UTRECHT UNIVERSITY

MASTER THESIS

Data-driven Diagnosis in Psychiatry

Author:
Wouter van der Klift

1st Supervisor:
Dr. M.R. Spruit

2nd Supervisor:
Dr. M.J.S. Brinkhuis

Daily Supervisor:
V.J. Menger

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Master of Business Informatics Research Unit
Department of Information & Computing Sciences

September 28, 2018

Abstract

Background: Diagnosing patients' mental disorders is based on symptoms in current society, causing high prevalence and co-morbidity in the field of mental healthcare. Improving treatment has called for adopting machine learning. One of these machine-learning techniques is cluster analysis, which is an essential technique in mental healthcare as it helps identify sub-types among mental disorders or groups of patients with higher symptom severity.

Problem: Researchers often face difficulties in choosing the appropriate clustering algorithm and, therefore, rely upon well-known clustering algorithms only, neglecting developments in other fields. Moreover, in mental-healthcare research, machine learning is often overlooked regarding applying it adequately. Recently, the cluster ensemble has been proposed and is actively used in the field of genomics. The ensemble displays promising results in robustness and novelty of finding clusters, outperforming standard well-optimized clustering algorithms. Even more importantly, the cluster ensemble alleviates the problem of choosing a clustering algorithm. However, this approach is relatively unknown in mental healthcare.

Objectives: A meta-algorithmic model (MAM) is developed and evaluated to mitigate both the problems researchers face in applying machine learning and choosing an appropriate clustering algorithm. The MAM is designed explicitly using the cluster ensemble. Furthermore, the cluster ensemble is examined using psychiatric data to find clusters based on severity of patients' symptoms during treatment, with the result evaluated by psychiatrists.

Method: First, we evaluated whether the cluster ensemble built by following our MAM would outperform a standard single clustering algorithm in an experimental setting using multiple datasets. Second, an exploratory case study was conducted in the Psychiatry Department of the UMCU. Data from DSM-IV diagnosed patients with schizophrenia or psychosis and assessed with the HoNOS were used. Twelve HoNOS features were included in the clustering process. What followed was visualization and identification of key features that define each cluster. Afterward, we interviewed three experts in the field of psychiatry to help explain our findings.

Results: We found that our MAM displayed increased performance over a standard clustering algorithm, with an average accuracy of 83% versus 75%. Then, we applied the same model to the data from the Psychiatry Department of the UMCU. These data included 744 patients, among whom the cluster ensemble found three clusters. Evaluation with experts resulted in the identification of a "severe cluster," "mild cluster," and "low problematic cluster," based on their underlying feature scores. The "severe cluster" clearly displayed high severity in the social and behavioral context of patients. For the "mild" and "low problematic" clusters, no clear separating features were found.

Conclusion: The MAM built for applying the cluster ensemble guides research further in mental healthcare to perform machine learning, and using, at the same time, a variety of clustering algorithms, invoking strong results. This approach may be extended to other domains as well. Finally, by finding clusters in psychiatry data, we have demonstrated that a certain group of patients can exhibit severe problems in their social environment that can be related to the severity of their positive and negative symptoms.

Acknowledgements

During my time at high school I got more and more intrigued by computers. Especially in how the components work and the Internet. After completing high school it was vocational school where my education began in IT, working my way upwards to the masters program Business Informatics at the University of Utrecht. This is where the journey ends for me as being a student. What a ride, and the things I have learned during this road is incredible. May we never stop learning.

First and foremost I would like to thank my daily supervisor Vincent Menger for his support and guidance during my thesis. The discussions and feedback have proven useful and provided me with some fruitful insights. Thank you for that, and the time you have spent to support me.

I also would like to thank everyone from the Psychiatry Department of the UMCU for their overall support. Especially Karin Hagoort and Floor Scheepers, who also made this thesis possible, and the several psychiatrists who have helped me to understand the outcome even better at the end.

Next, I would like to thank the following persons who have shown their guidance and support for my thesis:

- Dr. Marco Spruit, his role of first supervisor and bi-weekly contact person to shape the thesis by providing feedback
- Dr. Matthieu Brinkhuis, for his role as second supervisor, his support in some machine learning related questions and, without him knowing, being a wonderful teacher to motivate others to pursue statistics
- Prof. Dr. Peter van der Heijden and Dr. Ad Feelders for their guidance in some machine learning related questions
- Thomas Dedding, a good friend and fellow student. Thank you for the support and the good times we had at the Friday afternoon drinks. Cheers!
- Finally, my parents who supported me all the way from MBO to work my way up to a Masters degree. I cannot express in words how much this means to me.

Wouter van der Klift - September, 2018

Contents

List of Figures

List of Tables

Chapter 1

The transition to data-driven psychiatry

1.1 Introduction

General health issues, such as cancer, are remarkably well researched and diagnosed in today's society. Symptoms provide clues, while biomarkers, such as blood or genes, help draw conclusions. Thus, the combination of symptoms and biomarkers guides the clinical pathway to diagnosis, treatment and frequently to prognosis. In mental healthcare, however, *gold standards*, such as biomarkers, which articulate the type of mental illness a person is suffering, are absent (Bzdok and Meyer-Lindenberg, 2017), since there is limited evidence regarding the mixture of functions of brains, organs, genetics, and the environmental settings of mental disorders (Frances and Widiger, 2012; Huys, Maia, and Frank, 2016). Mental healthcare, therefore, continues to depend on a diagnostic approach solely based on symptoms (Kendell and Jablensky, 2003). Consequently, the field of mental healthcare faces high rates of co-morbidity and often ineffective treatment responses. (Cuthbert and Insel, 2013; Wigman et al., 2017).

The process of diagnosing patients is standardized in mental healthcare (Kendler, Zachar, and Craver, 2011) – using the Diagnostic Statistical Manual for Mental Disorders (DSM) (American Psychiatric Association, 2013). According to Frances and Widiger (2012), the DSM *"is a guide to psychiatric diagnosis - no more, no less."* Therefore, the DSM is regarded as an artifact, supporting the reliability, validity, and ultimately the decision-making process in diagnoses (Gara, Rosenberg, and Goldberg, 1992; Regier, Kuhl, and Kupfer, 2013). Thus, mental disorders are determined by the psychiatrist based on the symptoms a patient is displaying (Marquand et al., 2016).

Nevertheless, there are some detrimental effects to diagnosing solely on symptoms. A recent study by Boschloo et al. (2015) reveals that among the 12 major DSM disorders, several symptoms share a common denominator. The authors call this factor "bridge-connections," which are symptoms shared among disorders. As such, symptoms from mental disorders are not latent conditions (Fried et al., 2016; Os et al., 2013; Borsboom and Cramer, 2013). Contrarily, symptoms are a cause of a high prevalence of co-morbidity and heterogeneity among mental disorders (Cuthbert and Insel, 2013; Goekoop and Goekoop, 2016; Wigman et al., 2017). Therefore, the DSM is considered too rigid, leading to less specificity among mental disorders (Borsboom et al., 2011; Wigman et al., 2017; Krueger, 1999; Kendell and Jablensky, 2003; Hyman, 2010; Frances and Widiger, 2012; Cuthbert and Insel, 2013; Huys, Maia, and Frank, 2016).

Addressing these shortcomings, data-driven techniques, such as machine learning, have emerged in the field of mental healthcare. Machine learning can predict events or infer previously unknown structure from data, which are used to create new hypotheses (Iniesta, Stahl, and McGuffin, 2016; Passos et al., 2016). Essentially, machine learning can be used to create novel insights into mental disorders: for instance, to predict drug efficacy (Chekroud et al., 2016), depressive episodes (Loo et al., 2014; Wardenaar et al., 2014; Kessler et al., 2016), or suicide attempts (Hettige et al., 2017). Furthermore, machine learning is often used for the classification of patients and disorders (Passos et al., 2016; Gan et al., 2013), or to construct a new disorder taxonomy (Ross et al., 2015; Zhu et al., 2017). Finally, other data-driven techniques, such as network analysis, have been used to study the relationship and heterogeneity between the symptoms of various mental disorders (Boschloo et al., 2015; Fried et al., 2016; Borsboom et al., 2011; Goekoop and Goekoop, 2016; Borsboom and Cramer, 2013).

Another machine learning approach that received interest in the field of mental healthcare is cluster analysis (Everitt et al., 2011). Early studies that adopted cluster analysis studied the taxonomy of the DSM (Paykel, 1971; Everitt, Gourlay, and Kendell, 1971; Strauss, Bartko, and Carpenter, 1973; Gara, Rosenberg, and Goldberg, 1992). More recent studies used cluster analysis to find previously unknown sub-types in some well-established mental disorders (Prior et al., 1998; Lochner et al., 2008; Ross et al., 2015). Yet, only a few clustering algorithms —mainly hierarchical clustering— have been used in mental-healthcare studies (Loo et al., 2012). Meanwhile, other clustering algorithms might be more intuitive to use.

One problem with using cluster analysis is determining which algorithm to use, since each one is optimized for a certain data structure, meaning that results are variant to the algorithm in use, even on the same dataset. This specificity means researchers often do not know the appropriate clustering algorithm (Kuncheva and Hadjitodorov, 2004). In addition, healthcare studies in general do not adequately follow the necessary machine-learning steps to perform data analysis (Han et al., 2017), mainly because machine learning remains mostly limited to the computer-science community (Domingos (2012) in Spruit and Jagesar (2016)).

To address the first problem described above, the cluster ensemble offers an answer. The cluster ensemble is a novel approach for determining and finding the "right" number of clusters in a dataset. Recent studies by Iam-on, Boongoen and Garrett (2010), Yu, Li and You (2012) and Yu and You (2013) successfully applied the cluster ensemble to gene expression data by outperforming standard clustering techniques; that is, the identification of robust clusters while being more effective in handling noise within the data. This finding raises the question of whether mental healthcare can benefit from this approach also, alleviating the decision problem and simultaneously using multiple clustering algorithms.

For the second problem, *meta-algorithmic modeling* (MAM) offers an answer. First introduced by Spruit and Jagesar (2016), MAM is a generic method for building a transparent model that guides non-machine-learning experts into the field of machine learning. However, the model developed by Spruit and Jagesar (2016) is only suitable for the supervised domain in machine learning. Thus, the model is insufficient when dealing with the cluster ensemble, since this is an algorithm in the unsupervised domain in machine learning.

From a general point of view, one goal of this study is to guide researchers in mental healthcare to perform machine learning with the cluster ensemble, which requires the development of a new model for the unsupervised domain in machine learning. Second, there is scarce evidence concerning the effectiveness of the cluster ensemble in a mental-healthcare setting. As far as the authors are aware, there is currently only one study that adopted the cluster ensemble in a mental-healthcare setting. However, this study was limited to the autism spectrum, see Shen et al. (2007). Therefore, the aim is to guide and provide insights into the steps surrounding machine learning with the cluster ensemble and to experiment with the effectiveness of the cluster ensemble from a mental-healthcare perspective. To do this, we use data available from the Psychiatry Department of the University Medical Center of Utrecht (UMCU). The data consist of DSM-IV diagnosed patients who have schizophrenia or psychosis and who were assessed using the Health of the Nation Outcome Scale (HoNOS). The HoNOS routinely monitors the progress of patients' symptoms (e.g. cognitive functions and social functioning) and is used worldwide in the mental-healthcare sector. In our understanding, this study is unique in its efforts, since a novel MAM is developed and patients from the DSM-IV with schizophrenia or psychosis who have been monitored with the HoNOS have not been used (to this point) in a cluster ensemble setting.

1.2 Problem statement

In general, data-driven studies within mental healthcare do not follow a machine-learning approach (Han et al., 2017), remain limited to a single machine-learning algorithm (Shen et al., 2007), or face challenges in selecting a clustering algorithm (Kuncheva and Hadjitodorov, 2004) and report mixed results (Marquand et al., 2016). Consequently, there is a lack of knowledge that guides a researcher outside the computer-science community to perform robust machine learning, because knowledge regarding tuning algorithmic parameters is often insufficient (Spruit and Jagesar, 2016). With this in mind, there is a need to guide researchers in mental healthcare toward practical data analysis that does not adhere to a single “well-known” clustering algorithm, since novel insight into patient diagnostic data is paramount for the further development of psychiatry (Cuthbert and Insel, 2013). Therefore, the MAM was developed, which guides researchers in the domain of mental healthcare to perform effective data analysis using the cluster ensemble. Similarly, the cluster ensemble finds groupings in the data, but is capable of delivering equal or better results than a single clustering algorithm (Fred and Jain, 2002; Topchy, Jain, and Punch, 2003). Thus, more robust clusters can be found with new inferences to be drawn upon. Regarding precision in the field of psychiatry, Fernandes, Williams and Steiner (2017) state that , *"Precision psychiatry is to seek better lives for those suffering from mental illness by using tools capable of providing better and more accurate diagnosis, of ascertaining prognosis, guiding treatment and predicting response to treatment, and aiding the development of new and better pharmacological and non-pharmacological treatments."*

1.3 Research objectives

The focus in this study is twofold, as explained in the previous sections.

1. Development of a generic method that provides guidance to non-machine-learning experts to perform robust data analysis using the cluster ensemble in a healthcare setting
2. Deliver novel insights to extend the understanding of patients’ mental disorders based on both DSM-IV and HoNOS data

To this end, the two objectives in this study are as follows:

1. Describe and model a generic method for the cluster ensemble following MAM by Spruit and Jagesar (2016), which guides non-machine-learning experts
2. Utilize the cluster ensemble on the psychiatry data from the UMCU, consisting of DSM-IV diagnosed schizophrenic and psychosis patients, to find novel groupings based on features from the HoNOS

Describing and modeling the cluster ensemble with MAM is done using literature from the machine-learning community, the unsupervised domain in particular. Evaluating the cluster-ensemble meta-algorithmic model was done by following the steps described in the model and testing the cluster ensemble on several datasets. These tests were performed within R (RStudio, 2015), using the R package diceR (version 0.5.1) (Chiu and Talhouk, 2018).

For the data from the Psychiatry Department of the UMCU, we again used the cluster ensemble. The result was evaluated by experts within the field.

This study is limited to the diagnostic data (i.e. the DSM-IV and HoNOS) from mental-healthcare patients within the Psychiatry Department of the UMCU.

1.3.1 Research questions

Based on the study objectives, the following research question was derived:

"Which steps comprehend MAM for the cluster ensemble in the unsupervised domain, and to what extent can the cluster ensemble contribute to novel insights into mental disorders by utilizing and evaluating it to improve diagnosis and treatment that is in line with precision psychiatry?"

To answer the research question, and to guide the research objective, the following sub questions will be answered:

1. What are the steps to be described and modeled by a meta-algorithmic modeling method to perform data understanding, preparation, modeling, and evaluation of the cluster ensemble in an unsupervised domain?
   A step is made toward MAM to create an effortless and transparent model for non-machine-learning experts and healthcare researchers in this matter to understand the

cluster ensemble. This artifact serves to structure the steps needed to perform robust data analysis using machine-learning techniques. Subsequently, this model complements the MAM artifact for the supervised domain, as proposed in Spruit and Jagesar (2016).

Meta-algorithmic modeling is a method based on the principles of method engineering (ME), see Spruit and Jagesar (2016). Method engineering is defined as *"the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems."* (Brinkemper, 1996). The result is a meta-model (Weerd and Brinkkemper, 2009) based on a process-deliverable diagram (PDD). A PDD consists of a activity diagram on the left-hand side, and a class diagram on the right-hand side. Both diagrams are associated and connected to each deliverable in the process.

2. How accurate is the cluster ensemble compared with a standard clustering algorithm?; Using several real and synthetic datasets, the cluster ensemble is compared with a standard clustering algorithm. This comparison was done to measure the ensemble's accuracy, and therefore effectiveness, when clustering data.

3. Which number of clusters best describes the dataset according to internal index criteria, and which HoNOS features define each cluster? In general, the cluster ensemble is known for its robustness when compared with standard single clustering algorithms (Ghosh and Acharya, 2011). However, determining the optimum number of clusters still requires internal index criteria, since this addresses the accuracy of the ensemble (Jain, 2010). Second, knowing only the number of clusters is not useful; therefore, the aim is to determine which HoNOS features represent each individual cluster. This knowledge provides novel insights into how, based on the taxonomy of the DSM-IV, certain schizophrenia and psychosis patients are related to one another.

4. What do experts say about the clusters when evaluating them?
Expert evaluation is used to understand the outcome of the model; more specifically, to understand why certain disorders or features define each cluster, to ultimately label the clusters.

1.4 Research framework

To outline the steps taken in this research, a conceptual research model has been adopted from Verschuren, Doorewaard, and Mellion (2010) , and is tailored toward the needs of this project (see Figure 1.1).

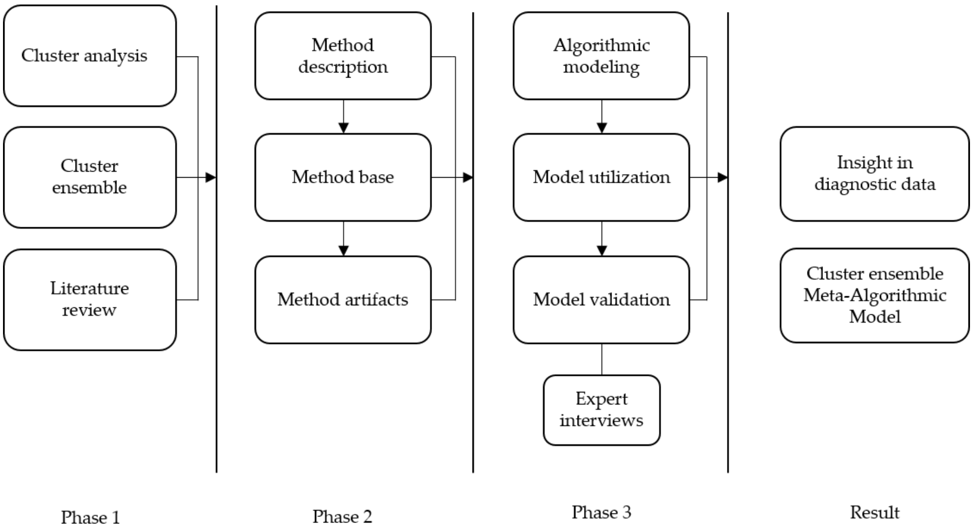


FIGURE 1.1: Conceptual research model showing research phases and end result.

Phase 1 represents preliminary steps that construct the base theory in this study. In essence, theory behind cluster analysis and the cluster ensemble sets the baseline for the construction of a generic cluster-ensemble meta-algorithmic model in healthcare. Thus, Phase 2 consists of describing and modeling the MAM method fragments necessary to perform data analysis. Finally, Phase 3 finishes the MAM model and begins modeling, utilizing, and evaluating cluster ensembles within the department of psychiatry as a case study.

1.4.1 Design Science Cycle

The cluster ensemble meta-algorithmic model serves the purpose to solve the problem in context, requiring a design science methodology. The paradigm in design science is the creation of scientific knowledge to solve the problem by constructional artifact development, with an emphasis on discovery-by-design (Baskerville, 2008). Therefore, the design-science-cycle, as stated in Wieringa (2014), was adopted. The design cycle consists of three consecutive tasks, beginning with problem investigation and moves toward treatment design and treatment validation (Wieringa, 2014). The aim of design science, therefore, is to follow a methodological approach in designing, evaluating, validating, and communicating the results of the artifacts created to solve the problem (Hevner et al., 2004). With this in mind, the research framework of this study is outlined and illustrated in Figure 1.2.

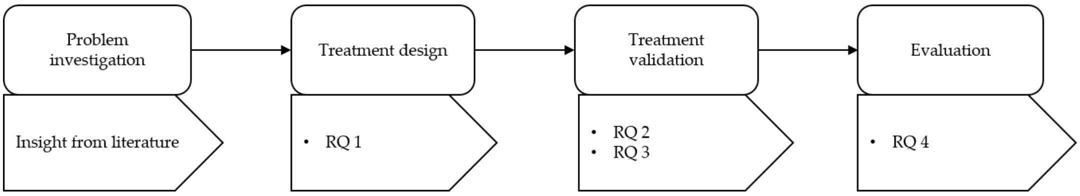


FIGURE 1.2: Design Science cycle, adopted and edited from Wieringa (2014).

Design science often begins with a problem-centered approach that aids in aligning the research questions with the design cycle (Peffers et al., 2007). Sub-question 1 (SQ1) is mandatory to gather insights into how a cluster ensemble works, and how the MAM is developed. Moreover, in the treatment design, several PDDs are described and amalgamated in a method base to develop the general algorithmic model. Second, the cluster ensemble was evaluated regarding its accuracy in several real and synthetic datasets, hence Sub-question 2 (SQ2). For the third sub-question (SQ3), the cluster ensemble was utilized and evaluated using the data from the Psychiatry Department of the UMCU. Finally, the fourth and final sub-question (SQ4) evaluated the cluster outcome from the previous research question with multiple experts from the Psychiatry Department of the UMCU.

1.5 Contribution to science and society

1.5.1 Scientific relevance

Multiple studies in the field of genomics empirically proved that the cluster ensemble is a relatively new and robust technique for class discovery (Iam-On et al., 2010; Yu et al., 2012; Yu et al., 2013). However, as far as the authors are aware, the cluster ensemble has only once been examined in the field of mental healthcare. Thus, this technique requires more experimental research to determine whether it is suitable for the domain of mental healthcare. Moreover, schizophrenia and psychotic patients with a DSM-IV diagnosis are unique in terms of experimental research, as most studies have focused on the pervasive development disorder spectrum (Shen et al., 2007). Finding groups with both DSM-IV and HoNOS data is, therefore, novel in its own way. The outcome can provide insights into how HoNOS features characterize the clusters for these DSM-IV-diagnosed patients, which is unique, as there is not a single study that has focused on this aspect.

Simultaneously, this study continues upon the development of the MAM framework, as proposed in Spruit and Jagesar (2016), by extending the framework towards the unsupervised field within machine learning, and in particular with a cluster-ensemble technique.

This study supports communities other than computer science to adopt machine learning in a correct manner, mitigating the difficulties of performing and understanding data analysis. Hence, the outcomes of both the MAM and cluster ensemble can provide meaningful ways for the field of mental healthcare to further refine precision psychiatry for the well-being of future patients.

1.5.2 Societal relevance

Providing MAM that is transparent allows researchers in healthcare to perform machine learning in the unsupervised domain. Subsequently, adopting the cluster ensemble can create novel insights. By evaluating both an MAM and an cluster ensemble in this case study demonstrates that psychiatry can build novel insights into their dataset. In turn, experts can evaluate how several features from the HoNOS determine patient groupings. In other words, psychiatrists can have better insights into the mental disorder and can work together toward a more personalized treatment.

Chapter 2

Cluster analysis and the cluster ensemble: A machine learning approach

2.1 The notion of machine learning

Many fields generate vast amounts of data in our daily lives. Healthcare is not an exception in this matter. After all, data is extracted from MRI scans, DNA, clinicians notes, the electronic health record, and so forth, requiring new ways of analyzing that data effective, and preferably at an individual level. This is where machine learning comes into play. Machine learning leads to new ways of analyzing the data, generating new knowledge and hypotheses, and finds beforehand unknown patterns within the data (Murphy, 2012). As such, research is more data-driven, and acts as a plateau that strengthens the outcome by involving both domain experts and machine-learning staff (Menger et al., 2016).

Cluster analysis finds patterns within the data. In the machine-learning domain cluster analysis operates at an unsupervised level. However, to understand the unsupervised level, the supervised level requires some explanation. Therefore, this chapter is devoted to begin with briefly explaining the differences between supervised and unsupervised learning. Then, starting from section 2.2, the overarching goal of cluster analysis is described, accompanied with its techniques, represented as the 'big-five'. In Section 2.3 the cluster ensemble is introduced. Finally, section 2.4 ends this chapter by providing technical insight in how the cluster ensemble works.

2.1.1 Supervised learning

Supervised learning is in general known for its predictive approach. Specifically, supervised learning deals with labels in the input and output data. Both known as predictor variables and response variables. The predictor variables consists of measurable predictor observations, for example, recorded blood sugar levels of diabetic patients. Gathering these observations results in a list, often denoted as: $\mathbf{x}_i$, i = 1. . .n, where i is a single observation. In turn, this list of observations is linked to an associated response variable, denoted as y_i. In a formal manner: based on a dataset D, predict the outcome of y_i using the observations $\mathbf{x}_i$. Thus, the ultimate goal of supervised learning is to predict future outcomes based on historic observations (James et al., 2013).

Since prediction is the primary goal of supervised learning, it deals with function approximation (Murphy, 2012). That is, the accuracy, or error, of the predicted outcome is estimated, known as the unknown function f. Thus, the most accurate prediction given a labeled dataset D, results in $\hat{Y} = \hat{f}(\mathrm{x})$. This simple formula states that estimating the most accurate prediction always comes with errors, and some bias. The prediction, therefore, will not be 100% accurate, and will contain some assumptions, hence the hat symbols on top of Y and f.

To elaborate on these errors that affect f, the accuracy of outcome $\hat{Y}$ depends on two error factors. These are the reducible error, and the irreducible error. As its name suggests, the reducible error can be improved. For instance, by applying the most appropriate supervised machine learning algorithm for the dataset at hand. However, the irreducible error is always present, and, therefore, cannot be reduced. In fact, every dataset will miss certain predictor variables that are needed to perfectly predict the response variable. For instance, to predict Y, the model will always have certain predictors, but at the same time the model will also lack certain predictors that influence Y. Hence the presence of the irreducible error. For detailed formalities of supervised learning, see James et al. (2013) and Murphy (2012).

Although supervised learning is one of the *"simpler"* methods —in the case of reducing as much as bias and error in the model— in machine learning, collecting labeled data is often a tedious task. Therefore, unsupervised learning can be applied first. For instance, to determine groups within the data, or to analyze the data preparatory to assign labels to it.

2.1.2 Unsupervised learning

Unsupervised learning does not deal with response variables. The data present in unsupervised learning is merely the predictor variables, similarly also its only output. This makes unsupervised learning more a method for class discovery than prediction, since it has no outcome to predict. Therefore, knowledge discovery is often the case in unsupervised learning (Murphy, 2012). Sometimes it is the first step in machine learning in order to get a global understanding of the data.

Since unsupervised learning is aimed at class discovery, a natural question that rises is the approximation of accuracy. The simplest answer is; *it depends on the algorithm and data set*. As we will see later on in the chapters, there is not a single *gold standard* for both the algorithm and the accuracy parameter. Thus, depending on the algorithm taken from the unsupervised domain, different validity measurements exists (Murphy, 2012), making unsupervised learning often subjective, since various algorithms from a single analogy have different assumptions about the data.

Two well-known unsupervised learning methods are principal components analysis (PCA) and cluster analysis (Iniesta, Stahl, and McGuffin, 2016). PCA seeks to reduce dimensionality in the data, in order to only select the most promising predictor variables. Last, clustering is sometimes followed after PCA to find groups in the data. Especially, when dimensionality poses a problem for clustering.

2.2 Cluster analysis techniques

The goal of cluster analysis is to discover groups within the data and is best described by Kaufman and Rousseeuw (1990): *"cluster analysis is the classification of objects into groups which share similarity among each other and impose a structure within the data, even if the structure is not directly present."* In essence, clustering creates patterns of distinct groups in a dataset by using observations that share similarity. Even if observations do not share direct similarities and the data does not contain a 'natural' cluster structure, even then a clustering algorithm often finds clusters (Tan, Steinbach, and Kumar, 2005). As such, cluster analysis proves useful for exploratory data analysis in the unsupervised domain. Making it a favored technique in psychology to refine or redefine current diagnostic criteria (Everitt et al., 2011).

Granted that clusters are to be found in a dataset, straightforward applying a clustering algorithm often results in inconsistent outcomes. To clarify, the outcome relies on the parameter settings of the algorithm, the proximity metric being used that defines (dis)similarity between observations and the order and amount in which observations or variables are presented (Jain, Murty, and Flynn, 1999). Moreover, there is no clear ground in what makes up a 'good' clustering because it often violates scale-invariance (proximity metrics that alter similarity between observations), richness (possibility to relate proximities in a matrix without proximity metrics) and consistency (shrinking similarities in one cluster and widening similarities in other clusters) that affect the end result (Kleinberg, 2002). Therefore, there is not a single clustering algorithm that is gold standard, since every algorithm brings assumptions. This is also known as the *no free lunch* principle in statistics, which in turn drives the development of new clustering algorithms when the established ones do not suit the needs of the researcher (Fred and Jain, 2002).

Choosing the appropriate clustering algorithm requires some basic understanding of its technique and often some domain knowledge. Therefore the 'big five' of clustering algorithms is discussed from a machine learning perspective as described by Zhou (2012). As such, a deviation is made from the groundwork by Jain, Murty and Flynn (1999) in a sense of *cluster analysis approaches*, since their work is based on the clustering methods by Jain and Dubes (1988). Meanwhile new clustering algorithms have been proposed since then.

2.2.1 Hierarchical methods

Hierarchical clustering forms a tree, with edges and nodes, called a dendrogram, that impose a clustering structure (a hierarchy) on the dataset (Zhou, 2012). Forming a hierarchy can be based on a top-down (divisive) or bottom-up (agglomerative) manner (Jain, Murty, and Flynn, 1999). The divisive method begins with placing the data in one big cluster and recursively splits the big cluster into smaller nested sub-clusters. That is, observations that are the most heterogeneous in a sub-cluster are split again until a hierarchy is formed with clusters of observations that are closely related to each other. In an agglomerative setting the method starts with placing each observation as a cluster on its own and iteratively starts merging these singleton clusters into bigger clusters. It stops when clusters are formed of homogeneous similarity. We will now discuss these two methods separately.

Agglomerative

In order to form clusters and determining their hierarchical order in a hierarchical agglomerative setting, several linkage methods can be used to determine the dissimilarity between the clusters found in a dataset. Here, five well-known agglomerative linkage methods are discussed, each having a distinct character to form the hierarchy of linking clusters. For the ease of reading we omit their mathematical proofs and refer to Murphy (2012) .

Single-linkage, also known as nearest neighbor, computes all pairwise dissimilarities of observations between the clusters and selects the *minimal* distance between the clusters (James et al., 2013). For instance, single-linkage starts with each observation as its own cluster and starts forming clusters in a chainlike fashion when pairs of observations are close to each other, nesting observations into clusters and linking clusters in the dendrogram based on the minimal distance (Jain, Murty, and Flynn, 1999; Murphy, 2012), see Figure 2.1(A).

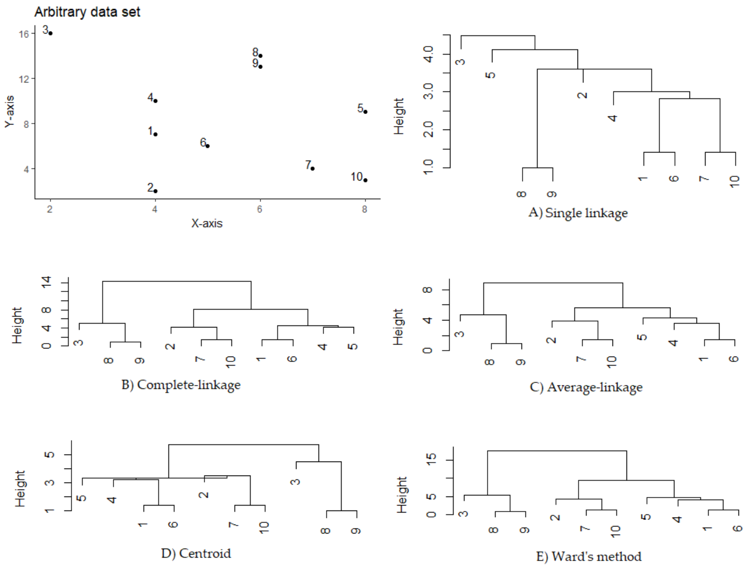
Complete-linkage performs cluster linkage in the opposite direction by taking the *maximum* distance between separate clusters. Picking observations that are both farthest away. In a somewhat similar fashion as single-linkage, complete-linkage nests observations and links clusters in the dendrogram based on minimal distance. However, by taking the maximum distance to form the linkage its end result differs from single-linkage, an example is given in Figure 2.1(B).

Average-linkage can be seen as the middle ground between the two previously mentioned linkage methods. Average-linkage computes the *mean* to separate clusters. Each observation within a cluster has an average pairwise distance to other clusters (Murphy, 2012). This results in a linkage of the dendrogram where clusters' average distance is equal to the average distance between observations in separate clusters (Rokach and Maimon, 2005), see Figure 2.1(C).

Moreover, there is centroid-linkage. Centroid-linkage creates clusters based on the *centroid mean* between pairs of observations. Hence, both means of a pair of observations is used to compute their centroid (middle ground). The linkage is done by computing centroids which are close to each other, see Figure 2.1(D).

To end the linkage types we close with Ward's method. With Ward's method clusters are created by computing the aggregate sum of squared error (SSE) between separate clusters. Linkage is similar fashion to complete-linkage, by taking the minimum SSE between cluster pairs. To illustrate, the clusters within the model are artificially merged and the resulting deviations are computed of the merged clusters. By taking the minimum deviation between a merged cluster pair, the clusters are linked in the dendrogram (Tan, Steinbach, and Kumar, 2005), see Figure 2.1(E) for Ward's method.

In general, average- and complete-linkage often yield balanced dendrograms with more useful hierarchies over single-linkage, because single-linkage often violates compactness of clusters by its chaining approach and is susceptible to noise (Murphy, 2012; James et al., 2013). However, complete-linkage is less versatile than single-linkage as it cannot identify concentric clusters (Jain, Murty, and Flynn, 1999). Moreover, Ward's method and centroid-linkage are robust techniques when it comes to multivariate distributions in a dataset, but



A) Single linkage

B) Complete-linkage

C) Average-linkage

D) Centroid

E) Ward's method

FIGURE 2.1: Various agglomerative linkage types showing the tree forming process of hierarchical clustering.

have difficulties when there are unequal groups and when merging of clusters occur the new created clusters may be more similar, known as inversion (Tan, Steinbach, and Kumar, 2005).

Divisive

In divisive hierarchical clustering the dendrogram is broken down into smaller clusters. This approach is in fact time consuming and is cumbersome, since it has difficulties to find the best split of the data set (Murphy, 2012). Therefore, some metrics can be used to efficiently build the dendrogram in a divisive (splitting) manner. Three of these methods are discussed.

Bisecting K-means can be used to split the initial cluster into two sub-clusters, and recursively applies this step until a satisfied split has been achieved, known as a stopping rule (Murphy, 2012). The split can be based on the diameter of the clusters, by picking the largest one and continue the split, or by dissimilarity of observations, that is splitting based on the farthest distance between two observations. Another method is by using a minimum spanning tree (MST) on the dissimilarity graph created from the dataset. This approach can be reflected to the previously mentioned methods in the agglomerative setting, by identifying the minimum distance or farthest distance between observations and splitting these into clusters. Thereon, the MST is used to cut links in the hierarchy where the sum of dissimilarity is the highest. Thus, it creates a hierarchy by linking all the clusters by their minimum dissimilarity. Last, dissimilarity analysis, also known as DIANA by Kaufman and Rousseeuw (1990), is used to split the clusters based on the average dissimilarity distance between observations in one cluster. To illustrate, by starting with one big cluster G, an observation i_j is moved to maximize the dissimilarity to each other observation i_n in G. Subsequently, the distance of the moved observation i_j is minimized with the other observations i_m in the new formed cluster H. This process is recursive and can be stopped when every observation i is a singleton cluster (Murphy, 2012).

Despite the appealing visualization approach of hierarchical clustering, it is not an efficient method when handling large datasets. In other words, it can be a slow cluster analysis technique to use and may require a high amount of computing power, because for every split or merge additional new observations are to be computed. Furthermore, backtracking is not allowed, causing the possibility that if a merge or split happened with a high degree of error this cannot be undone (Rokach and Maimon, 2005). Last, hierarchical clustering often introduces bias, as assumptions are made on the amount of clusters (Monti et al., 2003).

2.2.2 Partitioning methods

In contrast to hierarchical methods, partitioning methods do not impose a structure of nested clusters on the data (Jain, Murty, and Flynn, 1999). Instead, partitioning methods create a flat partition of the data in an attempt to discover clusters. To discover clusters within the data, partitioning methods often require a fixed number of clusters K to be set a priori (Steinbach, Ertöz, and Kumar, 2004). Consequently, the aim of partitioning methods can be expressed as followed: Based on observations n in a data set D, determine to partition the dataset into a fixed number of K clusters, such that n observations in cluster 1 are more similar to each other than observations in cluster 2 (Jain, 2010).

To illustrate the objective of a partitioning clustering method K-means is taken as an example because it is well-known throughout the literature (Jain, Murty, and Flynn, 1999; Rokach and Maimon, 2005; Zhou, 2012). K-means is a center-based algorithm, meaning that it tries to find a local optimum for each cluster. It requires that an observation or cluster center value is picked, often randomly, and will act as the centroid of neighboring observations. Thus, each observation n_j close to its centroid is placed within that cluster and observations can only be assigned to one cluster at time —this is known as hard clustering— (James et al., 2013). In practice, often multiple observations are assigned as a centroid. Therefore, observations are split, meanwhile multiple clusters are created with observations that are close to their centroid and farther from other centroids. Because the centroids can be picked at random K-means often requires random restarts to assign randomly new centroids in order to find the lowest sum of squared errors (SSE) in the outcome, equivalently the lowest within-cluster variation (Jain, Murty, and Flynn, 1999). For in-depth details of K-means see Tan, Steinbach and Kumar (2005) .

Since partitioning methods often require a fixed number of k clusters to be set a priori, it is subject to trivial clustering. That is to say, an increase in k will always lead to an improvement in the models' performance (Rokach and Maimon, 2005). Fortunately, several methods exist that can help in determining the optimal number of k. Some of these methods are known as Bayesian Criteria Information (BIC) or Akaike Information Criterion (AIC), which computes the outcomes of different k to a number of m models and selects the model with the lowest error. See James et al. (2013) for in-depth information.

2.2.3 Density-based methods

Next to creating clusters based on similarity in their distance, clusters can also be created based on their density. Density-based methods act almost similar to partitioning methods by forming clusters based on the distance to their centroid. Creating clusters in density methods is based on the radius (or epsilon) of the centroid and the amount of observations needed to form a dense region. To illustrate, centroid x_i has a fixed radius of ϵ and a parameter of $MinPts$ needed to form a cluster. Thus, density-based methods depend on two parameters: the radius (its neighborhood size around the centroid) ϵ and a minimum threshold of $MinPts$ in its neighborhood to form a cluster (Tan, Steinbach, and Kumar, 2005; Jain, 2010). As a result, regions with high-density are regarded as clusters which are separated by regions with low-density (Zhou, 2012).

An intuitive question that follows is to determine the radius of ϵ and $MinPts$ to perform density-based clustering. From a pragmatic viewpoint, $MinPts$ should always be >2, since using the number 2 creates the same effect as single-linkage, creating unnecessary clusters. Moreover, using $MinPts = 1$ will simply put every observation in a singleton cluster. Thus, experimenting with $MinPts$ is necessary to find the optimal result. Determining the ϵ is more straightforward. To determine the optimum radius of ϵ, the $MinPts$ comes again into play. By using k-Nearest Neighbors (kNN) and setting its value of k to act as $MinPts$ can help determine which value of ϵ is most optimal for a dataset.

DBSCAN, as proposed by Ester et al., (1996) , is a well-known density-based clustering algorithm that searches for density regions within a feature space in a non-parametric fashion. It handles density regions based on the Parzen window method. The Parzen window method basically estimates whether an object falls within the high-density region. Hence, the objects are considered as core-objects, border-objects or as noise (Tan, Steinbach, and Kumar, 2005). Core-objects are considered to handle ϵ distance and $MinPts$ to form a cluster, whereas border-objects can fall within a variable region of several core-objects and noise is simply eliminated. The algorithm handles the data in five steps, as described by Tan, Steinbach and Kumar (2005), and is illustrated in Figure 2.2:

1. Label all objects as core, border, or noise;
2. Eliminate noise-objects;
3. Put an edge between all core-objects that are within e distance of each other;
4. Form density regions from core-objects; and
5. Assign each border-object to one of the associated core-object clusters.

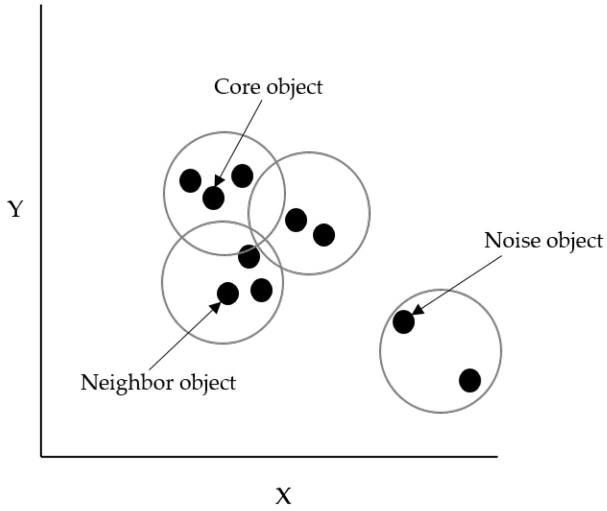


FIGURE 2.2: DBSCAN with core, border, and noise objects.

Since clusters are defined based on their density, density-based methods are well suited in handling noise, outliers and form clusters of arbitrary shapes and sizes (Tan, Steinbach, and Kumar, 2005). However, high-dimensionality poses a problem as it makes it difficult to distinguish regions of cluster between high-density and low-density (Jain, 2010). Yet CLIQUE, proposed by Agrawal et al. (1998) , overcomes the high dimensionality problem.

2.2.4 Grid-based methods

So far the previous methods apply structure to the data as in hierarchical methods or use a flat partition as in partitioning methods or density-based methods. Grid-based methods on the other hand form a so-called rectangular flat map, where data is formed onto a finite number of cells forming a grid structure (Zhou, 2012). To illustrate, each cell within a grid structure stores a number of observations and then form clusters from the cells in the grid structure (Cheng, Wang, and Batista, 2013). Particularly, each cell consists of multiple layers that stores granular information about the observations —such as the mean or standard deviation in the STING algorithm proposed by Wang, Yang and Muntz (1997)—, which can be inferred from the highest layer. Consequently, a grid mesh is obtained that allows to cluster cells instead of clustering observations from a dataset directly, allowing to efficiently handle large datasets, noise and outliers (Liao, Liu, and Choudhary, 2004). To better understand the grid-based clustering method, CLIQUE, as proposed by Agrawal et al. (1998) is taken as an example and is discussed along with OptiGrid by Hinneburg and Keim (1999) and WaveCluster by Sheikholeslami, Chatterjee and Zhang (1998).

CLIQUE is designed to automatically find subspaces in a high dimensional dataset to improve clustering results (Agrawal et al., 1998). Compared to other clustering algorithms, such as DBSCAN and K-means, CLIQUE uses subspaces in the data, as this is where new clusters can be found. Thus, by using subspaces only, more comprehensible clusters are to be found. These subspaces are found by partitioning the dataset into a number of k cells. In other words, the number of density regions found will be placed over a number of k cells of equal length, so that the cells will represent regions of equal density (Agrawal et al., 1998). As a result, high dimensional datasets can efficiently be used for clustering, because a few cells will represent the whole dataset (k « N). Although CLIQUE uses the definition of density to form its cells, it requires a threshold to define density (τ) and a defined number of units (ε) to create its cells. Subsequently, it automatically finds the clusters in the subspaces without user-defined cluster settings. The steps of the algorithm, therefore, can be summarized as follows:

1. Partition the data set into subspaces with non-overlapping rectangular cells containing attribute values of observations, by defining ε;

2. Identify dense cells based on the density parameter τ;

3. Form clusters from dense cells;

4. Arbitrarily start with a dense cluster, and find maximal connected regions with other dense cells in all dimensions; and

5. Repeat 4 until all cells are covered and generate minimal description for the cluster.

OptiGrid, proposed by Hinneburg and Keim (1999) overcomes a cluster pruning problem that is concentrated within CLIQUE. Cluster pruning can be seen as overlooking clusters in the created subspaces, as high dimensionality can cause a single observation belonging to just one dimension. Hence, there is sparse density and no dense cells. OptiGrid overcomes this problem by omitting step 3 from the CLIQUE technique by first starting with cutting the grid matrix into hyperplanes. By first cutting the grid matrix low density regions become separated from dense regions. As a result clusters cannot split, meanwhile separating sparse regions that are equal from dense regions. Second, by cutting the matrix, clusters are clearly identified because areas of high density are better preserved (Hinneburg and Keim, 1999).

WaveCluster is the last example in the grid-based methods that allows finding arbitrary shaped and nested clusters in large datasets. This is possible because WaveCluster uses wavelet transformation of observations. What it basically does is transforming the data in high frequency and low frequency profiles. The low frequency profiles are dense regions, while the high frequency profiles are sparse regions. This allows that clusters are more distinct and can be displayed at different levels of detail, from fine to coarse and regions of sparse density are easily identified and removed (Sheikholeslami, Chatterjee, and Zhang, 1998).

Grid-based methods often use properties from density-based methods by applying density to their cells to form clusters. However, as this method is efficient for large datasets it is still susceptible to non-uniformity of the distributions, posing difficulties for clustering quality. In addition, the parameters are sensitive to the curse of dimensionality as it affects the grid matrix, cell density and thus the formation of the desired clusters (Gan, Ma, and Wu, 2007).

2.2.5 Model-based methods

The last type of method discussed here is model-based clustering, which is a derivative from probabilistic mixture-modeling (Murphy, 2012). Model-based methods are known as *soft* clustering, where objects have a probability in belonging to several clusters. This is accomplished by forming clusters k on the basis of several distributions, such as Gaussian or Bernoulli. The underlying assumption is that clusters can be formed based on the distribution within a dataset. This allows to automatically determine the number of clusters based on standard statistics (Fahad et al., 2014). Therefore, model-based clustering can be presented as a Bayesian approach, since any probability model is inferred from the data in order to optimize the fit of clusters (Zhou, 2012). In turn, model-based methods are different from the methods previously discussed, since these do not use a probabilistic function.

In model-based clustering clusters are formed based on the underlying information residing within the data. In detail, it is assumed that different distributions are within a dataset and that each distribution represents a cluster (Fraley and Raftery, 2006). Thus, clusters can be derived by identifying the amount of distributions within a dataset. To determine these clusters and estimating the probability that an observation belongs to a cluster, the steps are as follows:

1. Place objects based on their distributions to an user unknown cluster;
2. Calculate the probability of objects with the same distribution to form a cluster;
3. Determine the cluster parameters based on the objects within; and
4. Find and calculate the probability that an object belongs to a cluster.

Although these steps are somewhat similar to K-Means, it remains rather unknown whether objects in a K-Means cluster are clustered accordingly, since the centroids often regroup to fit the data, but do not address the number of clusters within the data (Fraley and Raftery, 2006). Therefore, some strengths of applying a model-based method is to mathematically resolve the number of clusters. Consequently, as some of the previously mentioned methods are susceptible to the way data is presented, model-based methods overcome this problem by taking the distributions within the dataset. Using these distributions, model-based methods are capable to deal with heterogeneous cluster types, as observations can fall into multiple categories if features overlap. In addition, by employing statistical characteristics to discover clusters, model-based methods are less susceptible to subjective matters. In other words, there is a bias variance trade-off because it penalizes complex models with many clusters (Mun et al., 2008). To conclude, model-based methods explicitly focus on finding clusters among distributions in a dataset and assigning unobserved heterogeneous observations based on the highest probability to a certain cluster.

2.3 Introduction to the cluster ensemble

The previous section described various methods to perform cluster analysis, from a visual type dendrogram as in hierarchical clustering, to a probabilistic type as in model-based methods. Yet, not one method outperforms the other, since not one method is suited for all kinds of datasets. Thus, each method will result in different outcomes and without a priori knowledge it is hard to determine its validity of the true natural clusters obtained. Therefore, deciding on which cluster algorithm to use is key and requires expertise and insight of the data. However, only choosing the 'right' clustering algorithm is not the sole solution for recovering the natural clusters in the data. Optimizing the heuristics of such an algorithm, i.e. different parameter settings and several random restarts, are often necessary in order to get the most optimal result. Furthermore, even when the most appropriate clustering algorithm is used, spurious results are sometimes obtained even when there is no natural grouping present in the data (Jain and Law, 2005). This opens the debate whether clustering is an appropriate method to use in the first place. Turning to the supervised learning setting several things are less complicated, as there is a label output to compare with. In addition, techniques such as bagging, boosting and ensembles are well-understood. In the case of ensembles, outcomes of several supervised algorithms are used to optimize the end result with smaller error rates, according to Kittler et al., (1998) as cited in Fred and Jain, 2002. However, since data is more often unlabeled than labeled and when more insight is needed in the data, one can turn to the cluster ensemble approach that alleviates some of the pitfalls in clustering (Jain and Law, 2005).

In this section, the cluster ensemble is discussed following its philosophy and important work.

2.3.1 Philosophy behind the cluster ensemble

The philosophy behind the cluster ensemble is derived from the classifier combination in the supervised learning area (Fred and Jain, 2002). The basic idea is that the cluster ensemble holds a portfolio of various cluster partitions, which combined will result in a final clustering that encompasses all the information gathered (Fern and Lin, 2008). Thus, the aim of the cluster ensemble is to combine the strengths of various cluster partitions in order to achieve better results (Ghaemi et al., 2009). This motivates the key idea behind the cluster ensemble, such as improved clustering by diversity, better robustness of clustering by compensating in inherent randomness by several clustering algorithms and knowledge reuse as the cluster ensemble may hold legacy clusterings of data that can be reused in new training sets (Zhou, 2012).

2.3.2 Brief history of important work with the cluster ensemble

Cluster ensemble, consensus clustering, cluster fusion, or evidence accumulation clustering are various synonyms that all refer to the same idea; a combination of various cluster outcomes to leverage consensus across various clustering algorithms to combine into a single consensus solution (Fred and Jain, 2002; Strehl and Ghosh, 2002; Topchy, Jain, and Punch, 2003).

Fred and Jain (2002) explored the idea of combining results from multiple clusterings into a final solution, the *consensus solution*, that represents the true natural clusters within the data. This is based on a split-combine-and-merge strategy, emphasized as the evidence accumulation-based clustering in their study. The splitting process involves transposing a multi-dimensional dataset into many low-dimensional subspaces. The combine process is used to gather the results and combine clusters that have objects in the same clusters, likely to represent the true natural clusters. The final step involves merging similar clusters together from the splitting process and splitting the clusters from the combine step to preserve the true natural clusters. What stands out in their study is that a simple k-means is used to recover the natural clusters. By transposing various datasets into many subspaces and by varying in parameter settings and random initializations, the k-means algorithm was capable of identifying non-spherical clusters. This is exceptional as k-means cannot work with non-convex clusters.

Similar work by Topchy, Jain and Punch (2003) explored the idea of combining multiple weak clustering algorithms, in order to achieve comparable or improved performance over a single optimized algorithm. Their aim was whether diversity —different views on the data— would be a contributing factor to retrieve the true natural clusters. In addition, by knowing the true natural clusters beforehand a misalignment error rate was calculated, determining the contribution of diversity in finding clusters. Diversity is best explained by transposing a dataset into many lower 1-*d* subspaces and by splitting the subspaces with random hyperplanes. These two transposing methods clearly weakens the partitions, as the original data space is fragmented. However, by combination it is assumed that these weak partitions can be used to cluster the data at least as good as a single optimized clustering algorithm in original data space. Furthermore, it can even reveal a hidden structure that is unattainable for the single algorithm (Topchy, Jain, and Punch, 2003). Thus, many random subspaces set out for a broader view on the data. By employing only the k-means algorithm the authors prove that three parameters give optimal clusterings, namely the: 1. the number of combined outcomes, 2. the number of clusters specified in the parameter settings, and 3. the number of hyperplanes used to obtain the clusters. In sum, it is best to use a variety of heuristics to obtain as much as information as possible. Variety is critical, since a finite number of partitions is necessary to obtain good results with error rates lower than a single algorithm.

Fern and Brodley (2003) took a different approach with high dimensional data by proposing the use of Random Projection (RP) together with the cluster ensemble, in order to optimize clustering results. High dimensionality can of course be dealt with by using traditional dimensionality techniques, such as principal components analysis (PCA). However, as Fern and Brodley (2003) state: *"PCA chooses the projection that best preserves the variance of the data"*. Moreover, it is not always conducive to select only the most interesting features of a dataset, normalizing the data is not always applicable and PCA can make it hard to interpret clusters found when comparing to the original dataset (Fern and Brodley, 2003; James et al., 2013; Agrawal et al., 1998). Therefore, RP is used which circumvents the 'interestingness' by still using the original features. In combination with the cluster ensemble the authors proved that RP can outperform PCA in high dimensional datasets.

Last, a study by Strehl and Ghosh (2002) identified the cluster ensemble problem. Several algorithms, such as the Meta-CLustering Algorithm (MCLA) and Cluster-based Similarity Partitioning Algorithm (CSPA) are studied, which are aimed at solving the *correspondence problem*. This problem is particular for the unsupervised learning domain, since unlabeled data is handled in a symbolic manner. Thus, the cluster ensemble has to solve a correspondence problem before the consensus solution can be provided (Strehl and Ghosh, 2002). One way to handle the correspondence problem is by transforming the initial cluster outcomes to a hypergraph representation. In detail, initial clusters are mapped against each other, solving the symbolic labeling and hence the correspondence problem at hand. Furthermore, it is shown that robust results can be achieved in a consensus solution when diversity and quality are both taken into account. Diversity can be obtained by using various clustering algorithms, meanwhile quality is preserved by splitting the data into multiple lower dimensional subspaces.

2.4 Mechanisms that shape the cluster ensemble

In the previous sections the basics behind several clustering algorithms has been discussed, next to the key motivation of using a cluster ensemble. In this section a thorough explanation is given of the bits and bolts behind the cluster ensemble. Particularly, the steps from creating diversity and assuring quality are discussed, followed by methods for the consensus solution.

In basis, a cluster ensemble is made out of two stages, namely:

1. The generation stage; and
2. The consensus solution.

2.4.1 Generation stage

In the generation stage, base learners, equivalent to standard clustering algorithms, are used to perform the initial clusterings (Ghaemi et al., 2009). Base learners are often different clustering algorithms. The *golden rule* here is to use diversity, such as using multiple clustering algorithms. The rationale is that it will provide a robust clustering of the data.

Kuncheva and Hadjitodorov (2004) exploited in their study the role of diversity. Their conclusion is unequivocal; diversity improves accuracy of the cluster ensemble. Thus, leveraging diversity indeed improves the quality by means of accuracy and is not solely dependent of strong optimized algorithms as a base learner (Topchy, Jain, and Punch, 2003). However, diversity still remains an opaque expression up to this point. Diversity can essentially be seen as using different algorithmic setups and data presentations to the base learners. In Figure 2.3 the diversity methods are illustrated that conform to diversity for the generation stage. In sum, resampling of the data should be seen as using techniques like bootstrapping, as used in Minaei-Bidgoli, Topchy and Punch (2004) and Monti, Tamayo and Mesirov (2003), whereas different algorithms and parameters have been applied in studies by Topchy, Jain and Punch (2003), Fred and Jain (2002) and Kuncheva and Hadjitodorov (2004), meanwhile projections to subspaces by Fern and Brodley (2003) explored the effectiveness of RP. Last, subsetting the data is explored by Ayad and Kamel (2008) to explore many weak partitions from different cluster algorithms.

To formalize the generation stage in a more technical manner, consider having a dataset D consisting of observations $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$. To create diversity assume using the k-means algorithm with $k = 3, \ldots, 10$ and rs (random restarts) = 20. Then, 160 clustering partitions are made in total, presented as $\mathbb{P}$=$\{P_1, P_2, \ldots, P_{160}\}$. Essentially, each partition has formed clusters, for example $P_1 = \{C_1^i, \ldots, C_3^i\} \wedge \{C_1^j, \ldots, C_3^j\}$, where C^j a cluster with observations other than C^i. The result is that a cluster partition portfolio is created, denoted as $\mathbb{P}$, containing every possible cluster formation of the observations in $\mathbf{x}$. The goal, ultimately, is to find a consensus partition $P^* \in \mathbb{P}$ that results in a better representation of each partition in $\mathbb{P}$.

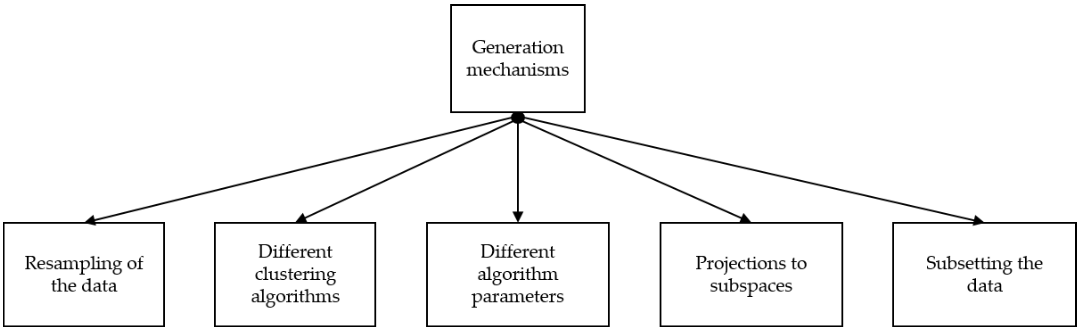


FIGURE 2.3: Generation methods for the cluster ensemble.

2.4.2 Consensus solutions

As partitions are created in the generation stage, the consensus stage will use these partitions to form a final *consensus solution*. Creating the consensus solution can be resolved with several methods, each with a different approach. These methods are discussed a comprehensive manner, so for more technical details see Vega-Pons and Ruiz-Shulcloper (2011).

In a broad view, the consensus solution can be obtained in two ways: 1. via object co-occurrence and 2. via median partition. It must be noted that the former method is used more often, basically because it obtains robust results (Fern and Lin, 2008; Iam-On et al., 2010).

Consensus with object co-occurrence

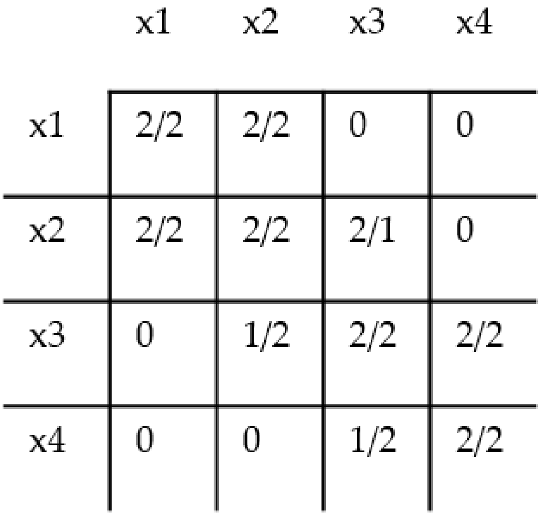
Methods based on object co-occurrence keep track of times whether an observation, or pairs of observations, are assigned to the same cluster over all the partitions created in the generation stage. To keep track of the co-occurrences the observations are assigned with cluster labels. However, these labels are not unique in the unsupervised setting. Thus, some consensus methods in the object co-occurrence method need to solve a correspondence problem (Ghaemi et al., 2009; Zhou, 2012).

The Voting and Relabeling method attempts to solve the correspondence problem by the assumption that each partition results in the same number of clusters. An illustrative example is given in Table 2.1. On the left-hand side of the table several label vectors are assigned to various objects. When observing, label vectors 1 and 2 are identical, whereas vector label 3 has some dispute and label vector 4 only consists of two clusters. It is reasonable to assume that the final partition will commit to the creation of 3 clusters, independent of the assignment (1,1,1,2,2,3,3,3) or (2,2,2,3,3,1,1,1). Thus, a voting process is performed that assigns objects to partitions in which they most likely occur. This resolves fuzzy partitions by creating distinct (hard) clusters in the Voting and Relabeling method (Vega-Pons and Ruiz-Shulcloper, 2011; Fern and Brodley, 2004).

TABLE 2.1: Example of a voting table with object co-occurrence.

v^1	v^2	v^3	v^4
1,1,1,2,2,3,3,3	2,2,2,3,3,1,1,1	1,1,3,3,2,2,2	1,1,0,0,2,2,0,1

The co-association method circumvents the correspondence problem by creating a (dis)similarity matrix. This matrix determines the frequency that objects X_i and X_j are in the same cluster for all partitions created. Hence, co-occurrence is counted by the times that objects form a cluster in independent runs of clusterings (Ghaemi et al., 2009). Then, assigned by a binary value, similarity between objects are measured in every single partition. A simple voting-K-means algorithm can be used to obtain the consensus partition. Fred and Jain (2002) proposed this as the evidence accumulation strategy. In Figure 2.4 an example is given of a co-association matrix.

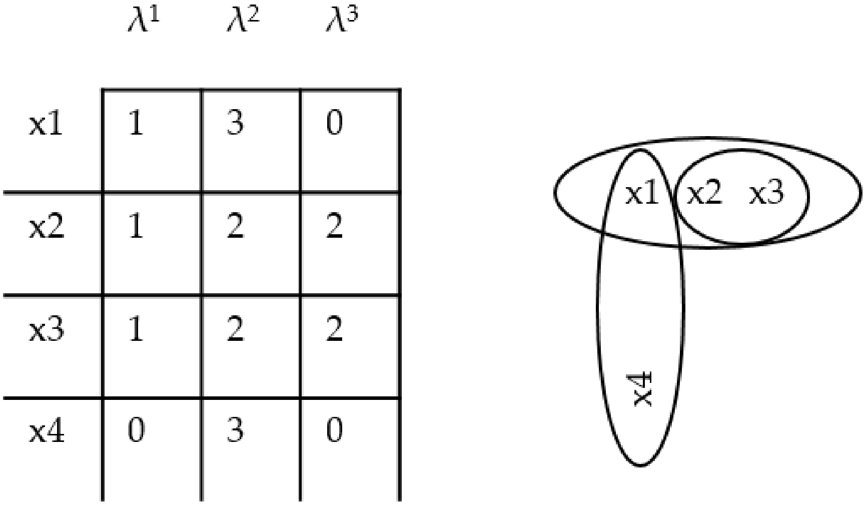


	x1	x2	x3	x4
x1	2/2	2/2	0	0
x2	2/2	2/2	2/1	0
x3	0	1/2	2/2	2/2
x4	0	0	1/2	2/2

FIGURE 2.4: Example of a co-association matrix illustrating which objects co-occur in a cluster over partitions.

Final in the object co-occurrence method are the hypergraph methods. These methods do not face a symbolic problem, rather objects with co-occurrence are transposed to a hypergraph. To illustrate, clusters are presented as hyperedges and its objects correspond to vertices. Each hyperedge contains information of a set of objects that belong to the same cluster (Ghaemi et al., 2009).

Strehl and Ghosh (2002) proposed the hypergraph in order to transform the correspondence problem from a symbolic problem towards a similarity matrix. The advantage is that the consensus solution now only needs to solve a mutual information problem. This problem is easily expressed with a binary matrix. Observations are mutual if they share the same cluster [1] or not mutual [0]. To solve the similarity matrix CSPA can be used. It considers whether objects co-occur, i.e. have similarity and are transposed to a similarity matrix. An example of creating a similarity matrix with a hypergraph method (CSPA) is given in Figure 2.5.



	λ^1	λ^2	λ^3
x1	1	3	0
x2	1	2	2
x3	1	2	2
x4	0	3	0

FIGURE 2.5: Example of a hypergraph method. The left: outcome of cluster partitions with objects assigned to a cluster. Right: hypergraph representation with observations as vertices and clusters as hyperedges.

Consensus with median partition

The consensus solution with median partition methods is obtained by solving an optimization problem. The optimization problem can be defined as maximizing the similarity amongst partitions in the cluster ensemble (Vega-Pons and Ruiz-Shulcloper, 2011). As defined by Ghosh and Acharya (2011); the distance between two clusterings is measured by defining the number of pairs of objects that are placed in the same cluster or in a different cluster. One of the well-known median partition methods is the Mirkin distance.

The Mirkin distance is a measure of symmetric similarity between the clusters. In other words, distances are minimized in order to find an optimal partition. This is done by comparison between partitions: (Vega-Pons and Ruiz-Shulcloper, 2011):

- n_{00}: Pairs of objects that were clustered separately in P_1 and P_2;
- n_{01}: Pairs of objects that were clustered in a different cluster in P_1, but in the same cluster in P_2;
- n_{10}: Pairs of objects that are co-clustered in the same cluster in P_1, but not in P_2; And
- n_{11}: Pairs of objects that were co-clustered in both P_1 and P_2.

Then, the number of disagreements between two partitions (n_{01}, n_{10}) is used as the symmetric difference distance, i.e. the Mirkin distance.

In conclusion, the cluster ensemble method defines two stages. The generation stage is used to generate as much as diverse partitions as possible, in order to improve clustering. Second, the consensus solution is used to obtain a final partition that achieves a similar or better result. In Figure 2.6 an illustration is given of the total cluster ensemble approach.

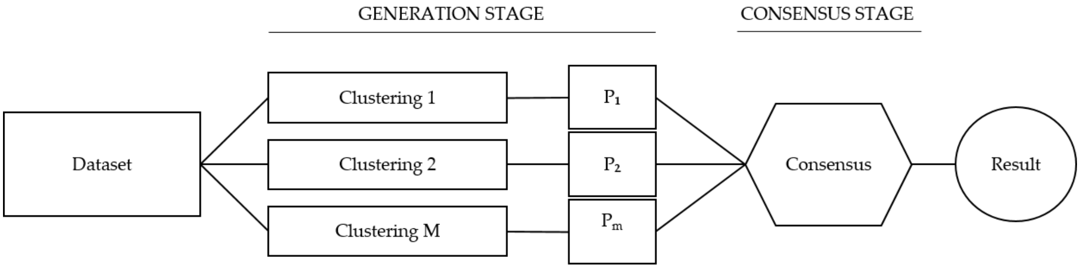


FIGURE 2.6: Overview of the cluster ensemble.

Chapter 3

Cluster ensemble concepts

In this chapter a general outline is created for the first study objective by following the CRISP-DM (CRoss-Industry Standard Process for Data Mining) methodology from Chapman et al. (2000). Several points are discussed that define how unsupervised data can be best understood, prepared, modeled, and evaluated when the cluster ensemble is used for data analysis. This chapter, therefore, will serve as a procedural manual for the development of the generic cluster ensemble meta-algorithmic model.

First, the CRISP-DM is briefly introduced, followed by the procedural and technical concepts that underlie some of the CRISP-DM steps, such as data understanding, preparation, modeling and evaluation. Since this is a technical project based on a single exploratory case study in the Psychiatry Department of the UMCU, non-technical tasks of the CRISP-DM are omitted, such as business understanding, deployment, and some activities related to the evaluation step. Next, each CRISP-DM phase is handled separately from an unsupervised machine learning perspective, aimed at clustering. Section 3.1 starts with a brief introduction to CRISP-DM and is followed by section 3.2 with the Data Understanding phase. Next, Data Preparation, and Data Modeling and Evaluation are discussed in sections 3.3 and 3.4, respectively. Last, a summary of this chapter is given in section 3.5. In Appendix A source code can be found for some of the images produced in this chapter.

3.1 Introduction to data mining with CRISP-DM

Capturing and storing data is common practice in the healthcare domain nowadays (Luo et al., 2016). Giving the opportunity for data mining to be explored. Data mining is an approach to uncover recently unknown patterns in the data and transform it into actionable format for the business in terms of prediction and description (Leventhal, 2010). In essence, new knowledge is created from the previously unknown patterns and is used to answer new business goals. Indeed, data mining has become an integral part in many domains.

One way to lead these data mining efforts is by following a methodology that provides guidance in generic ways to scrutinize the data. CRISP-DM is such a methodology that is generic to both industry and technology (Wirth, 2000). Particularly, depending on the data mining context, each domain can tailor the CRISP-DM model (Niakšu, 2015). The purpose of the CRISP-DM is to provide an iterative and flexible framework with a goal to increase and effectively use the knowledge gained from the data mining efforts (Azevedo and Santos, 2008). Furthermore, it is considered as the standard framework in the field of data mining and knowledge discovery (Spruit, Vroon, and Batenburg, 2014). Because it is a generic and flexible methodology, several studies have adapted the CRISP-DM by making it more suitable to their needs. For instance, Menger et al. (2016) adapted the CRISP-DM to enforce collaboration between medical experts and data scientists to improve knowledge creation and hypothesis testing in the healthcare domain, naming it CRISP-IDM. Subsequently, Niaksu (2015) proposed the CRISP-MED-DM, a methodology to perform multi-disciplinary collaboration in data mining for the medical domain. Last, the CRISP-DM is also considered as the cornerstone for development of meta-algorithmic models to perform data mining for non-tech savvy experts (Spruit and Jagesar, 2016; Spruit and Lytras, 2018). Therefore, the CRISP-DM will also serve its purpose in this study to develop the cluster ensemble meta-algorithmic model.

The CRISP-DM can be seen as a concatenation of iterative processes that are fitted to the situation at hand. It consists of six steps —which are explained in the following subsections—, and each step has its own phases with activities involved, for example, see Figure 3.1. These steps do not have to be followed in a linear fashion, the arrows to follow are only indicators of the most important and frequent dependencies between the phases (Wirth, 2000). Hence, the CRISP-DM allows to fit a data mining project that suits the situation at hand and symbolizes that data mining is not an one-step project by its cyclical arrows (Chapman et al., 2000).

Besides CRISP-DM there are also other methodologies guiding data mining efforts. These methodologies will not be explained thoroughly, since this is not the goal of our study. Instead, a short introduction is given to form a global overview and to justify the reason behind choosing CRISP-DM.

SEMMA is an acronym of Sample, Explore, Modify, Model, and Assess, with the goal of extracting valuable knowledge from data with data mining. In contrast to CRISP-DM, SEMMA focuses on the extraction of information from the data. Thus, the business understanding part is absent in this framework. Nonetheless, SEMMA provides sequential steps in performing data mining tasks for businesses. For in-depth information about SEMMA, refer to Matignon (2007)

KDD, Knowledge Discovery in Databases, focuses on an overall process of knowledge discovery from data. The emphasis of KDD is based on how techniques can be scaled to cope with large datasets, meanwhile extracting results and creating visualizations (Fayyad, Piatetsky-Shapiro, and Smyth, 1996). The methodology consists of five sequential steps:

1. Selection;
2. Preprocessing;
3. Transformation;
4. Data Mining; and
5. Interpretation and Evaluation

Absent in KDD are the business and data understanding parts. Whereas in the CRISP-DM the first step aims at setting the goal of the data mining project, KDD does not specifically makes it part of its own. Yet, this is considered as a fundamental part before a project can start (Azevedo and Santos, 2008). For in-depth information about KDD, refer to Fayyad et al. (1996).

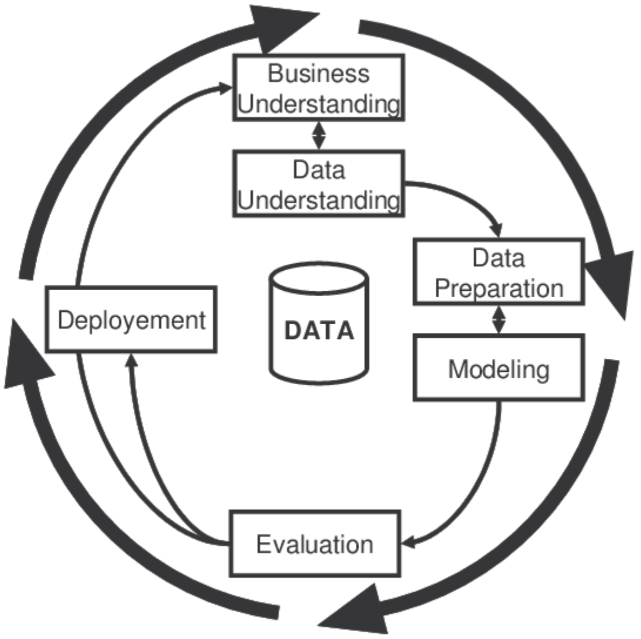


FIGURE 3.1: The CRISP-DM cycle by Chapman et al. (2000)

3.1.1 Business Understanding

The first and initial phase focuses on managing expectations (i.e. what is the main goal of the project, what are the underlying business objectives, which criteria are needed to measure success and what are the data mining goals to ensure the achievement of the objectives that satisfy the main goal). These intermediary steps form the basis of a project plan that is designed to satisfy the objectives and is crucial for every data mining project.

3.1.2 Data Understanding

The second step, which is crucial before moving on to data preparation, is to get an overall understanding of the data available. Apart from collecting the necessary data, quality assessments are incorporated as well. For instance, the check for missing values, if values need to be imputed, which features are in the data (if data is structured), the size of observations features and which types of features are present (i.e. *continuous, discrete or categorical*). Therefore, understanding the data is vital, since data is rarely perfect.

3.1.3 Data Preparation

Data preparation is the final step before Data Modeling and is often considered as a crucial step in a data mining project (Wirth, 2000). In fact, data preparation usually consumes between 50-70% of project time and effort (Chapman et al., 2000). Basic activities in this step can be seen as selecting the right data, cleaning the selected data, constructing new variables and observations from the present data and sometimes integrating data with other datasets.

3.1.4 Data Modeling

In the modeling phase the assessment of an algorithm can begin that is of interest to the business problem stated in the first step. Deciding which algorithms to use is not always clear cut. Thus, assessing multiple algorithms is often a good way to get insight in their performance and accuracy, since there is not one that is gold standard. In addition, the data modeling and data preparation phase are bilateral, since problems with the data might arise when modeling (Wirth, 2000). One rule in the modeling phase is that there are no violations made to the dataset. That is, a dataset may never be applied as both a training and validation set.

3.1.5 Evaluation

The models built from the previous phase are now examined against the business success criteria that underlie the business objectives from the first phase. It is to ensure that the algorithm used in the modeling phase ensures the achievement of the business goal. Furthermore, by following a review process the weaknesses and strengths of the data mining project are used to improve upon future data mining projects.

3.1.6 Deployment

In the last phase the deployment will take place. By any means, the deployment phase is not the sole end of a data mining project, but another start for improvement in the business.

3.2 Data Understanding in the unsupervised domain

Data understanding is the second phase of the CRISP-DM cycle and is fundamental in order to become familiar with the dataset in question. The goal is to generate graphical displays of the data. This provides a direct summary and eases examination and understanding of the data, which is the prime goal of Exploratory Data Analysis (EDA) (Behrens and Yu, 2012).

Exploratory Data Analysis is a term that stems from Tukey (1977), which is the search for misalignment in the unanticipated areas of a dataset (Gelman, 2004). From a different viewpoint, Behrens and Yu (2012) describe EDA as a process to easily detect relationships, and recognize patterns in a dataset, which is marginally exhaustive compared to text summarized descriptions. In overall, data mining projects following the CRISP-DM advocate the usage of EDA, since it aids in addressing data mining goals and helps formulating hypotheses and the necessary data transformations for the data preparation phase (Chapman et al., 2000).

The meta-algorithmic model built for the cluster ensemble requires a focus on clustering. The aim, therefore, is to determine in which ways data can be understood so that a raw estimate is given for: 1. the amount of clusters, and 2. the variances between features and observations. For brief explanatory purposes, the number of data understanding approaches is limited to two, and assume that the datasets in question will only consist of numerical values.

The first visualization technique Multivariate Kernel Density Estimates (KDE) shows regions of density and sparseness between features of a dataset. Thus, observations sharing similarity in a feature will form a dense region. The second and last visualization technique is a heatmap with a dendrogram. This technique is robust when the number of features within a dataset becomes a problem to visualize. For instance, the amount of plots to show between features can be expressed as $b = (p-1)p$. Where p is the total amount of features and b the total amount of resulting plots. Then, consider having a low dimensional dataset with 25 features. The result is 600 KDE plots which is infeasible to inspect. Therefore, a heatmap with a dendrogram both give a raw indication of the number of clusters found based on the ordering between features and observations and will show which features have high variance (Wilkinson and Friendly, 2009). With the latter being an important matter for the next step in the data preparation phase.

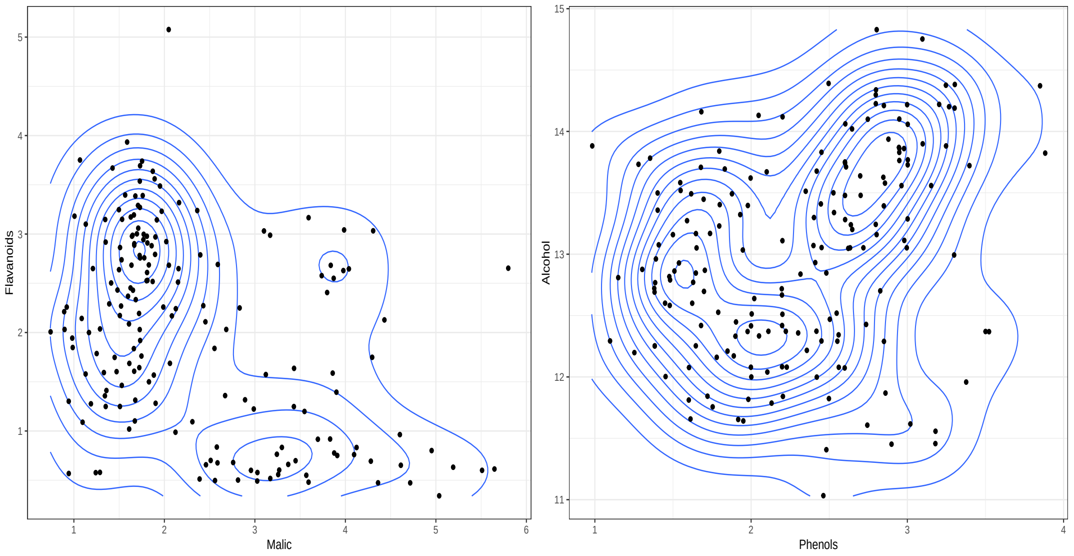
Although groupings may be evident from the visualization, there are some intrinsic assumptions made by these visualizations that affect the outcome in a certain manner. For example, the kernel being used to built the density or the type of linkage used to built the dendrogram. Thus, the most important aspect to keep in mind for this phase is that each visualization reveals a certain phenomenon of the dataset (Behrens and Yu, 2012).

To begin exploring these two visualization techniques we make use of two datasets. The first dataset is the Wine dataset (Aeberhard, 1991), which consists of 14 features and is partially explored with KDE. The second dataset is the Wisconsin Diagnostic Breast Cancer dataset (WDBC) (Wolberg, Street, and Mangasarian, 1995), consisting of 30 features and is explored with the heatmap with a dendrogram. The goal is to demonstrate that both techniques provide a useful visualization of the data.

3.2.1 Multivariate kernel density estimates

Visualizing numerical data via KDE provides a robust way to present binning of the observations when compared to histograms or scatter plots. While histograms are commonly used, the data should be interpreted with caution, since its representation is determined by the bin width. The bin width refers to the interval on the x-axis of the histogram, giving a raw estimation of the distribution in the data. Different bin widths will provide a different distribution, and therefore, other visualizations of the same dataset. In turn, this can lead to different interpretations of the same data (Behrens and Yu, 2012). Furthermore, scatter plots do not always reveal if observations share similarity, since they are only plotted according to their numerical value. Kernel Density Estimates on the other hand does not suffer from these two drawbacks, as there is no effect of overestimating or underestimating due to the bin width. Instead, density is only observed when observations are close to each other, which form together a dense peak. The benefit of using KDE over histograms and scatter plots, is that the density estimates seamlessly visualize multi-modality in a dataset. That is, in multivariate datasets dense regions usually indicate that there are clusters to be found in the data (Everitt et al., 2011).

In Figure 3.2 two KDE plots are displayed. Based on a selection of different variables from the Wine dataset, it seems that there are three clusters. In Figure 3.2(A) there is one high density region which is close to the left side of the plot, and two smaller regions located at the bottom, and at middle of the plot. In Figure 3.2(B) this is more clear by three separating density regions.



(A)

(B)

FIGURE 3.2: Multivariate KDE plots from the dataset Wine. (A) Variables malic & flavanoids. For (B) variables alcohol & phenols.

3.2.2 Heatmap with a dendrogram

Dendrograms, as described in section 2.2.1, provide a straightforward way to visually determine clusters within a dataset. The dendrogram displays in a tree like structure the (dis)similarity between observations. Thus, giving a raw indication of the cluster structure (Murtagh and Contreras, 2012). Consequently, the dendrogram can be split, known as tree cutting, branch pruning, or branch cutting (Langfelder, Zhang, and Horvath, 2008). Tree cutting is, however, a static and sometimes misleading process, giving small evidence of the total amount of clusters in the dataset. Indeed, as Sarstedt and Mooi (2014) point out, tree cutting only provides small evidence to the real clusters in a dataset. Thus, cutting should be done with some caution.

Dendrograms can also be displayed with a heatmap, serving the purpose of identifying whether the dataset contains features or observations with high variance. This typically occurs when features are not recorded at the same level (i.e. height versus weight of humans). In Figure 3.3 the heatmap and dendrogram is displayed for the WDBC dataset.

The heatmap shows the contrast between the observations and variables. For example, several variables have clearly low variance, indicated by a purple color, while some have average to high variance, indicated by a green or yellow color. From the dendrogram it is visible that there is a cluster structure, possibly with $k = 2$ to $k = 3$ if the tree is cut lower.

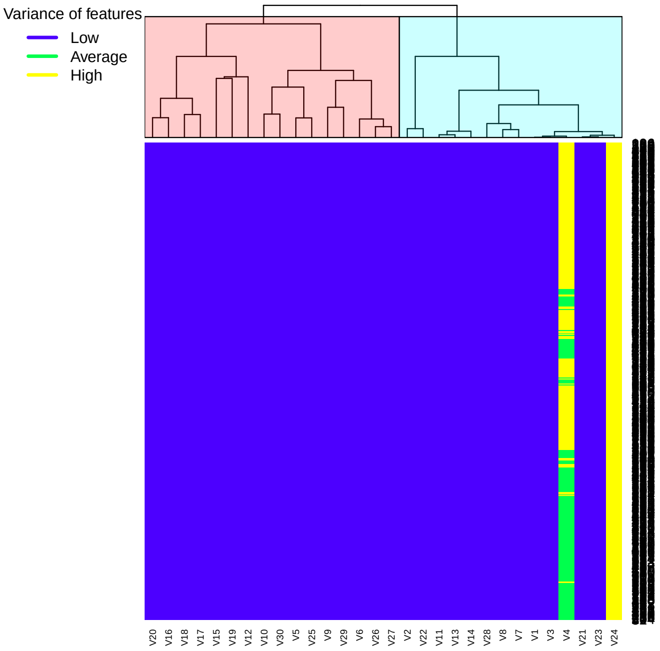


FIGURE 3.3: Heatmap with dendrogram from the WDBC dataset showing all features

3.3 Data Preparation for clustering

Preparing the data is the next step after a general understanding of the data has been collected. Preparing the data does not follow any prescribed order, but often requires several steps to make the data fit for modeling. Altogether, data preparation constitutes of steps for transforming the initial raw data into a final dataset (Wirth, 2000). As our meta-algorithmic model is centered around the cluster ensemble, feature scaling, setting the proximity metric, selecting features and dimension reduction techniques are discussed.

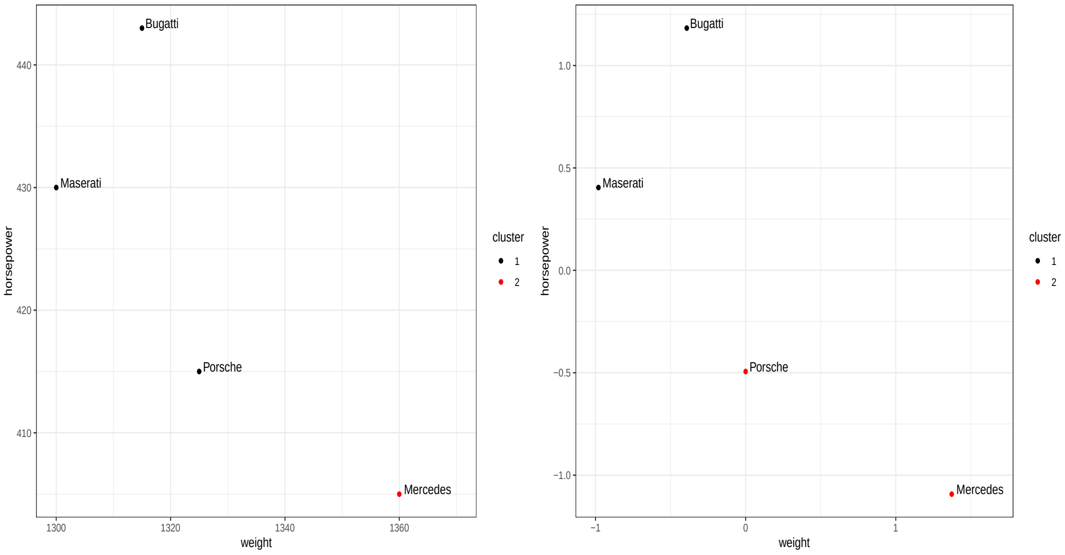
3.3.1 Feature scaling

Feature scaling, also known as *standardization* or *normalizing*, is common practice in clustering, and often in the supervised domain too (James et al., 2013). Standardization equalizes the variables. For example, by normalizing each variable to a z-score. The result is that each variable will have a mean of 0 and a variance of 1. Thereby, each variable is considered as equal in importance and it removes some bias by the clustering algorithm to place observations with high variance in the wrong clusters. Consequently, the final result will not tend to create unnatural clusters by penalizing observations with low variance.

Standardization can be performed in various ways for numerical data. As explained previously by creating distributions of a z-score. Milligan and Cooper (1988) examined several standardization methods for numerical data in combination with different hierarchical clustering methods. Their result indicate that standardization by range (i.e. observations in variables are bound to a minimum and maximum value with a varying mean and variance per variable), recover the best cluster structure within a dataset (Milligan and Cooper, 1988).

In addition to standardization there is also a special case of weighting the variables. Based on *domain knowledge* or self-judgment, weights are assigned to variables reflecting their importance. Everitt et al. (2011) discusses two methods which are *direct* or *indirect* weighting. The direct method increases weight on variables proportional to its total variability between and within groups. However, as Fleiss and Zubin (1969) in Everitt et al. (2011) state, this direct weighting method poses some serious disadvantages to cluster analysis. That is, between groups variance is an important measure that separates clusters. The indirect method is a form of perceived observing and testing, in which weights are placed on variables that only improve dissimilarities between the clusters. However, when dealing with high-dimensional data this method seems to be inefficient and is ill-favored in the case of cluster analysis, since it introduces subjectivity. Therefore, an indirect approach can cause unnoticed clusters to not emerge from the data (Everitt et al., 2011).

To demonstrate the importance of scaling, examine Figure 3.4. Assume that there are two clusters, one for Porsche and Mercedes, the second for Bugatti and Maserati. In figure 3.4(A) there is no scaling prior to cluster analysis. K-means in this example assigns Porsche to the wrong cluster, namely together with Bugatti and Maserati. Indeed, because the variable *weight* has a greater variance than *horsepower*, the Porsche is wrongfully assigned to the first cluster. In Figure 3.4(B), z-score scaling has been applied. Now it is clear that the Porsche has been clustered together with the Mercedes, as each variable is considered equal in importance.



(A)

(B)

FIGURE 3.4: Left A): No scaling performed. Right B): z-score scaling.

3.3.2 Proximity measures

Forming clusters requires, in general, a proximity metric that either defines the *distance* between observations or their *similarity* to each other. Take k-means for example, based on its clusters centroid observations are assigned to its nearest centroid. Defining 'nearest' however, requires proximity, such as the Euclidean distance, which allows to form clusters. As Kaufman and Rousseeuw (1990) note; proximity defines structure within a dataset.

Determining proximity requires careful consideration because each type of metric defines its (dis)similarity differently. For that reason, several of the most used proximity metrics from the literature are discussed, in the order of proximity metrics for *continuous*, *categorical* or *mixed-mode* features.

Proximity metrics for continuous variables

Clustering variables of a continuous type have typically designed metrics operating with triangle inequality. These are known as distance metrics. The most commonly used distance metric is the Euclidean distance, or the l_2 norm. It is a derivative from the general Minkowski distance, otherwise known as the l_r norm. The Euclidean distance is appealing because it transforms the observations to have physical distances. Also known as a matrix in Euclidean space. This matrix allows the distance between observations to be computed following the Pythagorean theorem. Hence triangle inequality. Analogous to the Euclidean distance is the Manhattan distance, or the l_1 norm, which is a second derivative from the Minkowski distance. In contrast to the l_2 norm, the l_1 norm takes a rectilinear approach to define the distance (e.g. distance is calculated following a street configuration).

Although both metrics discussed in the latter are designed for continuous variables, it should be noted that the l_1 norm is preferred when dealing with high-dimensional data. Specifically, the l_1 norm takes absolute distances between observations, making it by nature less susceptible to outliers than the l_2 norm (Frades and Matthiesen, 2010). Moreover, Aggarwal, Hinneburg and Keim (2001) argue that using lower l_k norms should be taken into consideration when dimensionality is high in a dataset. Specifically, a lower Minkowski distance than the l_2 norm, or even the l_1 norm is preferred.

Turning to the Minkowski distance, known as the l_r norm, is somewhat similar to both the Manhattan and Euclidean distances. The main difference is that the l_r norm takes a parameter from ≥ 0 to ∞. It should be clear that setting the parameter value to 1 or 2 will make the Minkowski act as a l_1 or l_2 norm. However, taking on values < 1 or > 2 naturally will result in different distances. Using different parameter values depends on the situation at hand, since it will change the distances between observations and therefore will result in different outcomes.

Adjacent to the distance metrics are similarity metrics. Similarity metrics, such as the Pearson correlation or Spearman correlation, are used to quantify the similarity between observations. In other words, when observations have a positive relationship a real number, for example ≤ 1, is given. Whereas in the case of a negative relationship a $\geq$ -1 is given. In turn, these relational numbers are used to define distances between observations. Everitt et al. (2011) suggests, however, to be cautious using similarity metrics. Variables should represent measurements that match. For example, in gene expression profile studies, as by Hristoskova et al. (2014), Iam-on et al. (2010) and Priness et al. (2007), similarity metrics are used, since there is evidence that there is a natural relationship between genes and tissues. At the same time it remains often a case of prudent decision-making to choose the right distance or similarity metric for each dataset (Everitt et al., 2011; Priness, Maimon, and Ben-Gal, 2007).

Proximity metrics for categorical variables

Categorical variables use in general a similarity metric. In most cases the variables are transformed to a binary scale, since this allows to define whether observations are similar. For instance, a value of 1 indicates unity between observations and a value of 0 indicates that observations differ maximum (Everitt et al., 2011). Again, it is a simple matter to convert similarity values to distances by taking **dist** = 1 - S_{ij}, where S is the similarity coefficient, and $_i$ and $_j$ the observations.

Three well-known similarity metrics for categorical data are the Jaccard coefficient, the coefficient index proposed by Sneath and Sokal (1973) and the Matching coefficient. With the first two similarity metrics, pairs of observations both with a coefficient of 0 are treated the same as observations with a coefficient of 1. For example, humans are treated the same in the Jaccard coefficient, since both male and female are humans. In case of the Matching coefficient the 0 often represents absence (total dissimilarity). For example, to distinct humans based on their reproductive organs. Thus, zero-to-zero and one-to-one matches are mutually exclusive in this type of similarity measure.

Determining which type of similarity measure to use is not always obvious. As Sneath and Sokal (1973) in Everitt et al. (2011) point out; each set of data must be considered, since there is no ground truth regarding the inclusion or exclusion of positive or negative matches.

Undoubtedly, each similarity measure will provide different means, so the decision-making process is important.

Proximity metrics for mixed-mode variables

Multivariate datasets often contain variables which are continuous, discrete and categorical. These datasets are referred to as being of mixed-mode. Surely, variables can be transformed to represent numbers (e.g. transform nominal and ordinal variables to natural numbers). Alternatively, variables can be transformed to allow a distance based metric suited for continuous data. On the other hand, Gower's general similarity metric can be applied to mixed-mode data. Gower's metric is in fact a similarity metric as its name suggests. Similarity between observations is based on whether both observations have a relationship in the variable. So, if both observations are correlated in the variable, a value of 1 is given. Whereas if one or both are not correlated a value of 0 is set. For continuous variables in the dataset a special derivative of the l_1 norm is used. Continuous variables are scaled to unit range, giving the highest observation a value of 1 and the lowest 0. Next, the resulting matrix from the mixed-mode dataset can be transformed to satisfy properties of the l_2 norm (i.e. triangle inequality), by recalculating the coefficients with $1 - |(i - j)|/range$ (Gower, 1971).

3.3.3 Feature selection

Feature selection, also known as subset selection or variable selection, mandates the search for the most interesting variables in a dataset. Feature selection can be performed with three different procedures. In detail, the search for the best subset begins with selecting one variable at time, and evaluate —using a measure criterion— whether the addition of another variable is meaningful for the final outcome. This is the *forward selection* procedure, starting with a model without features and iterates over the dataset to filter upon important features (James et al., 2013). Second, the *backward selection* procedure does the opposite. This procedure starts with all variables and uses an algorithm to create different subsets varying with variables. From these candidate subsets only the one with most interesting variables is picked. Last, the *hybrid selection* procedure is a mix of both worlds. Thus, it uses a measure criterion to filter upon the variables, and afterwards it uses an algorithm to select the best subset along candidate subsets. Accordingly, these strategies are known as *filter*, *wrapper* and *hybrid* methods in the machine learning community.

Using feature selection prior to modeling is a viable procedure to perform. In the supervised domain, for example, models with feature selection often outperform those without feature selection in terms of accuracy. Spruit and Jagesar (2016) argue that including features without informative importance (or any importance at all) causes noise to an algorithm, affecting its general performance. Indeed, by removing unnecessary features model accuracy and comprehensibility increase (Dash et al., 2002). However, this is for the supervised domain. In the unsupervised domain feature selection has often been overlooked in terms of techniques available compared to the supervised domain (e.g. see table 1 in Liu & Yu (2005)). Arguable this can be due to the general understanding of clustering, in which it is assumed that all features are equal in importance. Moreover, it is considered more problematic to define salient features in the unsupervised domain (Law, Figueiredo, and Jain, 2004). Contrast this to the supervised setting in which feature selection methods assess the goodness of features based on prediction accuracy, since it deals with a response variable. Something that is not available in the unsupervised domain. But including all features often introduce noise, and sparseness through dimensionality, both affecting the outcome for many clustering techniques (Parsons, Haque, and Liu, 2004). In fact, both comprehensibility and compactness of clusters decrease as uninformative features affect the process (Dash and Liu, 2000). Some recent developments therefore applied various feature selection procedures for clustering. Also coined as *filter* (Dash et al., 2002), *wrapper* (Parsons, Haque, and Liu, 2004) or *hybrid* (Dash and Liu, 2000). Each will be discussed briefly.

Dash et al. (2002) introduced a *filter* procedure to select the best subset among features prior to clustering. Based on the entropy distances between observations can be found for each feature. Entropy calculates the probability of equality, as in; the entropy is maximum if each observation is equal in the feature, otherwise it is low. So low entropy and high entropy are both indicators for heterogeneity or homogeneity in a feature. As a result, features can be selected a priori to modeling that will create meaningful clusters.

Dash and Liu (2000) proposed a *hybrid* procedure that first creates a subset of candidate features, using the same filter procedure as in the latter meanwhile using a *wrapper* method to choose the best subset out of the candidates. As the authors argue, using this type of procedure is suggested in cases of high-dimensionality, since selecting informative features among many requires a priori knowledge. Thus, a preliminary clustering algorithm can be used to *wrap* the best subset out of the candidates. Consequently, it must be noted that although the *wrapper* procedure often outperforms the *filter* procedure, it requires more time to be performed (Huan Liu and Lei Yu, 2005; James et al., 2013).

Last, Dy and Brodley (2000) proposed the *wrapper* method that incorporates both an initial clustering algorithm, and a criterion measure to evaluate the end result. The outcome is dependent on the criterion used, which is also a returning discussion in Parsons, Haque and Liu (2000), and Liu and Yu (2005). Therefore, feature selection can also be applied as an intermediary tool to gain better understanding of the dataset in question (Dy and Brodley, 2000).

To conclude, selecting an appropriate feature selection procedure often relies on the dataset in question. Filter models, for example, are convenient to use in small to medium-sized datasets up to a few hundred variables. Large datasets, where $n < p$, the *wrapper* or *hybrid* procedures are often used. These last two methods perform quick salient feature analysis, and afterward selecting the best subset out many candidates via an intermediary algorithm.

3.3.4 Feature extraction

The previous section discussed different techniques to select the most important features within a dataset. In a somewhat similar manner feature extraction also selects the most informative features from a dataset. However, the selected features are transformed prior to extraction. Thus, as dimensionality is reduced, transformation of the original data takes place. To provide some background behind this idea PCA is taken as an example, since it is an unsupervised feature extraction technique and often applied prior to modeling in both the unsupervised and supervised domain.

PCA extracts *principal components* (a collection of features that explains most of the variance) from a dataset, and it does so by transforming the features to a linear combination. In detail, each feature is centered to a mean of zero, and based upon the observations the variability can be calculated. Because PCA wants to find the most informative features (i.e. maximal variability), the first principal component is an aggregation of features that best explain the data. The first principal component forms a dimension which lies closest to all the observations, meaning that this dimension provides the best summary of the data. Subsequently, the second principal component is again a linear combination. However, this component is based on features that are maximal uncorrelated with the first principal component (James et al., 2013). Nonetheless also providing the best summary, but often at a lower degree. This process is iteratively repeated until the dataset ends up with a number of principal components that maximal differ from each other (otherwise orthogonal in direction when visualized). The result is a handful of principal components that explain most of the information in a dataset, hence dimensionality is reduced.

The reason why feature extraction techniques are an unsupervised method is simple. These techniques only use the observations within a dataset to reduce its dimensionality. Thus, class labels or response variables are not taken into account.

In addition to PCA there are other techniques available that also apply feature extraction, such as the Karhunen-Loéve transform (KLT) (Fukunaga, 1990), which is applicable in the absence of class labels (Dy and Brodley, 2000), and RP. A drawback of using a technique like PCA is that it transforms the dataset, affecting for example the clustering result. RP, however, preserves original features, is computational wise more efficient than PCA and is data-independent (Bingham and Mannila, 2001), meaning that it does not rely on variability of the observations. In turn, it does not affect clustering results. Making it a prominent solution to adopt for the cluster ensemble if dimensionality reduction is necessary.

Random Projection

The heart of RP lies in the Johnson-Lindenstrauss lemma, stating that each observation in a high-dimensional dataset can be projected to lower Euclidean subspace while nearly preserving the original distances (Bingham and Mannila, 2001). The result is that a large dataset is reduced in feature size, taking care of the *curse of dimensionality* meanwhile clustering obtains improved results, since clusters are more spherical in lower dimensions (Dasgupta, 2000; Fern and Brodley, 2003). As Achlioptas (2001) proves, applying RP brings a marginal loss of information for large datasets.

To understand RP better, see Figure 3.5. A simple representation of an arbitrary dataset is shown in three-dimensional space, but assume that this dataset is high-dimensional. By looking at the left-hand side we observe that neighbors of the observations are far (i.e. distances are based by the width size). As such, this dataset suffers from the *curse of dimensionality*. Consequently, eccentric clusters are formed, hence clustering has a hard time in finding clusters. By embedding this dataset to a lower subspace decreases sparseness, meanwhile keeping distances nearly preserved. Therefore, clusters are to be found more easily, and have a natural spherical shape, which is shown at the right-hand side in Figure 3.5. The optimistic result shown here, and discussed earlier, is that clusters tend to be better preserved after RP.

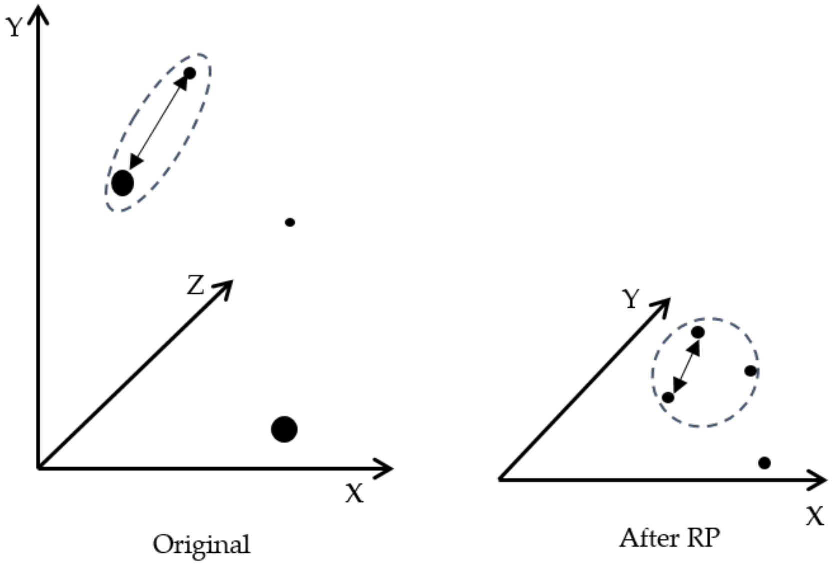
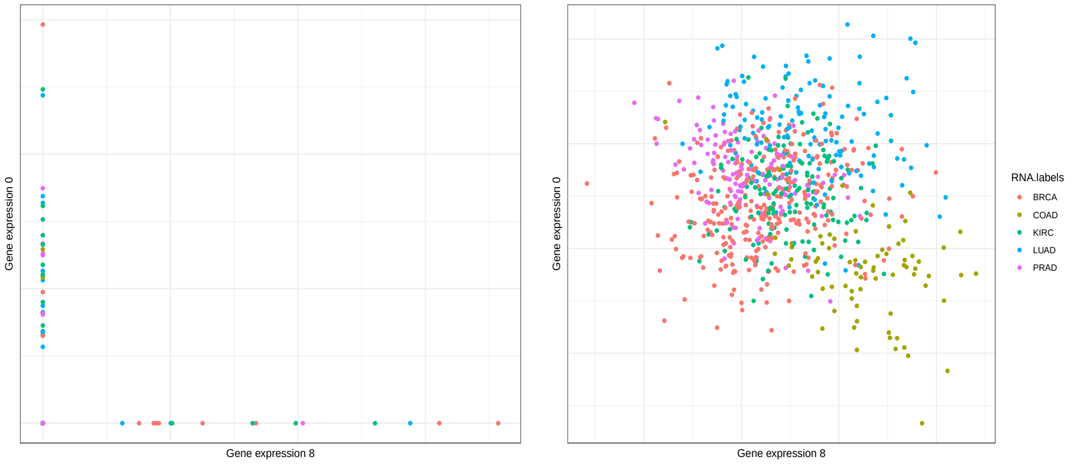


FIGURE 3.5: Random Projection, before (left) and after (right).

Now, lets consider a real high-dimensional dataset, namely the gene expression cancer RNA-Seq dataset (Weinstein et al., 2013). It has over 20,000 features and only 801 observations. The features contain extraction of gene expressions from patients having different types of tumor. In Figure 3.6(A) two of the original features are plotted, and notice the sparseness between observations. In Figure 3.6(B) RP has been performed on the same dataset, reducing it to 41 dimensions. Clearly, observations are less sparse and model comprehensibility is improved.



(A) Original

(B) After RP

FIGURE 3.6: (A): Gene expression 0 & 8 shown from the original dataset, p = 20531. (B): The same features shown after RP. Clearly observations are less sparse in the reduced dataset, p = 41.

RP, however, suffers from a major drawback when used prior to clustering. That is, each subspace is randomly created, making it variant to each clustering solution. Contrast this to PCA, which always gives the same outcome on the same dataset, making RP a serious challenge to be used. Despite the drawback, Fern and Brodley (2003) examined the robustness of RP against PCA in a cluster ensemble solution. Their findings are remarkable as RP was able to identify better cluster structures over various datasets compared to PCA. This concludes that RP is effective, and independent of observations, meanwhile it is likely to obtain natural cluster structure in high-dimensional datasets.

3.3.5 Feature construction and transformation

Raw data often contains features that are: 1. not directly suitable for the algorithm, 2. do not capture the right amount of information and, 3. has wrong or missing values. During these situations some *engineering* is necessary to obtain valuable features, which are often generalizable to other algorithms as well. Thus, the objective is to construct new features that are more informative than the original ones. In addition, constructing informative features makes the data more interpretable to both domain experts and the algorithm. For example, consider the feature ’Age’ in a dataset that contains the current age of each observation. Based on the age, a *dummy variable* can be assembled that categorizes the observations between child, adolescent and adult. By doing so, a classification or clustering algorithm can be used more effectively to classify or cluster a certain group of people based on the grouping from the dummy variable. In fact, Domingos (2012) argues that feature engineering is key to every machine learning project, since raw data is in general not in a ready format. This requires the construction of additional features that convey the right information. Another more complex example of feature construction, which is part of this study, is the construction of a detailed diagnosis from DSM data. This type of construction requires domain knowledge. In other words, knowledge is incorporated into the data to construct a more informative feature.

Sometimes, transformation of features is necessary to improve the accuracy and interpretation of the model. Prime examples are supervised algorithms that tend to work better when continuous variables follow a normal distribution. In cases like these, some preliminary understanding of the data is necessary (i.e. by knowing if there is skewness among variables). However, transforming variables, besides feature scaling discussed in section 3.3.1, is beyond the scope of this study, since unsupervised learning, and clustering especially, remain neutral whether variables follow a normal distribution or not. In fact, variables that do not follow a normal distribution or shows irregular peaks are often an indicator for clusters.

3.4 Data modeling the cluster ensemble

This section concerns the configuration of data modeling the cluster ensemble. Recall that for the cluster ensemble two stages are defined: 1. the generation stage and, 2. the consensus solution. From this perspective several steps are discussed that embody the modeling approach for the cluster ensemble.

3.4.1 Training data and testing data

Important in machine learning is the so-called "model fit". In basic terms this means that an algorithm's accuracy needs to be optimal. In context, an accuracy is not deemed optimal when the accuracy is estimated between 40-60%, since this is near flipping coins. The exact number of optimal accuracy, however, remains obscure. James et al. (2013) argues that accuracy of a model is domain dependent, since the irreducible error can contain many unaccounted variables which affect the accuracy.

To determine model fit, a dataset is usually split, since gathering the same data can be cumbersome. An algorithm, the cluster ensemble in this case, is therefore utilized twice. First, it is assessed on the training data. In most cases, the training data is about $\geq$ 60% of the original size. Meaning that an algorithm is always "overfit" with the training data. The second time an algorithm is used only once on the testing data, constituting around $\leq$ 30% of the original size. This time the algorithm is not "overfit", as less data is presented. Analogously, the true accuracy of an algorithm can be tested, hence the one-time usage of the testing data. In the context of our study this could mean; cluster observations with the training data and determine if the same clusters are to be found in the testing data. However, since we do not know the number of clusters a priori for our data, we only continue with the training data.

3.4.2 The generation stage

As discussed previously in chapter 2, the strength of the cluster ensemble lies within diversity. As counter intuitive this may sound, good diversity is often exploited by picking the "wrong" algorithms as each algorithm will make different assumptions about the shape of the data. Thus, hidden clusters in the data may be found by one algorithm, where others fail. Hadjitodorov and Kuncheva (2007) studied some important heuristics that define diversity for the cluster ensemble. The most contributing heuristics from that study are reported here in a summarized fashion.

Sampling of the observations

Sampling different subsets of observations is a robust method to create diversity in the partitions of the cluster ensemble (Minaei-Bidgoli, Topchy, and Punch, 2004). This allows to use a part of the data each time a new partition is formed. The basis is that a complete dataset is divided randomly, and that each iteration eventually will give other clusters. Two sampling methods are applicable in the generation stage. These are the subsample and the bootstrap method. The subsample method randomly selects a handful of observations, say 70% of the total size, and each in each partition the whole dataset is again split in a 70-30 rule. This type of subsample is also known as the *validation set* to test a classifier (James et al., 2013). On the other hand the bootstrap method is, in the domain of statistics, a robust sampling method (Efron, 1992). It replicates the data by repeatedly sampling observations with replacement. In practice, it turns out that the bootstrap method is quite salient in the supervised domain (James et al., 2013). Turning it to the unsupervised domain this method works quite well to determine cluster validity and creating diversity for the cluster ensemble (Minaei-Bidgoli, Topchy, and Punch, 2004). Having a large dataset, however, makes the bootstrap method computational expensive. Thus, robust clustering results from the cluster ensemble can not always rely on the assumption of various bootstrap samples. Instead, sometimes a small subsample, like the validation set, can be enough to determine the total amount of clusters in a dataset with an error rate that is competitive to that of the bootstrap (Topchy et al., 2004). Indeed, Minaei-Bidgoli, Topchy and Punch (2004) show that sometimes only a small fraction of the data suffices to detect cluster structure of an entire dataset. Upfront facing the cluster ensemble method however, it is sometimes unknown which method works best. We argue, as in line with the discussion of Minaei-Bidgoli et al. (2004), that as a rule of thumb the size of the dataset matters which sampling method is to be chosen.

Different algorithms and parameter settings

Something that is considered as a gold standard for diversity is the use of different algorithms and their parameter settings during the generation stage (Topchy, Jain, and Punch, 2005; Kuncheva and Vetrov, 2006). This boils down to the fact that there are no layman rules in selecting the right clustering algorithm (Hadjitodorov and Kuncheva, 2007). Thus, picking an inappropriate clustering algorithm is sometimes easily done, and since there is no response value to match against, it remains arbitrary whether the algorithm suffices in its context (Hadjitodorov, Kuncheva, and Todorova, 2006). Therefore, instead of running into the problem of selecting the wrong clustering algorithm, one can turn to the cluster ensemble (Ghosh and Acharya, 2011). The presumption is that even the simplest cluster ensemble outperforms a randomly chosen clustering algorithm (Hadjitodorov and Kuncheva, 2007). Moreover, even if only one clustering algorithm is used the ensemble is determined to be more consistent in its results (Kuncheva and Vetrov, 2006).

Some early work in the cluster ensemble domain already advocated the use of multiple algorithms to obtain robust results for the cluster ensemble (Fred and Jain, 2002; Strehl and Ghosh, 2002; Monti et al., 2003). This lead to studies that explored the efficacy of the cluster ensemble with single algorithms (Fred and Jain, 2002) and with different parameters (Topchy, Jain, and Punch, 2003; Fred and Jain, 2002; Kuncheva and Vetrov, 2006; Fern and Lin, 2008). The landscape of different parameters usual consists of a wide range in k, multiple (dis)similarity metrics, and many random initializations of the same algorithm. In addition, the efficacy of the cluster ensemble has also been studied extensively by using multiple algorithms at once (i.e. K-means, HC etc.) (Kuncheva and Hadjitodorov, 2004; Hadjitodorov and Kuncheva, 2007; Monti et al., 2003). A comparative study from Fred and Lourenco (2008) show that using multiple algorithms reports better results than the single algorithmic setup.

Subset of the features

Another strategy worth noting, but is not used in this study, since it is unavailable at the time being, is the use of random feature selection. Apart from RP, which is discussed in section 3.3.4, random feature selection uses the whole feature space. Greene et al. (2004) explored this idea by using *random sub-spacing* to built partitions. Each partition uses a different subset of features on each clustering run in the generation stage, allowing to find many different clusters in the data. In a somewhat similar fashion, Hadjitodorov and Kuncheva (2007) adapted this idea whilst incorporating a *genetic algorithm* to randomly create different subset of features that are included in each partition. Both studies report that their findings indeed improved the final clustering result.

3.4.3 The consensus stage

Popular consensus functions used in different studies deal with object co-occurrence of the cluster ensemble, for example, in Fern and Lin (2008), Topchy et al. (2004), Ayad and Kamel (2008), and Ghosh and Archaya (2011). Other consensus functions are also provided in an overview in section 2.4.2. Here, a highlight is given of the three most used and effective consensus functions based on the object co-occurrence, which are Majority Voting, CSPA and LCE.

CSPA

The CSPA transforms the cluster similarity matrix, which consists of cluster labels, to a similarity graph representation, see also Figure 2.5 in Chapter 2. CSPA is a simple, but effective consensus function that defines similarity only when two observations share the same cluster (Strehl and Ghosh, 2002). The objects in the similarity graph are represented by vertices, meanwhile the edges are the weights based on similarity. Thus, ties are broken when vertices have edges with a low weight, since similarity is absent. This process is often done with a graph partitioning algorithm, such as spectral clustering (Fern and Lin, 2008). Although CSPA can be a computational expensive consensus function, since it is dependent on the number of observations, it enforces the final clusters to be near equal sizes (Ghosh and Acharya, 2011).

Majority Voting

The majority voting process, or cumulative voting as by Ghosh and Acharya (2011), considers that each partition holds cluster labels, which are assigned to a similarity matrix. Consequently, a co-association matrix is formed that intuitively counts the number of cluster assignments between two observations (Kuncheva and Hadjitodorov, 2004). This results in a *winner takes all* fashion, in which the observation gets assigned to a cluster if it has the majority of a vote. A similar technique is also applied in the evidence accumulation strategy by Fred and Jain (2002).

LCE

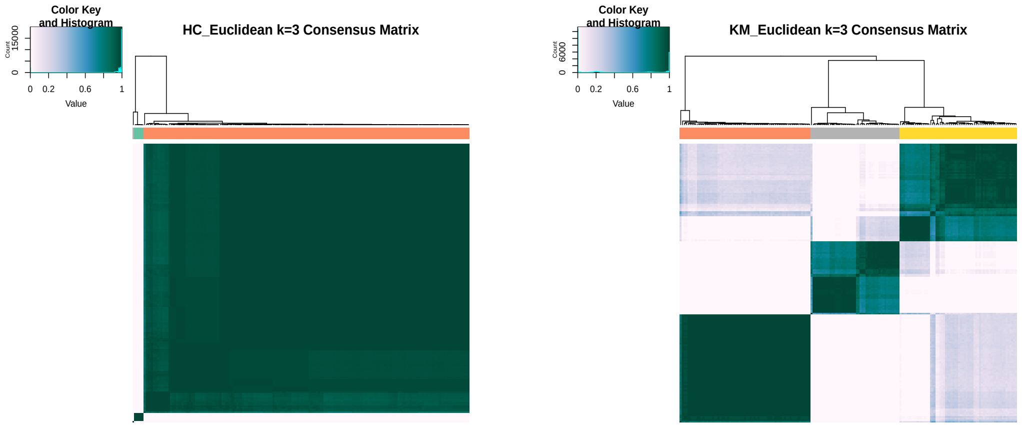
LCE is an extension to the graph-based technique from Fern and Brodley (2004), by applying a graph-based consensus function to an improved cluster association matrix (Iam-On et al., 2010). Consensus functions, such as, Majority Voting and CSPA in the latter, work with an co-association matrix in a binary format (i.e. one means similarity and zero dissimilarity), LCE employs a probabilistic format. Thus, each observation has a certain membership to a cluster label. However, as Iam-on et al. (2010) note, each partition with observations without a cluster label limits the quality of that certain partition. In the current schema's of the object co-occurrence functions, this is neglected. To calculate the probability that a cluster shares some observations within another cluster, which are considered dissimilar in the other consensus functions, LCE employs the so-called *weighted connected-triple*. Essentially, it measures the probability that two dissimilar clusters still share some observations through a mutual connected cluster. This enforces the consensus function to be more robust in deciding which cluster label belongs to a certain observation.

3.4.4 Interpreting the first cluster ensemble visualizations

Since many partitions are created with diversity in the cluster ensemble setting, output needs to be examined for which *k* and algorithm, clustering is consistent. Indeed, as discussed in Chapter 2, there is no gold standard for clustering. Thus, among the multiple algorithms in the ensemble there will be poor performing ones. Therefore, the cluster ensemble provides a heatmap, and distribution functions, such as the AUC (Area Under the Curve operator) and CDF (Cumulative Distribution Function), which guide researchers to determine the optimal result via visualization.

Clustering heatmap

The consensus function obtains a similarity matrix, which indicates the observations that are clustered together. As a result, observations with the highest consensus (i.e. similarity index) are represented in a dendrogram. From this dendrogram adjacency can be induced to other clusters, which maximizes the block-diagonal structure of the heatmap that results from the dendrogram (Monti et al., 2003). For example, consider the two heatmaps in Figure 3.7. Heatmap B, shows a block-diagonal structure that is slightly fragmented at the upper left-side and right-down side, with vague green blocks. Heatmap A, shows one big block-diagonal structure, and two small blocks at the left. From both heatmaps we can tell that there are three clusters, but each having a different size, indicated by the size of the block. However, we know the true size of each cluster, so we can determine that heatmap A does not cluster this dataset very well. We can confirm this by inspecting the distribution functions from the cluster ensemble.



(A) Heatmap HC

(B) Heatmap K-means

FIGURE 3.7: (A): Heatmap for $k = 3$ with HC. (B): Heatmap for $k = 3$ with K-means.

Cluster distribution functions

From both AUC and CDF, exploration can continue in determining the optimal result. With AUC a delta area is illustrated, which shows the relative improvement from the cluster result with respect to the best result of k. Thus, as Figure 3.8 is displaying, there is a decrease in cluster improvement for the dataset after $k = 3$, indicating that observations are not increasing in similarity. Thus, adding more clusters will not improve the outcome. Moreover, we observe that HC_Euclidean is showing a higher AUC value. This value can be translated back to the heatmap from Figure 3.7. Indeed, the biggest block on the left-hand side from Figure 3.7 shows high similarity of observations clustered into one cluster. As such, a higher decrease is expected if this block is broken down to a smaller cluster when $k = 4$ is chosen.

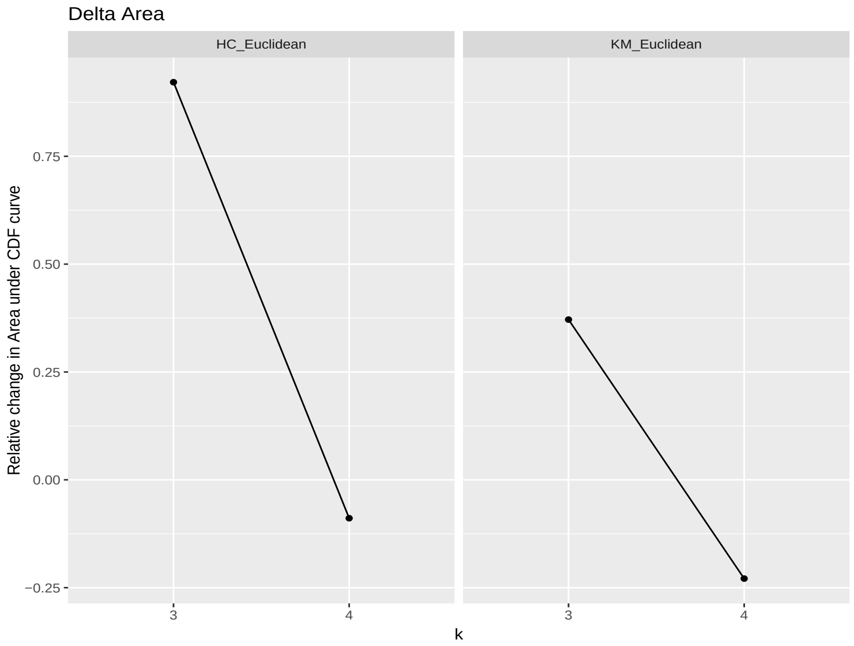


FIGURE 3.8: Cluster ensemble Area Under the operating Curve showing cluster performance decrease after $k = 3$.

Last, the CDF displays the "stability" of forming those clusters for each algorithm, based on the value of k. In layman terms, a flat line from $[0,1]$ indicates this stability, which we observe for the HC_euclidean plot in Figure 3.9. By looking back to the heatmap from Figure 3.7, we can see that this is the big block, followed by the stepwise increase as the CDF approaches 1, meaning that these steps indicate the smaller blocks formed in the heatmap. For KM_euclidean we see a different pattern. First, an increase in the step function, then a straight line, followed by another smaller increase as it approaches 1, which indicates that during the clustering process improvement was made to cluster observations. Hence, the three nearly equal sized blocks in the heatmap from Figure 3.7, and since we know the true nature of k, we can tell that K-means did a better job at finding those clusters. Thus, the goal with inspection is to find the k that has a maximized concentration among observations that are clustered together, and visualized with the CDF and AUC, displaying stability of the cluster performance (Monti et al., 2003).

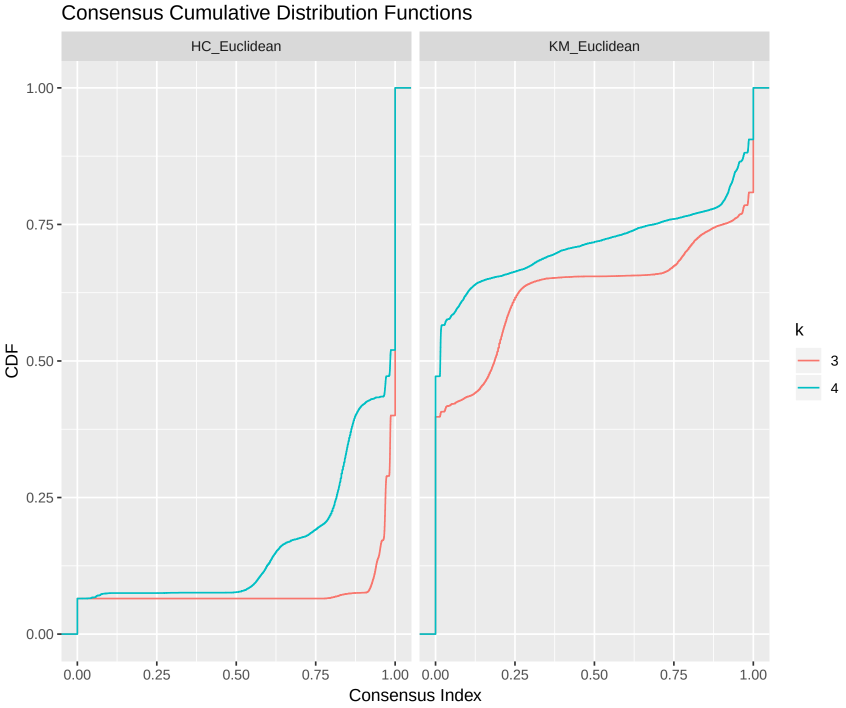


FIGURE 3.9: Cluster ensemble Cumulative Distribution Function showing cluster stability between two algorithms and two k values.

3.4.5 Evaluation: internal validity indexes for cluster validity

Since clustering does not work with a response variable, it is required to measure the quality of the clustering result. Measuring the result provides insight in how the algorithm performed, and provides guidance for researchers to select the most appropriate algorithm, or k, that obtained the best result (Naldi, Carvalho, and Campello, 2013). Otherwise known as the *validity of goodness* or *goodness of fit*, which requires a validation index that measures whether a certain partition obtained from the clustering is the overall best partition out of a set of partitions (Maulik and Bandyopadhyay, 2002).

Choosing the "right" validation index is, however, considered as a difficult task. Milligan and Cooper (1985) addressed this already more than two decades ago. Their study, albeit with some methodological flaws, which are addressed by Vendramin et al. (2010), found that each validity index is influenced by the data. Milligan and Cooper (1985), therefore, urged that each validity index in consideration should be interpreted with caution. Some studies that followed after Milligan and Cooper’s work, faced somewhat the same problem. Their work is not contradictory to that of Milligan and Cooper (1985), although their experiments considered other clustering algorithms, different validity indexes, and a different methodology to decide upon the quality of the validity index (Maulik and Bandyopadhyay, 2002; Liu et al., 2010; Vendramin, Campello, and Hruschka, 2010; Naldi, Carvalho, and Campello, 2013).

More recent work by Arbeleitz et al. (2013), is an extension upon the work of Milligan and Cooper (1985), with a contemporary approach by exploring more datasets (both real and synthetic), more clustering algorithms, newer validity indexes, and an improved framework to assess the quality of end results per validity index. The authors came with a non-controversial answer; each internal validity index is variant to the data at hand. Despite that there is not a single best validity index, the authors argue that among the 40 validity indexes examined, the Silhouette index, Davies-Bouldin index, and Calinski-Harabasz index, have proven to be the most stable ones (Arbelaitz et al., 2013). Being considered as stable means that the validity index did not change significantly in their end result when facing some data related challenges. For example, when noise is added, dimensionality is increased, and clusters started to overlap. Last, note that based on the results by Arbelaitz et al. (2013), cluster overlap always heavily affects the result of each cluster validity index.

From the work of Arbeleitz et al. (2013), three of the most promising internal validity indexes are discussed. Formal detailed descriptions of these indexes are omitted, but each index will contain a reference to their respected study that contains the detailed mathematical formula.

Silhouette

The silhouette index is both a visualization technique and metric for partitional clustering algorithms (e.g. the K-means discussed in Chapter 2). Rousseeuw (1987), introduced a way to measure compactness and separateness of clusters, while displaying the outcome via a visual approach. It only needs two requirements: 1. partitions obtained by the algorithm (i.e. the clusters), and 2. all the proximities between the objects. Based on these two requirements the silhouette index measures the pairwise difference of intra-cluster and inter-cluster distances. In short, the average distance from the cluster centroid to all within-cluster observations is measured (i.e. intra-cluster distance, or compactness). Second, the average distance from its centroid to another cluster centroid is also measured (i.e. the inter-cluster, or separateness). In addition, it also computes the distance to its closest neighbor. That is, if the observations in one cluster are close to another cluster, then probably this is the second best cluster to be used if the first cluster is discarded (Rousseeuw, 1987). Finally, the silhouette index that has a maximized value, between $[0,1]$, is chosen to be the best partition for the data.

Calinski-Harabasz

The Calinksi-Harabasz index computes both the minimum of the within-group sum of squares, and the maximum of between-group sum of squares of a dataset. In detail, the sample means of each centroid cluster, and the overall sample means of the data are taken. Consequently, a sum-of-squares matrix is created in which the sum of intra-cluster variances, and the sum of inter-cluster variances are stored. Analogously, this matrix can be used to sort clusters based on low average intra-cluster variance, and high average inter-cluster variance to measure compactness and separateness (Caliñski and Harabasz, 1974). Both metrics result in a ratio-scale that is maximized when both compactness and separateness score high. Last, the Calinski-Harabasz is an optimization index, which means that if the number of clusters grow it does not affect the outcome of the index (Naldi, Carvalho, and Campello, 2013).

Davies-Bouldin

The Davies-Bouldin index is somewhat related to the aforementioned method, the Calinski-Harabasz index. This index also takes the intra-cluster and inter-cluster variances into account (Naldi, Carvalho, and Campello, 2013). For each cluster in a dataset its similarity is compared to one another. Thus, based on the density of clusters, a decreasing similarity function can be calculated, since low average similarity between clusters indicate a strong separateness (Davies and Bouldin, 1979), however. In contrast to the aforementioned methods, the Davies-Bouldin requires a minimum index value. The lower the value the stronger the distinctness between clusters, hence the better clustering result (Liu et al., 2010).

3.4.6 Evaluation: external validity indexes for cluster validity

In contrast to internal validity indexes, external validity indexes require an auxiliary function (i.e. a reference class), which measures precision of the clustering algorithm. For instance, a dataset can be clustered, and the obtained clusters can be compared with the auxiliary function to test how well the clustering algorithm performed. In other words, the reference class can be seen as a response variable for the unsupervised setting. Moreover, with an external validity index it is simpler to determine if the clusters are acceptable or not (Kovács, Legány, and Babos, 2005). An example is the confusion matrix, which is a convenient index to use, since it captures "goodness", often named accuracy, of the clustering algorithm (Dhillon, 2001).

Apart from the many internal validity indexes, there also exist numerous external validity indexes. Here, three external validity indexes are discussed, whereof two are extensively used in the cluster ensemble setting. These two are the Jaccard index and Normalized Mutual Information (NMI). Both indexes are explored in various studies focusing on the cluster ensemble (Kuncheva and Vetrov, 2006; Strehl and Ghosh, 2002; Fern and Brodley, 2003; Kuncheva and Hadjitodorov, 2004). The last external validity index is the confusion matrix. Although less used, it still gives a simplified overview of the accuracy of the cluster ensemble.

Normalized Mutual Information

Normalized Mutual Information (Strehl and Ghosh, 2002), is an external validity index, which compares the obtained partition to the clusters from the original partition (i.e. the reference class). The end result of NMI is a selected single partition that is most similar to the reference class. Therefore, NMI measures the mutual information shared between the clustering result, and the reference class. To measure mutual information, the probability is taken that a certain cluster belongs to a class. For instance, cluster X represents reference class Y, with a fixed amount of n observations. As such, NMI is based upon the cluster entropy, which means that the actual outcome is calculated against the predicted outcome. However, as NMI normalizes entropy, and therefore ranges between $[0, 1]$, it is invariant to the number of clusters (Fern and Brodley, 2003). Thus, in the cluster ensemble setting NMI allows to compare different partitions with varying number of clusters.

Jaccard

The Jaccard index measures the (dis)similarity of observations between pairs of partitions. Thus, the Jaccard index can be seen as a counting pairs measurement that provides a certain agreement between the clusters obtained (Vega-Pons and Ruiz-Shulcloper, 2011), instead of an information theory measurement, such as NMI. For example, consider having two clusters, C_i and C_j, from two partitions. The disagreement between both clusters is measured by counting the number of observations that both clusters do not share, with respect to the reference class. As such, it gives an estimate of the precision for clusters found in the dataset. Subsequently, this can lead to determine the optimal number of clusters, as shown by Ben-Hur et al. (2001). Moreover, Ben-Hur et al. (2001) explain that the Jaccard index is useful when sub-samples of a dataset are used, via bootstrapping, to form clusters. Thus, making it a suitable external validity index for the cluster ensemble, since bootstrapping is an effective technique to acquire diversity.

Confusion matrix

The confusion matrix is a simple cross-table, showing results from the clustering process compared with the reference class. Although the confusion matrix stems originally from the supervised domain, by predicting accuracy of the classifier, it can also be used to determine accuracy of a clustering algorithm. For example, since observations are clustered, it is easy to cross-tabulate which observations are clustered correctly in the specified reference class label. Moreover, it also shows which observations are clustered in the wrong reference class label. Based on these two conditions, the accuracy and error rate can directly be deducted from the confusion matrix, for example, $\mathbf{err} = 1 - acc$, and the accuracy is, $\mathbf{acc} = correctobservations/(correctobservations + falseobservations)$.

3.4.7 Expert evaluation and cluster visualization

In addition to the previous section, external validation can also be driven by domain knowledge (i.e. expert evaluation). A preliminary requirement is that clusters are visualized, since this aid in human perception to understand the clustering results (Halkidi, Batistakis, and Vazirgiannis, 2002). Combining both can result in additional background information that aid in understanding each cluster. Hence the class discovery, and data understanding aspect that is subject to cluster analysis.

Expert evaluation, driven by domain knowledge, can aid in understanding clusters derived from data. In essence, it is a goal within the CRISP-DM cycle to evaluate the results with experts, and not solely rely on the validity indexes resulting from the model (Osei-Bryson, 2010). Halkidi et al. (2002) describe this as the interpretation part, which follows after validation of the results. The evaluation part can be explored in various ways, such as expert external validation, as discussed by Osei-Bryson (2010), or by dialog, as advocated by Chapman et al. (2000).

Visualization, on the other hand, is often used to visualize clusters. Especially when datasets consist of more than three variables, since in higher dimensional spaces clusters become difficult to interpret for humans (Halkidi, Batistakis, and Vazirgiannis, 2002). To visualize clusters in these higher dimensional spaces, some feature extraction is necessary. For instance, PCA, Multi Dimensional Scaling (MDS) (Kruskal, 1964), or t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van Der Maaten and Hinton, 2008), all embed observations to lower dimensional space. MDS and t-SNE, in particular, uses a non-linear approach, and preserve the original distances between the observations. Useful when a proximity metric is used to structure observations. Moreover, clusters can also be characterized, and visualized based on their underlying features. Henry, Tolan and Gorman-Smith (2005) argue that once observations are clustered, a simple mathematical formula, such as the mean, can be used to separate and identify unique clusters.

3.5 Summary

The CRISP-DM is the underlying framework for development of a cluster ensemble meta-algorithmic model. For that reason, several aspects of this framework are highlighted, such as data understanding towards data modeling and evaluation, with clustering as a base. Therefore, data understanding aims at visualizing densities within a dataset with multivariate kernel density estimates. In addition, variances between observations and features, and a preliminary cluster structure of the data is visualized with a heatmap and dendrogram. Moreover, data preparation focuses on the pragmatic use of techniques to prepare data for the modeling phase with clustering. For instance, rank normalization to best preserve distances between observations, and RP to reduce dimensionality without sacrificing the distances that exists between observations. The data modeling phase concerns the utilization and evaluation of the cluster ensemble. That is, the splitting of the data between a train and test set, the exploitation of diversifying the cluster ensemble by bootstrapping, and multiple algorithms, towards the consensus functions, such as Majority Voting and CSPA. Furthermore, evaluation of the cluster ensemble focuses on internal validity indexes, such as the silhouette, and external validity indexes, such as the confusion matrix. Last, expert evaluation and cluster visualization are both aimed at understanding the clusters derived from the data modeling phase. Hence, the closing part of data modeling and evaluation.

Chapter 4

Cluster ensemble: Meta-Algorithmic Model fragments

This chapter presents the cluster ensemble meta-algorithmic method fragments, and their related activities and concepts for the cluster ensemble, by following the outline that is set in Chapter 3. As discussed earlier, these method fragments are based on a recent study by Spruit and Jagesar (2016), which in turn follows the CRISP-DM cycle of Chapman et al. (2000), and the method engineering technique from Brinkkemper and Weerd (2009). Thus, this work expands upon previous work by introducing new meta-algorithmic method fragments for the unsupervised domain.

Section 4.1 presents domain independent meta-algorithmic method fragments, which form the cornerstone for a data mining project focused on using the cluster ensemble. Moreover, section 4.2 illustrates several domain specific handling tasks applied at the Psychiatry Department of the UMCU. Note that for each task in section 4.2, the general method fragments from section 4.1 served as a guide. Last, section 4.3 concludes this chapter, and therefore, providing an answer to the first research question of this study. The complete general meta-algorithmic model can be found in Appendix B.

4.1 General meta-algorithmic method fragments

4.1.1 Data understanding

As stated in Chapter 3, data understanding is an important preliminary step for the sequential phases that follow in the CRISP-DM framework. It is aimed at providing insight to the dataset in question, hence giving notion to the steps needed to be taken in the next phase. Creating different visualizations aid in grasping some understanding of the data.

As explained earlier in Chapter 3, KDE's are interpretable plots, which look somewhat similar to scatter plots. Thus, relationships between variables, and spreading of observations can easily be distinguished. However, the number of features in a dataset limits this technique. Therefore, to circumvent this problem a heatmap with a dendrogram can be explored. Although, this might not give us the sophistication by showing the spread of the observations as in KDE's, it does give an insight in the variance of features, and thereby giving some decisions to make in the data preparation and data modeling and evaluation phase. In Figure 4.1 the first meta-algorithmic method fragment for data understanding is displayed, followed by the activity table in Table 4.1, and concept table in Table 4.2.

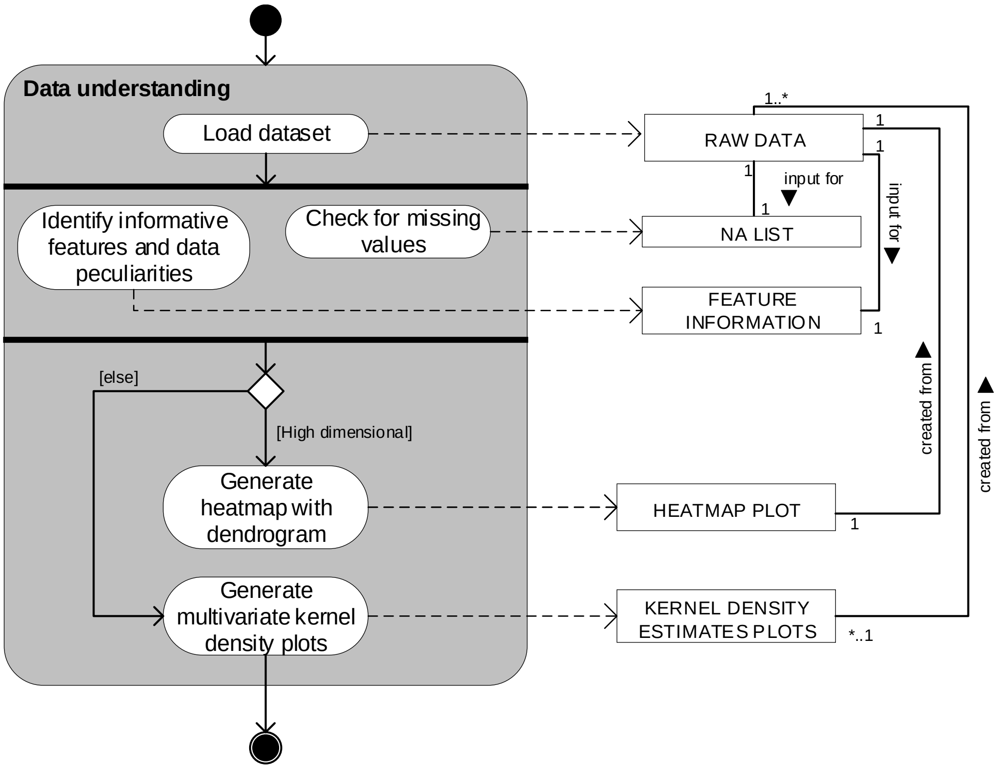


FIGURE 4.1: meta-algorithmic method fragment for Data Understanding.

TABLE 4.1: Data Understanding activity table.

Main activity	Sub-activity	Description
Data Understanding	Load dataset	The data is loaded into the working environment, creating the RAW DATA to be explored.
	Identify informative features and data peculiarities	A researcher identifies which features are of interest and which features can potentially be removed or merged. In addition, data inconsistency is identified at this moment. Together this will provide a summary in the form of FEATURES LIST which is called upon in the Data Preparation phase.
	Check for NA Values	Next to attaining understanding about the features and observations, the researcher also identifies whether features have missing values. This is a necessary step, since not every clustering algorithm nor feature extraction technique is capable in handling missing values. The end result is an overview of missing values in a NA LIST.
	Generate heatmap with dendrogram	If there are many features the data set should be considered as high dimensional. Therefore, the HEATMAP PLOT should be created, since this is suitable for high dimensional data sets.

Continued on next page

Table 4.1 – continued from previous page

Main activity	Sub-activity	Description
	Generate multivariate kernel density plots	Else, when there are a few features in a dataset the researcher should built KERNEL DENSITY ESTIMATES PLOTS, since this gives an understanding between groupings of observations between multiple features.

TABLE 4.2: Data Understanding concept table.

Concept	Description
RAW DATA	The RAW DATA is the original dataset containing all features and observations.
NA LIST	The NA LIST is an overview that identifies missing values of observations per feature.
FEATURES LIST	FEATURES LIST is a summary of informative features needed for the sequential phase, meanwhile listing peculiarities (i.e. conversion errors in the data.)
HEATMAP PLOT	A HEATMAP PLOT is a visualization of variances between features and observations, meanwhile structuring observations by features with a dendrogram.
KERNEL DENSITY ESTIMATES PLOTS	KERNEL DENSITY ESTIMATE PLOTS show multiple visualizations, based on the total number of features, and based on density between observations.

4.1.2 Data preparation

The data preparation phase consists of five situational steps. Thus, it depends on the dataset which steps are needed to be taken. However, raw data is in general never complete, nor in the right format. Therefore, *feature selection and engineering* is most often performed in the data preparation context. In addition, *feature scaling* is another general step that follows in the data preparation format, since features are often recorded differently. This step normalizes the features, preferably by range, as discussed in Chapter 2, while keeping the distances between observations intact.

Feature transformation entails some steps that are aimed at: 1. lowering the number of features, and 2. creating a matrix of transformed features that best describe the data with less features than the original dataset. For this step, techniques like RP or PCA are common.

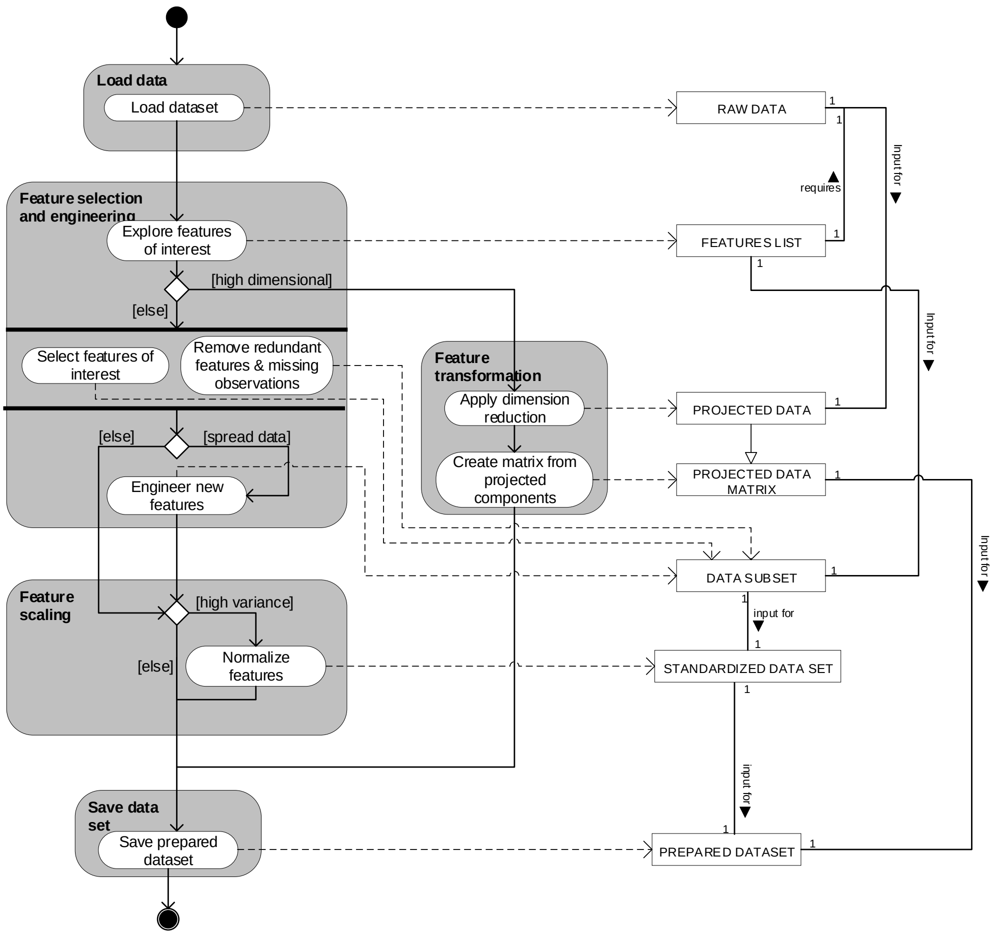


FIGURE 4.2: meta-algorithmic model fragment for Data Preparation.

TABLE 4.3: Data Preparation activity table

Main activity	Sub-activity	Description
Data Preparation	Load dataset	The data is loaded into the working environment, creating the RAW DATA to be explored.
Feature selection and engineering	Explore features of interest	The FEATURES LIST acts as a guide for the researcher to determine whether the RAW DATA requires some engineering or transformation.
	Select features of interest	If dimensionality does not pose a problem, the researcher can determine which features are of interest for the Data Modeling phase. Thus, informative features are included in the DATA SUBSET.
	Remove redundant features and missing observations	The RAW DATA contains often uninformative features and missing values. Therefore, the researcher removes features that are not of interest and missing values for observations, since both affect clustering results.

Continued on next page

Table 4.3 – continued from previous page

Main activity	Sub-activity	Description
	Engineer new features	Data can be spread throughout the RAW DATA, which necessitates the need to engineer new features. These newly engineered features are simply features that are either merged or transformed so that they are informative, and thus included in the DATA SUBSET.
Feature transformation	Apply dimension reduction	A feature transformation technique is applied which strips the uninformative features, as in PCA, or transposes the features to low-dimensional subspaces leaving the Euclidean distance intact via RP. The result from both techniques is a dataset that is reduced in dimensionality. Therefore, this activity can be vital in certain circumstances where dimensionality poses a problem for the cluster ensemble.
	Create matrix from projected components	PCA, for example, does not automatically give a matrix which is suitable for clustering. Therefore, the required principal components are extracted from PCA and fit into a data matrix.
Feature scaling	Normalize features	Depending on the DATA SUBSET and implicitly on the RAW DATA, if high variance between features is present the features should be normalized.
Save dataset	Save prepared dataset	After going through the several Data Preparation steps, the prepared dataset is saved. This serves as an input for the upcoming Data Modeling and Evaluation phase.

TABLE 4.4: Data Preparation concept table.

Concept	Description
RAW DATA	The RAW DATA is the original dataset containing all features and observations.
FEATURES LIST	The FEATURES LIST is used in the Data Preparation phase to decide whether the dataset requires engineering.
PROJECTED DATA	If dimensionality poses a problem, features can be transformed by applying a dimension reduction technique (i.e. RP or PCA), which result in PROJECTED DATA.
PROJECTED DATA MATRIX	From the previous concept, a PROJECTED DATA MATRIX is created. This matrix is suitable to be used in the Data Modeling and Evaluation phase.
DATA SUBSET	The DATA SUBSET is a dataset containing all informative features, and observations needed for the Data Modeling and Evaluation phase.
STANDARDIZED DATASET	The STANDARDIZED DATASET normalizes the variances between features, so each feature is treated as equal in the Data Modeling and Evaluation phase.
PREPARED DATASET	The PREPARED DATASET finalizes the Data Preparation phase, since this is the dataset being used as an input for the Data Modeling and Evaluation phase.

4.1.3 Data modeling and evaluation

The last general meta-algorithmic method fragment is the Data Modeling and Evaluation fragment. The main focus is on the steps needed to: 1. form clusters with the cluster ensemble, and 2. evaluate the outcome with both an internal and/or external validity index and expert evaluation. The method fragment has five main steps that are needed to derive the clusters, and to evaluate the outcome. The first main step is the *generation stage*, which involves setting the required parameters to diversify the partitions made by each repetition of a single clustering algorithm, in order to build a portfolio of different clusters. The second step aims at selecting the *consensus function*. The third, fourth, and fifth step are aimed at evaluating and visualizing the outcome.

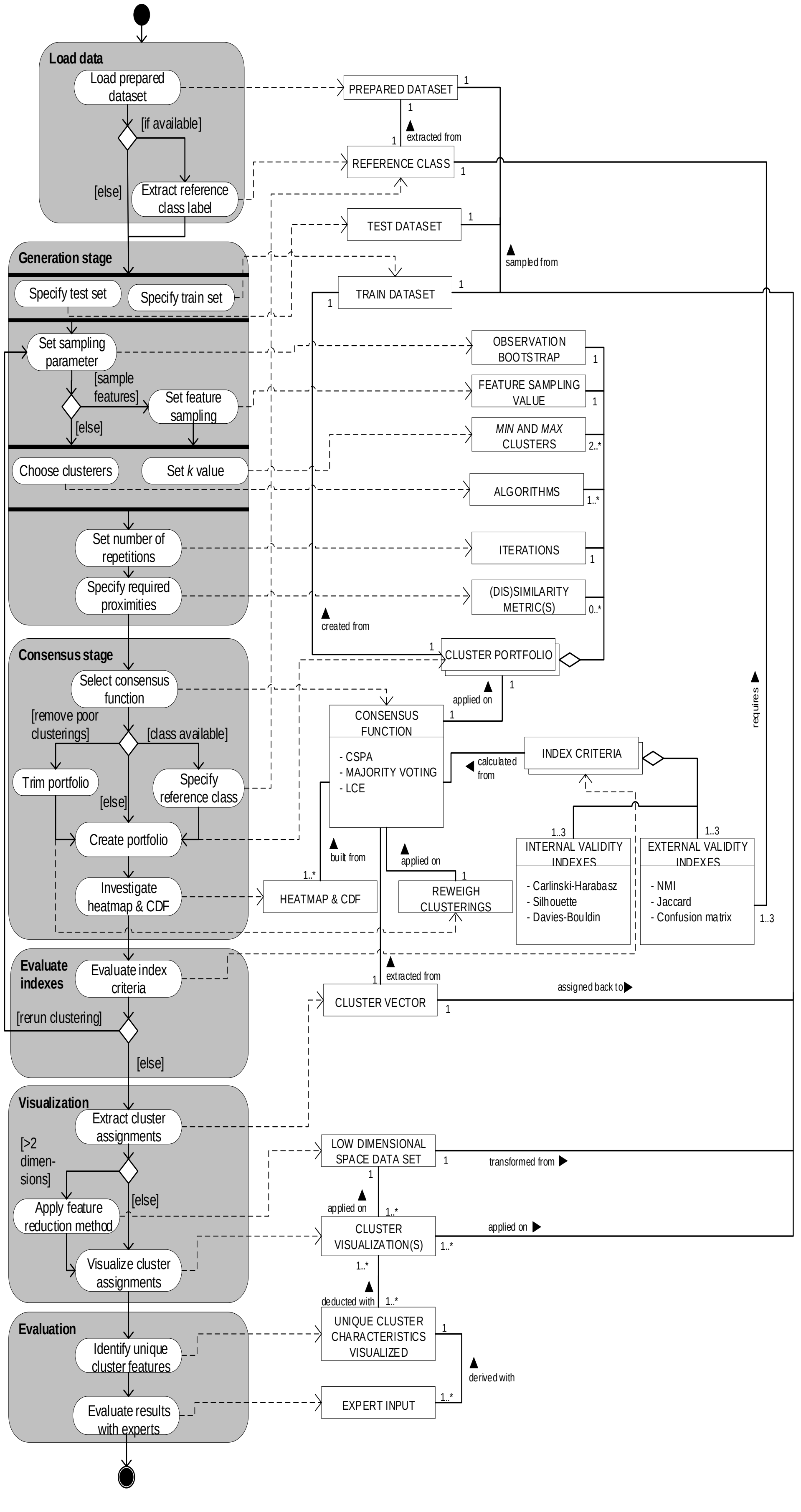


FIGURE 4.3: meta-algorithmic method fragment for Data Modeling and Evaluation.

TABLE 4.5: Data Modeling & Evaluation activity table.

Main activity	Sub-activity	Description
Load data	Load prepared dataset	The prepared data is loaded into the working environment, creating the `PREPARED DATASET` to be explored.
	Extract reference class	The reference class is extracted from the `PREPARED DATASET`, and is separately stored in a vector.
Generation stage	Specify train set	The `TRAIN DATASET` is split by a certain percentage (i.e. >60%) from the `PREPARED DATASET`.
	Specify test set	The `TEST DATASET` is a smaller and different split from the `PREPARED DATASET` (i.e. <30%), and is only used once after the training of the cluster ensemble is completed. The steps below that follow are also applicable to the `TEST DATA SET`.
	Set sampling parameter	The percentage of `OBSERVATION BOOTSTRAP` is specified which guides each `ITERATION`, and `ALGORITHMS` to use a specific amount of observations to use for the clustering process. Thus, some observations are held out in each separate run.
	Set feature sampling value	Following a *bagging* process each feature is included or excluded in the clustering run, diversifying the cluster ensemble process even more.
	Choose clusterers	The researcher defines which clustering algorithms are used in the generation stage, hence `ALGORITHMS`.
	Set *k* value	The researcher sets the parameter to a specified number of clusters to be found in the dataset. Note, the *minimum* number should be 2. The *maximum* number is often defined with the rule of thumb; take the square-root of the total number of observations. The main idea is to *over-cluster* the dataset.
	Set number of repetitions	This parameter value is used to set the number of repetitions, or `ITERATION`, for the cluster ensemble to repeat for each algorithm, proximity metric, and *k* cluster. Typically this number is set to a minimum of 200. With repetitions of $\geq$ 1000 the increase in accuracy is marginal against the cost in computational power.
	Specify required proximities	Some `ALGORITHMS` require a proximity matrix (e.g. hierarchical clustering types). This activity allows the researcher to define which proximity metric(s) should be used.
Consensus stage	Select consensus function	After the generation stage, the `CONSENSUS FUNCTION` is selected, allowing the researcher to choose between three object co-occurrence methods. These are: 1. Majority Voting, 2. CSPA, or 3. LCE.

Continued on next page

Table 4.5 – continued from previous page

Main activity	Sub-activity	Description
	Trim portfolio	This optional parameter allows the researcher to automatically remove weak clusterings (i.e. clusterings with weak results based on internal index criteria). As a result, the final partition in the latter, `CLUSTER PORTFOLIO`, is often optimized because spurious results are removed.
	Specify reference class	This optional parameter allows the researcher to use the original `REFERENCE CLASS` in order to calculate `EXTERNAL VALIDITY INDEXES`. This is useful when accuracy needs to be determined for the cluster ensemble.
	Create portfolio	If the required and/or optional activities in the latter have been performed, the cluster ensemble is ready to create clusterings and build the `CLUSTER PORTFOLIO`.
	Investigate heatmap & CDF	Results from the `CLUSTER PORTFOLIO` are visualized over multiple heatmaps, and distribution functions, depending on the parameter settings of *`MIN`* `AND` *`MAX`* `CLUSTERS`, `ALGORITHMS` and `(DIS)SIMILARITY METRIC(S)`. From these plots the researcher can determine the algorithm and k, which is the most optimal one for the `TRAIN DATASET`.
Evaluate indexes	Evaluate index criteria	The researcher extracts the index criteria from the `CONSENSUS FUNCTION`. This provides the researcher the opportunity to determine whether the cluster ensemble returned clusters with an optimum result. If not, then the cluster ensemble can be utilized again with different parameter settings, starting from the generation stage.
Visualization	Extract cluster assignments	The following step after creating the `CLUSTER PORTFOLIO`, and investigating the outcomes, is to extract the final cluster assignments, and store these in a separate `CLUSTER VECTOR`.
	Apply feature reduction method	When the `TRAIN DATASET` has more than three dimensions, the data needs to be reduced in order to visualize it. This requires the researcher to apply a feature reduction method. Several are available, from PCA and t-SNE, to MDS.
	Visualize cluster assignments	The researcher now needs to visualize the cluster assignments, in order to see how the `TRAIN DATASET` is clustered. It requires to merge the `CLUSTER VECTOR` with the `TRAIN DATA SET` to visualize the clusterings. This step is vital in order to check for: 1. arbitrarily shaped clusters and 2. spurious clusters.

Continued on next page

Table 4.5 – continued from previous page

Main activity	Sub-activity	Description
Evaluation	Identify unique cluster assignments	When clusters are visualized, it is often useful to determine which features from the TRAIN DATASET make up the clusterings. For instance, the researcher can use a radar chart or bar graph, which identifies the features that have higher loadings on each cluster assignment. From a more complicated machine learning perspective, the researcher can also apply unsupervised *Random Forests* on the CLUSTER VECTOR that is merged with the TRAIN DATASET. A result is that each feature loading can be extracted.
	Evaluate results with experts	The final activity is aimed at gathering domain knowledge for the clusters. The goal is to give means to each cluster. Experts in the field can be used to help evaluating, and to describe the clusters. For this activity, the researcher can use focus groups or interviews with experts.

TABLE 4.6: Data Modeling & Evaluation concept table.

Concept	Description
PREPARED DATASET	The PREPARED DATASET is the dataset containing only the informative features and complete observations that have been assembled in the Data Preparation phase.
REFERENCE CLASS	The REFERENCE CLASS is a vector containing *class* information about the observations. Sometimes this is already available in the FINAL DATASET.
TRAIN DATASET	The PREPARED DATASET is split, assigning 70% of the observations to the TRAIN DATA SET.
TEST DATASET	As the PREPARED DATASET is split, 30% of the observations go to the TEST DATA SET.
OBSERVATION BOOTSTRAP	This parameter value indicates the percentage of observations should be used for bootstrapping. Default this is set at 80% for the OBSERVATION BOOTSTRAP.
FEATURE SAMPLING VALUE	This optional parameter sets the number of features to be sampled per iteration. This is, however, not implemented at this stage. FEATURE SAMPLING VALUE is a standard concept in the cluster ensemble setting.
***MIN* AND *MAX* CLUSTERS**	The *MIN* AND *MAX* CLUSTERS specifies the total number of k to be used for the cluster ensemble.
ALGORITHMS	The ALGORITHMS used on the TRAIN DATASET to search for clusters.
ITERATIONS	A value parameter specifying the total number of ITERATIONS for each algorithm, value of k and proximity metric.
(DIS)SIMILARITY METRIC(S)	A parameter specifying which (DIS)SIMILARITY METRIC(S) to be used on the TRAIN DATASET, or TEST DATASET.
CLUSTER PORTFOLIO	The generation stage ends with creating a CLUSTER PORTFOLIO that holds all clusterings.
CONSENSUS FUNCTION	A CONSENSUS FUNCTION is used to create the final optimized partition, $P^* \in \mathbb{P}$.

Continued on next page

Table 4.6 – continued from previous page

Concept	Description
REWEIGH CLUSTERINGS	With REWEIGH CLUSTERINGS many weak partitions are filtered out the portfolio, $\mathbb{P}$. This optimizes the CONSENSUS SOLUTION.
HEATMAP & CDF	From the CONSENSUS FUNCTION the HEATMAP & CDF is build. For every algorithm chosen and value of k a heatmap plot is build. Both the distribution functions, and heatmap give an indication which k performs best.
CLUSTER VECTOR	When the final portfolio, P^*, is created, the researcher can extract the cluster assignments. This is stored in a separate vector.
INDEX CRITERIA	The index criteria consists of INTERNAL VALIDITY INDEXES which are always present. In addition, the EXTERNAL VALIDITY INDEXES are only present when a REFERENCE CLASS is available.
LOW-DIMENSIONAL SPACE DATASET	After the consensus function stage, a LOW-DIMENSIONAL SPACE DATASET is created from the PREPARED DATASET.
CLUSTER VISUALIZATION(S)	From the LOW-DIMENSIONAL SPACE DATASET CLUSTER VISUALIZATION(S) are applied to visualize the end result.
UNIQUE CLUSTER CHARACTERISTICS VISUALIZED	From the CLUSTER VISUALIZATION(S) it is often not clear what each cluster represents. Thus, UNIQUE CLUSTER CHARACTERISTICS VISUALIZED aims to reveal, for example, which features or observations, are making up the clusters.
EXPERT INPUT	The final concept is the evaluation process that is set to collect EXPERT INPUT to involve domain knowledge into the clusterings.

4.2 Data-driven diagnosis method fragments

In the previous chapter we mentioned that feature engineering is a viable process within the machine learning domain. Depending on the dataset at hand, a researcher often needs to determine which features require additional engineering. Hence, the *engineer new features* activity in the Data Preparation method fragment. In this section, two examples of *engineer new features* method fragments are illustrated that encapsulate the feature engineering process performed at the Psychiatry Department of the UMCU. Thus, these two method fragments are highly situational, and therefore, only applicable for the dataset from the Psychiatry Department of the UMCU.

In order to perform the cluster ensemble on the psychiatry data, new features were engineered after deleting the non-informative ones. For instance, mental disorders are recorded over multiple features. The goal is to create one informative feature that states the detailed mental disorder. Moreover, the HoNOS is assessed multiple times per patient. Since we are interested in each unique hospitalization (i.e. a patient can be re-hospitalized), we want each HoNOS observation to correspond to the correct patient. For that we have used their first and last HoNOS scores. This requires, however, an additional feature that checks whether the dates between the hospitalization of a patient, and their HoNOS assessment date correspond. As such, each date of an observation in the HoNOS should fall between a patients' registration and discharge date. Finally, there is also a merging step in which the two datasets, the DSM-IV and HoNOS, are merged together.

In the *Improving DSM disorder codes* we first trimmed the DSM dataset, and made sure that the detailed mental disorders are in one feature. Moreover, in the *Merging HoNOS and DSM data* we first trimmed the HoNOS dataset to remove observations with missing values. Furthermore, a feature is created that determines whether the HoNOS scores belong to the right patient from the DSM dataset. If a patient did not correspond to the right HoNOS observation, than this patient was deleted from the dataset. Both method fragments are illustrated in Figures 4.4-4.5 (i.e. DSM dataset and HoNOS dataset), and are both accompanied with a concept table and activity table.

4.2.1 Improving DSM disorder codes

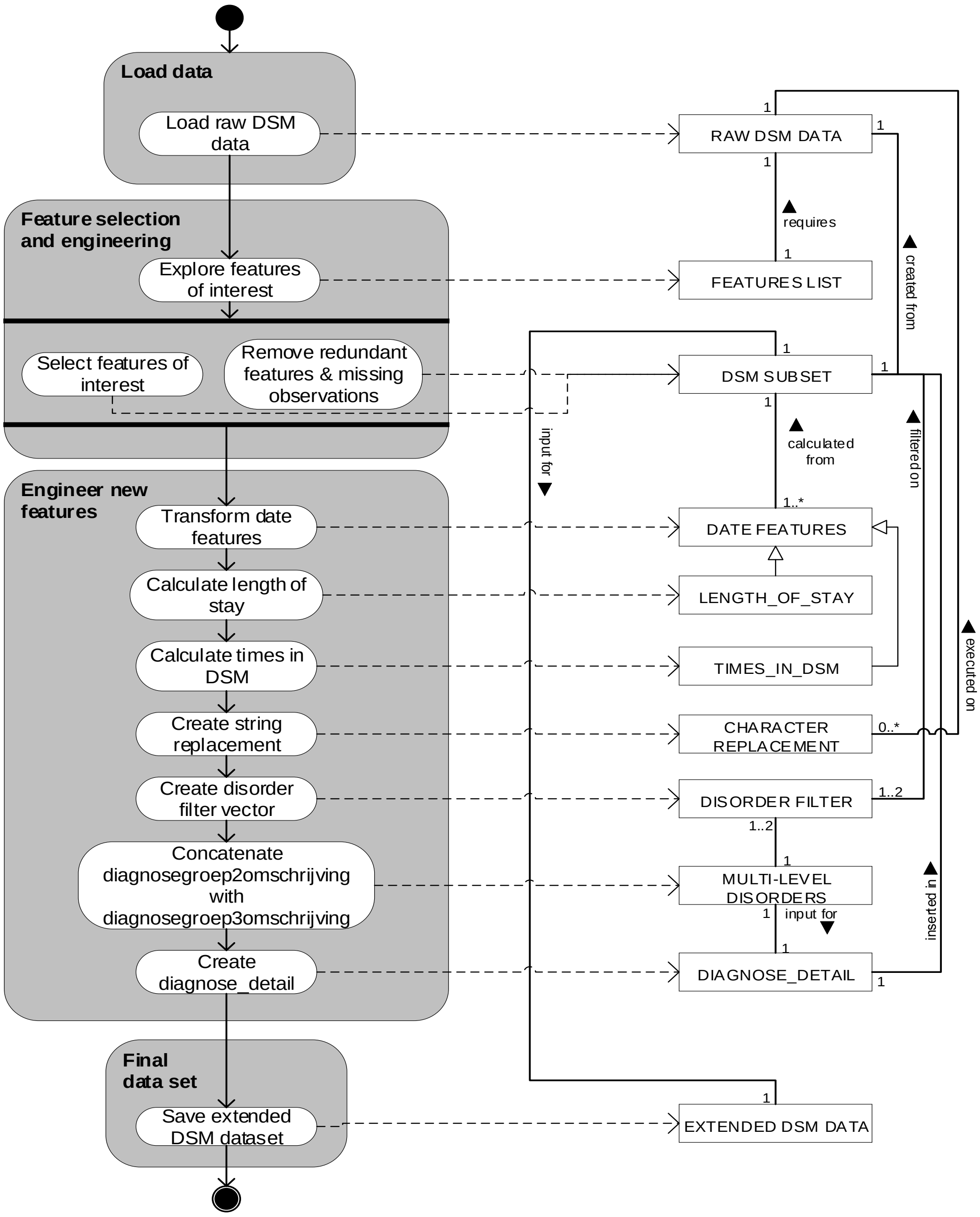


FIGURE 4.4: Engineer new features MAM fragment for Improving DSM disorder codes.

TABLE 4.7: Improving DSM disorder codes activity table.

Main activity	Sub-activity	Description
Load dataset	Load raw DSM data	The original DSM data is loaded into the working environment, creating the `RAW DSM DATA` to be engineered.
Feature selection and engineering	Explore features of interest	The researcher determines which features are informative, which contain missing values and which are redundant. This information is stored in the `FEATURES LIST`, and acts as a guide to determine whether the `RAW DSM DATA` requires some engineering or transformation.
	Select features of interest	Features of interest are selected and will return in the `DSM SUBSET`.
	Remove redundant features & missing observations	Features which satisfy the condition of $\geq$80% missing observations are removed, next to the informative ones. Subsequently, observations that do not have a *closed DBC, DSM-IV, diagnosiscode* or *hospitalization days* $\leq$*0* are removed.
Engineer new features	Transform date features	Each date feature that has the property of "character" is changed to the property of "date" in order to perform computations with it.
	Calculate length of stay	For each observation their total length of stay is computed, which is based on the date that they are hospitalized until discharge.
	Calculate times in DSM	For unique and duplicates of observations their number of hospitalizations is computed. The unique patients are patients who have been registered only once, or with a different disorder compared to the previous registration.
	Create replacement string	The `CHARACTER STRING` is used to replace conversion errors within the `RAW DSM DATA`. These conversion errors are text markup errors (e.g. Parano*Â*??de type → Paranoide type.)
	Create disorder filter vector	The `DISORDER FILTER` is used several times to search through features for the corresponding mental disorders. This is used to capture and concatenate details about the mental disorders.
	Concatenate diagnosegroep2omschrijving with diagnosegroep3omschrijving	With the `DISORDER FILTER` the corresponding detailed mental disorders are found which have their data spread over more than one feature. This is eventually concatenated to one feature.
	Create diagnose_detail	`DIAGNOSE_DETAIL` is the final feature that is created from multiple features that best describe the detailed mental disorder. This activity is in fact a merge between two vectors that both hold details of various mental disorders, resulting in a complete detailed list.

Continued on next page

Table 4.7 – continued from previous page

Main activity	Sub-activity	Description
Final dataset	Save extended DSM dataset	The extended DSM dataset is saved and is used as an input for the upcoming meta-algorithmic method fragment.

TABLE 4.8: Improving DSM disorder codes concept table.

Concept	Description
RAW DSM DATA	The `RAW DSM DATA` is the original dataset containing all features and observations.
FEATURES LIST	The `FEATURES LIST` is used to determine which features are informative, which are redundant, and which require some engineering.
DSM SUBSET	The `DSM SUBSET` is a dataset that contains all informative features, and observations needed for the Data Modeling phase.
DATE FEATURES	The `DATE FEATURES` changes the corresponding date variables from character to date, in order to perform calculations.
LENGTH_OF_STAY	The `LENGTH_OF_STAY` is calculated from `DATE FEATURES`, and computes the total amount of days that a patient with a specific disorder is in the `DSM SUBSET`.
TIMES_IN_DSM	As in the previous concept, the `TIMES_IN_DSM` calculates the total amount of times that a patient with the same disorder is re-hospitalized.
CHARACTER REPLACEMENT	A string vector that transforms text in each feature with the character property to the right text. It fixes conversion errors.
DISORDER FILTER	A string vector that contains multiple disorder names which is used to search through the `DSM SUBSET` in order to find the corresponding disorders.
MULTI-LEVEL DISORDERS	A new feature that is a concatenation of two original features in order to express mental disorders more detailed.
DIAGNOSE_DETAIL	The final feature that expresses the details of mental disorders in one feature.
PREPARED DSM DATA	The end result is the `PREPARED DSM DATA` which only contains the necessary features and observations.

4.2.2 Merging HoNOS and DSM data

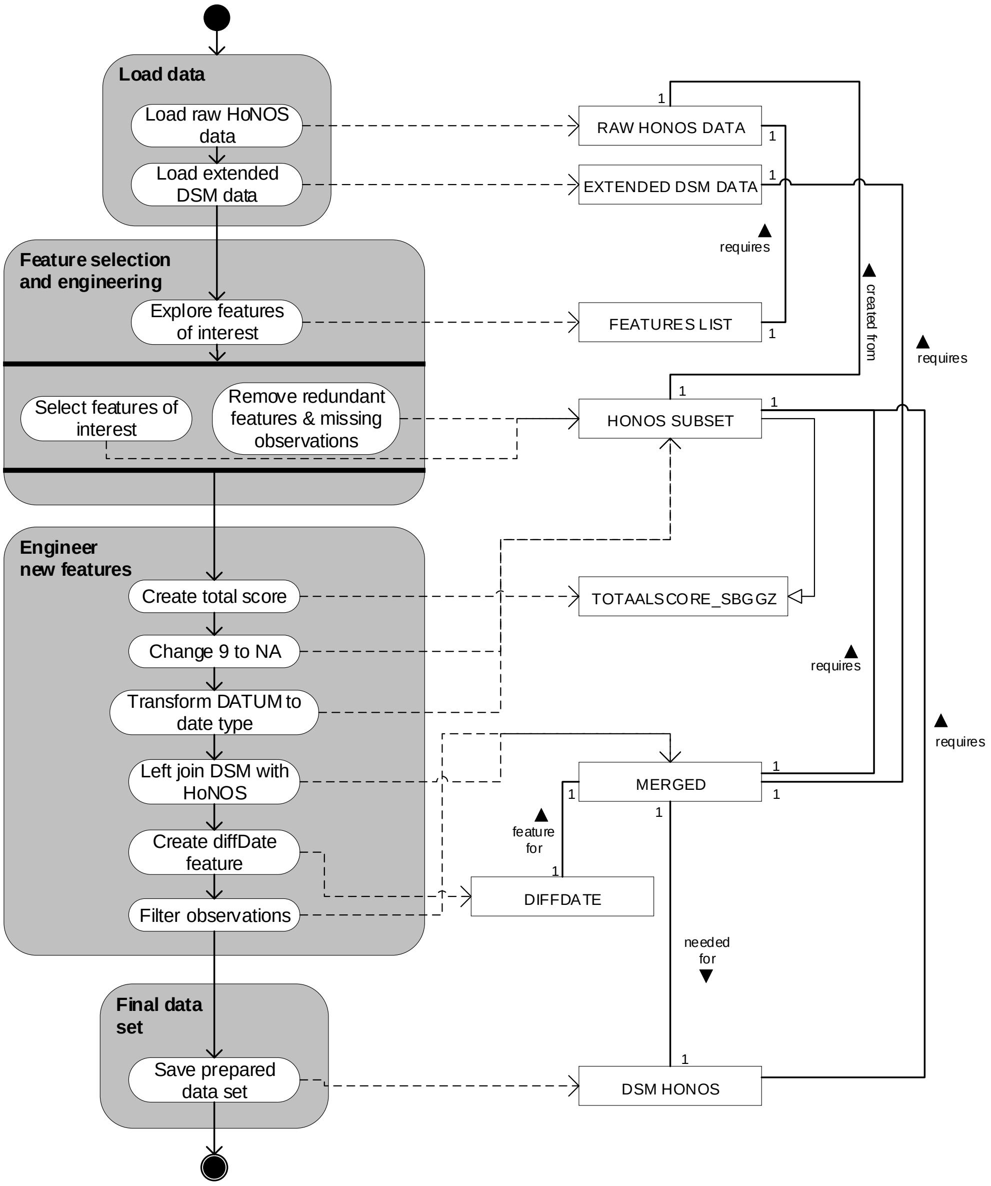


FIGURE 4.5: Engineer new features MAM fragment for Merging HoNOS and DSM data.

TABLE 4.9: Merging HoNOS and DSM data activity table.

Main activity	Sub-activity	Description
Load data	Load raw HoNOS data	The original HoNOS dataset is loaded into the working environment, creating the RAW HONOS DATA to be explored.
	Load extended DSM data	The EXTENDED DSM DATA is loaded into the working environment, which will be used later when it is time to merge the two datasets.
Feature selection and engineering	Explore features of interest	The researcher determines which features are informative, which contain missing values, and which are redundant. This information is stored in the FEATURES LIST, and acts as a guide to determine whether the RAW HONOS DATA requires some engineering or transformation.
	Select features of interest	Features of interest are selected, and will return in the HONOS SUBSET.
	Remove redundant features and missing observations	Features that store textual information next to the numerical information are removed, since we cannot use features of type "character". Next, features that are completely empty are removed. For observations, the ones that have NA-values in all their HoNOS questions are removed. The same goes for observations that do not have a plausible HoNOS date.
Engineer new features	Create total score	The total score of the HoNOS has a conversion error. Therefore, the total score of each observation is re-calculated.
	Change 9 to NA	Some HoNOS questions have the number 9 as their score. This number indicates that the patient did not provide an answer. This number is changed to NA, since it is not a valid number on the HoNOS Likert-scale.
	Transform DATUM to type date	Features that record a date are transformed from type "character" to "date", in order to do computations with it.
	Left join DSM with HoNOS	The DSM dataset is merged with the HoNOS dataset with a left join. This places all HoNOS features and observations to the right-side of the DSM dataset. The primary key used for merging is the pseudoID of the patients.
	Create diffDate feature	A feature is created that checks whether the date of the HoNOS falls between the date of the patients in the DSM. The output is a TRUE if it falls between the DSM date, else a FALSE.
	Filter observations	Every FALSE from the previous activity, create diffDate feature, is removed. The result is that each patient from the DSM now has their corresponding HoNOS scores at the time of hospitalization.
Final dataset	Save prepared dataset	The dataset, DSM HONOS, is saved. The dataset is now ready for the modeling phase.

TABLE 4.10: Merging HoNOS and DSM data concept table.

Concept	Description
RAW HONOS DATA	The `RAW HONOS DATA` is the original dataset containing all features and observations.
EXTENDED DSM DATA	The `EXTENDED DSM DATA` is the prepared dataset resulting from `DSM SUBSET`.
FEATURES LIST	The `FEATURES LIST` is used to determine which features are informative, which are redundant, and which require some engineering.
HONOS SUBSET	The `HONOS SUBSET` is a dataset that contains all informative features and observations needed for the Data Modeling and Evaluation phase.
TOTAALSCORE_ SBGGZ	The `TOTAALSCORE_SBGGZ` is the total score over all HoNOS questions per observation.
MERGED	The `MERGED` is a dataset that is a left join between the `HONOS SUBSET` and `EXTENDED DSM DATA`.
DIFFDATE	The `DIFFDATE` is a feature that checks whether dates between `EXTENDED DSM DATA` and `HONOS SUBSET` correspond to TRUE.
DSM HONOS	The final result after `DIFFDATE` is the `DSM HONOS` which is the dataset used for the Data Modeling and Evaluation phase.

Chapter 5

Cluster ensemble experimental evaluation

As stated in Chapter 1 the objective was to create meta-algorithmic model fragments that guides researchers to perform clustering using the cluster ensemble. In this chapter, both the model fragments and the cluster ensemble are evaluated on their effectiveness, expressed in terms of accuracy. By experimenting on several synthetic and real datasets, accuracy of a single clustering algorithm is compared with the cluster ensemble that follows the Data Modeling & Evaluation fragment presented in Chapter 4. Accordingly, this chapter provides an answer to the second research question:

2. *How accurate is the cluster ensemble compared to a standard clustering algorithm?*

In section 5.1 the experiment design is outlined. In section 5.2, details are given of the datasets that are used for the experiment. Section 5.3 concerns the set-up of the algorithms. Last, in section 5.4 results are reported.

5.1 Experiment design

Technical Action Research (TAR) is a way to validate the proposed treatment for the problem in context within an experimental setting (Wieringa, 2014). As such, the meta-algorithmic model created can be validated before the actual implementation takes place. The goal, therefore, is to evaluate the created model and, if necessary, adapt it for the actual implementation.

Evaluation of the meta-algorithmic model is conditioned within the following experiment: a single standard clustering algorithm (K-means) and the cluster ensemble are both evaluated on their accuracy. The cluster ensemble is built following the Data Modeling and Evaluation fragment (see section 4.1.3). The K-means algorithm is utilized with multiple restarts, centers and seeds. Evaluation of both is quantified using the accuracy metric from the confusion matrix. As such, the aim is to verify which algorithm is able to cluster observations better according to their original class labels. The final result provides an answer whether the cluster ensemble and model fragment both can be used as the proposed treatment to explore the data from the Psychiatry Department of the UMCU.

5.2 Dataset details

For our experiment we ran both algorithms on several real and synthetic datasets. In Table 5.1 an overview is given of each dataset by showing their original size according to their number of observations, features and ground truth clusters. Moreover, in the sub-sections that follow a short description for each dataset is given. The first four datasets are real datasets, and the last two are synthetic.

TABLE 5.1: Components of the datasets for the experiment.

Data set	Features	Obs	Ground truth k	Label (YES/NO)
WDBC	32	569	2	YES
Iris	5	150	3	YES
Seeds	8	210	3	YES
Abalone	9	4177	3	YES
Toy	3	373	2	YES
Aggregation	3	788	7	YES

5.2.1 Wisconsin Diagnostic Breast Cancer dataset

The WDBC, introduced in Chapter 3, is a two class dataset containing 32 features and 569 observations. Each feature describes the core property of a cell in a womans breast. Classes within this dataset are somewhat unbalanced, 357 observations are classified as 'benign' against 212 classifications of 'malignant'.

5.2.2 Iris dataset

The Iris dataset (Fisher and Harris, 1973) is a three class dataset containing five features, and 150 observations. Four out of five features contain values describing the classes; Setosa, Virginica, and Versicolor. The last feature contains the class labels. Each class is well-balanced over 50 observations each.

5.2.3 Seeds dataset

The Seeds dataset (Charytanowicz et al., 2010) is a three class dataset containing eight features, describing internal kernel structure of different varieties of wheat. Each class is evenly distributed over 70 observations.

5.2.4 Abalone dataset

The Abalone dataset (Nash et al., 1994) is a collection of physical measurements from abalone shelves. Although it does not stem from a machine learning study it is often used to test classifiers, making it also suitable for clustering. The dataset contains nine features and 4,177 observations, divided over female, infant and male shelves.

5.2.5 Toy dataset

The toy dataset is a synthetic dataset introduced by Jain and Law (2005). The dataset is a showcase to reveal problems with multiple clustering algorithms, because it has non-convex clusters. The dataset contains three features, and 373 observations. The classes are unbalanced, since the first class holds 276 observations and the second class only 97 observations.

5.2.6 Aggregation dataset

The last dataset is the aggregation dataset introduced by Gionis, Mannila and Tsaparas (2007). The dataset showcases the need of improving cluster robustness by combining multiple clustering outputs. It contains three features, and 788 observations. This synthetic dataset is also the only dataset that contains more than three classes within the experiment. Classes in this dataset are also unbalanced. The first class holds 45 observations, the second class 170, the third class 102, the fourth class 273, the fifth class 34, the sixth class 130, and the seventh class 34.

5.3 Experiment set-up

In order to provide a concise answer to our accuracy metric, and to experiment with our method fragment, we have chosen to narrow it down to the Data Modeling and Evaluation method fragment. Specifically, the activities; generation stage, consensus stage, and evaluate indexes are used. The visualization and evaluation part are omitted, since it would not serve as an elemental part in our quest to determine accuracy between a single algorithm and our cluster ensemble. For the other method fragments several activities in both Data Understanding and Data Preparation were introduced earlier in Chapter 3.

The experiment was done in twofold. First, each dataset was assessed with the standard K-means algorithm, since it is fast and well-known throughout literature. The *k* value initialization of the standard K-means was set according to the ground truth cluster numbers in Table 5.1 for each dataset. Moreover, the algorithm had multiple iterations and multiple random restarts to find the optimal within-cluster variance, SSE. Random seeds were set to their optimal parameter. The second part consisted of initializing the cluster ensemble according to the Data Modeling and Evaluation method fragment. Thereafter, utilization of the cluster ensemble on each dataset followed. For the generation stage multiple algorithms were used, such as partitioning algorithms (PAM, K-means), hierarchical algorithms (Agnes), model-based algorithms (spectral clustering) and grid-based algorithms (C-means). Moreover, bootstrap sampling was set at 80%, iterations of the algorithms was set at 300, and the number of clusters was set to the ground truth *k* per dataset. For the consensus stage Majority Voting was used. Afterward, poor performing cluster partitions were trimmed. Accuracy between the cluster ensemble and the single K-means was calculated using a confusion matrix. This was done for each dataset in this experiment. Last, the average accuracy for both algorithms was calculated by taking the accuracy outcomes separately of each algorithm from the confusion matrix.

Before clustering, uninformative features, such as patient ID's in the WDBC dataset and Rings in the Abalone dataset, were removed next to the features that represent the class labels. Several datasets, including WDBC, Abalone and Seeds were z-score scaled prior to clustering.

5.4 Experiment results

Figure 5.1 displayed below shows the accuracy for each dataset between the K-means and the cluster ensemble. The solid horizontal lines represent the overall average accuracy of both algorithms based on the six datasets examined.

The results show that there is an improvement in accuracy on each dataset for the cluster ensemble that is utilized according to the Data Modeling and Evaluation method fragment. The best result is achieved on the synthetic dataset Aggregation. Clusters in this dataset are hard to find for the K-means algorithm, mainly due to its arbitrarily shaped clusters. Moreover, some clusters were split by the K-means, causing them to be grouped with neighboring clusters. In contrast, the cluster ensemble did not perform split clusters, as such most clusters were identified accordingly. We observed the same routine for both algorithms in the Toy dataset. For the real datasets improvement in accuracy for the cluster ensemble is marginal, although it kept performing better than the single algorithm. For the Iris and Seeds datasets there is a small improvement in accuracy, 2% and 0.5% respectively. Improvements for the Abalone and WDBC datasets are somewhat better, 8.6% and 3.2% respectively. Poor performance for both algorithms on the Abalone dataset is not uncommon. The Abalone is known to be a hard clustering problem as observations are overlapping, for example, see Figure 5.2. However, since soft-clustering is used there is an improvement in accuracy, because observations can belong to more than one cluster.

To conclude, experimentation with six datasets, synthetic and real, showed an improved accuracy for the cluster ensemble that is modeled according to Data Modeling and Evaluation method fragment. For each dataset we have shown that using multiple algorithms, more iterations and removing poor cluster partitions bring effect for the cluster ensemble. As such, we determined that a single clustering algorithm has lower accuracy than our cluster ensemble. See Appendix C for the complete code that is used for this chapter.

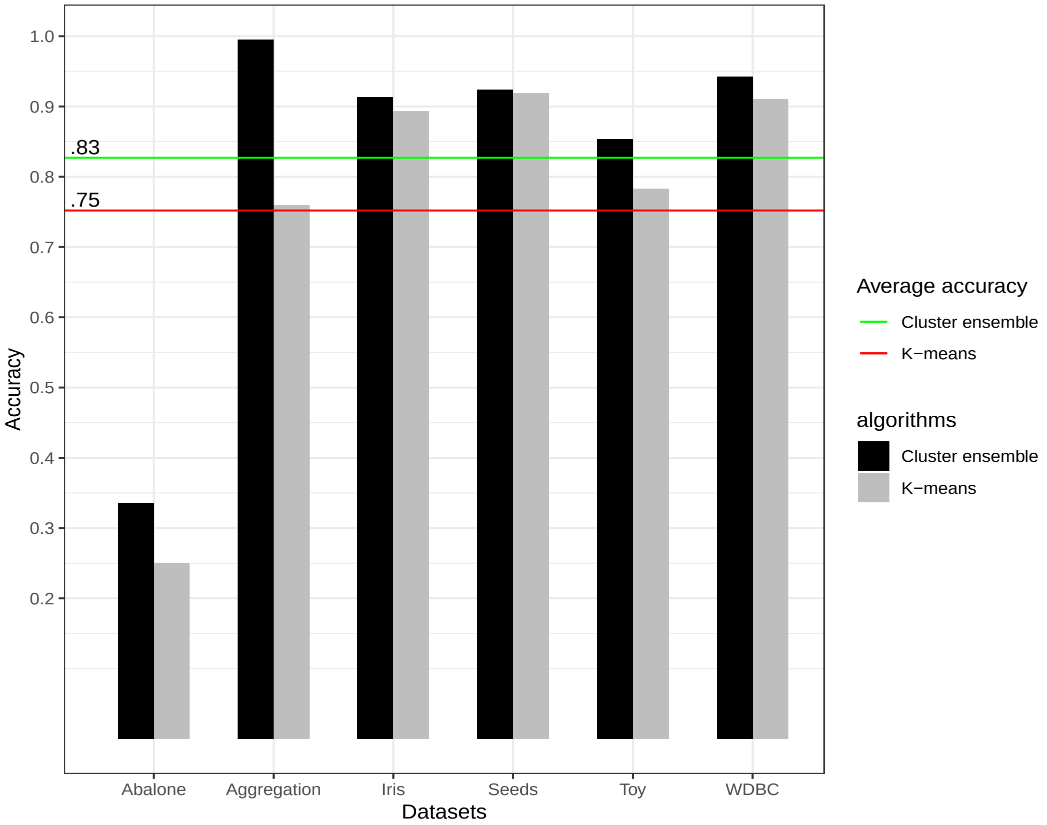


FIGURE 5.1: Accuracy between the cluster ensemble and K-means tested over six datasets.

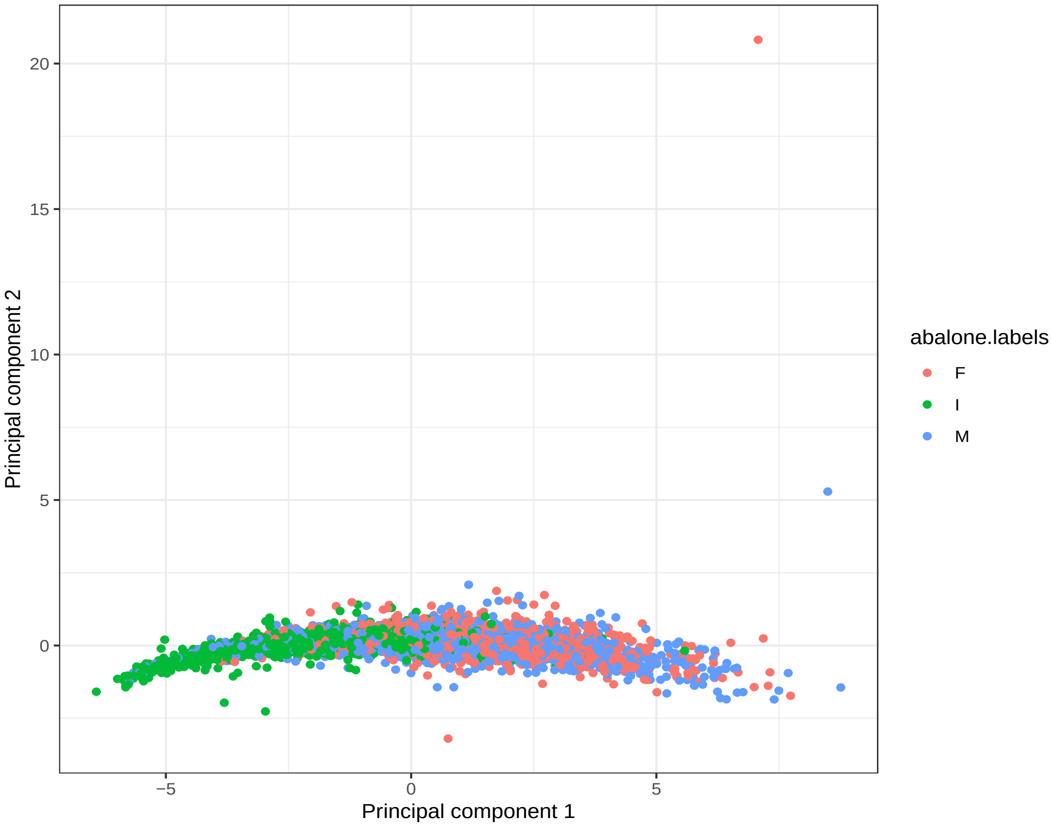


FIGURE 5.2: Two-dimensional projection of the Abalone dataset with PCA.

Chapter 6

Results from psychiatry data

As demonstrated in the previous chapter, the cluster ensemble is relatively robust in delivering accurate results for both synthetic and real datasets. In this chapter, we again use a cluster ensemble that is initiated with our Data Modeling and Evaluation method fragment, and also use the data understanding and data-preparation MAM fragments applied on the datasets from the Psychiatry Department of the UMCU. This approach is taken to provide insights into the steps performed. Particularly, this chapter provides an answer to the final research objective and its corresponding research questions:

- Utilize the cluster ensemble on the psychiatry data from the UMCU, consisting of DSM-IV diagnosed schizophrenic and psychosis patients, to find novel groupings that are based on features from the HoNOS

3. *Which number of clusters best describes the dataset according to internal index criteria, and which HoNOS features define each cluster?*
4. *What do experts say about the clusters when evaluating them?*

We begin this chapter by providing details of the dataset used for the cluster ensemble. Thus, section 6.1 provides insights into the data from a data-understanding and data-preparation perspective. In section 6.2, the algorithm details and how the results are evaluated are presented from the perspective of the data modeling and evaluation MAM fragment. Beginning from section 6.3, the first results are presented from the cluster ensemble, the internal index criteria is discussed, and the features of the clusters are visualized, answering SQ3. In section 6.4, the results are evaluated by domain experts, answering SQ4. Finally, in section 6.5 several other machine-learning techniques are discussed that have been investigated during this research. In Appendix D source code is given from this chapter.

6.1 Data insight

At the Psychiatry Department of the UMCU, multiple datasets are available containing specific patient information, such as the HoNOS and Diagnostic Treatment Combination —in Dutch, DBC— which we will refer to as the DSM. The DSM, for example, contains information about the mental disorder diagnosed for a patient, as well as the ICD codes (International Statistical Classification of Diseases and Related Health Problems) that belong to the diagnosed disorder and the DSM-related codes. In addition, time of hospitalization and discharge is recorded as well as the patient's unique ID. The HoNOS, on the other hand, is a list containing 12 questions related to the behavior of patients during hospitalization —for detailed information about HoNOS specifics, see Pirkis et al. (2005) — so psychiatrists can track treatment response. These 12 questions are divided into four subcategories, which we call the HoNOS *traits*, and refer to *behavior*, *impairment*, *symptoms*, and *social* conditions of a patient. In addition to the HoNOS, there is the HoNOSCA (Health of Nation Outcome Scale Childs and Adolescent), which is for juvenile patients. However, our study focuses on the DSM and the HoNOS to find clusters that separate conditions of mental disorders among adults to provide insights into which types of clusters exist.

As discussed in Chapter 4, both datasets are separated and they need to be merged for modeling. This merging required some feature engineering of the data. One of these major engineering tasks was to couple patients from the DSM to their corresponding HoNOS cases. The other major task was to simplify the DSM disorders into two features, such that it is apparent which type of mental disorder belongs to a patient. Merging both datasets resulted in the final dataset, which we refer to as the dsm_honos dataset. In Table 6.1-3 size in features and observations is provided from the original DSM and HoNOS dataset, as well as for the dsm_honos dataset.

TABLE 6.1: Original DSM

DSM	
	count
Features	131
Observations	25909

TABLE 6.2: Original HoNOS

HoNOS	
	count
Features	54
Observations	6234

TABLE 6.3: dsm_honos dataset

dsm_honos	
	count
Features	28
Observations	744

The size differences of observations and features between each dataset are notable. With the DSM being the largest dataset, it contained the most uninformative features for our modeling purpose. From the 131 features, we only required seven features of importance. These features were patient IDs, their mental disorder recorded over two features, hospitalization date and discharge date, and some descriptive features, such as times of hospitalization and total length in days of hospitalization. From the observations, an approximate 5,000 patients were removed. The patients removed did not have a clear diagnosis (i.e. their ICD and/or DSM codes were missing), no diagnosis was provided at all, or they were already diagnosed with the new DSM5, or were in the process of diagnosis, or had a hospitalization length of 0 days. The HoNOS dataset contained, in addition to the 12 informative features that record the HoNOS *traits*, many uninformative features also. Most of these consisted of text (i.e. psychiatrists' notes) and hence were not applicable for our modeling purpose. In total, we finished with 19 features for this dataset. Several features were of descriptive importance, such as age and sex, patient ID, three additional questions that are not standard HoNOS questions, and the date that the HoNOS was assessed. This final feature is of great importance, as it was used to couple the patients from the DSM to the HoNOS. The number of observations removed from the HoNOS was approximately 100. These were observations that did not have any HoNOS question answered, did not have a legitimate HoNOS registration, or did not have a legitimate HoNOS assessment date.

For the dsm_honos dataset, the DSM and HoNOS datasets were merged. Patients were coupled from the DSM to their HoNOS scores. Since our study focuses on patients suffering from schizophrenia or psychosis, the dsm_honos dataset was automatically reduced in patient size. Thus, patients who suffer from other mental disorders were removed. This resulted in a size of 1,227 observations, with patients only in the schizophrenia or psychosis spectrum. However, multiple observations had missing values. Imputing these missing values is a serious option, but harms to the outcome by presenting data as real, which in fact are not (Little and Rubin, 1989). On the other hand, removing too much data due to missing values creates bias, since the result is not representative. We, however, chose not to impute missing values and still retain more than 60% of the original size - 744 observations, which we expected to be sufficient for clustering. As such, patients suffering from schizophrenia and other psychotic disorders were only included with each of the 12 questions answered in the HoNOS. Finally, the categories among mental disorders for schizophrenia and psychosis were considered too large. Thus, smaller groups of mental disorders were placed under one common disorder, see Table 6.4 for an example. Additional descriptive statistics of the patients from the dsm_honos dataset are illustrated in Appendix E.

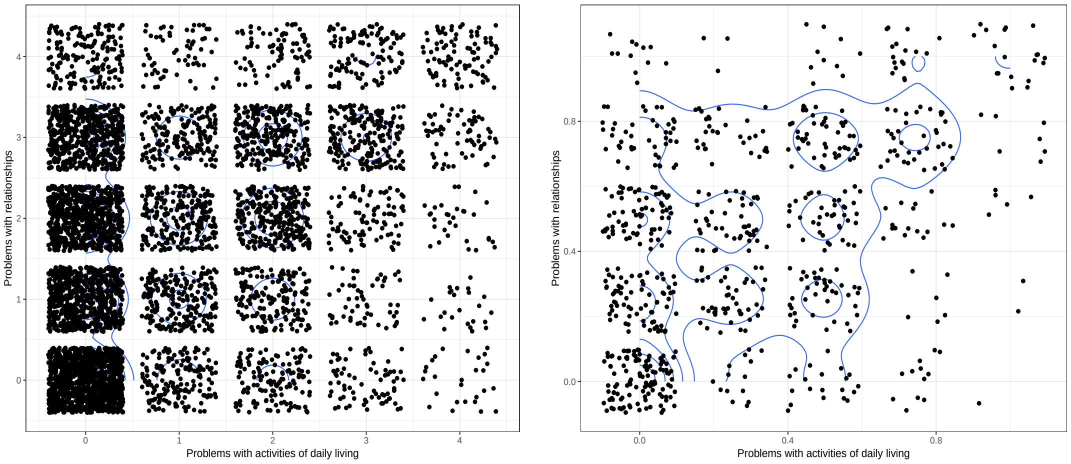
TABLE 6.4: Table showing sub-groups of disorders from original DSM and the resulting groups within the dsm_honos. Disorders in *italic font* represent the biggest groups within the dataset.

Original DSM disorders	dsm_honos **disorders**		
Original disorders with sub-groups	Disorders in main group	Total in dsm_honos	Prop. %
Brief psychotic disorder Brief psychotic disorder post partum Brief psychotic disorder with stress related factor(s) Brief psychotic disorder without stress related factor(s)	Brief psychotic disorder**	16	.0215
Psychotic disorder by substance Psychotic disorder by somatic disease	Psychotic disorder**	14	.0188
Psychotic disorder	*Psychotic disorder NOS**	296	.398
Schizoaffective disorder Schizoaffective disorder bipolar type Schizoaffective disorder depressed type	Schizoaffective disorder**	58	.0779
Disorganized schizophrenia	Disorganized schizophrenia	34	.0457
Catatonic schizophrenia	Catatonic schizophrenia	9	.0121
Undifferentiated schizophrenia	Undifferentiated schizophrenia	38	.0510
Paranoid schizophrenia	*Paranoid schizophrenia*	248	.333
Schizophrenia residual type	Schizophrenia residual type	8	.0107
Schizophreniform	Schizophreniform disorder	14	.0188
Delusional disorder Delusional disorder somatic type Delusional disorder persecutory type Delusional disorder erotomanic type	Delusional disorder**	9	.0121
Total		744	1

NOS* = Not otherwise specified.
** = Serves as common group for multiple sub-groups.
This to reduce the amount of sparse groups to preserve the overview.

6.1.1 Data preparation - rank normalization

The HoNOS is a Likert-scale questionnaire with a scale range from 1 to 4, with 1 representing "no problem" and 4 representing "severe to very severe problem." However, there is an unknown distance between the numbers (i.e. it is not known whether the distance from 2 to 3 or 3 to 4 is equal). Thus, it is impossible to determine the weights of each number. Moreover, Likert-scales are ordinal values instead of continuous numbers, requiring a correlation/similarity type of proximity metric only suitable for hierarchical clustering. However, as discussed in Chapter 2, similarity, and therefore ordinal values, can be transposed to dissimilarity, satisfying triangle inequality. Hence, we normalized the features with rank normalization, preserving the original distance and making it suitable for more clustering algorithms to work with. Figure 6.1, below, provides an idea of how the original HoNOS features look and the rank normalized ones in the final dataset.



(A) Original HoNOS

(B) rank normalized HoNOS in final dataset

FIGURE 6.1: KDE plot of difference between features "problems in relationships" and 'problems with activities in daily living' of the original HoNOS and rank normalized HoNOS in the final dataset. Option 'jitter' has been used to keep distances between observations in the same scale.

6.2 Data modeling and evaluation - algorithm and visualization details

For the data modeling and evaluation phase, the cluster ensemble was used to find clusters in the dsm_honos dataset. The full dataset was used, clustering all 744 observations. We were interested in discovering clusters and not in accuracy per se. Therefore, no test set was used. Sampling of the observations was set at 80%, and the number of k was set at three to six clusters. The algorithms used to build the ensemble were two hard-partitioning methods and one fuzzy soft clustering: PAM, K-means, and C-means respectively. Each algorithm had a repetition number of 500 iterations. For the proximity metric, the Euclidean distance was set. The consensus function set was Majority Voting, mainly because of its computational time compared with LCE. Poor performing algorithms were trimmed and reweighed.

First, the outcome from the cluster ensemble was visualized using a clustering heatmap, accompanied with consensus distribution functions (CDF) plots. From these figures, we can obtain an insight into which number k is optimal. Additionally, the internal validity indexes support the decision to choose the optimal k. Second, to visualize and evaluate the outcome, feature extraction was performed on the dsm_honos dataset. The MDS technique provided a sound visualization of the data in a two -dimensional space, especially because it allows the Euclidean distance to set proximity between observations. Cluster assignments from the ensemble were extracted and added to the MDS visualization to display clusters. Then, the identification of features was performed via visualizing a radar plot and multiple bar graphs. This method helps to understand cluster formation, as proposed in Henry, Tolan and Gorman-Smith (2005). To do so, the original dsm_honos dataset was used, as we needed the original, non-rank normalized features, since the mean and median scores per cluster were calculated. The radar plot uses the mean of each feature per cluster. The bar graphs use the median of each feature per cluster. Without doubt, both these scores provide, by their nature, different results, but using the median offers a clear identification of which mental disorders in each cluster contribute to the cluster formation. This aspect is further discussed, below, in the results section. Following the visualization of the clusters and features, the experts were interviewed to evaluate the results. Appendix D contains the R code for this study.

6.3 Results

The original portfolio $\mathbb{P}$ consisted of many different clusterings, since the ensemble clustered through $k = 3,4,5,6$ on a 500-iteration sample with three different algorithms. The consensus portfolio $P^* \in \mathbb{P}$ resulted in C-means with $k = 3$ as the final result (see Figure 6.2).

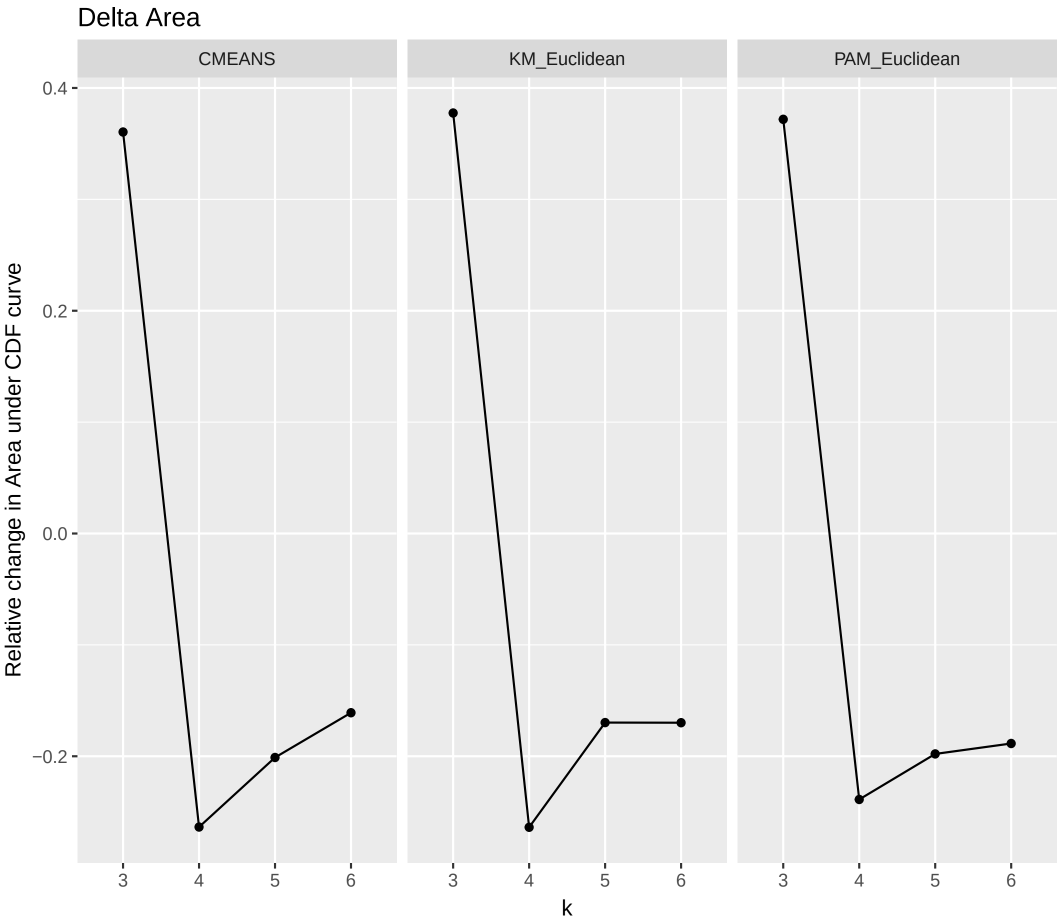


FIGURE 6.2: Delta Area plot that displays decline in cluster optimization after $k = 3$.

The delta area illustrated above displays the different values of k for each algorithm plotted. This delta area measures the relative improvement with respect to the best result of k. By examining the plot, it is clear that after $k = 3$ there is a rapid decrease in improvement of clusters. Also, there is relatively no improvement to be made by each algorithm, since they all display a similar plot. The decrease in improvement of $k > 3$ can be further examined in the CDF plot of the cluster ensemble illustrated in Figure 6.3.

The CDF plot displays how the consensus matrices obtained from the consensus portfolio approach the ideal k. Although we do not know at this time which k is optimal, we can derive from the CDF which k reflects stability in cluster forming. In the plot, each increase of k begins higher than the previous, reflected at the beginning of the axis (0.00). However, each higher value past $k = 3$ almost follows a linear increase as more observations are clustered (from left to right in the image). This pattern means that spurious clustering is happening, as there is variability in cluster assignments over observations. Monti et al. (2003) state that the predominance of zeros and ones affect the shape of the corresponding CDFs, meaning that high variability reflects a lack of stability in the distribution line. Stable clustering of observations represents a straight flat line across the $[0 - 1]$ range from the consensus matrices, and a final increase at the far-right side of the axis (1.00). Accordingly, to the plot, the C-means algorithm provides the most stable clustering for $k = 3$.

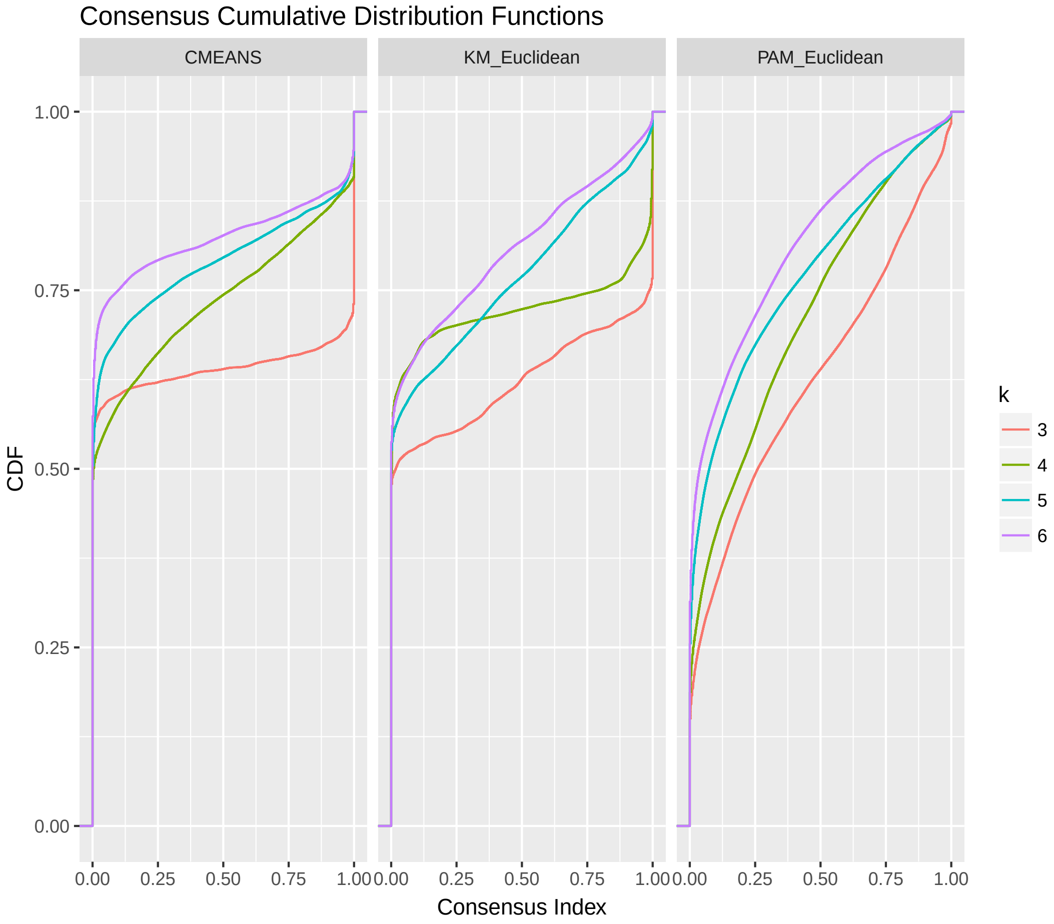
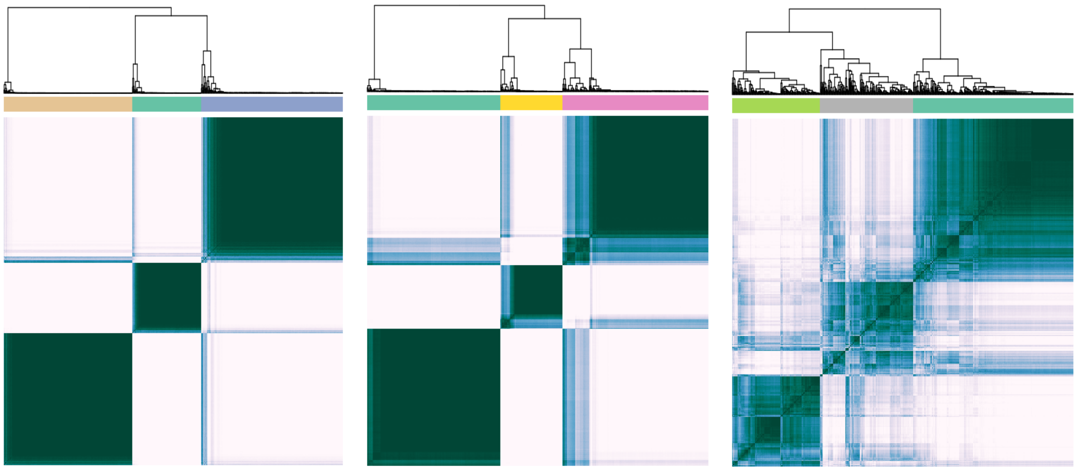


FIGURE 6.3: Consensus Cumulative Distribution Functions between multiple algorithms.

This spurious clustering result is clearly visible in the heatmap produced by the ensemble, illustrated below. Here, we display only the heatmap for $k = 3$; for other heatmaps that belong to $k = 4 \ldots 6$, see Appendix F. Visible in Figure 6.4 is the clustering result for each algorithm, with each block representing a cluster. Based on these three blocks, there are two larger outer blocks and one smaller block in the middle. The width of the blocks represents the size of the clusters. If we examine both Figure 6.4 and 6.5, there is a pattern visible, indicating that the C-means with $k = 3$ provides the optimal result for our dataset. As discussed in the previous paragraph, the distribution displaying a stable line provides a stable clustering. As both K-means and PAM display a linear line in their distribution, so does the heatmap display spurious clusterings. These spurious clustering are visible by the fragmented rectilinear lines that are formed. For example, the K-means algorithm displays multiple fragmented lines, such as in the top-right block, with a smaller one forming in it at the left-bottom of that same block. In the middle block, fragmentation is clearly happening by rectilinear lines appearing at both the top-left corner and the bottom-right corner, cascading toward the larger left and right blocks. For PAM, spurious clustering is definitely visible via the large amount of fragmentation happening in each block. Thus, this algorithm has a difficult time in placing observations in the similar clusters during each iteration.



(A) C-Means

(B) K-Means

(C) PAM

FIGURE 6.4: Cluster ensemble result of creating cluster structure between algorithms for $k = 3$.

6.3.1 Internal index criteria

Internal index criteria (discussed in detail in Chapter 3) measure inter-cluster and intra-cluster variances. Thus, the outcome from the criteria can be used to determine whether the cluster ensemble found a reasonable clustering for the dataset. "Reasonable" should be considered the most optimized clustering result. In fact, there is no clear definition of "good" or "wrong," since clustering lacks a clear definition in the machine-learning community. Therefore, from the ensemble, the most optimized k was chosen, based on the overall agreement among the indexes.

Figure 6.5 given below, illustrates three different internal index criteria, sorted for each value of k. In both the Calinski-Harabasz and Silhouette index, the C-means algorithm performed best. Both indexes report maximized values, which are an indicator for a better cluster result. Only for the Davies-Bouldin index the C-means algorithm perform better when more clusters were used. This result is indicated by the minimization of the index. However, this result is not a surprise, since this index evaluates intra-cluster similarities and inter-cluster differences. Thus, by adding more clusters, the Davies-Bouldin index criteria in general will improve. The Calinski-Harabasz index and the Silhouette index are both, in contrast to the Davies-Bouldin index, invariant to the number of clusters. Therefore, the verdict is that three clusters is the best result.

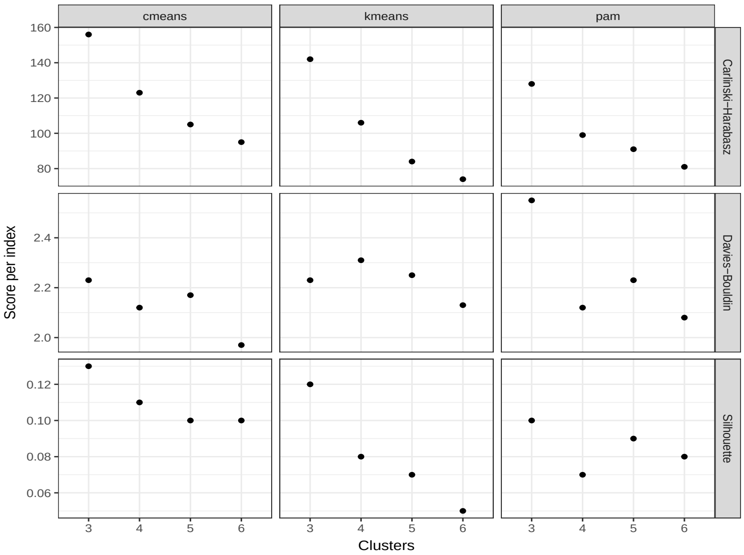


FIGURE 6.5: Carlinski-Harabasz, Davies-Bouldin, and Silhouette indexes for each k and algorithm of the cluster ensemble.

6.3.2 Clusters visualized

As discussed in section 6.2, visualization of the clusters was done by transforming the dbc.honos dataset into a two-dimensional feature space using MDS. Cluster assignments were added to the visualization to separate the clusters. Figure 6.6 displays the results from applying MDS with the cluster assignments added to the transformed matrix.

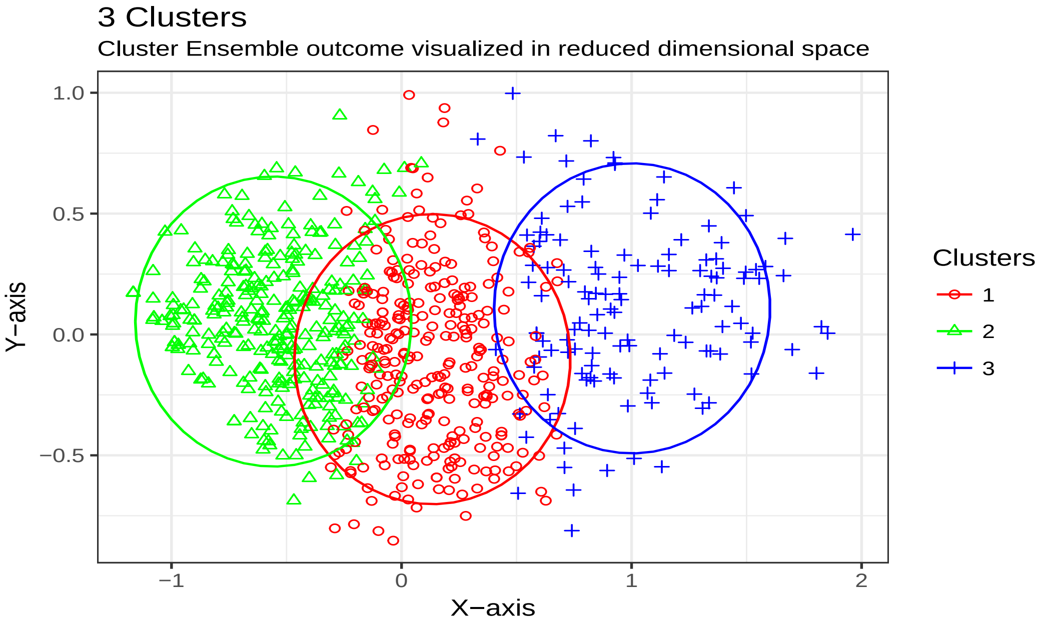


FIGURE 6.6: Cluster Ensemble visualized with MDS in two-dimensional space. Cluster 2, color green: 310 observations; Cluster 1, color red: 302 observations; Cluster 3, color : 134 observations.

Each cluster is defined by a different color and shape. For instance, the green cluster, Cluster 2, is the largest cluster, with 310 observations. The red cluster, Cluster 1, is the second largest cluster, with 302 observations; and the blue cluster, Cluster 3, is the smallest cluster, with 134 observations.

Since the C-means provided the best result in the cluster ensemble, it is expected that there is overlap between clusters, something that is noticeable in Figure 6.6. However, most observations are clearly separated from their neighboring clusters. According to our understanding, Cluster 2 is clearly separated from Cluster 3, and vice versa. In other words, there is not a single observation that belongs simultaneously to Clusters 2 and 3. Cluster 1, however, displays some overlap with the other two clusters.

6.3.3 Overall cluster characteristics

With the clusters visualized, it is important to define each of them; simply visualizing the clusters does not reveal their characteristics. To understand each cluster characteristic, a radar chart was employed, which reveals how each feature in the dbc.honos dataset contributes to the formation of the clusters – more specifically, how each feature characterizes the clusters. This approach helps in understanding cluster formation. In Figure 6.7, the radar chart is visualized. For each feature, we calculated the original cluster mean by using the non-rank normalized features. As such, the original Likert-scale value was used from the HoNOS.

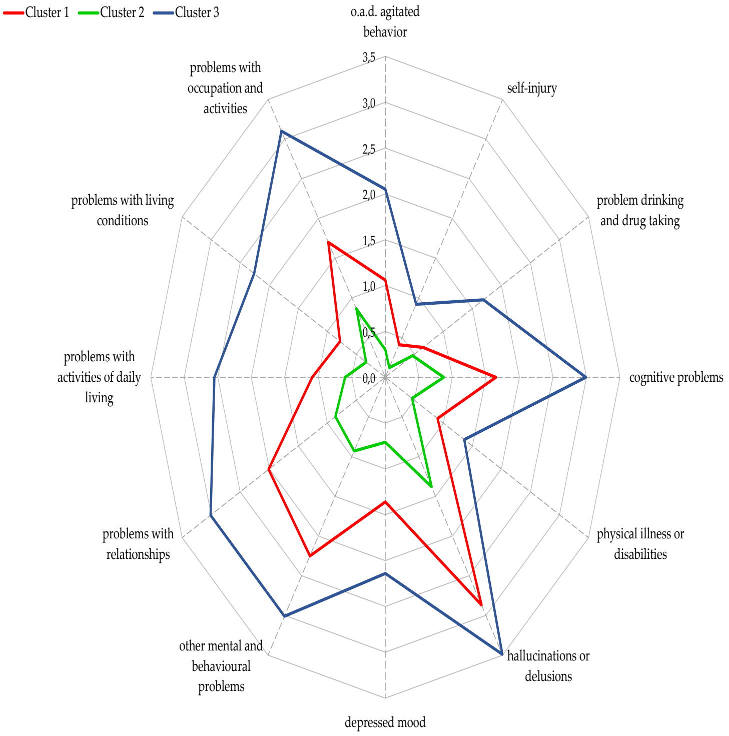


FIGURE 6.7: Radar chart of each HoNOS feature measured per cluster. Each feature is based on its mean value per cluster.

The radar chart illustrates an interesting behavior; each cluster has a somewhat similar shape. For instance, for features such as *self-injury*, *physical illness or disabilities*, and *depressed mood*, each exhibit a decline in their score for every cluster. Simultaneously, an increase in the score follows for the features immediately after the previous ones. Moreover, features such as *problems with occupation and activities*, *cognitive problems*, and *hallucinations or delusions* are defined by sharp increases in their score for each cluster. Thus, it seems that each cluster follows a similar pattern. What follows from this observation is whether features, such as *cognitive problems* and *hallucinations or delusions*, correlate with *problems with occupation and activities*. In other words, it is interesting to know whether some HoNOS features are indeed correlated with each other, especially those features that display a strong decline in their feature score, such as *self-injury* and *physical illness or disabilities*, each constituting a part of the traits *behavior* and *impairment*. Finally, in contrast to Cluster 1 and Cluster 2, note from the radar chart that the decline in features, beginning from *other mental and behavioral problems* to *problems with living conditions*, is more flattened for Cluster 3. This difference could indicate that patients in Cluster 3 are more or less defined by their *social* trait.

6.3.4 Cluster characteristics in-depth

As the radar chart illustrates a somewhat similar pattern between the clusters, it is difficult to identify the key differences. Therefore, the original Likert-scale feature median scores, separated into patients' disorders, were used to differentiate between the clusters. This approach allows us to determine which mental disorders from the dbc.honos dataset, as provided in Table 6.4, are represented by each cluster, and we can then observe which specific feature of the HoNOS distinguishes a cluster. As an end result, it is possible to determine which HoNOS *traits* define a cluster. This is done using a bar graph. What now follows is a selection of bar graphs that illustrate the key differences between each cluster.

Beginning with Cluster 3, termed the "severe cluster," the highest mean value scores per feature in the radar chart are displayed. Similarly, it is expected that each feature of this cluster has the highest median score when compared with Clusters 1 and 2. However, what is interesting is when each feature is characterized by their diagnostic labels from Table 6.4. By doing so, Cluster 3 is typically characterized by three key features. Figures 6.8-6.10 displays these three key features. Noticeable from these features is that in Figures 6.8 and 6.9 most observations are from Cluster 3. That is, Cluster 3 is mainly profiled by the overrepresentation of multiple disorder groups in the *problems with living conditions* and *problem drinking and drug taking* features. A simple comparison between each cluster and their feature mean scores reveals that observations from Cluster 3 score are, on average, ≥ 2 than Cluster 1, and > 3 to > 6 than Cluster 2. Moreover, from the radar chart, there is no sign of decline for Cluster 3 in the *problems with relationships*. Although every DSM-IV disorder from every cluster is represented in this feature, it is noticeable that Cluster 3 has multiple groups of observations that exhibit high scores. Hence, the term "severe cluster."

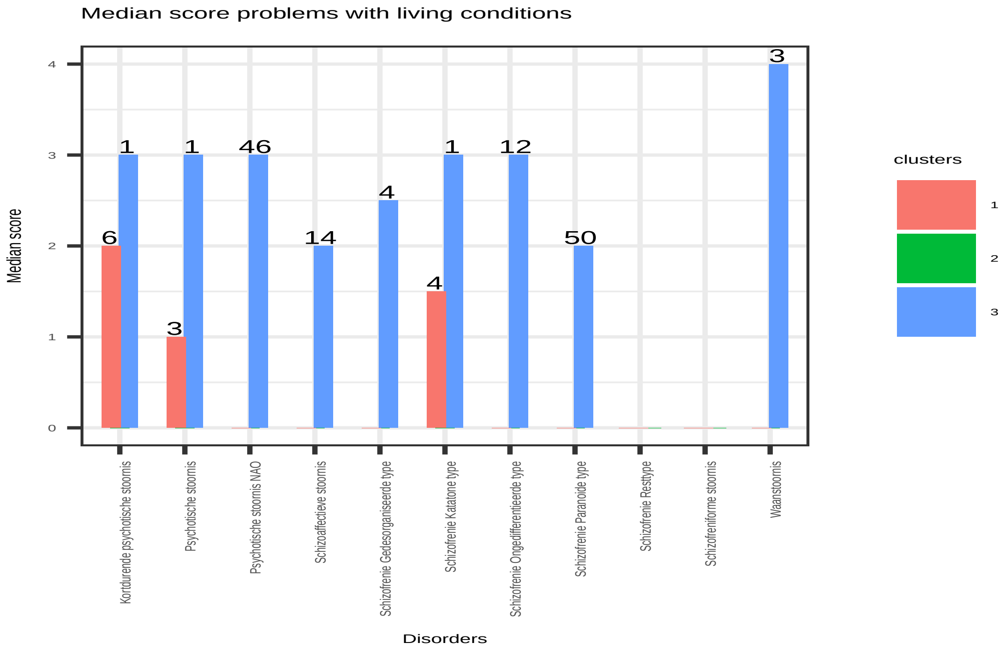


FIGURE 6.8: Bar graph of feature *Problems with living conditions.*

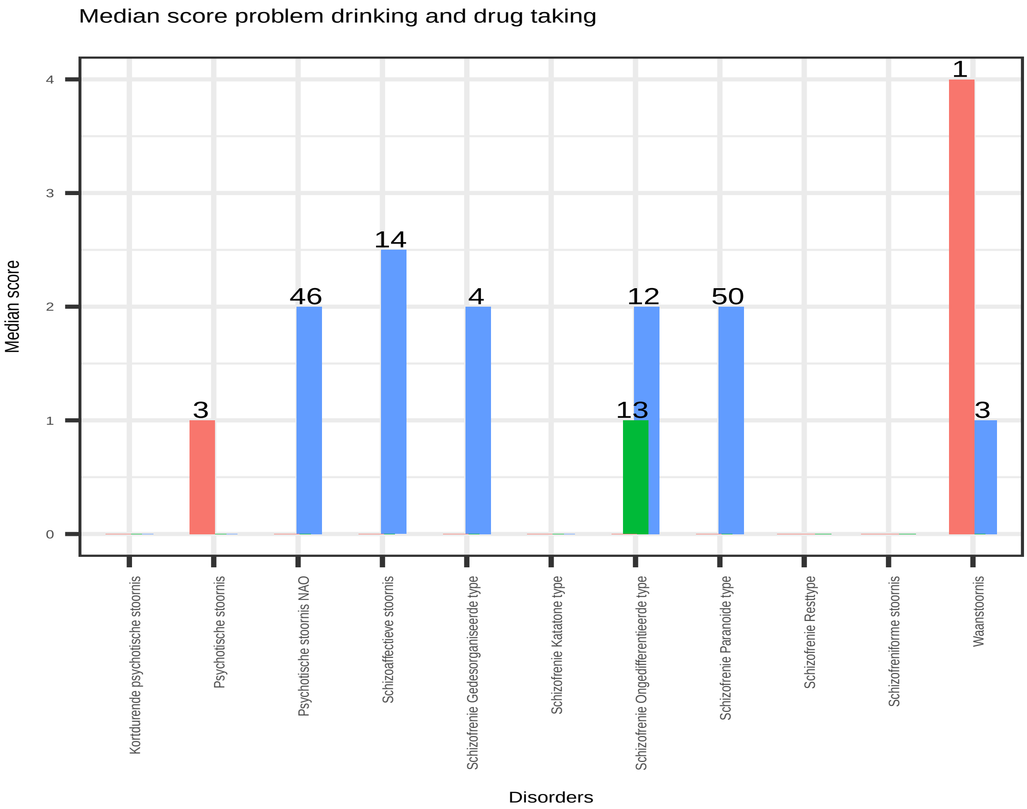


FIGURE 6.9: Bar graph of feature *Problem drinking and drug taking*.

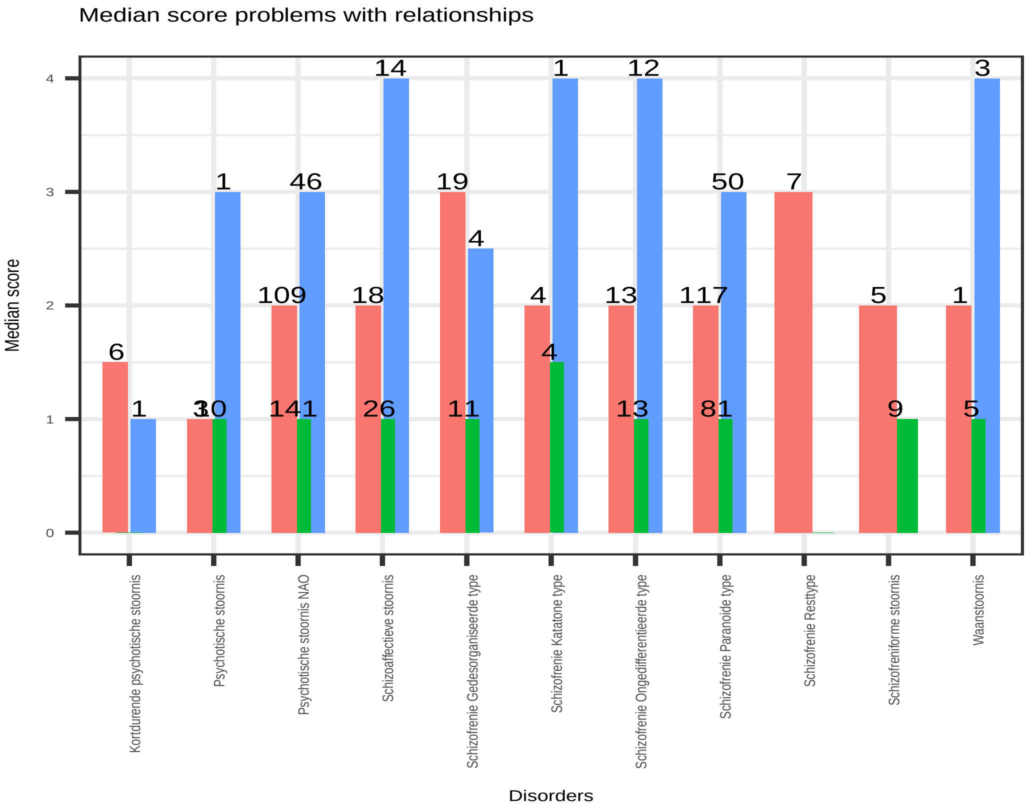


FIGURE 6.10: Bar graph of feature *Problems with relationships*.

Something that differentiates Cluster 1 from Cluster 2, termed the “mild cluster” and “low problematic cluster” respectively, is the feature *o.a.d. agitated behavior*. For example, by examining Figure 6.11, it is clear that, apart from Cluster 3, Cluster 2 is not present, something that we cannot determine from the radar chart. This, of course, is due to the scores by each patient in Cluster 2, as the total cluster mean is taken from this feature. The cluster median, however, has a more robust value and reveals that the majority of these patients do not display any signs of problems in the *o.a.d. agitated behavior* feature. Therefore, Cluster 1 is absent in this bar chart and is often absent also in other bar charts that were created for the other features. However, Cluster 1 still has a mean difference >3 in the *o.a.d. agitated behavior* feature when compared with Cluster 2.

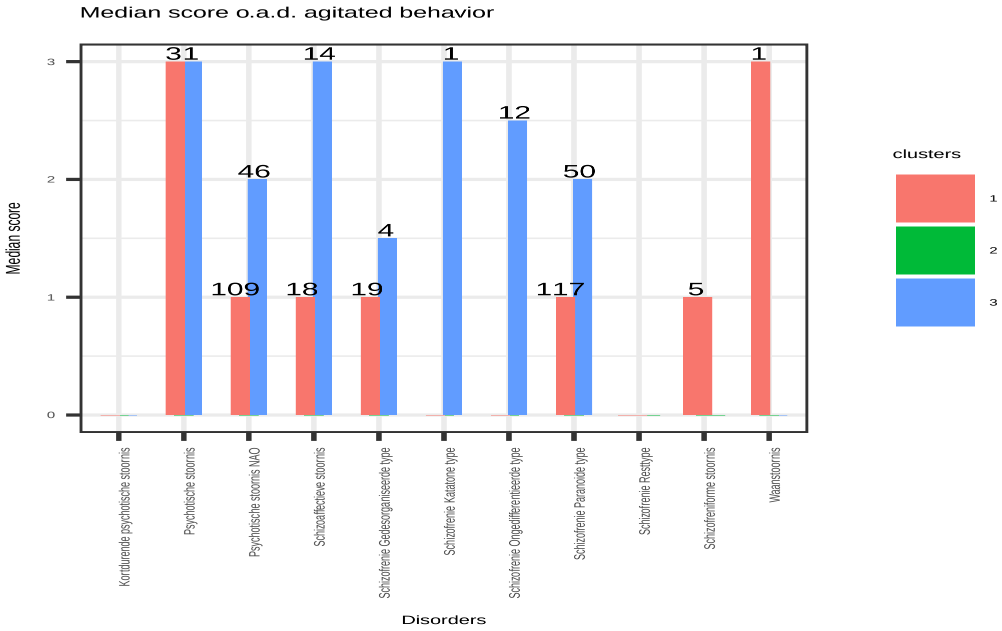


FIGURE 6.11: The feature *o.a.d. agitated behavior* that differentiates Cluster 1 from Cluster 2, based on the feature median score.

Overall, Cluster 2, the “low problematic cluster,” is mainly present in the following features: *cognitive problems*, *hallucinations or delusions*, *depressed mood*, *other mental and behavioral problems*, *problems with relationships*, and *problems with occupation and activities*. In these features, at least five out of nine DSM-IV diagnostic classes are represented, including the two largest patient groups: psychotic disorder NOS and paranoid schizophrenia. Based on this observation, patients in Cluster 2 are absent from the *behavior* trait.

In summary, the identification of three clusters led to differentiation between “low problematic” to “mild” and “severe,” in which the first cluster has the highest number of observations, 310, with 302 for the “mild” cluster, and down to 134 observations in the “severe” cluster. The key characteristics that define these three clusters are multiple features in which the “severe cluster” is clearly defined by two features that are part of the *social* trait. The “mild cluster” differentiates itself only from the “low problematic cluster” by one feature, which is part of the *behavior* trait. Furthermore, the "mild clusters" remains distinctive from the “severe cluster” by being partially outside of several features, constituting the *social*, *behavior* and *impairment* traits. The “low problematic cluster” is, in general, absent from the *behavior* and *impairment* trait and plays a lesser role in the *social* trait compared with the other two clusters.

6.4 Evaluation with experts

This section concludes the study by answering SQ4. Domain experts from the Psychiatry Department of the UMCU were consulted. These experts were asked to analyze the clusters and their features to provide a better understanding regarding how certain features form these clusters.

6.4.1 Expert evaluation

Three experts were involved in the process of providing domain-expert insight into the clusters. This insight was obtained via a semi-structured interview, in which a short presentation was presented to each expert individually about the results. Later, the experts were asked to provide their perceptions about the results, with the main focus being based on the results from Figures 6.9-6.11 and the pattern from the radar chart in Figure 6.8. Two experts had an average interview session of 30 minutes, while the other had an interview session of 10 minutes, with e-mail correspondence afterwards.

According to all three experts, the “severe cluster” does not display a remarkable nor contradictionary pattern based on its characteristic features. As one expert stated: *"The social problems (the social trait) score higher when the psychiatric symptoms are higher as well, and the psychiatric symptoms are also a cause of more social problems. Thus, my impression is that in both ways the features affect each other. Which in this case, and according to the outcome of the model, the psychiatric symptoms are not determined by a specific disorder, but rather a showcase of vulnerability for multiple disorders."* This statement is somewhat in line with another expert, who stated: *"The side effects in the social environment appear, since the patients already show an increase in their psychological problems at a, what seems to me, steady rate for each disorder in that specific cluster (Cluster 3)."* In addition, the same expert found it interesting to note that the brief psychotic disorder and catatonic schizophrenia both appeared in Clusters 1 and 3. Regarding that, the expert stated: *"To my understanding, brief psychotic disorder is rarely diagnosed in daily life. As for catatonic schizophrenia, this is more of a mixed-type disorder, hard to diagnose as well."* Then, the same expert added: *"This might be a bias within the model, or the disorder at that specific time frame indeed caused trouble for the patients."*

The final expert guided the focus more on the feature *problems with living conditions* and added to the previous statements that: *"It seems logical that positive symptoms affect the negative symptoms. However, due to the nature of the HoNOS, it might be the case that a psychiatrist deems a patient to have problematic living conditions, since the patient still lives in a neglected household."* What the expert meant explicitly is that a psychiatrist judges that a patient has problematic living conditions due to the severity of their diagnosed symptoms.

Examining the *problems with relationships* feature more closely, two experts could not provide an explicit answer, whereof one stated: *"It is hard to discriminate on that between each cluster. Just like the previous feature (problems with living conditions) this one is also affected by several psychological symptoms."* However, the other expert added: *"The heart of this symptom lies in misinterpreting someone's emotions. Thus, as we can see, each disorder group is present in this HoNOS question, and only differentiates itself by the severeness of the problem. Hence, the differences between the three clusters."*

The final symptom that separates the “severe cluster” from the others is with the HoNOS feature *problem drinking and drug taking*. One pattern observed from the radar chart is that *o.a.d. or agitated behavior* correlates with *problem drinking and drug taking*. Similarly, an expert stated, regarding this pattern: *"What we see is that this often goes hand in hand, and together forms a troublesome pair for the patient."* In addition, the same expert added: *"It is a stigma of psychiatrists to focus on this problem. And what I observe from the cluster is that a lot of patients indeed take our advice by heart."*

To conclude this section, it is interesting to discover what the experts think about the clusters in general; that is, finding out whether the “low problematic cluster,” “mild cluster,” and “severe cluster” do in fact differentiate from each other. What follows is a short summation of the experts' insights.

The first expert described the differences between the clusters and their underlying features as a normal phenomenon. Positive symptoms, such as hallucinations, are common in the mild, average, and severe states of a disorder. Even within the normal population of these disorders, the same trend occurs – not only within this dataset. Looking closely at the clusters, almost the same pattern occurs for the positive symptoms and negative symptoms. However, an interesting and deviating pattern is observed in self-injury. This pattern is only observed with patients who have severe problems. Thus, this is something that should only occur in the "severe cluster" and is, therefore, not represented in the normal population. The same pattern applies to the *problems with living conditions*. Only the severe patients deviate from the other patients, as it is not a normal phenomenon.

The second expert described the differences between the "low problematic cluster" and "mild cluster" to be somewhat equivalent. The expert continued: *"Having a low score on positive symptoms would be in line with a low score on negative symptoms. But, it may be that there is a coincident cluster generation between the both, making it hard to really differentiate between the 'low problematic cluster' and 'mild cluster.' Moreover, these sub-types of schizophrenia disorders are removed with the new DSM-5, basically because there is too much overlap. To explain, the catatonic sub-type is of mixing type, which is often misdiagnosed in practice, and therefore may cause bias. It would be interesting to see how the clusters are formed when the multiple sub-types are united to just one or two. Nonetheless, based on what is found, I would say that the severe patients are indeed well deviated from the other two clusters. Especially if we consider that there is just one small group that has severe problems in their surrounding environment."*

The final expert described the differences between the clusters as an interesting discovery. Initially, the expert thought that the "severe cluster" would be dominated by males, particularly of elderly age. However, this is not the case. Although males are indeed dominant in the dataset, the differences of age and sex between the clusters is negligible. Thus, the expert revised the answer regarding the differences between positive and negative symptoms to the same as the previous expert. In addition, the expert mentioned that, according to the DSM, there must be manifesting symptoms for the disorder to be diagnosed. Practicing psychiatrists, however, recognize predominantly the seriousness of the symptoms, rather than the conflicting limitations that come with these symptoms (e.g. the problems that occur in the *social* trait). However, the severity of the symptoms does not reveal the gravity of the limitations. Therefore, it is interesting to note what these clusters reveal in terms of groupings of patients and their overall symptoms.

In summary, based on the experts' evaluations, the "severe cluster" is indeed a deviating cluster from the other two clusters, especially since the *social* trait differentiates this cluster, followed by the *behavior* trait. Furthermore, the "severe cluster" is further defined by the feature *self-injury*, something that is only observed in severe cases among patients. Thus, it seems that there is some correlation between the features from the HoNOS in the "severe cluster." The difference between the "mild cluster" and "low problematic cluster" is less obvious because the features in general share almost the same outcome, and therefore it is more difficult to determine whether there is a correlation. However, there is an exception to be made for the *o.a.d. or agitated behavior* feature in the "mild cluster" to differentiate itself. Nonetheless, the overall correlation between *behavior*, *symptoms*, *impairment*, and *social* traits are less striking for the "mild cluster" and the "less problematic cluster."

6.5 Other machine learning approaches

This last section describes other machine-learning approaches that have been used to identify clusters, correlation between features, and features that characterize clusters within the dbc.honos dataset. As the original dataset is Likert scale based, it becomes more problematic to find clusters than, for example, ratio-scaled datasets. To identify if clusters are to be found within the dbc.honos dataset, we first evaluated the dbc.honos dataset with hierarchical clustering using various linkage-types, and by using multiple proximity metrics that are suited for these types of datasets (i.e. Spearman, Canberra, and Kendall). Moreover, multiple dissimilarity metrics were also used, such as squared Euclidean distance etc. With these proximity metrics we evaluated different linkage types. Afterwards, the cophenetic correlation coefficient was calculated, which determines if there is structure between the proximity matrix and the clustered matrix by the algorithm. The average cophenetic correlation coefficient was between 50% and 67%. The squared Euclidean distance and average linkage showed the highest correlation coefficient. As such, we decided to transform the features with rank-normalization and use a dissimilarity metric (i.e. Euclidean distance).

Second, Self-Organizing Maps (SOM) was evaluated on the dbc.honos dataset. The result showed sparse clusters with a few dense regions, indicating that this algorithm had a difficulty in grouping observations.

In addition, two cluster ensemble packages were evaluated simultaneously. Both packages indicated a k value of 3 or 4. By evaluating the results intensively the optimal $k = 3$, was chosen to start clustering with.

Finally, since it is interesting to know whether there is a correlation between each feature, a simple correlation matrix was devised. The result from the matrix showed no correlation between the features. Furthermore, *Random Forests* in an unsupervised setting was used to determine if some features contribute to the formation of clusters. This, however, turned out not to be true, since the error rate was too high from the Random Forests to give a decisive answer as an average above $\geq 50\%$ was reached each time the algorithm was run with different parameters.

Chapter 7

Conclusion

This study was conducted at the intersection of mental healthcare and information science, at the Psychiatry Department of the UMCU. The aim was to devise a meta-algorithmic model that guides non-machine-learning experts to adequately perform data mining tasks in the unsupervised domain of machine learning, and to find groups among diagnosed patients in the data from the psychiatry department by utilizing an unsupervised machine-learning algorithm (i.e. cluster ensemble). As such, the main research question of this study was as follows:

> *"Which steps comprehend the MAM for the cluster ensemble in the unsupervised domain, and to what extent can the cluster ensemble contribute to novel insights into mental disorders by utilizing and evaluating it, to improve diagnosis and treatment that is in line with precision psychiatry?"*

To structure this main research question, four sub-questions were formulated. Answers to these sub-questions were provided throughout this study and are summarized below. First, the answers to the sub-questions are reviewed, followed by an answer to the main research question.

7.1 Answers to the sub-questions

Sub-question 1, thoroughly discussed in Chapters 3 and 4, is answered based on literature from the machine-learning domain. Sub-question 2, discussed in Chapter 5, is based on experimental results with real and synthetic datasets. Sub-question 3, discussed in Chapter 6, is based on experimental results also, but now with data from the Psychiatry Department of the UMCU. Additionally, the same question is answered using theoretical foundations from the machine-learning domain. Finally, SQ4, discussed in Chapter 6 also, is answered based on results driven by domain experts from the psychiatry department.

7.1.1 Answer to Sub-question 1

What are the steps to be described and modeled by an MAM method to perform data understanding, preparation, modeling, and evaluation of the cluster ensemble in an unsupervised domain?
Several studies confirm that machine learning is often improperly executed outside the computer/information-science domain. Additionally, selecting an algorithm is considered a burdensome task. Therefore, the MAM was introduced by Spruit and Jagesar (2016). However, their model is only suited for the supervised domain. Therefore, **new steps were described** and a **new generic MAM was modeled** that guides non-machine-learning experts in the field of unsupervised learning. This model supports the use of a novel technique, termed "cluster ensemble," a robust technique, when compared with traditional clustering tasks, which alleviates the problem of selecting a proper algorithm (Topchy, Jain, and Punch, 2003). See Appendix A for the resulting model.

7.1.2 Answer to Sub-question 2

How accurate is the cluster ensemble compared with a standard clustering algorithm?
To illustrate that the cluster ensemble is a robust algorithm, several real and synthetic datasets were assessed. Both the cluster ensemble and a single clustering algorithm were tested for their accuracy. Based on the outcomes, **we showed that the cluster ensemble has a better**

overall accuracy of 83%, against 75% from the single clustering algorithm. Moreover, in all datasets, the cluster ensemble performed better in terms of accuracy.

7.1.3 Answer to Sub-question 3

Which number of clusters best describe the data set according to internal index criteria, and which HoNOS features defines each cluster?
Data from the Psychiatry Department of the UMCU was used to identify groupings in patients diagnosed with schizophrenia or psychosis. To do this, the cluster ensemble was applied to the data. The resulting heatmaps and CDF curve reveal that $k = 3$ is the optimal cluster structure for the psychiatry data. Moreover, two out of three internal validity indexes, **the Silhouette and Calinksi-Harabasz** index, support this finding by awarding the highest index score to $k = 3$.

Furthermore, it is clear that the features **problem drinking and drug taking**, and **problems with living conditions** separate the "severe cluster" from the other two. The "mild cluster" only separates itself with the feature **O.A.D. agitated behavior** from the "low problematic cluster." There is **not a single feature** that separates the "low problematic cluster" from the rest.

7.1.4 Answer to Sub-question 4

What do experts say about the clusters when evaluating them?
Three experts from the Psychiatry Department of the UMCU evaluated the clusters via an interview. All three experts confirmed during the evaluation that **the "severe cluster" is an exceptional cluster, as the features between the positive and negative symptoms seem to correlate.** That is, positive symptoms, such as *hallucinations* and *delusions* affect negative symptoms, like *sustaining daily activities* and *emotional expressions*. As such, the "severe cluster" has observations that show an increased severity in their *social* trait, because their *behavior* trait shows increased problematic scores. For the other two clusters, the **differences and correlation between the features are less obvious.** Thus, it is not known how these two clusters deviate from each other.

7.2 Conclusion of the main research question

Which steps comprehend the MAM for the cluster ensemble in the unsupervised domain, and to what extent can the cluster ensemble contribute to novel insights into mental disorders by utilizing and evaluating it to improve diagnosis and treatment that is in line with precision psychiatry?

Two objectives were at the heart of this study. The first objective concerned the development of a model that guides non-machine-learning experts in the unsupervised domain of machine learning by using the cluster ensemble. This objective resulted in the development of a generic MAM that follows the CRISP-DM cycle steps to cluster data. The resulting MAM is, therefore, a new development based upon previous work by Spruit and Jagesar (2016).

The second objective was the utilization of the cluster ensemble on data from the Psychiatry Department of the UMCU. These data consisted of patients who suffer from schizophrenia or psychosis. The results collected from this experiment reveal that there are three clusters in this dataset. We categorized these clusters as "severe," "mild," and "low problematic," in which the first cluster consists of patients who share some features that affect their social life and display some behavioral problems. Further evaluation with experts illustrated that there is some correlation between features in this cluster. For the other two clusters identified, it remains unclear which features exactly describe and differentiate them from each other, since we found no compelling evidence that the features deviate enough. As such, the experts were unable to find any correlation between the features nor differentiate them clearly.

Chapter 8

Discussion

8.1 Limitations

We distinguish the limitations of our study in three categories: 1. instrumental limitations; 2. data limitations; and 3. Research-design limitations.

Instrumental limitations are concentrated in the diceR package (version 0.5.1) (Chiu and Talhouk, 2018) used in our study to utilize the cluster ensemble. During our study, the package was still in development. Therefore, future updates may bring additional changes that could affect the end result of our study, for example, by adding more algorithms, such as DBSCAN, and by incorporating other consensus functions, such as HGPA. Moreover, diceR calculates the CDF, delta area, and AUC in a different way than, for example, ConsensusClusterPlus (Wilkerson and Hayes, 2010). That said, both programs provided different outcomes on various datasets and undoubtedly offered a different answer to our dataset in question. In addition, we could not explore the cluster ensemble using feature sampling. Unfortunately, this option was not available during our study. Thus, the end result may differ with feature sampling, as this always affects the cluster outcome.

Limitations to the data reside within the decisions made during our study and the availability of data. One of our decisions was to listwise delete rows if there were missing values in one of the HoNOS features. This decision resulted in the loss of 40% of the total data available during our study. The available data left was still sufficient for our study, however. Despite that, the impact of the data reduction remains unknown at this stage. We assume that the decision has affected the outcome in some way. Thus, it would be interesting to know what the outcome would have been if, for example, imputation by kNN had been used on the data to overcome the missing-values problem. Another decision was to use only the original 12 HoNOS features. However, the HoNOS dataset at the Psychiatry Department of the UMCU has three additional features that assess medication treatment, lack of motivation for treatment, and mania-like behavior of patients. Using only the 12 original features resulted in having the most observations following listwise deletion, which is one reason we decided to skip these three extra features. The other reason was to adhere closely to the original HoNOS features. In addition, GAF-scores (Global Assessment of Functioning) from the DSM dataset were not used in our study, as our main motivation was to use specifically the HoNOS features. For the availability of the data, we considered multiple datasets that were at our disposal at the Psychiatry Department of the UMCU. Two of these datasets were the Kennedy V-axis (Kennedy, 2008) and the PANSS (Positive and Negative Syndrome Scale) (Stanley, Fiszbein, and Opler, 1987) dataset. The latter dataset was specifically designed for schizophrenic and psychotic patients, as it monitors their positive and negative symptoms. However, after merging this dataset with the DSM and cleaning it, this resulted in a dataset that was considered too small for clustering (i.e. 150 observations in total). As such, it would be interesting to investigate which clusters are derived from the PANSS when a larger dataset is available. The same thinking applies to the Kennedy V-axis dataset, which was also too small for our study following merging and cleaning.

Finally, the research-design limitations concern the developed MAM in our study. Although we carefully described the MAM following literature from the machine-learning community, we still limited ourselves in an explanatory way. One of these limitations is the data-understanding part, which has many more methods than the two described in our study. It would be interesting to know which methods are exclusive to the unsupervised domain when dealing with high-dimensional data. Another limitation concerns the rigor of the MAM when used by non-machine-learning experts. The time allocated for this study did not allow us to experiment with the MAM, in addition to our own experimentation, on several real and synthetic datasets. Thus, it remains unclear whether non-machine-learning experts can benefit from the MAM to perform a cluster ensemble. Last, as the radar plot shows a similar shape between each cluster, it might seem that there is a confounding variable. This opens the debate whether some variables in the negative symptom spectrum, like *hallucinations or delusions*, *cognitive problems*, and *other mental and behavioral problems* need extra weights assigned to reduce their impact on creating clusters. Using this might give dissimilar patterns between clusters in the radar plot.

8.2 Future research

During the research, several opportunities for future research were identified. These future research possibilities are divided into two categories. First is the option for further empirical research with psychiatry data. The second category focuses on options for the cluster-ensemble MAM.

8.2.1 Empirical research with psychiatry data

In our research, we explored the concept of the cluster ensemble and psychiatry data consisting of patients who suffer from schizophrenia or psychosis. This exploration was performed using data from the HoNOS and DSM-IV. To the best of our understanding, there is little to no existing research using the cluster ensemble in the field of psychiatry, except for one study by Shen et al. (2007). Therefore, more empirical research is needed in the field of psychiatry with the cluster ensemble. As such, there are multiple options for exploration using the cluster ensemble and psychiatry data. One of these possibilities is using the HoNOS data together with the PANSS. Both sets of data are used to monitor patients, but the PANSS is more robust in assessing symptoms and behavior for the schizophrenic and psychotic spectrum. Thus, this approach may lead to other clusters or an improved clustering, as more features can contribute to novel clusters. Another option would be using the DSM-5 diagnosed patients with the HoNOS data. The DSM-5 has schizophrenic sub-types removed (i.e. catatonic, paranoid, residual, etc.). Thus, future research can explore the clusters more deeply as the DSM-5 is expected to be more reliable in the schizophrenic spectrum.

8.2.2 Empirical research with meta-algorithmic modeling

As our study only laid the groundwork for the cluster-ensemble MAM, it has not been experimented with in sufficient depth to be a reliable model. Although we evaluated the data modeling and evaluation fragment in Chapter 5, other fragments are lacking in terms of further evaluation. The cluster-ensemble MAM can benefit from further research if the model is examined more intensively in practice with actual participants. Benefits from the model may become more apparent via evaluation with before-after treatment (ex-ante, ex-post), such that the effectiveness and the validity of the model become evident. Naturally, extension of the model is another case for future work, such that the methods of current evaluation did not allow for: for instance, with improved data understanding methods to determine clusters beforehand, and to assign confidence of clusters obtained from the ensemble with significance testing. Finally, future research can explore ways to specialize the MAM: for instance, to be adapted for image analysis of fMRI scans.

Appendix A

R Code chapter 3

~~~
---
title: "Chapter 3 - Visualization"
output: html_notebook
---

This notebook is explores the steps needed according to the theory to perform EDA and data preparation before modeling of the cluster ensemble can begin.

In this notebook we will use the Wine dataset, and Breast Cancer Wisconsin dataset.


#Loading libraries and dataframes
```{r}
library(ggplot2)
library(heatmap3)
library(dplyr) #for chaining
library(GGally) #for pairing plots as in native R pairs
```

```{r}
wine <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data", header = F, sep = ",", stringsAsFactors = F)
wdbc <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data", header = F, sep = ",", stringsAsFactors = F)
#Set labels for Wine and Breastcancer
colnames(wine) <- c('Type', 'Alcohol', 'Malic', 'Ash',
                    'Alcalinity', 'Magnesium', 'Phenols',
                    'Flavanoids', 'Nonflavanoids',
                    'Proanthocyanins', 'Color', 'Hue',
                    'Dilution', 'Proline')
colnames(wdbc) <- c('ID','Diagnosis','V1','V2','V3','V4','V5','V6','V7','V8','V9','V10','V11','V12','V13','V14','V15','V16','V17','V18','V19','V20','V21','V22','V23','V24','V25','V26','V27','V28','V29','V30')
wdbc <- wdbc[,-c(1:2)]
```

#EDA on the Wine dataset
In chapter 3.2 we argue that we can use KDE and a heatmap to explore in a preliminary phase if our dataset in question contains some cluster structure. Note, that for small datasets, i.e. < 20 features, we can use KDE. If our dataset is bigger, we only want to explore it with the heatmap, since the number of plots becomes unfeasible to explore (plots = (1-p)p). Thus, for the Wine dataset we will use KDE, while for the wdbc dataset we will only use the heatmap.


```{r}
ggplot(wine, aes(Malic, Flavanoids)) + geom_density2d(linejoin = "round", position = "identity", lineend = "butt") + geom_jitter() + theme_bw()
```

```{r}
ggplot(wine, aes(Phenols, Alcohol)) + geom_density2d(linejoin = "round", position = "identity", lineend = "butt") + geom_jitter() + theme_bw()

```


From the heatmap we can directly spot that Proline and Magnesium have high variance, and that these probably belong to one or multiple classes in the dataset.
~~~

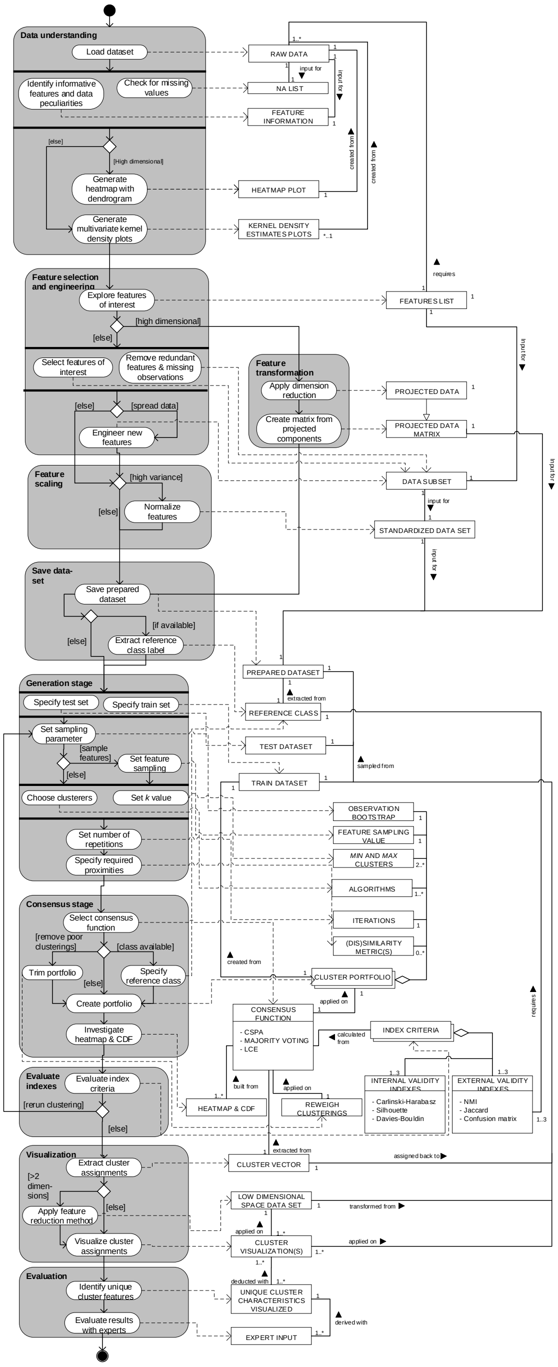
```

#EDA on the Breastcancer dataset

‘‘‘{r}
heatmap3(wdbc,ColSideCut=1.2,ColSideWidth=0.2,ColSideAnn = wdbc,showRowDendro = F ,
    col=topo.colors(3),RowAxisColors=1,legendfun=function() showLegend(legend=c("Low
    ", "Average","High"),col=topo.colors(3), title="Variance of features", cex = 1),
    verbose=F, na.rm = T)
’’’


# Scaling principle
‘‘‘{r}
#Create arbitrary car dataset
brand <- c("Maserati", "Bugatti", "Mercedes", "Porsche")
weight <- c(1300, 1315, 1360, 1325)
horsepower <- c(430, 443, 405, 415)
cars <- data.frame(brand, weight, horsepower)

#Plot without scale
ggplot(cars, aes(weight, horsepower)) + geom_point(aes(color = brand)) #Overview
    plot

#Cluster without scale
km_cars <- kmeans(cars[,c(2:3)], 2, 10, 5)
cars$cluster <- as.factor(km_cars$cluster)
ggplot(cars, aes(weight, horsepower)) + geom_point(aes(color = cluster)) + geom_text
    (aes(label=brand),hjust=-0.1, vjust=0) + coord_cartesian(xlim = c(1300, 1370)) +
     scale_color_manual(values = c("black", "red")) + theme_bw()

#Cluster with z-score scaling
cars <- cars[,c(1:3)]
cars_scale <- data.frame(scale(cars[,c(2:3)]))
cars_scale$brand <- brand
km_scale <- kmeans(cars_scale[,c(1:2)], 2, 10, 5)
cars_scale$cluster <- as.factor(km_scale$cluster)
ggplot(cars_scale, aes(weight, horsepower)) + geom_point(aes(color = cluster)) +
    geom_text(aes(label=brand),hjust=-0.1, vjust=0) + coord_cartesian(xlim = c(-1,
    1.65)) +  scale_color_manual(values = c("black", "red")) + theme_bw()
’’’



# Some Random Projection :)

‘‘‘{r}
#Load file, warning this is a bigger file than normal
tmpdir <- tempdir()
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/00401/TCGA-PANCAN-
    HiSeq-801x20531.tar.gz"
file <- basename(url)
download.file(url, file)

untar(file, compressed = ’gzip’, exdir = tmpdir )
list.files(tmpdir)
RNA <- read.table("data.csv", sep = ",", header = T)
RNA.labels <- read.table("labels.csv", sep = ",", header = T)
’’’

‘‘‘{r}
#Adjust dataset to make it a bit faster for processing, since it is only for
    explanatory purposes
RNA.labels <- RNA.labels$Class #Get samples
RNA <- RNA[,-1] #Remove samples from original
RNA$labels <- RNA.labels
’’’

‘‘‘{r}
RNA.dim <- ncol(RNA[,-20532]) #Number of dimensions in original space without labels
obs <- nrow(RNA) #number of obs in dataset
eps <- .5 #epsilon number determining loss of dimensions. Higher is lower dimensions
     but also a bit less accurate
RNA.dist <- as.matrix(dist(RNA[,-20532]))
rp.dim <- as.integer(ceiling(log(RNA.dim) / eps ^ 2)) + 1 #The number of dimensions
    for the RP matrix
```

~~~
```

```{r}
rp_matrix = matrix(rnorm(RNA.dim * rp.dim, 0, 1), RNA.dim, rp.dim) / sqrt(rp.dim)
```

```{r}
project.rp <- as.matrix(RNA[,-20532]) %*% rp_matrix #Create the projection matrix for RP, bound to the rp.dim
```


```{r}
rp.dataframe <- as.data.frame(project.rp) #Set to dataframe for ggplot
rp_plot <- ggplot(rp.dataframe, aes(V1, V24, color = RNA.labels)) + geom_point() + theme_bw() + labs(x="Gene expression 8", y="Gene expression 0")
rp_plot + theme(axis.ticks.x = element_blank(), axis.ticks.y = element_blank(), axis.text.x = element_blank(), axis.text.y = element_blank())
```

```{r}
original_plot <- ggplot(RNA, aes(gene_8, gene_0, color = RNA.labels)) + geom_point() + theme_bw() + labs(x="Gene expression 8", y="Gene expression 0")
original_plot + theme(axis.ticks.x = element_blank(), axis.ticks.y = element_blank(), axis.text.x = element_blank(), axis.text.y = element_blank(), legend.position = "none")
```


# Explaining the consensus visualizations

```{r}
library(diceR)
```

```{r}
wine.ensemble <- consensus_cluster(wine[,-1], nk = 3:4, p.item = 0.8, reps = 100, algorithms = c("km","hc"), hc.method = "single", distance = "euclidean", scale = T, type = "conventional", seed.data = 1234)

```

```{r}
ensemble.heatmap <- graph_heatmap(wine.ensemble)
```

```{r}
ensemble.cdf <- graph_cdf(wine.ensemble)
```

```{r}
ensemble.auc <- graph_delta_area(wine.ensemble)
```
~~~

chapter_3_–_experiment_datasets.Rmd

Appendix B

Complete meta-algorithmic method fragment



Appendix C

Cluster ensemble experiment R code

~~~
---
title: "Chapter 5 - Experimenting with synthetic and real data sets"
output: html_notebook
---

This notebook evaluates the accuracy of the cluster ensemble against standard single clustering algorithms.

In this notebook we will use several data sets available from the internet and the Iris dataset available within R.

#loading required libraries
```{r}
library(cluster)
library(diceR)
library(ggplot2)
library(EMCluster)
```

#loading real and synthetic data sets
```{r}
wine <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data", header = F, sep = ",", stringsAsFactors = F)
wdbc <- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data", header = F, sep = ",", stringsAsFactors = F)
iris <- iris
aggregation <- read.table("http://cs.joensuu.fi/sipu/datasets/Aggregation.txt", header = F, sep = "", stringsAsFactors = F)
toy <- read.table("http://cs.joensuu.fi/sipu/datasets/jain.txt", header = F, sep = "", stringsAsFactors = F)
abalone <- read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data", header = F, sep = ",", stringsAsFactors = F)
seeds <- read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/00236/seeds_dataset.txt", header = F, sep ="", stringsAsFactors = F)

#Set labels for WDBC, Abalone, Seeds, Aggregation and Toy

colnames(wdbc) <- c('ID','Diagnosis','V1','V2','V3','V4','V5','V6','V7','V8','V9','V10','V11','V12','V13','V14','V15','V16','V17','V18','V19','V20','V21','V22','V23','V24','V25','V26','V27','V28','V29','V30')
wdbc$Diagnosis <- as.factor(wdbc$Diagnosis)
wdbc.labels <- wdbc[,2]

colnames(abalone) <- c("Sex", "Length", "Diameter", "Height", "Whole_weight", "Shucked_weight", "Viscera_weight"
                       ,"Shell_weight", "Rings")
abalone$Sex <- as.factor(abalone$Sex)
set.seed(1)
abalone <- dplyr::sample_frac(abalone, size = 0.25, replace = T)
abalone.labels <- as.factor(abalone$Sex)

colnames(seeds) <- c("area", "perimeter", "compactness", "length", "width", "asymmetry", "kernel", "class")
seeds$class <- as.factor(seeds$class)
seeds.labels <- seeds[,8]

iris.labels <- iris[,5]
~~~

```

colnames(aggregation) <- c('X', 'Y', 'Cluster')
aggregation$Cluster <- as.factor(aggregation$Cluster)
aggregation.labels <- aggregation$Cluster

colnames(toy) <- c('X', 'Y', 'Cluster')
toy$Cluster <- as.factor(toy$Cluster)
toy.labels <- toy$Cluster

#Remove labels from original frames
abalone <- abalone[,-c(1,9)]
seeds <- seeds[,-8]
wdbc <-  wdbc[,-c(1,2)]
aggregation <- aggregation[,-3]
iris <- iris[,-5]
toy <- toy[,-3]

#set integer labels
int.iris.labels <- as.integer(iris.labels)
int.aggregation.labels <- as.integer(aggregation.labels)
int.wdbc.labels <- as.integer(wdbc.labels)
int.toy.labels <- as.integer(toy.labels)
int.abalone.labels <- as.integer(abalone.labels)
int.seeds.labels <- as.integer(seeds.labels)
‘‘‘


#Scale various data sets
‘‘‘{r}
wdbc <- data.frame(scale(wdbc))
abalone <- data.frame(scale(abalone))
seeds <- data.frame(scale(seeds))
‘‘‘

#single algorithms
## real data sets
We first begin with the Iris dataset. We know beforehand that the algorithm has to find 3 clusters. Therefore, we set it to k = 3. Afterwards we capture the cluster assignments and compare it to the original labels by a confusion matrix.
‘‘‘{r}
set.seed(1234)
km.iris <- kmeans(iris, centers = 3, iter.max = 10, nstart = 5)
km.iris.cluster <- as.factor(km.iris$cluster)
#Build confusion matrix
table(km.iris.cluster, iris.labels)
‘‘‘

ACC:   0.893


‘‘‘{r}
set.seed(1234)
km.abalone <-kmeans(abalone, centers = 3, iter.max = 10, nstart = 5)
km.abalone.cluster <- as.factor(km.abalone$cluster)
#Build confusion matrix
table(km.abalone.cluster, abalone.labels)
‘‘‘

ACC: 0.25

‘‘‘{r}
set.seed(1234)
km.seeds <- kmeans(seeds, centers = 3, iter.max = 10, nstart = 5)
km.seeds.cluster <- as.factor(km.seeds$cluster)
#Build confusion matrix
table(km.seeds.cluster, seeds.labels)
‘‘‘

ACC: 0.919

Next is the Wisconsis Breast Cancer dataset. A bigger dataset in terms of observations and features. There are 2 groups in this dataset. Benign and maligant. Again we will try the k-means to find and separate the two clusters.
```

```
‘‘‘{r}
set.seed(1234)
km.wdbc <- kmeans(wdbc, centers = 2, iter.max = 10, nstart = 5)
km.wdbc.cluster <- as.factor(km.wdbc$cluster)
#Build confusion matrix
table(km.wdbc.cluster, wdbc.labels)
‘‘‘

ACC: 0.910

## synthetic data sets
Starting with the aggregation dataset. Using k-means.
‘‘‘{r}
set.seed(1234)
km.aggregation <- kmeans(aggregation, centers = 7, iter.max = 10, nstart = 5)
km.aggregation.cluster <- as.factor(km.aggregation$cluster)
#Build confusion matrix
table(km.aggregation.cluster, aggregation.labels)
‘‘‘


‘‘‘{r}
ggplot(aggregation, aes(X,Y)) + geom_point(aes(color=aggregation.labels))
‘‘‘


ACC: 0.759

Final data set is the Toy dataset.
‘‘‘{r}
set.seed(1234)
km.toy <- kmeans(toy, centers = 2, iter.max = 10, nstart = 5)
km.toy.cluster <- as.factor(km.toy$cluster)
#Build confusion matrix
table(km.toy.cluster, toy.labels)
‘‘‘

ACC: 0.783



# Cluster ensemble
‘‘‘{r}
#specify algorithms
# Custom clustering algorithm
kaem <- function(d, k) {
return(as.integer(stats::kmeans(d, k, iter.max = 25, nstart = 12)$cluster))
}
assign("kaem", kaem, 1)

agnes <- function(d, k) {
return(as.integer(stats::cutree(cluster::agnes(d, diss = TRUE), k)))
}
assign("agnes", agnes, 1)


‘‘‘

## real data sets

‘‘‘{r}
ensemble.iris <- consensus_cluster(iris, nk = 3, reps = 300, p.item = 0.8,
     algorithms = c("kaem", "agnes", "pam", "gmm", "sc", "cmeans"), hc.method = "
     average", seed.data = 1234, distance = c("euclidean", "manhattan"))
ensemble.iris.evaluate <- consensus_evaluate(data = as.matrix(iris), ensemble.iris,
     trim = T, n = 3)
ensemble.iris.impute <- impute_missing(ensemble.iris.evaluate[["trim.obj"]][["E.new"
     ]][[1]], iris, 3)
mv.iris <- majority_voting(ensemble.iris.impute, is.relabelled = F)
ensemble.iris.cluster <- as.factor(mv.iris)
#Build confusion matrix
table(mv.iris, iris.labels)
‘‘‘
```

~~~

```{r}
ggplot(iris, aes(Petal.Length, Petal.Width, color = ensemble.iris.cluster)) + geom_point()
```


ACC: 0.913

```{r}
ensemble.abalone <- consensus_cluster(abalone, nk = 3, reps = 300, p.item = 0.8, algorithms = c("kaem", "cmeans", "pam", "sc"), distance = c("euclidean", "manhattan"), seed.data = 1234, minPts = 7, scale = T)
ensemble.abalone.evaluate <- consensus_evaluate(as.matrix(abalone), ensemble.abalone, trim = T, n = 1)
ensemble.abalone.impute <- impute_missing(ensemble.abalone.evaluate[["trim.obj"]][["E.new"]][[1]], abalone, 3)
mv.abalone <- majority_voting(ensemble.abalone.impute, is.relabelled = F)
table(mv.abalone, int.abalone.labels)
```

ACC: 0.336


```{r}
#Eerste run is zonder sc en cmeans
ensemble.seeds <- consensus_cluster(seeds, nk = 3, reps = 300, p.item = 0.8, algorithms = c("kaem", "agnes", "pam", "gmm", "sc", "cmeans"), hc.method = "average", distance = c("euclidean", "manhattan"), seed.data = 1234)
ensemble.seeds.evaluate <- consensus_evaluate(as.matrix(seeds), ensemble.seeds, trim = T, n = 1)
ensemble.seeds.impute <- impute_missing(ensemble.seeds.evaluate[["trim.obj"]][["E.new"]][[1]], seeds, 3)
mv.seeds <- majority_voting(ensemble.seeds.impute, is.relabelled = F)
table(mv.seeds, seeds.labels)
```

ACC: 0.924


```{r}
ensemble.wdbc <- consensus_cluster(wdbc, nk = 2, reps = 300, p.item = 0.8, algorithms = c("kaem", "agnes", "pam", "cmeans"), hc.method = "average", distance = c("euclidean", "manhattan"), seed.data = 1234)
mv.wdbc <- majority_voting(ensemble.wdbc[, , "PAM_Manhattan", "2"])
#Build confusion matrix
table(mv.wdbc,wdbc.labels)
```

ACC: 0.942

synthetic data sets

```{r}
ensemble.aggregation <- consensus_cluster(aggregation, nk = 7, p.item = 0.8, reps = 300, algorithms = c("kaem", "agnes", "pam", "gmm", "sc", "cmeans"), hc.method = "average", seed.data = 1234, distance = c("euclidean", "manhattan"))
ensemble.aggregation.evaluate <- consensus_evaluate(as.matrix(aggregation), ensemble.aggregation, trim = T, n = 2)
ensemble.aggregation_impute <- impute_missing(E = ensemble.aggregation.evaluate[["trim.obj"]][["E.new"]][[1]], aggregation, nk = 7)
mv.aggregation <- majority_voting(ensemble.aggregation_impute)
mv.aggregation <- as.factor(mv.aggregation)
#Build confusion matrix
table(mv.aggregation, aggregation.labels)
```

ACC: 0.995

```{r}
ggplot(aggregation, aes(X,Y, color = mv.aggregation)) + geom_point()
```
~~~

~~~


```{r}
ensemble.toy <- consensus_cluster(toy, nk = 2, reps = 300, algorithms = c("kaem", "sc"), hc.method = "average", seed.data = 1234, distance = c("euclidean", "manhattan"))
ensemble.toy.evaluate <- consensus_evaluate(as.matrix(toy), ensemble.toy, trim = T, n = 3)
ensemble.toy.impute <- impute_missing(ensemble.toy.evaluate[["trim.obj"]][["E.new"]][[1]], toy, nk = 2)
mv.toy <- majority_voting(ensemble.toy.impute, is.relabelled = F)
toy.clusters <- as.factor(mv.toy)
#Build confusion matrix
table(toy.clusters, toy.labels)
```

ACC: 0.853

Plot
Build plot for document to show differences in accuracy between ensemble and standard algorithm

```{r}
#Create data frame with results
dataset <- c(rep("WDBC",2), rep("Iris",2), rep("Seeds", 2), rep("Abalone", 2), rep("Toy",2), rep("Aggregation",2))
algorithms <- c("K-means", "Cluster ensemble")
results <- c(0.910,0.942,0.893,0.913,0.919,0.924,0.25,0.336,0.783,0.853,0.759,0.995)
overall.results <- data.frame(dataset, algorithms, results)


#Create bar graph
accuracy_plot <- ggplot(overall.results, aes(dataset, results, fill = algorithms)) + geom_bar(stat = "identity", position = "dodge", width = .6) + theme_bw() + labs(x="Datasets", y="Accuracy") + scale_fill_manual(values = c("black","gray")) + scale_y_continuous(breaks = pretty(overall.results$results, n=6))
accuracy_plot + geom_hline(aes(yintercept = 0.752, linetype = "K-means"), color = "red") + annotate("text", min(overall.results$results), 0.752, vjust = -0.25, hjust = -0.2, label = ".75") + geom_hline(aes(yintercept = 0.827, linetype = "Cluster ensemble"), color = "green") + annotate("text", min(overall.results$results), 0.827, vjust = -0.25, hjust = -0.2, label = ".83") + scale_linetype_manual(name = "Average accuracy", values = c(1,1), guide = guide_legend(override.aes = list(color = c("green", "red"))))
```
~~~

chapter_5_–_experiments.Rmd

Appendix D

Data-driven diagnosis R code

This notebook is used for the master thesis Data-driven Diagnosis in Psychiatry by Wouter van der Klift. Every file used in this notebook is property of UMC Utrecht, department psychiatry, division hersenen.

The notebook is build as follows: 1: Loading the necessary files, and showing some simple descriptive statistics to see what each file contains. 2: Understanding each file by some useful visualizations 3: Preparing the files by removing unused variables/features/-columns, removing NA's when necessary, and calibrating some observations/rows and finally merging to one last dataframe and subsetting it. 4: Modeling by beginning with basic cluster analysis algorithms, subsequently performing cluster ensembles on the dataframe 5: Evaluation of the cluster ensembles

When applicable and necessary a comment is given by a R-chunk piece to elaborate the what, and why.

Files used in this R-notebook are the HoNOS, DBC, Kennedy and PANSS.

```r
# 1. Loading the files and libraries for data analysis

library(ggplot2)
library(reshape2)
library(tidyverse)
library(dplyr)
library(lubridate)

raw.dbc <- read.csv("~/Werkbestanden/werkbestand_dbc.csv", header = T,
sep = ";", as.is = T, na.strings=c("", NA))
raw.honos <- read.csv("~/Werkbestanden/werkbestand_honos.csv", header = T,
sep = ";", as.is = T, na.strings=c("", NA))
raw.kennedy <- read.csv("~/Werkbestanden/werkbestand_kennedy.csv", header = T,
sep = ";", as.is = T, na.strings=c("", NA))
raw.panss <- read.csv("~/Werkbestanden/werkbestand_panss.csv", header = T,
sep = ";", as.is = T, na.strings=c("", "NA"))

#Since many variables, show the summary of each variable class
table(sapply(raw.dbc, class))

#Subset dataframe with numerical variables, ease of use for descriptive statistics
x <- raw.dbc[sapply(raw.dbc, is.numeric)]
logical <- raw.dbc[sapply(raw.dbc, is.logical)]

summary(x)
summary(logical)
```

Apparently as2_4 untill AS2_4_Omsch3 are all NA, so these are to be removed later. Furthermore we observe that the rest of the variables that are of type numeric, are related to time of the DBC, their unique dbc number etc. Rest is of character and will be explored later on.

```r
### 1.1.2 HoNOS
table(sapply(raw.honos, class))
y <- raw.honos[sapply(raw.honos, is.numeric)]
logical2 <- raw.honos[sapply(raw.honos, is.logical)]
summary(y)
summary(logical2)
```

Type_respondent to Opleidingsniveau are all empty. Furthermore, Totaalscore_SBGGZ has a high mean score compared to its median and also a huge Max. The same is for Totaal_score_HoNOS with a high mean and very high Max.

```
### 1.1.3 Kennedy
table(sapply(raw.kennedy, class))
z <- raw.kennedy[sapply(raw.kennedy, is.numeric)]
summary(z)
```

The Kennedy shows that there aren't empty variables, but the NA's are around 50% per variable.

```
### 1.1.4 PANSS
table(sapply(raw.panss, class))
w <- raw.panss[sapply(raw.panss, is.numeric)]
summary(w)
```

Variables look fine

```
# 2. Data understanding
## 2.1 DBC Understanding
```

First we will go through the DBC. The DBC contains the diagnostic criteria and treatment plans for the patients, and is known as the diagnose-behandelcombinatie. Thus, this file contains the pathway for a patient with a specific diagnosis and its treatment plan. Subsequently it contains scores from the GAF, the Global Assessment of Functioning.

Based on the diagnosis/impairment, a patient might have an extensive diagnosiscode. For example, someone with a diagnosis code on column diagnosecode as1_1.05.01.01 has a broad (not in detail) 'Gecombineerde type' in column diagnoseomschrijving. This, of course, does not give much information. So, breaking down the column diagnosecode with row as1_1.05.01.01 in (as1_1) -> (as1_1.05) -> (as1_1.05.01) -> (as1_1.05.01.01) will result in the following diagnosis: (as1_1 = stoornissen in de kindertijd), (as1_1.05 = Aandachtstekortstoornissen en gedragsstoornissen), (as1_1.05.01 = Aandachtstekortstoornis met hyperactiviteit), and (as1_1.05.01.01 = Gecombineerde type). Thus, to describe this patient we can have Aandachtstekortstoornis met hyperactiviteit gecombineerde type. On the other hand we can also have a patient whos diagnoseomschrijving is 'Dysthyme stoornis', with a diagnosecode of as1_6.01.02. This patient can be broken down as followed (as1_6) -> (as1_6.01) -> (as1_6.01.02), resulting in: (as1_6 = Stemmingsstoornissen), (as1_6.01 = Depressieve stoornis), and (as1_6.01.02 = Dysthyme stoornis). Lastly, going having a long diagnosecode such as as1_4.04.02.03.01 is also often not clear what their impairment is in the diagnoseomschrijving. By breaking this down into its parts we have (as1_4 = Aan een middel gebonden stoornis), (as1_4.04 = Aan cannabis gebonden stoornissen), (as1_4.04.02 = Stoornissen door cannabis), (as1_4.04.02.03 = Psychotische stoornis door cannabis), (as1_4.04.02.03.01 = Met wanen), that we might see as: Psychotische stoornis door cannabis met wanen.

Further down in the dataset there are columns like AS1_1_Diagnosecode, AS1_1_Omschrijving, AS1_1_Omsch1 etc. These columns also represent a diagnsosis that is established for a patient. So, patients will receive a definitive diagnosis, but can also be diagnosed with additional impairments, since there is overlap in disorders. For example, addiction might be in cannabis, but it is also likely that this is in alcohol.

Since there are no sexes and ages within this dataframe, we will focus on the length of stay, the diagnosis, and a combination of these two.

```
ggplot(raw.dbc, aes(Duur_DBC)) + geom_histogram(na.rm = T, binwidth = 10) +
theme_classic()
```

It seems that there are some patients with a DBC length of 0, while many have over a length of 350.

```
#Count total observations with DBC duur = 0
table(raw.dbc$Duur_DBC == 0)
raw.dbc %>% select(diagnosecode10, diagnosegroep2omschrijving, Duur_DBC) %>%
filter(Duur_DBC == 0 ) %>% ggplot(., aes(diagnosecode10)) + geom_bar(na.rm = T)
table(raw.dbc$zorgtypeomschrijving)
raw.dbc %>% ggplot(., aes("X-value",Duur_DBC)) + geom_boxplot(na.rm = T) +
labs(title="Duur DBC", y = "Length of stay") + theme_classic() +
stat_summary(aes(label=round(..y..,2)), fun.y=mean, geom="text", size=4, vjust = -0.5, na.rm = T) + stat_summary(aes(label=round(..y..)), fun.y = median, geom="text", size = 4, vjust = 1, na.rm = T) + scale_y_continuous(breaks = pretty(raw.dbc$Duur_DBC, n = 10))
```

Based on Openstaande DBC we can infer that these are patients without a closed DBC.

```
raw.dbc %>% select(OpenstaandeDbc, Einddatum) %>% count(OpenstaandeDbc, Einddatum)
```

Now lets check which disorders are mostly represented in the DBC.

```
#Check which disorders are most represented in descending order
raw.dbc %>%
  filter(!is.na(diagnosegroep1omschrijving)) %>%
  count(diagnosegroep1omschrijving) %>%
  arrange(desc(n)) %>%
  group_by(diagnosegroep1omschrijving)

raw.dbc %>%
  count(diagnosegroep3omschrijving) %>%
  arrange(desc(n)) %>%
  group_by(diagnosegroep3omschrijving)
```

Now we want to see the amount of NA's in the DBC.

```
#Overview of NA's within both dataframes
na.dbc <-  raw.dbc %>% select(everything()) %>%
summarise_all(funs(sum(is.na(.)))) %>% gather(., "variable") %>% filter(value >0)
na.dbc <- na.dbc[order(na.dbc$value, decreasing = T),]

na.dbc %>% ggplot(., aes(variable, value)) + geom_bar(stat = "identity", width = 0.5) +
labs(title="Missings in raw.dbc") + geom_hline(yintercept = 25000) + theme_classic() +
theme(axis.text.x = element_blank())
```

It seems that a lot of missing values are present in the DBC. However, we cannot impute this values so far, and so we are going to delete these columns that contain 80% or more of missing values.

Lets see if there are patients that already have been diagnosed with the DSM-5

Since diagnosecode, diagnosecode9 & diagnosecode10 represent the coding of the disorders by the ICD, let's see if there are some patients without such diagnosiscode.

```
raw.dbc %>% group_by(PseudoID) %>% filter(is.na(diagnosecode9)&is.na(diagnosecode10))
```

We now turn to the HoNOS (Health of the Nation Outcome Scales). The HoNOS is an instrument consisting of 12 items that measures the behavior, impairment, symptoms and social functioning of people with severe mental illness. This instrument is being assessed within the department of psychiatry since the first half of 2011.

In comparison with the DBC, the HoNOS has some useful variables to look further into, e.g. sexes, answered questions etc.

```r
#Frequency between sexes
raw.honos %>% group_by(Geslacht) %>%
  summarise(n=n()) %>%
  mutate(freq = n / sum(n))
#Age distribution between sexes
raw.honos %>% arrange(Geslacht) %>%
  ggplot(., aes(Geslacht, Leeftijd_honos, fill = Geslacht)) +
  geom_boxplot(na.rm = T) + theme_classic() + labs(title = "Age between sex", y = "Leeftijd") + stat_summary(aes(label=round(..y..,2)), fun.y=mean, geom="text", size=4, vjust = -0.5) + stat_summary(aes(label=round(..y..)), fun.y = median, geom="text", size = 4, vjust = 1) +
  scale_y_continuous(breaks = seq(0,100,10))
#It looks like that there are patients of juvenile age
raw.honos %>% ggplot(., aes(Leeftijd_honos)) +
geom_histogram(aes(y = ..density..), fill = "lightskyblue", na.rm = T) +
  geom_density(alpha = 0.3, fill = "orange", color = "orange") +
    scale_x_continuous(breaks = pretty(raw.honos$Leeftijd_honos, n = 10)) +
    theme_classic() + labs(title="Histogram of ages", x = "Age range", y = element_blank())
#Totaalscore HoNOS plotted by gender
is_outlier <- function(x) {
  return(x < quantile(x, 0.25) - 1.5 * IQR(x) | x > quantile(x, 0.75) + 1.5 * IQR(x))
}

select(raw.honos, Geslacht, Totaal_score_HoNOS) %>%
  filter(!is.na(Totaal_score_HoNOS)) %>% arrange(Totaal_score_HoNOS) %>%
    mutate(outlier = ifelse(is_outlier(Totaal_score_HoNOS),
      Totaal_score_HoNOS, as.numeric(NA))) %>%
    ggplot(., aes(Geslacht, Totaal_score_HoNOS, fill = Geslacht)) +
    geom_boxplot(outlier.color = "red") + geom_text(aes(label = outlier),
      na.rm = TRUE, hjust = -0.4, size = 3) + theme_classic()
```

Turns out that the Totaal_score_HoNOS column has some wrong values. So this needs to be fixed in the next section. It also seems that there is a column named Geldige_rom_meting, which has a Ja if the HoNOS is valid.

```r
raw.honos %>% mutate(Geldige_ROM_meting, as.factor(Geldige_ROM_meting)) %>%
  ggplot(., aes(Geldige_ROM_meting)) + geom_bar() + theme_classic() +
    labs(title="Geldige ROM meting", x = element_blank())
```

The observations as a Nee indicates that this is not a valid HoNOS, so these have to be removed. The NA's indicate that this is an older type of HoNOS (the first version) that has been assessed in the past. The column Aantal_vragen_te_gaan is an indicator whether the HoNOS is successfuly assessed. If more than three questions remain unanswered, it can be considered as a Nee in the Geldige_ROM_meting.

```r
raw.honos %>%
  ggplot(., aes(Aantal_vragen_te_gaan, Percentage_beantwoord, fill = Geldige_ROM_meting)) +
    geom_histogram(na.rm = T) + theme_classic()
#Checking distribution between Ja and Nee ROM METING
geldigerom <- c("Ja", NA_character_)
raw.honos %>% filter(Geldige_ROM_meting %in% geldigerom) %>%
  ggplot(., aes(Totaal_score_HoNOS, fill = Geldige_ROM_meting)) +
    geom_histogram(position = "dodge", aes(y = ..density..)) +
      stat_function(fun = dnorm, color = "red",
           args = list(mean = mean(raw.honos$Totaal_score_HoNOS),
             sd = sd(raw.honos$Totaal_score_HoNOS))) + theme_classic()
```

Next, we want to check if each question is correctly filled in following the Likert-scale (i.e. 1 to 4)

```r
#First we melt all the integer variables together in a dataframe, so that we can plot them in a single graph
```

```
V_Scores_molten <- melt(data = raw.honos, id.vars = "PseudoID", measure.vars = c("V1_Hyperactief_aggressief_destructief_of_geagiteerd_gedrag",
"V2_Opzettelijke_zelfverwonding", "V3_Problematisch_alcohol_of_druggebruik",
"V4_Cognitieve_problemen",
"V5_Lichamelijke_problemen_of_handicaps",
"V6_Problemen_als_gevolg_van_hallucinaties_en_waanvoorstellingen",
"V7_Problemen_met_depressieve_stemming", "V8_2_Overige_psychische_en_gedragsproblemen_B",
"V9_Problemen_met_relaties", "V10_Problemen_met_ADL", "V11_Problemen_met_woonomstandigheden", "V12_Mogelijkheden_voor_het_gebruik_en_verbeteren_van_vaardigheden_beroepsmatig_en_vrije_tijd"),
variable.name = "variable")
V_Scores_molten %>% ggplot(., aes(variable, value, color = variable)) + geom_jitter(na.rm = T)
```

Although it looks somewhat unclear, we can definitely see that most questions are answered within the scoring range of 0-4. So there are no transformation outliers. Furthermore, the most upper part are the scores which are 9. Meaning that these remained unanswered during the HoNOS interview. The same we will apply to the other scoring measures, the A1 to A3 scoring

```
A_Scores_molten <- melt(data = raw.honos, id.vars = "PseudoID", measure.vars = c("A1_Problemen_ten_gevolgen_van_maniforme_ontremming", "A2_Problemen_ten_gevolgen_van_gebrek_aan_motivatie_voor_behandeling", "A3_Problemen_ten_gevolgen_van_een_gebrek_aan_compliance_met_medicatie" ), na.rm = T)

A_Scores_molten %>% ggplot(., aes(variable, value, color = variable)) +
  geom_jitter() + theme_classic()
```

In these data there are also no peculiarities.

DATUM_expl checks whether observations have a plausible date

```
table(raw.honos$DATUM_expl)
```

Apparently there are 13 observations which have no plausible date. So these will be removed in the preparation step.

```
## Kennedy Understanding
```

The Kennedy assessment instrument, also known as Kennedy Axis V (K Axis), is used to capture the 'level of functioning' from a patient in six areas, subsequently also in one other area such as:

1. psychological impairment; (correlates with the GAF-score, and defines; 'no symptoms', 'psychosis', 'anxiety/depression', 'anti social skills', and 'lack of motivation') 2. social skills; (purely assesses the skills of the patient) 3. violence; (assesses violence by consciousness) 4. activities of daily living; (ADL, how an individual performs during daily living, i.e. at work or education) 5. substance abuse; (Whether a person is abusing drugs/alcohol, although hard to measure as they can lie) 6. medical impairment; (How somatic problems limit the functioning of a person) and 7. other problems; (Housing problems, financial problems, problems with the law) It is originally developed for use with the DSM-III-R Axis V, but its use has also been applied to the DSM-IV (Higgins and Purvis, 2000).

In contrast to the GAF, the Kennedy provides an overview of the level of functioning of a patient on all the different subscales. Therefore, the Kennedy is a bit more sophisticated as it measures the functioning of a person on a specific level (Ebben, 2006). These measurements in the Kennedy can range from a low of 0 (considered dysfunctional) to a high of 100 (no symptoms). However, the score of 50 cannot be seen as 'average', as healthy people have an average score of 85 on the Kennedy (Ebben, 2006).

The Kennedy measures the functioning of a patient over a period in time, such that it can indicate whether symptoms occur over time, and not just in one time frame.

The Kennedy file consists of 15 character columns and 13 of type integer. Every integer column from the Kennedy is complimented with a character column containing specific text information about the patient at that time.

Because we do not work with free text, lets first list the amount of NA's in the integer columns and plot it.

```
missing.kennedy <- raw.kennedy[, sapply(raw.kennedy, class) == "integer"] %>%
  summarise_all(funs(sum(is.na(.)))) %>% gather(., "variable") %>% filter(value >0)
  missing.kennedy <- missing.kennedy[order(missing.kennedy$value, decreasing = T),]
ggplot(missing.kennedy, aes(variable, value, fill = variable)) +
  geom_bar(stat = "identity", width = 0.5) +
      labs(title="Missings in kennedy") + theme(axis.text.x = element_blank()) +
        theme_classic()
```

It seems that many questions have missings. The next thing that would be interesting to know is which observations have missings in every integer column that is a question, and also has empty character columns.

```
raw.kennedy[,c(4:20)] %>% filter(complete.cases(.))

#Taking a look for duplicates
raw.kennedy %>% group_by(PseudoID) %>% filter(n()>1) %>%
  summarize(number=n()) %>%
  ggplot(., aes(number)) + geom_histogram(bins = 40) + theme_classic() +
      labs(title="Duplicates of PseudoID's")
```

There are roughly 600+ duplicates with PseudoID's, with the largest having 74 duplicates. Next, we want to check which duplicates have NA values in all there integer columns, since these are then unusable.

```
raw.kennedy %>% group_by(PseudoID) %>% filter(n()>1) %>%
  summarise_at(vars("Psychische_problemen"), .funs = sum(is.na(.)))
raw.kennedy[, sapply(raw.kennedy, class) == "integer"] %>%
  group_by(PseudoID) %>% select_if(function(x) any(is.na(x))) %>%
    summarise_each(funs(sum(is.na(.))))
```

```
## 2.4 PANSS Understanding
```

PANSS is one of the smallest dataframes being used. So, first we are going to see how many observations have NA values in their columns.

```
#Check for missing values
na.panss <-  raw.panss %>% select(everything()) %>%
  summarise_all(funs(sum(is.na(.)))) %>%
    gather(., "variable") %>% filter(value >0)
na.panss %>% ggplot(., aes(variable, value)) + geom_bar(stat = "identity", width =
    0.5) +
  labs(title="Missings in raw.panss") + theme_classic()
#Check for duplicates
raw.panss %>% group_by(PseudoID) %>% filter(n()>1) %>% summarise(duplicates = n())
    %>%
  ggplot(., aes(duplicates)) + geom_histogram() + theme_classic()
#Overview of gender
raw.panss %>% group_by(Geslacht) %>%
  summarise(n=n()) %>%
  mutate(freq = n / sum(n)) %>%
  ggplot(., aes(Geslacht, n)) + geom_bar(stat = "identity") + theme_classic() +
  labs(title="Count between sexes in PANSS", y="Total number")
#Age distribution
  raw.panss %>% filter(!is.na(Geslacht)) %>%
  ggplot(., aes(Geslacht, Leeftijd_panss, fill = Geslacht)) + geom_boxplot() +
  theme_classic() + stat_summary() +
```

```r
  stat_summary(aes(label=round(..y..,2)), fun.y=mean, geom="text", size=4, vjust = -0.5) +     stat_summary(aes(label=round(..y..)), fun.y = median, geom="text", size = 4, vjust = 1)
```

```r
# 3. Data Preparation
## 3.1 DBC Preparation
#First remove columns that are completely empty, which are 6
raw.dbc <- subset(raw.dbc[,which(unlist(lapply(raw.dbc, function(x)!all(is.na(x)))))], with = F)
#Remove columns with 80% or more NA
raw.dbc <- raw.dbc[, colSums(!is.na(raw.dbc))>=5180]

#Remove unused columns
raw.dbc <- select(raw.dbc, -zorgvraagzwaarte, -zorgtypecode, -zorgtypeomschrijving,
  -zorglijncode, -zorglijnoms, -afsluitredencode, -afsluitredenoms, -beroepsgroepnaam,
    -beroepsnaam, -artscode, -diagnoseomschrijving, -diagnosegroep1, -diagnosegroep2,
    -diagnosegroep3, -diagnosegroep4, -gafstartoms, -gafeindoms, -gafhoogoms, -as1_1,
    -as2_1, -as3_1, -as4_1, -as4_2, -factuurbedrag, -AS1_1_DiagnoseCode, -AS1_1_Omschrijving,
    -AS1_1_Omsch1, -AS1_1_Omsch2, -AS1_1_Omsch3, -AS1_1_Omsch4, -AS2_1_DiagnoseCode,
    -AS2_1_Omschrijving, -AS2_1_Omsch1, -AS2_1_Omsch2, -AS3_1_DiagnoseCode, -AS3_1_Omschrijving,
    -AS3_1_Omsch1, -AS4_1_DiagnoseCode, -AS4_1_Omschrijving, -AS4_2_DiagnoseCode, -AS4_2_Omschrijving)

#Remove patients with 0
raw.dbc <- filter(raw.dbc, Duur_DBC > 0)

#Remove observations that do not have a diagnosiscode
raw.dbc <- raw.dbc %>% filter_at(vars(diagnosecode, diagnosecode9, diagnosecode10), any_vars(!is.na(.)))

#Remove observations that took place with the DSM-5
raw.dbc <- filter(raw.dbc, !grepl("D5_", diagnosecode))

#Remove observations with both NA in diagnosecode9 & 10
raw.dbc <- raw.dbc %>% filter_at(vars(diagnosecode9, diagnosecode10), any_vars(!is.na(.)))

#Remove observations without diagnosis and disorders that come without an actual impairment, i.e. relatieproblemen
nodiagnose <- c("Andere aandoeningen en problemen die een reden voor zorg kun",
  "Andere problemen die een reden van zorg kunnen zijn",
    "Bijkomende codes/ geen diagnose",
    "Bijkomende problemen die een reden voor zorg kunnen zijn")
raw.dbc <- raw.dbc %>% filter(!(diagnosegroep1omschrijving %in% nodiagnose))

#Remove observations with Openstaande DBC = JA
raw.dbc <- filter(raw.dbc, OpenstaandeDbc == "Nee")

#Set date variables correct
raw.dbc <- raw.dbc %>%
  mutate_at(vars(Startdatum, Einddatum, DiagnoseDatum, GafScoreDatum), funs(as.Date))

#Fix conversion errors
dbc.characters <- raw.dbc[, sapply(raw.dbc, class) == "character"] %>%
  #Fix text
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "cocaA??negebruik"), replacement = "cocainegebruik")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "cocaA$??ne"), replacement = "cocaine")) %>% #Cocaine
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "opioA$??de"), replacement = "opioide")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "opioA$??degebruik"), replacement = "opioidegebruik")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "PassagA$re Tic−stoornis"), replacement = "Passagere Tic−stoornis")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "CoA??rdinatieontwikkelingsstoornis"),
    replacement = "Coordinatieontwikkelingsstoornis")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "ParanoA??de type"), replacement = "Paranoide type")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door (vermeld de somatische aandoening op"),
    replacement = "door somatische aandoening")) %>%

  #Alter the text
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door..(vermeld de somatische aandoening)"),
    replacement = "door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door (vermeld de somatische aandoening"),
  replacement = "door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door (vermeld de somatische aandoening)"),
    replacement = "door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "beA??nvloede"), replacement = "beinvloed")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Stoornissen aan een ander (of onbekend) middel gebonden"),
    replacement = "Stoornissen door een ander middel")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Angststoornissen door een middel isse"),
    replacement = "Angststoornissen door een middel")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Aan amfetamine (of een amfetamine verwant middel) gebonden s"),
    replacement = "Aan amfetamine (of een amfetamine verwant middel) gebonden stoornis")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Persoonlijkheidsverandering door (vermeld de somatische aand"),
    replacement = "Persoonlijkheidsverandering door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Katatone stoornis door somatische aandoening)"),
    replacement = "Katatone stoornis door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Voedings− en eetstoornis op zuigelingenleeftijd of de vroege"),
    replacement = "Voedings− en eetstoornis op zuigelingenleeftijd")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Gebonden aan zowel psychische factoren als een somatische aa"),
    replacement = "Gebonden aan zowel psychische factoren als een somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Persisterende dementie door middelen teweeggebracht (verwijs"),
    replacement = "Persisterende dementie door middelen")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Slaapstoornis door somatische aandoening as3)"),
    replacement = "Slaapstoornis door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Amnestische stoornis door (vermeld somatische aandoening)"),
    replacement = "Amnestische stoornis door somatische aandoening")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "Slaapstoornis door een middel (verwijs naar stoornissen aan"),
    replacement = "Slaapstoornis door een middel")) %>%

#Remove unneccesary text
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door Ac??A|.. (vermeld hier somatische aandoening die ni"),
    replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(codeer ook 331.1 ziekte va"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(codeer ook 333.4 zie"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(codeer ook 04"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(codeer ook 845.00 schedeltr"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(codeer elke middelspecifie"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(niet door een somatische aandoening)"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "of in de ado"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "of vroege"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "door..(vermeld de somatische aandoening)"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "doorAc??A|(vermeld de soma"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
   ., fixed(pattern = "door (vermeld de somatische aandoening op as3)"), replacement = "")) %>%
  mutate_at(vars(everything()), .funs = ~ str_replace(
    ., fixed(pattern = "(verwijs naar de stoorn"), replacement = ""))

#Match and overwrite columns from dbc.characters with unique.dbc
raw.dbc[,names(dbc.characters)] <- dbc.characters
rm(dbc.characters)
```

Create new column which is a truncate of diagnosegroep2omschrijving and diagnosegroep3omschrijving. For example, cluster a, b, and c —represented in diagnosegroep2omschrijving— does not tell much about a disorder, while in diagnosegroep3omschrijving it can state borderline peroonlijkheidsstoornis.

```r
#Create vector for use in filtering based on these conditions
conc.columns <- c("Conversiestoornis", "Kortdurende psychotische stoornis",
  "Obsessieve-compulsieve-stoornis", "Persoonlijkheidsverandering",
    "Posttraumatische stress-stoornis", "Schizoaffectieve stoornis", "Schizofrenie",
    "Sociale fobie", "Specifieke fobie", "Waanstoornis")

#Filter dataframe unique.dbc on diagnosegroep2 & 3 based on the conditions from conc.columns
diagnosegroup2_3 <- raw.dbc %>%
  select(PseudoID, dbcnummer, diagnosegroep2omschrijving, diagnosegroep3omschrijving) %>% filter(diagnosegroep2omschrijving %in% conc.columns) #For unique.dbc

#Paste diagnosegroep2 & 3 together in new column named concat, so that the information about the disorder is more detailed
diagnosegroup2_3 <- mutate(diagnosegroup2_3,
  concat = paste(diagnosegroep2omschrijving, diagnosegroep3omschrijving))

#Replace NA strings, so not actual NA values, by nothing, simply removes the NA text character
diagnosegroup2_3 <- diagnosegroup2_3 %>%
  mutate_at(vars(concat), .funs = ~ str_replace(., pattern = "NA", replacement = ""))

#Remove original columns and retain only the concatenated column
diagnosegroup2_3 <- diagnosegroup2_3[, -c(3:4)]

#Change concat column name back to diagnosegroep2omschrijving
names(diagnosegroup2_3)[names(diagnosegroup2_3)=="concat"] <- "diagnosegroep2omschrijving"
```

Create new column which is a replacement of diagnosegroep2omschrijving with diagnosegroep3omschrijving, as the disorder is more detailed in the latter. For example, Aandachtstekortstoornissen en gedragsstoornissen is less detailed than Oppositioneel-opstandige gedragsstoornis, or Borderline persoonlijkheidsstoornis instead of cluster b.

```r
#Based on some handwork these disorders from diagnosegroep2omschrijving
#need information from diagnosegroep3omschrijving in order to be complete/detailed.
disorders <- c("Aandachtstekortstoornissen en gedragsstoornissen",
  "Aan amfetamine (of een amfetamine verwant middel) gebonden stoornis",
    "Amnestische stoornissen", "Bipolaire stoornissen", "Cluster a", "Cluster b",
    "Cluster c", "Andere cognitieve stoornissen", "Communicatiestoornissen", "Delirium",
    "Dementie", "Depressieve stoornissen", "Leerstoornissen", "Nagebootste stoornis",
    "Overige stoornissen op zuigelingenleeftijd, kinderleeftijd o", "Aanpassingsstoornis",
    "Pervasieve ontwikkelingsstoornissen", "Pijnstoornis", "Primaire slaapstoornissen",
    "Psychotische stoornis", "Slaapstoornissen die samenhangen met een andere psychische s",
    "Stoornissen in de motorische vaardigheden", "Tic-stoornissen")

#Create dataframe with a filter applied to select observations only matching those in diagnosegroep2
group2 <- raw.dbc %>%
  select(PseudoID, dbcnummer, diagnosegroep2omschrijving, diagnosegroep3omschrijving) %>% filter(diagnosegroep2omschrijving %in% disorders)

#Change diagnosegroep2omschrijving with diagnosegroep3omschrijving, since this is more detailed
group2$diagnosegroep2omschrijving <- group2$diagnosegroep3omschrijving

#Remove diagnosegroep3omschrijving
group2 <- group2[,-4]
```

Now that new dataframes are created (diagnosegroup2_3 and group2) the next process is to combine these (rowbind) and use this new dataframe for merging with raw.dbc. In raw.dbc the diagnosegroep1omschrijving will be hoofddiagnose, meanwhile df.diagnosegroepen is used to replace diagnosegroep2omschrijving and diagnosegroep3omschrijving, in order to have detailed information about the disorder.

```r
#Bind rows from group2 and diagnosegroup2_3 together
df.diagnosegroepen <- rbind(group2, diagnosegroup2_3)
raw.dbc <- merge(raw.dbc, df.diagnosegroepen, by = "dbcnummer", all.x = T)

#Create new column diagnosegroep2omschrijving that if diagnosegroep2omschrijving
#has a NA value, take the other value from diagnosegroep2omschrijving
raw.dbc <- mutate(raw.dbc, diagnosegroep2omschrijving <- ifelse(is.na(diagnosegroep2omschrijving.y), diagnosegroep2omschrijving.x, diagnosegroep2omschrijving.y))

#Remove the original columns diagnosegroep2omschrijving and diagnosegroep3omschrijving, plus the other two from df.diagnosegroepen .x & .y and only keep the column made from the previous line
raw.dbc <- select(raw.dbc,
  -diagnosegroep2omschrijving.x, -diagnosegroep2omschrijving.y,
    -diagnosegroep3omschrijving,  -PseudoID.y)
raw.dbc <- raw.dbc[c(2, 1, 3:10,20,11:19)]
names(raw.dbc)[11]<- "diagnose_detail"
names(raw.dbc)[1] <- "PseudoID"
names(raw.dbc)[names(raw.dbc)=="diagnosegroep1omschrijving"] <- "hoofddiagnose"

raw.dbc <- ungroup(raw.dbc)
#raw.dbc <- raw.dbc %>% arrange(PseudoID) %>% group_by(PseudoID)
#Remove unneccesary columns
#raw.dbc <- select(raw.dbc, -diagnosecode)

#Clean environment
rm(df.diagnosegroepen, diagnosegroup2_3, group2, na.dbc, conc.columns, disorders, nodiagnose)
```

At this point we have a dataframe called raw.dbc which has the main diagnosis and the detailed diagnosis (a concatenation of diagnosegroep2 and diagnosegroep3). Furthermore, observations have been removed that are out of the scope (DSM-5), still have an open DBC, did not have a diagnosis (by diagnosecode, diagnosecode9 or 10) and had a mysterious length of stay with 0 days.

Now, we turn deeper into the DBC, by subsetting it with unique patients and duplicated patients (patients that have multiple disorders and therefore reoccur in the DBC, and duplicated patients with one disorder but reoccur due to their length of stay), and calculating their GAF scores. Later on, these will be combined back into one masterfile of the DBC.

```
#Calculate length of stay per patient
format.dbc <- raw.dbc %>%
  group_by(PseudoID, hoofddiagnose, diagnose_detail) %>%
    mutate(Total_Duur_DBC = sum(Duur_DBC)) %>% mutate(Times_in_DBC = (count = n()))

#Filter upon duplicates and unique.
format.dbc <- format.dbc %>%
  gather(format.dbc, gaf, gafstart_ondergrens,
      gafhoog_ondergrens, gafeind_ondergrens, na.rm = T) %>%
    group_by(PseudoID, hoofddiagnose, diagnose_detail, Startdatum,
      Einddatum, OpenstaandeDbc, Total_Duur_DBC, Times_in_DBC) %>%
    summarise(gaf_mean = round(mean(gaf)))
```

```
## 3.2 HoNOS

raw.honos$Totaalscore_SBGGZ <- raw.honos$Score_SBGGZ_1_2 +
  raw.honos$Score_SBGGZ_2_2 #Update Totaalscore_SBGGZ
raw.honos$Totaal_score_HoNOS <- ifelse(is.na(raw.honos$Totaalscore_SBGGZ),
  raw.honos$Totaal_score_HoNOS, raw.honos$Totaalscore_SBGGZ)
#Replace Totaal_score_HoNOS with Totaalscore_SBGGZ only if Totaalscore_SBGGZ has a value

#Remove NA
raw.honos <- filter(raw.honos, !is.na(Totaal_score_HoNOS))
#Remove geen Geldige ROM meting
geldigerom <- c("Ja", NA_character_)
raw.honos <- filter(raw.honos, Geldige_ROM_meting %in% geldigerom)
#Remove no plausible date
raw.honos <- filter(raw.honos, DATUM_expl == "Plausibele datum")
#Remove variables with only NA values in it
raw.honos <- subset(raw.honos[,which(unlist(lapply(raw.honos, function(x)!all(is.na(x)))))], with = F)

#Remove unused variables
format.honos <- select(raw.honos, -BEANTWID, -obs, -Instructie,
  -V1_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V2_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V3_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V4_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V5_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V6_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V7_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V8_1_Overige_psychische_en_gedragsproblemen_A,
    -V8_1_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V8_2_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V9_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V10_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V11_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -V12_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -A1_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -A2_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -A3_Toelichting_argumentatie_score_huidige_hulpbronnen,
    -DATUM_expl, -Geldige_ROM_meting, -Aantal_vragen_gereed,
    -Aantal_vragen_te_gaan, -Aantal_vragen_totaal, -Score_SBGGZ_1_2, -Score_SBGGZ_2_2)

#Set 9 to NA
format.honos[, c(2:13,15:17)][format.honos[, c(2:13,15:17)] == 9] <- NA

#Remove obs with only NA
honos.vragen <- c("V1_Hyperactief_aggressief_destructief_of_geagiteerd_gedrag",
"V2_Opzettelijke_zelfverwonding", "V3_Problematisch_alcohol_of_druggebruik",
```

```r
"V4_Cognitieve_problemen", "V5_Lichamelijke_problemen_of_handicaps", "V6_Problemen_als_gevolg_van_hallucinaties_en_waanvoorstellingen",
"V7_Problemen_met_depressieve_stemming", "V8_2_Overige_psychische_en_gedragsproblemen_B",
"V9_Problemen_met_relaties", "V10_Problemen_met_ADL", "V11_Problemen_met_woonomstandigheden", "V12_Mogelijkheden_voor_het_gebruik_en_verbeteren_van_vaardigheden_beroepsmatig_en_vrije_tijd", "A1_Problemen_ten_gevolgen_van_maniforme_ontremming", "A2_Problemen_ten_gevolgen_van_gebrek_aan_motivatie_voor_behandeling", "A3_Problemen_ten_gevolgen_van_een_gebrek_aan_compliance_met_medicatie")
format.honos <- filter_at(format.honos, .vars = honos.vragen, any_vars(!is.na(.)))

#Transform date
format.honos <- format.honos %>% mutate_at(vars(DATUM), funs(as.Date))

#Save prepared HoNOS
final.honos <- format.honos[,-c(14,18:20)]
```

```r
## 3.3 Kennedy

#Remove character columns starting with Toelichting
format.kennedy <- select(raw.kennedy, -matches("Toelichting_"))

#Remove other unneccessary columns
format.kennedy <- select(format.kennedy,
  -BEANTWID, -obs, -Vragen_en_voorstellen_voor_beleid,
    -Opnametijd, -Ontslagtijd)

#Remove obs with NA in each question
kennedy.vragen <- c("Psychische_problemen", "Sociale_vaardigheden",
  "Risico_op_agressie_zelfbeschadiging", "ADL_beroepsmatig_functioneren",
    "Middelenmisbruik", "Somatische_problemen", "Bijkomende_problemen",
    "Motivatie_voor_behandeling")
format.kennedy <- filter_at(format.kennedy, .vars = kennedy.vragen, any_vars(!is.na(.)))

#Set columns to date
format.kennedy <- format.kennedy %>%
  mutate_at(vars(DATUM, Opnamedatum, Ontslagdatum), funs(as.Date))

#Replace Opnamedatum by DATUM if Opnamedatum is NA
format.kennedy$Opnamedatum[is.na(format.kennedy$Opnamedatum)] <-
  format.kennedy$DATUM[is.na(format.kennedy$Opnamedatum)]

#Round up variables
format.kennedy <- format.kennedy %>% mutate_at(vars(kennedy.vragen), funs(round))

format.kennedy <- format.kennedy %>% filter_at(vars(kennedy.vragen), any_vars(!(.)<=1))

#Create column Qtr since some patients from the DBC can reappear with a different diagnosis.
#Thus the kennedy score should represent the diagnosis in that year.
format.kennedy <- format.kennedy %>%
  group_by(PseudoID, Qtr=year(DATUM))
calc.kennedy <-  format.kennedy %>% group_by(PseudoID, Qtr, DATUM) %>% summarise_at(vars(kennedy.vragen), mean, na.rm = T) %>% mutate_at(vars(kennedy.vragen), funs(round)) %>% slice(which.min(DATUM))

#Select patient info columns from original kennedy,
#ie Leeftijd_kennedy & Geslacht, then set age to last known age per patient
#and Opnamedatum to first.
ages.kennedy <- format.kennedy %>%
  group_by(PseudoID, Geslacht, Qtr) %>%
    summarise(Leeftijd_kennedy = last(Leeftijd_kennedy))

#Merge ages.kennedy together with calc.kennedy to create the final.kennedy
final.kennedy <- right_join(ages.kennedy, calc.kennedy, by = c("PseudoID","Qtr"), all.x = T)
```

```r
#PANSS

#Keep columns we want
format.panss <- select(raw.panss, -Rater_1, -Rater_2, -Datum_interview, -Toelichting, -OpnameID)

#Remove obs with NA
format.panss <- format.panss %>%
  filter_at(vars(Total_positive, Total_negative, Total_general, Overall_total),
    any_vars(!is.na(.)))
format.panss <- format.panss %>%
  filter_at(vars(Total_positive, Total_negative, Total_general),
    any_vars(!is.na(.)))
format.panss <- format.panss %>%
  filter_at(vars(Total_negative, Total_general),
  any_vars(!is.na(.)))

#Correct date
format.panss <- format.panss %>% mutate_at(vars(DATUM, Opnamedatum, Ontslagdatum), funs(as.Date))

#We take median if patients reappear more than twice in the panss
median.panss <- format.panss %>% group_by(PseudoID) %>% filter(n()>2)
median.panss <- median.panss %>% group_by(PseudoID) %>%
  summarise_at(vars(Total_positive, Total_negative, Total_general, Overall_total), median)
#Below <3 is mean
mean.panss <- format.panss %>% group_by(PseudoID) %>% filter(n()<=2)
mean.panss <- mean.panss %>% group_by(PseudoID) %>%
  summarise_at(vars(Total_positive, Total_negative, Total_general, Overall_total), mean)

#Merging panns
calc.panss <- rbind(median.panss, mean.panss)
ages.panss <- format.panss %>% group_by(PseudoID, Geslacht) %>%
  summarise(Opnamedatum = first(Opnamedatum), Leeftijd_panss = last(Leeftijd_panss))
final.panss <- right_join(ages.panss, calc.panss, by = "PseudoID")
```

```r
#Merging dataframes
x <- left_join(format.dbc, final.honos, by = "PseudoID") %>%
  filter_at(vars(honos.vragen), any_vars(!is.na(.)))
x <- x[,c(1:5,25,6:24,26,27)] #Reorder columns for overview
#Create column diffdate which checks if the date from the HoNOS falls between the DBC date. Output Boolean.
x['diffDate'] <- ymd(x$DATUM) %within% interval(ymd(x$Startdatum), ymd(x$Einddatum))
#Again, reoder placing diffDate next to the dates of DBC and HoNOS
x <- x[,c(1:6,28,7:27)]
#Keep obs with bool = T
x <- x %>% filter(!is.na(diffDate) & diffDate == TRUE)
#Remove obs where Startdatum and Einddatum are duplicates due to their HoNOS scoring
#and only retain the last value from DATUM (their HoNOS score)
x <- x %>% group_by(PseudoID) %>% distinct(Startdatum, Einddatum, .keep_all = T)
#x <- x %>% group_by(PseudoID, Geslacht, hoofddiagnose, diagnose_detail) %>%
  summarise_at(vars(honos.vragen), sum, na.rm = T) #Sum up HoNOS questions
#Create dbc.honos and reorder some columns
dbc.honos <- x[,c(1,27:28, 2:26)]

x_dbc.kennedy <- left_join(format.dbc, final.kennedy, by = "PseudoID") %>% filter(!is.na(Qtr))
x_dbc.kennedy <- x_dbc.kennedy[,c(1:3,10,4:5,13,6:9,11,14:21)]
x_dbc.kennedy['diffDate'] <- ymd(x_dbc.kennedy$DATUM) %within%
  interval(ymd(x_dbc.kennedy$Startdatum), ymd(x_dbc.kennedy$Einddatum))
x_dbc.kennedy <- x_dbc.kennedy %>% filter(diffDate == TRUE)
colnames(x_dbc.kennedy)[7] <- "DATUM_kennedy"
x_dbc.kennedy <- x_dbc.kennedy[,-21]
calc.dbc.ken <- x_dbc.kennedy[duplicated(x_dbc.kennedy[c("PseudoID", "diagnose_detail")]) |
duplicated(x_dbc.kennedy[c("PseudoID", "diagnose_detail")], fromLast = T),]
calc.dbc.ken <- calc.dbc.ken %>%
  group_by(PseudoID, hoofddiagnose, diagnose_detail, Geslacht, OpenstaandeDbc,Total_Duur_DBC,Times_in_DBC) %>%
    summarise_at(vars(gaf_mean,kennedy.vragen), mean) %>% mutate_at(vars(gaf_mean,kennedy.vragen ),
    funs(round(.,2)))
others.dbc.ken <-  x_dbc.kennedy[!duplicated(x_dbc.kennedy[1:3]),]
others.dbc.ken <- others.dbc.ken[,-c(5:7,12)]
dbc.kennedy <- full_join(others.dbc.ken, calc.dbc.ken) #Full join
rm(x_dbc.kennedy, calc.dbc.ken, others.dbc.ken)

dbc.panss <- left_join(format.dbc, final.panss, by = "PseudoID") %>%
  filter_at(vars(Total_positive, Total_negative, Total_general, Overall_total),
      any_vars(!is.na(.))) #Remove obs without any panss scoring
dbc.panss <- dbc.panss[,c(1:3,10,12,4:5,11,6:9,13:16)] #Reorder columns
dbc.panss['diffDate'] <- ymd(dbc.panss$Opnamedatum) %within% interval(ymd(dbc.panss$Startdatum),
ymd(dbc.panss$Einddatum))
dbc.panss <- dbc.panss %>% filter(diffDate == T)
dbc.panss <- dbc.panss[,-17] #Remove diffDate column
```

```
#Modeling
library(dplyr)
library(diceR)
library(kernlab)
library(fastcluster)
library(rgl)
library(Rtsne)
library(ggplot2)
library(reshape2)

dbc <- read.csv("~/dbc_honos.csv", sep = ";", header = T, stringsAsFactors = F)
schizofrenie <- dbc %>% filter(hoofddiagnose == "Schizofrenie en andere psychotische stoornissen")
set.seed(1)
train <- schizofrenie
train <- train[,c(1,5,14:25)]

#Rank normalizing
train_range <- train[,c(3:14)]
range_norm <- function(x){(x-min(x))/(max(x)-min(x))}
train_range <- as.data.frame(range_norm(train_range))

#Ensemble run
ensemble_rank <- dice(train_range, nk = 3:6, reps = 500,
  algorithms = c("km", "pam", "cmeans"), cons.funs = "majority",
    trim = T, reweigh = T, seed = 12345, plot = T, progress = T)

#Cluster sizes
table(ensemble_rank$clusters)
rank_clusters <- ensemble_rank$clusters
rank_clusters <- as.factor(rank_clusters)

#Visualization
library(MASS)
train_euc <- dist(train_range)
#train_euc <- amap::Dist(train_range[-c(63,215),], method = "euclidean", diag = F, upper = F)

fit <- cmdscale(train_euc,eig = T, k=2)

#Rotate the points, so that the clusters are represented from minimum to maximum, ie green to blue
point1 <- -fit$points[,1]
point2 <- fit$points[,2]
points_mds <- data.frame(point1, point2,rank_clusters)
visualize_mds <- ggplot(points_mds, aes(point1, point2, col = rank_clusters, shape = rank_clusters)) +
  geom_point()
visualize_mds + theme_bw() +
  ggtitle(label = "3 Clusters", subtitle = "Cluster Ensemble outcome visualized in reduced dimensional space") +
    stat_ellipse(aes(x=point1, y=point2,color=rank_clusters, group=rank_clusters), type = "euclid", level = -0.6) +
    scale_color_manual(name = "Clusters", labels = c("1", "2", "3"), values = c("red", "green", "blue")) +
    scale_shape_manual(name = "Clusters", labels = c("1","2","3"), values = c(1,2,3)) + labs(x = "X-axis", y = "Y-axis")

#Create cluster groups
original_groups <- select(train, diagnose_detail)
#Add clusters to vector
cluster_groups <- train
cluster_groups$cluster <- rank_clusters
df_cluster1 <- cluster_groups %>% filter(cluster == 1)
df_cluster2 <- cluster_groups %>% filter(cluster == 2)
df_cluster3 <- cluster_groups %>% filter(cluster == 3)

#Calculate median per variable per cluster and per disorder
median_clusters <- df_cluster_all %>%
  group_by(diagnose_detail, cluster) %>%
    summarise_at(.vars = c(3:14), funs(median(.)))
colnames(median_clusters) <- c("diagnose_detail","clusters",
  "H.A.D.of_geagiteerd_gedrag", "Zelfverwonding",
    "Alcohol_of_druggebruik", "Cognitieve_problemen",
    "Fysieke_problemen_of_handicaps", "Hallucinaties",
    "Depressieve_stemming",
    "Overige_psychische_en_gedragsproblemen",
    "Problemen_met_relaties", "Problemen_met_ADL",
    "Problemen_met_woonomstandigheden", "V.V.VT_en_werk")
median_clusters$count <- c(6,9,1,3,10,1,109,141,46,18,26,14,19,11,4,4,4,1,13,13,12,117,81,50,7,1,5,9,1,5,3)


#Create first bar graph, this can be copied to other features by specifying their name.
median_clusters$HAD0 = ifelse(median_clusters$H.A.D.of_geagiteerd_gedrag != 0, median_clusters$count, "")
ggplot(median_clusters, aes(diagnose_detail, H.A.D.of_geagiteerd_gedrag)) +
  geom_bar(stat = "identity", position =  position_dodge(width = 0.4),
    aes(fill = clusters)) + geom_text(aes(label = HAD0, group = clusters),
    position = position_dodge(width = 0.4) ,vjust = -0.15, size = 5, hjust = 0.6) +
    labs(title = "Median per cluster of H.A.D.of_geagiteerd_gedrag", x="Disorders", y=element_blank()) +
    theme_bw() + theme(text = element_text(size=10), axis.text.x = element_text(angle=90, hjust=1))
```

Appendix E

Some descriptive statistics from psychiatry data

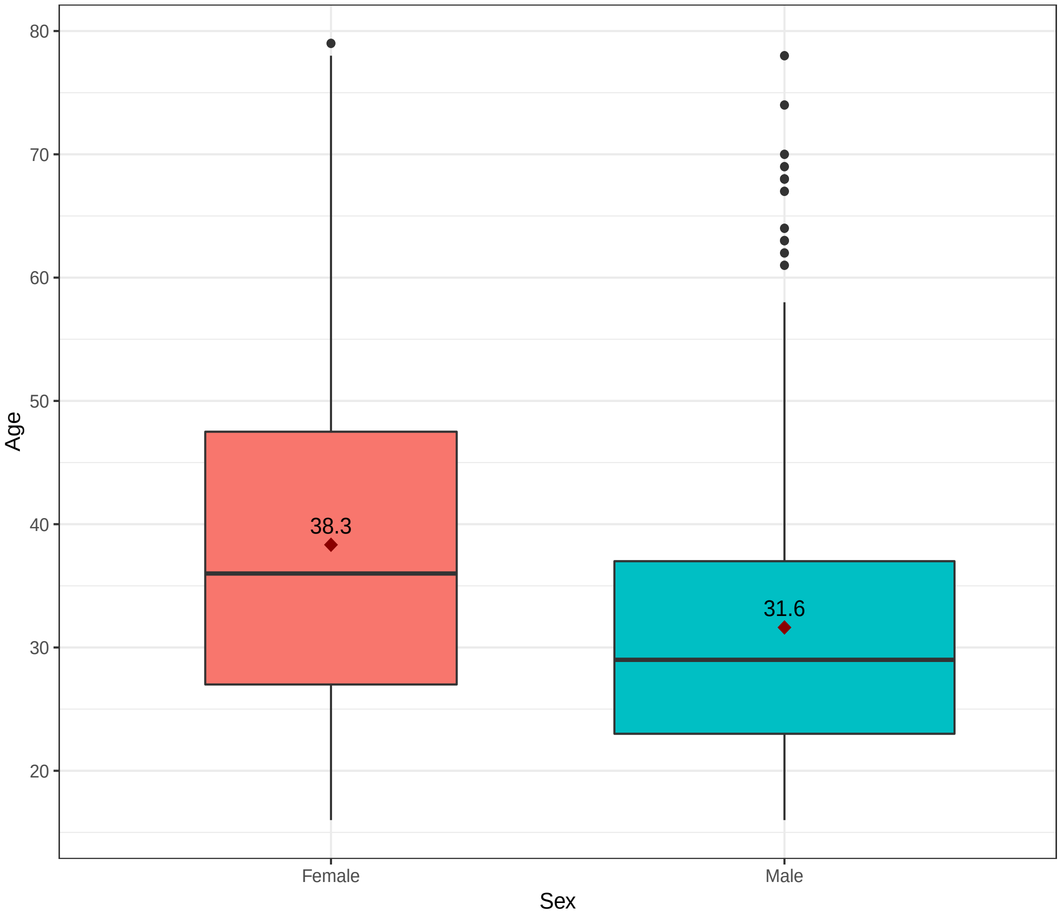


FIGURE E.1: Boxplot showing the age range of patients separated by sex. Average age indicated with red marker. Minimum age is 16 years old, to a maximum age of 78 for males and 79 for females.

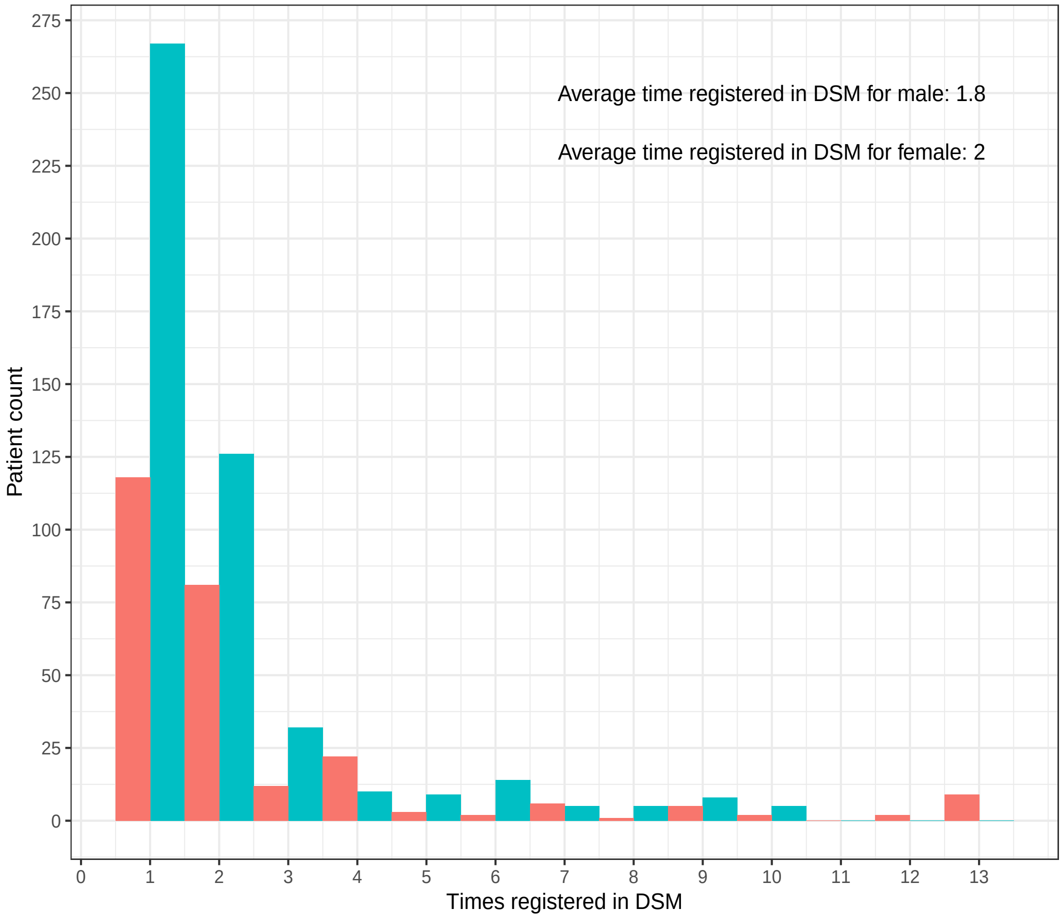


FIGURE E.2: Histogram showing times of patients being re-hospitalized in the DSM.

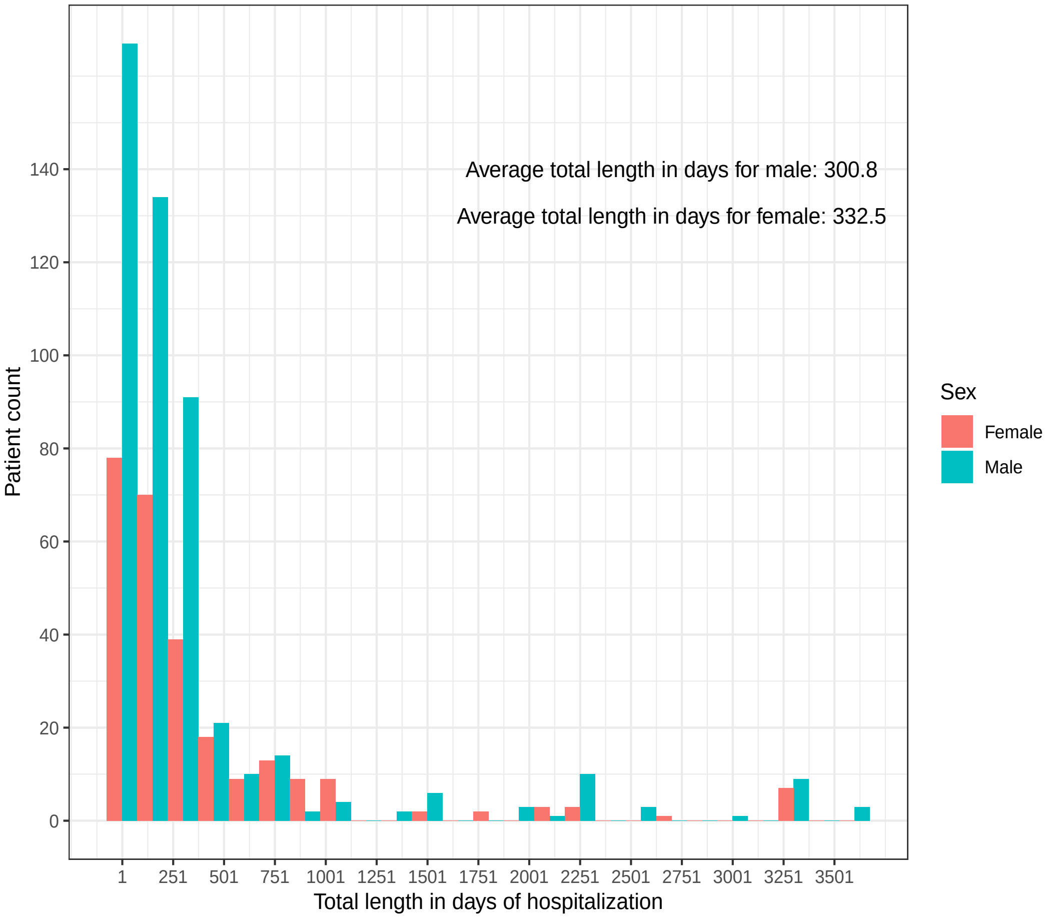


FIGURE E.3: Histogram showing total length of hospitalization days for patients based on each time they reoccur in the DSM.

Appendix F

Additional heatmaps from the cluster ensemble

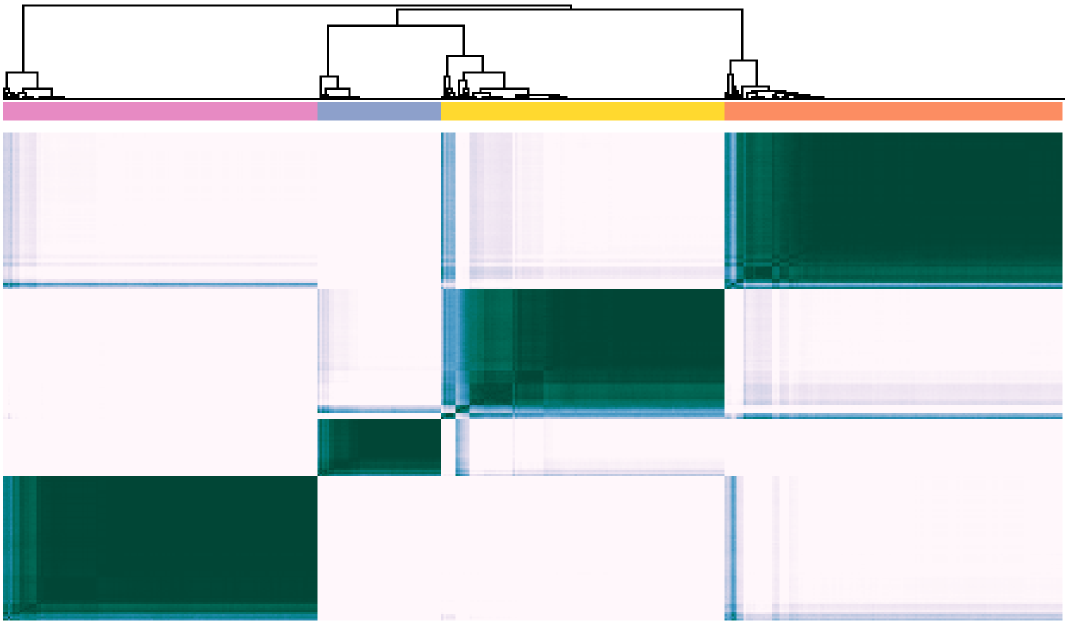


FIGURE F.1: Cluster ensemble heatmap for $k = 4$ with K-means

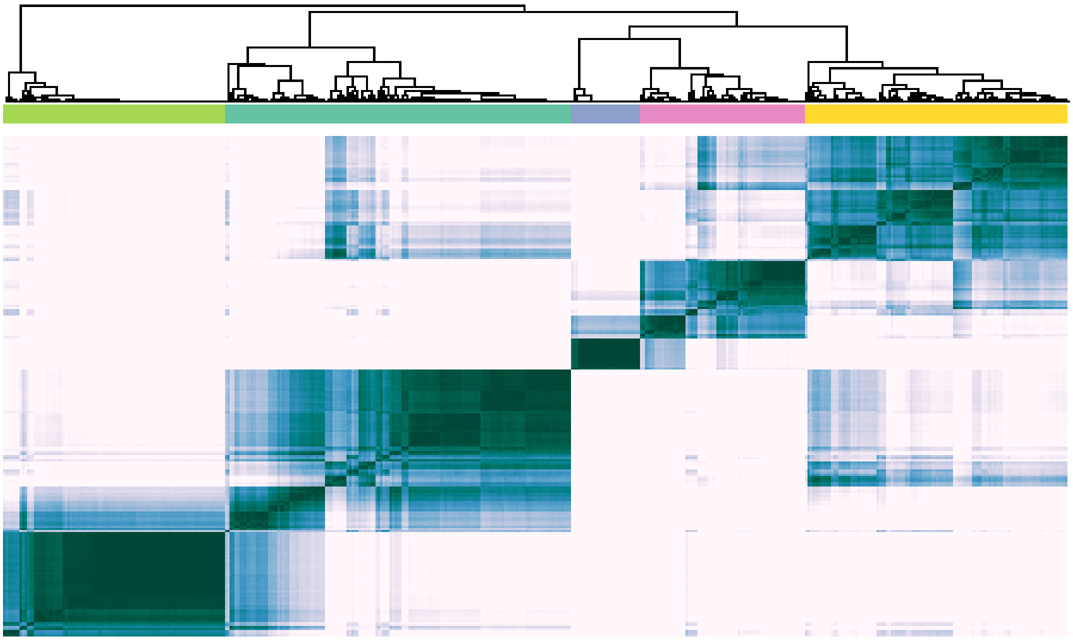


FIGURE F.2: Cluster ensemble heatmap for $k = 5$ with K-means

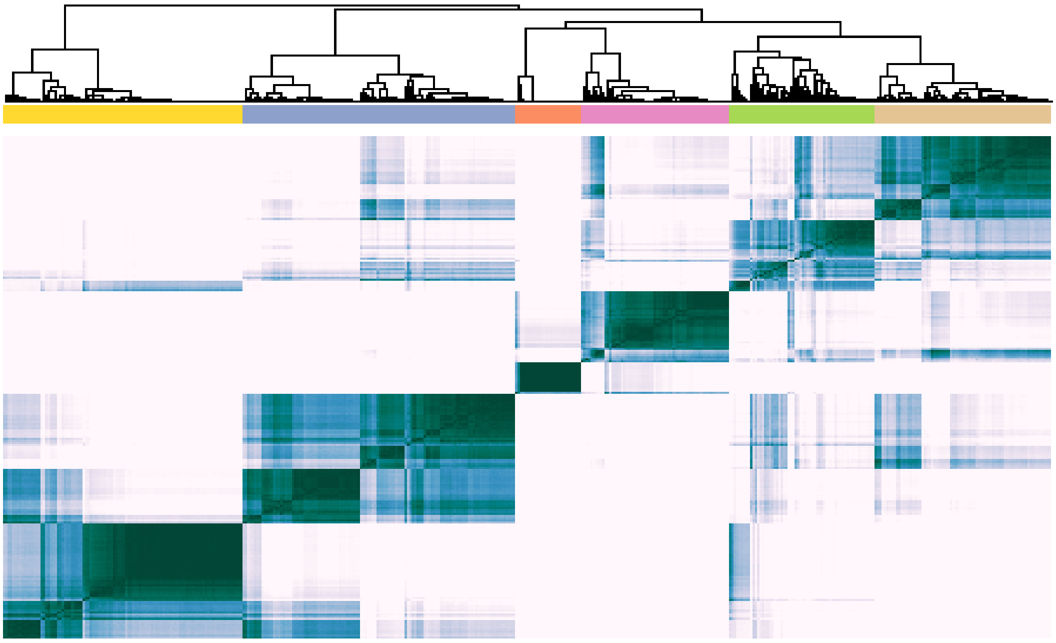


FIGURE F.3: Cluster ensemble heatmap for $k = 6$ with K-means

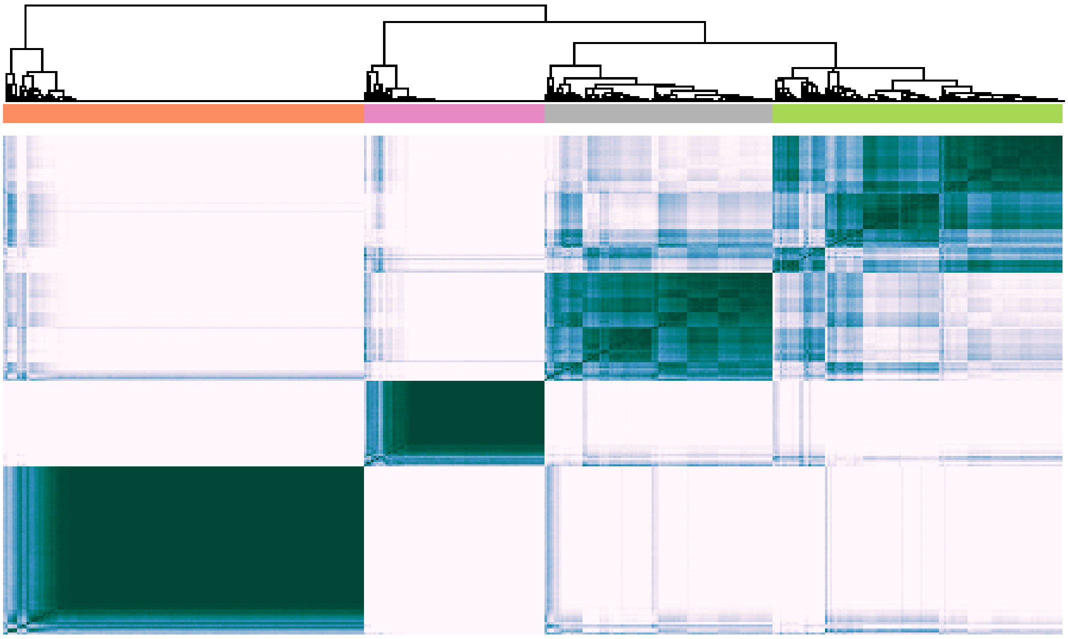


FIGURE F.4: Cluster ensemble heatmap for $k = 4$ with C-means

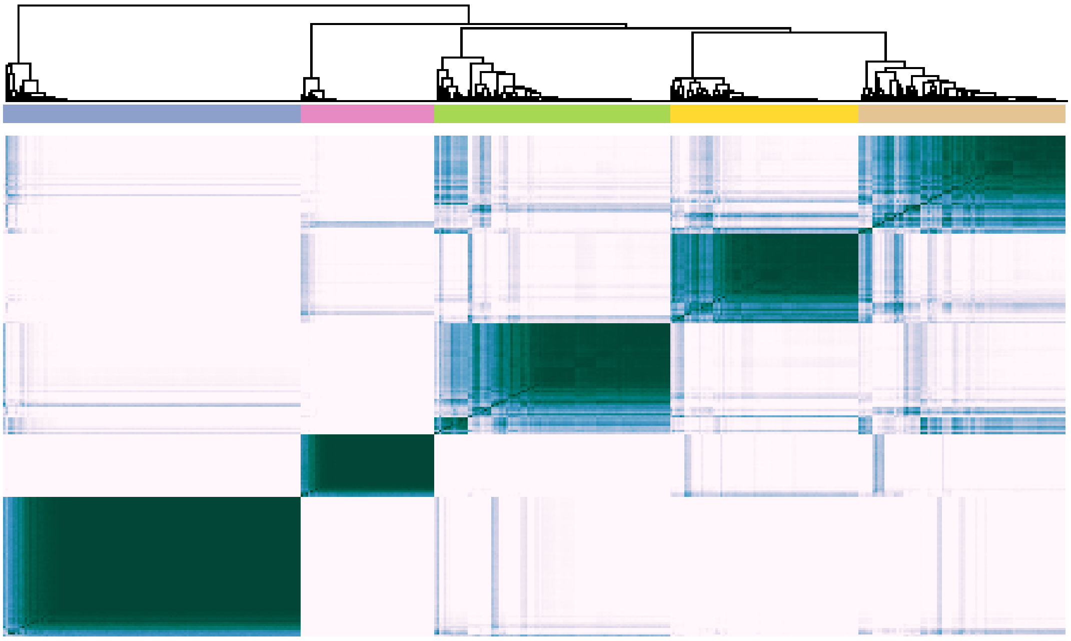


FIGURE F.5: Cluster ensemble heatmap for $k = 5$ with C-means

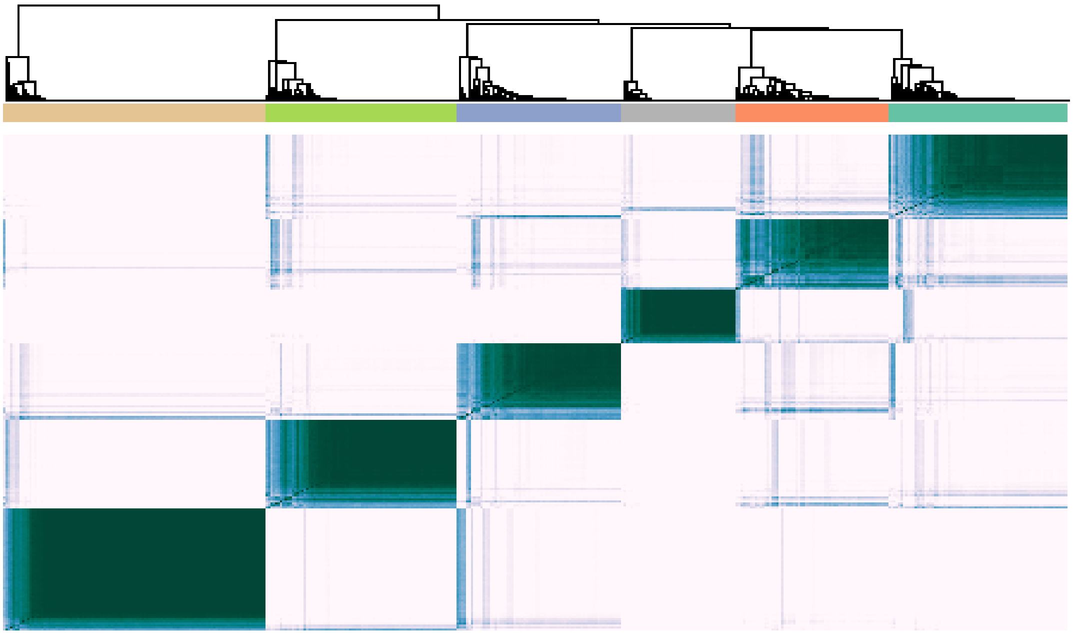


FIGURE F.6: Cluster ensemble heatmap for $k = 6$ with C-means

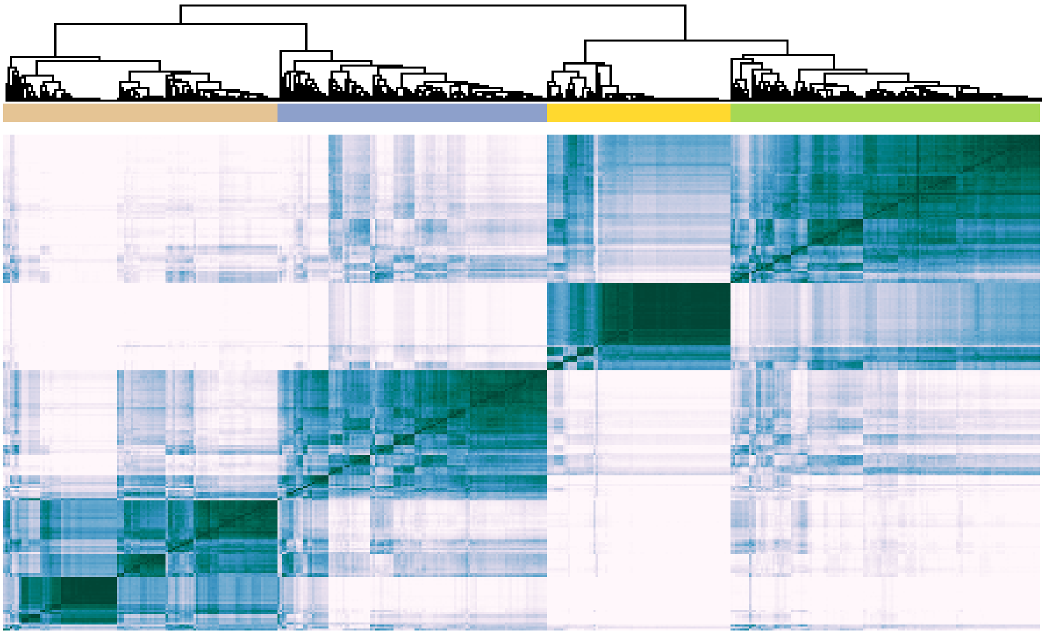


FIGURE F.7: Cluster ensemble heatmap for $k = 4$ with PAM

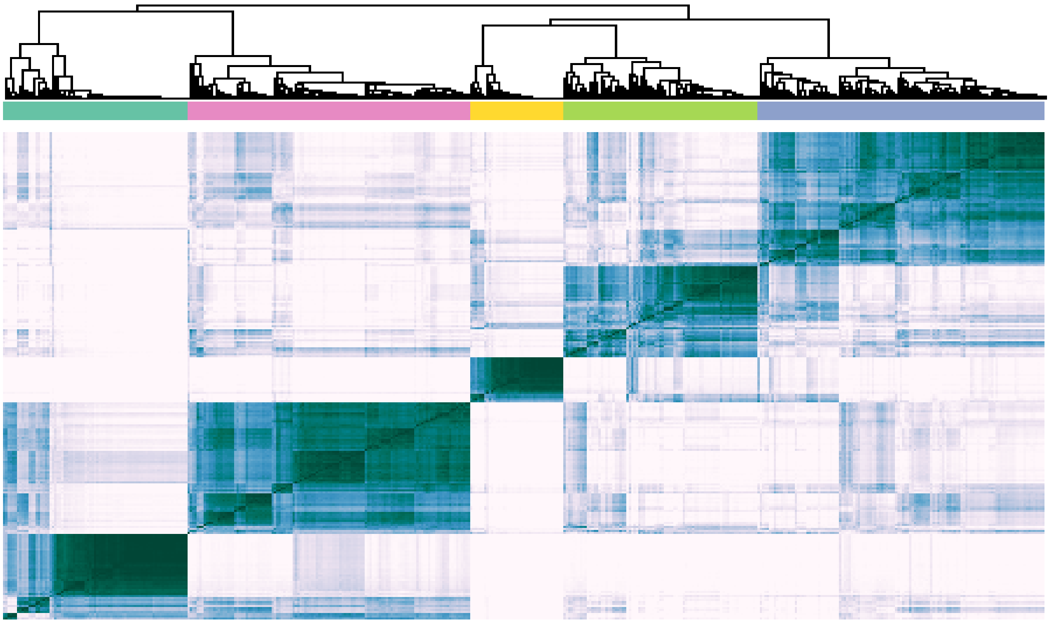


FIGURE F.8: Cluster ensemble heatmap for $k = 5$ with PAM

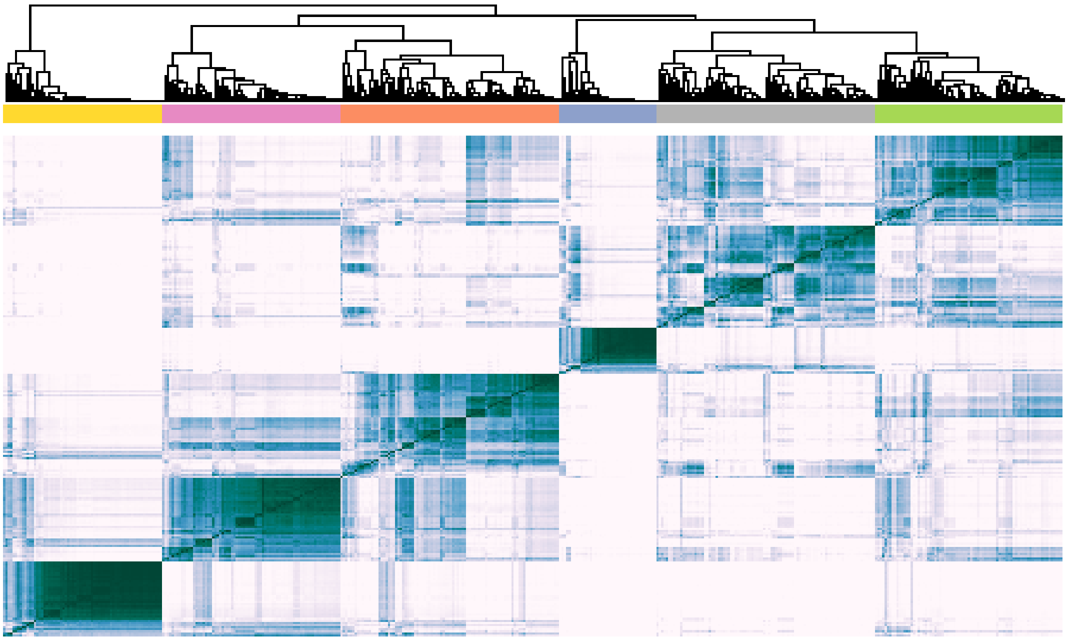


FIGURE F.9: Cluster ensemble heatmap for $k = 6$ with PAM

Appendix G

Conceptual paper for submission to Sage Publications, Health Informatics

Clustering severity of patients' symptoms in the schizophrenic and psychotic spectrum using the cluster ensemble

Vincent Menger, Wouter van der Klift and Marco Spruit

Abstract

Diagnosing patients mental disorders is based on symptoms in current society, causing high prevalence and comorbidity in the field of mental healthcare. Improving treatment has called for adopting machine learning. Researchers, however, face multiple challenges in applying machine learning. One of these problems is choosing an appropriate clustering algorithm and determining the amount of clusters. We show that the cluster ensemble can overcome this problem, since it does not require optimization of clustering algorithms and it finds clusters more accurately than a single algorithm. We experiment with the cluster ensemble on 744 DSM-IV diagnosed patients in the schizophrenic and psychotic spectrum that are assessed with the HoNOS. We found three clusters that indicate severity of symptoms among these patients. Two clusters are near equal size, 310 and 302 patients, that show low and mild symptom problems. The last cluster contains 132 patients that exhibit severe symptom problems, especially in their social environment.

Keywords

Cluster ensemble, DSM, HoNOS, Machine learning, Psychiatry

Introduction

General health issues, such as cancer, are remarkably well researched and diagnosed in today's society. Symptoms provide clues, while biomarkers, such as blood or genes, help draw conclusions. Thus, the combination of symptoms and biomarkers guides the clinical pathway to diagnosis, treatment and frequently to prognosis. In mental healthcare, however, biomarkers, which articulate the type of mental illness a person is suffering, are absent (Bzdok and Meyer-Lindenberg, 2017), since there is limited evidence regarding the mixture of functions of brains, organs, genetics, and the environmental settings of mental disorders (Frances and Widiger, 2012; Huys et al., 2016). Mental healthcare, therefore, continues to depend on a diagnostic approach solely based on symptoms (Kendell and Jablensky, 2003). Consequently, the field of mental healthcare faces high rates of co-morbidity and often ineffective treatment responses (Cuthbert and Insel, 2013; Wigman et al., 2017).

There are some detrimental effects to diagnosing solely on symptoms. A recent study by Boschloo et al. (2015) reveals that among the 12 major DSM (Diagnostic Statistical Manual for Mental Disorders) disorders, several symptoms share a common denominator. The authors call this factor "bridge-connections," which are symptoms shared among disorders. As such, symptoms from mental disorders are not latent conditions (Borsboom and Cramer, 2013; Fried et al., 2017; van Os et al., 2013). Therefore, the DSM is considered too rigid, leading to less specificity among mental disorders (Borsboom et al., 2011; Kendell and Jablensky, 2003; Krueger, 1999).

Data-driven techniques, such as machine learning, have emerged into the field of mental healthcare to address these shortcomings. Machine learning, in particular, can predict events or infer previously unknown structure from data, which are used to create new hypotheses (Iniesta et al., 2016). One of these machine learning techniques that received some interest in the field of mental healthcare is cluster analysis (Everitt et al., 2011). In previous studies it has been used to study the taxonomy of the DSM (Everitt et al., 1971; Gara et al., 1992; Paykel, 1971; Strauss et al., 1973), or to find previously unknown sub-types in some well-established mental disorders (Lochner et al., 2008; Prior et al., 1998; Ross et al., 2015). However, only a small selection of clustering algorithms have been used in prior mental healthcare studies (van Loo et al., 2012), meanwhile researchers keep facing difficulties in; applying machine learning correctly (Domingos, 2012), and selecting a suitable clustering algorithm with the right amount of clusters (Kuncheva et al., 2006).

Recently, the cluster ensemble is proposed and actively adopted in the field of biology (Fern and Lin, 2008). As in supervised ensemble learners, it allows to use a variety of different clustering algorithms to find structure within the data. It is considered more versatile and robust than a single optimized clustering algorithm (Fred and Jain, 2002; Strehl and Ghosh, 2002). In addition, it does not require to optimize a clustering algorithm and it determines the right size of clusters for a dataset (Kuncheva and Hadjitodorov, 2004), alleviating one of the problems researchers face when using cluster analysis. Nonetheless, it is not widely adopted in mental healthcare. As to date, only one study in mental healthcare showed interest in the cluster ensemble. Shen et al. (2007) used the cluster ensemble to identify sub-types of patients in the autism spectrum. However, more research is needed with the cluster ensemble in mental healthcare.

In this study we explore the cluster ensemble concept in a more rigorous way by consolidating it with the CRISP-DM (CRoss-Industry Standard Process for Data Mining) methodology from Chapman et al. (2000). In specific, the cluster ensemble and CRISP-DM are both translated to a *meta-algorithmic model* (MAM). By doing so, it alleviates the problem for researchers to follow machine learning steps and simultaneously the cluster analysis problem by using the cluster ensemble. The concept of MAM is adopted from Spruit and Jagesar (2016), which is a modeling technique designed for creating transparent models based on a process-deliverable diagram (van de Weerd and Brinkkemper, 2009). To this end, our study objective is to create multiple method fragments that adhere to the CRISP-DM cycle in which clustering is our main data analysis approach using the cluster ensemble as a machine learning technique. Second, the cluster ensemble is used on real psychiatric data that consist of 744 DSM-IV diagnosed patients suffering from schizophrenia or psychosis with a HoNOS (Health of Nation Outcome Scale) assessment. The data was used from the Psychiatry Department of the University Medical Center of Utrecht. In our understanding, this study is unique in its efforts, since patients from the DSM-IV with schizophrenia or psychosis who have been monitored with the HoNOS have not been used (to this point) in a cluster ensemble setting.

Background

Cluster analysis is used to discover groups within the data and is best described by Kaufman and Rousseeuw (1990): "*cluster analysis is the classification of objects into groups which share similarity among each other and impose a structure within the data, even if the structure is not directly present*." As such, cluster analysis proves useful for exploratory data analysis. Making it a favored technique in psychology to refine or redefine current diagnostic criteria (Everitt et al., 2011).

Jain and Dubes (1988) laid early groundwork in reviewing multiple types of clustering algorithms. The algorithms discussed in their review were hierarchical and partitioning methods. Since their introduction many clustering methods followed. It is argued that new clustering methods are developed if the existing ones do not suit the needs of the researcher (Fred and Jain, 2002). And so, after more than two decades, we find ourselves amidst more clustering algorithms. From classic hierarchical and partitioning, to grid-based, and probabilistic clustering methods (i.e. see Zhou (2012) for detailed list of available clustering algorithms). Undoubtedly contributing to what is discussed earlier that researchers find themselves having difficulties in selecting a suitable algorithm.

Clustering is, however, a variant technique to use. Based on the type of algorithm and dataset, each algorithm will give a different result. Even if there is no structure within the data, a clustering algorithm often will find clusters (Tan et al., 2005). Thus, the definition of what makes up a "good" cluster is frequently violated through scale-invariance, richness and consistency that affect the result (Kleinberg, 2002). Despite using internal validity indexes we are still not sure whether the true nature of clusters is found. This is supported in early ground work by Milligan and Cooper (1985), they already advocated discrepancy between multiple validity indexes. Similar results were obtained in a later, more profound, study by Arbelaitz et al. (2013). This paves the way to think about validity in clustering outcomes, since we cannot tell exactly if the result obtained is indisputable. Especially when a priori knowledge of data is absent, which is more likely as data collection keeps increasing. So, to improve validation of clustering results the cluster ensemble is advocated. Its approach is intuitive; find clusters with a diversity of algorithms, additionally with data extraction techniques, such as bootstrapping or feature sampling. Then, provide the result if multiple cluster partitions agreed upon similar clusters. Undoubtedly being more effective than a single algorithmic approach (Topchy et al., 2005).

The cluster ensemble spans two stages, namely; the generation stage and, the consensus solution. The first stage defines the creation of various cluster partitions. Kuncheva and Hadjitodorov (2004) point out that diversity in the generation stage is key in obtaining strong results. In basis, the result is not dependent on optimized algorithms, but rather on many weak partitions created by multiple, non-optimized, algorithms (Hadjitodorov et al., 2006; Topchy et al., 2003). The second stage defines the solving process of obtaining consensus amongst the partitions from the previous stage. This often goes by a winner-takes-all fashion as in Majority Voting (Fred and Jain, 2002; Ghosh and Acharya, 2011), or hypergraph methods proposed by Strehl and Ghosh (2002). In layman terms, observations assigned to the same cluster over many partitions will belong to that specific cluster in the consensus result. Both methods described belong to the object co-occurrence scheme known in Vega-Pons and Ruiz-Shulcloper (2011).

CRISP-DM

Since we integrated the cluster ensemble process in the CRISP-DM, brief explanation is needed. The CRISP-DM is a cycle consisting of six phases, individually known as:

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Data Modeling
5. Evaluation
6. Deployment

Each stage is sequential and defines different steps necessary to adhere to data mining efforts. For example, phases one and six are used to manage and control project needs. Meanwhile the other remaining phases involves data engineering efforts. For in-depth details of each phase, refer to Chapman et al., (2000).

In contrast to other data mining methods, such as SEMMA (Matignon, 2007), CRISP-DM is generic to both industry and technology used (Wirth, 2000). Depending on the data mining context, each domain can tailor the CRISP-DM model to their own specific needs (Niakšu, 2015). Such examples are found in Menger et al., (2016) and in Niakšu (2015). Last, CRISP-DM is considered the cornerstone for development of meta-algorithmic models to perform data mining for non-tech savvy experts (Spruit and Jagesar, 2016; Spruit and Lytras, 2018). Therefore, the CRISP-DM is also used for development of the cluster ensemble meta-algorithmic model.

Data and methods

744 Patients were used in this study that have been diagnosed with either schizophrenia or psychosis using the DSM-IV. For each patient we took their HoNOS scores. As such, patients could occur more than once in our dataset if their HoNOS was assessed more than once during their time of hospitalization.

The HoNOS consists of 12 questions, divided over four sub-categories, which we refer to as HoNOS *traits*:

1. Behavior
2. Symptoms
3. Impairment
4. Social

(*Extended information about the HoNOS questions, refer to* Pirkis et al. (2005).

Thus, for each patient we have used 12 features for our clustering process.

Our cluster ensemble used three algorithms, namely: K-means, PAM and C-means. Both K-means and PAM are partitioning methods (Jain, 2010). K-means uses centroids, often the cluster averages to form clusters (Jain et al., 1999). Somewhat similar is PAM, which uses actual observations as its centroids. Last, C-means is a probabilistic type and allows overlap of observations between clusters, hence it is known as a soft clustering approach (Mun et al., 2008).

The cluster scope is set at $k = 3$ to $k = 6$, using 500 iterations for each algorithm, with a bootstrap sample of 80%. We argue that starting with an initial three cluster structure removes the possibility for the cluster ensemble to create one big nested cluster that consists of two near equal sized clusters. The algorithm, therefore, is forced to split clusters that are close to each other.

Poor partitions were removed and the algorithms were reweighed using the best scores from three internal validity indexes; Silhouette, Davies-Bouldin and Calinski-Harabasz. Both Silhouette and Calinski-Harabasz maximize when inter-cluster and intra-cluster differences are maximized. The Davies-Bouldin index minimizes. This resulted in removal of PAM partitions, since these partitions scored low on indexes. Removing poor performing algorithms from the ensemble improves outcome in the consensus stage (Topchy et al., 2005).

The consensus stage used Majority Voting as the consensus solution. Majority Voting counts times that each observation is clustered to the same cluster over many partitions. In this winner-takes-all fashion the observation is clustered to the cluster with its majority of the vote.

Using Multi-Dimensional Scaling (MDS) we transformed the dataset to two-dimensional space and visualized the clusters, see Figure 1. Cluster formation was gathered from the cluster ensemble output from $k = 3$, as this was the best solution for our dataset according to the cluster ensemble. The triangles are Cluster 1, C^1, circles are Cluster 2, C^2, and plus-signs are Cluster 3, C^3.

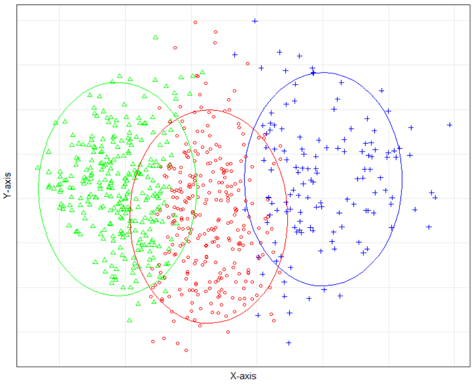


Figure 1: Clusters visualized with MDS

The HoNOS is a Likert-scale questionnaire, with scores ranging from 0 to 4 (0 indicating no problem and 4 indicating severe to very severe problem). We used these scores to calculate the mean between each cluster and created a radar plot to see how clusters deviate from each other. This type of plotting is also used in a study by Henry et al. (2005). Figure 2 displays the result obtained from the radar plot. Each label indicates one of the HoNOS features. The axis shows the mean score for each cluster on each feature. The solid line is C^3, the dotted line is C^1, and the striped line is C^2.

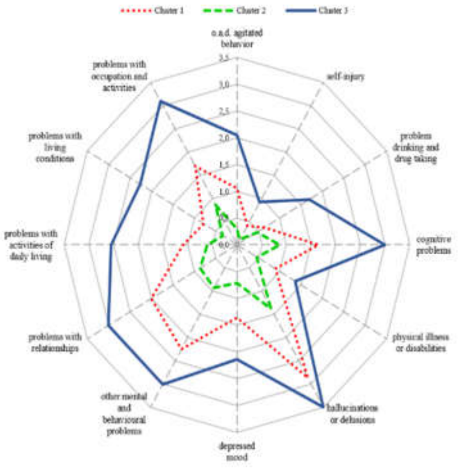


Figure 2: Radar plot of cluster features

Determining which feature actually contributes in defining each cluster, we again used the original HoNOS Likert-scale scores, but now we take the median scores per cluster. This method reveals which feature contributes to the formation of the cluster, since it shows which patient groups are represented in each feature.

Last, we consulted three experts in the field of psychiatry by interviewing them. Two experts had an interview time of 30 minutes, the other one only 10 minutes with e-mail correspondence afterward.

Data preprocessing

For our dataset we processed the original HoNOS features with rank normalization before the clustering process. Rank normalization is a feature scaling technique that minimizes variance between features, making it near equal in importance and best preserves distances between observations (Milligan and Cooper, 1988). Moreover, since the dataset is original of type Likert-scale, dissimilarity metrics are not suited. However, to determine cluster formation in partitioning methods, such as K-means and PAM, dissimilarity is needed. Rank normalization allows to transpose discrete features to ratio features, satisfying triangle inequality. As such, partitioning algorithms are suited for our dataset.

Data insight

Our initial patient class consisted of 21 disorders in the schizophrenic and psychosis spectrum. Condensation in classes was required to maintain overview. In our dataset paranoid schizophrenic and psychotic disorder NOS (Not Otherwise Specified) were the two biggest patient groups. Male to female ratio is almost 2 to 1, 65% is male and 35% is female. Average age of males lies at 31.6 years, for females this is 38.3 years. The age range in the dataset spans from 16 to 79.

Result

From the cluster ensemble three unique clusters were found in the data. These clusters, visualized in Figure 1, represent a “low problematic” C^2, “mild” C^1, and “severe” C^3, structure. It was apparent that both C^2 and C^1 clusters only differentiated on one item in the behavioral trait from the HoNOS. C^1 showed patients that exhibit problems in the overactive, aggressive, disruptive or agitated behavior feature. In addition, C^3 showed patients that exhibit severe problems in drug and alcohol taking feature, relationships feature and living conditions feature. For C^2 no feature was found that deviates itself from the other two clusters. We argue that missing such feature is an indicator for that specific cluster.

Conducting interviews with experts resulted in the following perspective. C^3 shows, according to the experts, correlation between patients' negative and positive symptoms of the disorder. Problems with living conditions and relationships follows from increased severity of the problems patients encounter during their hospitalization. Take C^1 or C^2, both do not exhibit increased problems in their social trait of the HoNOS, because their problems in the negative and positive symptoms is lower. Moreover, there is not a strong deviating pattern noticeable between the C^1 or C^2. As such, differences between these two clusters are more subtle than it is for C^3.

Conclusion

The results collected from this experiment reveal that there are three clusters in our dataset. We categorized these clusters as "severe," "mild," and "low problematic," in which the first cluster consists of patients who share some features that affect their social life and display some behavioral problems. Further evaluation with experts illustrated that there is some correlation between features in this cluster. For the other two clusters identified, it remains unclear which features exactly describe and differentiate them from each other, since we found no compelling evidence that the features deviate enough. As such, the experts were unable to find any correlation between the features nor differentiate them clearly.

Discussion

Although no clear separating features were found for the "mild" and "low problematic" clusters, it should be noted that some limitations may affected the end result. One of these limitations is the removal of observations with missing values. This resulted in a loss of 40% of the original size – from 1,277 to 744 observations after removal. Imputation by kNN is one approach to overcome the problem of missing values.

Second, the pattern between each cluster may indicate the presence of a cofound variable. Although we cannot circumvent this problem directly, since we are dependent on the data collecting manners of the Psychiatry department, assigning weights to common variables, such as hallucinations or delusions, might give another pattern.

Last, the PANNS (Positive and Negative Syndrome Scale) is more sophisticated for schizophrenic and psychotic patients than the HoNOS. Thus, future work might focus on this questionnaire to reveal clusters of severity among patients.

References

Arbelaitz O, Gurrutxaga I, Muguerza J, et al. (2013) An extensive comparative study of cluster validity indices. *Pattern Recognition* 46(1): 243–256.

Borsboom D and Cramer AOJ (2013) Network Analysis: An Integrative Approach to the Structure of Psychopathology. *Annual Review of Clinical Psychology* 9(1): 91–121.

Borsboom D, Cramer AOJ, Schmittmann VD, et al. (2011) The Small World of Psychopathology. *PLoS ONE* 6(11).

Boschloo L, Van Borkulo CD, Rhemtulla M, et al. (2015) The network structure of symptoms of the diagnostic and statistical manual of mental disorders. *PLoS ONE* 10(9): 1–12.

Bzdok D and Meyer-Lindenberg A (2017) Machine learning for precision psychiatry. *arXiv preprint*: 1–16.

Chapman P, Clinton J, Kerber R, et al. (2000) CRISP-DM 1.0 Step-by-step data mining guide.

Cuthbert BN and Insel TR (2013) Toward the future of psychiatric diagnosis: the seven pillars of RDoC. *BMC Medicine* 11(1): 1–8.

Domingos P (2012) A few useful things to know about machine learning. *Communications of the ACM* 55(10): 78.

Everitt BS, Gourlay AJ and Kendell RE (1971) An attempt at validation of traditional psychiatric syndromes by cluster analysis. *British Journal of Psychiatry* 119(551): 399–412.

Everitt BS, Landau S, Leese M, et al. (2011) *Cluster Analysis*. 5th ed. Wiley Series in Probability and Statistics. Chichester, UK: John Wiley & Sons, Ltd.

Fern XZ and Lin W (2008) Cluster Ensemble Selection. *Statistical Analysis and Data Mining* 1(3): 128–141.

Frances AJ and Widiger T (2012) Psychiatric Diagnosis: Lessons from the DSM-IV Past and Cautions for the DSM-5 Future. *Annual Review of Clinical Psychology* 8(1): 109–130.

Fred ALN and Jain AK (2002) Data clustering using evidence accumulation. In: *Object recognition supported by user interaction for service robots*, 2002, pp. 276–280. IEEE Comput. Soc.

Fried EI, van Borkulo CD, Cramer AOJ, et al. (2017) Mental disorders as networks of problems: a review of recent insights. *Social Psychiatry and Psychiatric Epidemiology* 52(1): 1–10.

Gara MA, Rosenberg S and Goldberg L (1992) A Cluster Analysis of Diagnosis and Symptoms. *The Journal of nervous and mental disease* 180(1): 11–19.

Ghosh J and Acharya A (2011) Cluster ensembles. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1(4): 305–315.

Hadjitodorov ST, Kuncheva LI and Todorova LP (2006) Moderate diversity for better cluster ensembles. *Information Fusion* 7(3): 264–275.

Henry DB, Tolan PH and Gorman-Smith D (2005) Cluster analysis in family psychology research. *Journal of Family Psychology* 19(1): 121–132.

Huys QJM, Maia T V and Frank MJ (2016) Computational psychiatry as a bridge from neuroscience to clinical applications. *Nature Neuroscience* 19(3): 404–413.

Iniesta R, Stahl D and McGuffin P (2016) Machine learning, statistical learning and the future of biological research in psychiatry. *Psychological Medicine* 46(12): 2455–2465.

Jain AK (2010) Data clustering: 50 years beyond K-means. *Pattern Recognition Letters* 31(8): 651–666.

Jain AK and Dubes RC (1988) Clustering Methods and Algorithms. In: *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey: Prentice Hall, pp. 55–142.

Jain AK, Murty MN and Flynn PJ (1999)

Data clustering: a review. *ACM Computing Surveys* 31(3): 264–323.

Kaufman L and Rousseeuw PJ (1990) *Finding Groups in Data: An introduction to Cluster Analysis*. Kaufman L and Rousseeuw PJ (eds). Wiley Series in Probability and Statistics. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Kendell R and Jablensky A (2003) Distinguishing Between the Validity and Utility of Psychiatric Diagnoses. *American Journal of Psychiatry* 160(1): 4–12.

Kleinberg J (2002) An impossibility theorem for clustering. *Advances in Neural Information Processing Systems*: 446–453.

Krueger R (1999) The structure of common mental disorders. *Archives of General Psychiatry* 56: 921–926.

Kuncheva LI and Hadjitodorov ST (2004) Using diversity in cluster ensembles. In: *IEEE International Conference on Systems, Man and Cybernetics*, 2004, pp. 1214–1219. IEEE.

Kuncheva LI, Hadjitodorov ST and Todorova LP (2006) Experimental Comparison of Cluster Ensemble Methods. In: *2006 9th International Conference on Information Fusion*, July 2006, pp. 1–7. IEEE.

Lochner C, Hemmings SMJ, Kinnear CJ, et al. (2008) Cluster Analysis of Obsessive-Compulsive Symptomatology: Identifying Obsessive-Compulsive Disorder Subtypes. *The Israel Journal of Psychiatry and Related Sciences* 45(3): 164–176.

Matignon R (2007) *Data mining using SAS enterprise miner*. John Wiley & Sons.

Menger V, Spruit M, Hagoort K, et al. (2016) Transitioning to a Data Driven Mental Health Practice: Collaborative Expert Sessions for Knowledge and Hypothesis Finding. *Computational and Mathematical Methods in Medicine* 2016: 1–11.

Milligan GW and Cooper MC (1985) An examination of procedures for determining the number of clusters in a data set. *Psychometrika* 50(2). Springer: 159–179.

Milligan GW and Cooper MC (1988) A study of standardization of variables in cluster analysis. *Journal of Classification* 5(2): 181–204.

Mun EY, von Eye A, Bates ME, et al. (2008) Finding Groups Using Model-Based Cluster Analysis: Heterogeneous Emotional Self-Regulatory Processes and Heavy Alcohol Use Risk. *Developmental Psychology* 44(2): 481–495.

Niakšu O (2015) CRISP Data Mining Methodology Extension for Medical Domain. *Baltic J. Modern Computing* 3(2): 92–109.

Paykel ES (1971) Classification of Depressed Patients: A Cluster Analysis Derived Grouping. *The British Journal of Psychiatry* 118(544): 275–288.

Pirkis JE, Burgess PM, Kirk PK, et al. (2005) A review of the psychometric properties of the Health of the Nation Outcome Scales (HoNOS) family of measures. *Health and Quality of Life Outcomes* 3: 1–12.

Prior M, Leekam S, Ong B, et al. (1998) Are There Subgroups within the Autistic Spectrum? A Cluster Analysis of a Group of Children with Autistic Spectrum Disorders. *Journal of Child Psychology and Psychiatry* 39(6): 893–902.

Ross J, Neylan T, Weiner M, et al. (2015) Towards Constructing a New Taxonomy for Psychiatry Using

Self-reported Symptoms. *Studies in Health Technology and Informatics* 216: 736–740.

Shen JJ, Lee PH, Holden JJA, et al. (2007) Using Cluster Ensemble and Validation to Identify Subtypes of Pervasive Developmental Disorders. *AMAI Annual Symposium Proceedings*: 666–670.

Spruit M and Jagesar R (2016) Power to the People! - Meta-Algorithmic Modelling in Applied Data Science. *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* 1: 400–406.

Spruit M and Lytras M (2018) Applied Data Science in Patient-centric Healthcare. *Telematics and Informatics* 35(April): 643–653.

Strauss JS, Bartko JJ and Carpenter WT (1973) The Use of Clustering Techniques for the Classification of Psychatric Patients. *British Journal of Psychiatry* 122: 351–540.

Strehl A and Ghosh J (2002) Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research* 3: 583–617.

Tan P-N, Steinbach M and Kumar V (2005) Cluster Analysis: Basic Concepts and Algorithms. In: *Introduction to Data Mining*. 1st ed. Pearson, pp. 487–555.

Topchy A, Jain AK and Punch W (2003) Combining multiple weak clusterings. In: *Third IEEE International Conference on Data Mining*, 2003, pp. 331–338. IEEE Comput. Soc.

Topchy A, Jain AK and Punch W (2005) Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(12): 1866–1881.

van de Weerd I and Brinkkemper S (2009) Meta-Modeling for Situational Analysis and Design Methods. In: *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications*. IGI Global, pp. 35–54.

van Loo HM, de Jonge P, Romeijn J-W, et al. (2012) Data-driven subtypes of major depressive disorder: A systematic review. *BMC Medicine* 10: no pagination.

van Os J, Delespaul P, Wigman J, et al. (2013) Beyond DSM and ICD: introducing “precision diagnosis” for psychiatry using momentary assessment technology. *World Psychiatry* 12(2): 113–117.

Vega-Pons S and Ruiz-Shulcloper J (2011) A SURVEY OF CLUSTERING ENSEMBLE ALGORITHMS. *International Journal of Pattern Recognition and Artificial Intelligence* 25(3): 337–372.

Wigman JTW, De Vos S, Wichers M, et al. (2017) A transdiagnostic network approach to psychosis. *Schizophrenia Bulletin* 43(1): 122–132.

Wirth R (2000) CRISP-DM : Towards a Standard Process Model for Data Mining. *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining* (24959): 29–39.

Zhou ZH (2012) Clustering Ensembles. In: Herbrich R and Graepel T (eds) *Ensemble methods: foundations and algorithms*. Boca Raton: Taylor & Francis Group, pp. 135–156.

Bibliography

Achlioptas, Dimitris (2001). "Database-friendly random projections". In: *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems - PODS '01*. New York, New York, USA: ACM Press, pp. 274–281. ISBN: 1581133618.

Aeberhard, Stefan (1991). *Wine Data Set*.

Aggarwal, Charu C., Alexander Hinneburg, and Daniel A. Keim (2001). "On the Surprising Behavior of Distance Metrics in High Dimensional Space". In: pp. 420–434.

Agrawal, Rakesh et al. (1998). "Automatic subspace clustering of high dimensional data for data mining applications". In: *ACM SIGMOD Record* 27.2, pp. 94–105.

American Psychiatric Association (2013). *Diagnostic and statistical manual of mental disorders (DSM-5®)*. American Psychiatric Pub.

Arbelaitz, Olatz et al. (2013). "An extensive comparative study of cluster validity indices". In: *Pattern Recognition* 46.1, pp. 243–256.

Ayad, Hanan and Mohamed S Kamel (2008). "Cumulative Voting Consensus Method for Partitions with Variable Number of Clusters". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.1, pp. 160–173.

Azevedo, Ana and Manuel Filipe Santos (2008). "KDD, SEMMA and CRISP-DM: a parallel overview". In: *IADIS European Conference Data Mining* January, pp. 182–185.

Baskerville, Richard (2008). "What design science is not". In: *European Journal of Information Systems* 17.5, pp. 441–443.

Behrens, John T. and Chong-ho Yu (2012). "Exploratory Data Analysis". In: *Handbook of Psychology: Research Methods in Psychology*. Ed. by Irving Weiner. Volume 2. Hoboken, NJ, USA: John Wiley & Sons, Inc. Chap. 2, p. 32. ISBN: 9780470619049.

Ben-Hur, Asa, Andre Elisseeff, and Isabelle Guyon (2001). "A stability based method for discovering structure in clustered data". In: *Biocomputing 2002*. Vol. 17, pp. 6–17. ISBN: 978-981-02-4777-5.

Bingham, Ella and Heikki Mannila (2001). "Random projection in dimensionality reduction". In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, New York, USA: ACM Press, pp. 245–250. ISBN: 158113391X.

Borsboom, Denny and Angélique O.J. Cramer (2013). "Network Analysis: An Integrative Approach to the Structure of Psychopathology". In: *Annual Review of Clinical Psychology* 9.1, pp. 91–121.

Borsboom, Denny et al. (2011). "The Small World of Psychopathology". In: *PLoS ONE* 6.11.

Boschloo, Lynn et al. (2015). "The network structure of symptoms of the diagnostic and statistical manual of mental disorders". In: *PLoS ONE* 10.9, pp. 1–12.

Brinkemper, Sjaak (1996). "Method engineering: engineering of information methods and tools". In: *Information and Software Technology* 38, pp. 275–280.

Bzdok, Danilo and Andreas Meyer-Lindenberg (2017). "Machine learning for precision psychiatry". In: *arXiv preprint*, pp. 1–16.

Caliñski, T. and J. Harabasz (1974). "A Dendrite Method Foe Cluster Analysis". In: *Communications in Statistics* 3.1, pp. 1–27.

Chapman, Pete et al. (2000). *CRISP-DM 1.0 Step-by-step data mining guide*.

Charytanowicz, Małgorzata et al. (2010). "Complete Gradient Clustering Algorithm for Features Analysis of X-Ray Images". In: *Information Technologies in Biomedicine* 69, pp. 15–24.

Chekroud, Adam Mourad et al. (2016). "Cross-trial prediction of treatment outcome in depression : a machine learning approach". In: *The Lancet Psychiatry* 3.3, pp. 243–250.

Cheng, Wei, Wei Wang, and Sandra Batista (2013). "Grid-based clustering". In: *Data Clustering*. Chapman and Hall/CRC, pp. 128–148.

Chiu, Derek and Aline Talhouk (2018). *Diverse Cluster Ensemble in R*.

Cuthbert, Bruce N and Thomas R Insel (2013). “Toward the future of psychiatric diagnosis: the seven pillars of RDoC”. In: *BMC Medicine* 11.1, pp. 1–8.

Dasgupta, Sanjoy (2000). “Experiments with Random Projection”. In: *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. UAI’00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 143–151. ISBN: 1-55860-709-9.

Dash, M. et al. (2002). “Feature selection for clustering - a filter solution”. In: *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* April 2014. IEEE Comput. Soc, pp. 115–122. ISBN: 0-7695-1754-4.

Dash, Manoranjan and Huan Liu (2000). “Feature Selection for Clustering”. In: *Knowledge Discovery and Data Mining: Current Issues and New Applications*, pp. 110–121. ISBN: 9783540858331.

Davies, David L. and Donald W. Bouldin (1979). “A Cluster Separation Measure”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2, pp. 224–227.

Dhillon, Inderjit S (2001). “Co-clustering documents and words using bipartite spectral graph partitioning”. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 269–274.

Domingos, Pedro (2012). “A few useful things to know about machine learning”. In: *Communications of the ACM* 55.10, p. 78.

Dy, Jennifer G and Carla E Brodley (2000). “Feature Subset Selection and Order Identification for Unsupervised Learning”. In: *ICML ’00 Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 247–254.

Efron, Bradley (1992). “Bootstrap methods: another look at the jackknife”. In: *Breakthroughs in statistics*. Springer, pp. 569–593.

Ester, Martin et al. (1996). “A density-based algorithm for discovering clusters in large spatial databases with noise.” In: *Kdd*. Vol. 96. 34, pp. 226–231.

Everitt, B. S., A. J. Gourlay, and R. E. Kendell (1971). “An attempt at validation of traditional psychiatric syndromes by cluster analysis.” In: *British Journal of Psychiatry* 119.551, pp. 399–412.

Everitt, Brian S. et al. (2011). *Cluster Analysis*. 5th ed. Wiley Series in Probability and Statistics. Chichester, UK: John Wiley & Sons, Ltd, p. 321. ISBN: 9780470977811.

Fahad, Adil et al. (2014). “A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis”. In: *IEEE Transactions on Emerging Topics in Computing* 2.3, pp. 267–279.

Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth (1996). “From Data Mining to Knowledge Discovery in Databases”. In: *AI Magazine* 17.3, p. 37.

Fern, Xiaoli Zhang and Carla E Brodley (2004). “Solving cluster ensemble problems by bipartite graph partitioning”. In: *Twenty-first international conference on Machine learning - ICML ’04*. New York, New York, USA: ACM Press, p. 36. ISBN: 1581138285.

Fern, Xiaoli Zhang and Wei Lin (2008). “Cluster Ensemble Selection”. In: *Statistical Analysis and Data Mining* 1.3, pp. 128–141.

Fern, X.Z. and C.E. Brodley (2003). “Random projection for high dimensional data clustering: A cluster ensemble approach”. In: *Proceedings of the Twentieth International Conference on Machine Learning* 20, pp. 186–193.

Fernandes, Brisa S. et al. (2017). “The new field of ‘precision psychiatry’”. In: *BMC Medicine* 15.1, p. 80.

Fisher, Judith L and Mary B Harris (1973). “Effect of Note Taking and Review on Recall”. In: *Journal of Educational Psychology* 66.3, pp. 321–325.

Fleiss, Joseph L and Joseph Zubin (1969). “On the methods and theory of clustering”. In: *Multivariate Behavioral Research* 4.2, pp. 235–250.

Frades, Itziar and Rune Matthiesen (2010). “Overview on Techniques in Cluster Analysis”. In: *Bioinformatics Methods in Clinical Research*. Ed. by Rune Matthiesen. Vol. 593. Methods in Molecular Biology. Totowa, NJ: Humana Press. Chap. 5, pp. 81–107. ISBN: 978-1-60327-193-6.

Fraley, Chris and Adrian E Raftery (2006). *MCLUST version 3: an R package for normal mixture modeling and model-based clustering*. Tech. rep. WASHINGTON UNIV SEATTLE DEPT OF STATISTICS.

Frances, Allen J. and Thomas Widiger (2012). “Psychiatric Diagnosis: Lessons from the DSM-IV Past and Cautions for the DSM-5 Future”. In: *Annual Review of Clinical Psychology* 8.1, pp. 109–130.

Fred, Ana and Anil K. Jain (2002). "Evidence Accumulation Clustering Based on the K-Means Algorithm". In: *Structural, Syntactic, and Statistical Pattern Recognition. Lecture Notes in Computer Science*, pp. 442–451. ISBN: 978-3-540-44011-6.

Fred, Ana and Andre Lourenco (2008). "Cluster Ensemble Methods: from Single Clusterings to Combined Solutions". In: *Supervised and Unsupervised Ensemble methods and their Application*. Ed. by Oleg Okun and Giorgio Valentini. Berlin, Heidelberg: Springer-Verlag. Chap. 1, pp. 3–30. ISBN: 9783540789802.

Fried, Eiko I. et al. (2016). "What are 'good' depression symptoms? Comparing the centrality of DSM and non-DSM symptoms of depression in a network analysis". In: *Journal of Affective Disorders* 189, pp. 314–320.

Fukunaga, K (1990). *Introduction to statistical pattern classification*.

Gan, Guojun, Chaoqun Ma, and Jianhong Wu (2007). "12. Grid-Based Clustering Algorithms". In: *Data Clustering: Theory, Algorithms, and Applications*. Society for Industrial and Applied Mathematics, pp. 209–217. ISBN: 978-0-89871-623-8.

Gan, Yiqun et al. (2013). "A note on relevance of diagnostic classification and rating scales used in psychiatry". In: *Computer Methods and Programs in Biomedicine* 112.1, pp. 16–21.

Gara, M.A., S. Rosenberg, and L. Goldberg (1992). "A Cluster Analysis of Diagnosis and Symptoms". In: *The Journal of nervous and mental disease* 180.1, pp. 11–19.

Gelman, Andrew (2004). "Exploratory data analysis for complex models". In: *Journal of Computational and Graphical Statistics* 13.4, pp. 755–779.

Ghaemi, R. et al. (2009). "A Survey: clustering ensembles techniques". In: *World Academy of Science, Engineering and Technology* 50, pp. 636–645.

Ghosh, Joydeep and Ayan Acharya (2011). "Cluster ensembles". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1.4, pp. 305–315.

Gionis, A., H. Mannila, and P. Tsaparas (2007). "Clustering aggregation". In: *ACM Transactions on Knowledge Discovery from Data* 1.1, pp. 1–30.

Goekoop, R. and J. G. Goekoop (2016). "Netwerkclusters van Symptomen als elementaire Syndromen in de psychopathologie: Consequenties voor de klinische praktijk". In: *Tijdschrift voor Psychiatrie* 58.1, pp. 38–47.

Gower, John C (1971). "A general coefficient of similarity and some of its properties". In: *Biometrics*, pp. 857–871.

Greene, D. et al. (2004). "Ensemble clustering in medical diagnostics". In: *Proceedings. 17th IEEE Symposium on Computer-Based Medical Systems*. June. IEEE Comput. Soc, pp. 576–581. ISBN: 0-7695-2104-5.

Hadjitodorov, Stefan T. and Ludmila I. Kuncheva (2007). "Selecting Diversifying Heuristics for Cluster Ensembles". In: *Multiple Classifier Systems*. Vol. 4472. May 2014. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 200–209. ISBN: 978-3-540-72481-0.

Hadjitodorov, Stefan T., Ludmila I. Kuncheva, and Ludmila P. Todorova (2006). "Moderate diversity for better cluster ensembles". In: *Information Fusion* 7.3, pp. 264–275.

Halkidi, Maria, Yannis Batistakis, and Michalis Vazirgiannis (2002). "Cluster validity methods". In: *ACM SIGMOD Record* 31.2, p. 40.

Han, Jin H et al. (2017). "Exploring Delirium's Heterogeneity: Association Between Arousal Subtypes at Initial Presentation and 6-Month Mortality in Older Emergency Department Patients". In: *The American Journal of Geriatric Psychiatry* 25, pp. 233–242.

Henry, David B., Patrick H. Tolan, and Deborah Gorman-Smith (2005). "Cluster analysis in family psychology research". In: *Journal of Family Psychology* 19.1, pp. 121–132.

Hettige, Nuwan C. et al. (2017). "Classification of suicide attempters in schizophrenia using sociocultural and clinical features: A machine learning approach". In: *General Hospital Psychiatry* 47, pp. 20–28.

Hevner et al. (2004). "Design Science in Information Systems Research". In: *MIS Quarterly* 28.1, p. 31.

Hinneburg, Alexander and Daniel a Keim (1999). "Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering". In: *International Conference on Very Large Databases (VLDB)*, pp. 506–517.

Hristoskova, Anna, Veselka Boeva, and Elena Tsiporkova (2014). "A formal concept analysis approach to consensus clustering of multi-experiment expression data". In: *BMC Bioinformatics* 15.1, p. 151.

Huan Liu and Lei Yu (2005). “Toward integrating feature selection algorithms for classification and clustering”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.4, pp. 491–502.

Huys, Quentin J M, Tiago V Maia, and Michael J Frank (2016). “Computational psychiatry as a bridge from neuroscience to clinical applications”. In: *Nature Neuroscience* 19.3, pp. 404–413.

Hyman, Steven E. (2010). “The Diagnosis of Mental Disorders: The Problem of Reification”. In: *Annual Review of Clinical Psychology* 6.1, pp. 155–179.

Iam-On, N et al. (2010). “A Link-Based Cluster Ensemble Approach for Categorical Data Clustering”. In: *IEEE Transactions on Knowledge & Data Engineering* 24, pp. 413–425.

Iniesta, R, D Stahl, and P. McGuffin (2016). “Machine learning, statistical learning and the future of biological research in psychiatry”. In: *Psychological Medicine* 46.12, pp. 2455–2465.

Jain, A. K., M N Murty, and P J Flynn (1999). “Data clustering: a review”. In: *ACM Computing Surveys* 31.3, pp. 264–323.

Jain, Anil K. (2010). “Data clustering: 50 years beyond K-means”. In: *Pattern Recognition Letters* 31.8, pp. 651–666. arXiv: `0402594v3 [cond-mat]`.

Jain, Anil K. and Richard C. Dubes (1988). “Clustering Methods and Algorithms”. In: *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey: Prentice Hall. Chap. 3, pp. 55–142. ISBN: 013022278X.

Jain, Anil K and Martin H C Law (2005). “Data Clustering: A User’s Dilemma”. In: *International Conference on Pattern Recognition and Machine Intelligence*, pp. 1–10.

James, Gareth et al. (2013). *An Introduction to Statistical Learning*. Vol. 103. Springer Texts in Statistics. New York, NY: Springer New York, pp. 127–173. ISBN: 978-1-4614-7137-0.

Kaufman, Leonard and Peter J Rousseeuw (1990). *Finding Groups in Data: An introduction to Cluster Analysis*. Ed. by Leonard Kaufman and Peter J. Rousseeuw. Wiley Series in Probability and Statistics. Hoboken, NJ, USA: John Wiley & Sons, Inc. ISBN: 9780470316801.

Kendell, Robert and Assen Jablensky (2003). “Distinguishing Between the Validity and Utility of Psychiatric Diagnoses”. In: *American Journal of Psychiatry* 160.1, pp. 4–12.

Kendler, K. S., P. Zachar, and C. Craver (2011). “What kinds of things are psychiatric disorders?” In: *Psychological Medicine* 41.06, pp. 1143–1150.

Kennedy, James A (2008). *Mastering the Kennedy Axis V: A new psychiatric assessment of patient functioning*. American Psychiatric Pub.

Kessler, Ronald C et al. (2016). “Testing a machine-learning algorithm to predict the persistence and severity of major depressive disorder from baseline self-reports”. In: *Molecular Psychiatry* 21.10, pp. 1366–1371.

Kittler, Josef et al. (1998). “On combining classifiers”. In: *IEEE transactions on pattern analysis and machine intelligence* 20.3, pp. 226–239.

Kleinberg, Jon (2002). “An impossibility theorem for clustering”. In: *Advances in Neural Information Processing Systems*, pp. 446–453. arXiv: `0607100v2 [physics]`.

Kovács, F., C. Legány, and A. Babos (2005). “Cluster validity measurement techniques”. In: *6th International symposium of hungarian researchers on computational intelligence* November.

Krueger, R (1999). “The structure of common mental disorders.” In: *Archives of General Psychiatry* 56, 921–926.

Kruskal, Joseph B (1964). “Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis”. In: *Psychometrika* 29.1, pp. 1–27.

Kuncheva, L.I. and S.T. Hadjitodorov (2004). “Using diversity in cluster ensembles”. In: *IEEE International Conference on Systems, Man and Cybernetics*. Vol. 2. IEEE, pp. 1214–1219. ISBN: 0-7803-8567-5.

Kuncheva, Ludmila I. and Dmitry P. Vetrov (2006). “Evaluation of stability of k-means cluster ensembles with respect to random initialization”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.11, pp. 1798–1808.

Langfelder, Peter, Bin Zhang, and Steve Horvath (2008). “Defining clusters from a hierarchical cluster tree: The Dynamic Tree Cut package for R”. In: *Bioinformatics* 24.5, pp. 719–720.

Law, M.H.C., M.A.T. Figueiredo, and A.K. Jain (2004). “Simultaneous feature selection and clustering using mixture models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.9, pp. 1154–1166.

Leventhal, Barry (2010). "An introduction to data mining and other techniques for advanced analytics". In: *Journal of Direct, Data and Digital Marketing Practice* 12.2, pp. 137–153.

Liao, Wei-keng, Ying Liu, and Alok Choudhary (2004). "A Grid-based Clustering Algorithm using Adaptive Mesh Refinement". In: *7th Workshop on Mining Scientific and Engineering Datasets of SIAM International Conference on Data Mining* 22, pp. 61–69.

Little, Roderick J. A. and Donald B. Rubin (1989). "The Analysis of Social Science Data with Missing Values". In: *Sociological Methods & Research* 18.2-3, pp. 292–326.

Liu, Yanchi et al. (2010). "Understanding of Internal Clustering Validation Measures". In: *2010 IEEE International Conference on Data Mining*. IEEE, pp. 911–916. ISBN: 978-1-4244-9131-5.

Lochner, Christine et al. (2008). "Cluster Analysis of Obsessive-Compulsive Symptomatology: Identifying Obsessive-Compulsive Disorder Subtypes". In: *The Israel Journal of Psychiatry and Related Sciences* 45.3, pp. 164–176.

Loo, Hanna M. van et al. (2014). "Major Depressive Disorder Subtypes To Predict Long-Term Course". In: *Depression and Anxiety* 31.9, pp. 765–777.

Loo, H.M. van et al. (2012). "Data-driven subtypes of major depressive disorder: A systematic review". In: *BMC Medicine* 10, no pagination.

Luo, Jake et al. (2016). "Big Data Application in Biomedical Research and Health Care: A Literature Review". In: *Biomedical Informatics Insights* 8.

Marquand, Andre F. et al. (2016). "Beyond Lumping and Splitting: A Review of Computational Approaches for Stratifying Psychiatric Disorders". In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging* 1.5, pp. 433–447.

Matignon, Randall (2007). *Data mining using SAS enterprise miner*. Vol. 638. John Wiley & Sons.

Maulik, U. and S. Bandyopadhyay (2002). "Performance evaluation of some clustering algorithms and validity indices". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.12, pp. 1650–1654.

Menger, Vincent et al. (2016). "Transitioning to a Data Driven Mental Health Practice: Collaborative Expert Sessions for Knowledge and Hypothesis Finding". In: *Computational and Mathematical Methods in Medicine* 2016, pp. 1–11.

Milligan, Glenn W and Martha C Cooper (1985). "An examination of procedures for determining the number of clusters in a data set". In: *Psychometrika* 50.2, pp. 159–179.

Milligan, Glenn W. and Martha C. Cooper (1988). "A study of standardization of variables in cluster analysis". In: *Journal of Classification* 5.2, pp. 181–204.

Minaei-Bidgoli, B, A Topchy, and W F Punch (2004). "A comparison of resampling methods for clustering ensembles". In: *Proceedings of the International Conference on Artificial Intelligence, IC-AI'04* 2, pp. 939–945.

Monti, S et al. (2003). "Consensus clustering: A resampling based method for class discovery and visualization of gene expression microarray data". In: *Machine Learning* 52.1, pp. 91–118.

Mun, Eun Young et al. (2008). "Finding Groups Using Model-Based Cluster Analysis: Heterogeneous Emotional Self-Regulatory Processes and Heavy Alcohol Use Risk". In: *Developmental Psychology* 44.2, pp. 481–495.

Murphy, Kevin P. (2012). *Machine Learning: a probabilistic perspective*. Cambridge, Massachusetts: MIT Press, p. 1050. ISBN: 9780262018029.

Murtagh, Fionn and Pedro Contreras (2012). "Algorithms for hierarchical clustering: An overview". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2.1, pp. 86–97.

Naldi, M. C., A. C.P.L.F. Carvalho, and R. J.G.B. Campello (2013). "Cluster ensemble selection based on relative validity indexes". In: *Data Mining and Knowledge Discovery* 27.2, pp. 259–289.

Nash, Warwick J et al. (1994). "The population biology of abalone (haliotis species) in Tasmania. I. Blacklip Abalone (h. rubra) from the north coast and islands of Bass Strait". In: *Sea Fisheries Division, Technical Report* 48.

Niakšu, Olegas (2015). "CRISP Data Mining Methodology Extension for Medical Domain". In: *Baltic J. Modern Computing* 3.2, pp. 92–109.

Os, Jim van et al. (2013). "Beyond DSM and ICD: introducing "precision diagnosis" for psychiatry using momentary assessment technology". In: *World Psychiatry* 12.2, pp. 113–117.

Osei-Bryson, Kweku Muata (2010). "Towards supporting expert evaluation of clustering results using a data mining process model". In: *Information Sciences* 180.3, pp. 414–431.

Parsons, Lance, Ehtesham Haque, and Huan Liu (2004). "Subspace clustering for high dimensional data". In: *ACM SIGKDD Explorations Newsletter* 6.1, pp. 90–105.

Passos, Ives Cavalcante et al. (2016). "Identifying a clinical signature of suicidality among patients with mood disorders: A pilot study using a machine learning approach". In: *Journal of Affective Disorders* 193, pp. 109–116.

Paykel, E. S. (1971). "Classification of Depressed Patients: A Cluster Analysis Derived Grouping". In: *The British Journal of Psychiatry* 118.544, pp. 275–288.

Peffers, Ken et al. (2007). "A Design Science Research Methodology for Information Systems Research". In: *Journal of Management Information Systems* 24.3, pp. 45–77.

Pirkis, Jane E. et al. (2005). "A review of the psychometric properties of the Health of the Nation Outcome Scales (HoNOS) family of measures". In: *Health and Quality of Life Outcomes* 3, pp. 1–12.

Priness, Ido, Oded Maimon, and Irad Ben-Gal (2007). "Evaluation of gene-expression clustering via mutual information distance measure". In: *BMC Bioinformatics* 8, pp. 1–12.

Prior, Margot et al. (1998). "Are There Subgroups within the Autistic Spectrum? A Cluster Analysis of a Group of Children with Autistic Spectrum Disorders". In: *Journal of Child Psychology and Psychiatry* 39.6, pp. 893–902.

Regier, Darrel A., Emily A. Kuhl, and David J. Kupfer (2013). "The DSM-5: Classfication and Criteria Changes". In: *World Psychiatry* 12.2, pp. 92–98.

Rokach, Lior and Oded Maimon (2005). "Clustering Methods". In: *Data Mining and Knowledge Discovery Handbook*. New York: Springer-Verlag. Chap. 15, pp. 321–352.

Ross, Jessica et al. (2015). "Towards Constructing a New Taxonomy for Psychiatry Using Self-reported Symptoms". In: *Studies in Health Technology and Informatics* 216, pp. 736–740.

Rousseeuw, Peter J. (1987). "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". In: *Journal of Computational and Applied Mathematics* 20.1, pp. 53–65.

RStudio, Team (2015). *RStudio: Integrated Development for R*. Boston, MA.

Sarstedt, Marko and Erik Mooi (2014). "Cluster Analysis". In: *the process, data, and methods using IBM SPSS statistics*. Berlin, Heidelberg: Springer Berlin Heidelberg. Chap. 9, pp. 273–324. ISBN: 978-3-642-12540-9.

Sheikholeslami, Gholamhosein, Surojit Chatterjee, and Aidong Zhang (1998). "Wavecluster: A multi-resolution clustering approach for very large spatial databases". In: *VLDB*. Vol. 98, pp. 428–439.

Shen, Jess Jiangsheng et al. (2007). "Using Cluster Ensemble and Validation to Identify Subtypes of Pervasive Developmental Disorders". In: *AMAI Annual Symposium Proceedings*, pp. 666–670.

Sneath, Peter H A and Robert R Sokal (1973). *Numerical taxonomy. The principles and practice of numerical classification.*

Spruit, Marco and Raj Jagesar (2016). "Power to the People! - Meta-Algorithmic Modelling in Applied Data Science". In: *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* 1, pp. 400–406.

Spruit, Marco and Miltiadis Lytras (2018). "Applied Data Science in Patient-centric Healthcare". In: *Telematics and Informatics* 35.April, pp. 643–653.

Spruit, Marco, Robert Vroon, and Ronald Batenburg (2014). "Towards healthcare business intelligence in long-term care: An explorative case study in the Netherlands". In: *Computers in Human Behavior* 30, pp. 698–707.

Stanley, Kay R., Abraham Fiszbein, and Lewis A. Opler (1987). "The Positive and Negative Syndrome Scale (PANSS) for Schizophrenia". In: *Schizophrenia Bulletin* 13.2, pp. 261–276.

Steinbach, Michael, Levent Ertöz, and Vipin Kumar (2004). "The Challenges of Clustering High Dimensional Data". In: *New Directions in Statistical Physics*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 273–309. ISBN: 3540431829.

Strauss, J.S., J.J. Bartko, and W.T. Carpenter (1973). "The Use of Clustering Techniques for the Classification of Psychatric Patients". In: *British Journal of Psychiatry* 122, pp. 351–540.

Strehl, Alexander and Joydeep Ghosh (2002). "Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions". In: *Journal of Machine Learning Research* 3, pp. 583–617.

Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar (2005). "Cluster Analysis: Basic Concepts and Algorithms". In: *Introduction to Data Mining*. 1st ed. Pearson. Chap. 8, pp. 487–555. ISBN: 978-0321321367.

Topchy, A., A.K. Jain, and W. Punch (2003). "Combining multiple weak clusterings". In: *Third IEEE International Conference on Data Mining*. December. IEEE Comput. Soc, pp. 331–338. ISBN: 0-7695-1978-4.

Topchy, A. et al. (2004). "Adaptive clustering ensembles". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* 1. IEEE, pp. 272–275. ISBN: 0-7695-2128-2.

Topchy, Alexander, Anil K. Jain, and William Punch (2005). "Clustering ensembles: Models of consensus and weak partitions". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.12, pp. 1866–1881.

Tukey, John W (1977). *Exploratory data analysis*. Vol. 2. Reading, Mass.

Van Der Maaten, L J P and G E Hinton (2008). "Visualizing high-dimensional data using t-sne". In: *Journal of Machine Learning Research* 9, pp. 2579–2605.

Vega-Pons, Sandro and José Ruiz-Shulcloper (2011). "A SURVEY OF CLUSTERING ENSEMBLE ALGORITHMS". In: *International Journal of Pattern Recognition and Artificial Intelligence* 25.3, pp. 337–372.

Vendramin, Lucas, Ricardo J. G. B. Campello, and Eduardo R. Hruschka (2010). "Relative clustering validity criteria: A comparative overview". In: *Statistical Analysis and Data Mining* 3.4, pp. 209–235.

Verschuren, Piet, Hans Doorewaard, and Michelle Mellion (2010). *Designing a research project*. Vol. 2. Eleven International Publishing The Hague.

Wang, Wei et al. (1997). "STING: A statistical information grid approach to spatial data mining". In: *VLDB*. Vol. 97, pp. 186–195.

Wardenaar, K. J. et al. (2014). "The effects of co-morbidity in defining major depression subtypes associated with long-term course and severity". In: *Psychological Medicine* 44.15, pp. 3289–3302.

Weerd, Inge van de and Sjaak Brinkkemper (2009). "Meta-Modeling for Situational Analysis and Design Methods". In: *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications*. IGI Global. Chap. 3, pp. 35–54. ISBN: 9781599048871.

Weinstein, John N et al. (2013). "The Cancer Genome Atlas Pan-Cancer analysis project". In: *Nature Genetics* 45.10, pp. 1113–1120.

Wieringa, Roel (2014). *Design Science Methodology for Information Systems and Software Engineering*, p. 493. ISBN: 9781605587196.

Wigman, Johanna T.W. et al. (2017). "A transdiagnostic network approach to psychosis". In: *Schizophrenia Bulletin* 43.1, pp. 122–132.

Wilkerson, Matthew D. and D. Neil Hayes (2010). *ConsensusClusterPlus: a class discovery tool with confidence assessments and item tracking*.

Wilkinson, Leland and Michael Friendly (2009). "History corner the history of the cluster heat map". In: *American Statistician* 63.2, pp. 179–184.

Wirth, Rüdiger (2000). "CRISP-DM : Towards a Standard Process Model for Data Mining". In: *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining* 24959, pp. 29–39.

Wolberg, William H., W. Nick Street, and Olvi L. Mangasarian (1995). *Breast Cancer Wisconsin (Diagnostic) Data Set*.

Yu, Zhiwen et al. (2012). "SC3: Triple Spectral Clustering-Based Consensus Clustering Framework for Class Discovery from Cancer Gene Expression Profiles". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9.6, pp. 1751–1765.

Yu, Zhiwen et al. (2013). "Hybrid Fuzzy Cluster Ensemble Framework for Tumor Clustering from Biomolecular Data". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 10.3, pp. 657–670.

Zhou, Z.H. (2012). "Clustering Ensembles". In: *Ensemble methods: foundations and algorithms*. Ed. by R. Herbrich and T. Graepel. Boca Raton: Taylor & Francis Group. Chap. 7, pp. 135–156. ISBN: 9781118914564.

Zhu, SuoYu et al. (2017). "A compromise solution between overlapping and overlooking DSM personality disorders in Chinese psychiatric practice". In: *Social Psychiatry and Psychiatric Epidemiology*, pp. 1–8.