KTH Teknikvetenskap

Predicting data traffic in cellular data networks

MARIANNE FJELBERG

Master Thesis

Department of Mathematics
The Royal Institute of Technology (KTH)
Stockholm, Sweden

Supervisor: Jimmy Olsson
Examiner: Jimmy Olsson

June 08, 2015

Abstract

The exponential increase in cellular data usage in recent time is evident, which introduces challenges and opportunities for the telecom industry. From a Radio Resource Management perspective, it is therefore most valuable to be able to predict future events such as user load. The objective of this thesis is thus to investigate whether one can predict such future events based on information available in a base station. This is done by clustering data obtained from a simulated 4G network using Gaussian Mixture Models. Based on this, an evaluation based on the cluster signatures is performed, where heavy-load users seem to be identified. Furthermore, other evaluations on other temporal aspects tied to the clusters and cluster transitions is performed. Secondly, supervised classification using Random Forest is performed, in order to investigate whether prediction of these cluster labels is possible. High accuracies for most of these classifications are obtained, suggesting that prediction based on these methods can be made.

Acknowledgements

I would like to extend my greatest gratitude to Rickard Cöster at Ericsson, who has shown enthusiasm, support and guidance throughout this thesis, and dedicated time to provide valuable input despite his busy schedule. I would also like to thank my supervisor at KTH, Jimmy Olsson, for his encouragement and for supervising me in this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The exponential increase in cellular data usage due to the market introduction of smartphones is evident, and research suggests the increase will continue its pace. In the Ericsson Mobility Report of June 2015 [1], it is presented that, worldwide, smartphone subscriptions will have more than doubled by 2020, where it is expected that 70% of the world's population will have a smart phone. In addition to this, it is expected that mobile video traffic, which is bandwidth demanding, will grow by approximately 55% per year from 2014-2020, reaching around 60% of all mobile data traffic by the end of that period. This considerable growth in data traffic entails challenges for the telecom industry, but also immense possibilities, where the informational data obtained from the network service and usage can be used for useful analysis and prediction of future events related to user behaviour. Having this knowledge allows for applications related to i.e. radio resource management (RRM), which is a topic for future work. RRM is a system for controlling certain parameters such as transmit power, user allocation and handover criteria, and RRM algorithm adaption typically includes resource optimisation and end-user performance improvements,. A brief overview of RRM is given in [2].

1.2 Previous work

Previous work in this area has been done, of which many have used clustering techniques. In [3] an improvement was found when using clustering techniques for data of one dimension compared to generic means, suggesting clustering patterns can be found in this kind of data. A co-clustering algorithm was used in [4], where a characterisation of user behaviour in terms of browsing profiles in a network was made.

Other work has taken a more statistical approach to analysing traffic and user behaviour. In [5] an extensive statistical analysis aiming to construct statistical models for data traffic in single cells has been performed, where it was found that there are great differences with regards to throughput between cells in a network. Furthermore, in [6], an analysis on subscriber mobility and temporal activity patterns was performed, finding that a small fraction of users create the majority of daily traffic and are mostly sporadic.

Machine learning techniques have been considered as well. In [7] it was assumed that base stations loads are time sequences of multi-variate Gaussian random variables, and modelled as a Gaussian Markov Random Field. This assumption is later ensured, suggesting that such patterns can be found in this kind of data, and rationalising the use of Gaussian Mixture Models in this thesis.

1.3 Purpose

This thesis aims to investigate whether one can predict future events tied to, for instance, user load, based on information about a range of features available from a base station. To learn more about this, the question whether it is possible to find natural grouping or clusters of users within a specific cell is investigated. If the clustering succeeds in finding groups with similar network usage characteristics, RRM algorithms could be adapted to each cluster. This is done by utilising Gaussian Mixture Models to cluster observations connected to different users and evaluating the accuracy of the model parameters. With this information, one can learn about signatures tied to specific clusters with regard to a range of features. Another relevant aspect to this matter is to learn if we can predict how long a subscriber will stay in a certain cluster and what cluster the user will be in next, which is investigated with Markov Chain probabilities for the cluster transitions. Combining these insights with supervised classification, where cluster identity is used as predictor variable, enables prediction of user behaviour tied to these features. Therefore, an implementation of Random Forest classification is performed in order to evaluate this matter.

This thesis aims to complement [3] by using higher-dimensional data sets of multiple features, and using different and more advanced clustering techniques. As opposed to evaluating user behaviour in terms of browsing profiles in [4], user behaviour tied to network characteristics is evaluated in this work. The results from [5] suggest that machine learning techniques such as clustering would improve statistical modelling for each cell, as will be investigated in this thesis. Moreover, taking the findings of [6] into account, it is evident that it would be useful for the base station to be able to identify such heavy users fast in order to optimise resources, something a clustering algorithm combined with classification will recognise as we will see in this work. In [7], an aggregated aspect was investigated, considering total load in base stations and evaluating amongst them, whereas in this thesis individual users and their load within a cell are considered.

Although previous work has analysed user behaviour and network characteristics, most analyses has been performed on 2G or 3G networks [3, 4, 5, 6, 7], either simulated or live networks. In this work, the analysis will be on a simulated 4G network, a network which is expected to grow to serve 3.7 billion of the world's subscriptions by 2020, accounting for 85% and 90% of all mobile subscriptions in Western Europe and North America respectively [1].

Chapter 2

Preliminaries

2.1 Cellular data network

A cellular data network is provided by transceivers in form of base stations, placed either at the ground or on buildings, each covering a land area. This land area is divided into cells, typically three, where the bandwidths are on different frequencies as to avoid interference between the cells.

In a LTE[*], or 4G, network, the fastest mobile broadband commercially available today, these cells provide radio coverage directly to portable transceivers such as mobile phones via radio waves. The network consists of the following elements:

- An *ENode B*, antennas and radio base stations.
- A *transport network* with microwave links, optical fibers and IP routers.
- A *gateway*, a connection to the internet and IP networks.
- A *mobility management entity (MME)*, a controller to manage mobility.
- A *home subscriber server database (HSS)*, contains information about all individual subscriptions such as a 4G plan.
- A *policy management system*, ensures that the policies the user has subscribed to are delivered accordingly.
- An *IP multimedia subsystem*, handles voice over LTE and other multimedia services.

Briefly explained, the network works in the following way: firstly, the MME establishes connections between a mobile device and an ENodeB base station while controlling the signalling with the terminal. All activity sent and received, such as

[*]Long Term Evolution

sending an email or watching a video, is done in so called IP packets, from the mobile device to the ENodeB base stations. From the ENodeB base station, the packages are sent over the transport network to the gateway, where a serving gateway routes and pushes the data forward while maintaining the connection between the device and the network. The packet data network gateway receives the packages, looks at the corresponding destinations, and based on that information sends the data on the right track. During this process, the policy management system counts the data packets and applies the policy rules found in the HHS. The packages are then sent through the internet and other IP networks back to the device. A schematic illustration of this procedure is found in Figure 2.1.

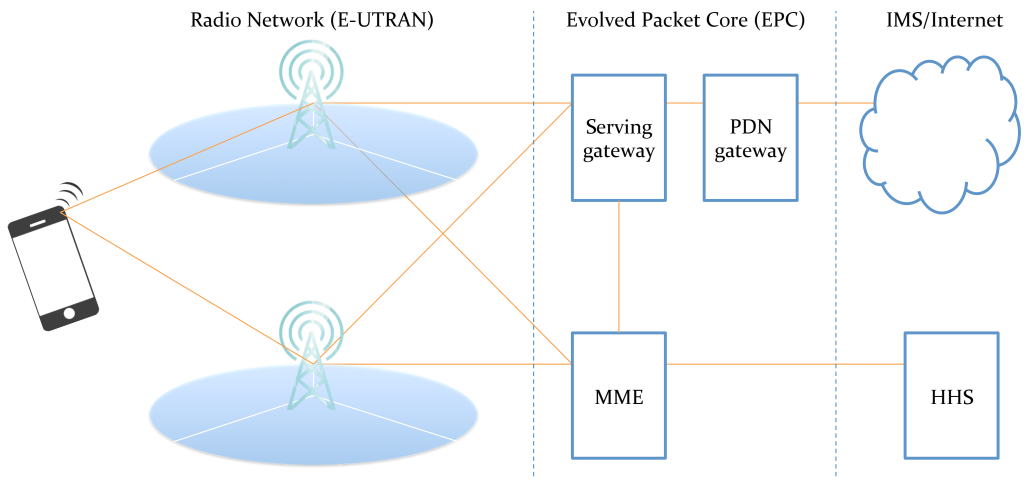


Figure 2.1: Schematic illustration of cellular network.

When providing network, the ENodeB base stations log data from users currently utilising the network in one of its cells. Those that are used in this thesis are explained in Section 2.1.1.

2.1.1 Notations

Handover

When a phone user moves from one cell to another cell while transmitting data through a device, the ENodeB base station will search for a new channel to attach in order to avoid interruption of the current session. When this is found, it will command the device to switch to that channel while, at the same time, switching the session onto the new channel. Handover may also occur within overlapping cells if the capacity within the current cell is full, in order to allow users located outside the overlapping area in need of data to connect. Information regarding handover is often available in a base station. In this thesis, the time remaining until such a handover happens is used, denoted *time to handover*.

Throughput

In networks, throughput is defined as the rate of successful message delivery over a communication channel. Usually measured in megabits per second (Mbit/s), which is the measure used in this thesis. Information regarding user throughput, the throughput for the user, and the cell throughput, the total throughput for all users currently in a cell, is used in this work.

Active users and users in queue

The ENodeB base station also logs information regarding the number of active users in a cell. Furthermore, the number of users in queue, that is to say the number of users currently awaiting a connection to that particular cell, is logged.

Channel Quality Indicator

The Channel Quality Indicator (CQI), is, as implicated by the name, an indicator of the quality of the communication channel, which in this work is represented by an ordinal value in the range $\{1, 2, \ldots, 15\}$, where 1 denotes the poorest and 15 the best channel quality. The indicator is not standardised, and each base station manufacturer uses its own definition. For further reading on the technical specifications of CQI, please consult [8].

Rank

An essential element of wireless communication in i.e. 4G networks is MIMO[†], which refers to the concept to at the very same time send and receive multiple data signals on the same radio channel by using multi-path propagation. Through this, a transmission technique called spatial multiplexing (SMX) is utilised. *Rank* is defined as the number of layers in an LTE spatial multiplexing transmission [9], meaning it is an indicator of how well multiple antennas work. Naturally, the multiple antennas work well if the signal from each antenna has no correlation or interference to the others. A higher rank is equivalent with no correlation. In this work, rank is in the range of $\{1, 2\}$.

2.2 Mathematical background

Unsupervised machine learning is based on the problem of identifying groups in data without labels in form of a response variable. A vital tool for this matter is clustering, which is based on the concept of dividing data into subsets which share a common signature with respect to some features. There have been developed an extensive number of different clustering techniques, some of them falling under the category of partitional clustering, where the number of components[‡], denoted K, is

[†]Multiple-input and multiple-output.

[‡]The words components and clusters are used interchangeably in this thesis.

pre specified (as opposed to hierarchical clustering where K is unknown). In this thesis, one clustering method based on centroid models (K-means) and one based on distribution models (Gaussian Mixture Models) are implemented.

Naturally, finding the optimal number K, as the smallest K that describes the model satisfactory in order to avoid overfitting, is an important task for these clustering methods. Overfitting leads to problems where the data fits the model so well that it will be problematic to fit other data to the model, because minor variations due to noise in data are included in the model, mistaken to be true signals of the underlying relationship in the data [10, p. 22].

Throughout this section of mathematical preliminaries, the supporting literatures used are [10], [11] and [12] unless stated otherwise.

2.2.1 Gaussian Mixture Models

Mixture distributions are useful in modelling of heterogeneity in a cluster analysis. In [13], Priebe showed that with $n = 10\ 000$ observations, a mixture of around 30 normals is sufficient to produce a good approximation of a log normal density, whilst a mixture of 10 000 normals was required for a kernel density estimator. This is due to the ability of the mixture model to model rather complex distributions when choosing a convenient number of components in order to obtain accurate representations of local areas supporting the true distribution. With this feature, local variations in the observed data are handled whereas a single parametric family would be unable to do so [14].

A Gaussian Mixture Model, often denoted GMM, is based on a parametric probability density function which is represented as a weighted sum of Gaussian component densities. The model parameters are estimated by utilising the Expectation-Maximisation (EM) algorithm on training data.

For a data set $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i$ is a d-dimensional vector of observations, we assume that the points are IID[§] and that their underlying density $p(\mathbf{x})$ is defined as a finite mixture model with K components. The parametric probability function is given by

$$p(\mathbf{x}|\lambda) = \sum_{k=1}^{K} w_k g_k(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where $\lambda = \{w_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}, k = 1, \ldots, K$ denotes the parameters of the GMM, namely the mixture weights w_k such that $\sum_{k=1}^{K} w_k = 1$, the means $\boldsymbol{\mu}_k$ and covariance matrices $\boldsymbol{\Sigma}_k$. $\boldsymbol{\Sigma}_k$ can be either full rank, constrained or diagonal, and furthermore - the parameters given by λ can be shared among the Gaussian components.

[§]Independent and identically distributed

The w_k indicates weight of densities, hence represents the probability that a randomly selected $\mathbf{x}$ was generated by component k. The parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ describes the density of the d-dimensional continuous-valued data vector of measurements $\mathbf{x}$, which are mathematically represented by the d-variate Gaussian functions $g(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), k = 1, \dots, K$ of the form

$$g_k(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

A common method for estimating the parameters λ given a sequence N of training vectors $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and GMM configuration settings, is the maximum-likelihood (ML) estimation. The estimation is carried out by maximising the likelihood of the GMM given the training data in X. Hence, we want to maximise

$$p(X|\lambda) = \prod_{i=1}^{N} p(\mathbf{x}_i|\lambda) \tag{2.1}$$

with the assumption that the vectors are independent due to the need of making the problem tractable although this assumption actually often is incorrect. Direct maximisation of (2.1) is not possible, but by utilising the expectation-maximisation (EM) algorithm estimates of the ML parameters can be obtained iteratively. This is performed by estimating a new, improved model $\bar{\lambda}$ from an initial model λ such that $p(X|\bar{\lambda}) \geq p(X|\lambda)$. By iterating this step until convergence, i.e. when the change in means is small, the model parameters are obtained.

The first step is to make an initialisation of parameters, and one approach is to use a result from K-means algorithm to initialise, by letting

$$\begin{aligned} \mu_{k,\text{ first iteration}} &\leftarrow \mu_{k,K\text{-means}} \\ \Sigma_{k,\text{ first iteration}} &\leftarrow \text{Cov}(\text{cluster}(K)) \\ \pi_{k,\text{ first iteration}} &\leftarrow \frac{\text{number of points in } k}{\text{total number of points}} \end{aligned} \tag{2.2}$$

Other approaches for choosing initial parameters are of course possible. Alternatively, we could start with a set of initial weights and doing the M-step first. However, assuming we start with a set of initial parameters given by (2.2), the next step is to perform the E-step. The E-step of the EM-algorithm is performed by determining the assignment score for each point $\mathbf{x}_i$ to each Gaussian k, namely computing

$$\gamma_{i,k} = \frac{w_k \mathcal{N}_d(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} w_j \mathcal{N}_d(\mathbf{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = \frac{w_k g_k(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} w_j g_j(\mathbf{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{2.3}$$

for all data points $1 \leq i \leq N$ and mixture components $1 \leq k \leq K$. $\gamma_{i,k}$ is called the responsibility, or membership weights, and describes to what extent this Gaussian i is responsible for this particular point or observation $\mathbf{x}_t$. The output of this step is an $N \times K$ matrix of responsibilities, with each row sum being equal to 1.

Next is the M-step, where the parameters for each Gaussian k are updated using the new $\gamma_{i,k}$ as given by (2.3). The re-estimation of the components of $\bar{\lambda}$ are performed using the formulas given by (2.4)-(2.6), in that order, on $1 \leq k \leq K$ to ensure a monotonic increase of the likelihood. The derivation can be found in appendix A.1.

$$\bar{w}_k = \frac{1}{N} \sum_{i=1}^{N} \gamma_{i,k} \tag{2.4}$$

$$\bar{\boldsymbol{\mu}}_k = \frac{\sum_{i=1}^{N} \gamma_{i,k} \cdot \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_{i,k}} \tag{2.5}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} \gamma_{i,k} \cdot (\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)'}{\sum_{i=1}^{N} \gamma_{i,k}} \tag{2.6}$$

As can be observed in (2.5), the updated means are calculated in a similar manner as for a standard empirical average with the exception of the fractional weight given by the responsibility $\gamma_{i,k}$. Note that the terms do not cancel as this is a vector equation with the d-dimensional vectors $\mathbf{x}_i$ and $\bar{\boldsymbol{\mu}}_k$. The same yields for (2.6), as the calculation is rather similar to that of the empirical covariance matrix, but also this one is weighted by $\gamma_{i,k}$. As is evident from (2.4)-(2.6), the parameters that are the best fit for the assignment scores are found in these steps.

For every implementation of the E-step and M-step, one iteration is made. This should be performed until the likelihood or the parameters converge. From (2.1) we have that the log-likelihood is defined as

$$\log l(\lambda) = \log P(X|\lambda) = \sum_{i=1}^{N} \log p(\mathbf{x}_i|\lambda) = \sum_{i=1}^{N} \left(\log \sum_{k=1}^{K} w_k g_k(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \tag{2.7}$$

The log-likelihood should be computed according to (2.7) after every iteration, and when convergence is found, the iteration is stopped.

2.2.2 K-means

The K-means algorithm is a simple variant of the EM algorithm for Gaussian Mixture Models, where it is assumed that the covariance matrices and weights are fixed such that $\boldsymbol{\Sigma}_k = \sigma^2 I$ and $w_k = 1/K$. With these assumptions, the only parameter estimation to be performed is on the cluster centers $\boldsymbol{\mu}_k$. K-means is based on the concept of minimising the "within-cluster sum of squares" (WCSS) for the sets $\mathbf{C} = \{C_1, \ldots, C_K\}$ given by

$$\arg\min_{\mathbf{C}} \sum_{k=1}^{K} \sum_{\mathbf{x}\in C_k} ||\mathbf{x} - \boldsymbol{\mu}_k||^2 \tag{2.8}$$

In a similar manner as for the EM-algorithm explained in Section 2.2.1, K-means uses an iterative refinement approach by starting of with an initial set of K mean values $\boldsymbol{\mu}_k, i = 1, \ldots, K$, assigning each observation to the nearest mean value $\boldsymbol{\mu}_k$ by minimising the WCSS given by (2.8). In other words, the observations are assigned according to the Voronoi diagram [15] generated by the current set of means $\boldsymbol{\mu}_k^{(t)}$ as

$$C_k^{(t)} = \{\mathbf{x} : ||\mathbf{x} - \boldsymbol{\mu}_k^{(t)}||^2 \leq ||\mathbf{x} - \boldsymbol{\mu}_j||^2 \quad \forall j, 1 \leq j \leq K\}$$

Following this step, new means are updated to be the centroids of the observations assigned to the respective cluster:

$$\boldsymbol{\mu}_k^{(t+1)} = \frac{1}{|C_k^{(t)}|} \sum_{\mathbf{x}\in C_k^{(t)}} \mathbf{x}$$

These steps are repeated until convergence, as for the EM-algorithm explained in Section 2.2.1.

Chapter 3

Method

The primary software used for implementation and analysis in this thesis is R [16]. However, as will be explained later in this section, MATLAB [17] is also used to some extent due to computational reasons. To obtain the data set, a simulation tool developed by Ericsson is used.

3.1 Environment and setup

The data set $\mathcal{D}$ is of the dimension 21 million rows with features listed in Table 3.1. For an explanation of the network features, please revisit Section 2.1.1.

Table 3.1: Information available in data set.

User network features		Cell network features	
User throughput	Continuous	Cell throughput	Continuous
Previous user throughput	Continuous	Number of active users	Discrete
Channel Quality Indicator	Discrete	Number of users in queue	Discrete
Rank	Discrete		
Handover	Logical		
Time to handover	Continuous		

Other features		
Time	Continuous	
User	Factor	Unique ID tied to each user in the simulation.
User group	Factor	Which one the user belongs to, see Table 3.2.
Velocity	Discrete	Velocity of user, see Table 3.2.
Base station	Factor	
Cell	Factor	
Position	Continuous	Coordinates of position in the (x, y)-plane.

The simulation environment is modelled after an American city model with a city center with a high population density, surrounded by a highway and a suburban area with a lower population density. A print of the map is shown in Figure 3.1.

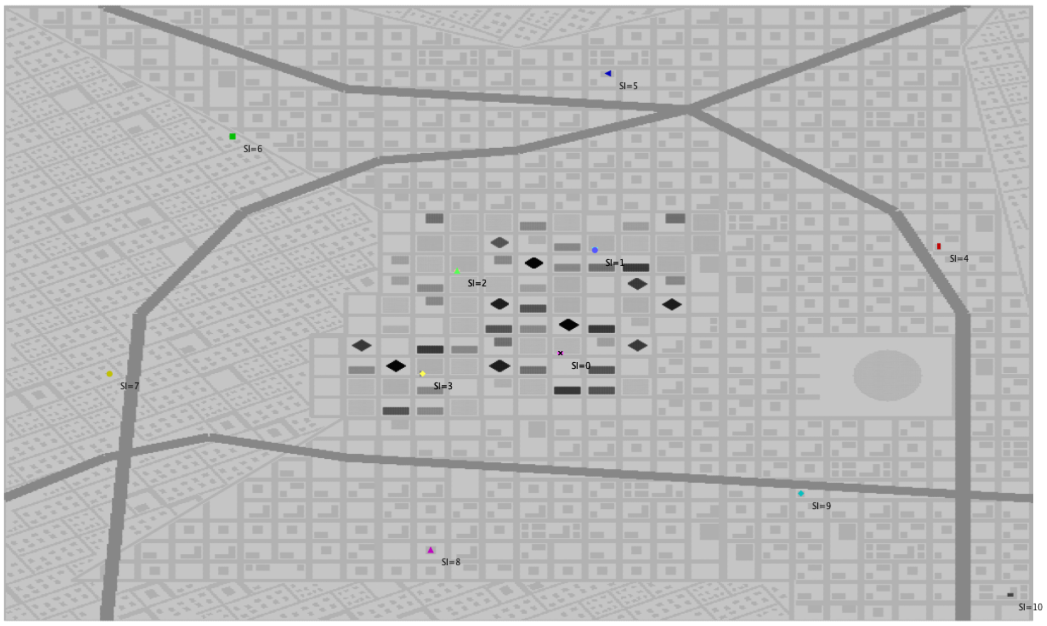


Figure 3.1: A print of the simulated city environment.

The 4G network is a setup of 11 base stations, together covering 33 cells, spatially distributed as in Figure 3.1. The data set is a result of a simulation of 60 minutes with a time step of 100ms. In this data set, each observation corresponds to one of 600 active users for a time step, where all users are continuously downloading 1 MB files*. The users are distributed amongst 6 typical user groups in the following manner:

Table 3.2: Users in the simulation.

Amount	Group identity	Description	Velocity of users
350	U_0	Indoor users	-
50	U_1	Pedestrian users walking outside	3 km/h
50	U_2	Car drivers on streets	30 km/h
50	U_3	Car drivers on streets	50 km/h
50	U_4	Car drivers on highway	70 km/h
50	U_5	Car drivers on highway	90 km/h

3.1.1 Limitations

Although the simulation tool used is very advanced, this simulation makes some limiting assumptions tied to the user generation that should be adressed. Firstly, it is obvious that the assumption that users in a network can be categorised in the simple manner stated in Table 3.2 is a rather vigorous simplification. However, this

*Note that voice calls are not performed over 4G networks today, hence the usage is mobile data without voice calls.

work does not aim to investigate behaviour tied to specific user groups, but merely uses this setup as a method to create some diversity amongst the users in the simulation. Nevertheless, one should bear in mind that greater diversity in user behaviour will most probably occur in real network data when investigating cluster signatures.

Secondly, the assumption that all users are downloading 1 MB files is quite contradictory to what was found in [6], where the network usage among users was found to vary significantly in a real 3G network, specifically it was found that a few users are hogging the majority of the network usage. Furthermore, this simulation assumes for simplicity that nobody leaves or arrives the city area during the simulation time. However, as the analyses are performed primarily on cell level, where users of course will leave and arrive during the simulation, this assumption will not limit the analyses. It should be noted that the users in this simulation will not bounce off the edges of the map, but merely continue in a mirrored point of the map. It should also be noted that 600 users are very low for a real network environment in a city.

Lastly, the data set is a result of only 60 minutes of simulated behaviour. As was found in [6], there are great variations in user behaviour during a day, as well as some variations in behaviour between weekdays, which this work will not deal with.

3.2 Clustering

The first clustering approach is to implement the K-means algorithm as explained in Section 2.2.2 due to its easy implementation and its documented effect on similar kind of data. In [3], a K-means clustering based on the probability density function (PDF) of a certain feature called signal-to-interference-ratio was used to create clustering classes. The resulting clusters with a characteristic PDF were used as labels for supervised classification of other data. In the paper, the classifier used was the K-nearest-neighbour classifier, a classifier that searches for the K nearest training vector based on an input vector sample, and assigns the class that is presented by the majority of its neighbours [18]. As mentioned in Section 1.1, it was found that it was an improvement with clustering and using a characteristic PDF for signal-to-interference-ratio for every cluster as opposed to using a mean PDF for all data within a cell.

However, although having the strength of being a simple method, the K-means algorithm is typically fast for low dimensional data - and it is found that the algorithm is too computationally slow for the large data sets in this work. Furthermore, as can be understood from the steps explained in Section 2.2.2, K-means might terminate at a local optimum and fail to find a global optima. Additionally, K-means will not identify outliers and might have trouble with noisy data, and will also be restricted to data which has the notion of a center.

Instead, clustering using the GMM algorithm explained in Section 2.2.1 is implemented. In [7] it was assumed that base stations loads are time sequences of multivariate Gaussian random variables, and modelled as a Gaussian Markov Random Field. This assumption was later ensured by the authors, suggesting that such patterns can be found in this kind of data, thus justifying the use of Gaussian Mixture Models in this thesis.

One of the primary strengths of the GMM algorithm is the interpretability of the model, i.e. it learns a generative model of each cluster which enables generation of new data based on the model learned through GMM clustering. Moreover, as the assignment of clusters to observations (E-step) and re-estimation of parameters (M-step) is based on probabilistic membership, the assumptions are stronger than for K-means. However, it should be noted that GMM also can terminate at a local optimum, hence the initialisation of parameters is rather important. Furthermore, the algorithm will experience problems with discovering clusters with non-convex shapes. Both the K-means and GMM algorithm requires a pre specified number K of clusters, a process explained in Section 3.2.1.

The $d = 5$ features used for clustering is those of typically available in a base station, namely

$$\{\text{user thr., cqi, time to handover, cell thr., \# active users}\} \tag{3.1}$$

denoting as explained in Section 2.1.1. Other features available in the data set that could have been included based on the availability criterion is *previous user throughput*, *number of users in queue* and *rank*. However, as the first two are highly correlated with *user throughput* and *number of active users* respectively, they are excluded in this work, as is *rank*, due to its range restricted to $\{1, 2\}$, in order to avoid an imbalanced set of features. As the magnitude of the features in the data set differs, scaling is used to normalise the values in order to allow for equal impact on the clustering. Knowledge of clusters *within* a cell is of primary interest in this thesis, hence subsets $\mathcal{C}_j$ of data for every cell is used for clustering;

$$\mathcal{C}_j \subseteq \mathcal{D} \quad \text{for } j = 1, \dots, 33.$$

The dimensions of $\mathcal{C}_j, j = 1, \dots, 33$ vary as the number of observations in each cell differ, ranging from 0.5-1 million observations in $\mathbb{R}^d$, where $\mathbb{R}^d$ denotes the features used in the clustering algorithm.

It should be noted, as will also be seen later in this thesis, that correlation between some of the variables in (3.1) exist, primarily between *cell throughput* and *number of active users* which are highly correlated. Additionally, some of the features are discrete. Both these issues can lead to singular or close-singular covariance matrices Σ_k. Therefore, when implementing the clustering algorithm, a regularisation term of 10^{-5} was added, in order to prevent singularity.

3.2.1 Number of components

As mentioned earlier in this chapter, the primary software used this thesis was R, and upon initialising the clustering the R package mclust for GMM clustering was the first approach. However, while mclust works well for smaller example data sets, the implementation is found to have a significantly long computation time for most data sets $\mathcal{C}_j$, even though the computer used is very powerful. Using instead the class gmdistribution in MATLAB, the computation time is found to decrease significantly.

However, as opposed to mclust, fit.gmdistribution does not automatically choose the optimal number of components K, thus K is required as an input parameter. In this section, two common approaches to determine the optimal number of components are implemented. As mentioned in Section 2.2, choosing K is an important task in order to describe a model sufficiently while still avoiding overfitting. The two complementing methods are found to suggest the same optimal number of K, which for this data set is $K = 5$.

Bayesian Information Criterion

A well-known way to determine the optimal number of components this is to investigate the Bayesian Information Criterion (BIC), as proposed in [19] for different K. The BIC for model selection is given by

$$\text{BIC} = -2\log l(\lambda) + K \log N^* \tag{3.2}$$

where $\log l(\lambda)$ is the log-likelihood as given by (2.7), K is the number of components and N^* the number of observations in the data set. The number of components is therefore determined by manually extracting the BIC for $\mathcal{C}_j, j = 1, \ldots, 33$, for every $K \in \{1, 2, \ldots, 8\}$. A graphical representation of the normalised BIC's can be found in Figure 3.2.

BIC is closely related to the Akaike information criterion (AIC) proposed in [20], substituting $\log N^*$ by 2 in (3.2). However, one should note that as they have different objective target models, they are not directly comparable: While AIC aims to select a model that in the most satisfactory way characterise an unknown, high dimensional reality - out-ruling reality from the set of candidate models being considered - BIC searches for the true model among the set of candidates.

Nevertheless, the practical difference is the size of the penalty, whereas the BIC penalises model complexity more deliberately than AIC. There is a chance that the AIC model chooses a too complex model independently of N^*, whereas BIC has a very little chance of doing so for a sufficient N^*. However, BIC has a larger chance than AIC to choose a model that is too small, independently of N^*. Generally, it is often wise to inspect both AIC and BIC together in model selection, and upon inspecting the AIC for this model it is found that the criteria agree on the same model.

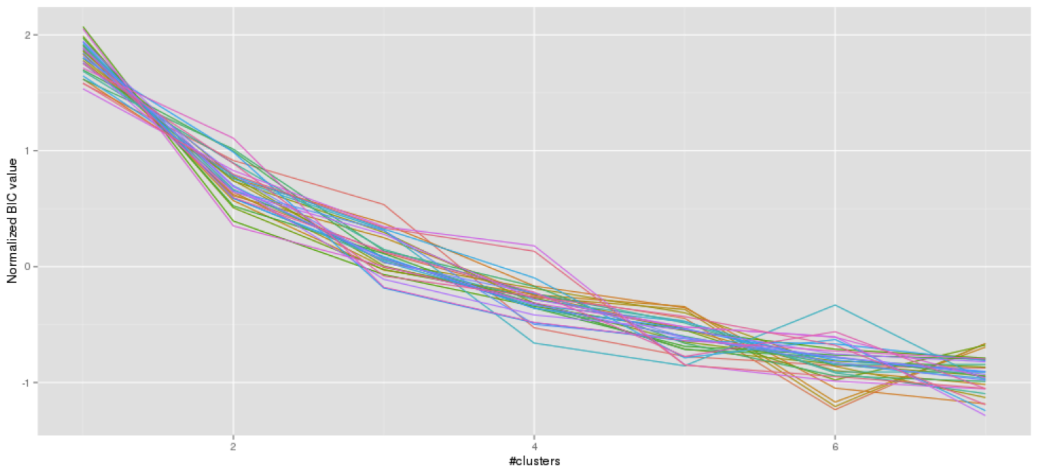


Figure 3.2: BIC for $\mathcal{C}_j, j = 1, \ldots, 33$.

As can be observed in Figure 3.2, the curves flat out after $k = 5$, thus representing the "elbow" indicating the appropriate number of clusters as denoted by [21]. The elbow represents an area of which when passed by increasing the number of components, the model will offer much less significant improvement. Evaluating (3.2), it is obvious that the BIC will continue to decrease until $K \to N$, symbolising a model where each observation has a unique component of its own, but at some point the problem of overfitting the model will occur.

The approach for avoiding overfitting while still optimising the accuracy of a model is therefore to choose a number of components around the area where the curves flat out, hence the number of components is chosen to be $K = 5$ for $\mathcal{C}_j, j = 1, \ldots, 33$. However, such a decision based on a visual interpretation of graphical material without a mathematical formalisation might be error prone. Indeed, by the looks of Figure 3.2, one could argue that $K = 4$ or $K = 6$ are just as good representations of the elbow and therefore optimal. Other approaches for determining the optimal number of components K have been suggested, one of then being the "gap statistic", also implemented in this work to complement the BIC argument.

Gap statistic

The concept of the gap statistic was introduced in [21], where a statistical procedure to formalise the elbow concept was proposed. The theory is based on the following approach:

For a data set $\mathcal{D}^* = \{\mathbf{x}_i \in \mathrm{R}^d, i = 1, \ldots, N^*\}$, we use the squared Eucludian distance to define the distance $d_{ii'}$ between observations i and i' such that

$$d_{ii'} = ||\mathbf{x}_i - \mathbf{x}_{i'}||^2$$

After clustering the data into K clusters $C_k, k = 1, \ldots, K$ where the observations in each cluster is given by $n_k = |C_k|$, the pairwise distances for all points in cluster k is given by

$$D_k = \sum_{i,i' \in C_k} d_{ii'}$$

Using this, the pooled within-cluster sum of squares (WCSS) W_K around the cluster means is given by

$$W_K = \sum_{k=1}^{K} \frac{1}{2n_k} D_k$$

The approach of the gap statistic is to standardise the graphical comparison of $\log W_K$ with an appropriate distribution with no obvious clustering, called a null reference distribution. This is typically generated by a uniform sampling using the parameter boundaries of the data. Through this comparison the optimal K is found to be the K for which $\log W_K$ falls the farthest below the reference curve given by the null distribution. Formalising this in a formula yields

$$\mathrm{Gap}_n(K) = E_n^*[\log W_K] - \log W_K = \frac{1}{B} \sum_{b=1}^{B} \log W_{Kb}^* - \log W_K \tag{3.3}$$

where E_n^* denotes the expectation under a sample of size n from the null reference distribution. This is estimated by computing the average of B copies generated with a Monte Carlo sample from the null distribution, each giving within-dispersion measures $W_{Kb}^*, b = 1, \ldots, B$ for the number of clusters K. These will exhibit a standard deviation

$$\mathrm{sd}(K) = \sqrt{\frac{1}{B} \sum_{b=1}^{B} \left(\log W_{Kb}^* - \frac{1}{B} \sum_{b=1}^{B} \log W_{Kb}^* \right)^2} \tag{3.4}$$

Combining the standard deviation in (3.4) with the simulation error yields the quantity s_K defined as

$$s_K = \sqrt{1 + 1/B}\ \mathrm{sd}(K)$$

The optimal $\hat{K}$ is the smallest K such that

$$\mathrm{Gap}_n(K) \geq \mathrm{Gap}_n(K+1) - s_{K+1}$$

The generality of this measure makes it applicable to any clustering method, and even other distance measures $d_{ii'}$ other than the squared Euclidian distance. Note that when using this on clustering by the use of Gaussian Mixture Models, $\log W_K$ has an interpretation as a log-likelihood as in (2.7) [22].

Implementing this method on the data set $\mathcal{C}_1$ yields estimates of the gap statistic as illustrated in Figure 3.3, for $K \in \{1, \ldots, 8\}$ and $B = 10$. Similar results are obtained for the other $\mathcal{C}_j, j = 2, \ldots, 33$ as well, indicating the same $\hat{K}$.

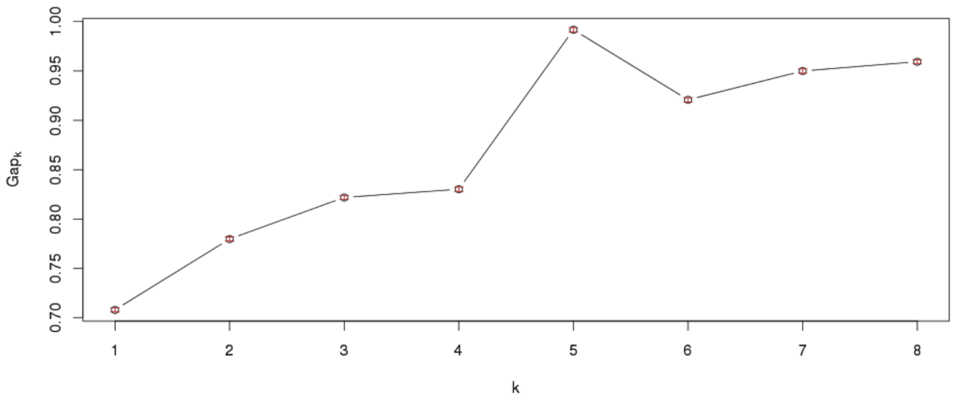


Figure 3.3: Gap statistic for $K \in \{2, \ldots, 8\}$.

It is evident from Figure 3.3 that the optimal number of components is $\hat{K} = 5$, confirming the optimal number of components found by investigating the BIC curves.

3.3 Classification

While clustering is a method of unsupervised learning, classification is a method of so called supervised learning, as the algorithm is dependent of having response variables/labels. A classification model is built on a training set of data, and tested on a different set of data called the test set. The output is a table carrying information regarding how many correct classifications were made. This kind of table is often called a confusion matrix or contingency table, see Figure 3.4.

The notations in Figure 3.4 denotes the following: a true positive classification is equivalent with a hit, true negative is equivalent with a correct rejection, false positive denotes a Type I error and false negative denotes a Type II error.

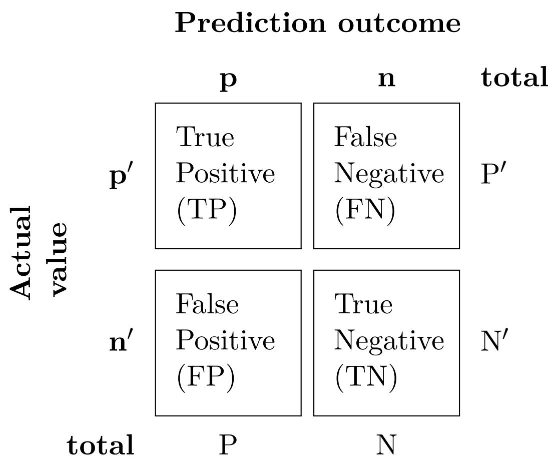


Figure 3.4: Outline of a confusion matrix.

3.3.1 Accuracy

When investigating the accuracies, the following terminology is used:

The sensitivity, also called the true positive rate given by

$$\text{sensitivity} = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{3.5}$$

and the specificity, also called the true negative rate

$$\text{specificity} = \frac{TN}{N} = \frac{TN}{FP + TN} \tag{3.6}$$

and the balanced accuracy, representing the average accuracy obtained on either class, given by

$$\frac{\text{sensitivity} + \text{specificity}}{2} \tag{3.7}$$

Often the measure of accuracy, given by $\frac{TN+TN}{P+N}$, is used instead of balanced accuracy [23]. However, for imbalanced[†] data sets this might be a deceiving performance measure. This is due to the problem arisen from when a classifier gets biased towards a class of greater representation, because when this classifier is applied to a test set imbalanced in a similar manner the accuracy estimate may be rather optimistic [24]. The classification in this work is performed on the responses given by a clustering algorithm and there is no guarantee that the clusters found during clustering are of equal sizes. Hence, the balanced accuracy is used as a measure of the results in this work.

[†]An imbalanced data set has different number of representatives from each of the classes.

3.3.2 One-vs-all

A common strategy for solving multi-class classification problems is to reduce it to multiple binary classification problems, one of which is called the one-vs-all strategy. With this approach, a classifier for each class (response variable) is used, treating each class as positive samples and the remaining classes as negative samples. However, using this strategy one should bear in mind two relevant problems associated with the approach [25, p. 338].

Firstly, there might be differences between the confidence and accuracy tied to the different binary classes, hence one should be sure to investigate all such outputs separately in complement to a mean value across the classes.

Furthermore, as the number of negative samples typically will outnumber the number of positive samples[‡], the classification algorithm might be biased towards getting the negatives correct (TN) rather than the positives (TP). This problem can be identified by experiencing much higher specificities than the corresponding sensitivities for the classifier. This problem is adressed by using the balanced accuracy instead of accuracy, but one should still look at both sensitivity and specificity to understand the balanced accuracy in case there are great differences between the two measures.

3.3.3 Algorithm

Many classification algorithms have been developed, and exhibit different pros and cons. The algorithm used in this thesis is based on CART[§] models, also known as decision trees, which is based on the following approach.

In a decision tree some input values x_k are tested against some threshold values t_l in every node in descending order, building deep trees ending in regions R_m based on these conditions, see Figure 3.5.

[‡]Here, the negative samples will be represented by observations from $K-1=4$ clusters, whilst the positive samples will come from a single cluster.

[§]Anagram for Classification And Regression Trees

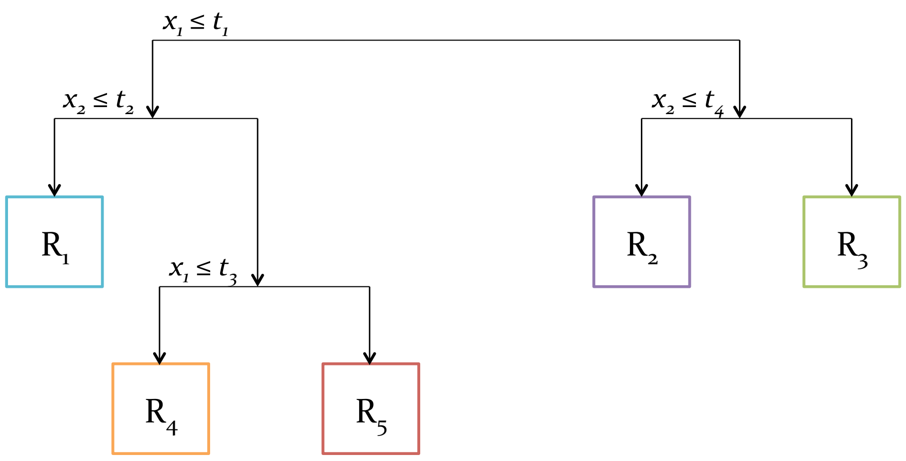


Figure 3.5: Schematic illustration of decision tree.

Mathematically, the model can be expressed as

$$f(\mathbf{x}) = \mathrm{E}[y|\mathbf{x}] = \sum_{m=1}^{M} w_m \mathrm{I}(\mathbf{x} \in R_m) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}; \mathbf{v}_m)$$

where w_m is the mean response in the corresponding region R_m, in which $\phi_m(\mathbf{x}; \mathbf{v}_m)$ is the basis function learned from the data with parameters $\mathbf{v}_m$, which denotes the choice of variable x_k to split on with the threshold value t_l.

To determine the best feature and its best value to split on, the following split function is used;

$$(j^*, t^*) = \arg \min_{j \in \{1,\ldots,D\}} \min_{t \in \mathcal{T}_j} c(\{\mathbf{x}_i, y_i : x_{ij} \le t\}) + c(\{\mathbf{x}_i, y_i : x_{ij} > t\})$$

where c denotes a cost function. The cost functions are acquired in a similar manner as for regression, where the cost is the sum of squared residuals $\sum_{i \in \mathcal{D}} (y_i - \bar{y})^2$ in a data set $\mathcal{D}$. To acquire this, the class-conditional probabilities are first estimated by

$$\hat{\pi}_c = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mathrm{I}(y_i = c) \tag{3.8}$$

In this thesis, the Gini index will be used as a measure of error rate. As (3.8) is the probability that a random entry in the leaf belongs to class c, $(1 - \hat{\pi}_c)$ denotes the probability that it would be misclassified, and the expected error rate is given by the Gini index as

$$\sum_{c=1}^{C} \hat{\pi}_c (1 - \hat{\pi}_c) = \sum_c \hat{\pi}_c - \sum_c \hat{\pi}_c^2 = 1 - \sum_c \hat{\pi}_c^2 \tag{3.9}$$

If a predictor variable has a low Gini value, or equivalently a higher decrease in Gini, it means that the variable plays a greater role in the partitioning of data into the classes.

As stated in [10, p. 544-550], advantages for CART models are inter alia that they can handle a mix of discrete and continuous inputs without difficulty, which is necessary for this data (recall Table 3.1). Moreover, CART models are able to perform automatic variable selection, which is an advantage as we do not have knowledge of which (if any) features have more impact on a cluster than others. Lastly, CART models are known to handle large data sets well, which is crucial with the dimensions of data sets in this work. To adress the main disadvantage of CART models, due to their greedy nature often resulting in quite low accuracy, the Random Forest algorithm was used.

In [26], Random Forest was found to be a highly accurate classification algorithm in an empirical comparison of such methods, as well as in [27], where a classifier from the Random Forest family was found to outshine the other classifiers in a comparison. The classification algorithm was developed by Breiman [28] and Cutler and is based on the concept of decision trees, but instead of growing a single, "deep" tree, random forest uses multiple shallow decision trees and aggregate over them to improve accuracy. Random forest is robust to overfitting, as each tree get a bootstrapped version of the original training data. Furthermore, randomness is used in every node when selecting which feature to split on. The responsive variable is determined by the votes (positive or negative) of the majority of the trees.

Chapter 4

Results and analysis

4.1 Clustering

The clustering was performed on all subsets $\mathcal{C}_j, j = 1, \ldots, 33$, but in order to facilitate the analysis and presentation of results, two cells representing different geographical locations are chosen as primary analysis objects, namely cell 1 and 19. The approximate coverage areas of the corresponding base stations, 1 ($\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$) and 6 ($\mathcal{C}_{19}, \mathcal{C}_{20}, \mathcal{C}_{21}$), are shown in Figure 4.1.

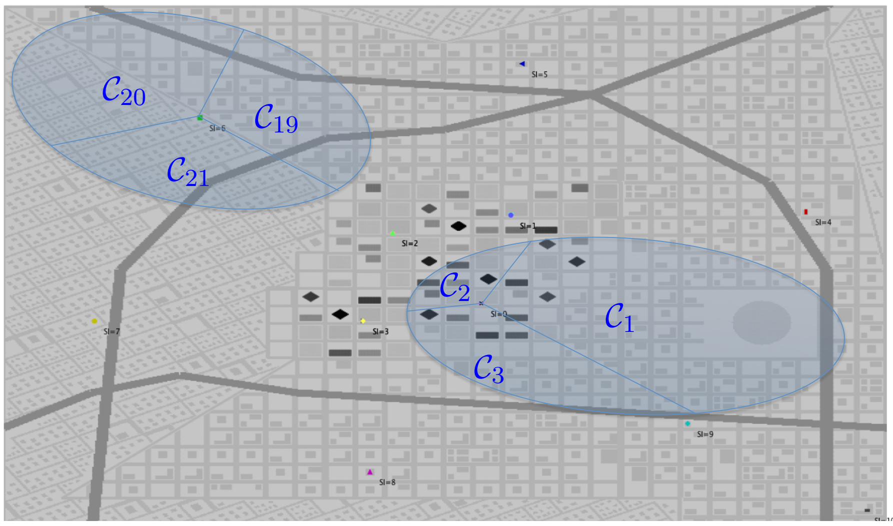


Figure 4.1: Coverage areas for base stations 1 and 6.

The clustering was performed on the $d = 5$ features in (3.1) with $K = 5$ components. Hence, the GMM output for every data set $\mathcal{C}_j, j = 1, \ldots, 33$ was a mixture of K multivariate normal distributions for clusters $C_k^{(j)}, k = 1, \ldots, K$ with means $\boldsymbol{\mu}_k \in \mathbb{R}^d$, and covariance matrices $\boldsymbol{\Sigma}_k \in \mathbb{R}^{d \times d}$.

	$j = 1$	$j = 19$
$C_1^{(j)}$	10.7%	14.8%
$C_2^{(j)}$	18.6%	18.0%
$C_3^{(j)}$	19.3%	17.0%
$C_4^{(j)}$	29.9%	33.7%
$C_5^{(j)}$	21.5%	16.5%

Table 4.1: Proportion of each cluster $C_k^{(j)}, k = 1 \dots, K, j = \{1, 19\}$ in $\mathcal{C}_1$ and $\mathcal{C}_{19}$.

The proportion of each cluster $C_k^{(j)}, k = 1 \dots, K, j = \{1, 19\}$ in $\mathcal{C}_1$ and $\mathcal{C}_{19}$ are found in Table 4.1.

4.1.1 Cluster signatures

Once the cluster stamp for each observation is obtained, it is of course of interest to understand the signature of each cluster. A representation of these signatures for the clusters in $\mathcal{C}_1$ can be found in Figure 4.2.

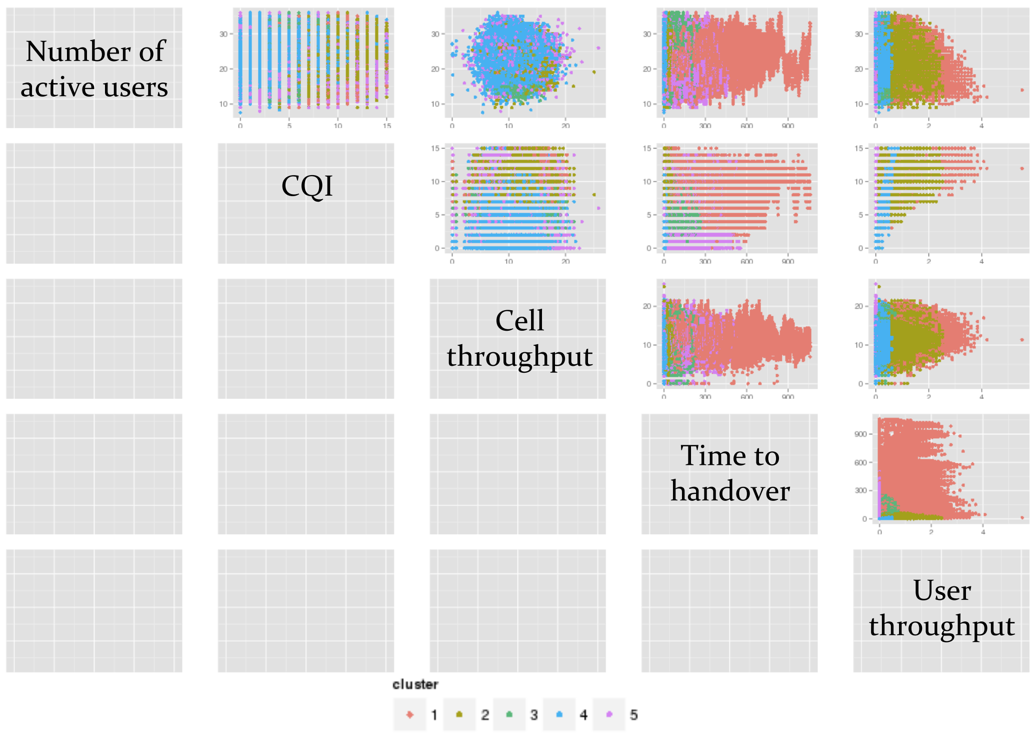
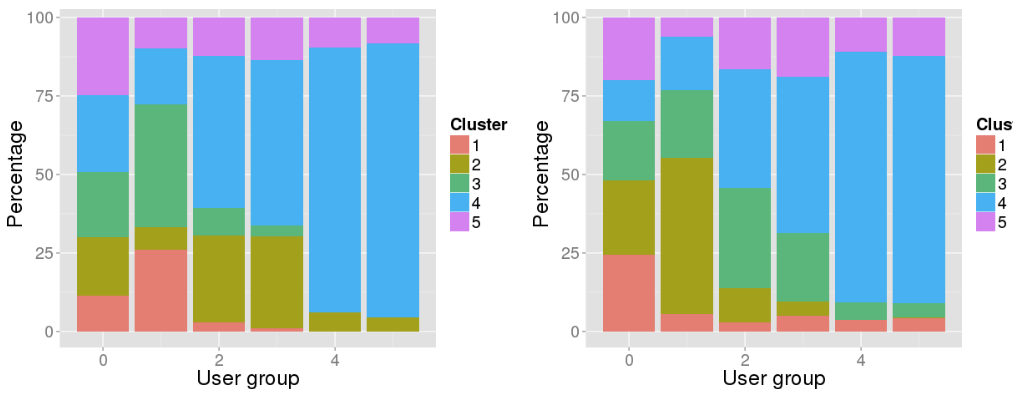


Figure 4.2: Observations tied to clusters in $\mathcal{C}_1$.

As can be seen in these figures, the clustering is reasonable as the behaviour of users differ significantly between clusters. For example, $C_1^{(1)}$ seem to represent the “heavy hitters” in [6], typically having high user throughput and representing a small group of users - users that are valuable to identify in an early stage from an RRM perspective. Note that when comparing cell throughput to the number of active users in Figure 4.2 , no significant cluster behaviour can be found - which is expected as the parameters naturally are highly correlated.

Using this kind of information, if one can use a classification algorithm to classify users in $\mathcal{C}_1$ into i.e. $C_2^{(1)}$ based on a short time frame of information, one can expect that this user will experience a high channel quality but not high user throughput, and have short time until handover - thus preparing for finding a new channel earlier than when a user approaches the cell boundaries. One way to do this classification is to use the Random Forest classification algorithm, which will be presented in Section 4.2.

A quite interesting observation is that the GMM did to some extent manage to identify the user groups as presented in Table 3.2, see Figure 4.3. Note that a majority of the highway drivers, U_4 and U_5, are represented by a single cluster in both cells, $C_4^{(1)}$ and $C_4^{(19)*}$, and some clusters are not even tied to any highway driver. In fact, the only user group that seems to have a close-uniform distribution amongst cluster identities is the indoor users in U_0. This might be a result of the fact that $|U_0| = 7|U_l|, l = 1, \dots, 5$, but differences in cluster proportions can still be seen amongst the equal sized $U_1, \dots, U_5$.



(a) $\mathcal{C}_1$ (b) $\mathcal{C}_{19}$

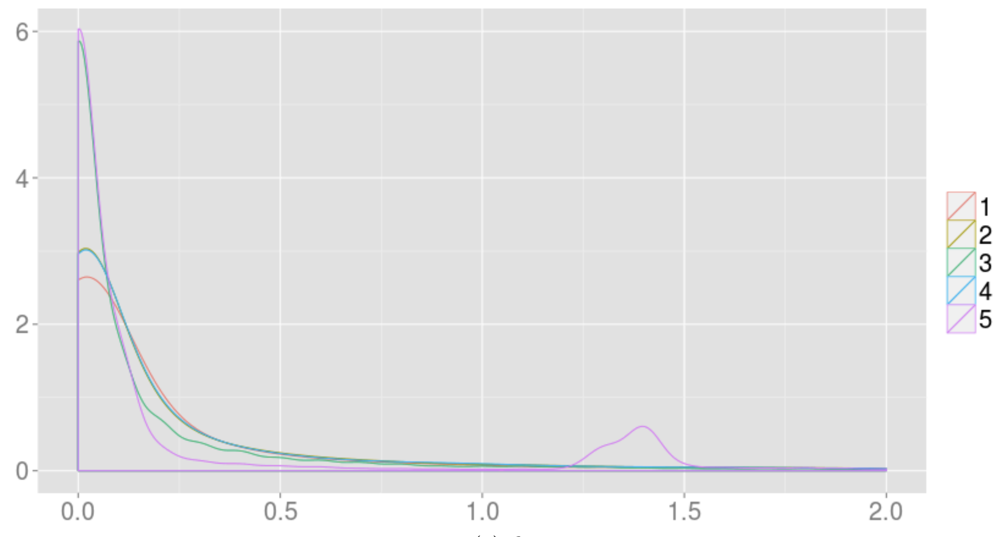
Figure 4.3: A graphical illustration of the proportion of observations by each user group $\in U_0, \dots, U_5$ tied to each of the clusters for $\mathcal{C}_1$ and $\mathcal{C}_{19}$.

*Note that $C_4^{(1)}$ and $C_4^{(19)}$ are the proportionally biggest clusters according to Table 4.1.

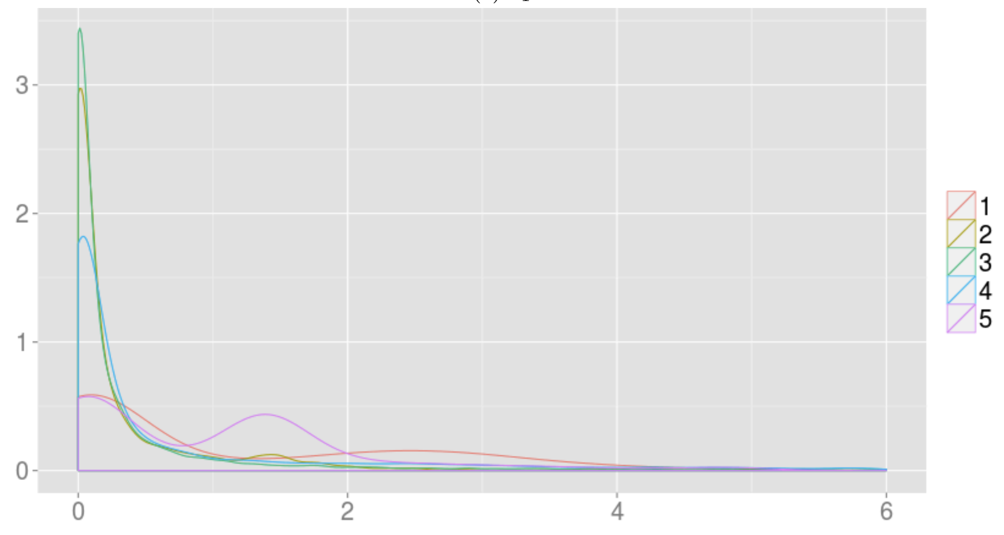
4.1.2 Temporal behaviour

Time in cluster

An important aspect to enable prediction based on this clustering, is to know how long a user is expected to stay in a cluster once entered and which cluster the user is expected to be in next, given the knowledge of which cluster the user belongs to at the moment. The densities of time spent in the clusters are shown in Figure 4.4. Note that the densities do have a heavy right tail, due to occasional visits as high as two minutes, but they are cut in these figures for presentational purposes.



(a) $\mathcal{C}_1$



(b) $\mathcal{C}_{19}$

Figure 4.4: Density of time in cluster in seconds.

As can be observed in Figure 4.4, the expected time in a cluster differs only slightly amongst $C_k^{(1)}, k = 1, \ldots, K$, with the exception of $C_5^{(1)}$. Furthermore, observe that the time spent in each cluster is very short, most being < 0.5 seconds, suggesting that a classification should be made in a very short time frame in order to be useful. However, in $\mathcal{C}_{19}$ a higher time spent in each cluster can be observed, as well as differences amongst $C_k^{(19)}, k = 1, \ldots, K$. Using this information in combination with cluster signatures as presented in Section 4.1.1 can enable prediction of future user behaviour that can be useful for RRM.

The time spent in clusters as presented above are quite low, hence an aggregated look at this might be useful. In [5] the user sessions[†] D_s was found to follow a exponential distribution in logarithmic scale as $D_s = 10^{\Delta}$ where $\Delta \sim \text{Exp}(0.3591)$. Hence, it was found that D_s exhibits heavy-tail characteristics with Pareto index[‡] $\alpha_D = 1.21$, resulting in a mean value of $\bar{D}_s = 5.71$ s. From this distribution, a majority of the users will have a user session $D_s > 1.8$ s, and some sessions will be much longer. Hence, one could expect that a user should belong to a certain cluster, a "main" cluster, a majority of the time within a certain longer time frame corresponding to a user session, with only a few observations along the time series tied to another cluster. Such observations would divide the aggregated longer visit to multiple short visits with this "interruption", resulting in a higher density for shorter time in cluster. Hence, it might be of greater interest to aggregate this series and investigate whether such a main cluster does exist, and if it does - evaluate the time spent in the main cluster and the corresponding expected behaviour.

Cluster transitions

The information regarding which cluster the user is likely to be in next can be evaluated using Markov Chains, and investigating the $K \times K$ transition matrix of the clusters in a cell. The element in row k' and column k in the transition matrix indicates the probability of being in cluster k at time t given that it is in cluster k' at time $t - 1$,

$$p_{k'k} = P\left(x_t = C_k^{(j)} | x_{t-1} = C_{k'}^{(j)}\right)$$

The transition matrices for the clusters in $\mathcal{C}_1$ and $\mathcal{C}_{19}$ is given by

$$T_{\mathcal{C}_1} = \begin{bmatrix} 0.914 & 0.017 & 0.053 & 0.000 & 0.017 \\ 0.009 & 0.835 & 0.038 & 0.106 & 0.012 \\ 0.030 & 0.037 & 0.854 & 0.006 & 0.072 \\ 0.000 & 0.066 & 0.004 & 0.898 & 0.033 \\ 0.008 & 0.009 & 0.067 & 0.045 & 0.871 \end{bmatrix} \quad T_{\mathcal{C}_{19}} = \begin{bmatrix} 0.910 & 0.002 & 0.015 & 0.042 & 0.032 \\ 0.002 & 0.962 & 0.033 & 0.001 & 0.003 \\ 0.014 & 0.034 & 0.895 & 0.047 & 0.011 \\ 0.018 & 0.001 & 0.024 & 0.932 & 0.025 \\ 0.027 & 0.005 & 0.009 & 0.054 & 0.906 \end{bmatrix}$$

[†]Defined as all consecutive seconds in which the same user transmits at least one bit.

[‡]α in $F(x) = 1 - (c/x)^{\alpha}$

As can be observed from the diagonal elements, the probability for staying in the same cluster varies, where the differences does not originate from the sizes as presented in Table 4.1. The high probabilities of staying is quite expected, as the time step for the probability of 100 ms is very low. Furthermore, differences amongst the probabilities can be observed, suggesting that a person is more prone to transfer to certain clusters than others. For example, a person in $C_4^{(1)}$ is more than 16 times as likely to be in $C_2^{(1)}$ compared to $C_3^{(1)}$, and have an infinite small probability to be in $C_1^{(1)}$ next.

Recall that the proportions of the clusters differed amongst the user groups $U_0, \ldots, U_5$. In some ENodeB base stations it is possible to determine the velocity of users in the cell, hence with that information one could inspect separate transition matrices for $U_0, \ldots, U_5$, which for the subset $\mathcal{C}_1$ are given by the following:

$$\underbrace{\begin{bmatrix} 0.92 & 0.01 & 0.05 & 0.00 & 0.02 \\ 0.01 & 0.85 & 0.03 & 0.10 & 0.01 \\ 0.03 & 0.03 & 0.85 & 0.01 & 0.08 \\ 0.00 & 0.08 & 0.00 & 0.89 & 0.03 \\ 0.01 & 0.01 & 0.07 & 0.02 & 0.89 \end{bmatrix}}_{U_0}, \underbrace{\begin{bmatrix} 0.93 & 0.00 & 0.06 & 0.00 & 0.01 \\ 0.01 & 0.74 & 0.15 & 0.09 & 0.01 \\ 0.04 & 0.03 & 0.91 & 0.00 & 0.02 \\ 0.00 & 0.03 & 0.01 & 0.94 & 0.02 \\ 0.04 & 0.01 & 0.08 & 0.03 & 0.84 \end{bmatrix}}_{U_1}, \underbrace{\begin{bmatrix} 0.76 & 0.16 & 0.07 & 0.00 & 0.01 \\ 0.02 & 0.80 & 0.06 & 0.11 & 0.01 \\ 0.02 & 0.20 & 0.69 & 0.01 & 0.08 \\ 0.00 & 0.06 & 0.00 & 0.90 & 0.04 \\ 0.00 & 0.03 & 0.07 & 0.17 & 0.73 \end{bmatrix}}_{U_2}$$

$$\underbrace{\begin{bmatrix} 0.62 & 0.35 & 0.02 & 0.00 & 0.01 \\ 0.01 & 0.82 & 0.03 & 0.12 & 0.02 \\ 0.01 & 0.24 & 0.66 & 0.01 & 0.08 \\ 0.00 & 0.07 & 0.00 & 0.88 & 0.05 \\ 0.00 & 0.04 & 0.02 & 0.20 & 0.74 \end{bmatrix}}_{U_3}, \underbrace{\begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.41 & 0.00 & 0.56 & 0.03 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.04 & 0.00 & 0.91 & 0.05 \\ 0.00 & 0.03 & 0.00 & 0.41 & 0.56 \end{bmatrix}}_{U_4}, \underbrace{\begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.37 & 0.00 & 0.59 & 0.04 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.03 & 0.00 & 0.92 & 0.05 \\ 0.00 & 0.03 & 0.00 & 0.51 & 0.46 \end{bmatrix}}_{U_5}$$

It can be observed from the above matrices that the transition probabilities are significantly higher for some clusters than others for the vehicular user groups (U_2, U_3, U_4, U_5) moving with high velocities - especially for the highway users that are only represented in three of the clusters. More specifically, for every row there is one probability that, except for the diagonal element, is considerably greater than the others. This implies that if we have information regarding a users velocity available, the probability of predicting C_k at time t correctly is much higher than without that information, given that the prediction is made on the highest probability. Note that transition matrices for U_0 and U_1, which together accounts for 2/3 of the users, resemble that of the aggregated $T_{\mathcal{C}_1}$.

One reason behind these differences are most probably that *time to handover* is included as a clustering feature in the GMM model, which naturally will differ significantly for a vehicular user moving fast through the city than an indoor user or a pedestrian moving slowly across the map. However, as was shown in Section 4.1.1, the cluster signatures differs significantly for the other features as well, hence discards a circular argument starting and ending with *time to handover*.

Note that the concept of aggregating over a main cluster as mentioned above could be applied here as well.

4.1.3 Prediction of load

For RRM applicational purposes, a valuable concept would be if one could predict the user throughput tied to a user. Thus, another clustering using the GMM algorithm was performed without user throughput as a feature, hence the GMM algorithm was performed on the data set with features

$$\{\text{cqi, time to handover, cell thr., \# active users}\} \tag{4.1}$$

The resulting clustering identities $\breve{C}_k^{(1)}, k = 1, \ldots, K$ was compared to the "hidden" variable *user throughput* in the full data set. A graphical representation of the density of user throughput for each of the assigned clusters for $\mathcal{C}_1$ is shown in Figure 4.5.

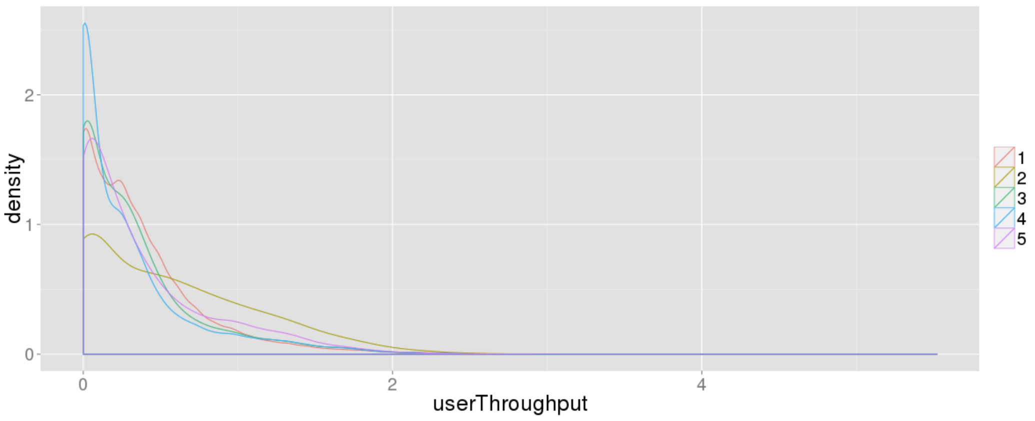


Figure 4.5: Density of user throughput for every cluster, assigned without the help of *user throughput*.

It can be observed that the distribution of *user throughput* differs amongst the clusters even though the feature is not used in the clustering. While $\breve{C}_1^{(1)}$, $\breve{C}_3^{(1)}$ and $\breve{C}_5^{(1)}$ seem to have quite similar distributions, $\breve{C}_2^{(1)}$ and $\breve{C}_4^{(1)}$ seem to follow another distribution. Moreover - it seems possible to identify users with typically higher load demands, as can be observed by $\breve{C}_2^{(1)}$, which would be valuable information for RRM.

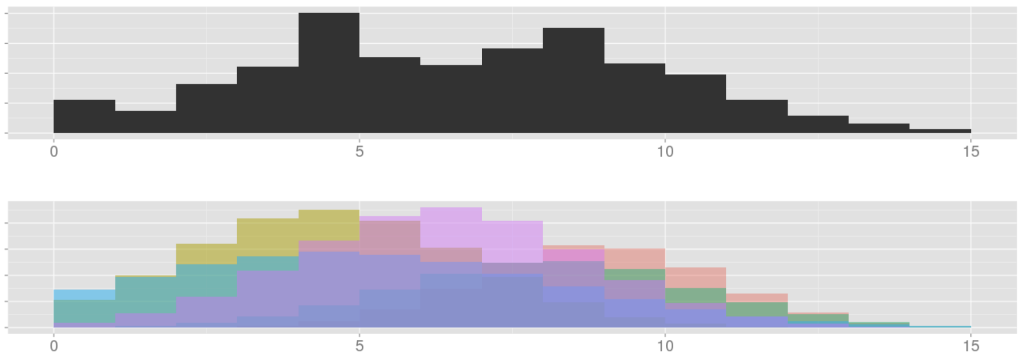
4.1.4 Validation

To investigate whether the model parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are in fact representative of the data, another clustering is performed on a training set of data, being the first

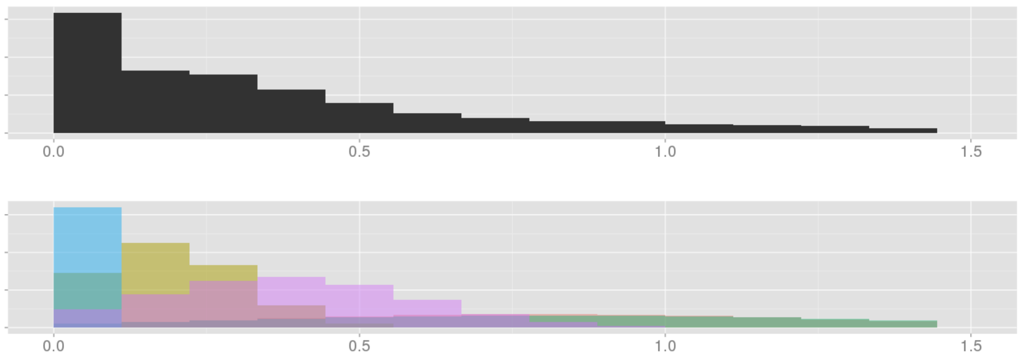
30 minutes, and compared with a test set of data, being the last 30 minutes. Hence, the set notations are

$$\mathcal{C}_j^{\text{train}}(\tau) = \{\mathbf{x} \in \mathcal{C}_j : t \leq \tau\} \quad , \quad \mathcal{C}_j^{\text{test}}(\tau) = \{\mathbf{x} \in \mathcal{C}_j : t > \tau\} \tag{4.2}$$

where $\tau = 1800s$. A comparison of the PDFs from the mixture model of the K clusters' multivariate normal distributions $\mathcal{N}_d(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for each of the d features as obtained from GMM on $\mathcal{C}_j^{\text{train}}$, with the density curves for the corresponding features in $\mathcal{C}_j^{\text{test}}$ can be seen in Figure 4.6. The discrete features *Channel Quality Index* and *number of active users* are graphically presented as histograms instead[§]. In this section, the results of clustering on $\mathcal{C}_1$ are presented. Similar results were found for the other $\mathcal{C}_j, j = 2, \ldots, 33$.



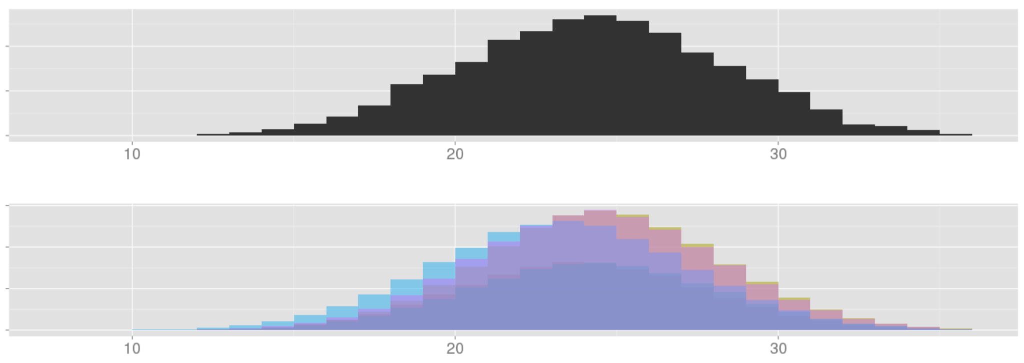
(a) Channel Quality Indicator.



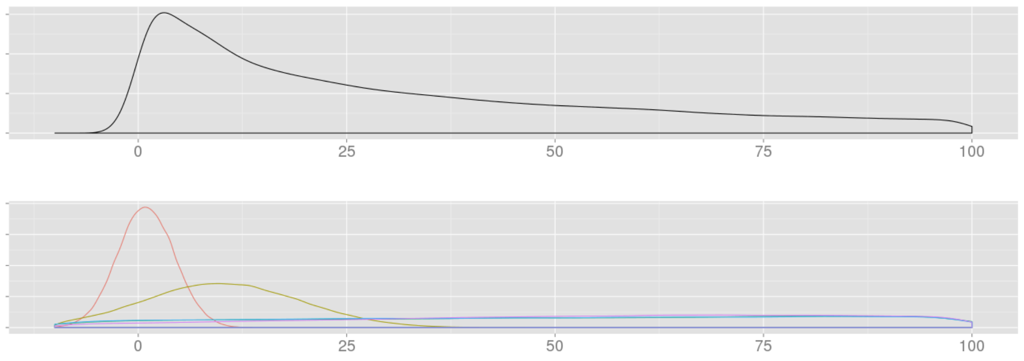
(b) User throughput.

Figure 4.6: Comparison of model built on $\mathcal{C}_1^{\text{train}}$ with data from $\mathcal{C}_1^{\text{test}}$.

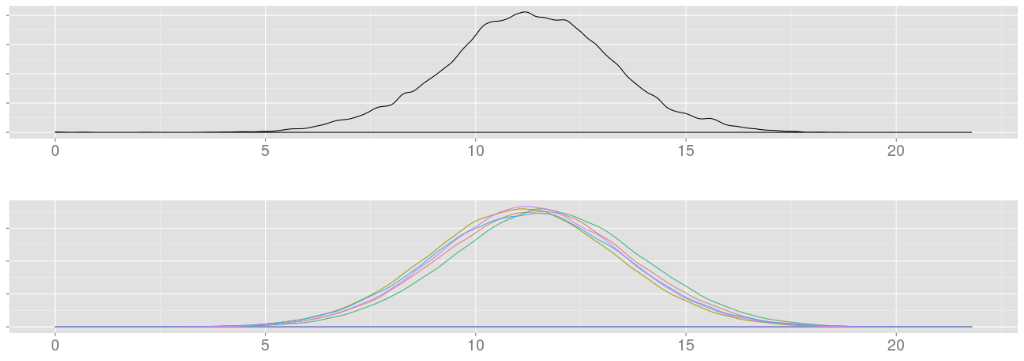
[§]The distributions of *user througput* are plotted as discrete values because of properties of the simulation, yielding values as $0.06n$ where n is an integer. For small values on the x-axis it will therefore appear discrete.



(c) Number of active users.



(d) Time to handover.



(e) Cell throughput.

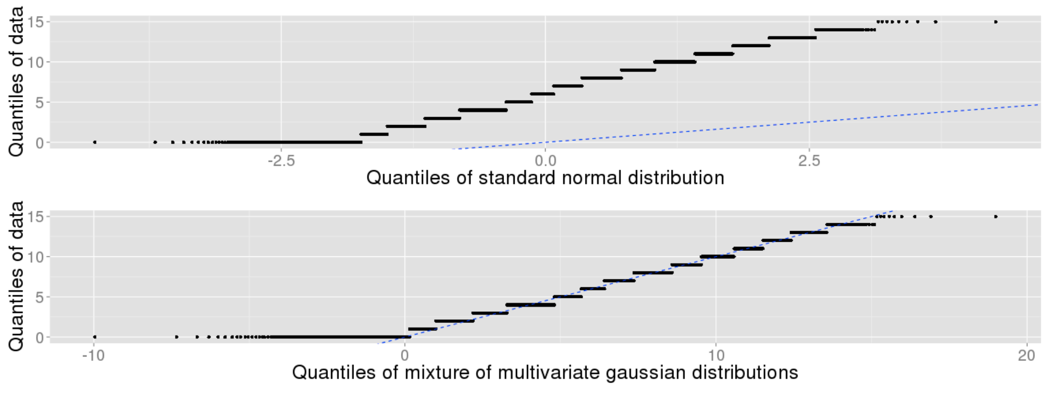
Figure 4.6: Comparison of model built on $\mathcal{C}_1^{\text{train}}$ with data from $\mathcal{C}_1^{\text{test}}$, continued.

Although the model was not built on the data in the test set, it is evident from these figures, that the distributional model is representative. This can be further illustrated by looking at the Q-Q-plots¶ [29] of the features and comparing the quantiles, which is a more robust approach. In Figure 4.7 the quantiles of the d

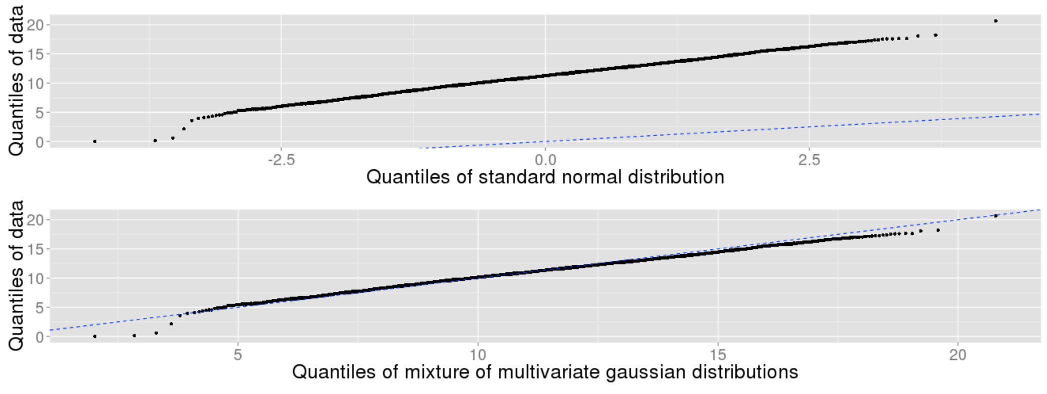
¶Quantile-Quantile plots

features from $\mathcal{C}_j^{\text{test}}$ (on the y-axes) are compared to both a quantiles of a standard normal distribution $\mathcal{N}(0,1)$ and to quantiles of the mixture model of multivariate Gaussian distributions $\mathcal{N}_d(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), k = 1, \ldots, K$ as built on $\mathcal{C}_j^{\text{train}}$.

The underlying theory of a Q-Q-plot is that two similar distributions will yield points located approximately on the identity line $y = x$, whilst if they are a linear combination of one another the points will be located on another straight line. This can be observed in the upper plots in Figure 4.7(a)-(c) to be the case for *Channel Quality Indicator*, *cell throughput* and *number of active users* - suggesting that the distribution of the feature can be modelled as a linear combination of a standard normal distribution. Plotting against the quantiles of the mixture model of multivariate Gaussians however, the quantiles are located on the line $y = x$, justifying that the model found is representative of the data.

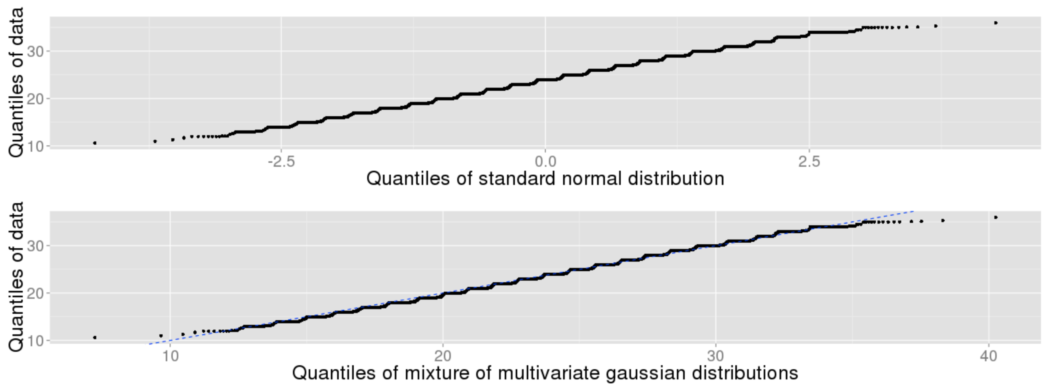


(a) Channel Quality Indicator.

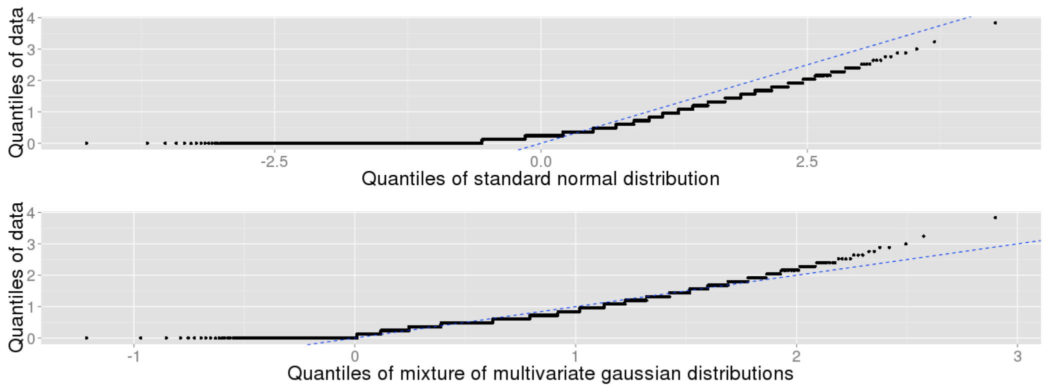


(b) Cell throughput.

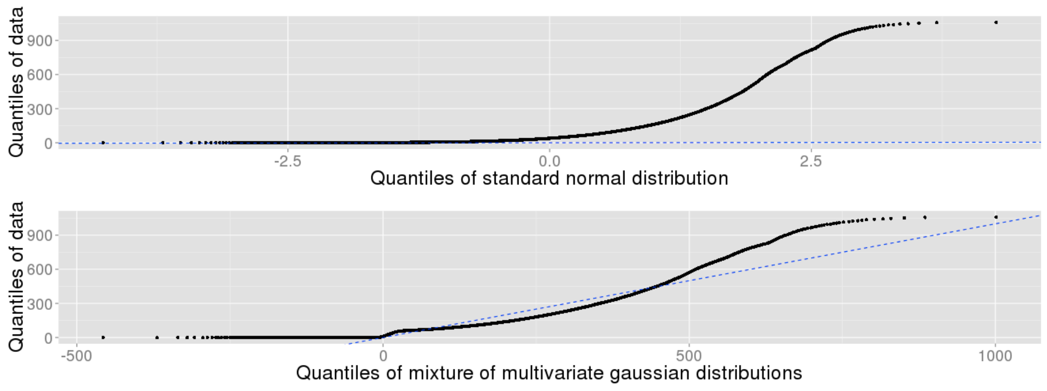
Figure 4.7: QQ-plots of data in $\mathcal{C}_1^{\text{test}}$ vs first a standard normal distribution, second those of the mixture of multivariate Gaussians built on $\mathcal{C}_1^{\text{train}}$.



(c) Number of active users.



(d) User throughput.



(e) Time to handover.

Figure 4.7: QQ-plots of data in $\mathcal{C}_1^{\text{test}}$ vs first a standard normal distribution, second those of the mixture of multivariate Gaussians built on $\mathcal{C}_1^{\text{train}}$, continued.

In the upper plots of Figure 4.7(d)-(e) a non-linear relationship with the quantiles of $\mathcal{N}(0,1)$ can be observed, suggesting that another model might be appropriate for *user throughput* and *time to handover*. Indeed, when evaluating the lower plots, the mixture of multivariate gaussians $\mathcal{N}_d(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ seem to be a better fit for *user* *throughput*. However, as can be interpreted from the upper tail that curves upwards in the lower plots in Figure 4.7(e), there still seems to be some long tailed characteristics in the data that the GMM could not model, due to the property of Gaussians to not have heavy tailed distribution. Another possible reason behind this behaviour is that *time to handover* is obviously time dependent, which might create problems as the training set consist of data from the first 30 minutes and test set of the last. When investigating the QQ-plot of the quantiles of the full data set compared to quantiles of the GMM model built on that data set, more of a linear relationship can be observed, see Figure 4.8. One could argue that to divide $\mathcal{C}_j^{\text{train}}$ and $\mathcal{C}_j^{\text{test}}$ by random sample instead of a time threshold τ might address this. However, heavy tail characteristics are still present when comparing the full data set with GMM, suggesting that the non-linearity in Figure 4.7(e) might originate from both reasons discussed.

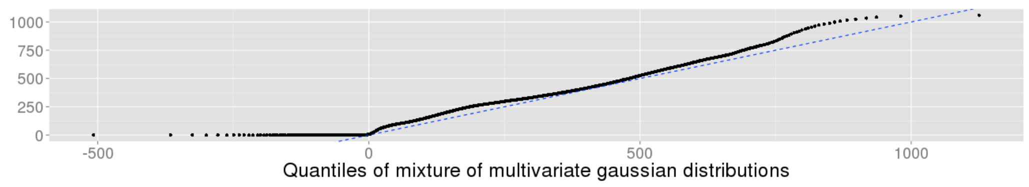


Figure 4.8: Quantiles of *time to handover* in full data set $\mathcal{C}_1$ vs corresponding quantiles of GMM model built on $\mathcal{C}_1$.

4.2 Classification

As explained in Section 3.3, classification based on the Random Forest algorithm is performed on the data set, using the cluster names obtained from the full clustering output as responsive variables. The model is built on training sets $\mathcal{C}_j^{\text{train}}(\tau)$ and evaluated on test sets $\mathcal{C}_j^{\text{test}}(\tau)$ as defined in (4.2). Recall that Random Forest is robust against overfitting, maintaining immune to issues with choosing to many trees to grow. Hence the number of trees was chosen to be the R standard of 500, although similar results were obtained for both a tenth of that amount and, naturally, higher number of trees.

For presentational simplicity, the output from the classification algorithm performed on the subset $\mathcal{C}_1$ is presented. A small, random sample of the data is used in order to increase computational time, and high accuracies was found when using as little as 1% of the data. The confusion matrix for a run where $\tau = 1800s$ can be found in Table 4.2.

As can be seen in Table 4.2, the Random Forest classifier yields a rather accurate result on the test set. It is more interpretable to investigate the test sensitivities, specificities and balanced accuracies, as given by (3.5)-(3.7), which are given in Ta-

		Reference				
		1	2	3	4	5
Prediction	1	9094	15	39	0	6
	2	84	18579	94	79	3
	3	78	98	18354	25	0
	4	0	87	31	32704	102
	5	10	0	0	53	20465

Table 4.2: Confusion matrix for test set for Random Forest classification on $\mathcal{C}_1$.

ble 4.3 for different compilation settings.

As mentioned in Section 3.3.2, one problem with the One-vs-all strategy is that the confidence values might differ between the binary classes. Hence, the mean accuracies of the binary classes, presented in bold, should be complemented by information regarding the accuracies for each class. However, in for most of the compilations run the differences are not of great extent, hence it is encouraged to look the mean values here for simplicity if not stated otherwise.

Another possible issue introduced in Section 3.3.2, is the problem of unbalanced distributions. This is adressed by using the measure of balanced accuracy rather than accuracy, as explained in Section 3.3.1. Nevertheless, one should also investigate both the Sensitivity and Specificity of the classification. The specificity will typically be greater than sensitivity due to the imbalance issue, although the sensitivity is of course more of interest in a one-vs-all classification[‖]. A result of this issue can be observed in Table 4.3, where the specificities are consequently higher than the sensitivities, although the extent of the differences varies. Specifically, the differences are quite high for settings making it harder for the classifier (i.e. excluding *user throughput* or *time to handover*), and seem to be typically greater for the smaller clusters than for the larger ones[**], as expected as the imbalance between the positive and negative class increases.

4.2.1 Varying time frame of training data

One interesting question is how much training data is needed in order to perform a good classification. Therefore, the data sets as denoted in (4.2) are varied with $\tau \in \{60, 600, 1800\}$, denoting 1, 10 and 30 minutes of training data respectively. Note that a sample is used for the classification, leaving $|\mathcal{C}_j^{\text{train}}(60)| = |\mathcal{C}_j^{\text{train}}(600)| = |\mathcal{C}_j^{\text{train}}(1800)|$ and similarly for $|\mathcal{C}_j^{\text{test}}(\tau)|$. Hence it is only the time frame of data

[‖]It is of greater interest to know which cluster the observation is, rather than what cluster it is not.

[**]Recall the size of the clusters as presented in Table 4.1.

that varies.

Upon reducing the time frame training set, no significant differences occur, suggesting that only a short time frame of data, 1 minute, is needed in order to train the decision trees and still get correct classification. Nevertheless, as was discussed in Section 4.1.2, if cluster visits are mainly in the range of $< 1s$, a much shorter time frame of data should be needed in order to make this useful. In this thesis, the data set was too small to perform a classification with such a short time frame of training data, but may be investigated in a future work. The results do still point towards a scenario where training on 1/60 of a time frame yields accurate classification results for the rest of the time frame.

4.2.2 Varying classification variables

The Gini index was introduced in (3.9). Computing the Mean Decrease Gini for the different features yields the results found in Figure 4.9, along with a representation of the Mean Decrease Accuracy.

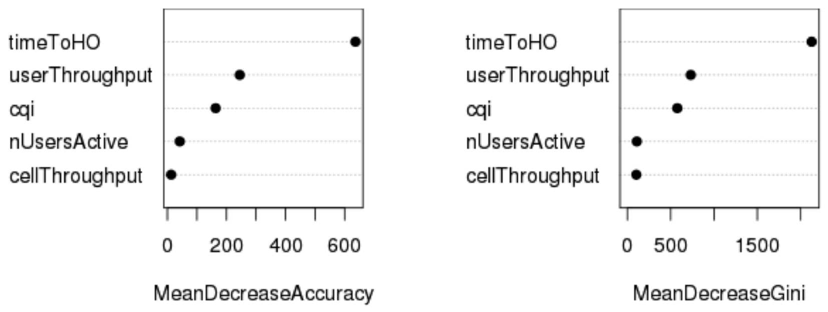


Figure 4.9: Mean Decrease Accuracy and Mean Decrease Gini for the predictor variables.

It was proposed in [28] to complement the Mean Decrease Gini with this value, representing the scenario of randomly permuting the predictor variable in out-of-bag samples. However, the output of both measures seem to be in more or less accordance, and both implies that the predictor variable *time to handover* is of greatest importance for the classifier.

Time to handover

To adress a possible scenario where information about the time to handover is not available, it is investigated how well the classifier would perform without this predictor variable. Furthermore, to gain knowledge about expected time until handover can be valuable from a RRM perspective, in the same manner as for *user throughput*.

It can be seen from Table 4.3 that the exclusion of *time to handover* as a predictor variable leads to a drop of the balanced accuracy to around 0.7 for all τ, with quite high standard deviations. Upon inspecting the mean sensitivities, they are found to be only just above 0.5, also with a rather high standard deviation, suggesting that the individual classes should be evaluated. Indeed, for every τ there are two clusters where the classifier seem to correctly perform a positive classification in a minority of the cases, where $C_1^{(1)}$ is correctly classified in less than 20% of the cases. Recall from Figure 4.2 that $C_1^{(1)}$ seem to represent "heavy hitters", which are valuable to identify from a RRM perspective. Hence, for scenarios where information regarding *time to handover* is not available, the introduction of other possible predictor variables should be evaluated in order to increase accuracy.

User throughput

As mentioned previously, knowledge of expected throughput for a user might be valuable in from a RRM perspective. Therefore, it is investigated whether the classifier still would be able to classify correctly without using *user throughput* as a predictor variable. As can be seen in Table 4.3, the mean balanced accuracies are lower when excluding *user throughput*, though not quite as low as when excluding *time to handover.* This is can be expected from the results found in Figure 4.9.

Because the standard deviation of the mean balanced accuracies and the mean sensitivities are quite high, it is advisable to look at the full table to complement the means in the same manner as for *time to handover.* It is evident that the classifier performs significantly worse - a sensitivity as low as 0.143 is found for $C_5^{(1)}$ when $\tau = 60s$. However, it is does not seem to be quite as problematic for the classifier when given a larger time frame of training data, $\tau \in \{600, 1800\}$, where the sensitivities are kept over 0.38. Moreover, the issues seem to arise in particular for $C_5^{(1)}$, while the sensitivities for the other clusters all are > 0.7 for all τ. Recall from Figure 4.2 that users in $C_5^{(1)}$ tend to have a very low user throughput. In a viable scenario where it is of higher importance to recognise a high load user than a low load user, this classification can still be valuable if a sensitivity > 0.7 is considered acceptable.

Class	Sensitivity	Specificity	Balanced Acc.	$\tau[s]$	CQI	#U	CT	UT	HO
mean	**0.991** $\pm$ **0.005**	**0.998** $\pm$ **.001**	**0.994** $\pm$ **0.003**	1800	X	X	X	X	X
C_1	0.983	0.999	0.991						
C_2	0.989	0.997	0.993						
C_3	0.990	0.998	0.994						
C_4	0.996	0.997	0.996						
C_5	0.995	0.999	0.997						
mean	**0.566** $\pm$ **0.289**	**0.894** $\pm$ **0.054**	**0.730** $\pm$ **0.142**	1800	X	X	X	X	-
C_1	0.193	0.967	0.580						
C_2	0.712	0.914	0.813						
C_3	0.506	0.859	0.682						
C_4	0.458	0.826	0.642						
C_5	0.964	0.905	0.934						
mean	**0.720** $\pm$ **0.192**	**0.931** $\pm$ **0.031**	**0.826** $\pm$ **0.092**	1800	X	X	X	-	X
C_1	0.774	0.976	0.875						
C_2	0.785	0.897	0.841						
C_3	0.785	0.915	0.850						
C_4	0.872	0.922	0.897						
C_5	0.384	0.946	0.665						
mean	**0.989** $\pm$ **0.006**	**0.998** $\pm$ **0.001**	**0.994** $\pm$ **0.003**	600	X	X	X	X	X
C_1	0.979	0.999	0.989						
C_2	0.988	0.997	0.993						
C_3	0.990	0.997	0.994						
C_4	0.994	0.997	0.995						
C_5	0.995	0.999	0.997						
mean	**0.547** $\pm$ **0.289**	**0.892** $\pm$ **0.077**	**0.719** $\pm$ **0.143**	600	X	X	X	X	-
C_1	0.149	0.971	0.560						
C_2	0.712	0.908	0.810						
C_3	0.411	0.888	0.650						
C_4	0.554	0.764	0.660						
C_5	0.908	0.927	0.918						
mean	**0.723** $\pm$ **0.184**	**0.930** $\pm$ **0.026**	**0.827** $\pm$ **0.091**	600	X	X	X	-	X
C_1	0.805	0.967	0.886						
C_2	0.794	0.897	0.846						
C_3	0.759	0.914	0.837						
C_4	0.858	0.932	0.895						
C_5	0.399	0.941	0.671						
mean	**0.948** $\pm$ **0.059**	**0.990** $\pm$ **0.007**	**0.969** $\pm$ **0.028**	60	X	X	X	X	X
C_1	0.844	0.995	0.919						
C_2	0.973	0.990	0.981						
C_3	0.969	0.980	0.974						
C_4	0.984	0.990	0.987						
C_5	0.971	0.997	0.984						
mean	**0.528** $\pm$ **0.222**	**0.885** $\pm$ **0.078**	**0.706** $\pm$ **0.112**	60	X	X	X	X	-
C_1	0.180	0.950	0.565						
C_2	0.689	0.903	0.796						
C_3	0.645	0.797	0.720						
C_4	0.433	0.810	0.621						
C_5	0.693	0.963	0.828						
mean	**0.645** $\pm$ **0.283**	**0.911** $\pm$ **0.053**	**0.778** $\pm$ **0.137**	60	X	X	X	-	X
C_1	0.716	0.968	0.842						
C_2	0.765	0.878	0.821						
C_3	0.792	0.836	0.814						
C_4	0.807	0.942	0.875						
C_5	0.143	0.929	0.536						

Table 4.3: Table of accuracy from random forest classifications. NA: Number of active users, CT: Cell throughput, UT: User throughput, HO: time to handover.

Chapter 5

Conclusions

This thesis aimed to investigate whether one can predict future events based on information regarding a range of features available from a base station. This was done by clustering using Gaussian Mixture Models, which showed different cluster signatures tied to each cluster. The GMM recognised both spatially different typical user groups and user load characteristics without help of such information, suggesting that this kind of model can be used for such predictions.

The time in each cluster was found to be rather low, and based on previous work this might be interpreted to be a result of one main cluster being interrupted by other cluster identities along the time series of observations within a certain time frame, for the different users. Future work might investigate this matter, if such a pattern exists and how long a user is expected to stay in a cluster based on that pattern, in order to enable the usage of classification for prediction of current and next cluster identity and thereby predict user behaviour.

While investigating cluster transitions, some differences were found among the probabilities. However, even greater and more useful differences were found when separating the transitions based on user group, with the argument that some base stations are able to recognise user velocity. Moreover, if the pattern of a main cluster does exist, this might be further improved.

A validation of the GMM found the model viable. However, as discussed in Section 4.1.4, some of the features could be better modelled using other distributional clustering techniques more appropriate for the heavy-tail characteristics of the feature distributions, alternatively clustering techniques not dependent on distribution. Furthermore, other approaches for dividing the data into training set $\mathcal{C}_j^{\text{train}}$ and test set $\mathcal{C}_j^{\text{train}}$ could be evaluated, as the concept of introducing a validation set as well in order to prevent overfitting of the model. Moreover, although the data was found to be representative, one might consider other approaches for modelling this kind of data as some of the features are discrete-valued.

Upon performing a Random Forest classification with the cluster labels as response variables, high accuracies were obtained – even for shorter time frames. Having information of time to handover for a user was found to be viable for obtaining higher accuracies, suggesting that other predictor variables should be introduced in cases where such information is not available. On the other hand, high accuracies were still obtained when using the classifier while excluding the user throughput, suggesting that valuable prediction can be made based on these methods. However, the data set used in this thesis was too small to perform a classification trained on a time frame of data corresponding to seconds, or even milliseconds, in accordance with the obtained time spent in cluster, which might be investigated in a future work.

As addressed in Section 3.1.1, some limitations are evident due to the usage of simulated rather than live data. An evaluation of these matters with a greater data set, with more diversity in both users and user behaviour as well as covering a longer time frame than 60 minutes would most probably yield different results. Nevertheless, if this method is implemented on live data on real base stations, machine learning techniques can be applied to constantly update and aggregate the training set with new information in order to make the cluster all the more sophisticated over time.

References

[1] Ericsson (2015, June). *Ericsson Mobility Report.* `http://www.ericsson.com/res/docs/2015/ericsson-mobility-report-june-2015.pdf`

[2] Zander, J. (1996, May), Radio resource management - an overview. In *Vehicular Technology Conference, 1996. Mobile Technology for the Human Race., IEEE 46th* (Vol. 1, pp. 16-20). IEEE.

[3] Martín-Sacristán, D., Monserrat, J. F., Calabuig, D., & Cardona, N. (2010, April). Mobile Terminal Session SIR Prediction Method Based on Clustering and Classification Algorithms. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE* (pp. 1-5). IEEE.

[4] Keralapura, R., Nucci, A., Zhang, Z., L., & Gao, L. (2010, September). Profiling users in a 3g network using hourglass co-clustering. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking.* (pp. 341-352). ACM.

[5] Laner, M., Svoboda, P., Schwarz, S., & Rupp, M. (2012, April). Users in cells: a data traffic analysis. In *Wireless Communications and Networking Conference (WCNC), 2012 IEEE* (pp. 3063-3068). IEEE.

[6] Paul, U., Subramanian, A.P., Buddhikot, M. M. &Das, S.R. (2011, April). Understanding traffic dynamics in cellular data networks. In *INFOCOM, 2011 Proceedings IEEE* (pp. 882-890). IEEE.

[7] Paul, U., Ortiz, L., Das, S. R., Fusco, G., Buddhikot, M. M. (2014, April). Learning Probabilistic Models of Cellular Network Traffic with Applications to Resource Management. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DYSPAN)* (pp. 82-91). IEEE.

[8] 3GPP. LTE Release 10 (2014, March). `http://www.3gpp.org/ftp/Specs/archive/36_series/36.213/36213-ac0.zip`

[9] PCTEL RF Solutions. Maximizing LTE Performance Through MIMO Optimization. `http://rfsolutions.pctel.com/artifacts/MIMOWhitePaperRevB-FINAL.pdf`

[10] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* Cambridge, MA: The MIT Press. ISBN: 978-0-262-01802-9

[11] Hastie, T., Tibshirani, R. &Friedman, J. (2009). *The elements of statistical learning (Vol. 2, No.1).* New York: Springer.

[12] Flach, P. (2012). *Machine learning: the art and science of algorithms that make sense of data.* Cambridge University Press.

[13] Priebe, C.E. (1994, September). Adaptive Mixtures. In *Journal of the American Statistical Association, 89*(427), 796-806.

[14] McLachlan, G. & Peel, D. (2000). *Finite Mixture Models.* John Wiley & Sons, Inc.

[15] Okabe, A., Boots, B., Sugihara, K., Chiu, S. N. & Kendall, D. G. (2009). *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams* (Vol. 501). John Wiley & Sons, Inc.

[16] R Core Team (2014). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org/`

[17] MATLAB 2014b (2014). The MathWorks, Inc. Natick, Massachusetts.

[18] Altman, N. S. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician, 46*(3), 175-185.

[19] Schwarz, G. (1978). Estimating the dimension of a model *The annals of statistics, 6*(2), 461-464.

[20] Akaike, H. (1974). A new look at the statistical model identification. *Automatic Control, IEEE Transactions on, 19*(6), 716-723.

[21] Tibshirani, R., Walthner, G. & Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 63*(2), 411-423.

[22] Scott, A. & Symons, M. (1971). Clustering methods based on likelihood ratio criteria. *Biometrics*, 387-397.

[23] Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861-874.

[24] Brodersen, K. H., Ong, C.s., Stephan, K. E. & Buhmann, J. M. (2010, August) The balanced accuracy and its posterior distribution. In *Pattern Recognition (ICPR), 2010 20th International Conference on.* (pp. 3121-3124). IEEE.

[25] Bishop, C. M. (2006). *Pattern recognition and machine learning* New York: Springer. ISBN: 978-0-387-31073-2

[26] Caruana, R. & Niculescu-Mizil, A. (2006, June). An Empirical Comparison of Supervised Learning Algorithms. In *Proceedrings of the 23rd international conference on Machine learning.* (pp. 161-168). ACM.

[27] Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems?. *The Journal of Machine Learning Research, 15*(1), 3133-3181.

[28] Breiman, L. (2001, October). Random Forests. *Machine Learning, 45*(1), 5-32.

[29] Thode, H. C. (2002). *Testing For Normality* (Vol. 164). CRC Press.

Appendix A

Mathematical derivations

A.1 Derivation of EM-parameters

We have from (2.7) that the log-likelihood is given by

$$\log l(\lambda) = \sum_{i=1}^{N} \left(\log \sum_{k=1}^{K} w_k g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

Using that $\frac{\partial}{\partial x} \log f(x) = \frac{1}{f(x)} \frac{\partial}{\partial x} f(x)$, and that

$$\frac{\partial}{\partial \boldsymbol{\mu}_k} g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1}$$

taking the derivative with respect to $\boldsymbol{\mu}_j$ yields

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\mu}_k} \log l(\lambda) &= \sum_{i=1}^{N} \frac{1}{\sum_{k=1}^{K} w_k g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \frac{\partial}{\partial \boldsymbol{\mu}_k} \left(\sum_{k=1}^{K} w_k g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \\
&= \sum_{i=1}^{N} \frac{1}{\sum_{k=1}^{K} w_k g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} w_k g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} \\
&= \sum_{i=1}^{N} \gamma_{i,k} (\mathbf{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1}
\end{aligned}$$

Setting this equal to 0 to find the optimal $\boldsymbol{\mu}_k$ yields

$$\sum_{i=1}^{N} \gamma_{i,k} \mathbf{x}_i = \sum_{i=1}^{N} \gamma_{i,k} \boldsymbol{\mu}_k \implies \bar{\boldsymbol{\mu}}_k = \frac{\sum_{i=1}^{N} \gamma_{i,k} \cdot \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_{i,k}}$$

In a similar manner, we first obtain the derivative of $g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with respect to $\boldsymbol{\Sigma}_k$ is

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_k} g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = g(\mathbf{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \frac{1}{2} \left(-\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} \right)$$

Hence, we obtain the derivative of the log-likelihood to be

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\Sigma}_j} \log l(\lambda) &= \sum_{i=1}^{N} \frac{w_k g(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k=1}^{K} w_k g(\mathbf{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \frac{1}{2} \left(-\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}\right)\\ &= \sum_{i=1}^{N} \gamma_{i,k} \frac{1}{2} \left(-\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}\right)\end{aligned}$$

setting this equal to zero yields

$$\begin{aligned}0 &= \sum_{i=1}^{N} \gamma_{i,k} \frac{1}{2} \left(-\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)'\boldsymbol{\Sigma}_k^{-1}\right)\\ &= \sum_{i=1}^{N} \gamma_{i,k} \left(-\boldsymbol{\Sigma}_k\boldsymbol{\Sigma}_k^{-1} + \boldsymbol{\Sigma}_k\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)'\boldsymbol{\Sigma}_k^{-1}\right)\end{aligned}$$

and solving for $\boldsymbol{\Sigma}_k$ yields

$$\sum_{i=1}^{N} \gamma_{i,k} = \sum_{i=1}^{N} \gamma_{i,k}(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)'\boldsymbol{\Sigma}_k^{-1} \implies \bar{\boldsymbol{\Sigma}}_k = \frac{\sum_{i=1}^{N} \gamma_{i,k} \cdot (\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \bar{\boldsymbol{\mu}}_k)'}{\sum_{i=1}^{N} \gamma_{i,k}}$$

We have the constraint that $\sum_{k=1}^{K} w_k = 1$, which can be handled by expressing w_k in terms of unconstrained variables β_k such that

$$w_k = \frac{\exp \beta_k}{\sum_{k=1}^{K} \exp \beta_k}$$

from which the derivative with respect to the unconstrained variables can be obtained to be

$$\frac{\partial w_k}{\partial \beta_j} = \begin{cases} w_k - w_k^2 & \text{if } j = k \\ -p_j p_k & \text{otherwise} \end{cases}$$

Using this, and the chain rule of differentiation, we have

$$\frac{\partial \log l(\lambda)}{\partial \beta_k} = \frac{\partial \log l(\lambda)}{\partial w_k} \frac{\partial w_k}{\partial \beta_k} = \sum_{i=1}^{N} (\gamma_{i,k} - w_k)$$

Setting this equal to 0 to find the optimal w_k yields

$$\bar{w_k} = \frac{1}{N} \sum_{i=1}^{N} \gamma_{i,k}$$