Multivariate statistical analysis of thematic changes in customer feedback

Master's Thesis

Mikko Lopperi

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ
Department of Mathematics and Statistics

6th July 2023

Author
Mikko Lopperi

Title
Multivariate statistical analysis of thematic changes in customer feedback

Degree programme
Master's Degree Programme in Statistics and Data Science

Date	**Pages**
6th July 2023	$54 + 10$

Abstract

This thesis conducts a multivariate statistical analysis of thematic changes in customer feedback, primarily focusing on multivariate methods. The study data were obtained from Aiwo Digital Oy, which received it from their client companies. The analysis focused on pseudonymized binary-coded theme variables, which indicate whether the theme occurred in an individual feedback. In addition to themes, there were also background variables, and sentiment, which indicated the tone of the feedback.

The primary goal was to group themes that behaved similarly over the study period. We applied hierarchical clustering to group the binary multivariate data. The thesis discusses various similarity measures between binary theme vectors and dissimilarity measures between clusters. The gap statistic and the silhouette coefficient were considered criteria for choosing an appropriate number of clusters. We clustered 79 theme variables into two groups. We aggregated data on a weekly basis and investigated the theme occurrences of different theme groups. Finally, we discovered seven themes (Group 1) that exhibited similar behavior throughout the study period.

We discussed the theory of metric multidimensional scaling (MDS) and applied metric MDS to visualize the multidimensional theme data in a low-dimensional space. We calculated bootstrap confidence intervals for theme occurrences. Through an investigation of the confidence intervals, we discovered that not all changes in Group 1 appeared to be solely due to natural variation in the data. We applied negative binomial regression to model theme counts depending on the week and the sentiment. Feedback in which themes of Group 1 occurred appeared to be primarily negative.

For an interpretation of the results, after the study, we were given the real themes behind the pseudonymized themes of Group 1 by Aiwo. Five

themes in Group 1 related to usability and two to customer service. We concluded that the changes in these themes were likely due to the change in the user interface or in the method of use of some client applications. The negative feedback may indicate how the changes in usability have been received. Still, it is also worth noting that negative feedback is typically received when something does not function as expected. A thorough analysis of the customer feedback at the text level or the client's assessment would be necessary for a more accurate interpretation.

Keywords: themes of customer feedback, binary data, hierarchical clustering, the gap statistic, the silhouette coefficient, multidimensional scaling, bootstrap confidence intervals, negative binomial regression

Tekijä
Mikko Lopperi

Otsikko
Tilastollinen monimuuttuja-analyysi asiakaspalautteiden teemamuutoksista

Tutkinto-ohjelma
Tilastotieteen ja datatieteen maisteriohjelma

Päivämäärä	**Sivumäärä**
6.7.2023	$54 + 10$

Tiivistelmä

Tässä opinnäytetyössä tehdään asiakaspalautteen teemamuutosten tilastollinen monimuuttuja-analyysi keskittyen ensisijaisesti monimuuttujamenetelmiin. Tutkimusaineisto on hankittu Aiwo Digital Oy:ltä, joka on saanut aineiston asiakasyrityksiltään. Analyysi keskittyi pseudonymisoituihin teemamuuttujiin, jotka ovat binäärikoodattuja, ja osoittavat, esiintyikö teema yksittäisessä palautteessa. Teemojen lisäksi datassa oli taustamuuttujia ja tunne, joka ilmaisi palautteen sävyä.

Ensisijaisena tavoitteena oli ryhmitellä teemat, jotka käyttäytyivät samalla tavalla tutkimusjakson aikana. Käytimme hierarkkista ryhmittelyä binäärisen monimuuttujadatan ryhmittelemiseen. Opinnäytetyössä tarkastellaan erilaisia samanlaisuusmittoja binääristen teemavektoreiden välillä ja erilaisuusmittoja ryhmien välillä. Aukkosuuretta ja siluettisuuretta tarkasteltiin kriteereinä optimaalisen ryhmämäärän valintaan. Ryhmittelimme 79 teemamuuttujaa kahteen ryhmään. Aggregoimme päivittäisen datan viikkotasolle ja tutkimme eri teemaryhmien teemaesiintymiä. Löysimme seitsemän teemaa (ryhmä 1), jotka osoittivat samanlaista käyttäytymistä koko tutkimusjakson ajan.

Käsittelimme metrisen moniulotteisen skaalauksen (MDS) teoriaa ja käytimme MDS:ää moniulotteisen teemadatan visualisointiin matalaulotteisessa avaruudessa. Laskimme uusioluottamusvälit teemaesiintymille. Tutkimalla luottamusvälejä havaitsimme, että kaikki ryhmän 1 muutokset eivät näyttäneet johtuvan ainoastaan satunnaisesta vaihtelusta. Käytimme negatiivista binomiregressiota temaesiintymien mallintamiseen viikosta ja tunteesta riippuen. Palaute, jossa ryhmän 1 teemoja esiintyi, oli enimmäkseen negatiivista.

Tulosten tulkintaa varten, saimme Aiwolta jälkikäteen tiedot todellisista

teemoista, jotka olivat ryhmän 1 pseudonymisoitujen teemojen takana. Viisi teemaa ryhmässä 1 liittyi käytettävyyteen ja kaksi asiakaspalveluun. Päättelimme, että näiden teemojen muutokset saattoivat johtua käyttöliittymän tai joidenkin asiakassovellusten käyttötavan muutoksesta. Negatiivinen palaute voi indikoida, miten käytettävyyden muutokset on otettu vastaan. On syytä myös huomioida, että merkittävä määrä negatiivista palautetta annetaan tyypillisesti silloin, kun jokin ei toimi odotetulla tavalla. Tarkempi tulkinta vaatisi asiakaspalautteiden analysointia tekstitasolla tai asiakasyrityksen omaa arviota.

Avainsanat: asiakaspalautteiden teemat, binäärinen data, hierarkkinen ryhmittely, aukkosuure, siluettisuure, moniulotteinen skaalaus, uusioluottamusvälit, negatiivinen binomiregressio

Contents

1 Introduction

This thesis aims to conduct a multivariate statistical analysis of thematic changes in customer feedback. The research topic was proposed by Aiwo Digital Oy, the company that inspired this study. In this chapter, we first briefly describe the background of the Aiwo service based on Häkkinen (2023) and then go through the aims of this thesis.

Aiwo offers AI-powered analytics with which their client companies can detect phenomena relating to their business or organization's operations in real-time from natural language data masses. Customer contact information or free-form feedback from personnel surveys are usually used as data sources. The service enables the identification of customer or organizational phenomena. Important data sources are customer service phone conversations, transcribed into text before analysis.

The Aiwo service analyzes the data and its associated metadata consisting of background information such as the age and gender of the feedback giver. The service characterizes natural language data within various semantic contexts, such as themes and sentiment. The service classifies the text or its parts based on expert-taught data in language and content areas, which are used to train classification models. The most detailed classification is achieved through thematic categories, which divide the text or its parts according to the narrative's topic. Typically, there are around a hundred theme categories, but this can vary widely depending on the client company. Typical themes include pricing, contracts, offers, terms of payment, competing products, bidding, competition, new customers, and themes related to product groups and individual products. In addition to thematic classification, the service can classify feedback according to its tone as positive, negative, or neutral. The corresponding three-level classification is the classification of failure demand in customer communication, which can be used either alongside or in place of sentiment classification.

In a continuous real-time service, Aiwo receives customer data through interfaces. An important aspect of the analysis is the detection of trends and temporal deviations. Typically, trends are reflected in client operating environment changes, which can be observed through customer feedback. Detecting such trends and temporal deviations is crucial to understanding different factors' impact on customer behavior. Client companies can access the service through a web-based interface or by integrating their system with the service API to receive analysis results. The web-based user interface offers online analytics, enabling users to view the analysis results using different search criteria, grouping, and aggregation based on the text clas-

sifications described earlier, background variables, and comparison of time intervals. A user can freeze a view that they find interesting so that they can repeat the same search, grouping, and aggregation at new time intervals. However, effective use of online analytics requires a thorough understanding of the client's content area and experimental study of the analysis results. Computational support mainly provides values describing variable-specific changes. While this tool can already detect significant phenomena from the analysis result, this online analytics tool could be enhanced with computational methods highlighting smaller but significant considerations from the analysis result.

This thesis's central challenge is identifying temporal changes in the relationships between theme categories and tone categories using statistical methods. The analysis focuses on categorical time series data obtained from text classification. In the data, events are recorded daily, but the technique should make a comparison between longer periods. In most cases, a week is the shortest period where changes might occur, but a month or even a quarter is often an interesting time window for detecting changes. Phenomena are often reflected in changes across multiple theme, sentiment, and failure demand categories. Additionally, background variables may explain or help allocate the shift to a specific group. Currently, identifying simultaneous multivariate changes using an exploratory web-based user interface is practically difficult, and therefore support from computational methods is sought for this. Systematically working through the data to find dependencies in changes is a tedious process, and this task could be automated using computational methods, making it more efficient.

The relative size of changes has been identified as a challenge based on customer feedback. Large client companies have extensive organizations and product ranges, resulting in many theme categories in customer service data. In such cases, significant changes in the data for the clients are often in the order of a few percent. In large customer volumes, changes of that size can be economically significant. We aim to determine whether these changes are within the limits of natural variation. This thesis seeks a method or set of techniques to serve as a decision-making support tool.

The thesis aims to explore, test and review multivariate methods which can help to identify thematic groups that behave in the same way over time. The goal of this work is to provide information about strategies that can be effective or ineffective. Customer feedback provides valuable information on improving the quality of products and services and the overall customer experience. In a broader sense, this thesis aims to enhance the customer experience by developing and implementing effective methods for analyzing

and utilizing customer feedback data.

Pearson (2018) lists three motivations for analyzing data: to understand events, predict outcomes, and guide decision-making. The thesis aims to examine how changes in the client's operating environment affect the themes of customer feedback. The emphasis is on exploratory data analysis and descriptive techniques rather than predictive modeling. The analysis was carried out with the R programming language (see Appendix).

This thesis mainly focuses on multivariate methods, although other statistical methods are also discussed. We will end the introduction with a brief overview of the remaining chapters in the thesis. Chapter 2 describes and visualizes the data to get familiar with the subject. Chapter 3 focuses on clustering methods, which are used to find thematic groups that behave similarly over time. Chapter 4 introduces and applies the multidimensional scaling theory to visualize multidimensional data in lower-dimensional space. In Chapter 5, we involve bootstrapping to define confidence intervals for theme occurrences to describe the natural variation in the data. In Chapter 6, we use statistical modeling in the form of negative binomial regression. Finally, in Chapter 7, we summarize our findings and draw conclusions.

2 Data exploration

The study data was customer data obtained from Aiwo Digital Oy. The data included theme categories and sentiment classified from real customer feedback. Anyhow, the names of all themes had been pseudonymized for privacy reasons, which means that we do not have any domain knowledge when performing the analysis. The original dataset included 133844 observation records, each representing feedback from an individual customer. The features included 79 themes and information on the date, city, age group, and sentiment.

Although the data was initially collected daily, the high resolution proved too fine and resulted in patterns being obscured by noise. In addition, a weekly pattern was observed, with fewer feedback responses received on weekends than on weekdays (Fig. 2.1). For some of the analysis, the data was aggregated on a weekly basis to provide a more apprehensible analysis (Fig. 2.2). Since the observations from the first and last week only covered partial weeks, we excluded them to ensure the consistency of our analysis on a weekly basis. After this, the data contained information from customer feedback given during six months period, starting from week 48 of 2021 and ending in week 10 of 2022. The dataset consisted of 124836 observations,

covering 15 full weeks of data.

Themes were classified as binary variables denoted as $x_1, x_2, ..., x_{79}$, indicating a value of one if the theme occurred and zero otherwise. Due to the large number of zero values for the theme variables, the data can be considered sparse regarding its information content. The sentiment of the feedback was measured as a three-class variable, with a value of minus one indicating negative feedback, zero indicating neutral feedback, and one indicating positive feedback. The dataset included 63208 negative, 5885 neutral, and 55743 positive feedback, so feedback was generally classified as either positive or negative. The city variable refers to the specific business unit associated with the feedback, and the age group refers to the age of the feedback giver. Fig. 2.1 and Fig. 2.2 display the daily and weekly sums of the themes, respectively. When these figures were examined, it appeared that using weekly smoothed data would be more appropriate to conclude our analysis. We performed clustering, multidimensional scaling, and negative binomial regression with the original data, where each observation represented individual feedback. We aggregated data on a weekly basis for the calculation of confidence intervals and interpretation purposes.

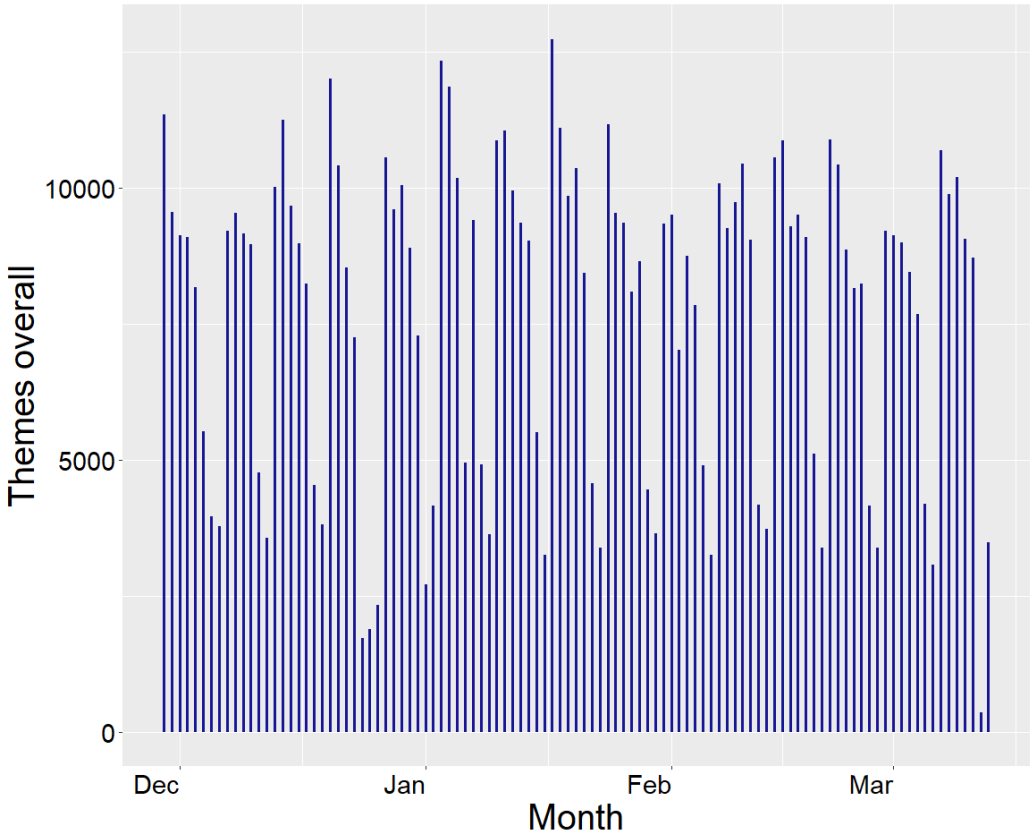

Figure 2.1: Overall daily feedback themes

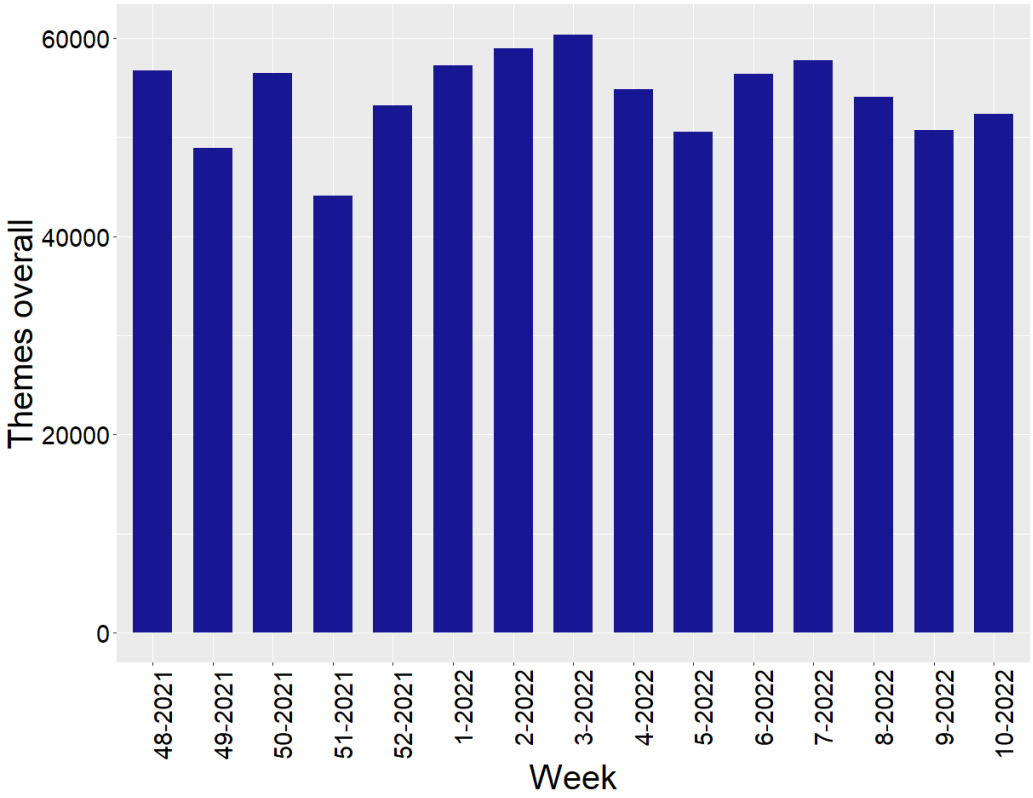

Figure 2.2: Overall weekly feedback themes

Fig. 2.2 visualizes weekly changes in overall theme occurrences. The most significant change occurred between week 50 of 2021 and week 51 of 2021 due to Christmas Eve falling on a Friday, resulting in less feedback received during weekdays of week 51.

3 Clustering methods

In this chapter, we discuss clustering methods. The analysis focused on 79 thematic variables to identify those that exhibited similar behavior throughout the study period. We applied clustering for this purpose and dimension reduction.

Clustering is an unsupervised statistical learning approach that refers to a broad class of methods for finding subgroups in a data set (James et al., 2021). Unsupervised learning aims to find new and interesting patterns and

insights from the data (Nwanganga and Chapple, 2020). Clustering is a tool for analyzing unstructured multivariate data (Izenman, 2008). The fundamental aim of clustering is to discover 'natural groupings' of a set of individuals (Chatfield and Collins, 1980). It can serve as either an exploratory task or a preprocessing step, depending on the specific application and goals of the analysis (Aggarwal and Reddy, 2014). Unlabeled data is partitioned into subgroups based on a chosen similarity metric in clustering. The two objectives of clustering are high intraclass similarity and low interclass similarity, which means that the similarity of items within a particular cluster is maximized, and the difference between items within one cluster and another cluster is maximized (Nwanganga and Chapple, 2020).

Clustering has been applied in diverse fields, such as life sciences, behavioral and social sciences, earth sciences, medicine, engineering sciences, and information and policy sciences (Anderberg, 2014). Examples of practical, real-world applications are identifying customer segments based on their purchasing behavior or demographics or detecting anomalous activity within networks (Nwanganga and Chapple, 2020). The two fundamental steps in cluster analysis are the choice of similarity measure, which we discuss next, and the choice of clustering algorithm (Härdle and Simar, 2014).

3.1 Similarity measures for binary vectors

The choice of similarity measure is often a domain-specific consideration where we define what it means for two or more items to be similar. It must be based on knowledge of the data being studied (James et al., 2021). In this section, we discuss the similarity of binary vectors. A proximity measure is needed when performing cluster analysis. The proximity measure can be a measure of similarity or dissimilarity, and for binary data, it is customary to use a similarity measure (Chatfield and Collins, 1980; Härdle and Simar, 2014). In this study, we use a hierarchical clustering algorithm that takes a dissimilarity matrix as an input. We define possible similarity measures, then calculate a similarity matrix and convert it into a dissimilarity matrix.

A similarity coefficient $s(i, j)$ is usually symmetric and takes on values between 0 and 1 depending on how close objects i and j are. When the value is 1, it means that objects are identical. Value 0 means that they are not similar at all, and for values falling between 0 and 1, there are varying degrees of similarity. Hereafter, we assume that the following requirements are met:

$$0 \leq s(i, j) \leq 1,$$

$$s(i, i) = 1,$$

$$s(i,j) = s(j,i)$$

for all objects i and j (Kaufman and Rousseeuw, 2009).

To measure the similarity between binary vectors, we evaluate pairs of observations (x_i, x_j) against each other, where $x_i^T = (x_{i1}, ..., x_{ip})$, $x_j^T = (x_{j1}, ..., x_{jp})$, and $x_{ik}, x_{jk} \in \{0, 1\}$. Following Härdle and Simar (2014), we consider four cases:

$$x_{ik} = x_{jk} = 1,$$

$$x_{ik} = 0, x_{jk} = 1,$$

$$x_{ik} = 1, x_{jk} = 0,$$

$$x_{ik} = x_{jk} = 0.$$

Define

$$a = \sum_{k=1}^{p} I(x_{ik} = x_{jk} = 1),$$

$$b = \sum_{k=1}^{p} I(x_{ik} = 0, x_{jk} = 1),$$

$$c = \sum_{k=1}^{p} I(x_{ik} = 1, x_{jk} = 0),$$

$$d = \sum_{k=1}^{p} I(x_{ik} = x_{jk} = 0).$$

There are many different similarity coefficients defined for binary objects in the literature. This study considers four common similarity measures for binary objects: Pearson's correlation coefficient, the simple matching coefficient, Jaccard's coefficient, and Yule's Q coefficient (see Řezanková and Everitt, 2009), which are defined as follows:

$$s_{Pearson} = \frac{ad - bc}{\sqrt{(a+b) + (a+c) + (b+d) + (c+d)}}, \tag{1}$$

$$s_{SM} = \frac{a+d}{a+b+c+d}, \tag{2}$$

$$s_{Jaccard} = \frac{a}{a+b+c}, \tag{3}$$

$$s_Q = \frac{ad - bc}{ad + bc}. \tag{4}$$

Definitions for many more different binary similarity and distance measures can be found in Choi et al. (2010).

The similarities can be arranged in a matrix $S(n \times n)$. A proximity matrix is a commonly used term for similarity and dissimilarity matrices (Kaufman and Rousseeuw, 2009). Since we want to apply clustering algorithms designed for dissimilarities, we need a transformation from similarity measure $s(i, j)$ to dissimilarity measure $d(i, j)$. Possible transformations are defined below (Kaufman and Rousseeuw, 2009; Cox and Cox, 2001).

$$d(i, j) = 1 - s(i, j), \tag{5}$$

$$d(i, j) = c - s(i, j) \text{ for some constant c}, \tag{6}$$

$$d(i, j) = \sqrt{2(1 - s(i, j))}. \tag{7}$$

The choice of the transformation will vary based on the nature of the problem (Cox and Cox, 2001).

The dissimilarity matrix D with elements $d_{ij} = \sqrt{1 - s_{ij}}$ has Euclidean properties if the similarity matrix S with elements $0 \leq s_{ij} \leq 1$ and $s_{ii} = 1$ is positive semidefinite (Cox and Cox, 2001). A symmetric matrix is positive definite if all of its eigenvalues are positive, and a positive semidefinite matrix also allows eigenvalues $\lambda = 0$ (Strang, 2016).

In practice, we calculated the similarity between x_i^T and x_j^T for all $i = 1, 2, ..., 79$ and $j = 1, 2, ..., 79$ using Formulas $1 - 4$, resulting four different 79×79 similarity matrices. We verified the positive-semidefiniteness of these matrices using R package 'matrixcalc' (Novomestky and Kelly, 2022). The similarity matrices based on Pearson's correlation coefficient, the simple matching coefficient, and Jaccard's coefficient were positive semidefinite, while the similarity matrix based on Yule's Q coefficient did not meet this criterion.

We then transformed positive semidefinite similarity matrices S into dissimilarity matrices D using Formula 7. We selected Formula 7 for the transformation as it results in dissimilarity matrices with Euclidean properties, which are required for methods such as metric multidimensional scaling (Cox and Cox, 2001). As a result, we obtained three different dissimilarity matrix options with Euclidean properties for later use.

3.2 Clustering algorithms

Next, we focus on choosing a clustering algorithm, discussing possibilities, and determining which best suits our binary data. Partitional clustering and hierarchical clustering are the most widely studied clustering algorithms (Aggarwal and Reddy, 2014). The primary difference between the two algorithms is that partitional techniques allow the assignment into groups to change during the process. In contrast, hierarchical clustering cannot change the assignment once groups are found (Härdle and Simar, 2014). Partitional clustering algorithms aim to divide the data into clusters by optimizing a specific objective function and improving the partitions' quality (Aggarwal and Reddy, 2014). The most popular partitional clustering algorithm is k-means clustering (Aggarwal and Reddy, 2014).

The choice of clustering algorithm is heavily influenced by the nature of the data type being analyzed (Aggarwal and Reddy, 2014). Theme data of this study is a special case of categorical data in which all attributes are binary. Clustering algorithms face significant challenges when working with categorical data sets (Aggarwal and Reddy, 2014). Binary variables are sometimes treated as interval-scaled using the usual formulas for distance metrics like Euclidean or Manhattan distance. Although standard formulas may yield satisfactory results in some cases, it is worthwhile to keep in mind that there are approaches designed specifically for binary data analysis (Kaufman and Rousseeuw, 2009). Another difficulty arises when clustering categorical data since specific methods, like k-means or k-medians, compute cluster representatives by calculating the means or medians of the data points in each cluster. However, while these statistics are naturally defined for continuous data, their use with discrete data calls for specific adaptations (Aggarwal and Reddy, 2014). This study focused mainly on hierarchical clustering, which we explain further in the following section.

3.3 Hierarchical clustering

For the binary data of this study, we chose to use the hierarchical algorithm, which is discussed in this section.

Hierarchical clustering can be divided into two categories: agglomerative and divisive methods (Sasirekha and Baby, 2013). Agglomerative processes operate with a bottom-up approach starting from the finest partition. Each data object forms a cluster, and the two clusters with the closest distance merge into one cluster. This procedure is repeated until all objects are agglomerated into one cluster. Divisive methods operate with a top-down approach starting with the coarsest partition possible, where all data objects are

in the same cluster, and the single cluster splits into smaller-sized clusters. This procedure is repeated until each object forms its own cluster (Sasirekha and Baby, 2013; Härdle and Simar, 2014). Next, we will discuss more about agglomerative hierarchical methods which are used in this study.

We need a proximity measure for agglomerative hierarchical clustering to decide which clusters should be combined. The proximity measure represents a metric of dissimilarity between sets of objects. We already have a dissimilarity metric between the pairs of objects, but we still need a linkage criterion that defines the dissimilarity between two groups (Sasirekha and Baby, 2013). Next, we describe some standard distance measures between two clusters denoted as C_i and C_j.

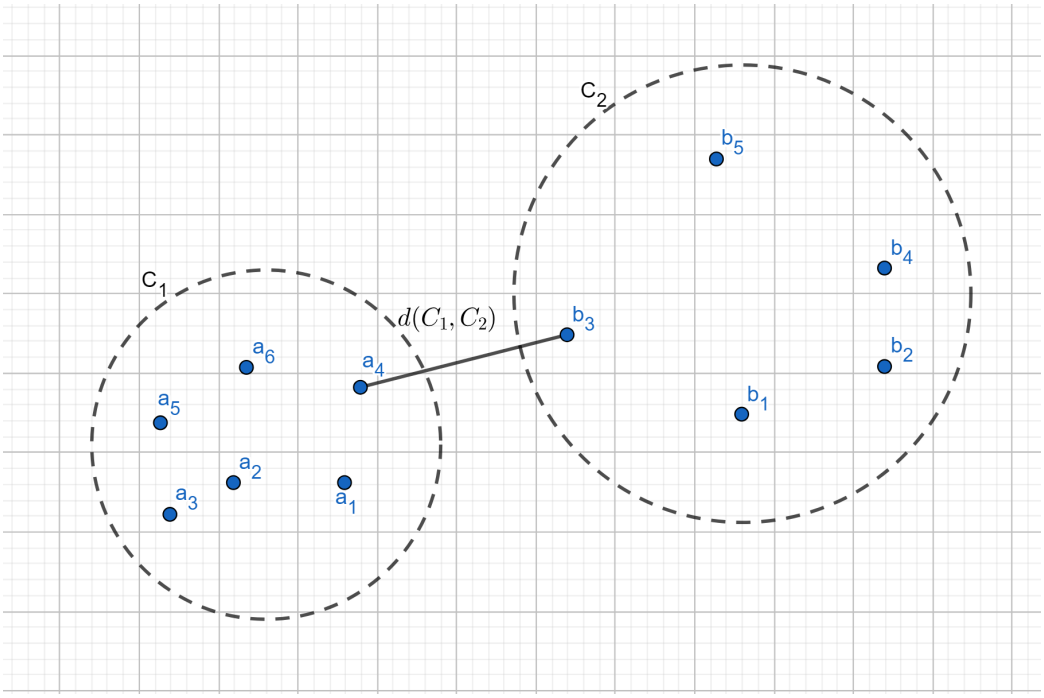

Figure 3.1: In the single linkage, the distance between two groups is defined as the minimum distance between two objects in each group (Zaki and Meira, 2014)

- **Single linkage:** The minimum distance between an object in C_i and an object in C_j:

$$d(C_i, C_j) = \min_{a \in C_i, b \in C_j} d(a, b)$$

(Zaki and Meira, 2014). See Fig. 3.1 for an illustration.

The single linkage algorithm, which is also known as the nearest neighbor algorithm (Härdle and Simar, 2014), calculates the pairwise distances between all objects in cluster C_i and all objects in cluster C_j, selecting the smallest value (James et al., 2021). The construction of a single linkage tends to result in the formation of large clusters (Härdle and Simar, 2014), in which single objects are fused one at a time (James et al., 2021).

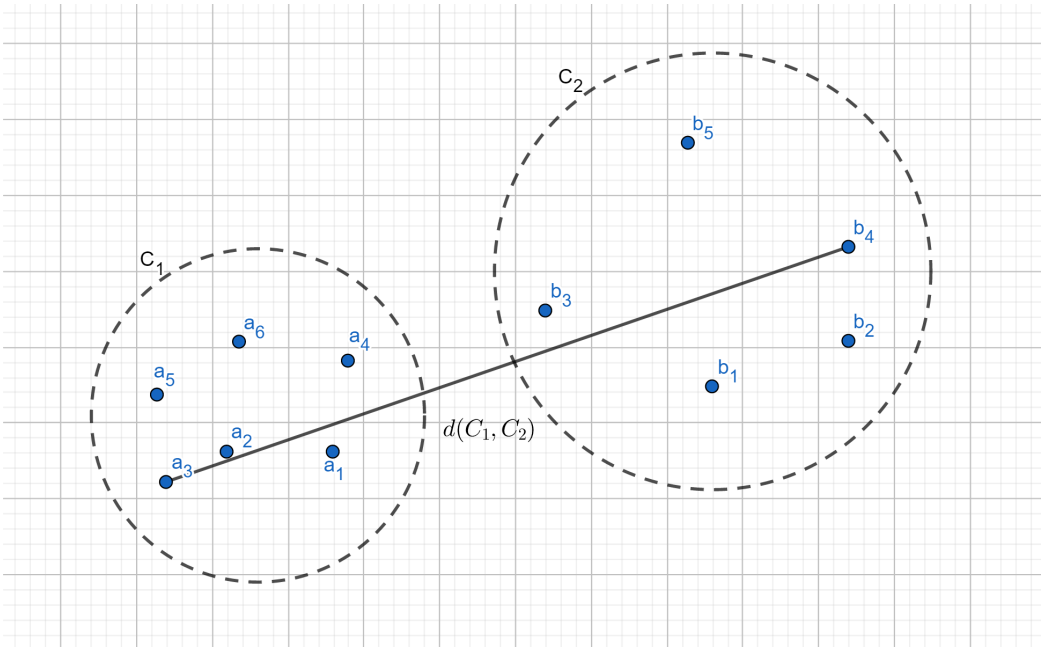

Figure 3.2: In the complete linkage, the distance between two groups is defined as the maximum distance between two objects in each group (Zaki and Meira, 2014)

- **Complete linkage:** The maximum distance between an object in C_i and an object in C_j:

$$d(C_i, C_j) = \max_{a \in C_i, b \in C_j} d(a, b)$$

(Zaki and Meira, 2014). See Fig. 3.2 for an illustration.

The complete linkage algorithm, which is also known as the farthest neighbor algorithm (Härdle and Simar, 2014), calculates the pairwise distances between all objects in cluster C_i and all objects in cluster C_j, selecting the largest value (James et al., 2021). This kind of construction tends to result in groups where all points are close to each other (Härdle and Simar, 2014).

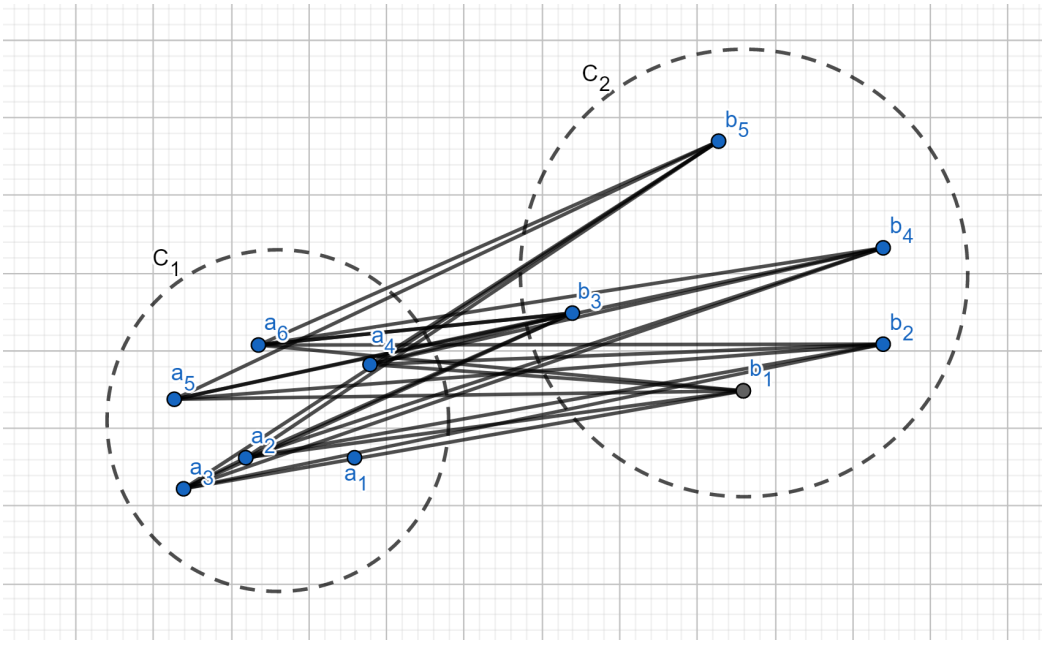

Figure 3.3: In the average linkage, the distance between two groups is defined as the average distance between two objects in each group (Zaki and Meira, 2014)

- **Average linkage:** The average pairwise distance between objects in C_i and C_j:

$$d(C_i, C_j) = \frac{1}{|C_i|\,|C_j|} \sum_{a \in C_i, b \in C_j} d(a, b),$$

  where $|C_i|$ is the number of objects in cluster C_i (Zaki and Meira, 2014). See Fig. 3.3 for an illustration.

The average linkage algorithm calculates the pairwise distances between all objects in cluster C_i and all objects in cluster C_j, recording the average of these values (James et al., 2021). This algorithm can be seen as a compromise between the single and the complete linkage algorithms (Härdle and Simar, 2014).

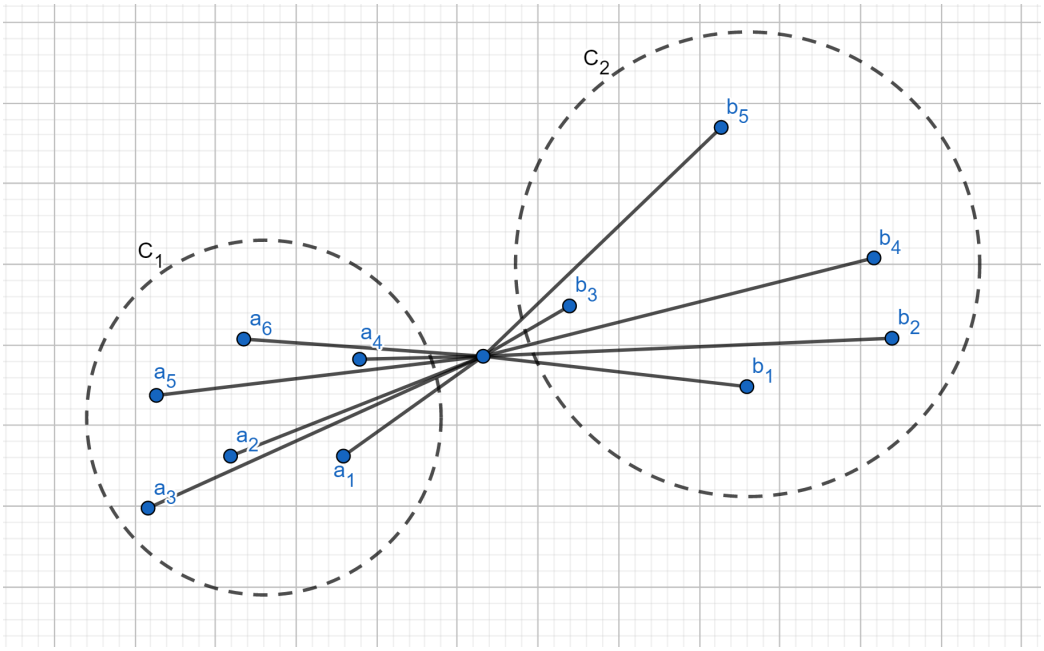

Figure 3.4: In Ward's method, the distance between two groups is determined as the increase in the sum of squared errors (SSE) when the two clusters are combined (Zaki and Meira, 2014)

- **Ward's method:** For object a in cluster C, and a distance d (potentially also a dissimilarity can be used), we define the cluster's center as

$$C^* = \frac{1}{|C|} \sum_{a \in C} a,$$

where $|C|$ is the number of objects in cluster C (Murtagh and Legendre, 2014). Define the sum of squared errors as

$$SSE_i = \sum_{a \in C_i} d^2(a, C_i^*),$$

(Murtagh and Legendre, 2014; Zaki and Meira, 2014) and write it in terms of all pairwise distances as

$$\sum_{a \in C_i} d^2(a, C_i^*) = \frac{1}{|C_i|} \sum_{a,b \in C_i, a<b} d^2(a, b)$$

(Murtagh and Legendre, 2014). In the context of vectors $a, b \in \mathbb{R}^p$,

the inequality $a < b$ holds true when $\|a\| < \|b\|$, i.e., the magnitude of vector a is smaller than the magnitude of vector b.

Define the SSE for a clustering $C = \{C_1, ..., C_m\}$ as

$$SSE = \sum_{i=1}^{m} SSE_i$$

(Zaki and Meira, 2014). The distance between two groups is defined as the increase in the sum of squared errors when the two clusters C_i and C_j are merged into C_{ij}:

$$d(C_i, C_j) = \Delta SSE_{ij} = SSE_{ij} - SSE_i - SSE_j$$

(Zaki and Meira, 2014). See Fig. 3.4 for an illustration of Ward's method.

Ward's method aims to join clusters so that the variation inside this cluster does not increase excessively (Härdle and Simar, 2014). This leads to groups as homogenous as possible (Härdle and Simar, 2014).

The Ward's hierarchical clustering method was first described by Ward Jr (1963) (Murtagh and Legendre, 2014). Since then, there has been various generalizations of the method (Murtagh and Legendre, 2014). In this thesis, we apply the Ward2 algorithm, implemented in R function 'hclust()' with the method option 'ward.D2'. The Ward2 algorithm implements Ward's method (Murtagh and Legendre, 2014). For a more specific presentation of Ward's method, the reader is guided to Murtagh and Legendre (2014).

Next, we present an agglomerative hierarchical clustering algorithm (Algorithm 1), which was modified from Härdle and Simar (2014) and Aggarwal and Reddy (2014).

Algorithm 1 Agglomerative hierarchical clustering

1: Each object forms a unique cluster
2: Compute the dissimilarity matrix D between all the objects
repeat
3: Identify the pair of clusters with the closest distance
4: Merge the two identified clusters into a single cluster
5: Compute the distances between the new cluster and remaining clusters and generate an updated dissimilarity matrix D
until All clusters are agglomerated into one cluster

Hierarchical clustering results can be represented as a tree called a dendrogram (James et al., 2021). The dendrogram is cut at the desired height,

which depends on the number of clusters we want. So far, we have discussed different options of similarity/dissimilarity measures, clustering algorithms, and linkage types to use when performing clustering. However, we still need to decide the number of clusters to assign to our data. The following section focuses on different criteria for choosing the number of clusters.

3.4 The number of clusters

When performing clustering, one of the decisions is determining the number of clusters to obtain (James et al., 2021). We have no previous information regarding the number of natural clusters of theme variables in our dataset. One way to determine the number of clusters is through visual inspection of the dendrogram, but other methods exist. This section presents two criteria for selecting an appropriate number of clusters: the gap statistic and the silhouette coefficient. Analysis of the number of clusters for theme variables was performed with R package 'factoextra' (Kassambara and Mundt, 2017).

3.4.1 The gap statistic

To compute the gap statistic, the within sum of squares (WSS) value (9) for clusters of the observed dataset is compared with a reference dataset that has no apparent clusters (Malik and Tuckfield, 2019). The gap statistic is the difference in the total WSS between the observed and the reference dataset (Nwanganga and Chapple, 2020). The reference dataset is a sample from the uniform distribution between our observed dataset's minimum and maximum values (Malik and Tuckfield, 2019). The optimal number of clusters is the number that yields the maximum value of the gap statistic (Malik and Tuckfield, 2019). Next, we demonstrate the theory of the gap statistic, more specifically based on Tibshirani et al. (2001).

Suppose we have clustered variables into k clusters $C_1, C_2, ..., C_k$, with C_r denoting the indices of variables, and $n_r = |C_r|$ is the number of variables in cluster r, respectively. Define

$$D_r = \sum_{i,j \in C_r} d_{i,j}$$

as the sum of the pairwise distances for all objects in cluster r, and set

$$W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r, \tag{9}$$

where W_k is the WSS value for clusters. The idea is to standardize the $\log(W_k)$ graph by comparing it with what would be expected under an appropriate null reference distribution of the data. The optimal number of clusters is the value of k, for which $\log(W_k)$ deviates the most from the expected values of the reference distribution. Define

$$Gap_n(k) = E_n^*[log(W_k)] - log(W_k),$$

where E_n^* denotes expectation under a sample of size n from the reference distribution. We have an estimate $\hat{k}$ when we maximize $Gap_n(k)$. To implement the gap statistic, we need to determine an appropriate reference distribution and assess the sampling distribution of the gap statistic.

Tibshirani et al. (2001) suggests two options for the reference distribution:

- **E**ach reference feature is generated uniformly from the range of values that are observed for that feature;
- **T**he reference features are generated by sampling from a uniform distribution within a box that aligns with the principal components of the data.

The advantage of the first method is its simplicity. The second method considers the shape of the data distribution and ensures that the procedure is rotationally invariant as long as the clustering method used is also invariant. The first method was utilized in the practical implementation of this thesis. Next, we demonstrate the computational implementation of the gap statistic based on Tibshirani et al. (2001).

In both cases, $E_n^*\left\{log(W_k)\right\}$ is estimated by an average of B copies $\log(W_k^*)$, each computed from a Monte Carlo sample $X_1^*, ..., X_n^*$ drawn from the reference distribution. Each reference dataset is clustered, giving within-dispersion measures $W_{kb}^*, b = 1, 2, ..., B, k = 1, 2, ..., K$. The estimated gap statistic is calculated

$$Gap(k) = (\frac{1}{B}\sum_b log(W_{kb}^*) - log(W_k)).$$

Let

$$\bar{l} = \frac{1}{B}\sum_b \log(W_{kb}^*),$$

and calculate the standard deviation

$$sd_k = [\frac{1}{B}\sum_b \left\{\log(W_{kb}^* - \bar{l}\right\}^2]^{1/2}.$$

Account additionally the simulation error in $E_n^* \left\{ log(W_k) \right\}$ and define the error in $Gap(k)$ as

$$s_k = sd_k \sqrt{(1 + \frac{1}{B})}.$$

To decide the optimal number of clusters, we choose the smallest k such that

$$Gap(k) \geq Gap(k+1) - s_{k+1}.$$

For a more specific presentation about the theory of the gap statistic, the reader is guided to Tibshirani et al. (2001).

3.4.2 The silhouette coefficient

The silhouette coefficient SC can be used to approximate the optimal number of clusters in data (Zaki and Meira, 2014). It can be used to quantify the quality of clusters (Malik and Tuckfield, 2019). This section presents the silhouette coefficient based on Izenman (2008).

Let C_K be a particular clustering of the data into K clusters and $c(i)$ the cluster containing ith object. Let a_i be the average dissimilarity between ith object and all other members of the same cluster $c(i)$. Let $d(i, c)$ be the average dissimilarity of the ith object to all members of c, which is some other cluster than $c(i)$. Calculate $d(i, c)$ for all clusters c other than $c(i)$. Let

$$b_i = min_{c \neq c(i)} d(i, c).$$

Calculate the ith silhouette width

$$s_{iK} = \frac{b_i - a_i}{max(a_i, b_i)},$$

where $-1 \leq s_{iK} \leq 1$.

When a_i is small, s_{iK} gets large positive values which indicate that the ith object is well-clustered. When b_i is small, s_{iK} gets large negative values which indicate poor clustering. When $a_i \approx b_i$ and $s_{iK} \approx 0$, the object lies between two clusters. Let $\bar{s}_K$ be the average silhouette width, the average of all the $\{s_{iK}\}$. The silhouette coefficient was defined by Kaufman and Rousseeuw (2009) as

$$SC = max_K \{\bar{s}_K\},$$

and it can be used as a clustering diagnostic. Kaufman and Rousseeuw (2009) also gave subjective interpretations for different values of SC:

- $0.71 \leq SC \leq 1.00$ - a strong structure has been found,
- $0.51 \leq SC \leq 0.70$ - a reasonable structure has been found,
- $0.26 \leq SC \leq 0.50$ - the structure is weak and could be artificial,
- $SC \leq 0.25$ - no substantial structure has been found.

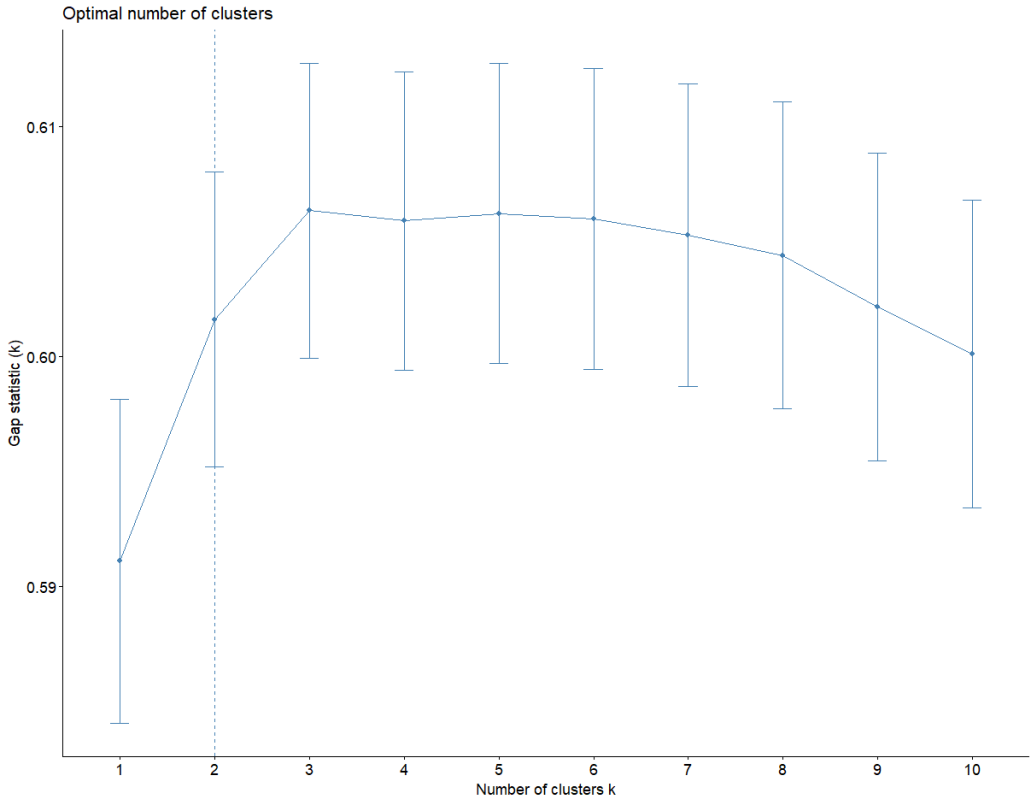

Figure 3.5: The gap statistic

3.4.3 The results

Next, we calculated the gap statistic and the average silhouette widths with different numbers of clusters for the hierarchical clusterization of the theme data, and visualized the results with R function 'fviz_nbclust()' from package 'factoextra' (Kassambara and Mundt, 2017). The gap statistic (see Fig. 3.5) suggests two clusters for the theme data. Fig. 3.6 shows that the average silhouette width is the largest with two clusters. $SC \leq 0.25$, which suggests that no substantial structure has been found.

Based on Fig. 3.5 and Fig. 3.6, an appropriate number of clusters for hierarchical clusterization of the theme data might be two clusters. We decided to explore the clustering results with two clusters.

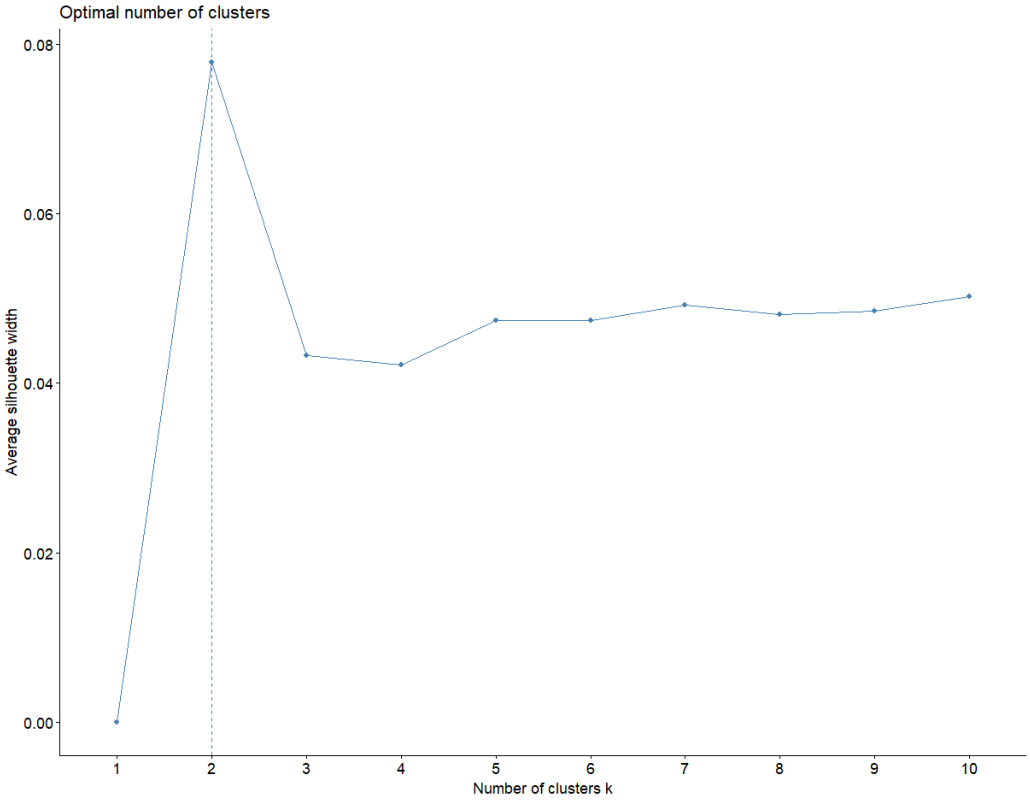

Figure 3.6: The average silhouette widths

3.5 Hierarchical clustering applied on theme variables

When performing hierarchical clustering, we must decide what dissimilarity measure to use, what type of linkage to use, and where to cut the dendrogram to obtain clusters (James et al., 2021). In practice, we experiment with various choices and select the one that yields the most useful or interpretable solution (James et al., 2021). We are clustering theme variables, and Izenman (2008) recommended correlation-based distance for clustering variables. In correlation-based distance, the emphasis is on the shapes of the observation profiles rather than their magnitudes (James et al., 2021). Clustering was performed with R function 'hclust()'.

Earlier, we calculated four different distance matrices using common similarity measures: Pearson's correlation coefficient, the simple matching coefficient, Jaccard's coefficient, and Yule's Q coefficient. We tried clustering with all of these matrices but found that the matrix based on Pearson's correlation coefficient best suited our purposes, which could also be reasoned from the previous paragraph.

We studied the number of clusters with the gap statistic and the silhouette coefficient and decided to obtain two clusters. We tried clustering with single linkage, average linkage, complete linkage, and Ward's method. Using a single linkage resulted in one big cluster in which each object joined one by one. Using average linkage with two clusters resulted in seventy-seven themes in one and two themes in another. Using complete linkage with two clusters resulted in fifty-five themes in one and twenty-four themes in another. Using Ward's method with two clusters resulted in seven themes in one and seventy-two in another cluster. We decided that using Ward's method might yield the most interpretable solution.

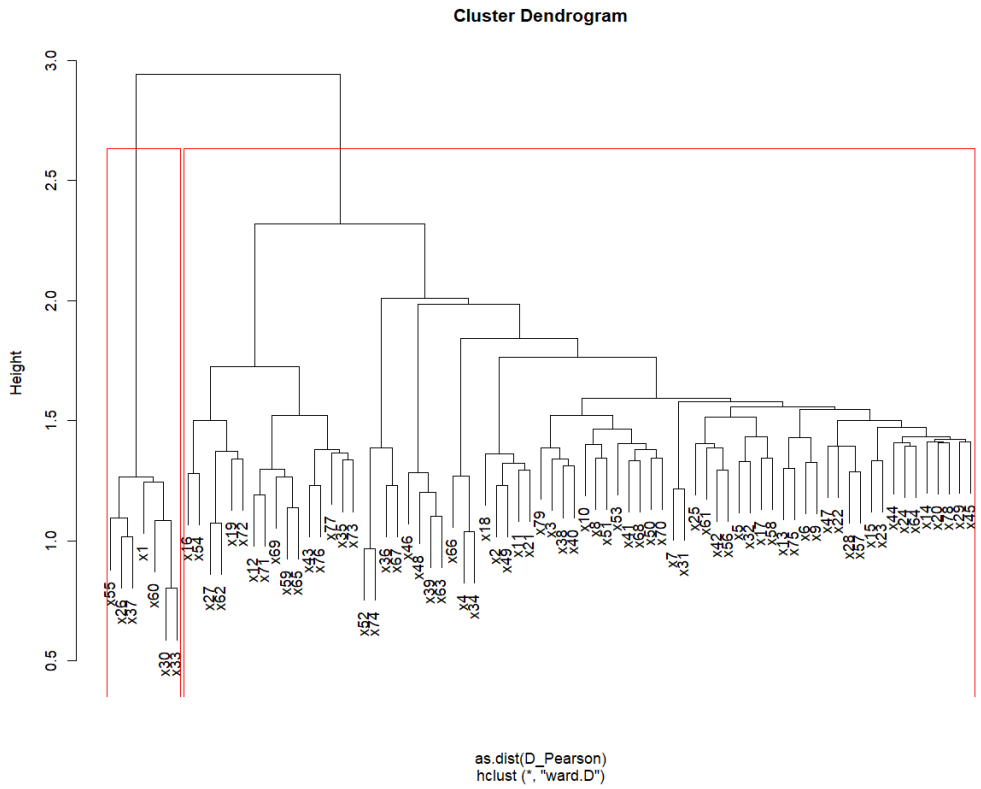

Figure 3.7: Results of hierarchical clustering when using correlation-based dissimilarity, Ward-method, and the tree is cut so that two clusters are obtained

Fig. 3.7 shows the hierarchical clustering results with the Ward method in a tree-based representation.

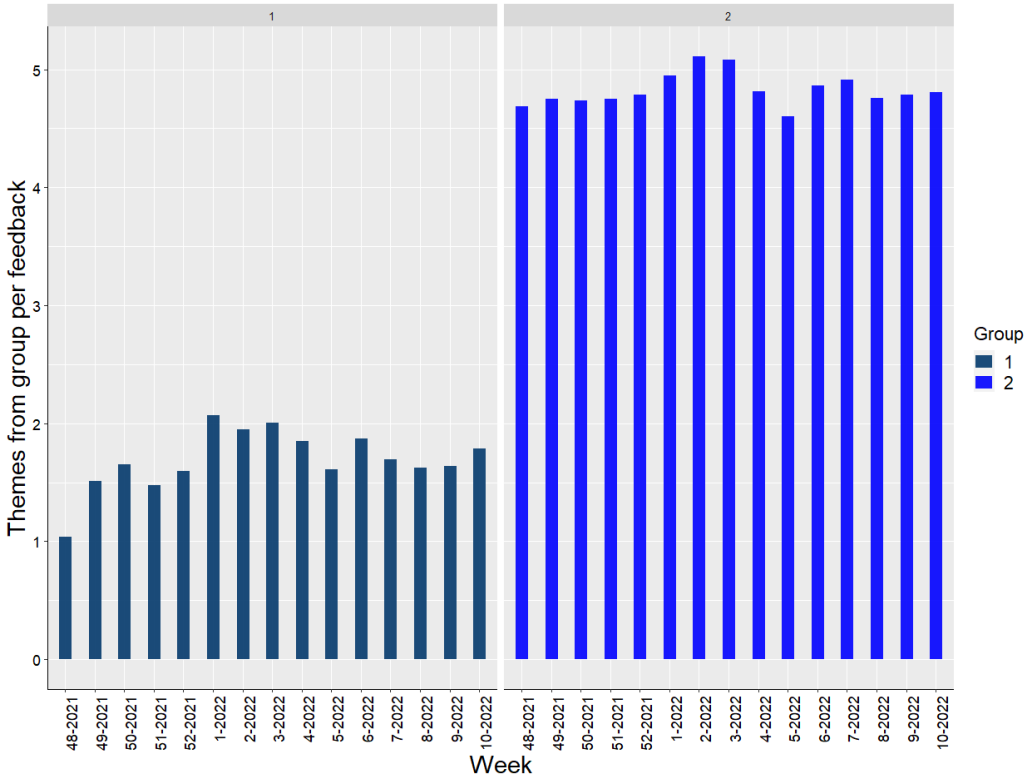

Figure 3.8: The relative proportions of the theme groups obtained from hierarchical clustering with distance based on Pearson's correlation and Ward's method

Next, we aggregate themes x_i for each cluster C_j, where $i = 1, 2, ..., 79$ and $j = 1, 2$ as

$$y_j = \sum_{x_i \in C_j} x_i$$

and visualize groups' weekly occurrences.

As we can see from Fig. 3.8, there are significant changes in Group 1. We should further investigate whether changes in those themes refer to some phenomena. In Group 2, there are more than ten times more themes than in Group 1, but probably fewer changes. Next, we investigate further the seven individual themes in Group 1.

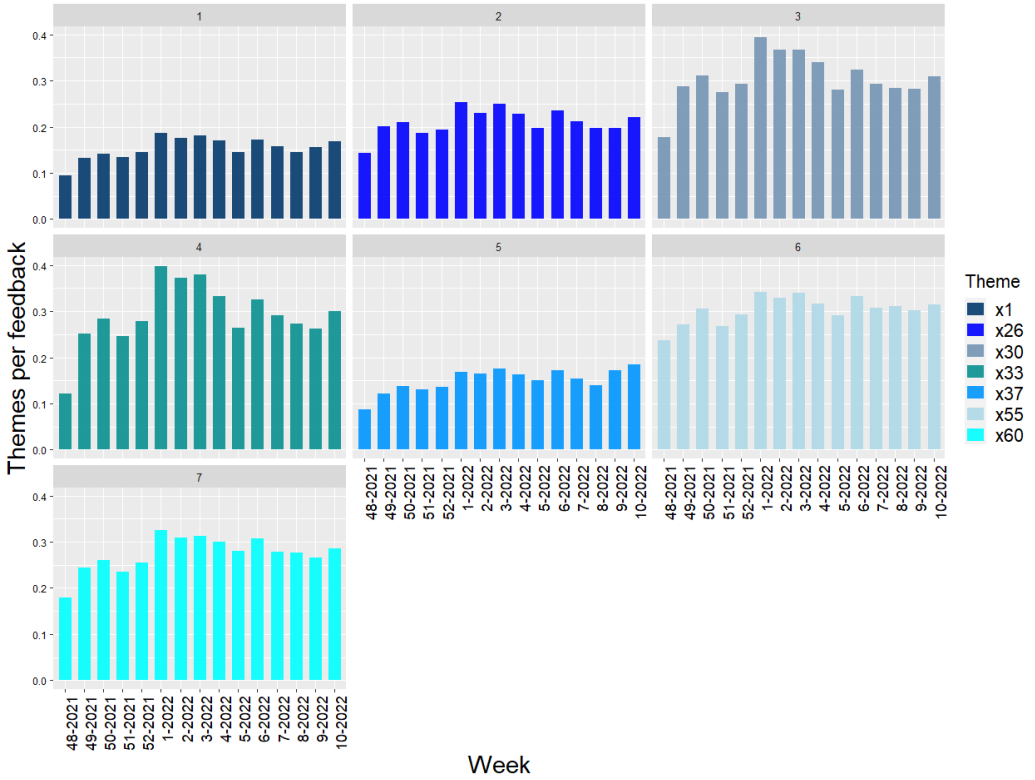

Figure 3.9: The relative proportions of changing themes from Group 1

As we can see from Fig. 3.9, the changes in themes from Group 1 follow the same trends. We investigated afterward what real themes were behind the seven pseudonymized themes of Group 1. We discovered five themes related to usability and two to customer service (Häkkinen, 2023). The reason why these themes have been emphasized might be a change in the user interface or in the method of use of some client applications. Customer service themes might have been emphasized due to a growing demand for guidance when usability has changed. In Chapter 6, we explore the sentiment of the themes in Group 1 to determine how the changes in usability have been received.

4 Multidimensional scaling

In this chapter, we apply multidimensional scaling (MDS) as a technique for dimension reduction and visualization of multidimensional data in a low-dimensional space. MDS aims to find a low dimensional space, usually Euclidean, in which the distances between the points represent the original dissimilarities between the objects in the higher dimension (Cox and Cox, 2001). Metric MDS is applied when the original dissimilarities are measured on a metric scale, and nonmetric MDS is based on the rank order of dissimilarities (Cox and Cox, 2001; Härdle and Simar, 2014). Next, we introduce the metric MDS theory and then apply it to visualize the relationships between the themes in the data.

4.1 Metric multidimensional scaling - Classical scaling

The two main metric MDS methods are the so-called classical scaling and the least squares scaling, of which the former will be considered in this chapter. Suppose we know the coordinates of n points in the p-dimensional Euclidean space. Then we can calculate the Euclidean distances between each pair of points (Chatfield and Collins, 1980). The MDS considers the inverse problem. We know the distances between each pair of points and aim to recover the coordinates (Chatfield and Collins, 1980). Next, we present the theory of classical scaling based on Cox and Cox (2001) and Härdle and Simar (2014).

The classical scaling begins with $(n \times n)$ distance matrix $D = (d_{ij})$ that is computed from Euclidean geometry. Dissimilarities can also be used if they correspond to Euclidean distances. In the sense that the inner product matrix B, which is later defined, is positive semidefinite.

Consider n points in a p-dimensional Euclidean space, where the coordinates are given by x_i $(i = 1, ..., n)$ where $x_i = (x_{i1}, ..., x_{ip})^T$. Call $X = (x_1, ..., x_n)^T)$ the coordinate matrix and place the centroid of the configuration points at the origin

$$\sum_{k=1}^{n} x_{ki} = 0 \ \ (i = 1, ...p).$$

The squared Euclidean distance between ith and jth point is given by

$$d_{ij}^2 = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2 = (x_i - x_j)^T (x_i - x_j).$$

Let the inner product matrix be B, where

$$[B]_{ij} = b_{ij} = \sum_{k=1}^{p} x_{ik}x_{jk} = x_i^T x_j.$$

B can be found from the known squared distances, and then from B, the unknown coordinates may be recovered. Observe that

$$d_{ij}^2 = x_i^T x_i + x_j^T x_j - 2x_i^T x_j = b_{ii} + b_{jj} - 2b_{ij}. \tag{10}$$

Assuming that the coordinates are centered, that is $\sum_{i=1}^{n} x_i = 0$. Hence

$$\frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 = \frac{1}{n}\sum_{i=1}^{n} b_{ii} + b_{jj}$$

$$\frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 = b_{ii} + \frac{1}{n}\sum_{j=1}^{n} b_{jj}$$

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 = \frac{2}{n}\sum_{i=1}^{n} b_{ii}. \tag{11}$$

With $a_{ij} = -\frac{1}{2}d_{ij}^2$, and

$$a_{i\bullet} = \frac{1}{n}\sum_{j=1}^{n} a_{ij},\ a_{\bullet j} = \frac{1}{n}\sum_{i=1}^{n} a_{ij},\ a_{\bullet\bullet} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij},$$

$$d_{i\bullet}^2 = \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2,\ d_{\bullet j}^2 = \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2,\ d_{\bullet\bullet}^2 = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2,$$

solving (10) and (11) gives:

$$b_{ij} = -\frac{1}{2}(d_{ij}^2 - d_{i\bullet}^2 - d_{\bullet j}^2 + d_{\bullet\bullet}^2) = a_{ij} - a_{i\bullet} - a_{\bullet j} + a_{\bullet\bullet}.$$

Define the matrix A as (a_{ij}), and observe that:

$$B = HAH$$

where H is the centering matrix,

$$\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^T,$$

with $\mathbf{1} = (1, 1, ..., 1)^T$, a vector of n ones. The inner product matrix B can be expressed as:

$$B = XX^T,$$

where $X = (x_1, ...x_n)^T$ is the $(n \times p)$ matrix of coordinates. The rank of B is then

$$rank(B) = rank(XX^T) = rank(X) = p.$$

It is required that the matrix $B = HAH$ is symmetric, positive semidefinite, and of rank p, and hence it has p non-negative eigenvalues and $n - p$ zero eigenvalues. We can now express B as:

$$B = \Gamma\Lambda\Gamma^T,$$

where $\Gamma = diag(\lambda_1, ..., \lambda_p)$, the diagonal matrix of the eigenvalues of B, and $\Gamma = (\gamma_1, ..., \gamma_p)$, the matrix of the corresponding eigenvectors. Hence, we can express the coordinate matrix X as:

$$X = \Gamma\Lambda^{\frac{1}{2}},$$

which contains the point configuration in $\mathbb{R}^p$.

Next, we consider the number of desired dimensions by examining the eigenvalues. The measure of the proportion of variation explained by p dimensions is defined by

$$\frac{\sum_{i=1}^{p} \lambda_i}{\sum_{i=1}^{n-1} \lambda_i}.$$

We can use the measure above to decide the number of dimensions. A symmetric matrix is positive definite if all of its eigenvalues are positive, and a positive semidefinite matrix also allows eigenvalues $\lambda = 0$ (Strang, 2016). If the dissimilarities lead to a matrix B which is not positive semidefinite, the preceding measure is modified so that negative eigenvalues are omitted from the sum in the nominator. If B is not positive semidefinite, we can make B positive semidefinite by adding a constant to all the dissimilarities. A dissimilarity matrix can be used as a distance matrix when B is positive semidefinite.

4.2 A numerical example of classical MDS

Next, we illustrate classical scaling by calculating the coordinates for the first four theme variables in two-dimensional space. We follow the procedure from Wickelmaier (2003).

The dissimilarity matrix based on Pearson's correlation coefficient for the first four theme variables is defined as

$$D = \begin{bmatrix} 0.00 & 1.30 & 1.32 & 1.41 \\ 1.30 & 0.00 & 1.39 & 1.44 \\ 1.32 & 1.39 & 0.00 & 1.40 \\ 1.41 & 1.44 & 1.40 & 0.00 \end{bmatrix}.$$

When A is defined as $a_{ij} = -\frac{1}{2}d_{ij}^2$, we have

$$A = \begin{bmatrix} 0.00 & -0.84 & -0.87 & -1.00 \\ -0.84 & 0.00 & -0.97 & -1.04 \\ -0.87 & -0.97 & 0.00 & -0.99 \\ -1.00 & -0.54 & -0.99 & 0.00 \end{bmatrix}.$$

The centering matrix H is calculated by

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - 4^{-1} * \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.75 & -0.25 & -0.25 & -0.25 \\ -0.25 & 0.75 & -0.25 & -0.25 \\ -0.25 & -0.25 & 0.75 & -0.25 \\ -0.25 & -0.25 & -0.25 & 0.75 \end{bmatrix}.$$

B is calculated by

$$B = HAH = \begin{bmatrix} 0.64 & -0.17 & -0.20 & -0.28 \\ -0.17 & 0.71 & -0.26 & -0.28 \\ -0.20 & -0.26 & 0.70 & -0.24 \\ -0.28 & -0.28 & -0.24 & 0.80 \end{bmatrix}.$$

The eigenvalues of B are

$$\lambda_1 = 1.08,\ \lambda_2 = 0.96,\ \lambda_3 = 0.82,\ \lambda_4 = 0.00.$$

The eigenvalues of a symmetric matrix B are $\lambda \geq 0$, so B is a positive semi-definite matrix and the original dissimilarities can be used as distances in Euclidean space. For a two-dimensional representation of four theme variables, we need the first two largest eigenvalues and the corresponding eigenvectors of B:

$$\gamma_1 = \begin{bmatrix} -0.33 \\ -0.47 \\ -0.01 \\ 0.82 \end{bmatrix},\ \gamma_2 = \begin{bmatrix} 0.04 \\ -0.54 \\ 0.79 \\ -0.29 \end{bmatrix}.$$

The coordinate matrix X is then calculated by

$$X = \Gamma\Lambda^{1/2} = \begin{bmatrix} -0.33 & 0.04 \\ -0.47 & -0.54 \\ -0.01 & 0.79 \\ 0.82 & -0.29 \end{bmatrix} * \begin{bmatrix} 1.08 & 0.00 \\ 0.00 & 0.96 \end{bmatrix}^{1/2} = \begin{bmatrix} -0.25 & 0.02 \\ -0.36 & -0.36 \\ -0.01 & 0.53 \\ 0.62 & -0.20 \end{bmatrix}.$$

For the visual representation of calculated coordinates, see Fig. 4.1. The proportion of variation explained by two dimensions is given by

$$\frac{\lambda_1 + \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4} = \frac{1.08 + 0.96}{1.08 + 0.96 + 0.82 + 0.00} = 0.71.$$

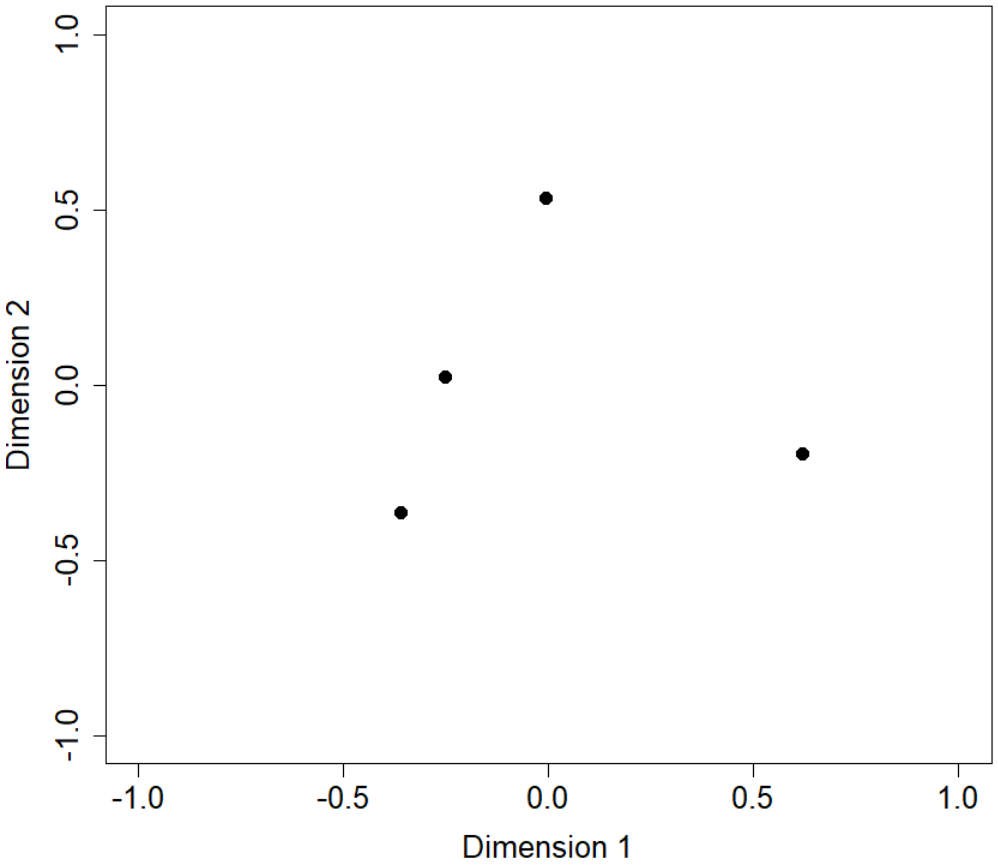

Figure 4.1: Classical MDS representation of four theme variables in two dimensions

4.3 Classical MDS applied on theme variables

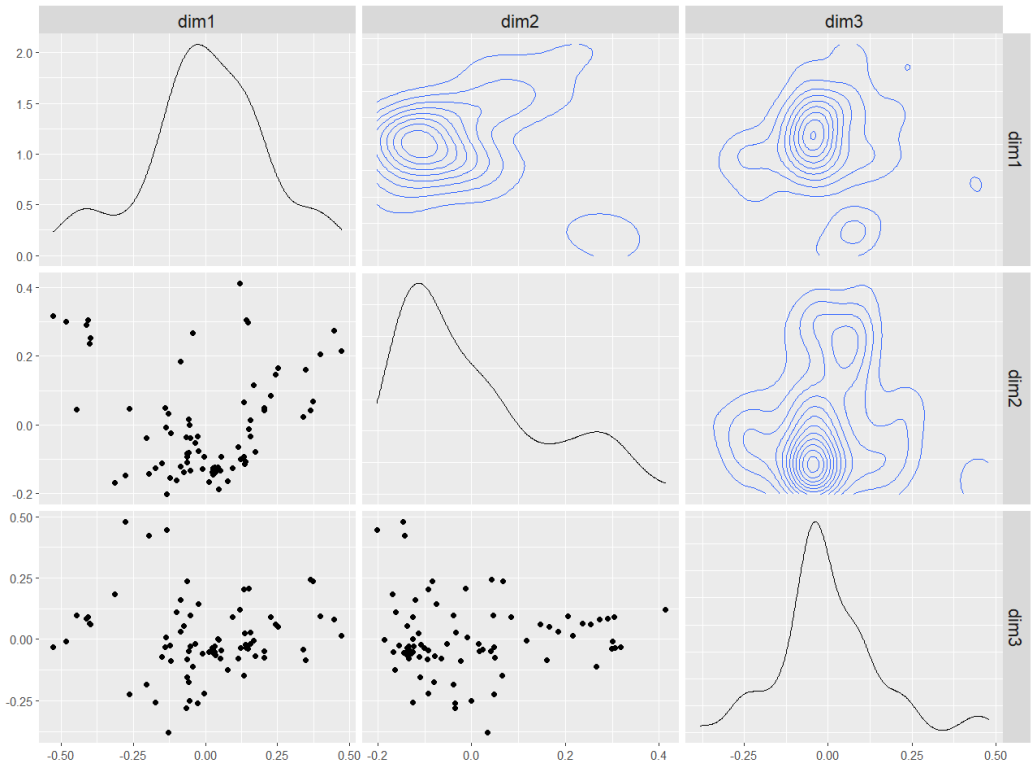

Figure 4.2: Metric MDS applied on theme variables

We performed classical scaling for all 79 theme variables. Metric MDS was performed with R function 'cmdscale()' from package 'stats'. The coordinates were defined in three-dimensional space. For a visual representation, see Fig. 4.2.

Next, we combined the results of hierarchical clustering and MDS. For a visual representation, see Fig. 4.3.

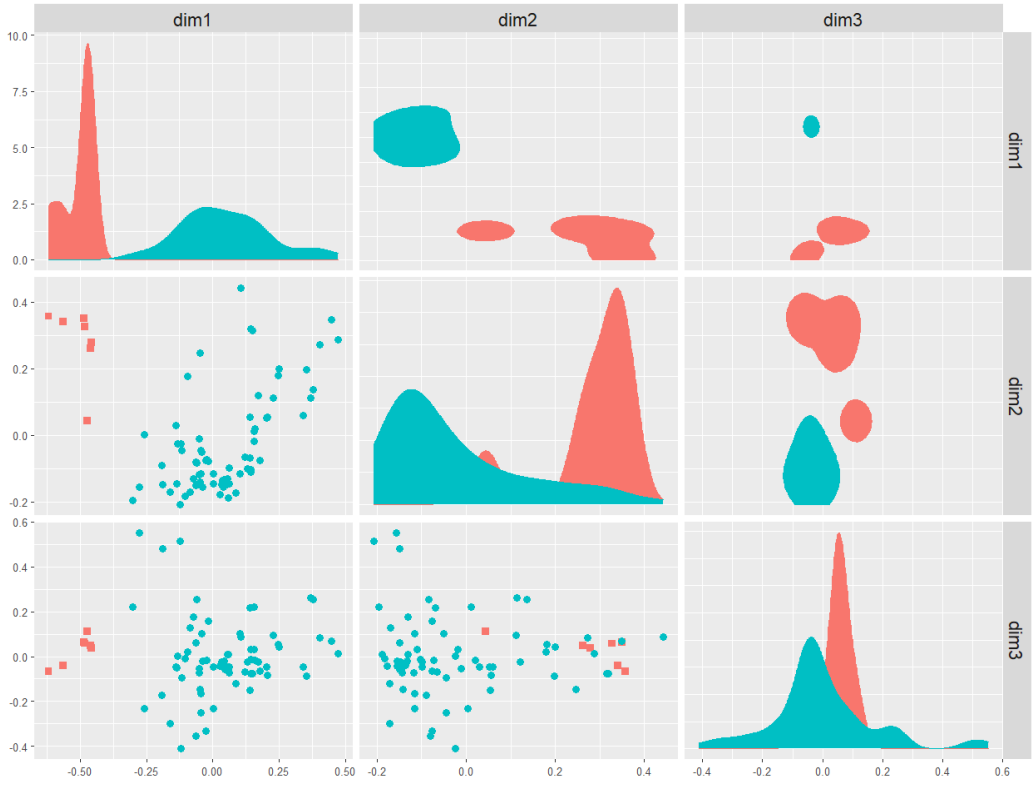

Figure 4.3: Visualization of themes and theme groups based on the results of metric MDS and hierarchical clustering

From Fig. 4.3, we can observe that themes in Group 1 are well separated from the rest of the themes, even though three dimensions explain only 11% of the total variation in 79 dimensions. It would be desirable for the MDS solution to explain a larger proportion of the variation, but achieving this demands more dimensions, making the visualization more challenging.

5 Bootstrap confidence intervals

One of the thesis aims was to determine whether the changes observed in themes could be due to natural variation. In this chapter, we apply the repeated sampling method to construct bootstrap confidence intervals. The repeated sampling method allows us to construct approximate confidence intervals, which help us determine whether weekly changes in themes can be due to random variation or some underlying phenomena.

The bootstrap is a data-based simulation technique for making statistical inferences (Tibshirani and Efron, 1993) which can be used when the underlying statistical distribution of the data is unknown or when the assumptions of normality do not apply (Ramachandran and Tsokos, 2020). With the bootstrap procedure, the approximate sampling distribution of the statistic, conditional on the observed data, can be easily calculated (Ramachandran and Tsokos, 2020).

5.1 Standard normal intervals

Consider $\hat{\theta}$ as the estimate of a parameter θ and $\hat{se}$ as its estimated standard error. Let $\hat{\theta}^*$ represent a random variable drawn from the distribution $N(\hat{\theta}, \hat{se}^2)$,

$$\hat{\theta}^* \sim N(\hat{\theta}, \hat{se}^2).$$

Then the standard normal confidence interval of θ is defined as

$$[\hat{\theta}_{lower}, \hat{\theta}_{upper}] = [\hat{\theta} - z^{1-\alpha} * \hat{se}, \hat{\theta} - z^{\alpha} * \hat{se}], \tag{12}$$

where $\hat{\theta}_{lower}$ is 100αth percentile of $\hat{\theta}^*$'s distribution and $\hat{\theta}_{upper}$ is $100(1-\alpha)$th percentile of $\hat{\theta}^*$'s distribution (Tibshirani and Efron, 1993).

5.2 The procedure for finding bootstrap confidence intervals

Next, we describe how to construct bootstrap confidence intervals following the treatment of this topic by Ramachandran and Tsokos (2020). The procedure for finding confidence intervals for the mean follows the algorithm below (Algorithm 2).

Algorithm 2 Bootstrap confidence intervals for the mean

1: Resample N times from the original sample with replacement, N being in the hundreds or the thousands.
2: Calculate the sample mean for each of the resamples.
3: Arrange the sample means in order of magnitude.
4: Determine the 95 percent confidence interval from the middle 95 percent of the sample means, with the 2.5th percentile being the value at the position (0.025)(N+1) and the 97.5th percentile being the value at the position (0.975)(N+1) of the ordered means. If these values are not integers, round them to the nearest integer.

5.3 Bootstrap confidence intervals for theme occurrences

We calculated 95% confidence intervals for the weekly occurrence of themes in different groups per feedback. In this way, we quantify the limits within which natural variation in the data is likely to exist. We calculated the bootstrap confidence intervals using R function 'boot' in package 'boot' (Canty and Ripley, 2017).

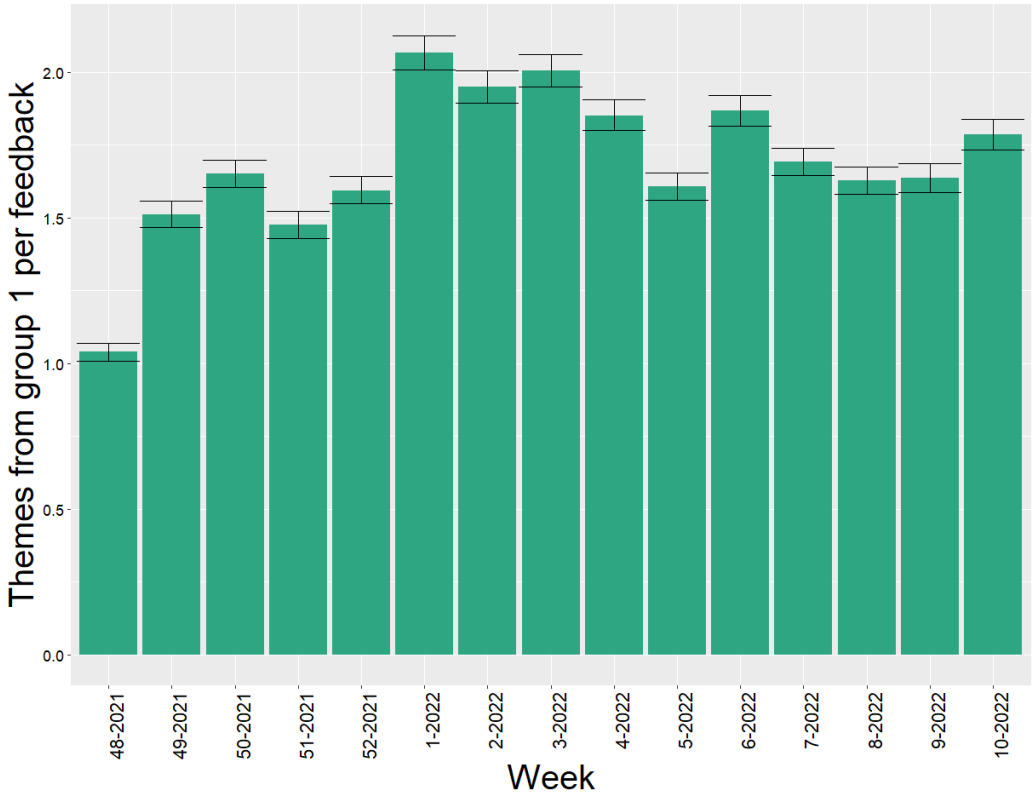

Figure 5.1: Weekly themes from group 1 per feedback with 95% confidence intervals

Fig. 5.1 shows that the largest increases in the occurrence of themes from group 1 are in week 49 of 2021 and week 1 of 2022. Theme fluctuations between week 7 of 2022 and week 9 of 2022 may be attributed to natural variation.

6 Negative binomial regression

In this chapter, we introduce negative binomial regression to model theme counts of a particular theme group in individual feedback depending on the week. The count of theme group occurrences in individual feedback is a count response variable that can be modeled with the Poisson regression model. However, the Poisson regression model assumes that the mean of the count response variable equals its variance (Liu, 2022). We noticed that the observed variance of theme count was greater than the mean, violating this assumption. This phenomenon, where the variance is greater than the mean, is called overdispersion, which was dealt with using negative binomial regression.

6.1 The negative binomial distribution

This section discusses the relation between the negative binomial distribution and the Poisson distribution mainly based on Zhou and Carin (2013). Count data is commonly modeled with the Poisson distribution. Denote k as the count

$$k \sim Pois(\lambda),$$

with a probability mass function

$$f(k;\lambda) = \frac{\lambda^{\mu} e^{-\lambda}}{k!},\ k \in \mathbb{Z}_+.$$

Often count data are overdispersed, which can be dealt with by placing a gamma prior to λ:

$$k \sim Pois(\lambda),$$
$$\lambda \sim Gamma(r, \frac{p}{1-p}).$$

Marginalizing out λ, we obtain the negative binomial distribution

$$k \sim NB(r,p),$$

with a probability mass function

$$f(k;r,p) = \frac{\Gamma(k+r)}{k!\Gamma(r)}(1-p)^k p^r,\ k \in \mathbb{Z}_+, \tag{13}$$

where $r > 0$, $p \in [0,1]$ are parameters. Thus, the alternative name for the negative binomial distribution is the gamma-Poisson mixture distribution.

Its mean $\mu = \frac{rp}{1-p}$ is smaller than its variance $\sigma^2 = \frac{rp}{(1-p)^2} = \mu + r^{-1}\mu^2$, and it is usually preferred over the Poisson distribution when counts are overdispersed (Zhou and Carin, 2013).

When $r \in \mathbb{N}$, by using the identity $\Gamma(x) = (x-1)!$ in (13), we derive another form for a probability mass function of a negative binomial distribution:

$$\begin{aligned} f(k; r, p) &= \frac{\Gamma(k+r)}{k!\Gamma(r)}(1-p)^k p^r \\ &= \frac{(k+r-1)!}{(r-1)!k!}(1-p)^k p^r \\ &= \binom{k+r-1}{k}(1-p)^k p^r, \end{aligned}$$

where r represents the number of successes, k represents the number of failures, and p represents the probability of success on each trial.

6.2 The model formula and incidence rate ratios

This section presents the negative binomial regression model formula and incidence rate ratios based on Liu (2022). When the variance equals the mean, the negative binomial model is the same as the Poisson model. The negative binomial regression model can be presented in the form:

$$ln(\mu) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \varepsilon, \tag{14}$$

where $ln(\mu)$ is the log link function; μ is the mean; β_0 is the intercept; $\beta_1, \beta_2, ..., \beta_p$ are the coefficients for the predictors, and ε is the error term. Exponentiate both sides of (14) and assume that $\varepsilon = 0$. The predicted mean of the count response variable can be written as

$$\mu = exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p).$$

Sometimes, we want to model the number of events during a period or in a location, then our count response variable can be referred to as an incident rate. Define the incidence rate μ/t as the expected number of events per unit time or location, and the negative binomial regression model can be presented in the form:

$$ln(\mu/t) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \varepsilon.$$

By the calculation rules of the logarithm, the incidence rate $ln(\mu/t) = ln(\mu) - ln(t)$, and the equation can be written as

$$ln(\mu) = ln(t) + \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \varepsilon,$$

where $ln(t)$ is the offset term.

Consider then a simple negative binomial regression model with one predictor, which can be written as

$$ln(\mu) = \alpha + \beta X + \varepsilon.$$

By exponentiating both sides, we get

$$\mu = exp(\alpha + \beta X).$$

When the independent variable X is a categorical variable with values of 0 and 1, the expected counts or the incident rates of the response variable are

$$\mu = \begin{cases} exp(\alpha), \ X = 0 \\ exp(\alpha + \beta), \ X = 1. \end{cases}$$

By comparing group 1 ($X = 1$) to group 2 ($X = 0$), we get the incidence rate ratio:

$$IRR = \frac{exp(\alpha + \beta)}{exp(\alpha)} = \frac{exp(\alpha)exp(\beta)}{exp(\alpha)} = exp(\beta).$$

6.3 Negative binomial regression applied on theme occurrences and sentiment

We modeled the theme count of Group 1 occurrences in individual feedback, independent variables being the week numbers as categorical variables. We applied negative binomial regression using R function 'glm.nb()' from package 'MASS' (Venables and Ripley, 2002).

We define the negative binomial regression model as

$$ln(\mu) = \beta_1 Week_{48-2021} + \beta_2 Week_{49-2021} + ... + \beta_{15} Week_{10-2022} + \varepsilon,$$

where the individual variables $Week_{48-2021}, Week_{49-2021}, ..., Week_{10-2022}$ are categorical variables with the values of 0 and 1, depending on the week that feedback was given. We excluded the intercept term for easier interpretation. Coefficients and standard errors of the negative binomial regression are given in Table 1.

Table 1: Coefficients and standard errors of the negative binomial regression

Week	Estimate	exp(Estimate)	Std. Error
48-2021	0.0387	1.04	0.0148
49-2021	0.4130	1.51	0.0155
50-2021	0.5011	1.65	0.0143
51-2021	0.3891	1.48	0.0163
52-2021	0.4666	1.59	0.0148
1-2022	0.7259	2.07	0.0144
2-2022	0.6675	1.95	0.0144
3-2022	0.6958	2.01	0.0142
4-2022	0.6162	1.85	0.0146
5-2022	0.4747	1.61	0.0150
6-2022	0.6251	1.87	0.0144
7-2022	0.5262	1.69	0.0144
8-2022	0.4870	1.63	0.0147
9-2022	0.4927	1.64	0.0152
10-2022	0.5795	1.79	0.0149

Next, we discuss the interpretation of the exponentiated coefficients of the negative binomial regression. By exponentiating β_1, we get the mean of the theme count from Group 1 in week 48-2021 as

$$\mu_{48-2021} = exp(\beta_1) = exp(0.0387) = 1.04.$$

Then, we can calculate confidence intervals with (12). For example, 95% confidence intervals for the count of themes from group 1 in individual feedback in the week 48-202 are

$$[1.04 - 1.96 * 0.0148,\ 1.04 + 1.96 * 0.0148] = [1.01,\ 1.07].$$

In the same way, we get the mean of theme count from Group 1 in individual feedback for all the weeks and their confidence intervals, which are displayed in Table 2.

Table 2: Theme count means from Group 1 and their 95% confidence intervals

Week	Count	Low	High
48-2021	1.04	1.01	1.07
49-2021	1.51	1.47	1.56
50-2021	1.65	1.60	1.70
51-2021	1.48	1.43	1.52
52-2021	1.59	1.55	1.64
1-2022	2.07	2.00	2.13
2-2022	1.95	1.89	2.01
3-2022	2.01	1.95	2.06
4-2022	1.85	1.80	1.91
5-2022	1.61	1.56	1.66
6-2022	1.87	1.82	1.92
7-2022	1.69	1.65	1.74
8-2022	1.63	1.58	1.68
9-2022	1.64	1.59	1.69
10-2022	1.79	1.73	1.84

Results in Table 2 are visualized in Fig. 6.1, which is similar to Fig. 5.1, which we got earlier with bootstrap.

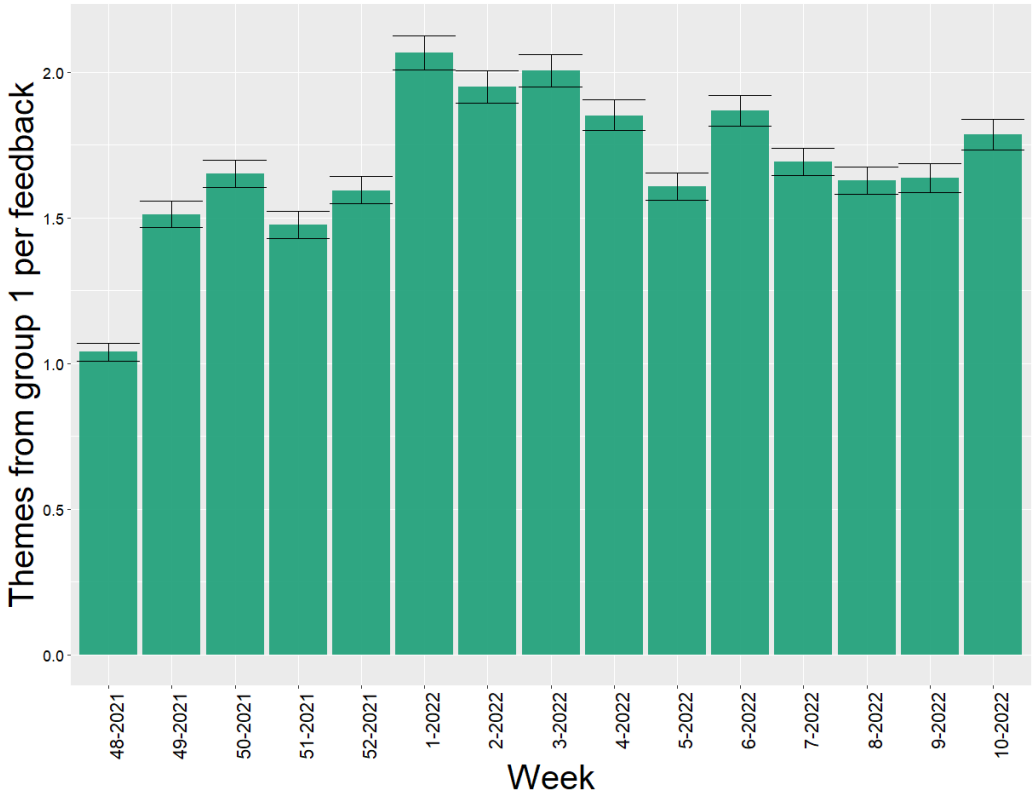

Figure 6.1: Weekly themes from Group 1 per feedback with 95% confidence intervals

Let us reiterate the themes of Group 1, which we were told to be related to usability and customer service. Next, we explore the sentiment of the Group 1 to determine how the changes in usability have been received. We define the negative binomial regression model as

$$ln(\mu) = \beta_0 + \beta_1 Sentiment_0 + \beta_2 Sentiment_1 + \varepsilon,$$

where the individual variables $Sentiment_0$ and $Sentiment_1$ are categorical variables with the values of 0 and 1, depending on the sentiment of the feedback. The intercept refers to negative feedback, the reference that neutral and positive feedback is compared to. Coefficients and standard errors of the negative binomial regression are given in Table 3.

Table 3: Coefficients and standard errors of the negative binomial regression, where the predictor is the sentiment

Sentiment	Estimate	exp(Estimate)	Std. Error
Intercept	0.7966	2.22	0.0049
Neutral	-1.7541	0.17	0.0254
Positive	-0.5991	0.55	0.0076

From Table 3, we can see that the mean of the theme count from group 1 in individual negative feedback is 2.22. Individual feedback is 83% less likely to be neutral and 45% less likely to be positive than negative. From Fig. 6.2, we can see how different sentiments behaved in the whole 15 weeks of study period.

As seen from Table 3 and Fig. 6.2, there has been much more negative than positive feedback regarding Group 1. The sentiment may indicate how the changes in usability have been received. It is also worth noting a significant amount of negative feedback is typically received when something does not function as expected. Additionally, it is common to encounter initial errors following the changes in usability, which could explain the considerable increase in negative feedback at week 52 of 2021. A rise in positive feedback may indicate that there are usability improvements as well. For a more accurate interpretation, a thorough analysis of the customer feedback at the text level or the client's assessment would be required (Häkkinen, 2023).

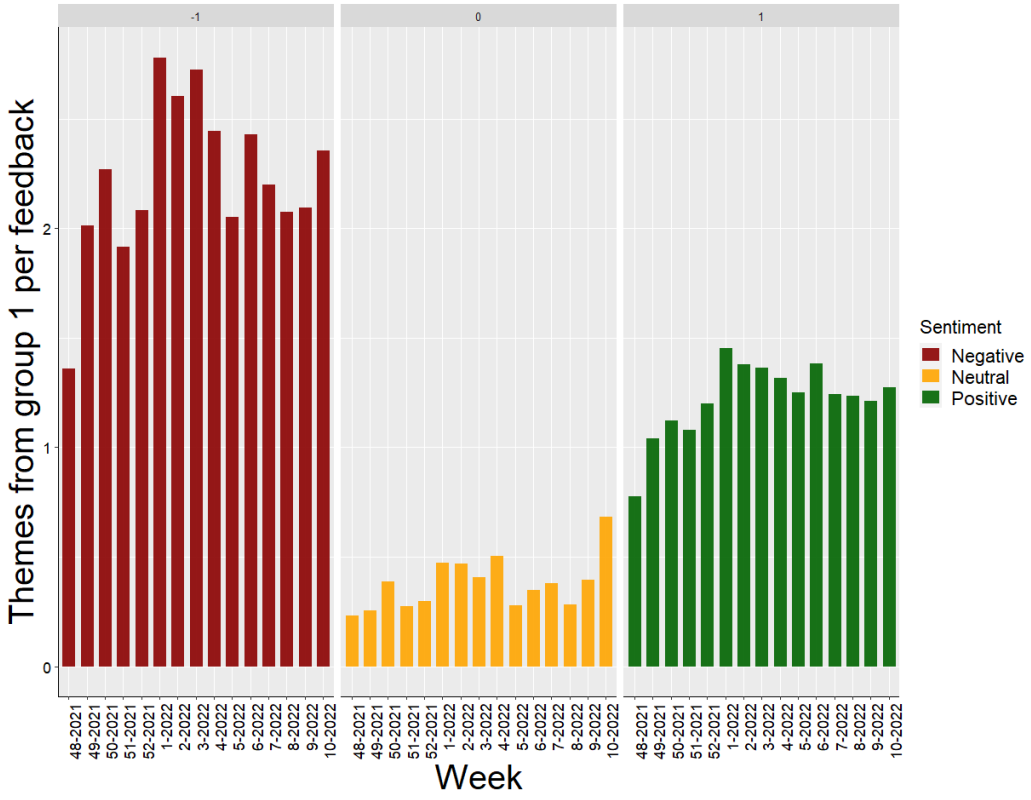

Figure 6.2: Weekly themes from Group 1 per feedback with 95% confidence intervals in different sentiments

7 Conclusions

We conducted a multivariate statistical analysis of thematic changes in customer feedback. The main focus of the study was 79 binary theme variables. Clustering, multidimensional scaling, bootstrap confidence intervals, and negative binomial regression were the methods we used.

We explored different similarity measures designed for binary variables. We concluded that Pearson's correlation coefficient was suitable for theme variables since we aimed to discover themes that exhibited similar behavior throughout the study period. We defined the 'natural groupings' of themes by agglomerative hierarchical clustering. We considered different proximity measures to define the dissimilarity between different clusters. We noticed that the distance measure determined by Ward's method was suitable. We explored the gap statistic, the silhouette coefficient, and the elbow method for choosing the optimal number clusters. We assigned themes to two clusters based on the gap statistic and the silhouette coefficient. By visual inspection, we discovered a group of seven themes (Group 1) which included the most significant changes on a weekly basis during the study period. We were informed that five themes in Group 1 related to usability and two to customer service. These themes may have been emphasized due to a change in the user interface or in the method of use of some client applications.

We applied metric multidimensional scaling on themes to visualize data in a low-dimensional space. The theory of classical scaling was discussed and applied in a numerical example. We combined the results of hierarchical clustering and MDS and distinguished different theme groups in three-dimensional space.

We defined the 95% confidence intervals of theme occurrences of Group 1 by bootstrapping and used these confidence intervals to quantify the limits within which natural variation in the data is likely to exist. We aimed to decide whether the smaller changes in data could be due to natural variation or some underlying phenomena. We noticed there were changes that natural variation did not explain.

We used negative binomial regression to model theme counts of theme Group 1 in individual feedback depending on the week. The negative binomial regression was another method to quantify changes in data. With the model, we calculated the 95% confidence intervals and observed that they were similar to bootstrap confidence intervals. We also modeled theme counts of Group 1 in individual feedback depending on the sentiment. We discovered that most of the feedback that contained themes from Group 1 was negative. We concluded that while this might indicate how changes in

usability have been received, this might also be because customers tend to give feedback when something does not function as expected. Additionally, we pondered that it is common to encounter initial errors following changes in usability. We concluded that there was also a rise in positive feedback, which may also indicate usability improvements. However, a thorough analysis of the customer feedback at the text level or the client's assessment would be necessary for a more accurate interpretation.

Additional analyzes, such as a change point analysis, could have been performed for the study data. However, that was beyond the scope of this thesis.

References

Charu C. Aggarwal and Chandan K. Reddy. *Data Clustering: Algorithms and Applications*. Chapman and Hall/CRC, 2014.

Michael R Anderberg. *Cluster analysis for applications: probability and mathematical statistics: a series of monographs and textbooks*, volume 19. Academic press, 2014.

Angelo Canty and Brian Ripley. Package 'boot'. *Bootstrap Functions. CRAN R Proj*, 2017.

C. Chatfield and A.J. Collins. *Introduction to Multivariate Analysis*. Chapman and Hall/CRC, 1st edition, 1980.

Seung-Seok Choi, Sung-Hyuk Cha, Charles C Tappert, et al. A survey of binary similarity and distance measures. *Journal of systemics, cybernetics and informatics*, 8(1):43–48, 2010.

Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling*. Chapman and Hall/CRC, 2nd edition, 2001.

E. Häkkinen. Personal communication, 2023. Aiwo Digital Oy.

Wolfgang Karl Härdle and Léopold Simar. *Applied Multivariate Statistical Analysis*. Springer, 4th edition, 2014.

Alan Julian Izenman. *Modern Multivariate Statistical Techniques: regression, classification, and manifold learning*. Springer, 2008.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshiran. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2nd edition, 2021.

Alboukadel Kassambara and Fabian Mundt. Package 'factoextra'. *Extract and visualize the results of multivariate data analyses*, 76(2), 2017.

L. Kaufman and P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley and Sons, 2009.

Xing Liu. *Categorical Data Analysis and Multilevel Modeling Using R*. SAGE Publications, 2022.

A. Malik and B. Tuckfield. *Applied unsupervised learning with R: Uncover hidden relationships and patterns with k-means clustering, hierarchical clustering, and PCA*. Packt Publishing Ltd., 2019.

Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31:274–295, 2014.

Frederick Novomestky and Maintainer S Thomas Kelly. Package 'matrixcalc'. 2022.

Fred Nwanganga and Mike Chapple. *Practical Machine Learning in R*. Wiley, 2020.

R. K. Pearson. *Exploratory data analysis using R*. CRC Press, 2018.

Kandethody M Ramachandran and Chris P Tsokos. *Mathematical statistics with applications in R*. Academic Press, 2020.

Hana Řezanková and B Everitt. Cluster analysis and categorical data. *Statistika*, 89(3):216–232, 2009.

K Sasirekha and P Baby. Agglomerative hierarchical clustering algorithm-a. *International Journal of Scientific and Research Publications*, 83(3):83, 2013.

Gilbert Strang. *Introduction to linear algebra*. Wellesley-Cambridge Press Wellesley, MA, 5th edition, 2016.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. 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, 2001.

Robert J Tibshirani and Bradley Efron. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57(1), 1993.

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL `https://www.stats.ox.ac.uk/pub/MASS4/`. ISBN 0-387-95457-0.

Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

Florian Wickelmaier. An introduction to mds. *Sound Quality Research Unit, Aalborg University, Denmark*, 46(5):1–26, 2003.

Mohammed J Zaki and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

Mingyuan Zhou and Lawrence Carin. Negative binomial process count and mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):307–320, 2013.

Appendix

R script

Multivariate statistical analysis of thematic changes in customer feedback

```
#-------------------------------------------------------------------------------
# Preprocessing

# Loading libraries
library(tidyverse)
library(dplyr)
library(glue)
library(stringr)
library(ggplot2)
library(lubridate)
library(ISOweek)
library(matrixcalc)
library(factoextra)
library(ggthemes)
library(MASS)
library(GGally)
library(boot)

# Reading data
data <- read.csv("themes.csv", fill = T, na.strings = "", encoding="UTF-8")
Sys.setlocale("LC_TIME", "English")

# Data manipulation
data[is.na(data)] <- 0
data$date <- as.Date(data$timestamp)
data$City[data$City == 0] <- NA
data$Age[data$Age == 0] <- NA
data$Age.class[data$Age.class == 0] <- NA
data$isoweek <- isoweek(data$date)
data$year <- isoyear(data$date)
data <- data %>% filter(isoweek != 11, isoweek != 15, isoweek != 47)
data$week <- c(rep(NA, dim(data)[1]))
for(i in 1:dim(data)[1]){
  if(data$isoweek[i] == 48) data$week[i] = "48-2021"
  if(data$isoweek[i] == 49) data$week[i] = "49-2021"
  if(data$isoweek[i] == 50) data$week[i] = "50-2021"
  if(data$isoweek[i] == 51) data$week[i] = "51-2021"
  if(data$isoweek[i] == 52) data$week[i] = "52-2021"
  if(data$isoweek[i] == 1) data$week[i] =  "1-2022"
  if(data$isoweek[i] == 2) data$week[i] = "2-2022"
```

```
  if(data$isoweek[i] == 3) data$week[i] = "3-2022"
  if(data$isoweek[i] == 4) data$week[i] = "4-2022"
  if(data$isoweek[i] == 5) data$week[i] = "5-2022"
  if(data$isoweek[i] == 6) data$week[i] = "6-2022"
  if(data$isoweek[i] == 7) data$week[i] = "7-2022"
  if(data$isoweek[i] == 8) data$week[i] = "8-2022"
  if(data$isoweek[i] == 9) data$week[i] = "9-2022"
  if(data$isoweek[i] == 10) data$week[i] = "10-2022"
}
data$week <- factor(data$week, levels=c("48-2021", "49-2021", "50-2021", "51-2021", "52-2021",
                                        "1-2022", "2-2022", "3-2022", "4-2022", "5-2022",
                                        "6-2022", "7-2022", "8-2022", "9-2022", "10-2022"))

themes <- data.frame(data[, 2:21],
                     data[, 24:26],
                     data[, 28:37],
                     data[, 39:61],
                     data[, 63:85],
                     week = data[, 92],
                     sentiment = data[, 87],
                     agegroup = data[, 23],
                     city = data[, 27],
                     date = data$date,
                     isoweek = data$isoweek)
names(themes)[] <- paste0('x', 1:(ncol(themes)))
colnames(themes)[80:85] <- c("week", "sentiment", "agegroup", "city", "date", "isoweek")

#-----------------------------------------------------------------
# Data exploration

themes$themes_overall <- rowSums(themes[, 1:79])
themes_summary_daily <- themes %>% group_by(date) %>%
  summarize(Themes_overall = sum(themes_overall))
themes_summary_week <- themes %>% group_by(week) %>%
  summarize(Themes_overall = sum(themes_overall))

# Visualization
# Figure 1.1
ggplot(themes_summary_daily, aes(date, Themes_overall)) +
  geom_segment( aes(x=date, xend=date, y=0,
                    yend=Themes_overall), size=1, col = "darkblue", alpha=0.9) +
  xlab("Month") +
  ylab("Themes overall") +
  theme(axis.title.y = element_text(size = rel(2.5), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(2.5), angle = 00)) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1, size=20,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=20,color="black"))

# Figure 1.2
ggplot(themes_summary_week, aes(week, Themes_overall)) +
  geom_segment( aes(x=week, xend=week, y=0,
                    yend=Themes_overall), size=15, col = "darkblue", alpha=0.9) +
  xlab("Week") +
```

```
  ylab("Themes overall") +
  theme(axis.title.y = element_text(size = rel(2.5), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(2.5), angle = 00)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=20,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=20,color="black"))

#-----------------------------------------------------------------
# Functions for calculating similarity measures and dissimilarity matrices

# Function to calculate similarity measures between two binary vectors.
# Returns Pearson's correlation coefficient, the simple mathcing coefficient,
# Jaccard's coefficient and Yule's Q coefficient in a dataframe.
# Input: two binary vectors xi and xj.
similarity_measures <- function(xi, xj){
  a <- 0; b <- 0; c <- 0; d <- 0
  for(k in 1:length(xi)){
    if(xi[k] == xj[k] && xi[k] == 1){
      a <- a + 1
    }
    if(xi[k] == 0 && xj[k] == 1){
      b <- b + 1
    }
    if(xi[k] == 1 && xj[k] == 0){
      c <- c + 1
    }
    if(xi[k] == xj[k] && xi[k] == 0){
      d <- d + 1
    }
  }
  s_Pearson <- (a*d-b*c) / sqrt((a+b)*(a+c)*(b+d)*(c+d))
  s_SM <- (a + d) / (a + b + c + d)
  s_Jaccard <- a/(a + b + c)
  s_Q <- (a*d - b*c) / (a*d + b*c)
  return(data.frame(s_Pearson=s_Pearson, s_SM=s_SM, s_Jaccard=s_Jaccard, s_Q=s_Q))
}

# Function to calculate similarity matrix.
# Returns the similarity matrix based on the wanted similarity coefficient.
# Similarities are calculated between variables (columns).
# Input: data as a dataframe and similarity coefficient as a string.
# Uses function similarity_measures().
S_matrix <- function(data, s_measure){
  S_matrix <- matrix(NA, nrow = dim(data)[2], ncol = dim(data)[2])
  rownames(S_matrix) <- colnames(data)
  colnames(S_matrix) <- colnames(data)
  similarities <- as.data.frame(matrix(ncol = 4, nrow = 1))
  for(i in 1:dim(data)[2]){
    for(j in 1:dim(data)[2]){
      xi <- data %>% pull(i)
      xj <- data %>% pull(j)
      similarities <- similarity_measures(xi, xj)
      S_matrix[i, j] <- similarities[[s_measure]]
    }
```

```
  }
  return(S_matrix)
}
# Calculating similarity matrices
S_Pearson <- S_matrix(themes[, 1:79], "s_Pearson")
S_SM <- S_matrix(themes[, 1:79], "s_SM")
S_Jaccard <- S_matrix(themes[, 1:79], "s_Jaccard")
S_Q <- S_matrix(themes[, 1:79], "s_Q")

# Checking whether similarity matrices are positive semi-definite
is.positive.semi.definite(round(S_Pearson, 5), tol=1e-8)
is.positive.semi.definite(round(S_SM, 5), tol=1e-8)
is.positive.semi.definite(round(S_Jaccard, 5), tol=1e-8)
is.positive.semi.definite(round(S_Q, 5), tol=1e-8)

# Function to transform a similarity matrix to a dissimilarity matrix
# using Formula 7.
D_matrix <- function(S_matrix){
  D_matrix <- sqrt(2*(1 - S_matrix))
  return(D_matrix)
}
D_Pearson <- D_matrix(S_Pearson)
# Other dissimilarity matrices
# D_SM <- D_matrix(themes[, 1:79], "s_SM")
# D_Jaccard <- D_matrix(themes[, 1:79], "s_Jaccard")
# D_Q <- D_matrix(themes[, 1:79], "s_Q")
#
# Hierarchical clustering:
# When we use a dissimilarity matrix in hclust() function,
# we must plug it in in form: as.dist(D_matrix).
#
# Multidimensional scaling:
# When we use a dissimilarity matrix in cmdscale() function,
# we must plug it in in form: D_matrix.

#-----------------------------------------------------------------
# Clustering

# Hierarchical clustering with Ward's method
mclust_w <- hclust(as.dist(D_Pearson), method = "ward.D2")

# The number of clusters
# Figures 3.5, 3.6 and 3.7
set.seed(31052023)
fviz_nbclust(D_Pearson, hcut, method = "gap_stat")
fviz_nbclust(D_Pearson, hcut, method = "silhouette")
# Based on gap_stat and silhouette two clusters could be justified.

# Cutting dendrogram so that two clusters are obtained
table(cutree(mclust_w, k = 2))
cutree(mclust_w, k = 2)
# Figure 3.8
plot(mclust_w)
```

```
rect.hclust(mclust_w, k = 2, border = "red")

# Cluster memberships
groups <- as.data.frame(cutree(mclust_w, k = 2))
colnames(groups) <- "cluster"
groups$column <- seq(1:79)

# Aggregating themes for each cluster
themes$group1 <- rowSums(themes[, groups$column[groups$cluster==1]])
themes$group2 <- rowSums(themes[, groups$column[groups$cluster==2]])

# Saving the results for later use
write.csv2(themes, "clustering_ward2.csv")
# Reading the interim results
# themes <- read.csv2("clustering_ward2.csv")

themes$week <- factor(themes$week, levels=c("48-2021", "49-2021", "50-2021", "51-2021", "52-2021",
                                            "1-2022", "2-2022", "3-2022", "4-2022", "5-2022",
                                            "6-2022", "7-2022", "8-2022", "9-2022", "10-2022"))
theme_summary <- themes %>% group_by(week) %>%
  summarize(group1 = sum(group1), group2 = sum(group2),
            overall_feedback = length(x1))

# Data to long form
theme_groups <- as.data.frame(theme_summary)
themes_l <- reshape(theme_groups, direction = "long", idvar = "week",
                    times = c(1:2), varying = list(c("group1", "group2")))

# A Color palette for visualizations
bluePalette <- c("#03396c",  "#0000FF", "#7393B3",
                      "#088F8F", "#0096FF", "#ADD8E6", "#00FFFF")

# Figure 3.9
ggplot(themes_l, aes(week, group1/overall_feedback, colour=as.factor(time))) +
  geom_segment( aes(x=week, xend=week, y=0,
                    yend=group1/overall_feedback), size=5, alpha=0.9) +
  facet_wrap(~as.factor(time)) +
  scale_colour_manual(values=bluePalette, name="Group",
                      labels=c("1","2","3","4","5", "6", "7")) +
  xlab("Week") +
  ylab("Themes from group per feedback")+
  theme(axis.title.y = element_text(size = rel(1.8), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(1.8), angle = 00)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=12,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=12,color="black"))+
  theme(legend.title=element_text(size=15), legend.text=element_text(size=15)) +
  theme(panel.border=element_blank()) +
  theme(axis.line=element_line())

# Next, we investigate further seven themes in Group 1.
theme_summary_group1 <- themes %>% group_by(week) %>%
  summarize(x1 = sum(x1), x26 = sum(x26),
            x30 = sum(x30), x33 = sum(x33),
```

```
            x37 = sum(x37), x55 = sum(x55),
            x60 = sum(x60),
            overall_feedback = length(x2))
themes_group1 <- as.data.frame(theme_summary_group1)
teemat_l2 <- reshape(themes_group1, direction = "long", idvar = "viikko",
                     times = c(1:7), varying = list(c("x1", "x26","x30","x33",
                                                      "x37", "x55", "x60")))

# Figure 3.10
ggplot(teemat_l2, aes(week, x1/overall_feedback, colour=as.factor(time))) +
  geom_segment( aes(x=week, xend=week, y=0,
                    yend=x1/overall_feedback), size=5, alpha=0.9) +
  facet_wrap(~as.factor(time)) +
  scale_colour_manual(values=bluePalette, name="Theme",
                      labels=c("x1","x26","x30","x33","x37", "x55", "x60")) +
  xlab("Week") +
  ylab("Themes per feedback")+
  theme(axis.title.y = element_text(size = rel(1.8), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(1.8), angle = 00)) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1, size=8,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=8,color="black"))+
  theme(legend.title=element_text(size=15), legend.text=element_text(size=15)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=12,color="black"))

#-----------------------------------------------------------------
# Multidimensional scaling
# Metric MDS
# Numerical example with four themes
D <- D_Pearson[1:4, 1:4]
A <- data.frame(matrix(nrow=4, ncol=4))
for(i in 1:4){
  for(j in 1:4){
    A[i, j] <- -1/2 * D[i, j]^2
  }
}
I <- diag(4)
n <- dim(D)[1]
H<- I - n^(-1) * matrix(rep(1, n*n), nrow = n, ncol = n)
B <- H %*% A %*% H
eigenvalues <- eigen(B)
eigenvectors <- eigenvalues$vectors
E <- as.matrix(data.frame(eigenvectors[, 1], eigenvectors[, 2]))
Lambda <- as.matrix(cbind(c(eigenvalues$values[1], 0),
                          c(0, eigenvalues$values[2])))

# Proportion of variance explained by two dimensions
(eigenvalues$values[1] + eigenvalues$values[2])/sum(eigenvalues$values)
# 71 %
X <- E %*% Lambda ^(1/2)
round(X, 2)
plot(X, xlab="Dimension 1", ylab="Dimension 2", xlim=range(-1, 1),
     ylim=range(-1, 1),  pch = 19, cex.lab=1.5, cex=1.5, cex.axis=1.5)
```

```r
# Metric MDS applied on all the themes
mds <- cmdscale(D_Pearson,eig=TRUE, k=3) # k is the number of dim
MDS_xyz <- data.frame(dim1 = mds$points[,1],
                      dim2 = mds$points[,2],
                      dim3 = mds$points[,3])
# Scree plot
plot(mds$eig, type = "b", xlab = "Dimension", ylab = "Eigenvalue")
eigenvalues <- mds$eig
# Check that eigenvalues are positive
eigenvalues > 0
sum(eigenvalues[1:3])/(sum(eigenvalues))
# Proportion of variance explained by three dimensions is 11 %

# Is B=HAH positive semidefinite?
n <- dim(D_Pearson)[1]
A <- -1/2 * D_Pearson^2
I <- diag(79)
H<- I - n^(-1) *  matrix(rep(1, n*n), nrow = n, ncol = n)
B <- H %*% A %*% H
is.positive.semi.definite(round(B, 5), tol=1e-8)

# Figure 4.2
ggpairs(MDS_xyz[1:3], aes( ), upper = list(continuous = wrap("density", alpha = 1),
                                           combo = "box"),
        lower = list(continuous = wrap("points", alpha = 3, size=2),
                     combo = wrap("dot", alpha = 1, size=1) )) +
  theme(axis.text=element_text(size=10)) +
  theme(strip.text.x = element_text(size = 15),
        strip.text.y = element_text(size = 15))

MDS_xyz$Cluster <- cutree(mclust_w, k = 2)
# Figure 4.3
ggpairs(MDS_xyz[1:3], mapping = ggplot2::aes(color = as.factor(MDS_xyz$Cluster),
                                             pch=as.factor(MDS_xyz$Cluster),
                                             fill = factor(MDS_xyz$Cluster)),
        lower = list(continuous = wrap("points", alpha = 1,    size=2.5)),
        upper = list(continuous = wrap("density", alpha = 1), combo = "box")
)+
  theme(strip.text.x = element_text(size = 15),
        strip.text.y = element_text(size = 15)) +
  scale_shape_manual(values=c(15:20, 7))

#-----------------------------------------------------------------
# Bootstrap confidence intervals

# Preprocessing
# Reading the clustering results
# themes <- read.csv2("clustering_ward2.csv")
theme_group1 <- data.frame(group1 = themes$group1,
                           isoweek = themes$isoweek)
for(i in 1:dim(theme_group1)[1]){
  if(theme_group1$isoweek[i] == 48) theme_group1$weeknumber[i] = 1
  if(theme_group1$isoweek[i] == 49) theme_group1$weeknumber[i] = 2
```

```
  if(theme_group1$isoweek[i] == 50) theme_group1$weeknumber[i] = 3
  if(theme_group1$isoweek[i] == 51) theme_group1$weeknumber[i] = 4
  if(theme_group1$isoweek[i] == 52) theme_group1$weeknumber[i] = 5
  for(j in 1:10){
    if(theme_group1$isoweek[i] == j) theme_group1$weeknumber[i] = 5+j
  }
}

# Function to calculate bootstrap confidence intervals
confidence_intervals <- function(data) {
  lower <- rep(NA, 15)
  upper <- rep(NA, 15)
  estimate <- rep(NA, 15)
  weeknumber <- rep(NA, 15)
  N = 10000
  for(i in 1:15){
    # Following Algorithm 2 Bootstrap confidence intervals for the mean
    # Steps 1 and 2
    bootstrap <- boot(data$group1[data$weeknumber == i], function(u, k) mean(u[k]),R=N)
    estimate[i] <- bootstrap$t0
    # Step 3
    arranged <- sort(bootstrap$t)
    # Step 4
    lower[i] <- arranged[round((0.025)*(N+1), 0)]
    upper[i] <- arranged[round((0.975)*(N+1), 0)]
    weeknumber[i] = i
  }
  return(data.frame(estimate, lower, upper, weeknumber))
}
confidence_intervals <- confidence_intervals(theme_group1)
confidence_intervals$week <- c(rep(NA, dim(confidence_intervals)[1]))
confidence_intervals$week[confidence_intervals$weeknumber==1] <- "48-2021"
confidence_intervals$week[confidence_intervals$weeknumber==2] <- "49-2021"
confidence_intervals$week[confidence_intervals$weeknumber==3] <- "50-2021"
confidence_intervals$week[confidence_intervals$weeknumber==4] <- "51-2021"
confidence_intervals$week[confidence_intervals$weeknumber==5] <- "52-2021"
confidence_intervals$week[confidence_intervals$weeknumber==6] <- "1-2022"
confidence_intervals$week[confidence_intervals$weeknumber==7] <- "2-2022"
confidence_intervals$week[confidence_intervals$weeknumber==8] <- "3-2022"
confidence_intervals$week[confidence_intervals$weeknumber==9] <- "4-2022"
confidence_intervals$week[confidence_intervals$weeknumber==10] <- "5-2022"
confidence_intervals$week[confidence_intervals$weeknumber==11] <- "6-2022"
confidence_intervals$week[confidence_intervals$weeknumber==12] <- "7-2022"
confidence_intervals$week[confidence_intervals$weeknumber==13] <- "8-2022"
confidence_intervals$week[confidence_intervals$weeknumber==14] <- "9-2022"
confidence_intervals$week[confidence_intervals$weeknumber==15] <- "10-2022"

confidence_intervals$week <- factor(confidence_intervals$week,
                                    levels=c("48-2021", "49-2021", "50-2021",
                                             "51-2021", "52-2021", "1-2022",
                                             "2-2022", "3-2022", "4-2022",
                                             "5-2022", "6-2022", "7-2022",
                                             "8-2022", "9-2022", "10-2022"))
```

```r
# Figure 5.1
ggplot(confidence_intervals, aes(week, estimate)) +
  geom_segment( aes(x=week, xend=week, y=0,
                    yend=estimate), size=22, col = "#1B9E77", alpha=0.9) +
  xlab("Week") +
  ylab("Themes from group 1 per feedback") +
  theme(axis.title.y = element_text(size = rel(2.5), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(2.5), angle = 00)) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1, size=12,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=12,color="black"))+
  theme(legend.title=element_text(size=15), legend.text=element_text(size=15)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=15,color="black")) +
  geom_errorbar(aes(x=week, ymin = low, ymax = high), width = 1)

#-----------------------------------------------------------------
# Negative binomial regression

nbr <- glm.nb(group1 ~ week - 1, data=themes, link = "log")
summary(nbr)
coef <- coefficients(nbr)
low <- exp(confint(nbr))[, 1]
high <- exp(confint(nbr))[, 2]
estimate <- exp(coef)

negative_binomial_regression <- data.frame(week = theme_summary$week,
                                           estimate = estimate,
                                           low = low,
                                           high = high)
# Figure 6.1
ggplot(negative_binomial_regression, aes(week, estimate)) +
  geom_segment( aes(x=week, xend=week, y=0,
                    yend=estimate), size=22, col = "#1B9E77", alpha=0.9) +
  xlab("Week") +
  ylab("Themes from group 1 per feedback") +
  theme(axis.title.y = element_text(size = rel(2.5), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(2.5), angle = 00)) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1, size=12,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=12,color="black"))+
  theme(legend.title=element_text(size=15), legend.text=element_text(size=15)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=15,color="black")) +
  geom_errorbar(aes(x=week, ymin = low, ymax = high), width = 1)

# Sentiment
nbr_sentiment <- glm.nb(group1 ~ as.factor(sentiment), data=themes, link = "log")
summary(nbr_sentiment)
themes_sentiment <- themes %>% group_by(sentiment, week) %>%
  summarize(group1 = sum(group1),
            overall_feedback = length(x1),
            ratio = sum(group1)/length(x1))

# Figure 6.2
ggplot(themes_sentiment, aes(week, ratio)) +
  geom_segment( aes(x=week, xend=week, y=0,
```

```
                    yend=ratio, colour = as.factor(sentiment)), size=5, alpha=0.9) +
  facet_wrap(~as.factor(sentiment)) +
  xlab("Week") +
  ylab("Themes from group 1 per feedback")+
  theme(axis.title.y = element_text(size = rel(1.8), angle = 90)) +
  theme(axis.title.x = element_text(size = rel(1.8), angle = 00)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size=12,color="black")) +
  theme(axis.text.y = element_text(angle = 0, hjust = 1, size=12,color="black"))+
  theme(legend.title=element_text(size=15), legend.text=element_text(size=15)) +
  theme(panel.border=element_blank()) +
  scale_colour_manual(values=c("darkred","orange","darkgreen"),
                      labels = c("Negative", "Neutral", "Positive")) +
  guides(color=guide_legend("Sentiment")) +
  theme(axis.line=element_line())

#-------------------------------------------------------------------------------------------
```