

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Okruhlica** Jméno: **Štefan** Osobní číslo: **384491**  
Fakulta/ústav: **Fakulta elektrotechnická**  
Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**  
Studijní program: **Otevřená informatika**  
Specializace: **Interakce člověka s počítačem**

## II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

**Shlukování a vizualizace tabulkových dat s nominálními atributy**

Název diplomové práce anglicky:

**Clustering and visualization of tabular data with nominal attributes**

Pokyny pro vypracování:

Seznamte se s technikami pro vizualizaci tabulkových dat. Zaměřte se zejména na vizualizační techniky, které umožňují detekovat shluky dat a vlastnosti těchto shluků. Dále se seznamte s technikami pro shlukování dat s nominálními a ordinálními atributy. Na základě analýzy navrhnete a implementujete algoritmus pro shlukování dat s nominálními a ordinálními atributy. Dále navrhnete jak detekované shluky a jejich vlastnosti vizualizovat v rozšířeném nástroji XDAT (výsledek jiné diplomové práce [4]). Algoritmus pro shlukování dat a navrženou vizualizaci do nástroje XDAT integrujte. Výsledný algoritmus pro shlukování dat a vizualizaci detekovaných shluků otestujte alespoň na pěti rozdílných sadách tabulkových dat se vzrůstajícím počtem atributů (minimum jsou 4 atributy).

Seznam doporučené literatury:

- [1] T. Munzner, Visualization design and analysis, CRC Press, 2015.
- [2] Z. He, S. Deng, and X. Xu. Approximation algorithms for k-modes clustering. In Proceedings of the 2006 international conference on Intelligent computing: Part II (ICIC'06), pp. 296–302. Springer-Verlag, Berlin, Heidelberg, 2006.
- [3] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2, pp. 283–304, Springer-Verlag, Berlin, Heidelberg, 1998.
- [4] M. Janda, Vizualizace n-rozměrných heterogenních dat, Diplomová práce, ČVUT v Praze, 2017.

Jméno a pracoviště vedoucí(ho) diplomové práce:

**Ing. Ladislav Čmolík, Ph.D., Katedra počítačové grafiky a interakce**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **10.03.2021**

Termín odevzdání diplomové práce: **13.08.2021**

Platnost zadání diplomové práce: **19.02.2023**

Ing. Ladislav Čmolík, Ph.D.  
podpis vedoucí(ho) práce

podpis vedoucí(ho) ústavu/katedry

prof. Mgr. Petr Páta, Ph.D.  
podpis děkana(ky)

## III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

\_\_\_\_\_  
Datum převzetí zadání

\_\_\_\_\_  
Podpis studenta

Master Thesis



Czech  
Technical  
University  
in Prague

**F3**

Faculty of Electrical Engineering  
Department of Computer Graphics and Interaction

## Clustering and Visualization of Tabular Data with Nominal Attributes

Bc. Štefan Okruhlica

Supervisor: Ing. Ladislav Čmolík, Ph.D.  
Field of study: Open Informatics  
Subfield: Human-Computer Interaction  
August 2021



## Acknowledgements

I thank my research project supervisor Ing. Ladislav Čmolíkov, Ph.D., for all of his professional help, guidance, and patience.

## Declaration

I declare that I have worked on this thesis independently and that I have stated all of the used information sources following the Dean's directive for diploma thesis and the comprehensive final examination in Bachelor's and Master's programs at the FEE at CTU.

In Prague 13. August 2021

## Abstract

This master's thesis discusses the different approaches to the clustering and subsequent visualization of nominal and ordinal data. As such, it explains the tabular dataset, defines attribute types, and outlines its visualization task. It explains the several clustering methods (hierarchical clustering, k-means, k-modes, parallelogram clustering) for clustering nominal data and the Dice similarity measure for counting distance. It also contains the description of the used algorithm - parallelogram clustering. Finally, it shows how the algorithm was implemented into the XDat tool and subsequently shows the experimental results of four additional datasets.

**Keywords:** visualization, clustering, nominal data, tabular data, k-modes

**Supervisor:** Ing. Ladislav Čmolík, Ph.D.

## Abstrakt

Tato diplomová práce se zabývá různými přístupy ke clusterování a následné vizualizaci nominálních a ordinálních dat. Ukazuje tabulkové datasety, definuje typy atributů a popisuje úkoly pro vizualizaci. Dále vysvětluje různé clusterovací metody (hierarchické clusterování, k-means, k-modes, paralelogramové clusterování) pro clusterování nominálních dat. K výpočtu vzdálenosti je vystvěleno Dicova míra podobnosti. Ukazuje jakým způsobem byl implementován použitý algoritmus paralelogramového clusterování v nástroji XDat. Výsledky jsou prezentovány na čtyř datasetech.

**Klíčová slova:** vizualizace, clustrování, nominální data, tabulková data, k-modes

# Contents

<b>1 Introduction</b>	<b>1</b>
1.1 Structure .....	1
<b>2 Analysis and Design</b>	<b>3</b>
2.1 Tabular Data .....	3
2.1.1 Spatial Data .....	4
2.1.2 Other Abstract Data .....	4
2.2 Attribute Types .....	4
2.2.1 Nominal/Categorical Attributes	4
2.2.2 Ordered Attributes .....	4
2.3 Visualization tasks .....	5
2.4 Particularities of Nominal Data ..	7
2.5 Visualization of Tabular Data ..	12
2.5.1 Parallel Coordinates .....	12
2.5.2 Scatterplot Matrix .....	14
2.5.3 Mosaic Plot .....	15
2.5.4 Dimension Reduction Methods	16
2.5.5 Parallel Sets .....	17
2.6 Clustering of Tabular Data .....	20
2.6.1 Dice Measure .....	21
2.6.2 Adapted Priority Dice Measure	22
2.6.3 Silhouette .....	25
2.6.4 Hierarchical Clustering .....	25
2.6.5 Transforming Nominal Data to Quantitative Data .....	27
2.6.6 K-Modes .....	27
2.6.7 Parallelogram Clustering .....	39
<b>3 Implementation</b>	<b>45</b>
3.1 Used Technologies .....	45
3.2 XDat .....	45
3.3 Structure of Classes .....	46
3.4 Parallelogram Method User Interface .....	48
<b>4 Experimental Results</b>	<b>53</b>
4.1 Cars Dataset .....	53
4.2 States Dataset .....	57
4.3 New York Hotels Dataset .....	60
4.4 Election Dataset .....	64
<b>5 Conclusion</b>	<b>69</b>
<b>A Bibliography</b>	<b>71</b>

## List of Figures

2.1 Comparison of survived and perished passengers of Titanic [Davne]. . . . .	6	2.17 Mosaic plot method used on Titanic dataset with only nominal and ordinal attributes. . . . .	17
2.2 Composition of passengers of Titanic based firstly on sex and secondly on class [Davne]. . . . .	6	2.18 Parallel sets method used on Titanic dataset [Davne]. . . . .	17
2.3 Visualization of the iris dataset using the PCA method [Hal20]. . . . .	8	2.19 Parallel sets method used on Titanic dataset using brushing [Davne]. . . . .	18
2.4 Visualization of the example subset of the Titanic dataset . . . . .	9	2.20 Parallel sets method used on a dataset of mushroom characteristics. . . . .	18
2.5 Visualization of the distances in the example subset of the Titanic dataset. . . . .	9	2.21 Parallel sets method used on Titanic dataset using the bundle layout. . . . .	19
2.6 All possible selections of centroids for the first case of clustering on the example dataset. . . . .	10	2.22 Parallel sets method used on Titanic dataset using the tree layout. . . . .	20
2.7 Solution for the first case of clustering on the example dataset . . . . .	10	2.23 Parallel sets method used on a dataset of mushroom characteristics using the tree layout. . . . .	20
2.8 Both possible selections of centroids for the second case of clustering on the example dataset. . . . .	11	2.24 Visualization of prioritized distances for the representative subset of the Titanic dataset. Selection of centroids (left), outcome of clustering (right). . . . .	24
2.9 All four possible solutions for the second case of clustering on the example dataset. . . . .	11	2.25 Visualization of the first four levels of hierarchy created from the Titanic dataset. . . . .	27
2.10 Example of parallel coordinates method [Kosne]. . . . .	12	2.26 Comparison for random clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right). . . . .	31
2.11 Example of brushing on parallel coordinates method [Kosne]. . . . .	13	2.27 Comparison for column clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right). . . . .	33
2.12 Example of a cluttered visualization using parallel coordinates method [HWne]. . . . .	13		
2.13 Common patterns in Cartesian coordinates (top) and their dual representation in parallel coordinates (bottom) [HWne]. . . . .	13		
2.14 Parallel coordinates method used on Titanic dataset with only nominal and ordinal attributes. . . . .	14		
2.15 Complete scatterplot matrix without color (left) upper panel of colored scatterplot matrix based on iris species (right) for the iris dataset [STHne]. . . . .	15		
2.16 Parallel coordinates method used on Titanic dataset with only nominal and ordinal attributes. . . . .	16		

2.28 Comparison for occurrence clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	36	2.33 Comparison for parallelogram clustering with threshold of 0.6. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	44
2.29 Comparison for two-occurrence clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	39	3.1 Showcase of added and packages and class (in cyan) and modified classes and packages (in cyan and in cursive) relevant to the implemented parallelogram clustering method. .	46
2.30 Comparison for parallelogram clustering with threshold of 0.0. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	41	3.2 Showcase of all added method (except those shown in Figure 3.1) that were used for all tested clustering methods. ....	47
2.31 Comparison for parallelogram clustering with threshold of 0.2. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	42	3.3 Main menu of the XDat tool without any loaded data. New menu item "Cluster" is highlighted in red.	49
2.32 Comparison for parallelogram clustering with threshold of 0.4. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).....	43	3.4 Main menu of the XDat tool after the load of the Titanic dataset. ...	49
		3.5 Extension of the "Cluster" menu item of the main menu showing the "Cluster Using Parallelograms" option highlighted in red. ....	50
		3.6 Dialog window for the parallelogram clustering method. Highlighted in red are slider to set the threshold for the clustering (1), input fields for setting of priority for the priority vector (2) and a selection for the primary attribute (3).....	50
		3.7 Showcase of all added method (except those shown in Figure 3.1) that were used for all tested clustering methods. ....	51
		4.1 Comparison for parallelogram clustering of the Cars dataset with <b>threshold of 0.1</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	55



4.2 Comparison for parallelogram clustering of the Cars dataset with <b>threshold of 0.2</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	55
4.3 Comparison for parallelogram clustering of the Cars dataset with <b>threshold of 0.3</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	56
4.4 Comparison for parallelogram clustering of the Cars dataset with <b>threshold of 0.4</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	56
4.5 Comparison for parallelogram clustering of the States dataset with <b>threshold of 0.1</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	59
4.6 Comparison for parallelogram clustering of the States dataset with <b>threshold of 0.2</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	59
4.7 Comparison for parallelogram clustering of the States dataset with <b>threshold of 0.3</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	60
4.8 Comparison for parallelogram clustering of the New York Hotels dataset with <b>threshold of 0.2</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	62
4.9 Comparison for parallelogram clustering of the New York Hotels dataset with <b>threshold of 0.3</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	63
4.10 Comparison for parallelogram clustering of the New York Hotels dataset with <b>threshold of 0.4</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	63
4.11 Comparison for parallelogram clustering of the Election dataset with <b>threshold of 0.1</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).	66

4.12 Comparison for parallelogram clustering of the Election dataset with <b>threshold of 0.3</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right). . . . .	67
4.13 Comparison for parallelogram clustering of the Election dataset with <b>threshold of 0.5</b> . Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right). . . . .	67

## List of Tables

2.1 Representative subset of the Titanic dataset. . . . .	7
2.2 Sums of the subset of the Titanic dataset. . . . .	8
2.3 Representative subset of the Titanic dataset with only unique data points. . . . .	22
2.4 Representative subset of the Titanic dataset with only unique data points transformed into dataset with only binary attributes. . . . .	22
2.5 Distances between unique data points of the representative subset of the Titanic dataset. . . . .	22
2.6 Prioritized distances between unique data points of the representative subset of the Titanic dataset. . . . .	24
2.7 Modes for cluster resulting from the random method. . . . .	31
2.8 Modes for cluster resulting from the random method. . . . .	33
2.9 Sums for the occurrence method when attribute "Class" was selected as a basis for clustering. . . . .	35
2.10 Initial modes for clusters resulting from the occurrence method. . . . .	35
2.11 Final modes for clusters resulting from the occurrence method. . . . .	36
2.12 Sums for the occurrence method when attribute "Class" was selected as a basis for clustering. . . . .	38
2.13 Initial and final modes for clusters resulting from the occurrence method. . . . .	38





# Chapter 1

## Introduction

Visualizing nominal and mixed data of high dimension is a complex task that requires hours of work to analyze the data and experiment with different visualization methods. This thesis aims to ease some of these difficulties by offering a visualization of clustered nominal data. In addition, the offered visualization highlights some underlying trends present in the dataset and can help with subsequent data analysis, clustering, and visualization.

We have almost always problems when visualizing tabular nominal or mixed data. We must deal with the high dimension of these data, but we have to also work around the particularities that come with nominal data. High-dimensional data requires visualization methods suited for high dimension or dimension reduction methods which inevitably introduce inaccuracy to our visualization. Concerning the mentioned particularities of nominal data, the chief amongst them is our inability to compare them. We will demonstrate this issue in comparison to numeric data. Let us have a man and woman. We can readily compare their height, a numeric data attribute, for example, on a linear axis. However, we will have a much harder time comparing their sex. If we also decide to visualize it on a linear axis, which sex will we place on the left and which will we place on the right? Is one more important than the other? There are no definite answers here, which is one reason why visualization of nominal data requires an in-depth data analysis.

For this purpose, we would like to offer a tool that will ease the work with the analysis of nominal data. The clustered dataset that we will offer will at the very least help inform the user about the trends present in the data and, in so doing, will decrease the amount of time needed to analyze the data. Furthermore, the created cluster can be further utilized to extract centroid items from them for different clustering methods or to inform of trends present in the data or that no trend could be found in the dataset.



### 1.1 Structure

Beginning with the analysis of the data, we start with chapter 2 Analysis and Design. In this chapter, we gradually discuss the type of dataset that we will use in chapter 2.1 Tabular Data. We continue to showcase the different types of attributes present in any dataset. I discuss the nominal, ordinal and



## Chapter 2

### Analysis and Design

In this chapter, the type of dataset that will be used will be introduced (chapter 2.1 Tabular Data). It is, generally speaking, a large tabular dataset of high dimensions. I will also discuss the various types of attributes (chapter 2.2 Attribute Types) that can be present in any dataset. I will be using nominal and ordinal attributes, quantitative attributes, although present in many datasets, will not be used in our computations. Furthermore, I will speak about the visualization tasks (chapter 2.3 Visualization tasks) that I expect to fulfill with our visualization. Continuing with some basic discussion about some of the more problematic aspects of work with nominal data (chapter 2.4 Particularities of Nominal Data). I then discuss the different visualization methods that I could use for the purposes in chapter 2.5 Visualization of Tabular Data. The chapter is finished with discussion about clustering methods in chapter 2.6 Clustering of Tabular Data

### 2.1 Tabular Data

The primary and sole focus of this thesis is tabular datasets. Tabular datasets are datasets presented and saved as a table, meaning that the dataset is divided into rows and columns of values. Each column of the datasets represents a single attribute, each row of the dataset represents a single item. Every intersection of any column with any row is called a cell, and it represents a value of the selected attribute of the selected item [MM15].

I expect datasets with high dimensionality, meaning that the datasets have many columns (or attributes). Datasets with high dimensions require visualization methods that can work with many attributes or methods for dimension reduction. I have chosen the former and will visualize the dataset using the parallel sets method. I also expect large datasets, which means many rows (or items) present in the dataset. I discuss the mentioned particularities in chapter 2.4 Particularities of Nominal Data.

Let us also mention that there are other types of datasets. Though these datasets will not be used and further discussed in this thesis, it is vital to understand their scope.



Together with nominal attributes, ordinal attributes are the second focus of this thesis. Unlike nominal attributes, it is much easier to visualize ordinal attributes since they have an order. Having order means I visualize them either in ascending or descending order.

### ■ Quantitative Attributes

Quantitative attributes are ordered attributes at measurable intervals, which allow the use of arithmetic operations [MM15]. Again to put it simply, they are numbers numeric in nature. I can further divide them into discrete and continuous attributes [UC ne].

I do not use quantitative attributes in this thesis. However, quantitative attributes may be present since I often have a mixed data set, meaning that the dataset attributes are not of a particular type. If I use such a dataset, quantitative attributes are entirely avoided and not used in the computations.

## ■ 2.3 Visualization tasks

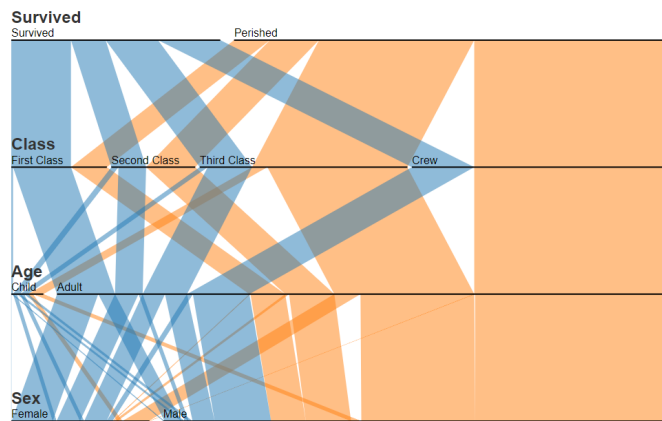
Since I am visualizing nominal tabular datasets, I would like to use this visualization to discover trends present in the data. I can use this to discover trends that can have commercial usage, especially in market research and marketing research studies. Nevertheless, it can also be used to learn new information and make connections about historical data [MM15].

For example, let us focus on historical data concerning the Titanic, specifically its passengers. The used dataset has four attributes, three nominal and one ordinal. The three nominal attributes are "Survived" (with two categories, "Survived" and "Perished"), "Age" (with two categories "Child" and "Adult"), and "Sex" (with two categories "Male" and "Female"). The ordinal attribute is "Class" with four categories "First Class", "Second Class", "Third Class", and "Crew". In Figure 2.1 I can analyze the proportions of passengers that survived and that perished. The blue parts of the plot show passengers that survived, the orange parts show passengers that perished. These parts allow us to see that mainly the first and second class passengers survived, while the passengers from the third class and the crew perished.

The second task of the visualization is to discern clusters of data. These clusters would represent items that are similar to the other items in the same cluster and at the same time dissimilar in another way to items from different clusters. When these clusters can be discerned and understood, I can use them for many purposes. For example, they can be used to make personas. A persona is a simplified model for a user/customer that can be used as a stand-in and has specific characteristics. I can use user/customer data clustering to find a basis for these personas and build the personas around them.

Let us again demonstrate an example of the passengers of Titanic in Figure 2.2. Let us build three personas. I will omit the age attribute since the majority of the passengers were adults. The first is a woman from the

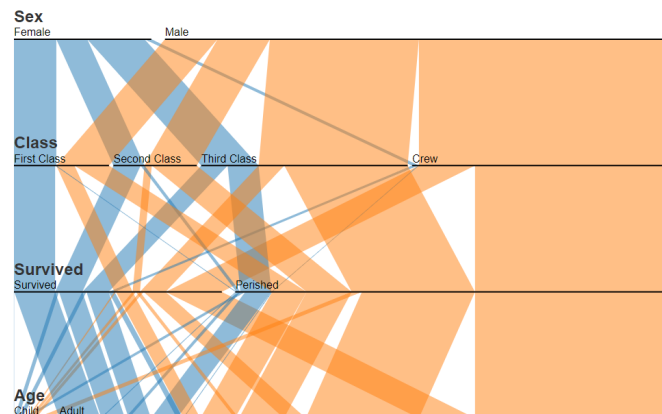




**Figure 2.1:** Comparison of survived and perished passengers of Titanic [Davne].

first or second class, and the second is a man also from the first or second class. I can see with both these personas that their fates were quite similar. The women mostly survived, the men mostly perished, but some survived. A third persona is a man either from the third class or the crew, who mostly perished.

Now let us assume that I have a company producing titanic commemorative products. I can already see that I should focus on the first persona (a woman from first or second class) and produce more high-end products directed at women. I can take a minor focus on the second persona (a man from first or second class) with more high-end luxury items. I should avoid making any products for the third persona (a man from the third class or crew) since I know they have a lower income based on their class, and less of them survived. I can assume that they have a more negative connotation associated with titanic. On the other hand, I see that a significant number of the survived compared to the survivors for the first and second personas, so a market for the third persona could exist.



**Figure 2.2:** Composition of passengers of Titanic based firstly on sex and secondly on class [Davne].

Even though this example might have been a bit morbid, I used it only to demonstrate the usage of clustering of nominal data. The proposed personas are very shallow and would be of no particular use to anyone.

## 2.4 Particularities of Nominal Data

Before I explain the various visualization methods for nominal data and which method I used, and why allow me to explain some of the difficulties connected to nominal data.

First, lets have an example dataset. In Table 2.1 we can see an subset of the Titanic dataset used in figures 2.1 and 2.2. This example dataset is in scale to the original dataset.

The dataset consists of two attributes. They are "Survived" (with two categories, "Survived" and "Perished") and "Sex" (with two categories, "Male" and "Female"). Both of these attributes are nominal.

Item ID	Sex	Survived
1	Female	Survived
2	Female	Survived
3	Female	Perished
4	Male	Survived
5	Male	Survived
6	Male	Survived
7	Male	Perished
8	Male	Perished
9	Male	Perished
10	Male	Perished
11	Male	Perished
12	Male	Perished
13	Male	Perished
14	Male	Perished
15	Male	Perished

**Table 2.1:** Representative subset of the Titanic dataset.

The first thing we should mention is a limited number of unique items present in any dataset consisting of nominal, ordinal, or nominal and ordinal data. Let  $n$  be a number of attributes, let  $X_i$  be the  $i$ -th attribute of the dataset, and let  $|X_i|$  be a number of categories of the  $i$ -th attribute. The total number of unique items that can be present in such a dataset is:

$$\prod_{i=1}^n |X_i|. \tag{2.1}$$

In the case of our example dataset in the Table 2.1 it is  $2 \cdot 2 = 4$  unique items. In the case of the original titanic dataset, it is 32 unique items. In both cases, the number of possible unique items is much smaller than all

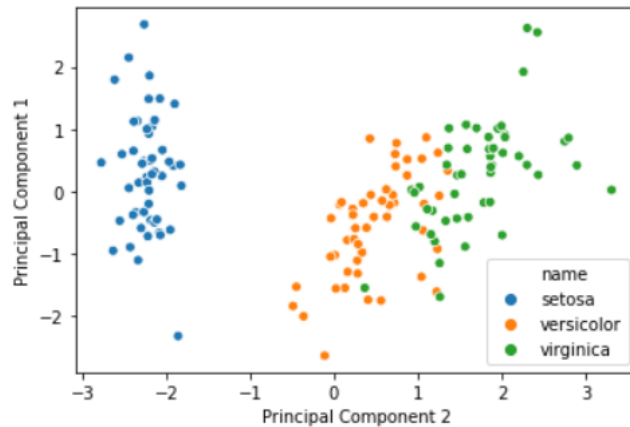
items. In the case of our example, it is four unique items with 15 items in total. For the original dataset, it is 32 unique items with 2201 items in total. The inevitable conclusion is that the dataset consists of a high degree of duplicate items.

Let us now have a look at another table. Table 2.2 shows the sums of the example dataset.

	Survived	Perished	Sum
Female	2	1	3
Male	3	9	12
Sum	5	10	15

**Table 2.2:** Sums of the subset of the Titanic dataset.

Now, we would like to visualize this dataset to depict every item as a dot in a two-dimensional graph. One axis is the attribute "Sex", and the other is the attribute "Survived". What we would like to see is something similar, as shown in Figure 2.3. In this figure, we can see each item having one point in the graph.

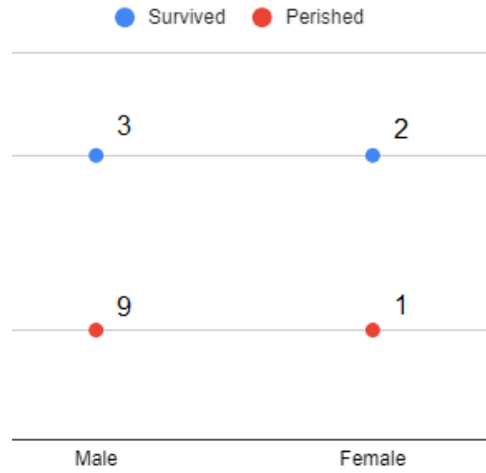


**Figure 2.3:** Visualization of the iris dataset using the PCA method [Hal20].

When we attempt to visualize our example dataset in this manner, the outcome is very different. For example, we can see in Figure 2.4 that only four points are visible in the visualization. Therefore, we can augment the visualization by adding further information about the number of items associated with each point. In this case, with a number, we could also use different sizes of dots, for example.

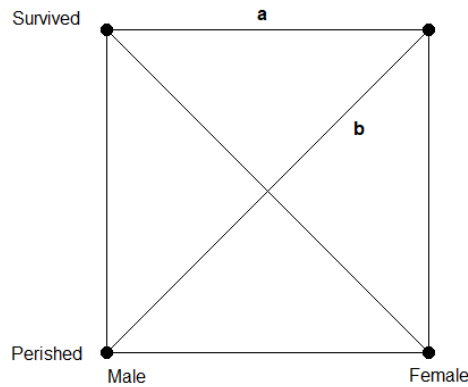
Not only is this quality of nominal data problematic from the point of view of visualization, but it is more problematic for clustering. Let us have a look at Figure 2.5. We can see that the plot is the same as the plot in Figure 2.4, except I have added lines representing the distances between every point. At this point, it is not particularly important how we computed these lengths. Let us assume we have and that it holds  $a < b$ . All four sides have length  $a$ ,

and both diagonals have length  $b$ .



**Figure 2.4:** Visualization of the example subset of the Titanic dataset

Before I show the problems associated with clustering, I will give a brief explanation of clustering. You will find detailed explanation of clustering with description of various methods in chapter 2.6 Clustering of Tabular Data.



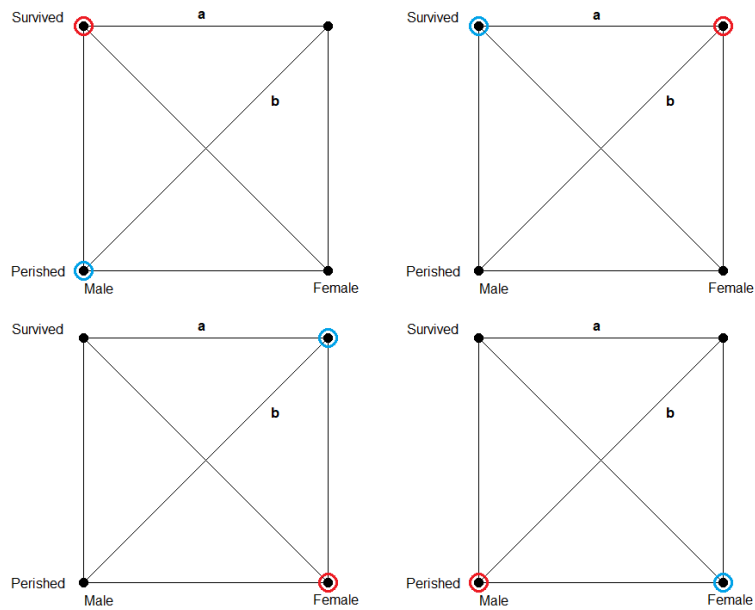
**Figure 2.5:** Visualization of the distances in the example subset of the Titanic dataset.

Clustering is the process of grouping items from a dataset into distinct groups. In other words, it is a partitioning of the dataset into disjunctive subsets. Items in a cluster should be in some way similar to each other and at the same time dissimilar to items from other clusters. This similarity and dissimilarity measure is determined based on the distance between the items. It can also be determined based on a distance measure of the items in question.

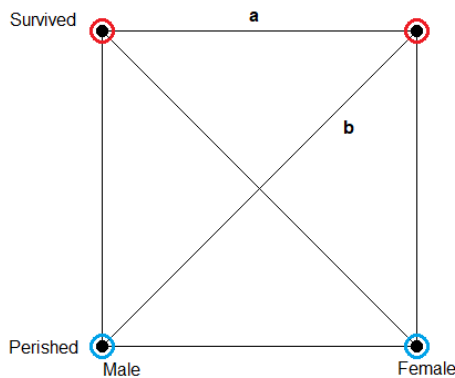
In some clustering methods, we select  $k$  distinct items from the dataset,  $k$  being the number of clusters we want. We call these items centroids, and we

gradually attach all other items to these centroids based on their distances from these centroids. I describe this method called k-means since I will be using it to demonstrate the problems with clustering of nominal data.

Let us discuss two different situations that can happen when we will attempt to cluster the example date shown in Figure 2.5. The first case is when both centroids are located on the same side of the graph, meaning their distance is  $a$ . There are four possible selections as shown in Figure 2.6. The first cluster item is highlighted in red (also the centroid of the first cluster). The second cluster is highlighted in blue.



**Figure 2.6:** All possible selections of centroids for the first case of clustering on the example dataset.

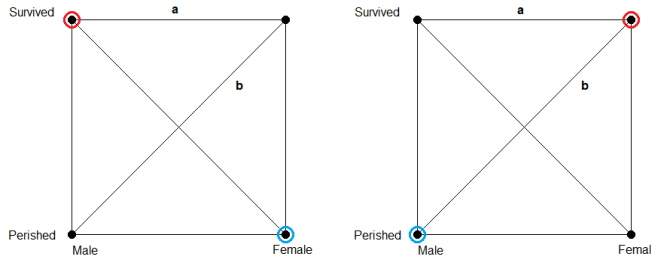


**Figure 2.7:** Solution for the first case of clustering on the example dataset

Let us only focus on the figure in the top left corner since all the other cases are analog. We can see that there is only one way of clustering the

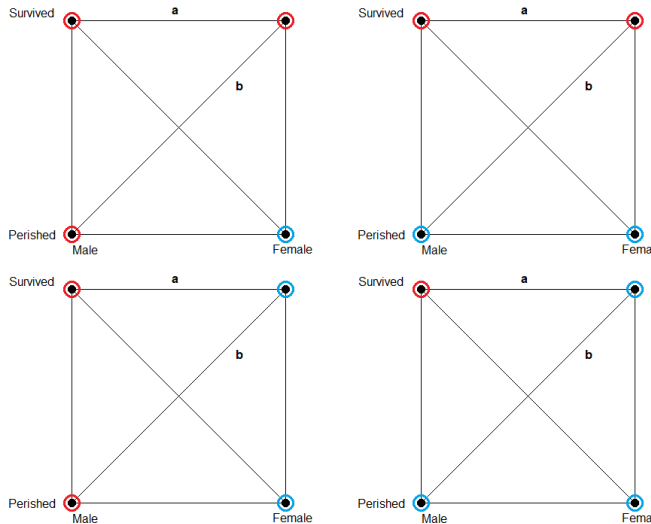
remaining two points. I will add them to the closest centroid with distance  $a$ . The outcome of the clustering can be seen in Figure 2.7.

The first case was without any problem. Now let us look at the second case, where the distance between the centroids is  $b$ , which means that there are two possible selections of the centroids. These selections can be seen in Figure 2.8. The clusters are colored the same way as in the first example; red highlights the first cluster, and blue highlights the second one.



**Figure 2.8:** Both possible selections of centroids for the second case of clustering on the example dataset.

Let us focus on the selection on the left since the selection on the right would be analog. Unlike in the first case, where we had only one way of clustering the remaining two items, we now have four possibilities since both remaining items have the same distance from both of the centroids. Figure 2.9 shows these four possible ways to cluster the remaining items.



**Figure 2.9:** All four possible solutions for the second case of clustering on the example dataset.

This outcome is the chief issue when clustering nominal data. All nominal data of any dimension are ultimately vertices of  $n$ -dimensional polygons, and they suffer from the same issue described in the second case. The difference being that instead of two different distance sizes, there are more of them.

This increase in the number of sizes creates ambivalence in the clustering that can not be resolved without a broader understanding of the data. In many cases, though, the broader understanding is not enough to determine the necessary differences to overcome this ambivalence.

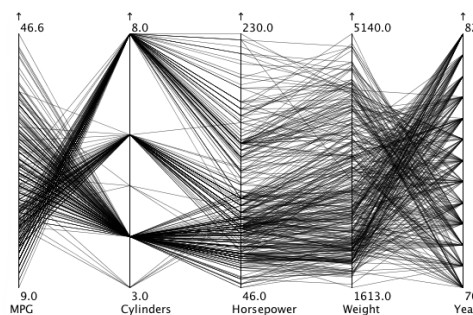
## 2.5 Visualization of Tabular Data

There exist specific methods tailored for the visualization of tabular datasets. I will discuss five of these methods and how they fit my visualization task. They are parallel coordinates in chapter 2.5.1, scatterplot matrix in chapter 2.5.2, mosaic plot in chapter 2.5.3, dimension reduction methods in general in chapter 2.5.4 and parallel sets in chapter 2.5.5.

### 2.5.1 Parallel Coordinates

Parallel coordinates is a visualization method for multivariate data and datasets with high dimensions. First, all the axis (each axis representing a single attribute of the datasets) are placed parallel. The orientation of the axis does not matter, but they are generally placed either vertically or horizontally. Next, a line joining through all of the attribute axes represents each item in the dataset. Finally, an intersection of a line and axis corresponds to a single cell in the tabular dataset, in other words, it is an intersection of a column, and a row [HWne].

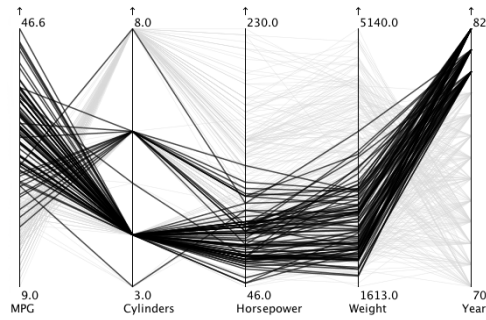
In Figure 2.10 we can see an example of the parallel coordinates method on a dataset of cars from 1970 to 1982.



**Figure 2.10:** Example of parallel coordinates method [Kosne].

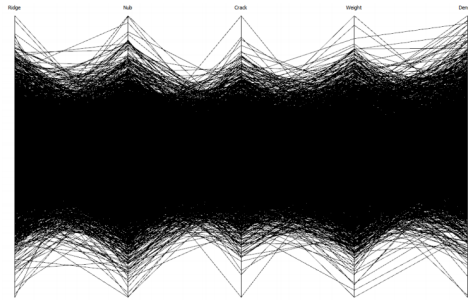
The parallel coordinates visualization generally offers the user the ability to change it dynamically. Mainly the user can reorganize the position of all of the axis. Secondly, parallel coordinates allow for brushing or highlighting of a selected item or a subset of items. An example of brushing can be seen in Figure 2.11.

In this way, a high-dimensional space can be transformed into a two-dimensional one. This transformation is not without a trade-off since we can compare only one attribute pair at a time. Another limitation of the method is the number of shown items. If this number becomes too large, the



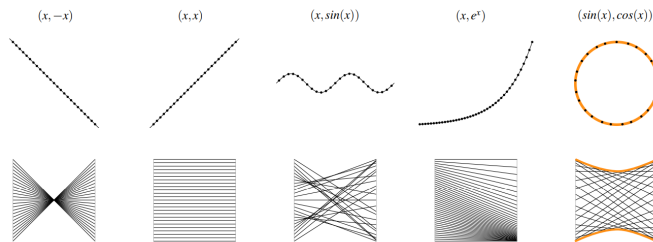
**Figure 2.11:** Example of brushing on parallel coordinates method [Kosne].

resolution can not display the lines as separate, and it becomes impossible to discern anything from the graph. In Figure 2.12 you can see this problem on a dataset of geometric features of pollen grains. In this case, brushing becomes essential to understanding the visualization.



**Figure 2.12:** Example of a cluttered visualization using parallel coordinates method [HWne].

Furthermore, we can use parallel coordinates to determine the relationship between two neighboring axes. In Figure 2.13 you can see common patterns that can be seen when using the parallel coordinates method.

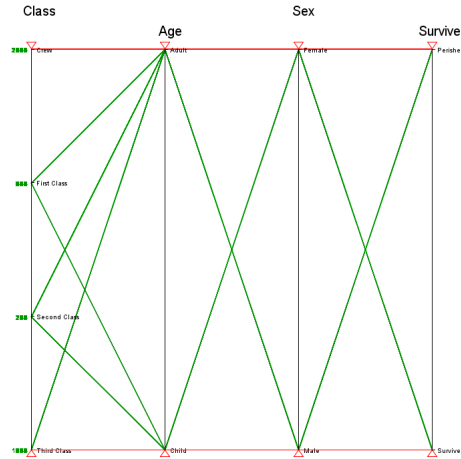


**Figure 2.13:** Common patterns in Cartesian coordinates (top) and their dual representation in parallel coordinates (bottom) [HWne].

The features mentioned above of the parallel coordinates method make it a helpful tool for visualizing tabular datasets. However, issues arise when all or most of the attributes of the dataset are nominal or ordinal. As can be seen in Figure 2.14 the parallel coordinates method becomes useless when faced with



only nominal and ordinal data. As mentioned in chapter 2.4 Particularities of Nominal Data there is a fixed small amount of unique items present in any given dataset of purely nominal and ordinal attributes. As such, the parallel coordinates method visualizes all of these items into a small number of lines. Ultimately, this gives no answers to the data structure and the trends present in the data.



**Figure 2.14:** Parallel coordinates method used on Titanic dataset with only nominal and ordinal attributes.

The conclusion is evident. For visualization of mainly nominal and ordinal data, the parallel coordinates method can not be used.

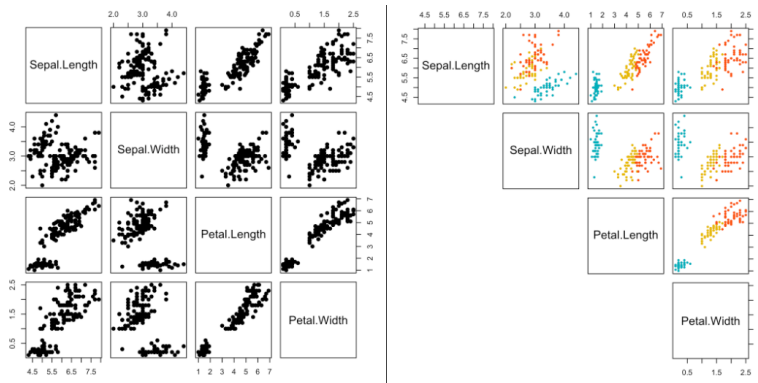
## 2.5.2 Scatterplot Matrix

Scatterplot matrix is another visualization method for visualizing multi-dimensional tabular data. Similar to the parallel coordinates method, the scatterplot matrix also provides a comparison of pairs of attributes. As can be seen in Figure 2.15 where we can see the scatterplot matrix for the iris dataset, we assign every attribute to both x and y axis. Intersections of these attributes are then 2D plots showing the relation between the two selected attributes [EDF08].

These intersection plots allow us to see correlations and clusters for the visualized pairs, but spotting correlations and clusters across multiple attributes becomes more complicated.

As can be seen left in Figure 2.15 it is not necessary to show both upper and lower parts of the scatterplot matrix since they are symmetric. This symmetry can be seen in the right image in Figure 2.15 where only the upper part of the scatterplot matrix is visible. We can further enhance the quality of the visualization by using color, in this case, used to differentiate the species of iris.

Just as parallel coordinates, a scatterplot matrix can also utilize brushing to highlight interesting parts of the plots. Another issue that both the parallel coordinates and scatterplot matrix share is the number of attributes that



**Figure 2.15:** Complete scatterplot matrix without color (left) upper panel of colored scatterplot matrix based on iris species (right) for the iris dataset [STHne].

can be visualized. Since the dataset can have many attributes, it becomes impossible to show all of them in a scatterplot matrix. The main limiting factor again is the final resolution of the image and the need for the user to read the data from the final visualization. As such, it often needs to limit the number of visualized attributes.

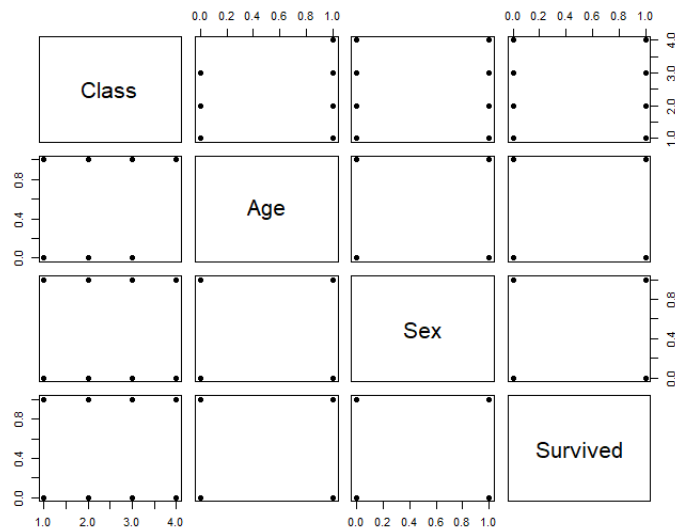
In Figure 2.16 we can see another scatterplot matrix. This time it was done over the Titanic dataset composed of only nominal and ordinal attributes. Just as it was with the parallel coordinates method, we are again faced with a limited number of unique items present in any purely nominal and ordinal dataset. In this case, we can see that almost all available positions are filled with a dot signaling that an item with these values is present in the dataset. This occurrence is not surprising since, for a dataset with 2201 items, there can be only 32 unique items. Only one dot is missing from the plots of "Class" and "Age", meaning in this case, there were no children present in the crew of Titanic. No other useful information can be extracted from this scatterplot matrix.

It is again evident that the scatterplot matrix can not visualize nominal and ordinal tabular datasets for our visualization tasks.

### ■ 2.5.3 Mosaic Plot

The mosaic plot method is often used visualization method for nominal data. We divide the area of the plot into nested rectangles, and we are switching the orientation of the cutting plane between vertical and horizontal. Figure 2.17 shows this method applied to the nominal and ordinal dataset of Titanic. The sizes of each rectangle are proportional to the number of items that fit the criteria [The12].

As a first method designed for nominal data, we can see a significant improvement from the previously discussed methods. However, this method also comes with problems. Mainly it is a fact that we can only communicate the hierarchical relations between the attributes. As a result, we lose some of



**Figure 2.16:** Parallel coordinates method used on Titanic dataset with only nominal and ordinal attributes.

the information about the relationships between the attributes of interest.

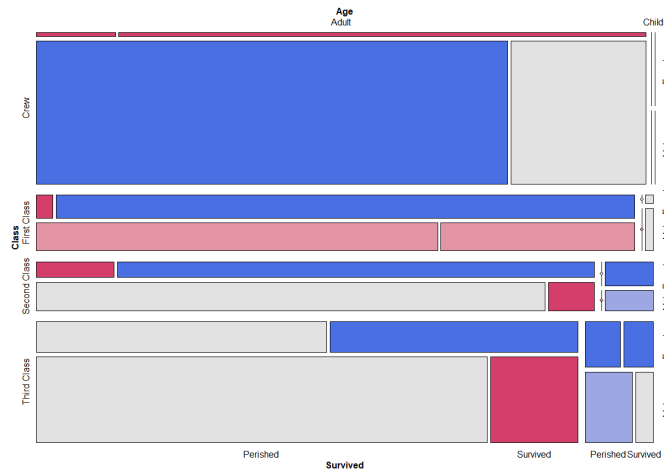
Even though the mosaic plot is so far best suited for the visualization of nominal data, it does not suit my needs regarding the visualization task specified in chapter 2.3 Visualization tasks. Mainly in regards to the trends, we would like to see in the final visualization.

## 2.5.4 Dimension Reduction Methods

Even though there is a multitude of dimension reduction methods such as PCA, Isomap, and MDS, all of these methods rely in their computation heavily on distance between items [OCAD11]. Because I am focused on nominal and ordinal data, it is not easy to find a distance measure that would make sense in a 2D or 3D plot. By assigning any nominal or ordinal attribute a fixed position in a 2D or 3D plot, we inevitably assign information to these items that they did not have. Though this problem might not affect the calculations themselves, it would be vital to the final visualization.

Same as other mentioned methods, dimension reduction methods also have issues with the low number of unique items. The dimension reduction would have to apply only to these unique items.

Taking all of these issues together, it is clear that dimension reduction methods would require us to make additional assumptions about the data and compute their distances. As was shown in chapter 2.4 Particularities of Nominal Data distances between the unique items in any nominal and ordinal datasets are very similar. From the perspective of visualization, they introduce bias as to the position and distance of these items.

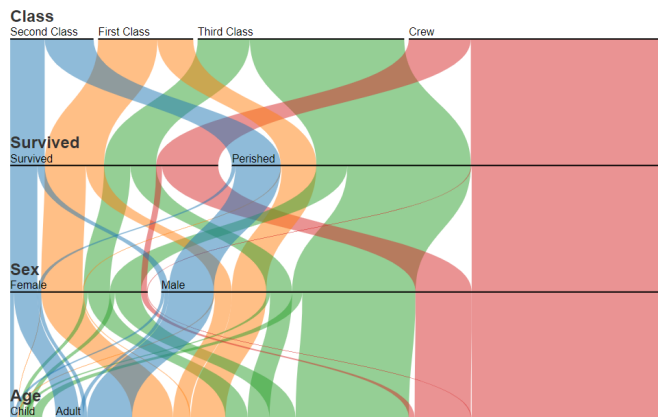


**Figure 2.17:** Mosaic plot method used on Titanic dataset with only nominal and ordinal attributes.

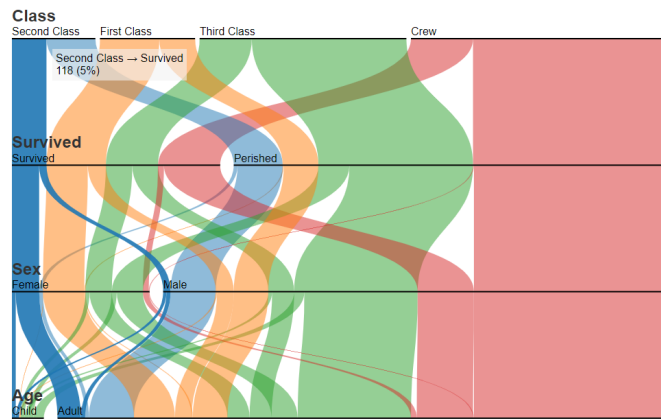
### 2.5.5 Parallel Sets

The parallel sets method, as the name suggests, shares a basis with the parallel coordinates method. Same as parallel coordinates, parallel sets arrange each attribute as an axis. These parallel axes are generally either vertical or horizontal. Unlike parallel coordinates, which map each item onto these axes, parallel sets map using parallelograms [KBH06]. Let us have attribute  $X_1$  and its category  $x_{1,1}$  and attribute  $X_2$  and its category  $x_{2,1}$ . Attributes  $X_1$  and  $X_2$  are mapped on neighboring axis in parallel sets method. A parallelogram between categories  $x_{1,1}$  and  $x_{2,1}$  is rhomboid. The width of the rhomboid corresponds to number of items, that have value of attribute  $X_1$  as  $x_{1,1}$  and value of attribute  $X_2$  as  $x_{2,1}$ .

Figure 2.18 we can see the parallel sets method. Curves are used instead of rhomboids to visualize the parallelograms. Just as with the parallel coordinates method, brushing can be used to highlight certain features of the plot (see in Figure 2.19).



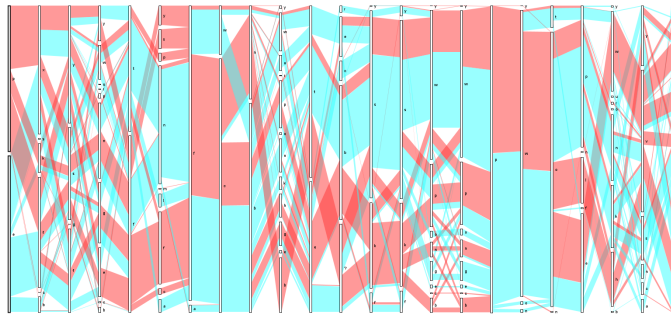
**Figure 2.18:** Parallel sets method used on Titanic dataset [Davne].



**Figure 2.19:** Parallel sets method used on Titanic dataset using brushing [Davne].

The parallel sets method combines approaches from both the parallel coordinates method and mosaic plot method and is the visualization method that best suits my selected visualization tasks. Furthermore, it brings the ability of parallel coordinates to showcase relations between the attributes and mosaic plot ability to portray nominal data. As such, I have selected parallel sets as the visualization method for my thesis.

We should not omit, that parallel sets also inherits similar issue as parallel coordinates. The main concern is the number of attributes that can be visualized and the number of attributes that can be visualized meaningfully (see Figure 2.20). An increasing number of attributes can cause the parallel sets plot to fragment into an increasing number of parallelograms, making the plot's end ineligible for the user.



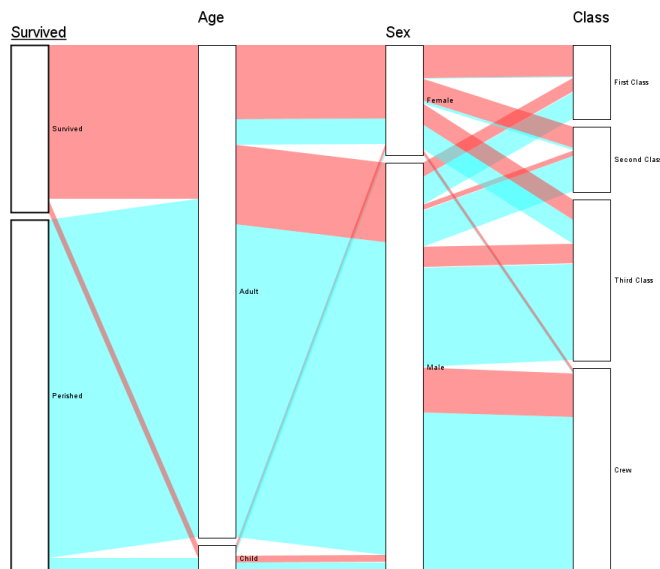
**Figure 2.20:** Parallel sets method used on a dataset of mushroom characteristics.

The parallel sets method offers two distinct layouts. Both of these layouts have distinct desirable and undesirable features.

### ■ Bundle Layout

As can be seen in Figure 2.21 bundle layout seems more compact and less prone to fragmentation. The reason for this is that each pair of axis creates its parallelograms without knowing the parallelograms before it. As shown, this creates a more compact design, but we lose the information about the

specificities of items present in the dataset.



**Figure 2.21:** Parallel sets method used on Titanic dataset using the bundle layout.

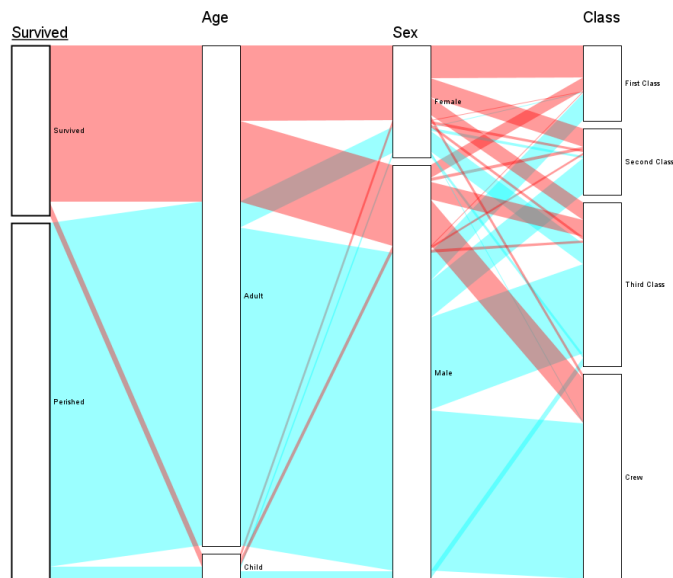
## ■ Tree Layout

Figure 2.22 shows tree layout. We can see that specifically in the last pair of axes between "Sex" and "Class" the fragmentation of the parallelograms is increased in comparison to the bundle layout in Figure 2.21. This trend of increasing fragmentation in the direction of the end of the plot is typical for tree layout.

Unlike bundle layout, tree layout offers extra information. Please note the red parallelogram in Figure 2.21 between axis "Age" and "Sex" between categories "Child" and "Male" (the parallelogram in question is the bottom-most thin red line). This parallelogram tells us that indeed there were male children (that survived) the Titanic. However, this information is not carried over to the next pair of attributes, "Sex" and "Class", so we have no way of knowing from which class were the surviving boys. If we now compare this with tree layout in Figure 2.22, we can see the same parallelogram as in Figure 2.21 is present, although now it is in steep inclination upward, but still the last red parallelogram in the plot.

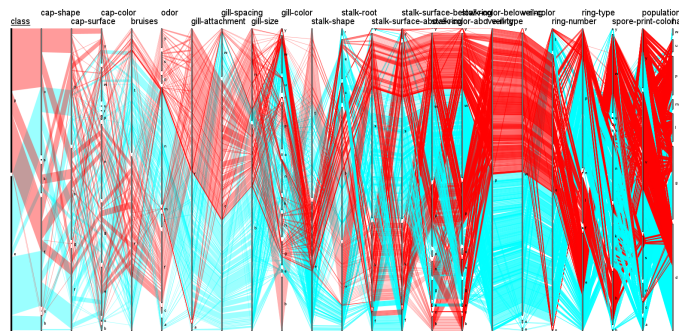
Furthermore, we can see that the parallelogram splits into three smaller parallelograms between the next pair of axes, "Sex" and "Class". This parallelogram tells us that the boys that survived were from "first class", "second class", and "third class". Furthermore, we see that the least number of boys were from "first class", the second least from "second class" and most of them from "third class".

Even though it may seem that tree layout offers more information and is



**Figure 2.22:** Parallel sets method used on Titanic dataset using the tree layout.

better than bundle layout, it does not have to be so. Tree layout becomes useless with an increasing number of attributes since the end of the plot becomes more akin to parallel coordinates in appearance and thus utterly unreadable. Figure 2.23 shows how tree layout can become unreadable. Compare Figure 2.23 with Figure 2.20. Both of these plots used the mushroom dataset but used different layouts.



**Figure 2.23:** Parallel sets method used on a dataset of mushroom characteristics using the tree layout.

## 2.6 Clustering of Tabular Data

In this chapter, I will discuss the clustering methods that I have tried, the distance measure for the nominal and ordinal data, and a measure for comparing the quality of clusterings.

In the chapter 2.6.1 Dice Measure I will introduce the similarity measure called Dice measure. I will continue with this topic in chapter 2.6.2 Adapted

Priority Dice Measure where I will introduce an altered version of the Dice measure that is more suitable for clustering of nominal data. Next I showcase the silhouette for measuring of quality of clustering in chapter 2.6.3 Silhouette. These three chapters are followed by four additional chapters that discuss the various clustering methods that I have tested.

Firstly its hierarchical clustering in chapter 2.6.4 Hierarchical Clustering, followed by brief discussion on the topic of data transformation in chapter 2.6.5 Transforming Nominal Data to Quantitative Data. I give significant amount of space to the explanation of the k-modes method in chapter 2.6.6 K-Modes and the four subsequent variations in which I have used the k-modes method in chapters 2.6.6 Random Method, 2.6.6 Column Method, 2.6.6 Occurrence Method and 2.6.6 Two-Occurrence Method.

The final chapter 2.6.7 Parallelogram Clustering, concerns the method that I have selected for implementation - parallelogram method.

### 2.6.1 Dice Measure

The first problem when clustering is to find an appropriate measure of the distance between nominal attributes. A good measure of distance is similarity and dissimilarity measure like the Dice measure [XLQ12]. This measure can further be adopted as the Gower measure, which can be used for mixed data [Gow71].

Let us have a data point  $i$  and  $j$  and binary attribute  $x$ . Next, let us define three variables  $a(i, j), b(i, j), c(i, j)$  -  $a(i, j)$  is a measure of similarity between two data points, and  $b(i, j)$  and  $c(i, j)$  are measures of dissimilarity between two data points.

- If the value of  $x$  is equal to 1 for both data points  $i$  and  $j$ , then increment  $a(i, j)$ .
- If  $x$  is equal to 1 for  $i$  and equal to 0 for  $j$ , then increment variable  $b(i, j)$ .
- If  $x$  is equal to 0 for  $i$  and equal to 1 for  $j$ , then increment variable  $c(i, j)$ .

Count the distance between data points  $i$  and  $j$  as:

$$d(i, j) = 1 - \frac{2a(i, j)}{2a(i, j) + b(i, j) + c(i, j)}. \quad (2.2)$$

And it holds that  $d(i, j) = d(j, i)$ , because switching the order of  $i$  and  $j$  has no effect on the value of  $a$  and will result in swap of values between  $b$  and  $c$ .

We can summarize, that  $d(i, j) \in [0, 1]$ .  $d(i, j) = 0$  if the data points  $i$  and  $j$  are exactly the same, and  $d(i, j) = 1$  if the data points  $i$  and  $j$  are dissimilar in every attribute.

Now that we know how to count the Dice measure, I will demonstrate it no an example. Let us first transform the data subset of the Titanic dataset from Table 2.1 to show unique items. These unique items are shown in Table 2.3



Item ID	Sex	Survived	Count
1	Female	Survived	2
2	Female	Perished	1
3	Male	Survived	3
4	Male	Perished	9

**Table 2.3:** Representative subset of the Titanic dataset with only unique data points.

with an added column for the actual count of each unique data point/item from the original dataset.

Now, we need to transform these values into binary values. This transformation requires that every attribute be divided into as many binary attributes as there are categories for this attribute. Let us demonstrate this on Table 2.3. The transformed dataset with binary attributes is shown in Table 2.4 where the binary attributes are denoted by the category they represent.

Item ID	Female	Male	Survived	Perished
1	1	0	1	0
2	1	0	0	1
3	0	1	1	0
4	0	1	0	1

**Table 2.4:** Representative subset of the Titanic dataset with only unique data points transformed into dataset with only binary attributes.

We can count the distance measures between the data points in Table 2.3. The counted distances are shown in Table 2.5. As was mentioned in chapter 2.4 Particularities of Nominal Data the distances between nominal items are fixed and from a small set of values. As mentioned, this creates problems when clustering the nominal data, and thus, we need to modify the Dice measure to prevent or limit this problem.

Item ID	1	2	3	4
1	0	$\frac{1}{3}$	$\frac{1}{3}$	1
2	$\frac{1}{3}$	0	1	$\frac{1}{3}$
3	$\frac{1}{3}$	1	0	$\frac{1}{3}$
4	1	$\frac{1}{3}$	$\frac{1}{3}$	0

**Table 2.5:** Distances between unique data points of the representative subset of the Titanic dataset.

## 2.6.2 Adapted Priority Dice Measure

As shown in the chapters 2.6.1 Dice Measure and 2.4 Particularities of Nominal Data, we need a solution for our distance measure so that we can eliminate

possible clusterings to a minimum. Let us first modify the Dice measure to suit our needs better.

As show in chapter 2.6.1 Dice Measure Dice measure is composed of three values  $a(i, j), b(i, j), c(i, j)$ . Since we do not care for the order of comparison between the items, in order words the items themselves do not have a priority or inherent hierarchy, it is unnecessary to differentiate between  $b(i, j)$  and  $c(i, j)$ . For this purpose I will merge them into  $b(i, j)$  and it now holds:

- If the value of  $x$  is equal to 1 for both data points  $i$  and  $j$ , then increment  $a(i, j)$ .
- If  $x$  has different value for  $i$  than for  $j$ , then increment variable  $b(i, j)$ .

We can than augment the formula for the Dice measure to:

$$d(i, j) = 1 - \frac{2a(i, j)}{2a(i, j) + b(i, j)}. \quad (2.3)$$

And it still holds that  $d(i, j) = d(j, i)$ .

Now let us focus on the problem at hand - to create a greater diversity between the distances between nominal data. To achieve this goal, I introduce a priority parameter that augments the Dice measure.

Let us have data points  $i$  and  $j$  and binary attribute  $x$ . Let us have a vector of priorities  $\vec{p}$ , where  $|\vec{p}|$  is equal to the number of original attributes, and the values of  $\vec{p}$  are from  $[1, \infty]$ . Let us have  $\vec{a}$  as a vector of similarity and  $\vec{b}$  as a vector of dissimilarity. It holds that  $|\vec{p}| = |\vec{a}| = |\vec{b}|$ . I define  $\vec{a}$  and  $\vec{b}$  as:

$$\vec{a}_k = \begin{cases} 1 & \text{if data points } i \text{ and } j \text{ have **same** value for attribute } x, \\ 0 & \text{if data points } i \text{ and } j \text{ have **different** value for attribute } x. \end{cases} \quad (2.4)$$

$$\vec{b}_k = \begin{cases} 1 & \text{if data points } i \text{ and } j \text{ have **different** value for attribute } x, \\ 0 & \text{if data points } i \text{ and } j \text{ have **same** value for attribute } x. \end{cases} \quad (2.5)$$

Vectors  $\vec{a}$  and  $\vec{b}$  are meant to replace  $a(i, j)$  and  $b(i, j)$ .  $\vec{a}$  instead of being a sum of all matches between items  $i$  and  $j$  now holds information about where these matches occurred. Similarly  $\vec{b}$  holds information about where mismatches occurred instead of being a sum of these mismatches.

We can than define distance  $d(i, j)$  as:

$$d(i, j) = 1 - \frac{2(\vec{a} \times \vec{p}^T)}{2(\vec{a} \times \vec{p}^T) + (\vec{b} \times \vec{p}^T)}. \quad (2.6)$$

Where  $\vec{p}^T$  is transposed vector  $\vec{p}$ . Vectors  $\vec{p}$  will assure that if two items are similar in a high priority attribute, they will be pulled close together. It will also ensure that if two items are dissimilar in a high priority attribute,

they will be pulled apart. This priority vector ensures that the distances will get skewed.

Let us demonstrate the outcome of priorities on the data from Table 2.3.

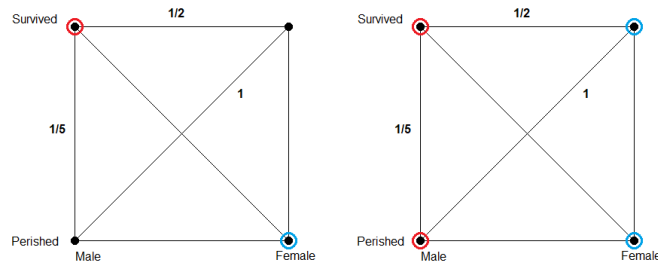
We will use  $\vec{p} = (2, 1)$  and subsequently  $\vec{r} = (1, 2)$ . The counted prioritized distance are shown in Table 2.6. To demonstrate the vectors  $\vec{a}$  and  $\vec{b}$  I will show how they would look like for items 1 and 2 from Table 2.4.  $\vec{a} = (1, 0)$  and  $\vec{b} = 0, 1$ .

Item ID	1	2	3	4
1	0	$\frac{1}{5}$	$\frac{1}{2}$	1
2	$\frac{1}{5}$	0	1	$\frac{1}{2}$
3	$\frac{1}{2}$	1	0	$\frac{1}{5}$
4	1	$\frac{1}{2}$	$\frac{1}{5}$	0

**Table 2.6:** Prioritized distances between unique data points of the representative subset of the Titanic dataset.

We can see that since I increased the priority of attribute "Sex" to 2, then if the two items have the same gender, they are closer together (meaning their distance measure is smaller). On the other hand, if the items agree in attribute "Survived" but disagree in attribute "Sex", they are further apart.

The visualization of the prioritized distances is shown in Figure 2.24 (left plot), where I have marked in red a centroid of the first cluster and in blue centroid of the second cluster in the left plot. In the right plot, we see the outcome of clustering for the new distances. In this case, depending on the choice of centroids, there is only one way to attach the remaining two items to the two clusters.



**Figure 2.24:** Visualization of prioritized distances for the representative subset of the Titanic dataset. Selection of centroids (left), outcome of clustering (right).

I will note that although both plots are shown on a square, it is not to scale. With the distance values for side  $\frac{1}{5}$  and  $\frac{1}{2}$  and the diagonal having the value of 1 it is impossible to portray such geometry. This geometry failure is caused by the fact that the Dice measure is not a measure of distance but rather a similarity measure. I would also like to add that subsequently, any such visualization is on its basis incorrect. Here it is only presented in regards to the simplification of clustering.

As shown in the Figure 2.24 (left plot) I have selected the same case as in chapter 2.4 Particularities of Nominal Data. Instead of four possible ways to

cluster the remaining items, the prioritized Dice measure allowed us to shrink the possibilities to 1. With a high-dimension dataset, it will become important to specify the priority vector as best as possible. The more distinct the values will be, the better. This distinction is, of course, a complex problem. It will require the user to understand the data and have a clear goal for what the visualization is meant to show.

### 2.6.3 Silhouette

One measure of discerning the correctness of clustering is to use silhouette. For every data point/item, silhouette measures its similarity to its assigned cluster, and its dissimilarity to all other clusters [AT07].

Let us assume we have clustered the dataset into  $k$  clusters, and we have a set of assigned data points/items for every cluster  $C_i, i \in \mathbb{N}, i \in [0, \dots, k]$ .

Then for every data point  $j \in C_i$  and every cluster, we count:

$$a(j) = \frac{1}{|C_i| - 1} \sum_{l \in C_i} d(j, l). \quad (2.7)$$

Where  $a(j)$  is a measure of dissimilarity with all other data points in the same cluster,  $d(j, l)$  is a distance between  $j$  and any other element from  $C_i$ . This structure means we ideally want to minimize  $a(j)$  to increase similarity inside the cluster.

Then for every data point  $j \in C_i$  and every cluster, we count:

$$b(j) = \min_{i \neq k} \frac{1}{|C_k|} \sum_{l \in C_k, k \neq i} d(j, l). \quad (2.8)$$

Where  $b(j)$  is minimal average to dissimilarity to all other clusters,  $d(j, l)$  is a distance between  $j$  and all elements from another cluster  $C_k$ . This structure means we want to maximize  $b(j)$  to maximize the minimal difference between all other clusters [AT07].

Now we can define silhouette for every data point  $j$  of the dataset as:

$$s(j) = \begin{cases} 1 - a(j)/b(j) & \text{if } a(j) \leq b(j), \\ 0 & \text{if } a(j) = b(j), \\ b(j)/(a(j) - 1) & \text{if } a(j) \geq b(j). \end{cases} \quad (2.9)$$

Based on this equation, we strive for maximal possible  $s(j)$  for every data point, which will ensure that clusters are clearly defined.

To count the clustering's overall silhouette, we count the mean of all silhouettes of all data points.

### 2.6.4 Hierarchical Clustering

The first attempt I have made at clustering the data was hierarchical clustering. The idea was simple - creating a hierarchy of nominal data will allow me to cut the tree at specific levels, which will produce the desired clusters.

■ **Input:**

- Dataset  $D$   $k$ -dimensional (having  $k \in \mathbb{N}$  nominal or ordinal attributes) containing  $n \in \mathbb{N}$  items.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $p_e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- $X_j$ ,  $j \in \mathbb{N}$ ,  $j \in [1, \dots, k]$  is one attribute of  $D$ ,  $|X_j|$  is number of categories of the given attribute.
- List  $T$  of single node trees containing  $n$  trees. Each node  $t_i$ ,  $i \in \mathbb{N}$ ,  $i \in 1, \dots, n$  represents one item of dataset  $D$ . Each  $t_i$  has two children (initial they are empty, making each node a leaf) and a vector  $\vec{b}_i$  a binary representation of the given item.  $|\vec{b}_i| = \prod_{j=1}^k |X_j|$  (as shown in Table 2.4).
- Matrix  $S$  of size  $n \times n$  of distances between all items (Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

■ **Output:** List  $T$  containing one binary tree containing all the items of the dataset as leaf.

■ **Algorithm:**

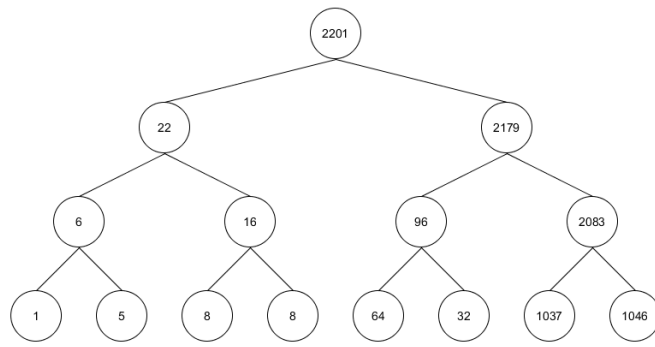
1. Find the lowest value from  $S$  and the two associated trees  $t_x$  and  $t_y$  from  $T$ .
2. Create new tree  $t_z$  and set item  $t_x$  as its left child and item  $t_y$  as its right child.
3. Set  $\vec{b}_z$  of  $t_z$  to be an average value of  $\vec{b}_x$  of  $t_x$  and  $\vec{b}_y$  of  $t_y$ .

$$\vec{b}_{za} = \begin{cases} \text{if } b_{xa} = b_{ya} \text{ then } b_{za} = b_{xa} = b_{ya}, \\ \text{if } b_{xa} \neq b_{ya} \text{ then select either } b_{xa} \text{ or } b_{ya} \text{ based} \\ \text{on the total number of occurrences of disputed categories} \\ \text{in the dataset, selecting the one with highest number} \\ \text{of occurrences.} \end{cases} \quad (2.10)$$

4. Remove  $t_x$  and  $t_y$  from  $T$  and add  $t_z$  to  $T$ .
5. Remove rows and columns corresponding to  $t_x$  and  $t_y$  from  $S$  and add new row and column for  $t_z$  to  $S$  and count all the distances using Dice measure.
6. Check if  $|T| = 1$ . If yes, return  $T$  else return to step 1 [Mad12].

Using this algorithm leads to hugely unbalanced results, which means that left subtree and right subtree, on any given level, will have a hugely different number of leaves (or items in other words) associated with them. Figure 2.25 shows the first four levels of this hierarchy. Each node contains some of its leaves. We can see that the tree is heavily tilted to the right.

This approach to clustering nominal and ordinal data proved futile, as it was impossible to glean any information from the resulting clusters.



**Figure 2.25:** Visualization of the first four levels of hierarchy created from the Titanic dataset.

### 2.6.5 Transforming Nominal Data to Quantitative Data

Since I am focused on clustering mainly nominal and ordinal data, methods for transforming nominal attributes to quantitative attributes proved difficult. These methods find a quantitative attribute of the highest variance and map it on the nominal attributes. In this way, they introduce dependence on the quantitative attribute not initially present in the data.

Even though the resulting transformation allows us to work with the transformed attributes as with quantitative attributes, its negatives outweigh them. Most importantly, they need a quantitative attribute. In purely nominal and ordinal datasets, such transformation is impossible. Secondly, the dependency on the quantitative attribute as a framework to build the transformation introduces inaccuracies into the dataset. These inaccuracies confuse the results [SJJ10].

As such, I have decided not to use these transformation methods.

### 2.6.6 K-Modes

K-Modes method is an alteration to the k-means method meant for use on nominal data. Let us first discuss the k-means method and then the difference from the k-modes method. The algorithm selects  $k$  items called centroids, and all other items are attached to these centroids based on their distance. We always attach an item to the closest centroid. Once all elements are attached, we count new centroids as an average item of all items currently attached to the centroid. We then reattach all of the items again. We repeat this procedure until there is no change in item membership to the clusters.

**K-means** algorithm is as follows:

■ **Input:**

- Dataset  $D$  containing  $n \in \mathbb{N}$ , items.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $k \in \mathbb{N}$ ,  $k > 1$  determining number of clusters.



- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $k \in \mathbb{N}$ ,  $k > 1$  determining number of clusters.
- Array  $c$ ,  $|c| = n$ ,  $c_i = 0$ ,  $i \in \mathbb{N}$ ,  $i \in [1, \dots, n]$  for marking items to clusters.
- Matrix  $S$  of size  $n \times n$  of distances between all items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

■ **Output:** Marked array  $c$ .

■ **Algorithm:**

1. Select randomly  $k$  items from  $D$  to use as modes and mark them in array  $c$  with different numbers from  $[1, \dots, k]$ .
2. Attach all items to the modes by marking them in the array  $c$  based on their distance from the centroids. Attach item to the closest mode.
3. If the array  $c$  was not changed during step 2 return array  $c$ , end algorithm. Else go to step 4.
4. For each cluster count a new mode as an average item of all items belonging to the centroid. Go to step 2 [Zhe98], [ZSX06].

I have used the k-modes algorithm in four different ways trying for the best possible approach to clustering the dataset. I will now discuss all four of these methods.

In every method discussed, I have used the Titanic dataset to demonstrate the results. In Figure 2.26, Figure 2.27, Figure 2.28 and Figure 2.29 the left plots portray the original dataset without clusters, the right plots show the dataset clustered (notice the left-most axis on all the right plots called "cluster"). Also, for all the figures, the top plots are unordered in regards to their categories. The bottom plots' categories are ordered. Mainly it is the attribute "Class" which is ordinal. They remain three attributes "Age", "Sex" and "Survived" are ordered in a way to make the parallel sets plot as readable as possible.

All of the following methods also used for the demonstration purposes the priority vector  $\vec{p} = (3, 1, 1, 2)$  for the attributes "Class", "Age", "Sex" and "Survived" respectively. This priority vector was used for counting the priority Dice measure as shown in chapter 2.6.2 Adapted Priority Dice Measure.

## ■ Random Method

The first method of random clustering aims at taking as much work away from the user as possible. It attempts all possible clusterings using a different number of modes and using the silhouette to determine the best possible clustering.



**Random method** algorithm is as follows:

■ **Input:**

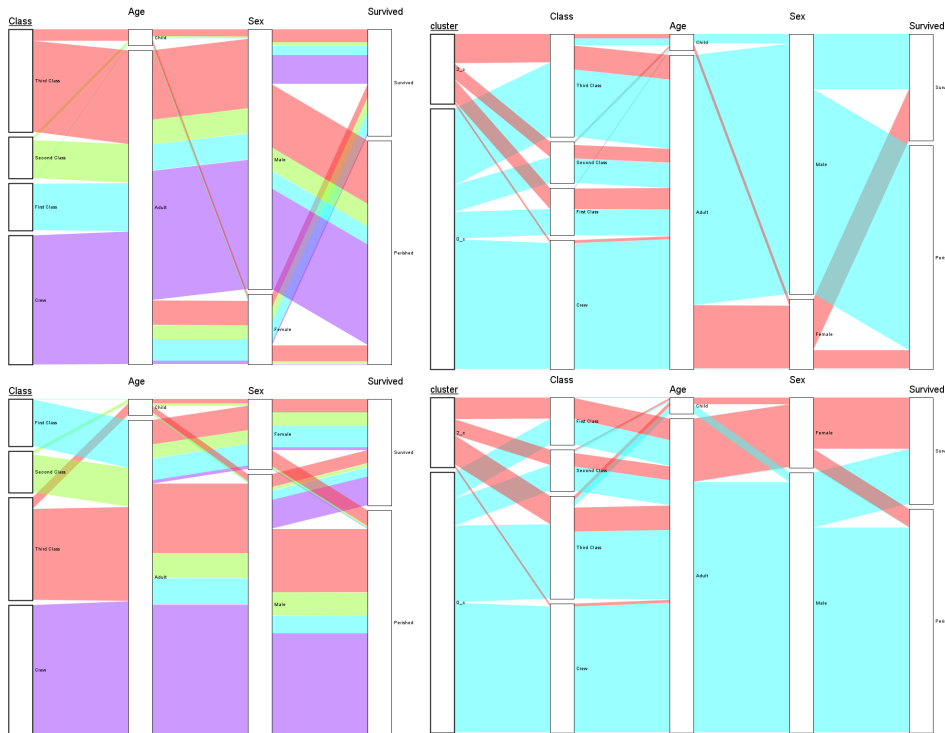
- Dataset  $D$  containing  $n \in \mathbb{N}$  items,  $X_i$  is  $j$ -th attribute,  $|X_i|$  is number of categories of attribute  $X_i$ , there is  $k \in \mathbb{N}$  nominal and ordinal attributes.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $p_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $maxModes \in \mathbb{N}$  determining maximum number of modes that could be used,  $maxModes < \frac{\sqrt{n}}{2}$ .
- Number  $iter \in \mathbb{N}$  determining maximum number of iterations per number of modes,  $iter > 0$ .
- List of arrays  $C$ ,  $|C| = maxModes$ ,  $c_i \in C$ ,  $i \in \mathbb{N}$ ,  $i \in [1, \dots, k]$ ,  $c_{ij} = 0$ ,  $j \in [1, \dots, n]$  for marking items to clusters. One such array per number of modes.
- Array  $Z$ ,  $|Z| = maxModes$ ,  $z_r \in \mathbb{R}$ ,  $r \in \mathbb{N}$ ,  $r \in [1, \dots, maxModes]$  for storing maximum achieved silhouette per number of modes.
- Matrix  $S$  of size  $n \times n$  of distances between all items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

■ **Output:** Marked array  $c_i$ .

■ **Algorithm:**

1. for  $i=2$  to  $maxModes$ 
  - a. for  $j=1$  to  $iter$ 
    - (i) Run  $k$ -modes algorithm with  $k = i$  modes.
    - (ii) Count silhouette of the clustering and if it is higher than silhouette saved in  $Z_i$ , saved the new silhouette into  $Z_i$  and save the clustering into  $c_i$ .
2. Find  $max(Z_i)$  and return corresponding  $c_i$  as a result.

The final two clusters for right plots in Figure 2.26 have their modes described in Table 2.7. Even though from the description of the modes it would seem that the clusters should be different, when we look at plots on the right of Figure 2.26, we can see that the main focus is on the last two attributes, "Sex" and "Survived". Attributes "Class" and "Age" have very little to do with the clusters since we cannot discern any prevailing trend amongst them. As such, we could filter/brush the dataset using the "Sex" attribute, and we would receive the same results. As such, we can not in good conscience use the method that requires additional resources for computation if the result can be achieved easily.



**Figure 2.26:** Comparison for random clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

We should also mention that in its nature, the algorithm is random in selecting the initial modes, and should we run the algorithm multiple times (with different seeds for the random functions), we can achieve different results. However, on average, the results agree in using two clusters, and they achieve a silhouette of about 0.7, which is quite good.

Cluster ID	Class	Age	Sex	Survived
2c	Third Class	Adult	Female	Survived
0c	Crew	Adult	Male	Perished

**Table 2.7:** Modes for cluster resulting from the random method.

### Column Method

Having learned from the random method that the clusters tend to mimic the category distribution of one of the attributes, I have decided to alter the method to take advantage of this phenomenon.

The column method will use the attributes as a skeleton for its clusters instead of using all possible number of modes multiple times in an attempt to arrive at a local maximum.

**Column method** algorithm is as follows:

■ **Input:**

- Dataset  $D$  containing  $n \in \mathbb{N}$  items,  $X_j$  is  $j$ -th attribute,  $|X_j|$  is number of categories of attribute  $X_j$ , there is  $k \in \mathbb{N}$  nominal and ordinal attributes.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $maxModes \in \mathbb{N}$  determining maximum number of modes that could be used,  $maxModes < \frac{\sqrt{n}}{2}$ .
- Number  $maxModes \in \mathbb{N}$  determining maximum number of iterations per number of modes,  $iter > 0$ .
- Two arrays  $c, o$ ,  $|c| = n$ ,  $|o| = n$ ,  $c_i = 0$ ,  $o_i = 0$ ,  $i \in \mathbb{N}$ ,  $i \in [1, \dots, n]$  for marking items to clusters.
- Number  $z \in \mathbb{R}$  for storing maximum achieved silhouette.
- Matrix  $S$  of size  $n \times n$  of distances between all items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

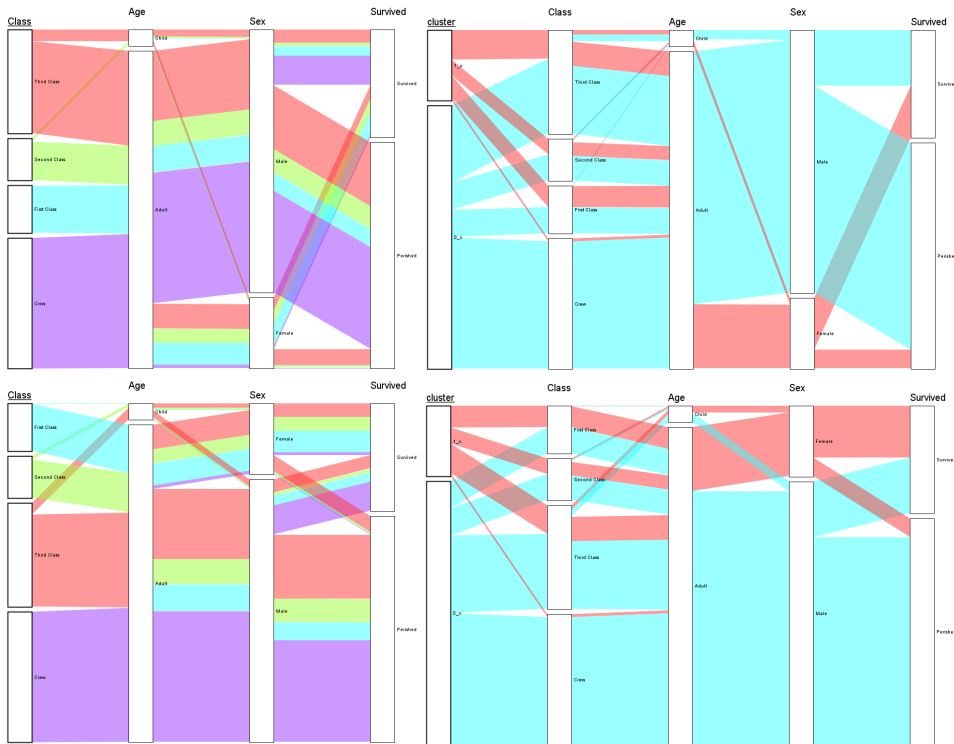
■ **Output:** Marked array  $o$ .

■ **Algorithm:**

1. for  $i=1$  to number of nominal or ordinal attributes
  - a. If number of categories of  $i$ -th attribute is smaller than  $maxModes$  continue, else return to step 1 and continue iteration.
  - b. for  $j=1$  to  $iter$ 
    - (i) Run  $k$ -modes algorithm with  $k = |X_i|$  modes in such a way, that every mode has different value for attribute  $X_i$ .
    - (ii) Count silhouette of the clustering and if it is higher than silhouette saved in  $z$ , saved the new silhouette into  $z$  and save the clustering into  $o$ .
2. Return array  $o$  as a result.

We can see the outcome of this method in Figure 2.27. Furthermore, we can notice that the outcome is identical to the outcome of Figure 2.26. This outcome further boosts the fact that the final modes are, in both cases, the same. We can see it in Tables 2.7 and 2.8. The silhouette of the clustering is, of course, the same for both of these examples.

Based on these two results, we could indeed summarize that this is the correct way to cluster this dataset. But the same points made in chapter 2.6.6 Random Method apply here. It would be unnecessary to cluster the dataset if simple brushing would have the same result.



**Figure 2.27:** Comparison for column clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

Cluster ID	Class	Age	Sex	Survived
0c	Crew	Adult	Male	Perished
1c	Third Class	Adult	Female	Survived

**Table 2.8:** Modes for cluster resulting from the random method.

### ■ Occurrence Method

The occurrence method is again made with the limitations of the column method in mind. Instead of running clustering based on all attributes, I require input from the user, which will determine which attribute will be used to create the initial modes. The second change is in the way of choosing these modes. Instead of selecting them at random, with the requirement that all the modes are different in the given attribute in the column method, I try to take a different approach.

Let us attribute  $X_i, i \in [1, \dots, \text{number of attributes}]$ ,  $|X_i|$  denotes the number of categories of attribute  $X_i$ .  $x_{ij}, j \in [1, \dots, |X_i|]$  is j-th category of attribute  $X_i$ . Let us have a list of items  $L$  which all have the value for attribute  $X_i$  set to category  $x_{ij}$ . Create new item  $m$  by setting each of its attributes  $X_i$  to the value of category  $x_{ik}$  that has highest occurrence for the

given attribute in the list  $L$ .

In this way, my selection creates a possibly nonexistent item that is the best representative of the list  $L$ .

**Occurrence method** algorithm is as follows:

■ **Input:**

- Dataset  $D$  containing  $n \in \mathbb{N}$  items,  $X_j$  is  $j$ -th attribute,  $|X_j|$  is number of categories of attribute  $X_j$ , there is  $k \in \mathbb{N}$  nominal and ordinal attributes.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $a \in \mathbb{N}$  determining the selected attribute as a basis for clustering,  $1 < a \leq k$ .
- 3D matrix  $sums$  used to save the sums of occurrences. Sizes of the array are in order  $|X_a|$ ,  $k$  and  $|X_i|$ ,  $i \in \mathbb{N}$ ,  $i \in [1, \dots, k]$ .  $sums[u][v][w] \in \mathbb{N}$ ,  $u, v, w \in \mathbb{N}$ ,  $u \in [1, \dots, |X_a|]$ ,  $v \in [1, \dots, k]$ ,  $w \in [1, \dots, |X_i|]$ .
- Array  $c$ ,  $|c| = n$ ,  $c_l = 0$ ,  $l \in \mathbb{N}$ ,  $l \in [1, \dots, n]$  for marking items to clusters.
- Matrix  $S$  of size  $n \times n$  of distances between all items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

■ **Output:** Marked array  $c$ .

■ **Algorithm:**

1. Count all sums for the selected attribute  $X_a$  and mark the occurrences of all attributes based on the attribute  $X_a$  and save to matrix  $sums$ .
2. Determine initial modes based on the sums from matrix  $sums$ , selecting for each mode the category with highest occurrence. If two or more categories have the same number of occurrences select one of them randomly.
3. Run k-modes algorithm.
4. Return marked array  $c$ .

I will demonstrate this in an example. We have selected the attribute "Class" as a basis for our clustering. The algorithm will create four modes based on the occurrences associated with the four categories of attribute "Class". The categories are "First Class", "Second Class", "Third Class" and "Crew". Outcome of step 1 can be seen in Table 2.9. Most of the categories were abbreviated.

■ Attribute "Class":

- "First Class" to "1st".

- "Second Class" to "2nd".
- "Third Class" to "3rd".
- "Crew" remained unchanged.
- Attribute "Age":
  - "Child" to "C."
  - "Adult" to "A."
- Attribute "Sex":
  - "Female" to "F."
  - "Male" to "M."
- Attribute "Survived":
  - "Survived" to "S."
  - "Perished" to "P."

We can see in Table 2.9 that there are four rows, each used to denote one mode based on the "Class" attribute. This result can be seen in the number of occurrences with the attribute "Class" as it creates a sparse diagonal matrix of size 4x4 at the beginning of the table.

Class	Class				Age		Sex		Survived	
	1st	2nd	3d	Crew	C.	A.	F.	M.	S.	P.
1st	325	0	0	0	6	319	145	180	203	122
2nd	0	285	0	0	24	261	106	179	118	167
3rd	0	0	706	0	79	627	196	510	178	528
Crew	0	0	0	885	0	885	23	862	212	673

**Table 2.9:** Sums for the occurrence method when attribute "Class" was selected as a basis for clustering.

Table 2.10 shows how to select the appropriate modes based on the occurrences shown in Table 2.9. It is a straightforward procedure where we always select the category with the highest occurrence for each of the four attributes.

Table 2.11 shows how the k-modes algorithm changed the initial modes in Table 2.10.

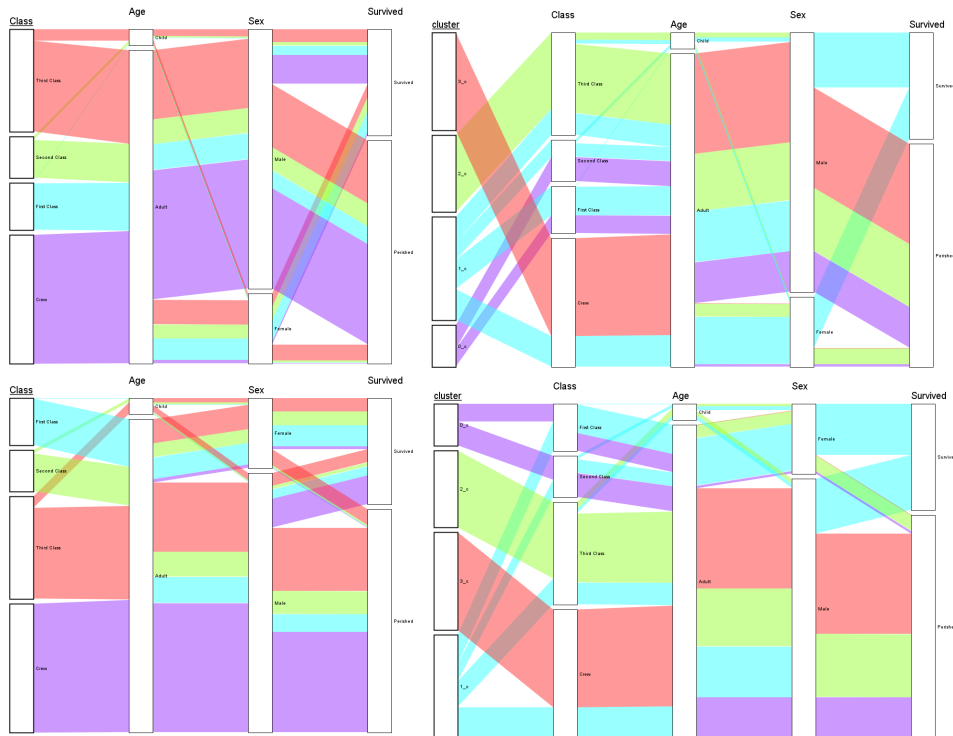
Cluster ID	Class	Age	Sex	Survived
1c	First Class	Adult	Male	Survived
0c	Second Class	Adult	Male	Perished
2c	Third Class	Adult	Male	Perished
3c	Crew	Adult	Male	Perished

**Table 2.10:** Initial modes for clusters resulting from the occurrence method.

Cluster ID	Class	Age	Sex	Survived
1c	Crew	Adult	Male	Survived
0c	Second Class	Adult	Male	Perished
2c	Third Class	Adult	Male	Perished
3c	Crew	Adult	Male	Perished

**Table 2.11:** Final modes for clusters resulting from the occurrence method.

Tables 2.10 and 2.11 are arranged as to fit the data in Table 2.9, which is the reason why their first column may seem arranged haphazardly.



**Figure 2.28:** Comparison for occurrence clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

Now let us discuss the results of this method. When we compare the final modes in Table 2.11, we can immediately see that three of the modes have the same categories for attributes "Age", "Sex" and "Survive". Those are modes with IDs "0c", "2c" and "3c". This match is the first sign that the clustering did not find diverging trends in the dataset. Another sign is that no mode has the category "Female" for the attribute "Sex". Looking at Figure 2.28, specifically the right plots, we can see that the differently colored parallelograms do not show a trend beyond the match mentioned above for three of the clusters in three attributes. As such, this outcome is not very

promising.

### ■ Two-Occurrence Method

The two occurrence method is very similar to the occurrence method. The main difference is that instead of selecting one attribute as a basis for the clusters, it selects two.

Let us have two nominal or ordinal attributes  $X_i$  and  $X_j$ , where  $|X_i|$  is the number of categories for attribute  $X_i$  and similarly so for attribute  $X_j$ . There will be  $|X_i||X_j|$  clusters as a result.

**Two-Occurrence method** algorithm is as follows:

#### ■ Input:

- Dataset  $D$  containing  $n \in \mathbb{N}$  nominal or ordinal items,  $X_j$  is  $j$ -th attribute,  $|X_j|$  is number of categories of attribute  $X_j$ , there is  $k \in \mathbb{N}$  nominal and ordinal attributes.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Numbers  $a, b \in \mathbb{N}$  determining the two selected attributes as a basis for clustering,  $1 < a \leq k$ ,  $a \neq b$ ,  $1 < b \leq k$ .
- 3D matrix *sums* used to save the sums of occurrences. Sizes of the array are in order  $|X_a||X_b|, k$  and  $|X_i|, i \in [1, \dots, k]$ .  $sums[u][v][w] \in \mathbb{N}$ ,  $u, v, w \in \mathbb{N}$ ,  $u \in [1, \dots, |X_a||X_b|]$ ,  $v \in [1, \dots, k]$ ,  $w \in [1, \dots, |X_i|]$ .
- Array  $c$ ,  $|c| = n$ ,  $c_l = 0$ ,  $l \in \mathbb{N}$ ,  $l \in [1, \dots, n]$  for marking items to clusters.
- Matrix  $S$  of size  $n \times n$  of distances between all items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

#### ■ Output: Marked array $c$ .

#### ■ Algorithm:

1. Count all sums for the two selected attributes  $X_a$  and  $X_b$  and mark the occurrences of all attributes based on the Cartesian products of attributes  $X_a$  and  $X_b$  and save the results to matrix *sums*.
2. Determine initial modes based on the sums from matrix *sums*, selecting for each mode the category with highest occurrence. If two or more categories have the same number of occurrences select one of them randomly.
3. Run k-modes algorithm.
4. Return marked array  $c$ .

In the following example, attributes "Class" and "Survived" were selected as the basis for the clusters. This basis means that eight modes were created and their sums allocated as shown in Table 2.12. We can again see the sparse



Class + Survived	Class				Age		Sex		Survived	
	1st	2nd	3d	Crew	C.	A.	F.	M.	S.	P.
1st + S.	203	0	0	0	6	197	141	62	203	0
2nd + S.	0	118	0	0	24	94	93	25	118	0
3rd + S.	0	0	175	0	27	151	90	88	178	0
Crew + S.	0	0	0	212	0	212	20	192	212	0
1st + P.	122	0	0	0	0	122	4	118	0	122
2nd + P.	0	167	0	0	0	167	13	154	0	167
3rd + P.	0	0	528	0	52	476	106	422	0	528
Crew + P.	0	0	0	673	0	673	3	670	0	673

**Table 2.12:** Sums for the occurrence method when attribute "Class" was selected as a basis for clustering.

matrices, this time four of them at both ends of the table. The names of categories are again abbreviated in the same way as in chapter Occurrence Method

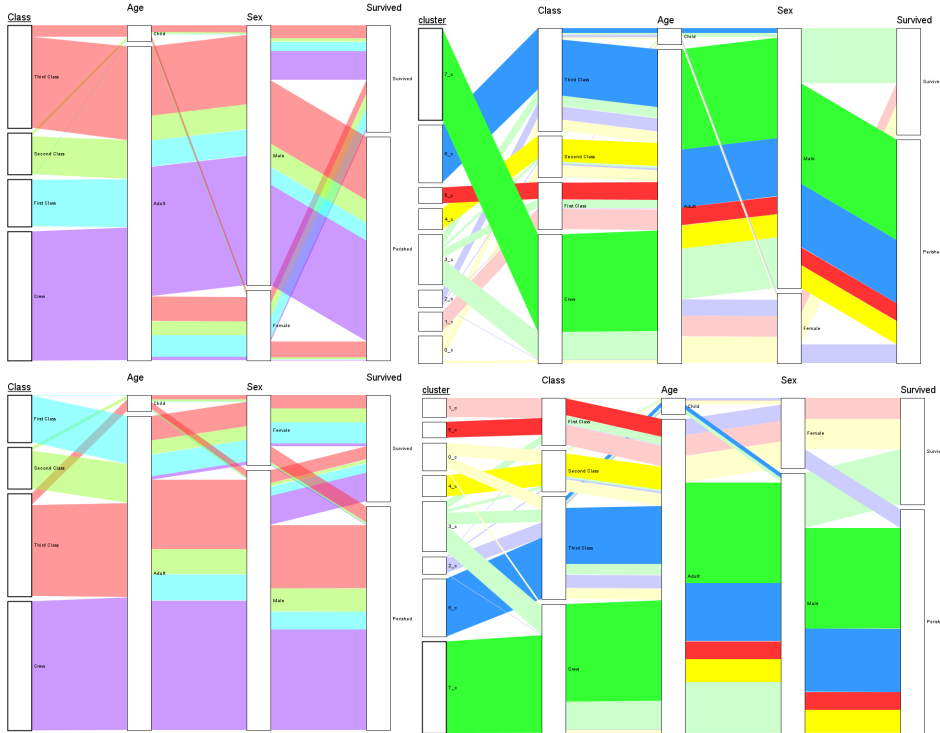
Table 2.13 shows how the initial modes were selected, and at the same time, it shows how the modes looked after the end of the k-modes method.

Cluster ID	Class	Age	Sex	Survived
1c	First Class	Adult	Female	Survived
0c	Second Class	Adult	Female	Survived
2c	Third Class	Adult	Female	Survived
3c	Crew	Adult	Male	Survived
5c	First Class	Adult	Male	Perished
4c	Second Class	Adult	Male	Perished
6c	Third Class	Adult	Male	Perished
7c	Crew	Adult	Male	Perished

**Table 2.13:** Initial and final modes for clusters resulting from the occurrence method.

If we have a look at Figure 2.29, we can notice four colors and two shades used to color the clustered plots on the right. The four colors each denoting a category from the attribute "Class". The shade shows which categories from attribute "Survived". The lighter shades are associated with the category "Survived" and the darker shades are associated with the category "Perished".

This clustering is the best yet, as we can finally see some trends to be highlighted. These trends are caused most apparently by the addition of the other sex in the clusters. Moreover, we can see distinction beyond the attributes "Sex" and "Survived". We can also use the different shades to see the proportions of passengers that survived and that perished. There are also some more confusing clusters. For example, cluster 3c (portray in light green in Figure 2.29) for some reason clustered together items from "First Class", "Third Class", and "Crew" that survived, but it is unlikely, that these items



**Figure 2.29:** Comparison for two-occurrence clustering using the k-modes method. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

had much in common in reality.

## 2.6.7 Parallelogram Clustering

The final method I have used was the method I call parallelogram clustering. All previous attempts utilizing the k-modes method showed me that the cluster would likely follow the data distribution based on the attributes and categories. If we make such an assumption, we can instead focus on a couple of modes around which we cluster the rest. In addition, we can focus on the parallelograms themselves. Specifically, we can not focus just on parallelograms between two pairs of attributes as in bundle layout. We need to focus on the entire "line" of parallelograms through all of the attributes.

We deduce the actual number of such parallelograms. The number of such unique "lines" will be the same as the number of unique items present in the dataset since all other same items will be mapped on the parallelograms created from the unique items. Let us have  $X_i$  as the  $i$ -th attribute and let  $|X_i|$  be the number of categories of attribute  $X_i$ . The total number of these parallelogram "lines" is:

$$\prod_{i=1}^n |X_i| = \text{uniqueItems}. \quad (2.11)$$

Furthermore we can learn more from the outcome of the best k-modes method, the occurrence and two-occurrence method (described in chapters 2.6.6 and 2.6.6 respectively). I add the existence of the primary attribute, which is used to select the basis for the clusters. In the case of this method, it means that it will select for each category of the selected primary attribute a parallelogram "line" with the highest number of items present.

Another new input is the threshold value from the interval  $[0, \dots, 1]$ . This threshold value is used to compare other parallelogram "lines" to the selected parallelogram "lines" that are used as clusters. Thus, we can simplify each parallelogram "line" to an individual item. Furthermore, since all items from the parallelogram "line" are identical, we can easily compare their distance. The distance can be measured between two unique items instead of the two parallelogram "lines".

If we set the threshold value to  $t$ , we can add to clusters those parallelogram "lines" that have their distance from them less or equal to the threshold. Thus, parallelogram "line" is added to a cluster with the lowest distance from it.

**Parallelogram method** algorithm is as follows:

■ **Input:**

- Dataset  $D$  containing  $n \in \mathbb{N}$  nominal or ordinal items,  $X_j$  is  $j$ -th attribute,  $|X_j|$  is number of categories of attribute  $X_j$ , there is  $k \in \mathbb{N}$  nominal and ordinal attributes.
- Priority vector  $\vec{p}$ ,  $|\vec{p}| = k$ ,  $\vec{p}_e \in \mathbb{N}$ ,  $e \in \mathbb{N}$ ,  $e \in [1, \dots, k]$ .
- Number  $t \in \mathbb{R}$ ,  $t \in [0, \dots, 1]$  as a threshold.
- Number  $a \in \mathbb{N}$ ,  $a \in [1, \dots, k]$  as an index of the primary attribute.
- Array *uniqueItems* of lists  $L$ . List  $L_u$ ,  $L_u \subset L$ ,  $u \in \mathbb{N}$ ,  $u \in [1, \dots, \text{uniqueItems}]$  of items representing unique parallelogram "lines".
- Array  $c$ ,  $|c| = n$ ,  $c_l = 0$ ,  $l \in \mathbb{N}$ ,  $l \in [1..n]$  for marking items to clusters.
- Matrix  $S$  of size  $\text{uniqueItems} \times \text{uniqueItems}$  of distances between all unique items (priority Dice measure is used as distance). Let  $s \in \mathbb{R}$  be any element of matrix  $S$ .

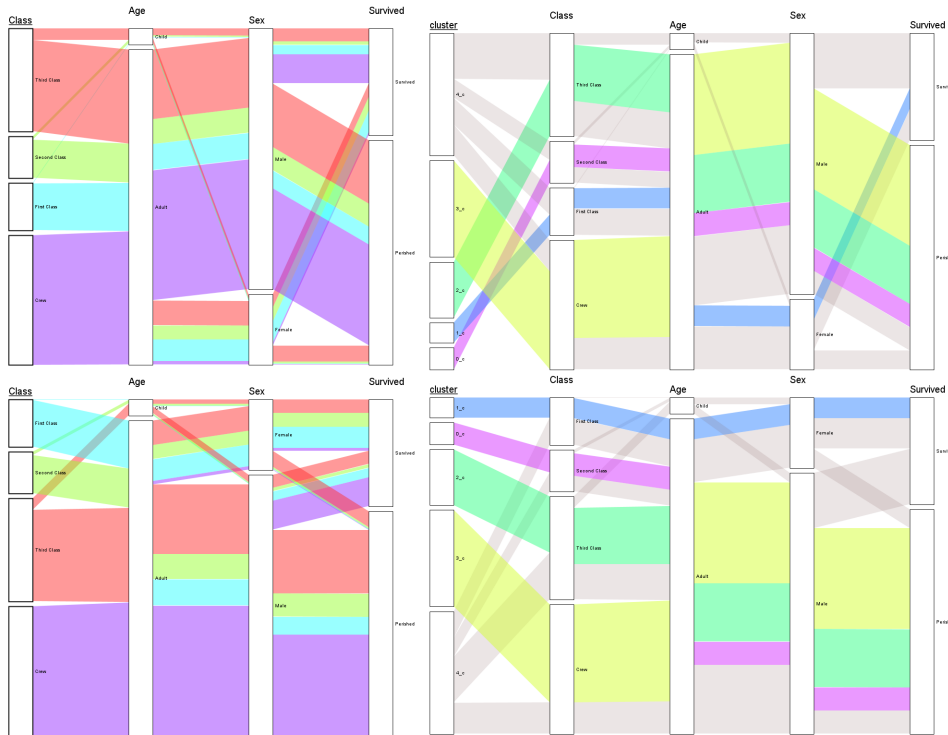
■ **Output:** Marked array  $c$ .

■ **Algorithm:**

1. Fill all list of  $L_u$  of array  $L$  with items. Each list consists of all items that create one parallelogram "line". In other words all items that have the same values for all of the nominal and ordinal attributes of the dataset.
2. Based on the  $X_a$  attribute, select  $|X_a|$  parallelogram "lines", that will serve as clusters. Each of these parallelogram "lines" have different value for the attribute  $X_a$ . Mark all items of these parallelogram "lines" in the array  $c$  with values from interval  $[1, \dots, |X_a|]$ . Use different value for each parallelogram "line".

3. For each other parallelogram "lines" check if its smallest distance to one of the cluster parallelogram "line" is less or equal to the threshold  $t$ . If there is such a parallelogram "line", mark all of its items in the array  $c$ . Mark the items in the same way as the cluster.
4. Return marked array  $c$ .

This algorithm will produce at most  $|X_a| + 1$  if there are some parallelogram "lines", that were not assigned to any cluster, or it will produce  $|X_a|$  clusters if all parallelograms were assigned. If the threshold value is set to 0, only the clusters selected in step 2 will be marked. If the threshold value is set to 1, all items of the data set will be marked.



**Figure 2.30:** Comparison for parallelogram clustering with threshold of 0.0. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

In this way, the user has control over the amount of clustering that occurs. The unassigned items can also be a powerful tool since they can show items that are in some way too different or too dissimilar from the selected cluster.

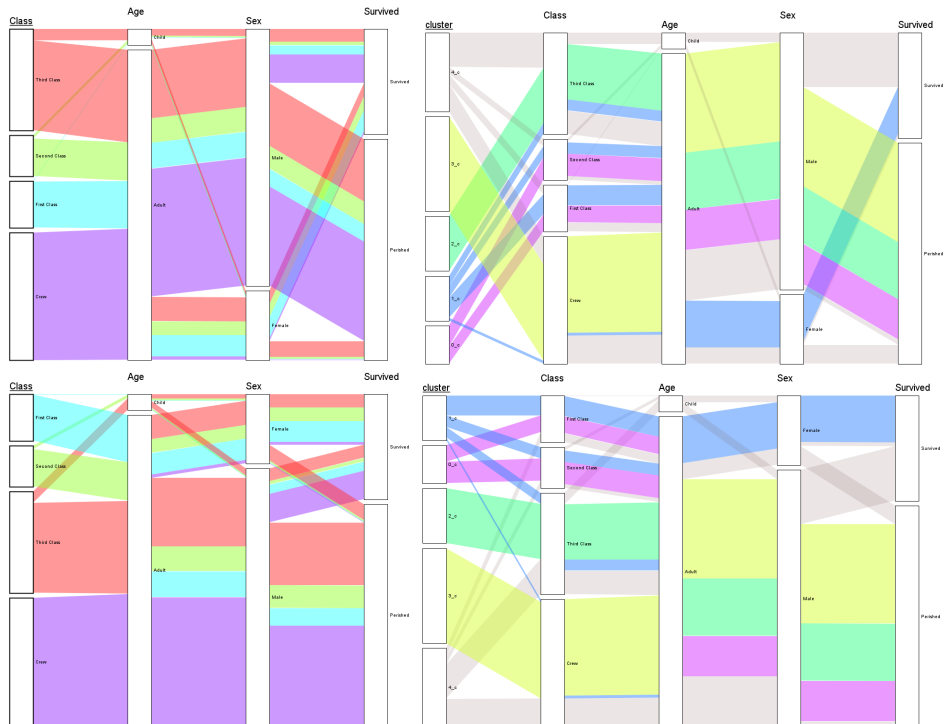
Since the number of clusters is derived from the number of categories of the primary attribute, I would recommend selecting an attribute with a medium number of categories from the dataset as a primary attribute. Selecting an attribute with a high number of categories will result in a very fragmented clustering. Conversely, selecting an attribute with a low number of categories will result in a very compacted clustering that will limit our understanding of the data.

I will now demonstrate the results of this method on four examples. All of these examples were run on the Titanic dataset, and all were run with the priority vector  $\vec{p} = (3, 1, 1, 2)$  for the attributes "Class", "Age", "Sex" and "Survived" respectively. This priority vector was used for counting the priority Dice measure as shown in chapter 2.6.2 Adapted Priority Dice Measure. I have selected the attribute "Class" as the primary attribute for all of these examples. They differ only in the value of threshold used - threshold values were 0.0, 0.2, 0.4, and 0.6, respectively. Clusters are annotated 0\_c, 1\_c, 2\_c and 3\_c. Cluster 4\_c represents the unassigned items. The first three examples will have all five clusters. The last example does not have any unassigned items and does not have cluster 4\_c.

Let us start with the threshold value of 0.0 as show in Figure 2.30. Here we can see the single parallelogram "lines" used as the clusters.

By looking at this example in the plots on the right of Figure 2.30, we already see some trends arising. We have the "First Class", "Adult", "Female", "Survived" cluster 1\_c in blue, which is diametrically different from the other clusters that are all "Adult", "Male" and "Perished".

We can also notice that a significant portion of the dataset has been clustered - over half of all the items. This clustered amount speaks further to the lack of diversity that some nominal and ordinal datasets can have regarding the unique items present.

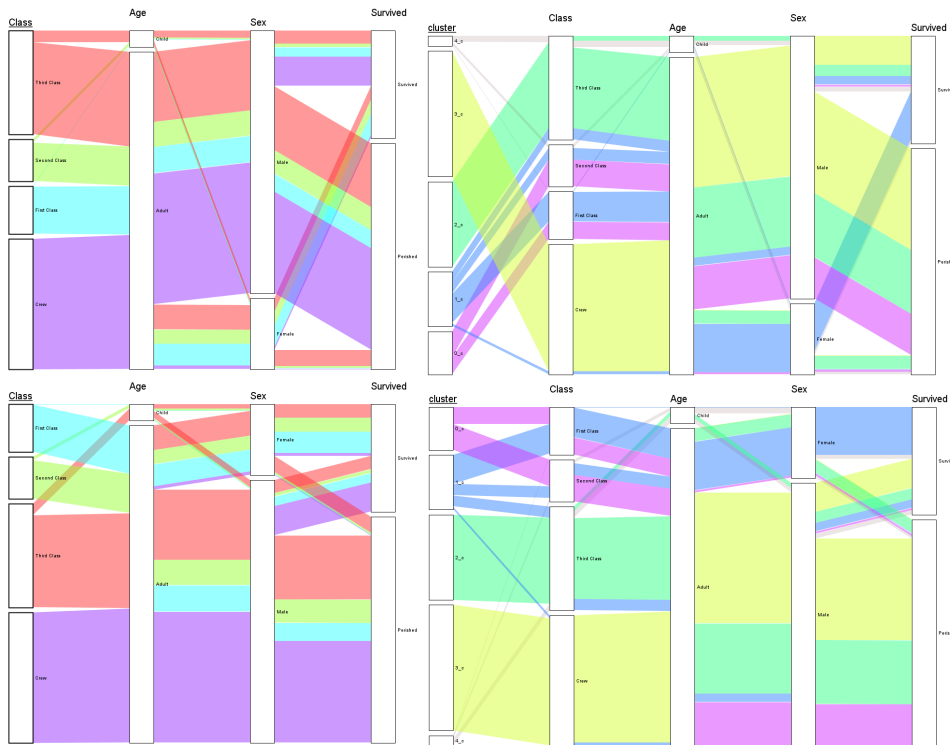


**Figure 2.31:** Comparison for parallelogram clustering with threshold of 0.2. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

Figure 2.31 shows the result with threshold set at 0.2. Further development can be seen especially with clusters 1\_c (in blue) and 0\_c (in purple). Where cluster 1\_c takes on other "Female", "Survived" items from the other classes. Cluster 0\_c tasks charge of all the "Male", "Perished" from both "First Class" and "Second Class". Since the "First Class" was initially represented by a cluster with "Female" and "Survived", items, it is clear why cluster 0\_c took charge of these items.

I would say that for this dataset, threshold 0.2 gives the best result. The resulting clustering is clearly defined. There is not a large portion of unassigned items. All clusterings ignored the category "Child" of attribute "Age" since it is comparably tiny. It is no different here, but I believe that it does not negatively affect the result.

We can now have a look at Figure 2.32 with the threshold of 0.4. We can see that almost all items were assigned. The majority of the unassigned items are items with the category "Child" of the "Age" attribute.

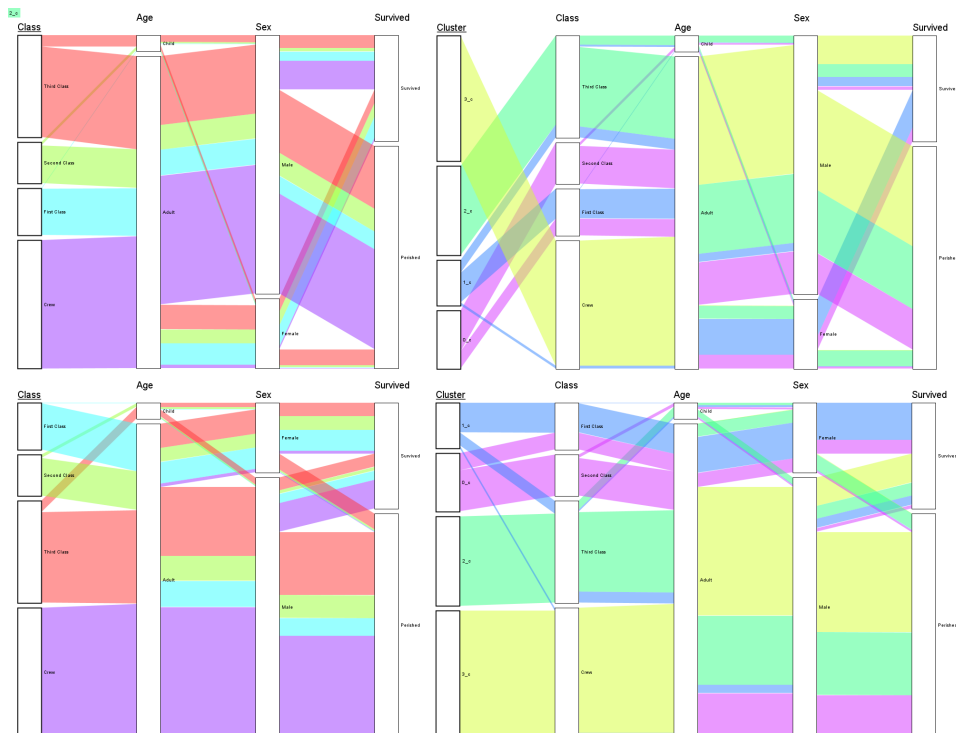


**Figure 2.32:** Comparison for parallelogram clustering with threshold of 0.4. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

Even though this clustering still offers a very similar result to the clustering with a threshold of 0.2, I prefer the 0.2 threshold clustering. The most important benefits of the 0.4 clusterings can again be seen with clusters 1\_c (in blue) and 0\_c (in purple). Cluster 1\_c takes more "Female", "Survived" items, and similarly cluster 0\_c takes more of "Male", "Perished" items of

the two higher classes. The remaining two clusters took on other items that, in my opinion, polluted their meaning of clusters of "Male", "Perished" form "Third Class" and "Crew" respectively for clusters 2\_c and 3\_c.

Lastly we have threshold 0.6 in Figure 2.33. The first thing to notice is, that cluster 4\_c is not present in the dataset, meaning that all items were assigned to one of the four remaining clusters. This assignment means there would be no point in running this method with a higher threshold, as it would end the same way.



**Figure 2.33:** Comparison for parallelogram clustering with threshold of 0.6. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

Since only a small number of items were unassigned with a threshold of 0.4, the result is not very different for the threshold of 0.4. We can see the clusters get more and more "polluted" with different values.

In the end, I have selected the parallelogram clustering method as the method I will use in my implementation. I have made this selection because I think that the clear distinction between the clusters shown in Figure 2.31 with threshold 0.2 offers the best view of the trends of this particular dataset.

## Chapter 3

### Implementation

In this chapter I will focus of the implementation details of my algorithm described in chapter 2.6.7 Parallelogram Clustering. Firstly I state the used technologies in chapter 3.1 Used Technologies. Chapter 3.2 XDat explains the basic functionality of the XDat application. I discuss the structure of used classes in chapter 3.3 Structure of Classes. Finally I present the update interface for the newly added parallelogram clustering method in chapter 3.4 Parallelogram Method User Interface.

#### 3.1 Used Technologies

I have used the application XDat version 2.2 with modification by Bc. Martin Janda. I have made modifications to this version 2.2 of XDat used Java JDK 8 for implementing further changes. I have developed these changes using Apache NetBeans IDE 12.0. I have also used XDat version 2.4 for some additional testing.

#### 3.2 XDat

XDAT [xdane], or X-dimensional Data Analysis Tool is a data visualization tool for the visualization of parallel coordinates[xdane]. It also supports the plotting of data in 2D scatter charts. XDAT is free software licensed under the GPL, and I will be using it to visualize my data. Specifically, I will be using the XDAT version 2.2 modified by Martin Janda (modifications by Martin Janda are mentioned below).

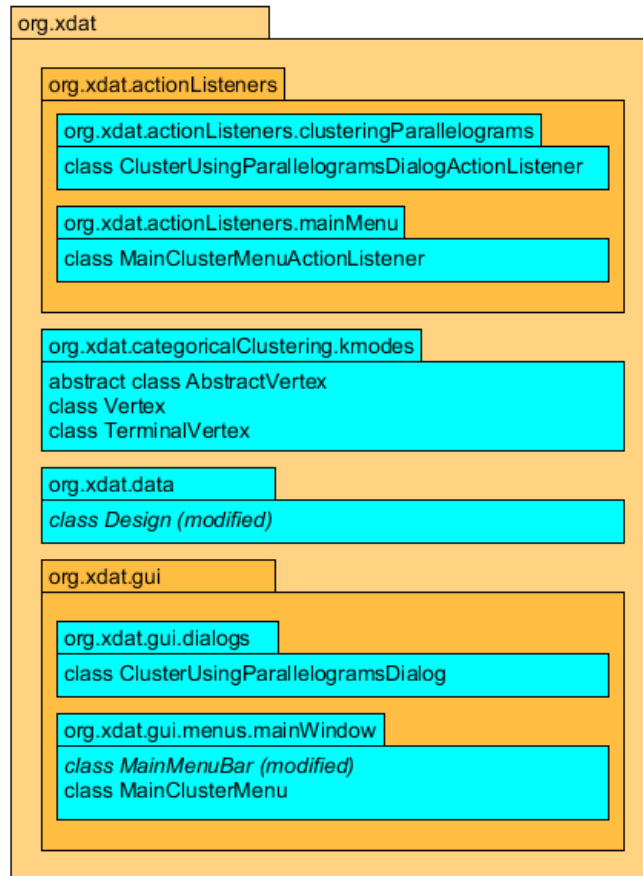
During his work on XDAT as a part of his master's thesis, Martin Janda made many changes to the capabilities of XDAT. Although you can find the list of all these changes in his bachelor thesis, I am most interested in visualizing parallel sets. As mentioned above, my work stands on the functionality added to the XDat application, specifically the visualization of the parallel sets method. Martin Janda's parallel sets visualization extension to the XDat application offers both layouts - bundle and tree [Jan21].

I have further expanded the capabilities of XDat by adding the functionality for clustering nominal and ordinal data.



### 3.3 Structure of Classes

Figure 3.1 shows the structure of packages and classes created for the final clustering method - parallelogram method. Newly added or modified classes are highlighted in cyan. Only modified classes are highlighted with cursive and with the notation "(modified)".



**Figure 3.1:** Showcase of added and packages and class (in cyan) and modified classes and packages (in cyan and in cursive) relevant to the implemented parallelogram clustering method.

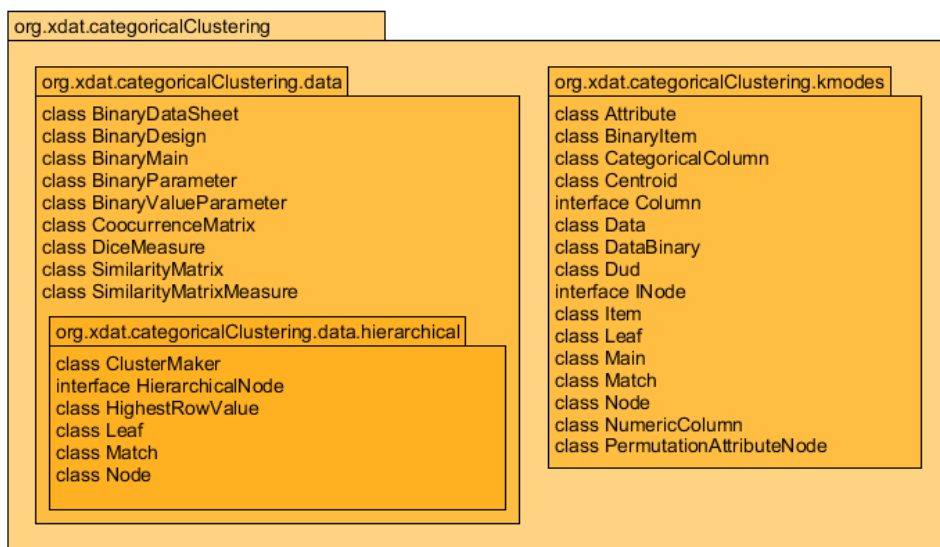
As their name suggests classes `ClusterUsingParallelogramsDialogActionListener`, `MainClusterMenuActionListener`, `ClusterUsingParallelogramsDialog`, `MainMenuBar` and `MainClusterMenu` are GUI classes. Class `MainMenuBar` represents the main menu of the application. To this class, I added a new menu item in class `MainClusterMenu`, which allows the user to initiate the clustering. Class `MainClusterMenu` is controlled by the action listener class `MainClusterMenuActionListener`. The necessary setup for parallelogram clustering and the setting of all necessary parameters happens in a dialog window expressed by class `ClusterUsingParallelogramsDialog` and opens through the `MainClusterMenu` class. Finally the `ClusterUsingParallelogramsDialogActionListener`

is an action listener class that controls the ClusterUsingParallelogramsDialog class.

Class Design represents one item of the dataset. It was modified in order to allow for the calculation of the priority Dice measure.

Classes AbstractVertex, Vertex and TerminalVertex. Classes Vertex and Terminal Vertex are children of AbstractVertex. These classes were used to find parallelograms and build a tree-like structure representing them. Thus, each Vertex represents one unique category in the hierarchy, and each TerminalVertex also represents one unique category in the hierarchy, but simultaneously being a leaf.

Packages and classes shown in Figure 3.2 represent all added classes that were not shown in Figure 3.1. These classes were used for the testing of all of the clustering methods mentioned above. As such, they are rough and lack any form of a user interface.



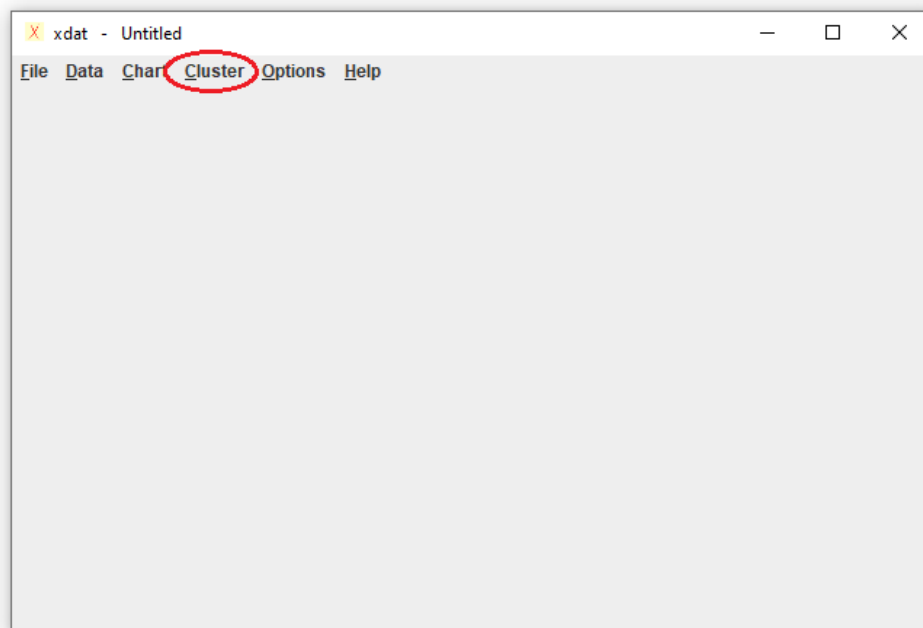
**Figure 3.2:** Showcase of all added method (except those shown in Figure 3.1) that were used for all tested clustering methods.

In the package org.xdat.categoricalClustering.data, we find most of the classes used in the hierarchical clustering. These classes are the most "raw" of all of them since they were my first attempt at implementing the transformation of the dataset into a binary structure.

Class BinaryDataSheet shelters the dataset in binary form and is composed of classes BinaryDesign and BinaryParameter. The former represents an item of the dataset in binary form. The latter represents a single attribute and is further composed of the BinaryParameterValue class, representing a given category.

CoocurrenceMatrix class counts the number of occurrences of categories. SimilarityMatrix class is used to store the distance measures of elements. DiceMeasure class, which implements the SimilarityMatrixMeasure, is used to calculate the basic Dice measure without the priority modification. Simi-





**Figure 3.3:** Main menu of the XDat tool without any loaded data. New menu item "Cluster" is highlighted in red.

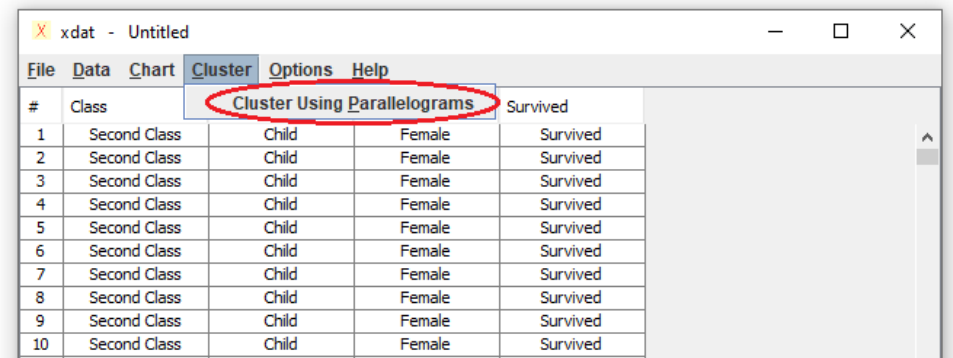
 A screenshot of the XDat tool's main menu after loading the Titanic dataset. The window title is "xdat - Untitled". The menu bar contains "File", "Data", "Chart", "Cluster", "Options", and "Help". The main area of the window displays a table with the following data:
 

#	Class	Age	Sex	Survived
1	Second Class	Child	Female	Survived
2	Second Class	Child	Female	Survived
3	Second Class	Child	Female	Survived
4	Second Class	Child	Female	Survived
5	Second Class	Child	Female	Survived
6	Second Class	Child	Female	Survived
7	Second Class	Child	Female	Survived
8	Second Class	Child	Female	Survived
9	Second Class	Child	Female	Survived
10	Second Class	Child	Female	Survived
11	Second Class	Child	Female	Survived
12	Second Class	Child	Female	Survived
13	Second Class	Child	Female	Survived
14	Second Class	Child	Male	Survived
15	Second Class	Child	Male	Survived
16	Second Class	Child	Male	Survived
17	Second Class	Child	Male	Survived
18	Second Class	Child	Male	Survived
19	Second Class	Child	Male	Survived
20	Second Class	Child	Male	Survived
21	Second Class	Child	Male	Survived
22	Second Class	Child	Male	Survived

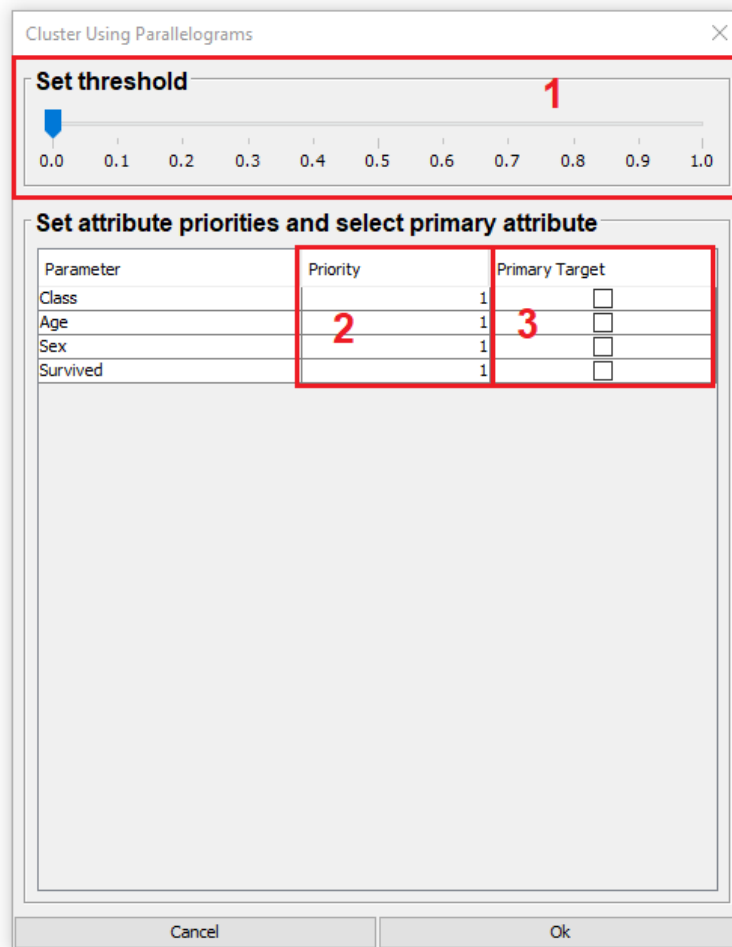
**Figure 3.4:** Main menu of the XDat tool after the load of the Titanic dataset.

The third area is used for the selection of the primary attribute. Only one attribute can be selected as a primary attribute. If the user selects no attribute, the first attribute in the list is used.

Finally in Figure 3.7 we can see the result of clustering. The method



**Figure 3.5:** Extension of the "Cluster" menu item of the main menu showing the "Cluster Using Parallelograms" option highlighted in red.



**Figure 3.6:** Dialog window for the parallelogram clustering method. Highlighted in red are slider to set the threshold for the clustering (1), input fields for setting of priority for the priority vector (2) and a selection for the primary attribute (3).

The screenshot shows a window titled 'xdat - Untitled' with a menu bar containing 'File', 'Data', 'Chart', 'Cluster', 'Options', and 'Help'. Below the menu bar is a table with the following columns: '#', 'Class', 'Age', 'Sex', 'Survived', and 'Cluster'. The 'Cluster' column is highlighted with a red border. The table contains 22 rows of data, all with 'Survived' status and 'c\_unassigned' cluster values.

#	Class	Age	Sex	Survived	Cluster
1	Second Class	Child	Female	Survived	c_unassigned
2	Second Class	Child	Female	Survived	c_unassigned
3	Second Class	Child	Female	Survived	c_unassigned
4	Second Class	Child	Female	Survived	c_unassigned
5	Second Class	Child	Female	Survived	c_unassigned
6	Second Class	Child	Female	Survived	c_unassigned
7	Second Class	Child	Female	Survived	c_unassigned
8	Second Class	Child	Female	Survived	c_unassigned
9	Second Class	Child	Female	Survived	c_unassigned
10	Second Class	Child	Female	Survived	c_unassigned
11	Second Class	Child	Female	Survived	c_unassigned
12	Second Class	Child	Female	Survived	c_unassigned
13	Second Class	Child	Female	Survived	c_unassigned
14	Second Class	Child	Male	Survived	c_unassigned
15	Second Class	Child	Male	Survived	c_unassigned
16	Second Class	Child	Male	Survived	c_unassigned
17	Second Class	Child	Male	Survived	c_unassigned
18	Second Class	Child	Male	Survived	c_unassigned
19	Second Class	Child	Male	Survived	c_unassigned
20	Second Class	Child	Male	Survived	c_unassigned
21	Second Class	Child	Male	Survived	c_unassigned
22	Second Class	Child	Male	Survived	c_unassigned

**Figure 3.7:** Showcase of all added method (except those shown in Figure 3.1) that were used for all tested clustering methods.

augments the dataset loaded into the XDat tool. It adds the column/attribute "Cluster" and sets the values for every item based on its membership to the clusters (highlighted in red). The source file for the dataset is not augmented. That would require the user to use the XDat functionality to save the new dataset.



## Chapter 4

### Experimental Results

In this chapter, I test the parallelogram clustering method on four different datasets. All of these datasets are either nominal and ordinal or mixed. The testing is done in order of ascending dimensions of the datasets.

#### 4.1 Cars Dataset

The cars dataset consists of 6 ordinal attributes. They are:

- "Economy"
  - categories: "very bad", "bad", "average", "good", "very good".
- "Cylinders"
  - categories: "8", "5", "4", "3".
- "Power"
  - categories: "very strong", "strong", "average", "weak", "very week".
- "weight"
  - categories: "very heavy", "heavy", "average", "light", "very light".
- "0-60 mph"
  - categories: "very good", "good", "average", "bad", "very bad".
- "year"
  - categories: "very old", "old", "average", "new", "very new".

For the purpose of parallelogram clustering I have used priority  $\vec{p} = (5, 2, 4, 1, 3, 1)$ . I have clustered dataset Cars four times with different thresholds - 0.1, 0.2, 0.3 and 0.4. I have used attribute "Economy" as the primary attribute for the basis of clustering. This means, that the dataset will be divided into five clusters. In all Figures 4.1, 4.2, 4.3 and 4.4 there are 6



clusters present. The sixth cluster "c\_unassigned" represents the unassigned items.

Figure 4.1 shows the clustering with threshold 0.1. Already we can see the trend that is the same throughout all for examples. All five clusters are evenly spread through the attributes and their five categories.

This allows us to define the two parallelograms at the opposite ends - cluster c0 and cluster c3. Cluster c0 represents "very good" "Economy", "8" "Cylinder", "average" "Power", "very heavy" "Weight", "good" "0-60 mph" and "very old" "Year". Opposite to c0 is cluster c3 "bad" "Economy", "3" "Cylinder", "very weak" "Power", "very light" "Weight", "average" "0-60 mph" and "very new" "Year".

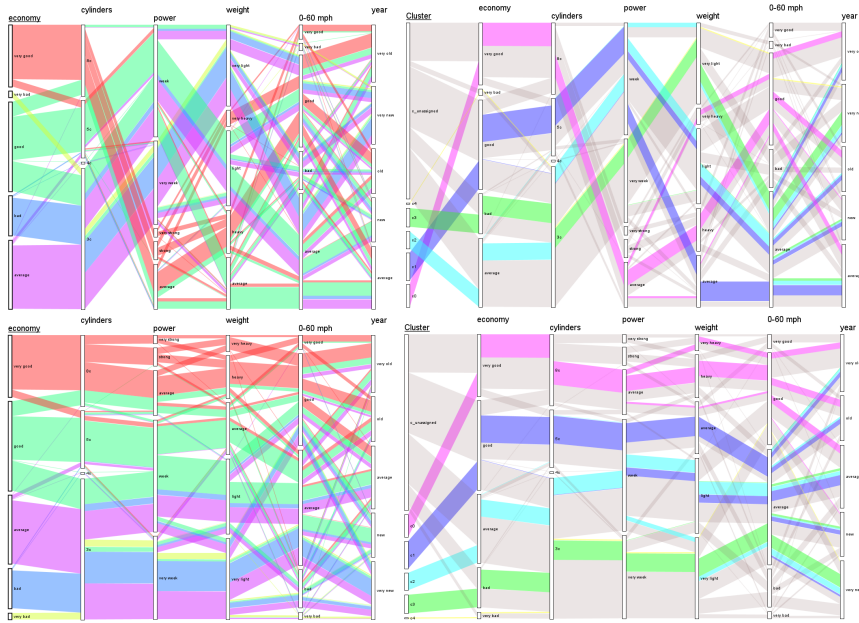
We can already notice that the last two attributes, "0-60 mph" and "Year" cause the most problems. This outcome is to be expected, as in the case of "0-60 mph" we can expect most cars to be average in their acceleration. In the case of the "Year" attribute, we can already see that cars of varying performance were made in every period with the trend to faster, lighter, and more expensive cars to the present and with slower, heavier, and less expensive cars to the past.

In Figure 4.2 we can see the clustering with threshold 0.2. As was expected, the trend shown in Figure 4.1 is only more strongly reinforced with a more lenient threshold.

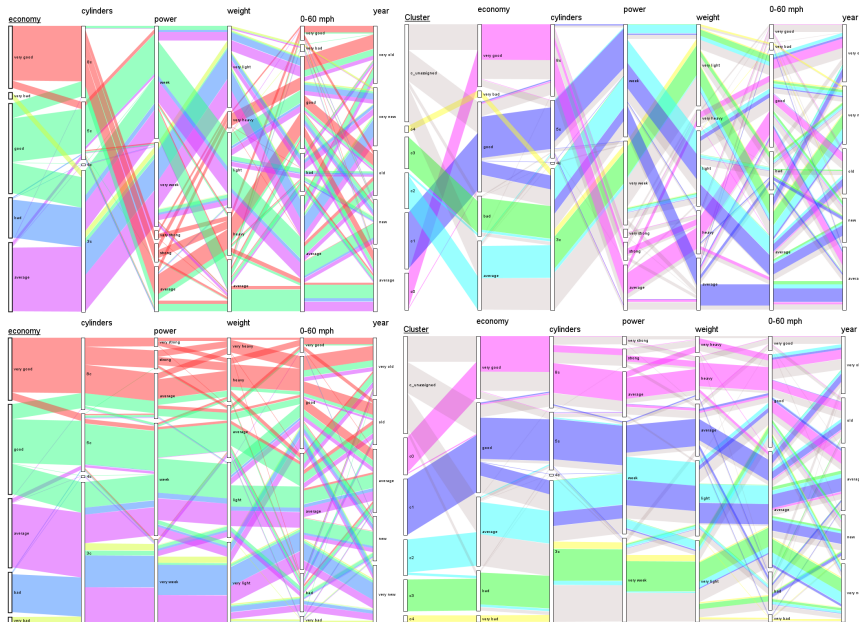
Continuing this trend is also the clustering with threshold 0.3 seen in Figure 4.3. Here we can see that the plot becomes more cluttered to the end in the two problematic attributes, "0-60 mph" and "Year".

Even so, I consider the threshold 0.3 as the best for the current set of used parameters as the clustering is relatively uncluttered and can be read very well.

In the final set of plots in Figure 4.4, we can see that the clusters in the plot become increasingly more tangled between each other. This increasing entanglement follows the trend we saw with the clustering of the Titanic dataset in chapter 2.6.7 Parallelogram Clustering.

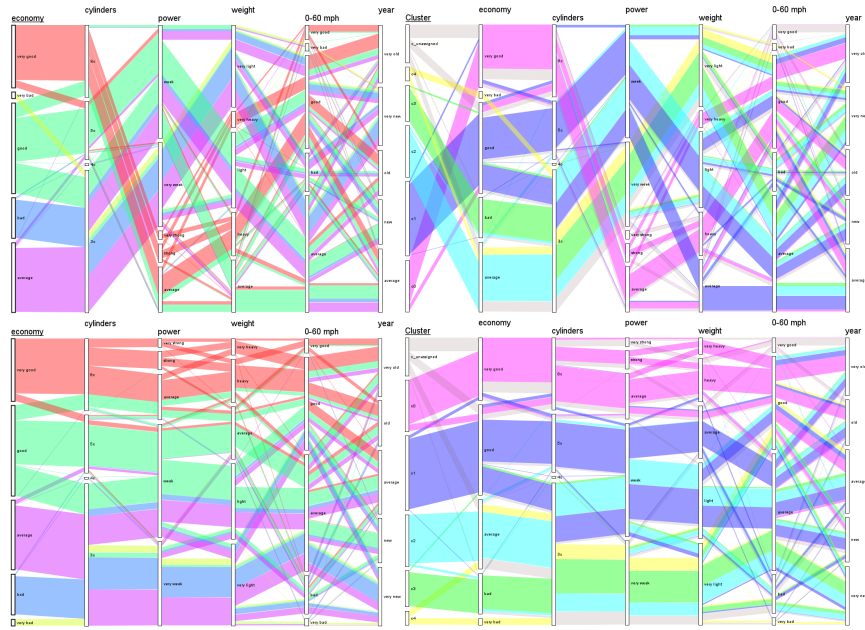


**Figure 4.1:** Comparison for parallelgram clustering of the Cars dataset with **threshold of 0.1**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

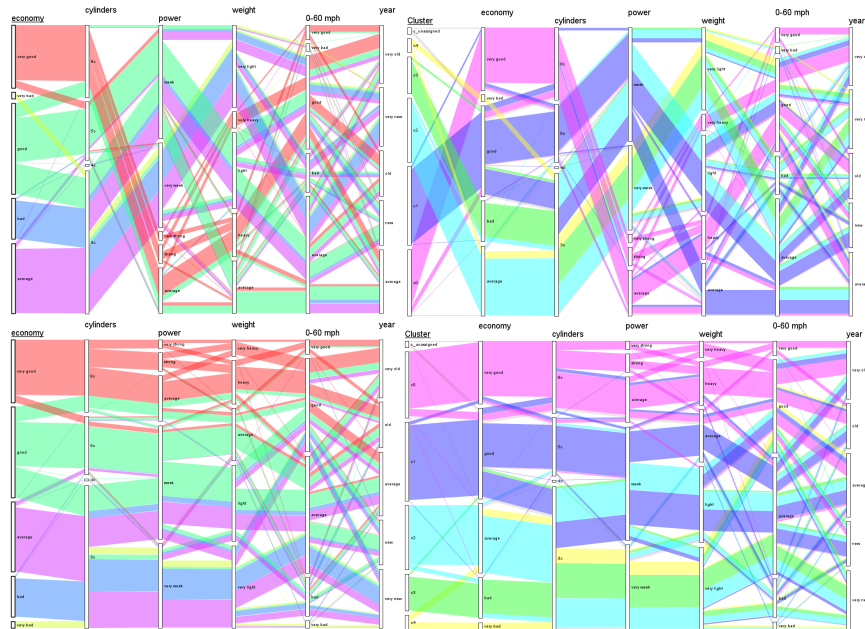


**Figure 4.2:** Comparison for parallelgram clustering of the Cars dataset with **threshold of 0.2**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

#### 4. Experimental Results



**Figure 4.3:** Comparison for parallelogram clustering of the Cars dataset with **threshold of 0.3**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).



**Figure 4.4:** Comparison for parallelogram clustering of the Cars dataset with **threshold of 0.4**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

## 4.2 States Dataset

Dataset States describes the member and candidate states of the European Union and other European states. Its nominal and ordinal attributes are:

- "Country"
  - categories: "Albania", "Austria", "Belgium", "Bulgaria", "Croatia", "Cyprus", "Czech Republic", "Denmark", "Estonia", "Finland", "France", "Germany", "Greece", "Hungary", "Iceland", "Ireland", "Italy", "Latvia", "Liechtenstein", "Lithuania", "Luxembourg", "Macedonia", "Malta", "Montenegro", "Netherlands", "Norway", "Poland", "Portugal", "Romania", "Serbia", "Slovakia", "Slovenia", "Spain", "Sweden", "Switzerland", "Turkey", "United Kingdom".
- "Language"
  - categories: "Albanian", "German", "Dutch; French; German", "Bulgarian", "Croatian", "Greek; Turkish", "Czech", "Danish", "Estonian", "Finnish; Swedish", "French", "German", "Greek", "Hungarian", "Icelandic", "English; Irish", "Italian", "Latvian", "German", "Lithuanian", "French; German; Luxembourgish", "Macedonian", "Maltese; English", "Serbian", "Dutch", "Norwegian", "Polish", "Portuguese", "Romanian", "Serbian", "Slovak", "Slovene", "Spanish", "Swedish", "German; French; Italian; Romansh", "Turkish", "English".
- "European Union"
  - categories: "Member", "Candidate", "-".
- "European Single Market"
  - categories: "Member", "-".
- "European Monetary Union"
  - categories: "Not Applicable", "Member", "Candidate", "-".
- "European Free Trade Agreement"
  - categories: "-", "Member".
- "Currency"
  - categories: "Denar", "Dinar", "Euro", "Forint", "Franc", "Koruna", "Krona", "Króna", "Krone", "Kuna", "Lek", "Leu", "Lev", "Lira", "Pound Sterling", "Złoty".
- "Currency Code"

- categories: "MKD", "RSD", "EUR", "HUF", "CHF", "CZK", "SEK", "ISK", "DKK", "NOK", "HRK", "ALL", "RON", "BGN", "TRY", "GBP", "PLN".

This dataset also contains several quantitative attributes. Although they were not used in the clustering or shown in the plots in Figures 4.5, 4.6 and 4.7, I mention them here to give the full account of the dataset.

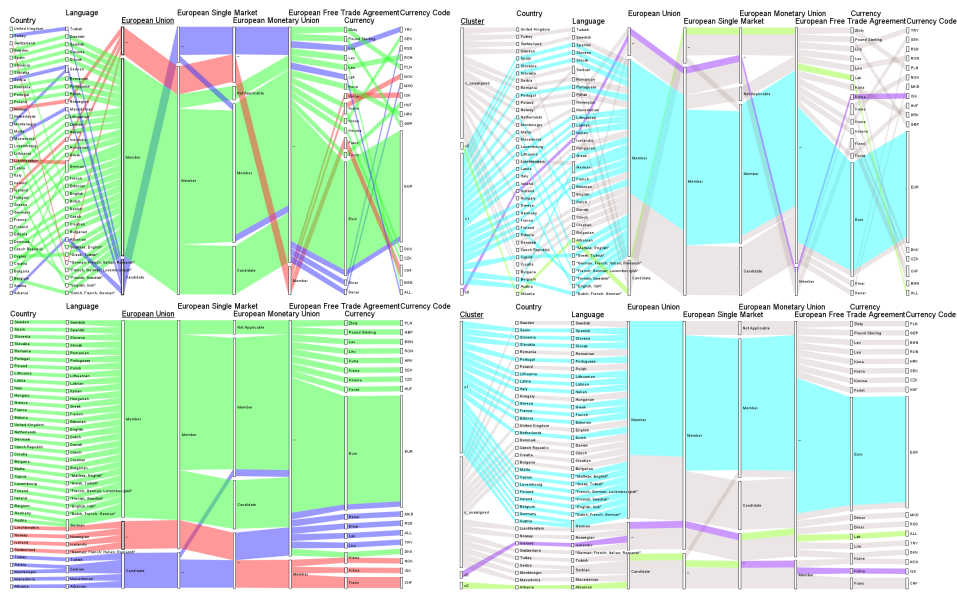
- "Accession Year",
- "Council Votes",
- "European Parliament Seats",
- "Population",
- "Area (km<sup>2</sup>)",
- "GDP (€| millions)",
- "GDP (\$| millions)",
- "GDP per capita (\$|millions)".

For parallelogram clustering, I have used priority  $\vec{p} = (1, 1, 5, 4, 3, 2, 1, 1)$  for the nominal and ordinal attributes in the same order as they are shown at the start of this chapter. I have clustered dataset States three times with different thresholds - 0.1, 0.2, and 0.3. I have opted to use the attribute "European Union" as the primary attribute for the basis of clustering, as it is the most crucial attribute in my view. This selection means that there are three clusters. For the thresholds 0.1 and 0.2, there is also a cluster for the unassigned items. For threshold 0.3, no unassigned items remain.

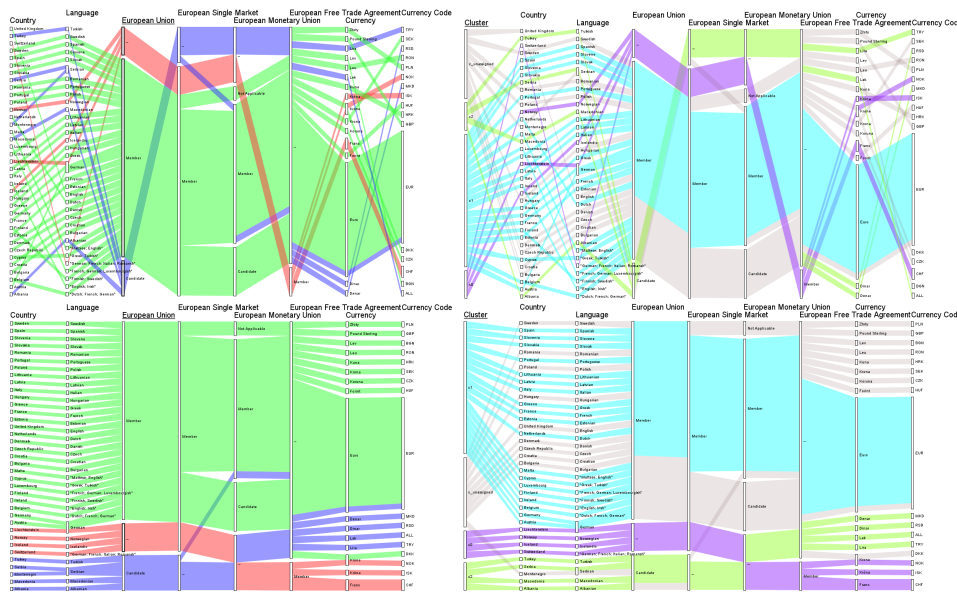
Figure 4.5 shows clustering with the threshold of 0.1. Already we can see that the clustering is following the "European Union" primary attribute distribution. Cluster c1 clusters European Union members, cluster c2 focuses on the candidate countries, and cluster c0 has the remaining unaffiliated countries.

In Figure 4.6 with the clustering at the threshold of 0.2, we can see an enlargement of the clusters c0, c1, and c2. All of these clusters follow the same trends as in Figure 4.5.

When clustering with threshold 0.3 as seen in Figure 4.7, we can see that all the items have been assigned. In the case of the States dataset, I find the threshold 0.3 as best. All clusters are clearly defined, and all the clusters follow a clear trend in the data. The only thing worth mentioning is the stray green parallelogram seen in the bottom right plot of Figure 4.7. This stray parallelogram is partly caused by the parallel sets algorithm and bundle layout. The bundle layout reorganizes every parallelogram between all pairs. It is further caused by Montenegro's having adopted the Euro without being officially a member of the Monetary Union.

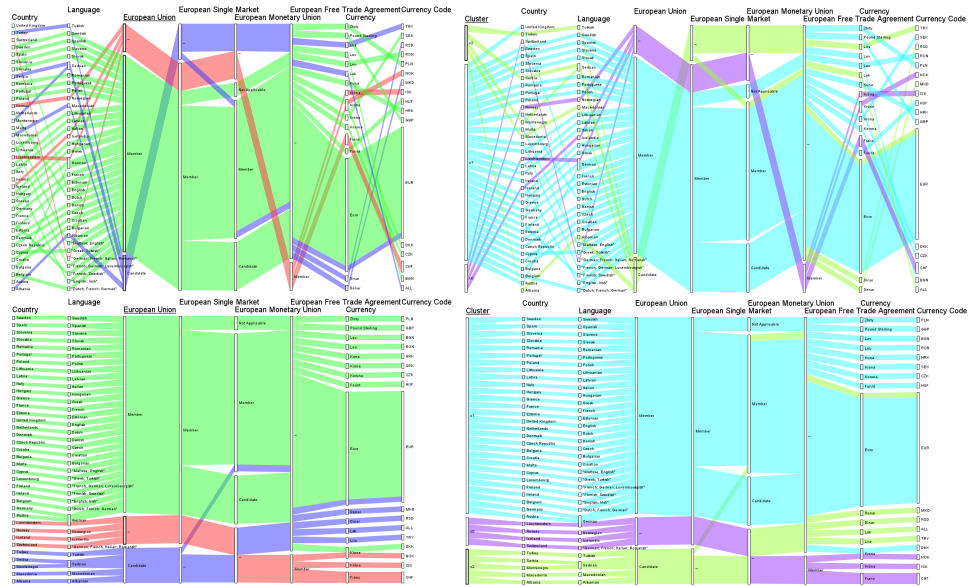


**Figure 4.5:** Comparison for parallelogram clustering of the States dataset with **threshold of 0.1**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).



**Figure 4.6:** Comparison for parallelogram clustering of the States dataset with **threshold of 0.2**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

## 4. Experimental Results



**Figure 4.7:** Comparison for parallelogram clustering of the States dataset with **threshold of 0.3**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

### 4.3 New York Hotels Dataset

New York Hotels dataset describes New York Hotel quality service and available services. There are nine nominal and ordinal attributes, seven of which will be used in clustering of the dataset. They are:

- "Name".
- "Badge"
  - categories: "Exceptional", "Superb", "Fabulous", "Very Good", "Good", "Undefined".
- "Stars"
  - categories: "5", "4.5", "4", "3.5", "3", "2.5", "2", "1.5", "1".
- "District".
- "Distance".
- "Free Wifi"
  - categories: "No", "Yes".
- "Breakfast Included"
  - categories: "No", "Yes".

- "Cancellation"
  - categories: "Cancellation policy", "Free cancellation", "Non-refundable".
- "Free Parking"
  - categories: "No", "Yes".
- "Room Type"
  - categories: "2 people", "3 people", "4 people", "5 people", "6 people".

Attributes "Name" and "District" will not be used even though they are nominal. These attributes have so many unique categories that it becomes impossible to distinguish them when visualized by the parallel sets method and are thus unusable. These two attributes do take part in the clustering of the dataset. Attribute "Distance" is mixed (contains both quantitative and nominal categories) and thus can not be used for the clustering.

Further more there are five attributes, that will not be used. They are:

- "Rating",
- "Price".

The remaining two attributes "Rating" and "Price" are qualitative and also can not be used. Neither of these attributes will be visualized in Figures 4.8, 4.9 and 4.10.

For the purpose of parallelogram clustering I have used priority  $\vec{p} = (1, 6, 4, 1, 1, 4, 3, 3, 2, 1)$ . The vector  $\vec{p}$  contains all the elements mentioned in the list of nominal and ordinal attributes at the top of the chapter. As is described above, only some of these will be used. I have clustered the dataset New York Hotels three times with different thresholds - 0.2, 0.3, and 0.4. I have used the attribute "Badge" as the primary attribute for the basis of clustering.

Figure 4.8 shows the clustering at the threshold of 0.2. We can immediately see that most of the items have not been clustered. Even so, we can see some correlation between the "Badge" and "Stars" attributes shown through the clusters. The clusters tend to connect items with either high values for "Badge" and "Stars" or low values.

We should also have a look at the four services attributes "Free Wifi", "Breakfast Included", "Cancellation" and "Free Parking". Again, we can see that all clusters are somewhat muddled through these four attributes, and no clear trend stands out.

In Figure 4.9 we can see clustering with 0.4 threshold. We should note that on the one hand, the trend with the "Badge" and "Stars" attributes continues. On the other hand, the clusters "changed position". For example cluster c2 depicted in both Figures 4.8 and 4.9 in blue tended for poorer values in the attributes "Badge" and "Star", but with the threshold 0.4 now tends towards the middle of the spectrum.

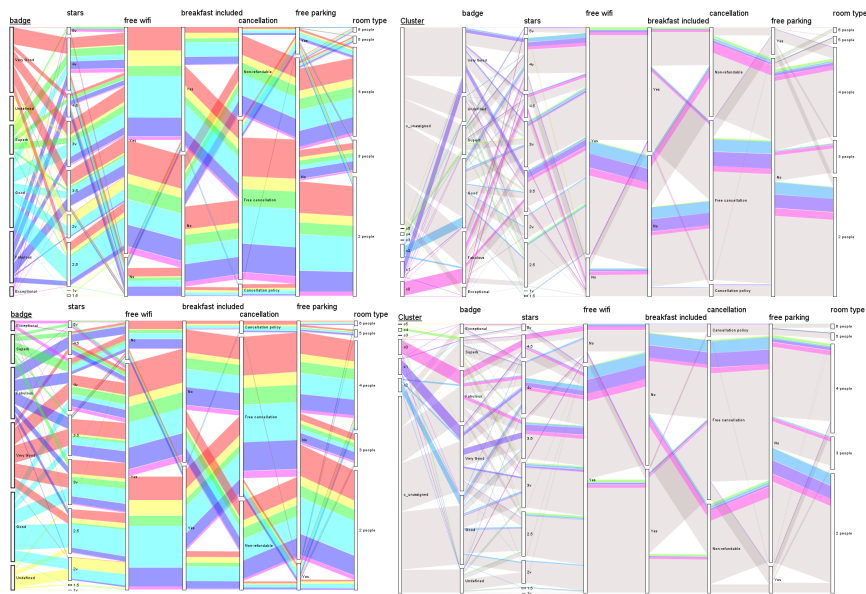


Also, the other trend concerning the four services attributes continues. No clear trend is apparent amongst them.

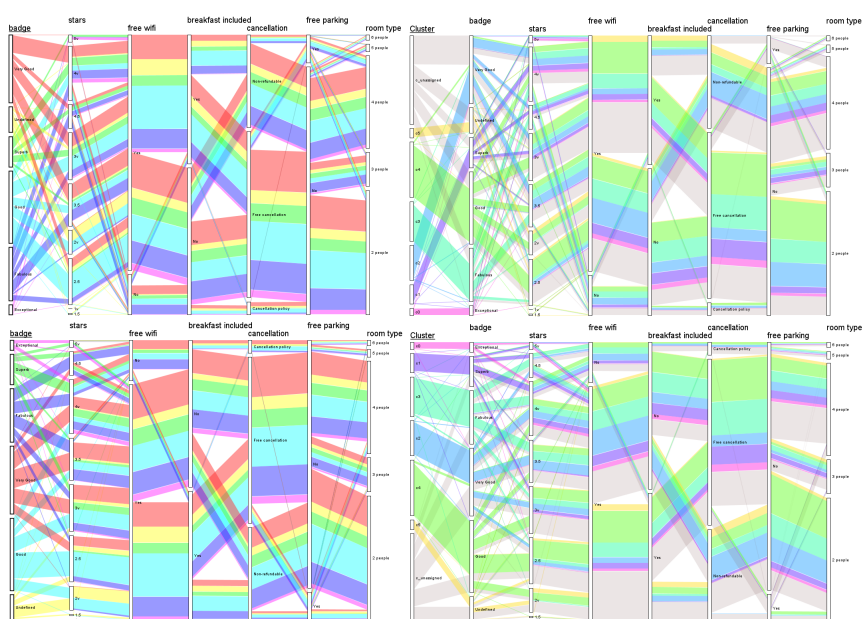
Comparing all three clusterings, I consider the threshold 0.3 the best.

As can be seen in Figure 4.10 both above mentioned trends continue. We can see more clutter between the "Cluster" and "Badge" attributes which could mean that the threshold was too high and marginal elements were added to the clusters. Cluster c5 in yellow in Figure 4.10 is a prime example having badge values from both the "Fabulous" and "Undefined" categories.

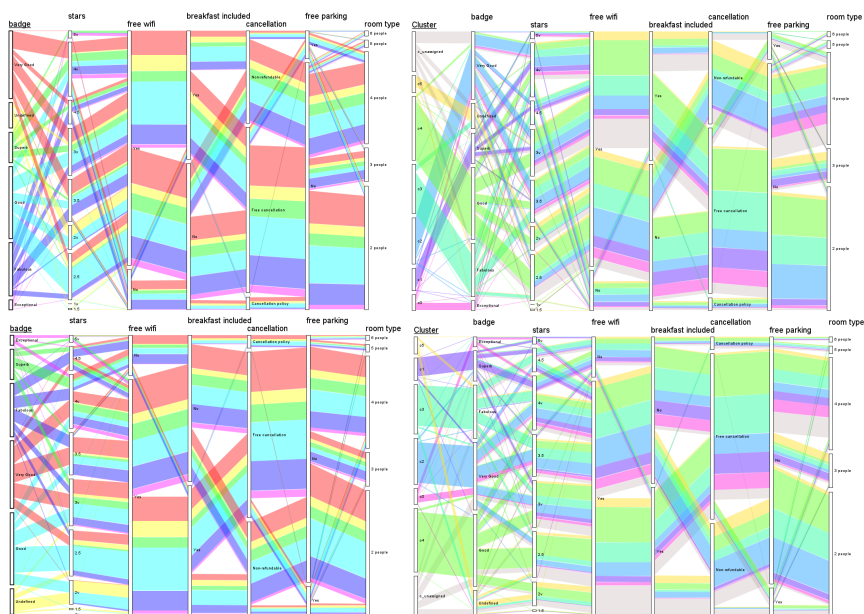
In conclusion to the New York Hotel dataset clustering, I would say that no apparent clusters are present. Beyond the first two attributes, we can not discern any meaningful trend. In this case, I see the issue with the dataset, as it simply contains data devoid of any meaningful clusters (when we observe the dataset on the seven visualized attributes).



**Figure 4.8:** Comparison for parallelogram clustering of the New York Hotels dataset with **threshold of 0.2**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).



**Figure 4.9:** Comparison for parallelogram clustering of the New York Hotels dataset with **threshold of 0.3**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).



**Figure 4.10:** Comparison for parallelogram clustering of the New York Hotels dataset with **threshold of 0.4**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).

## 4.4 Election Dataset

Election dataset represents the opinions of 26 Czech political parties on 34 question. Since it would be impossible to visualize all 34 question and hope that any information could be read from such a plot, I have limited the dataset to the first fifteen questions. The attributes are:

- "Party Name"
    - categories: "ANO", "Blok proti islamizaci", "ČSNS", "ČSSD", "DSSS", "Hnutí Cesta", "KDU-ČSL", "KSČM", "Národ Sobě", "Občané 2011", "ODA", "ODS", "Piráti", "Prokopské údolí (SPDV)", "Radostné Česko", "Realisté", "Referendum o EU", "Rozumní", "Řád národa", "SPD", "SPO", "SPR-RSČ Miroslava Sládka", "Starostové a nezávislí", "Svobodní", "TOP 09", "Zelení".
1. "Česká republika by měla vystoupit z Evropské unie." In English "The Czech Republic should leave the European Union."
  2. "Elektronická evidence tržeb by měla být zrušena." In English "Electronic records of sales should be cancelled."
  3. "Pacienti by měli mít možnost si připlatit za nadstandardní péči." In English "Patient should have the option to pay for extra care."
  4. "Studenti na veřejných vysokých školách by měli platit školné." In English "Public college students should pay tuition."
  5. "Česká republika by měla postavit další bloky jaderných elektráren." In English "The Czech Republic should build additional blocks for nuclear power plants."
  6. "Právo nosit zbraň by mělo být zaručeno v ústavě." In English "The law to carry weapon should be in the constitution."
  7. "Státní úředníci mají mít právo kontrolovat, čím doma topíte." In English "Government officials should have right to check what fuel you use at home for heating."
  8. "Ministerstvo vnitra má hlídat dezinformace na internetu a upozorňovat na ně." In English "Ministry of Interior should control disinformation on the internet and alert the public to it."
  9. "Česko by mělo co nejdříve přijmout euro." In English "Czechia should switch to euro as soon as possible."
  10. "Češi by si měli povinně spořit na důchod v soukromých fondech." In English "The Czechs should be required to save money for their retirement in private funds."

11. "Česko by mělo přijmout princip přerozdělování uprchlíků podle kvót." In English "Czechia should accept the princip of immigration quotas."
12. "Sankce proti Rusku by se měly zrušit." In English "Sanctions against Russia should be cancelled."
13. "Inkluze ve školách by se měla zrušit." In English "Inclusion in schools should be cancelled."
14. "Stát má regulovat služby sdílené ekonomiky (Airbnb, Uber)". In English "The government should regulate services of shared economy (Airbnb, Uber)."
15. "Instituce ombudsmana je zbytečná a měla by se zrušit." In English "The institution of ombudsman is unnecessary and should be cancelled."

All the question attributes have five categories. They are: "yes - important", "yes", "undecided", "no", "no - important".

For parallelogram clustering, I have used priority  $\vec{p} = (1, 10, 9, 8, 8, 6, 1, 5, 7, 10, 2, 3, 4, 4, 1, 2)$  for the nominal and ordinal attributes in the same order as they are shown at the start of this chapter. I have clustered dataset Elections three times with different thresholds - 0.1, 0.3, and 0.5. I have used the question "Q 1" as the primary attribute for the basis of clustering. Any of the questions could have been used depending on the interest of the user. However, I would advise against using the attribute "Party Name" as a primary attribute. Since all of its categories are unique, it would result in 26 clusters, and the outcome would be the same way as if we highlighted the attribute "Party Name".

Now let us have a look at Figure 4.11 clustered at the threshold of 0.1. We can see that each cluster represents only one political party, and aside from knowing each party's preferences, we do not learn anything of interest.

Let us rather have a look at Figure 4.12. This clustering was done at the threshold of 0.3, and it looks much better. Of special interest are clusters c1 in the dark blue, cluster c4 in yellow and cluster c0 in purple. These clusters combine political parties with very similar opinions, unlike the rest of the unassigned items that were too different from being assigned to a cluster.

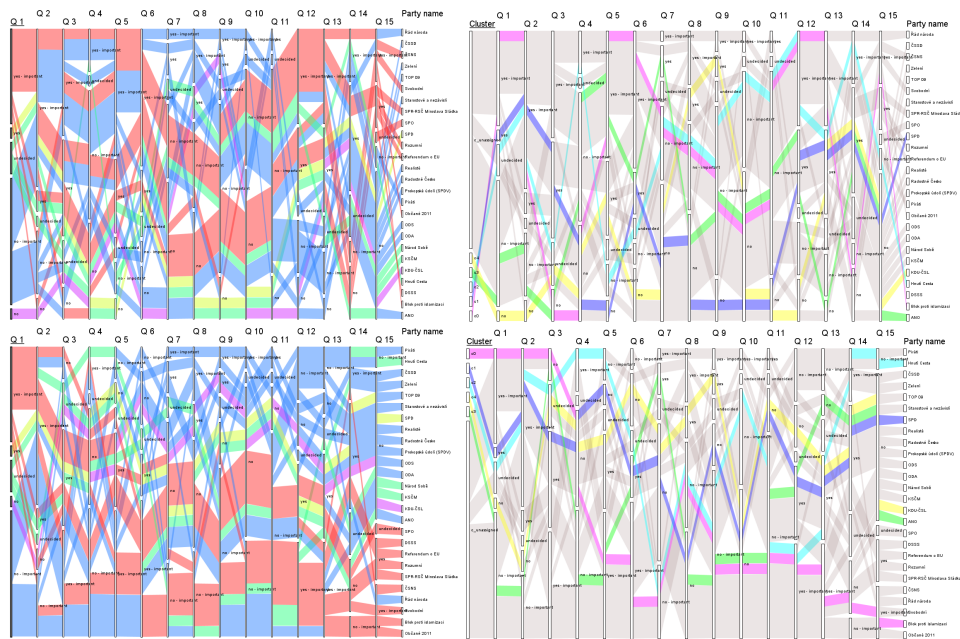
Especially the cluster c1 representing political parties "Rozumni", "SPR-RSČ Miroslava Sládka", "ČSNS", "Svobodní", "Blok proti islamizaci" and "Občané 2011".

When we compare these results to Figure 4.13 where threshold 0.5 was used, we can see another interesting aspect. Cluster c1 (in blue) remains mainly unchanged only with the addition of "ODS". This consistency shows that it is a very compact cluster. Even more cohesive was cluster c4 (in yellow) which was not changed at all.

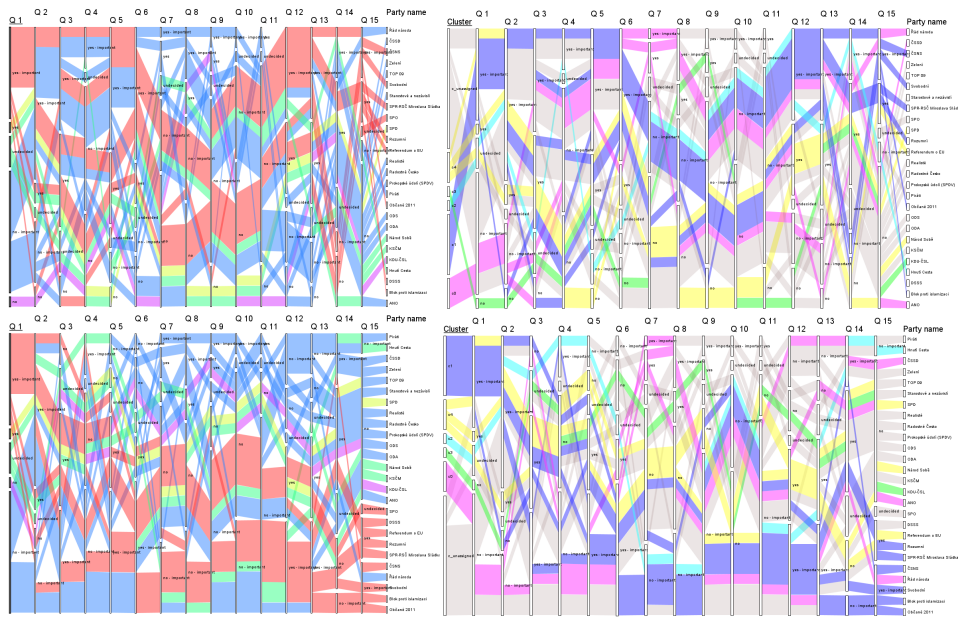
The remaining clusters increased in size, taking on other items. We can see, especially on the c0 cluster (in purple), that it became too muddled to give any helpful trend about to data.

Other interesting items are cluster c3 (in green) and c\_unassigned cluster (in gray). We can see that the political party "KDU-ČSL" is in some way unique as it did not cluster with any other item. Also, both of the political parties "Piráti" and "TOP 09" remained unassigned. Their being unassigned tells us that the parties are very different from the other parties based on the clustering criteria we selected.

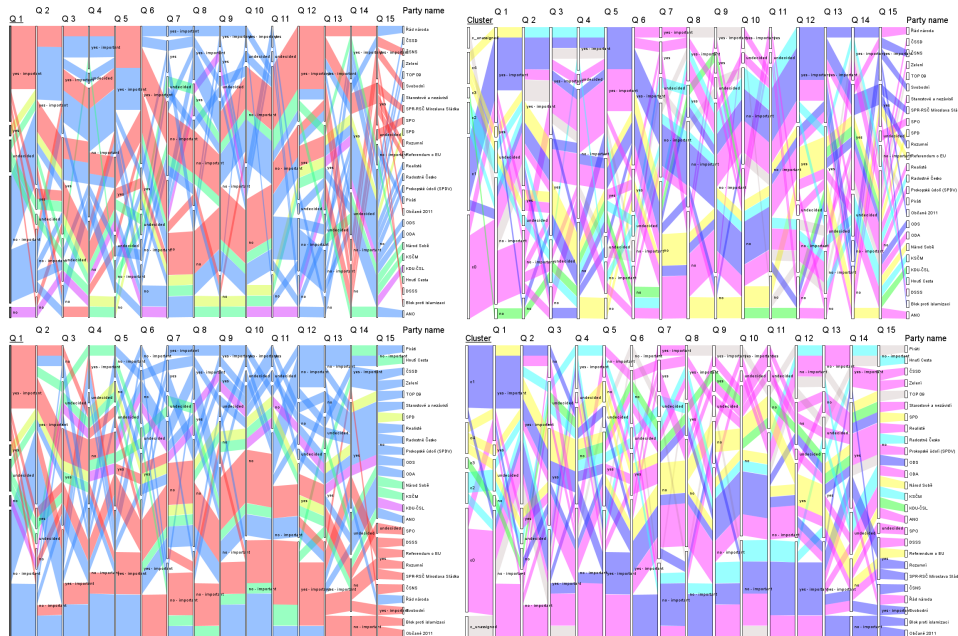
After taking a comprehensive look at the clustering of the Election dataset, I would say that the threshold of 0.3 is most valuable, especially when informed by the clustering at threshold 0.5.



**Figure 4.11:** Comparison for parallelogram clustering of the Election dataset with **threshold of 0.1**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).




**Figure 4.12:** Comparison for parallelogram clustering of the Election dataset with **threshold of 0.3**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).



**Figure 4.13:** Comparison for parallelogram clustering of the Election dataset with **threshold of 0.5**. Original dataset without change (top left), original dataset with reordered categories (bottom left), clustered dataset (top right), clustered dataset with reordered categories (bottom right).





## Chapter 5

### Conclusion

In my thesis, I have attempted to find a clustering method for nominal and ordinal datasets that adequately clusters the data. I have discussed multiple methods, finally ending on the parallelogram clustering method. I showed the results of the parallelogram clustering method in chapters 2.6.7 Parallelogram Clustering and 4 Experimental Results. Same as all other methods, parallelogram clustering methods have their positives and negatives. Based on the experimental result, I find the method to be useful. The method produces solid clusters if indeed clusters can be found in the given dataset. These results are partly by putting more responsibility on the user in the form of parameters that the method requires. In this way, I am a bit disappointed as I would have liked a single-click method with great results.

I have analyzed five different visualization methods finding, that the parallel sets method with the bundle layout method offers the best visualization for large high-dimension tabular datasets. Throughout my thesis, I have shown on figure after figure the visualization using the parallel sets method, and it brings evident results considering the complex nominal datasets it portrays.

Furthermore, I have tested seven clustering methods to find the best method for the clustering of nominal data. Through trying new methods and their testing, I have arrived at the parallelogram clustering method that utilizes the core principles of every nominal and ordinal dataset. Putting emphasis on the unique elements and letting the user take part in the burden, the resulting final method of parallelogram clustering worked well on four out of the five datasets I have tested on.

Further work is required, especially in selecting the primary attribute for the parallelogram clustering method and with increased automation of some tasks. Adding a secondary attribute to the primary attribute just as I have done in chapter 2.6.6 Two-Occurrence Method would result in greater granularity of the clusters. In this way, the smaller clusters could become more specialized. On the other, a large number of clusters could make the visualization unreadable.

The second field of interest is automation - mainly the automatic reordering of attribute axes and their categories to minimize the number of crossings of parallelograms in the final plot. This reordering would be a significant help for the user as it would significantly reduce the time needed to rearrange



the attributes in the best way. Still, it would not be a panacea as it is often needed to prioritize the position of essential attributes and deprioritize the position of others as the visualization task requires. This prioritization would ultimately have to be done either by the user or with the user's instruction.

I consider the parallelogram method as the main product of my thesis. Since the birth of the parallelogram clustering method would not be possible without the initial creation of the variants of the k-modes method, these variants are also an essential part of the final result. Without testing the parallelogram method on a large sample of different nominal or ordinal datasets, it is difficult to say how useful the method itself will be. Nevertheless, I have high hopes for its success.



## Appendix A

### Bibliography

- [AT07] S. Aranganayagi and K. Thangavel, *Clustering categorical data using silhouette coefficient as a relocating measure*, International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007), vol. 2, 2007, pp. 13–17.
- [Davne] J. Davies, *Parallel Sets*, Aug. 10, 2021 [Online].
- [EDF08] N. Elmqvist, P. Dragicevic, and J. Fekete, *Rolling the dice: Multi-dimensional visual exploration using scatterplot matrix navigation*, IEEE Transactions on Visualization and Computer Graphics **14** (2008), no. 6, 1539–1148.
- [Gow71] J. C. Gower, *A general coefficient of similarity and some of its properties*, Biometrics **27** (1971), no. 4, 857–871.
- [Hal20] B. Hall, *An intuitive approach to pca*, Jun 2020.
- [HWne] J. Heinrich and D. Weiskopf, *State of the art of parallel coordinates*, EUROGRAPHICS 2013/ M. Sbert, L. Szirmay-Kalos, 2021 [Online].
- [Jan21] M. Janda, *Visualization of n-dimensional heterogenous data*, Master’s thesis, Dept. of Computer Graphics and Interaction, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, CZ, 2021.
- [KBH06] R. Kosara, F. Bendix, and H. Hauser, *Parallel sets: interactive exploration and visual analysis of categorical data*, IEEE Transactions on Visualization and Computer Graphics **12** (2006), no. 4, 558–568.
- [Kosne] R. Kosara, *Parallel Coordinates*, Aug. 7, 2021 [Online].
- [Mad12] T. S. Madhulatha, *An overview on clustering methods*, CoRR **abs/1205.1117** (2012), 719–725.
- [MM15] T. Munzner and E. Maguire, *Visualization analysis amp; design*, CRC Press, 2015.

- [OCAD11] L. Oukhellou, E. Come, P. Aknin, and T. Denoeux, *Semi-supervised feature extraction using independent factor analysis*, 2011 10th International Conference on Machine Learning and Applications and Workshops, vol. 2, 2011, pp. 330–333.
- [S JL10] M. Shih, J. Jheng, and L. Lai, *A two-step method for clustering mixed categorical and numeric data*, Journal of Applied Science and Engineering **13** (2010), 11–19.
- [STHne] STHDA, *Scatter Plot Matrices - R Base Graphs*, Aug. 7, 2021 [Online].
- [The12] M. Theus, *Mosaic plots*, WIREs Computational Statistics **4** (2012), no. 2, 191–198.
- [UC ne] UC REGENTS, *What Is The Difference Between Categorical, Ordinal And Interval Variables?*, Aug. 8, 2021 [Online].
- [xdane] xdat.org, *XDat - A free parallel coordinates software*, Jul. 18, 2021 [Online].
- [XLQ12] X. Xuanhua, Z. Liyuan, and W. Qifeng, *A variation coefficient similarity measure and its application in emergency group decision-making*, Systems Engineering Procedia **5** (2012), 119–124.
- [Zhe98] H. Zhexue, *Extensions to the k-means algorithm for clustering large data sets with categorical values*, Data Mining and Knowledge Discovery **2** (1998), no. 3, 283–304.
- [ZSX06] H. Zengyou, D. Shengchun, and X. Xiaofei, *Approximation algorithms for k-modes clustering*, Computational Intelligence (Berlin, Heidelberg) (H. De-Shuang, L. Kang, and I. G. William, eds.), Springer Berlin Heidelberg, 2006, pp. 296–302.