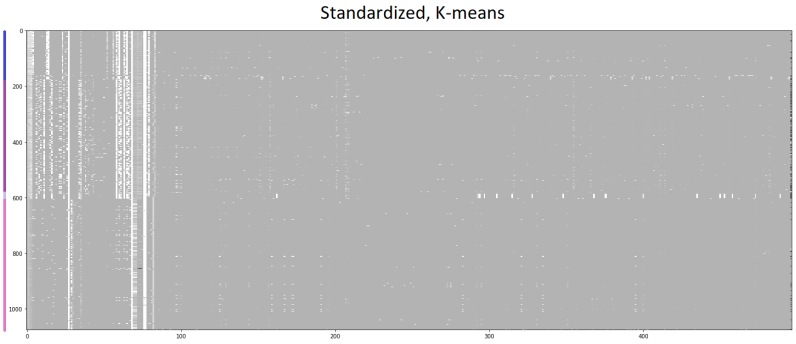

Figure A.22: Heat map of k-means clustering.

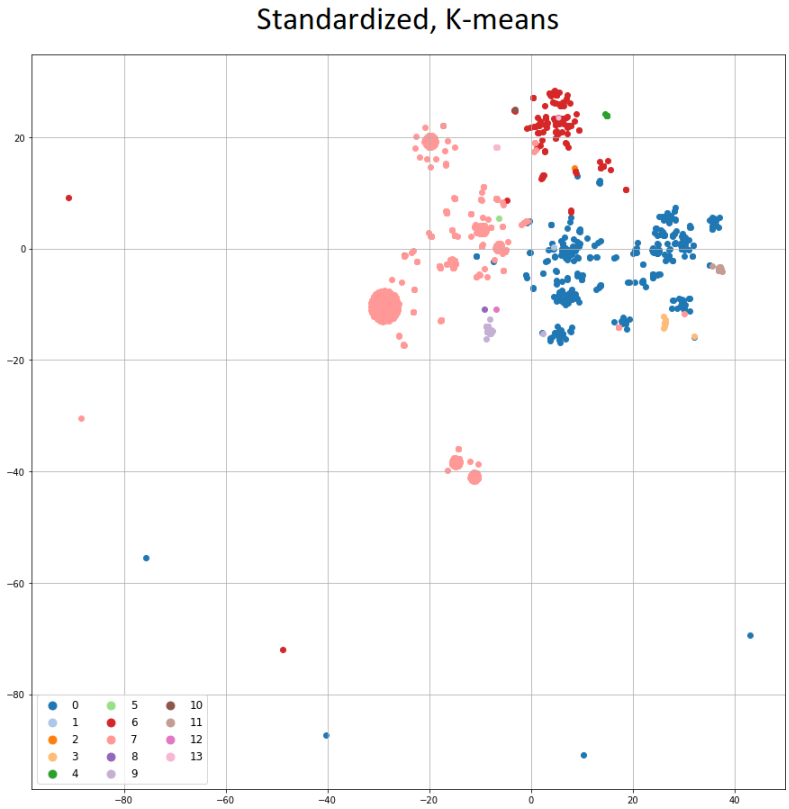

Figure A.23: t-SNE plot of the result from k-means.

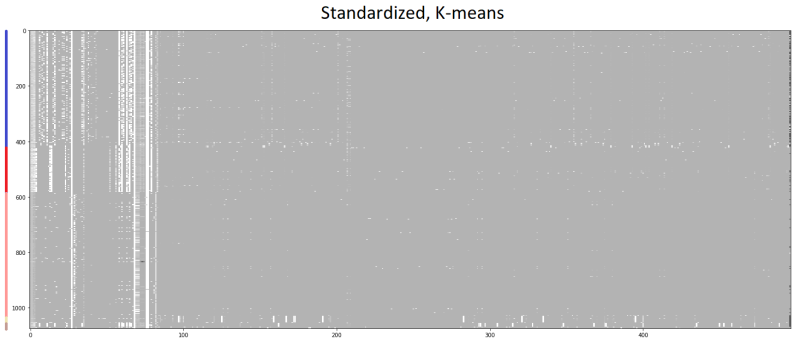

Figure A.24: Heat map of k-means clustering.