Erasmus School of Economics

Erasmus University Rotterdam

Master Thesis Econometrics and Management Science

Robust clustering of rating scale data using DBSCAN

Henk-Jan Wermelink (484701)

University Supervisor: Dr. A. Alfons

Second Assessor: Dr. A. Archimbaud

April 25, 2022

Abstract

This research investigates whether density-based spatial clustering of applications with noise (DBSCAN) is suitable to use on rating scale data. The rating scale is frequently used in surveys within the social sciences domain, and previous research has shown that this type of data often contains noise caused by careless respondents. We analyse the performance and limitations of DBSCAN on this type of data. We start with a simulation that assesses how dimensionality and item scale size affect the performance of DBSCAN. This is followed by analysing the robustness properties of DBSCAN to careless responses. Finally, we showcase the performance in a real-world example. Our results show that DBSCAN performs increasingly well when the number of items and size of the item scale increases but lacks in scenarios where a small scale and number of items are used. Furthermore, it shows robustness against careless responses, being most effective when a respondent responds to at least 50% of items carelessly. On real-world data, it produces promising results, but we do not have decisive evidence of being superior to other methods. Nevertheless, we argue that DBSCAN can be used as a robust alternative on rating scale data when the variability in the data is large enough. In our case, when the number of items exceeds 5 in combination with at least a 7-point scale, or when multiple constructs are used regardless of the number of items and their scale.

Erasmus University Rotterdam

Contents

1 Introduction

Cluster analysis is a widely used technique to group objects in data that appear to be similar based on their features. Clustering these objects helps to create groups with similar characteristics and is used in various fields. This research focuses on clustering rating scale data, a type of data retrieved from closed-ended survey questions frequently used in social sciences. Respondents indicate their answer in a pre-set number of ordered categories, for example ranging from "fully agree" to "fully disagree" and having an arbitrary number of categories in between.

In general, clustering aims to get homogeneous subsets (clusters) within a heterogeneous data set to do proper testing or modelling. In most clustering algorithms, all objects are assigned to a cluster. This can cause problems as assigning outlying observations to a cluster will increase the heterogeneity within a cluster. To avoid this, various robust clustering methods have been proposed. For example, Fritz et al. (2012) proposes a trimming approach and developed the *tclust* package in R, Guha et al. (2000) introduced a robust clustering algorithm using links (ROCK), and more recently, Coretto et al. (2016) developed the robust improper maximum likelihood estimator (RIMLE).

Some research focuses specifically on clustering ordinal data. For example, Zhang et al. (2020) proposes a distance-based clustering algorithm with automated distance learning. Their proposal consists of a new distance measure that allows for inter-category distance and takes the order information of the ordinal data into account.

None of these methods has been specifically designed or studied for robust clustering of rating scale data. At the same time, robust clustering is important with this type of data. In surveys, it is common to have noise observations that are caused by untruthful respondents. Oftentimes this is defined as careless responding which is responding without giving sufficient or any attention to the instructions, resulting in random, patterned or inconsistent responses. Schroeders et al. (2021) suggest that 3% to 46% of respondents typically respond carelessly to one or more items. This is an issue as Credé (2010) suggests that as little as 5% of careless respondents can affect the result of a study.

Therefore, this research analyses the performance and limitations of density-based spatial clustering of applications with noise (DBSCAN), introduced by Ester et al. (1996), specifically on rating scale data. We believe DBSCAN could be a promising method for this application because it allows for any cluster shape, is flexible in the use of similarity measure and its theoretical robustness to noise. The general idea behind DBSCAN is that it groups observations close to each other (dense regions), while observations in low-density regions will not be assigned to a cluster, which should make the method robust.

The paper covers three main topics. First: we assess if DBSCAN is a suitable method for rating scale data and determine what dimensionality the discrete data needs to have for DBSCAN to work. Second: we design a simulation to investigate how robust DBSCAN is against random and careless responses. Third: we use real-world data to illustrate the practical performance of DBSCAN.

The main contributions of this research lie in showcasing the performance and limitations of DBSCAN on rating scale data. It gives the reader a reference to how DBSCAN performs on this type of data with different number of items and item scale sizes. We analyse the robustness properties of DBSCAN and specifically look into its robustness to careless responding, which is the most common type of noise in rating scale data. To our knowledge, there has not been research that specifically proposed or analysed robust clustering methods for rating scale data.

Our findings suggest that DBSCAN can be used as a robust alternative on data with a larger number of items (≥ 5) in combination with a bigger item scale (≥ 7), but should not be used on data with smaller scales and number of items. From that point the method starts outperforming the commonly used k-means algorithm when 5% of noise is present in the data. Moreover, the method is suitable for instances with multiple constructs regardless of the number of items and item scale. The method shows robustness against random noise as well as real careless responses. It is most effective when one assumes a careless respondent should be labelled as noise when it responds at least 50% of items carelessly. The practical example does not provide decisive evidence of a performance gain compared to commonly used methods like k-means but does suggest that formed clusters are of higher quality.

The structure of this paper is as follows. In Section 2 we discuss the relevant literature for this research. In Section 3 we introduce the methods that are used. Section 4 elaborates on the design of two simulations studies and discusses their results. In Section 5 we present and discuss the results of a real-world example. Section 6 concludes the findings of the study and suggests further research directions. Finally, Section 7 describes the limitations of this research.

2 Relevant literature

This section discusses the relevant literature for this research. Three main topics are covered. First, Section 2.1 covers the literature about existing robust clustering methods. Next, Section 2.2 discusses literature relevant to the rating scale and its assumptions. Finally, Section 2.3 displays literature related to different types of response styles.

2.1 Robust clustering

Clustering is a useful technique to identify groups of observations that appear similar to each other but dissimilar to observations in different groups. However, for a clustering method to be useful in practice it has to be robust (Davé et al., 1997). A clustering method is robust when it is not influenced significantly by noise or outlying observations. In the past decades, a lot of research has been done on creating new and robustifying existing clustering methods.

In the most simple form, statisticians suggest increasing the number of clusters to deal with individual or small groups of outliers (Garcia-Escudero et al., 2010). The idea is that these outliers are heterogeneous with respect to the regular observations and, therefore, will form their own cluster. The authors note that this may not be the most practical solution as isolated outliers would have to be grouped each in their own cluster. Moreover, Garcia-Escudero et al. (2010) notes that in some applications, the user already knows how many clusters to look for in the data because of domain knowledge and is unaware of separately clustering outlying observations.

Researchers have also introduced several trimming approaches to make clustering robust. The idea behind trimming is that instead of trying to cluster the outlying observations, these observations are trimmed before clustering. Cuesta-Albertos et al. (1997) proposed a procedure to robustify the popular distance-based k-means method. The trimmed k-means method lets the user specify a percentage of items α to be trimmed from the data, resulting in a version of k-means with a bounded influence function. The downside of this method is that it needs the user to find a proper α for their data set, and moreover, it assumes spherical clusters. Over the years, more iterations of trimming approaches have been introduced. Fritz et al. (2012) created the *tclust* package in R that implements several clustering methods that use the trimming approach.

Guha et al. (2000) introduced a robust clustering algorithm for categorical attributes, called robust clustering using links (ROCK). Instead of a distance measure, ROCK relies on a concept called links to measure similarity between observations. These links are used in a hierarchical clustering algorithm to form the clusters.

Besides linkage and distance-based methods, another popular type of robust clustering is density-based clustering. An example of this is DBSCAN, the method we intend to use. Density-based methods assume that observations connected to the same dense region are part of the same cluster. Density-based methods also require a distance function but are flexible because any similarity measure can be used. An advantage of density-based clustering is that it can identify noise as items that are not close to any dense region. Moreover, it does not assume a specific shape of the cluster. However, this type of clustering method lacks in separating clusters if they are connected and is sensitive for its parameters.

2.2 Rating scales

In this research, we focus specifically on rating scale data. The rating scale is a set of close-ended items which uses a scale containing a fixed number of ordered category options. For example, scales can range from 1 "oppose a great deal" to 7 "favour a great deal", or 1 "not likely" to 5 "very likely". One of the most commonly used rating scales in social sciences is the Likert scale introduced by Likert (1932). The Likert scale is a subset of the rating scale that measures the agreement on a statement from "fully disagree" to "fully agree" with an arbitrary number of options in between.

According to Harpe (2015), Likert intended in its original paper that the distances between each category are equal, which would mean the data could be used as interval scale. However, there has been much debate about whether the Likert scale is interval or ordinal. Ordinal data differs from interval, as it also assumes a rank order between the categories but does not assume equal intervals. Researchers suggest that Likert is ordinal because it cannot be assumed that the distance between "fully agree" and "agree" is equal to "agree" and "neutral". Jamieson (2004) suggests that, although Likert is strictly speaking ordinal, it is common practice in literature to treat Like-type categories as interval as researchers assume equal intervals in their research. Jamieson gives examples of multiple frequently cited medical papers where descriptive statistics as mean and standard deviation are used on Likert scale data.

Wu et al. (2017) performed a simulation study and suggested that the more Likert scale options per item, the closer it becomes to the interval scale. The authors note that it is a long-standing debate but that the scales are strictly speaking ordinal. They suggest an 11-point scale should be used to come closer to the assumption of equal intervals. The discussion has an impact on the decisions we make in this research. K-means uses the mean of a cluster in the method and therefore assumes an interval scale. In both DBSCAN and k-means, we use the Euclidean distance without altering the scales, and therefore we also assume interval scale. Our research consists of a carefully designed simulation study for which the assumption of equal intervals holds. For the empirical example, however, we are unsure of this. The data used contains a 7-point scale which is smaller than the suggested 11-point scale by Wu et al. (2017). Therefore, the assumption of equal intervals is potentially violated.

2.3 Response styles

When using rating scales in questionnaires the responses may be subject to different response styles. Response styles are a respondent's tendency to use a specific range of the item scale regardless of the content. Van Vaerenbergh et al. (2013) suggests that response styles can threaten the validity of results and sometimes can even be an alternative explanation for research findings.

There are various types of response styles. The first group of styles is regarded to how a respondent uses the scale to answer questions. Examples of this are acquiescence response styles (ARS), where respondents tend to agree by default. On the other hand, there is a dis-acquiescence response style (DARS) where people have the opposite tendency to disagree by default. Moreover, some respondents have response styles that only use the middle of the scale (mid-point response style) or only the outer response options (extreme response style). The respondent's different response styles are personal preferences, but literature has also shown that they can be linked to cultural differences (Johnson et al., 2005). Literature provides several methods to deal with this kind of response styles. Schoonees et al. (2015) introduced a constrained dual scaling (CDS) method that aims to detect and correct the four most common response styles. Furthermore, Takagishi et al. (2019) extended this method to use k-means clustering on CDS corrected data.

Secondly, there is a group of response styles that (almost) completely disregard the content when answering. Baumgartner et al. (2001) refers to this as non-contingent responding (NCR), meaning respondents respond randomly or carelessly in disregard of the item content. The term careless responding is often used interchangeably with NCR. Schroeders et al. (2021) suggest that 3% to 46% of respondents typically respond carelessly to one or more items. The authors state that the reason for this is a lack of attention or personal motivation. Bowling et al. (2021) studied the effect of the length of a survey on the number of careless responses. The authors found that respondents respond increasingly careless when progressing further into the survey. There seems to be no consensus in literature on when a careless respondent is troublesome regarding the number of items responded carelessly. Maniaci et al. (2014) suggests using a cut-off of 50%, meaning if a respondent answered more than 50% of items carelessly, the observation should labelled as a careless respondent.

This research focuses on the second group of response styles that include careless responding. We will not correct or simulate the first group of response styles. Moreover, we keep the discussion open on when a respondent should be labelled as careless and show results based on different cut-off assumptions.

3 Methodology

This section covers the methodology used in this research. In Section 3.1 we cover the k-means clustering algorithm. Next, Section 3.2 covers the DBSCAN algorithm and discusses how we determine its parameters. Lastly, Section 3.3 outlines the metrics used to evaluate performance.

3.1 K-means

This research uses k-means clustering as a baseline to compare performance. K-means is a commonly used clustering method that is introduced by MacQueen et al. (1967). The method splits a data set with n observations into k different clusters, where k is defined by the user. It is an iterative centroid-based method with the objective to minimise the within-variance of the k clusters. In k-means, the objective function is defined as

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2,$$

where c_j is the mean of all points in cluster j. With k-means all observations n are assigned to a cluster which makes it sensitive to outlying observations and noisy data. Moreover, due to the centroid-based nature of the method, it favours clusters that are spherical or convex shaped.

3.1.1 Algorithm

K-means uses an iterative algorithm to find the optimal value of the objective function J. In this research, we use a variation of the k-means algorithm introduced by Hartigan et al. (1979), which follows the same principle as the original method but differs in how points are assigned to clusters. We use this variation over the original as Slonim et al. (2013) suggest Hartigan's method has superior convergence properties. The method consists of three main parts. First, all points n are randomly assigned to a cluster S_j where $j \in 1, \dots, k$ and it calculates the centroid c_j of each cluster k by taking the mean of all points in S_j. Second, after the initialisation, it proceeds with an iterative update step. The method iterates over all observations $x \in S_j$ and considers moving x to any of the alternative clusters S_p, where $p \neq j$. It assesses the change in objective by computing the within-variance of both clusters in both scenarios. Let J_{Sj} be the part of the overall objective function where only the observations in cluster S_j are considered. We define J_{Sj} as

$$J_{Sj} = \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 .$$

For every alternative cluster p, we define the change in objective as

$$D(x, j, p) = \Delta(x, j, p) = J_{Sj} + J_{Sp} - J_{S_j \setminus \{x\}} - J_{S_p \cup \{x\}},$$

where j refers to the current cluster of observation x. The method then chooses the $p \in 1, \ldots, k$, which has the largest value for $D(x, j, p)$. If the value is > 0, x moves to the new cluster. The algorithm stops when no combination of (x, j, p) can be found such that $D(x, j, p) > 0$. As the method stops at local minima, the method is run with w different initial clusters, where the result is the cluster for which the lowest local minimum is found. In this research we use $w = 250$.

3.1.2 Optimal k

With k-means clustering, the user has to input the number of clusters, k, which the method will form. Determining the optimal value for k is difficult as there is no uniformly superior objective metric to determine the best value for k. Instead, there are some heuristics to determine the number of clusters (Kodinariya et al., 2013). We use the elbow method as a starting point. The idea behind the elbow method is to try different values for k and compare how the Within-Cluster-Sum of Squared Errors (WCSS) changes when increasing the value for k. The WCSS is identical to the objective function J discussed in the previous section. We run k-means with $k = 1, \ldots, 10$ and compute the WCSS for every result of the k runs. Then, we plot a line graph with k on the x-axis and WCSS on the y-axis. We choose the value for k that corresponds to the point where the graph shows a rapid change in slope, the elbow point. The idea behind it is that after that value for k, adding an extra cluster does not give much extra information.

The elbow method is subjective but often used due to a lack of alternatives. This is because in most clustering tasks, the truth label is unknown, and therefore the "correct" model does not exist. Therefore, we use the elbow plot as an indication of on which range of k we should evaluate the method. We then compute the average silhouette score introduced by Rousseeuw (1987) (Section 3.3.2) for every k in this range and use the best performing k.

3.2 Density-based spatial clustering of applications with noise

Density-based spatial clustering of applications with noise (DBSCAN) is a clustering algorithm proposed by Ester et al. (1996). The method is based on the principle that observations close together (dense region) are clustered, while observations not close to other observations are discarded and labelled as noise. This should make the algorithm robust. Like k-means, DBSCAN uses a distance measure but is more flexible as it allows for any distance or similarity measure to be used. Euclidean distance is most commonly used in literature and is also used in this research. Unlike k-means, DBSCAN does not need the number of clusters k to be specified by the user. Moreover, it allows for any cluster shape. However, DBSCAN is sensitive to its two user-specified parameters ϵ and $MinPts$. Furthermore, these parameters are hard to determine and can affect the number of clusters formed by the method. We elaborate on this issue in Section 3.2.2 and 3.2.3.

3.2.1 Algorithm

The algorithm has two global parameters, ϵ and $MinPts$. ϵ denotes the maximum distance between two observations for them to be reachable from each other. $MinPts$ indicates the number of observations that need to be reachable for an observation to be denoted as a core point. Together, these two parameters determine how dense a region must be to create a cluster. The algorithm starts at a random observation x_i and checks how many other observations can be reached within an ϵ distance from the starting observation. If at least $MinPts$ observations x_j where $x_i \neq x_j$ are reachable, the observation x_i is denoted as a core point and starts a new cluster S_j. Next, from each of the observations that were reachable from the core point, we check if there are at least $MinPts$ observations within an ϵ distance of this observation. If this is the case, we denote the observation as a core point, and all reachable observations from this point are be considered. If not, we denote the observation as a border point and do not consider the reachable observations from this point. In both cases, the observation is added to cluster S_j. All points that are not classified as either core or border points are labelled as noise. The process is repeated until all observations have been considered.

Figure 1 illustrates the DBSCAN method with 8 observations. The dotted lines around the observations are the range ϵ and $MinPts = 3$. The green observations are core points as each of them can reach at least 2 other points within an ϵ distance. The blue points are border points because they are reachable from a core point but do not reach at least 2 others. The red points are noise because no core points can be reached.

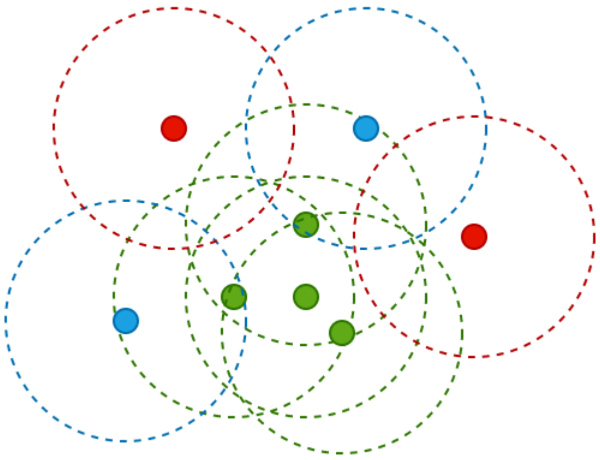

Figure 1: Illustration of the DBSCAN method with $MinPts = 3$. Green: core point. Blue: border point. Red: noise point.

3.2.2 Parameters

The user-specified parameters, ϵ and $MinPts$, determine the radius (ϵ) and the minimum amount of observations the algorithm needs in this radius to start a cluster ($MinPts$). There has to be some variability within the data to find an appropriate radius. This may not be the case since we are dealing with discrete rating scale data. To illustrate, consider a situation in the univariate space where there is one question with a 5-point scale. In this case, it is hard to determine an appropriate ϵ for the problem as many duplicates exist in the data. We either use a very small ϵ which means every option on the 5-point scale forms its own cluster as long as the number of people answering that option is greater or equal to $MinPts$. Or ϵ is chosen so that adjacent categories on the scale are reachable, and hence the algorithm likely forms one big cluster. The takeaway is that the algorithm cannot identify dense and sparse regions and therefore does not work as intended. Hence, we need some variability in the data for DBSCAN to work. We expect applying the method to a multivariate case where a respondent answers multiple questions will increase the variability and, therefore, the performance of DBSCAN. Furthermore, increasing the number of points on the scale is also expected to higher the variability and therefore have a similar effect.

3.2.3 Optimal parameters

Finding the optimal combination of ϵ and $MinPts$ is not a trivial task. Choosing a small $MinPts$ results in clusters with more noise, while choosing a large number may classify regular observations as noise. Often the default value $MinPts = 4$ is used for two-dimensional data, or various values are evaluated based on the user's domain knowledge (Schubert et al., 2017). Furthermore, Sander et al. (1998) suggests to take into account the dimension of the data, they propose to use $MinPts = dimension * 2$. After $MinPts$ is determined a value for ϵ has to be chosen. We do this by computing the distance to the k-nearest neighbour for every observation using $k = MinPts - 1$, as suggested by the original paper (Ester et al., 1996). A k-distance graph is then made, sorting the distances in increasing order.

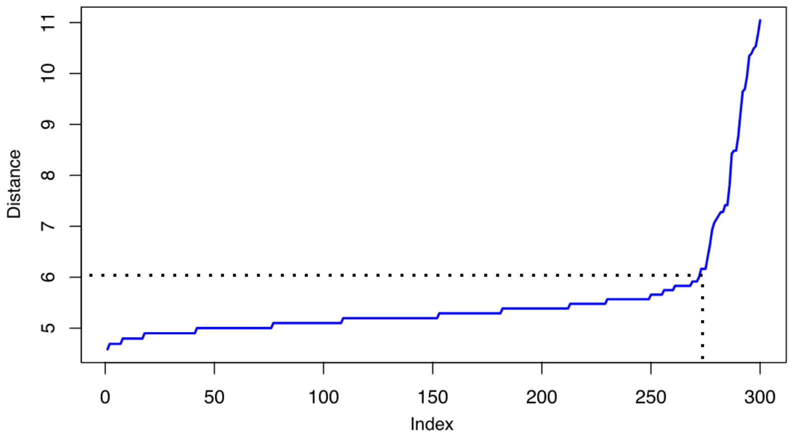

Figure 2: Illustration of the elbow method. Plotting the $k = MintPts - 1$ nearest neighbour distance of every observation in increasing order.

The elbow method is used to find a suitable value for ϵ, choosing the point where the biggest change in slope is observed. The idea behind this is to find an ϵ that is not too big as in that case, a big portion of all observations will be in the same cluster and not too small as a big part of the regular observations will then be denoted as noise. Finding the exact point at which an elbow is formed is hard and subjective. Moreover, in this research, we run simulations with various simulation settings and values for $MinPts$. This makes it practically impossible and inconsistent to manually pick ϵ based on the elbow method for every single run. Therefore, we use the k-nearest neighbour distance plots in combination with the silhouette score (Section 3.3.2) to determine the parameters of DBSCAN. The distance plots are used to find a range of ϵ for which it is reasonable to evaluate DBSCAN. We choose the range of ϵ in such a way that all elbow regions in the plots are included. We evaluate DBSCAN on the determined range and choose the parameters that maximise the silhouette score.

3.3 Performance

To evaluate how DBSCAN and k-means perform in the simulations, we measure their performance in terms of the adjusted Rand index (ARI) introduced by Hubert et al. (1985). Moreover, for the real-world data and parameter selection, we use the silhouette score.

3.3.1 Adjusted Rand index

The ARI is a commonly used evaluation metric to measure the similarity between two clusters. In our case, it measures the similarity between the true cluster and the cluster created by the clustering method. The Rand index can be viewed as the percentage of correct decisions made by the method. In addition, the adjusted Rand index adds a correction for chance by incorporating a comparison with a random model. The original formula from Hubert et al. (1985) is

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}\right] / \binom{n}{2}},$$

given two clustering results S and Q on n observations, containing $i \in 1, \ldots, c$ and $j \in 1, \ldots, k$ clusters respectively. The formula is based on a contingency table with c rows and k columns, representing the clusters of S and Q. Each cell in the table shows the number of observations that are both in S_i and Q_j, $n_{ij} = S_i \cap Q_j$. In the formula, $n_{i.}$ is defined as the sum of row i and $n_{.j}$ as the sum of column j. This results in the contingency table, which is derived from the original paper of Hubert et al. (1985), shown in Table 1.

	Q_1	Q_2	$\ldots$	Q_k	Row sum
S_1	n_{11}	n_{12}	$\ldots$	n_{1k}	$n_{1.}$
S_2	n_{21}	n_{22}	$\ldots$	n_{2k}	$n_{2.}$
$\vdots$	$\ldots$	$\ldots$	$\ddots$	$\ldots$	$\vdots$
S_c	n_{c1}	n_{c2}	$\ldots$	n_{ck}	$n_{c.}$
Column sum	$n_{.1}$	$n_{.1}$	$\ldots$	$n_{.k}$	

Table 1: ARI contingency table illustration, with c cluster in S and k clusters in Q.

A score of 1 indicates that the two clusters are identical. In the simulation study, this means that the clustering method perfectly models observations to their true label. An index of 0 indicates that the cluster is equally good to a method that randomly assigns observations to clusters. There is no lower bound for the ARI, and negative values generally indicate that there are intentional wrongly assigned observations, which we do not expect to encounter in our research.

3.3.2 Silhouette score

The silhouette score is a method for internal cluster validation. It measures for every observation how close it is to its own cluster and to observations in other clusters. The original formula for the silhouette coefficient given by Rousseeuw (1987) is

$$s(i) = \frac{b(i) - a(i)}{\max\{b(i), a(i)\}}.$$

In the formula $s(i)$ is the silhouette coefficient of observation i. The coefficient consists of $a(i)$ and $b(i)$. Where $a(i)$ is the average distance between observation i and all other observations in the same cluster and $b(i)$ is the average distance between observation i and all other observations that are not in the same cluster.

The average of all silhouette coefficients of the observations is often used as the overall evaluation metric. This score varies between -1 and 1, where 1 indicates that the clusters are dense and well separated. Moreover, one can compute the silhouette score for every cluster in the data set to analyse the quality of individual clusters. We must keep in mind that using the silhouette score as a comparison measure between different clustering algorithms can be misleading. The score favours clustering methods that create convex shaped clusters and, therefore, would have a bias toward such methods if other cluster shapes are present. Moreover, in the case of DBSCAN, it treats the observations labelled as noise as a cluster while it is actually not.

4 Simulations

We design two simulations to simulate questionnaire data based on rating scale items. To do this, we use the *GenOrdCat* method in the *simstudy*[1] package. The method allows for the simulation of correlated multivariate ordinal data. In Section 4.1 we discuss the design of the first simulation that aims to assess the overall performance of DBSCAN on rating scale data by varying the number of items and item scale size. Thereafter, Section 4.2 outlines the results. Section 4.3 introduces the design of the second simulation where a real-world example is used as a baseline. Moreover, noise is modelled as careless responding. Finally, Section 4.4 shows the result of the second simulation.

4.1 Design simulation 1

This section discusses the design of the first simulation study that evaluates DBSCAN on different simulated data sets, varying the number of items, item scale size, constructs, and noise.

[1]https://cran.r-project.org/web/packages/simstudy/simstudy.pdf

Section 4.1.1 outlines the general settings of the simulation. Next, the construction of the clusters and noise are discussed in Section 4.1.2 and 4.1.3 respectively. Lastly, Section 4.1.4 shows how parameters for the methods are selected.

4.1.1 Settings

We simulate data with the number of respondents $n = 300$ and generate $r = 100$ replications. We use two settings for the number of constructs $x = 1, 2$, varying the number of items per construct $m = 3, 5, 7$, with different number of options $q = 5, 7, 11$ per item. The number of clusters $C = 3$ is held constant in this simulation to represent three groups of respondents. The three groups can be identified as having a negative, neutral, or positive opinion to a certain construct or combination of multiple constructs (Section 4.1.2). In order to measure the robustness, we vary the amount of noise $o = 0\%, 5\%, 10\%$ (Section 4.1.3). We use correlated items within the constructs. In psychology, a construct is valid when we see a high correlation between the items within the construct. This shows that the items are related and hence measure the same construct. Both data with a valid and an invalid construct are simulated. We set the correlation coefficient $\rho = 0.8$ for the valid case and $\rho = 0.6$ for the invalid case. Furthermore, we use a compound symmetry structure as the scales of the items are equal. The constructs are generated independently from each other.

Table 2: Simulation Settings.

Simulation settings	
# Constructs	$1, 2$
# Items per construct	$3, 5, 7$
Item scale size	$5, 7, 11$
# Clusters	3
# Respondents	300
# Replications	100
Noise	$0\%, 5\%, 10\%$

4.1.2 Clusters

We simulate the data with three clusters, each representing a combination of negative, neutral, and positive opinions towards each construct. In the simulation, we randomize which cluster has which opinion while ensuring every cluster is unique. To illustrate, in a simulation where there is only one construct, each cluster has one unique opinion on that construct. If there are multiple constructs, we randomly assign the opinions on a certain construct between the clusters without duplicates. Hence, each cluster has a unique combination of opinions. For every opinion category c, we define a matrix of probabilities (*baseprobs*) based on which the discrete data is generated. Each row represents a single item, with the columns corresponding to the answer options. In this research, we define the base probabilities using a normal distribution that is identical for every item in the construct. We start by defining the normal distribution for every opinion. We define μ as $p = 10, 50, 90$th percentile of the q-point scale for the negative, neutral, and positive category respectively. Meaning for an item with a q-point scale ranging from $1 - q$, the μ_c of each category is $\mu_c = 1 + (q - 1) * p_c/100$. The standard deviation σ_c of each category is also based on the size of the scale q, $\sigma_c = q/10$. This results in a normal distribution for every item in category c given a q-point scale such that

$$X_c \sim \mathcal{N}(1 + (q - 1) * p_c/100,\ (q/10)^2)\,.$$

In Figure 3 we give a visual representation of the resulting distributions of each opinion category using a 7-point scale.

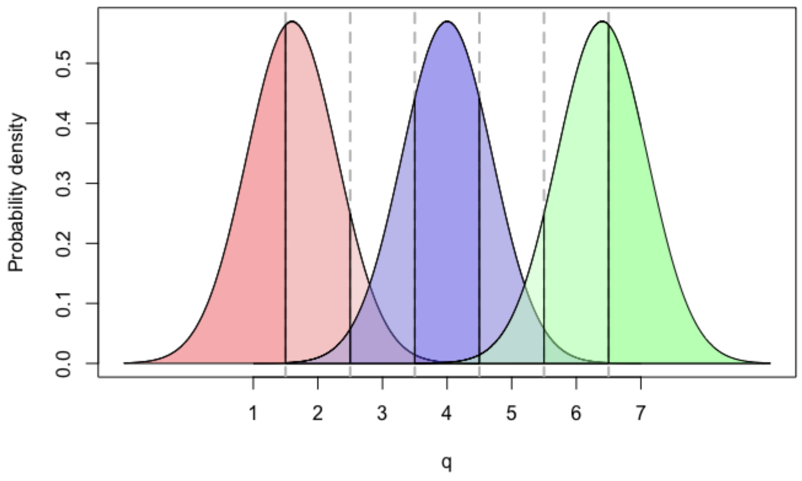

Figure 3: Illustration of probability distributions with a 7 point scale. Red = negative. Blue = neutral. Green = positive.

For this simulation, we create $h = 1, \ldots, q$ intervals defined as $(h-\frac{1}{2}, h+\frac{1}{2})$ for $h = 2, \ldots, q-1$, $(\infty, h+\frac{1}{2})$ for $h = 1$, and $(h-\frac{1}{2}, \infty)$ for $h = q$. We use the probability density function of X_c to determine the q probabilities for each row of the *baseprobs* matrix based on these intervals. Note that this results in a difference in variance between the opinion categories. The negative and positive categories are bounded on the low and high end of the q-point scale hence have lower variation than the neutral category.

4.1.3 Noise

The noise is generated from a discrete uniform distribution $\mathcal{U}(1, q)$. We use $\rho = 0$ as no correlation between the items is assumed. This simulates complete random noise that represents respondents giving random answers. Moreover, the noise is modelled such that a noise observation responds randomly to all items.

4.1.4 Parameter selection

In Section 3.2.3 we discussed that we use silhouette score to determine the optimal combination of ϵ and $MinPts$ for DBSCAN. Moreover, we discussed that there is no consensus of what a good starting point is for $MinPts$. Different suggestions have been made to either use 4 as a default value (for 2-dimensional data), take into account the dimension of the data, or using domain knowledge. In the end, it is a trade-off between choosing a bigger $MinPts$ if we expect to have noisy data and smaller when not. In this simulation, we use a wide range of $MinPts = 10, 20, 30, 40, 50$ to cover all suggestions and be sure there is an appropriate value for every simulation setting. The elbow method is used to find an appropriate range for ϵ to evaluate DBSCAN on. In Figure 4, the distance to the k-nearest neighbour are shown for varying simulations settings with 1 construct, where $k = MinPts - 1$. We increase the number of items and scale size in the data from top left to bottom right. With low dimensionality and scale, we see that it is difficult to spot where and if an elbow appears in the graph. This is because the data contains a lot of duplicates. The first graph shows that for half of all observations the 29th nearest neighbour is at a distance 0 and even more than half for the 19th nearest neighbour. This implies that the data consist of more than 50% non-unique values. If we look at the graph in the bottom right, we see a clearer point where an elbow can be spotted. This is between index 250 and 300 and spans around $\epsilon = 1.5$ to $\epsilon = 3.5$.

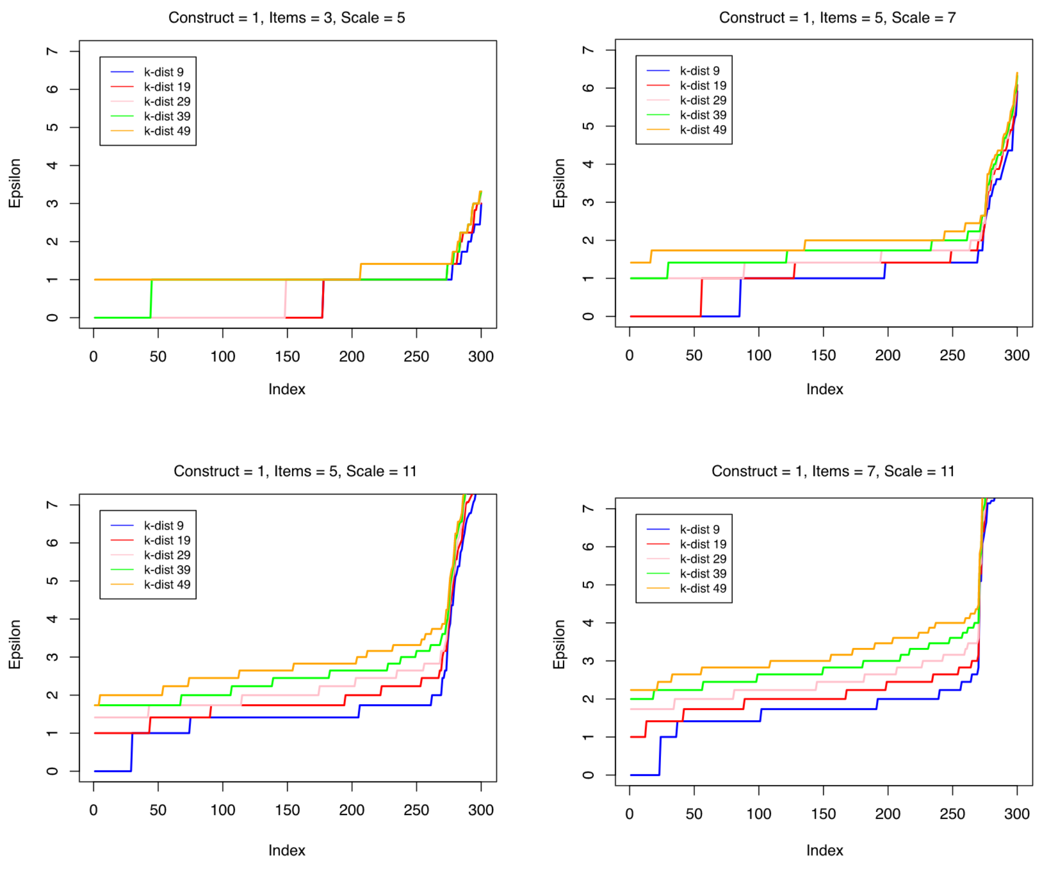

Figure 4: k-distance plot for simulation settings with 1 construct.

In Figure 5 we show several k-nearest neighbour distance plots with the same settings as in Figure 4, but this time having 2 constructs. A clear elbow can be spotted in all graphs, ranging from around $\epsilon = 1.0$ in the top left graph to $\epsilon = 5.5$ in the bottom right graph. All graphs combined, we use a range for ϵ between 1.0 and 5.5 with increments of 0.1 to make sure that we cover the full range of elbows with the different simulation settings and values for $MinPts$.

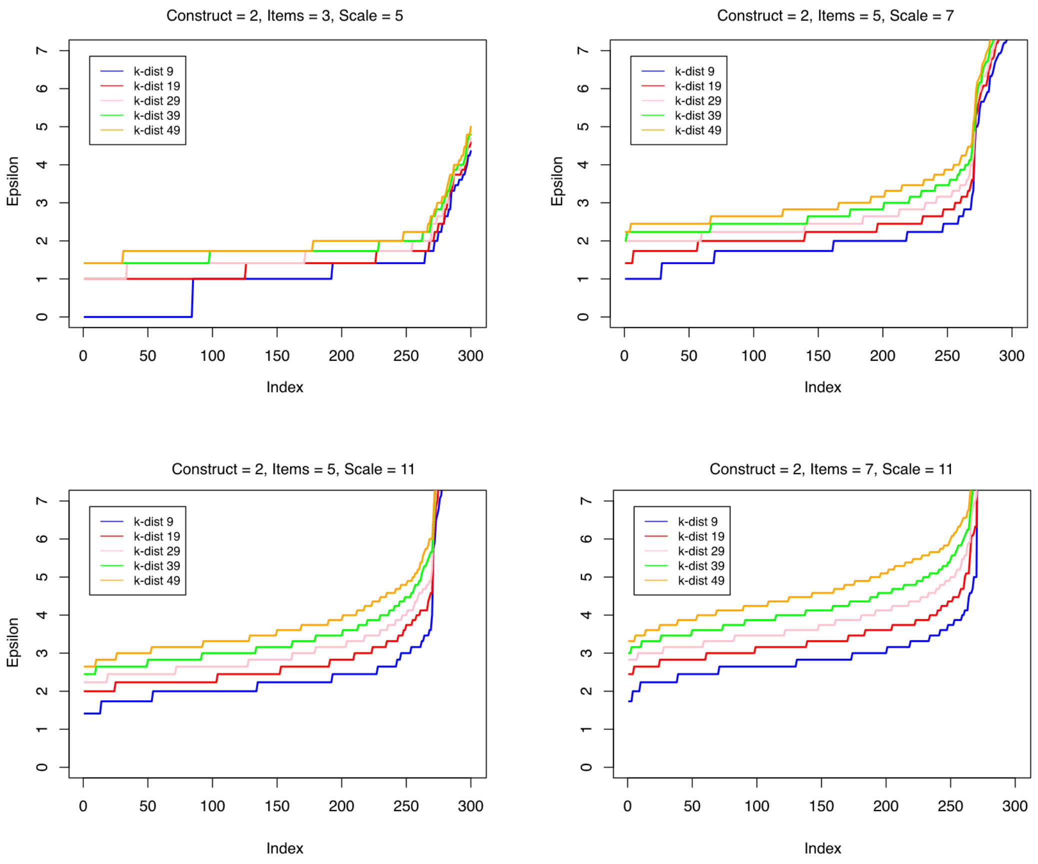

Figure 5: k-distance plot for simulation settings with 2 constructs.

For k-means, we have to choose the optimal parameter k. As discussed in Section 3.1.2 we plot the WCSS for different values for k and determine the range for k where an elbow shape is formed. In Figure 6 it can be seen that the elbow point looks to be starting to shape at $k = 3$. To make sure that we find the optimal k, we run the k-means algorithm for $k = 3, 4$ and select the k that maximises the silhouette score.

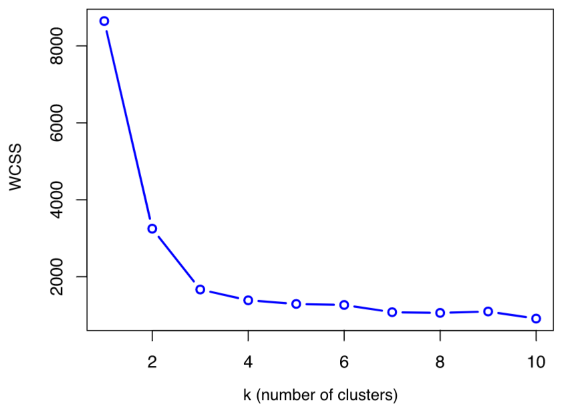

Figure 6: WCSS plot for simulation 1.

4.2 Results simulation 1

In this section, we discuss the results of simulation 1. Table 3 shows the simulation done with 1 construct and it consists of two panels. Panel A uses a valid construct ($\rho = 0.8$) and panel B an invalid construct ($\rho = 0.6$). Table 4 shows the results with 2 constructs, again with panel A using a valid construct and panel B an invalid construct. Furthermore, varying values for noise, item scale, and items per construct are used and displayed in both tables. All results are the mean of the simulation done with 100 replications. The structure of this section is as follows: Section 4.2.1 discusses the effect of the number of items and item scale size on the performance of DBSCAN, followed by Section 4.2.2 that covers the robustness, and lastly, Section 4.2.3 assesses the effect of construct validity on the performance.

4.2.1 Number of items & item scale size

The results indicate that both the number of items and size of the scale have an impact on how well DBSCAN performs. Table 3 shows that DBSCAN performs poorly when there is a small scale and a small number of items. Looking at the results with 1 valid construct in Table 3 panel A, we see that with 0% noise DBSCAN performs worse compared to k-means in all instances. However, increasing the number of items and their scale reduces the performance difference, reaching 0.01 difference in terms of ARI when using 7 items on an 11-point scale. The performance difference is especially noticeable when a 5-point scale is used. Moreover, we see a larger confidence interval in the results of DBSCAN in this scenario. This indicates that DBSCAN has a hard time finding the appropriate parameters for the data. The elbow plots discussed in Section 4.1.4 also suggest this. When we look at Figure 4 in that section, we see that there is no smooth elbow for the first two figures using a small number of items and scale. The distances take a clear step and then shoot directly up to observations that are noise. This means that there is low variability within in the data, resulting in an ϵ that is either so small that it only captures duplicates as a cluster or takes a distance that merges multiple clusters together.

Increasing the number of constructs helps the performance of DBSCAN. When comparing Table 3 and 4, it can be seen that DBSCAN performs significantly better when adding an additional construct with the same scale and number of items. Moreover, the difference between the performance of k-means and DBSCAN also drastically reduces. This is in line with the expectation that adding more variability helps DBSCAN form the proper clusters. Hence, adding an extra construct especially helps as the items between constructs are not expected to be as correlated compared to items within constructs.

4.2.2 Robustness

Results show that DBSCAN is more robust compared to k-means. In every scenario, the performance of DBSCAN is less affected by the noise than k-means. We see a drop in k-means performance when adding 5% noise, anywhere between 0.03 and 0.08 in terms of the ARI. DBSCAN loses 0.05 in the worst scenario, but stays flat in most scenarios. Looking at Table 3 panel A, we see that DBSCAN starts outperforming k-means in case noise is present in the data when the number of items and item scale size increases. This happens from a 7-point scale with 5 items. Increasing the percentage of noise also increases the performance difference between the two methods as k-means shows a similar drop going from 0% to 5% compared to going from 5% to 10%, while DBSCAN remains mostly flat. In some instances with a small number of items and a small item scale, we see the performance of DBSCAN increasing when adding noise. This can occur when observations that were wrongly labelled as noise in the first place become actual noise.

4.2.3 Validity of constructs

There is a performance difference between simulations done with a valid construct, $\rho = 0.8$, and an invalid construct, $\rho = 0.6$. This is most notable when comparing the results based on the simulation with 1 construct in Table 3 panel A & B. The lower ρ allows for more variability between answers to items from a respondent, resulting in more variations of possible answers and therefore in fewer duplicates. This helps DBSCAN find an appropriate distance for which a cluster should be formed. It also reduces the problem discussed in Section 4.2.1 of results that either merge multiple clusters or form clusters containing only duplicates.

Table 3: Simulation results with 1 construct. ARI (95% confidence interval).

Panel A: Correlation = 0.8 & Constructs = 1							
		items = 3		items = 5		items = 7	
Scale	Noise	k-means	DBSCAN	k-means	DBSCAN	k-means	DBSCAN
5	0%	**0.81**(±0.01)	0.59(±0.03)	**0.81**(±0.01)	0.67(±0.02)	**0.82**(±0.01)	0.58(±0.02)
	5%	**0.75**(±0.01)	0.57(±0.03)	**0.75**(±0.01)	0.67(±0.02)	**0.77**(±0.01)	0.65(±0.02)
	10%	**0.69**(±0.01)	0.56(±0.03)	**0.70**(±0.01)	0.67(±0.02)	**0.71**(±0.01)	0.66(±0.01)
7	0%	**0.86**(±0.01)	0.77(±0.02)	**0.86**(±0.01)	0.82(±0.01)	**0.87**(±0.01)	0.84(±0.01)
	5%	**0.80**(±0.01)	0.75(±0.02)	0.80(±0.01)	**0.84**(±0.01)	0.81(±0.01)	**0.86**(±0.01)
	10%	**0.74**(±0.01)	0.73(±0.02)	0.74(±0.01)	**0.84**(±0.01)	0.75(±0.01)	**0.86**(±0.01)
11	0%	**0.89**(±0.01)	0.87(±0.01)	**0.91**(±0.01)	0.89(±0.01)	**0.91**(±0.01)	0.90(±0.01)
	5%	0.82(±0.01)	**0.86**(±0.01)	0.84(±0.01)	**0.89**(±0.01)	0.84(±0.01)	**0.91**(±0.01)
	10%	0.76(±0.01)	**0.84**(±0.01)	0.77(±0.01)	**0.89**(±0.01)	0.78(±0.01)	**0.91**(±0.01)

Panel B: Correlation = 0.6 & Constructs = 1							
		items = 3		items = 5		items = 7	
Scale	Noise	k-means	DBSCAN	k-means	DBSCAN	k-means	DBSCAN
5	0%	**0.85**(±0.01)	0.71(±0.03)	**0.86**(±0.01)	0.79(±0.02)	**0.90**(±0.01)	0.79(±0.02)
	5%	**0.79**(±0.01)	0.65(±0.03)	**0.80**(±0.01)	0.76(±0.02)	**0.83**(±0.01)	0.80(±0.02)
	10%	**0.73**(±0.01)	0.61(±0.03)	0.74(±0.01)	**0.76**(±0.02)	0.76(±0.01)	**0.78**(±0.02)
7	0%	**0.90**(±0.01)	0.83(±0.01)	**0.91**(±0.01)	0.88(±0.01)	**0.92**(±0.01)	0.90(±0.01)
	5%	**0.84**(±0.01)	0.79(±0.02)	0.85(±0.01)	**0.89**(±0.01)	0.86(±0.00)	**0.91**(±0.01)
	10%	**0.77**(±0.01)	0.76(±0.02)	0.78(±0.01)	**0.89**(±0.01)	0.79(±0.00)	**0.91**(±0.01)
11	0%	**0.92**(±0.01)	0.91(±0.01)	**0.94**(±0.01)	0.93(±0.01)	**0.95**(±0.00)	0.94(±0.01)
	5%	0.86(±0.01)	**0.90**(±0.01)	0.88(±0.00)	**0.93**(±0.00)	0.88(±0.00)	**0.94**(±0.00)
	10%	0.79(±0.00)	**0.87**(±0.01)	0.80(±0.00)	**0.92**(±0.01)	0.81(±0.00)	**0.94**(±0.00)

Table 4: Simulation with 2 constructs. ARI (95% confidence interval).

Panel A: Correlation = 0.8 & Constructs = 2

		items = 3		items = 5		items = 7	
Scale	Noise	k-means	DBSCAN	k-means	DBSCAN	k-means	DBSCAN
5	0%	**0.96(**±0.00**)**	0.93(±0.01)	**0.97(**±0.00**)**	0.94(±0.01)	**0.98(**±0.00**)**	0.95(±0.01)
	5%	0.89(±0.00)	**0.93(**±0.01**)**	0.90(±0.00)	**0.95(**±0.01**)**	0.90(±0.00)	**0.96(**±0.00**)**
	10%	0.82(±0.00)	**0.91(**±0.01**)**	0.84(±0.00)	**0.95(**±0.01**)**	0.87(±0.00)	**0.96(**±0.00**)**
7	0%	**0.98(**±0.00**)**	0.97(±0.00)	**0.99(**±0.00**)**	0.97(±0.00)	**0.99(**±0.00**)**	0.98(±0.00)
	5%	0.91(±0.00)	**0.97(**±0.00**)**	0.91(±0.00)	**0.97(**±0.00**)**	0.92(±0.00)	**0.98(**±0.00**)**
	10%	0.85(±0.00)	**0.96(**±0.00**)**	0.87(±0.00)	**0.97(**±0.00**)**	0.89(±0.00)	**0.98(**±0.00**)**
11	0%	**0.99(**±0.00**)**	0.98(±0.00)	**0.99(**±0.00**)**	0.99(±0.00)	**0.99(**±0.00**)**	0.98(±0.00)
	5%	0.91(±0.00)	**0.98(**±0.00**)**	0.92(±0.00)	**0.99(**±0.00**)**	0.92(±0.00)	**0.99(**±0.00**)**
	10%	0.86(±0.00)	**0.97(**±0.00**)**	0.89(±0.00)	**0.99(**±0.00**)**	0.89(±0.00)	**0.99(**±0.00**)**

Panel B: Correlation = 0.6 & Constructs = 2

		items = 3		items = 5		items = 7	
Scale	Noise	k-means	DBSCAN	k-means	DBSCAN	k-means	DBSCAN
5	0%	**0.98(**±0.00**)**	0.95(±0.00)	**0.99(**±0.00**)**	0.97(±0.00)	**0.99(**±0.00**)**	0.97(±0.01)
	5%	0.90(±0.00)	**0.95(**±0.00**)**	0.91(±0.00)	**0.97(**±0.00**)**	0.91(±0.00)	**0.98(**±0.00**)**
	10%	0.85(±0.00)	**0.93(**±0.01**)**	0.88(±0.00)	**0.97(**±0.00**)**	0.88(±0.00)	**0.98(**±0.00**)**
7	0%	**0.99(**±0.00**)**	0.98(±0.00)	**1.00(**±0.00**)**	0.99(±0.00)	**1.00(**±0.00**)**	0.99(±0.00)
	5%	0.92(±0.00)	**0.97(**±0.00**)**	0.92(±0.00)	**0.99(**±0.00**)**	0.92(±0.00)	**0.99(**±0.00**)**
	10%	0.86(±0.00)	**0.96(**±0.00**)**	0.89(±0.00)	**0.98(**±0.00**)**	0.90(±0.00)	**0.99(**±0.01**)**
11	0%	**1.00(**±0.00**)**	0.99(±0.00)	**1.00(**±0.00**)**	0.99(±0.00)	**1.00(**±0.00**)**	0.99(±0.00)
	5%	0.92(±0.00)	**0.98(**±0.00**)**	0.93(±0.00)	**0.99(**±0.00**)**	0.93(±0.00)	**0.99(**±0.00**)**
	10%	0.88(±0.00)	**0.97(**±0.00**)**	0.90(±0.00)	**0.99(**±0.00**)**	0.91(±0.00)	**0.98(**±0.01**)**

4.3 Design simulation 2

In the previous section we outlined a theoretical example of simulating rating scale based survey data. While this serves to illustrate the effectiveness of DBSCAN in these scenarios, it may lack in representing a more real-world setting. Therefore we design a simulation that represents a more realistic scenario in this section. Section 4.3.1 goes over the settings used in this simulation. Next, Section 4.3.2 explains how noise is modelled as careless responding from real-world data. Lastly, Section 4.3.3 covers the parameters selected in this simulation.

4.3.1 Settings

We use the survey data of a study from Schroeders et al. (2021) as a baseline. This survey consists of $x = 6$ constructs each with $m = 10$ items on a $q = 5$ point scale. $n = 605$ respondents participated in the survey and consisted of two groups. The first group, $n = 361$, got instructions to read every question carefully when answering, while the second group, $n = 244$, were instructed to quickly finish the survey without carefully reading the questions. This second group is emulating careless responses.

This simulation tries to replicate this real-world scenario as close as possible. We use the same survey size of $x = 6$ constructs each with $m = 10$ items on a $q = 5$ point scale. The data of $n = 300$ respondents, consisting of $C = 3$ clusters, is generated with the same method as in the previous section (4.1.2), meaning every cluster has a unique opinion (negative, neutral or positive) on each construct. Instead of using a theoretical ρ, we use the correlation matrix of the real-world data in the data generating process. An overview of the settings is shown in Table 5.

Table 5: Simulation Settings.

Simulation settings	
# Constructs	6
# Items Per Construct	10
Item Scale Size	5
# Clusters	3
# Respondents	300
# Replications	100
Noise	0%, 10%, 20%, 30%, 40%

Noise is not randomly generated but is sampled from the careless responses in the real-world data. Moreover, we do not define a respondent as noise only when it responds completely careless. Instead, in practical examples, respondents may become careless after a certain point in the survey. More details on how we generate noise is discussed in Section 4.3.2. We vary the amount of noise $o = 0\%, 10\%, 20\%, 30\%, 40\%$.

4.3.2 Noise

In simulation 1, we defined noise as respondents who randomly responded to all items in the survey. However, this is not a completely accurate representation as the literature suggests that humans cannot produce random responses and often use certain patterns to quickly finish the survey (Figurska et al., 2008). Next to that, careless respondents do not always respond to all of the items carelessly. A study of Brühlmann et al. (2020) analysed careless responding in a survey where respondents were asked to self-report if their responses were subject to carelessness. The survey asked respondents on a 7-point scale (1 = never, 4 = half of the time, 7 = all the time) if they responded carelessly in an indirect manner with three different items. 54% of respondents indicated that they did not carelessly answer any of the survey items, while 46% admitted having answered one or multiple items carelessly. This study uses the 46% of careless respondents to determine how many items such respondents typically answer carelessly. To do this we transform the 7-point scale linearly into percentages of items being answered careless (1 = 0%, 4 = 50%, 7 = 100%). Next, we use the average response to the three items related to careless responding and round them to the nearest integer. We discard the respondents that fall into the "1" category and determine the relative group size of respondents based on their self-reported severity of careless responding. The result is shown in Table 6. We end up with five different groups as nobody is part of the "7" category. We see that most careless respondents (51%) fall into the least severe group that indicated to have answered 17% of items carelessly. The size of the groups decreases when the severity of careless responding increases.

Table 6: Distribution of careless respondents based on their self-reported careless responding score.

Original answer	2	3	4	5	6	7
Percentage of careless responses (%)	17	33	50	67	83	100
Relative group size (%)	51	26	16	6	1	0

We generate the noise based on the data of Table 6 in combination with the careless survey data collected by Schroeders et al. (2021). Depending on the amount of noise o we want to use, we randomly pick $t = n * o$ observations from the regular observations generated by the method discussed in Section 4.1.2. We then divide the t observations into five groups, G, shown in Table 6, by their relative group size. Every group has a unique percentage of careless responses p_G. For every observation t_G we replace a p_G percentage of responses with a sequence of item responses sampled from the careless data of Schroeders et al. (2021). To be as realistic as possible, we replace the p_G percentage of the last item responses in the survey. It reflects the research of Baumgartner et al. (2001), which suggests respondents become more careless when they progress further in the survey. This results in noise consisting of five different groups each with varying portions of carelessly responded items, ranging from 17% to 83%. In Figure 7 we give a visual illustration of how noise looks like in this case. Every row is a careless respondent and every column represents a survey item. Green indicates a regular response while red indicates a careless response.

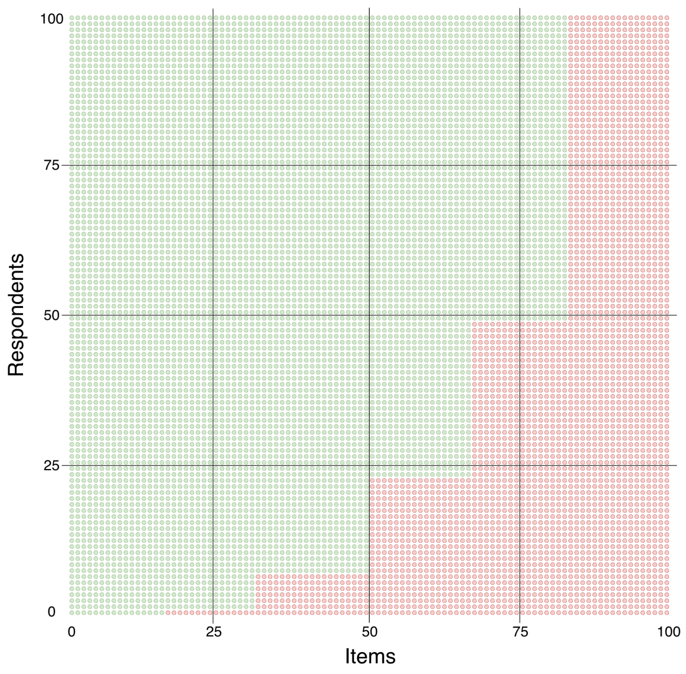

Figure 7: Illustration of simulating 100 careless respondents responding to 100 items. Green = regular response. Red = careless response.

4.3.3 Parameter selection

An appropriate range for parameters ϵ and $MinPts$ is found by the elbow method. Like in Section 4.1.4 we use a wide range of $MinPts = 10, 20, 30, 40, 50$. Again we plot the k-nearest neighbour distance for $k = MinPts - 1$. The result is shown in Figure 8. The elbow seems to occur between 5.0 and 7.0. To cover the full range, we use a range from 5.0 to 7.0 for ϵ with 0.1 increments.

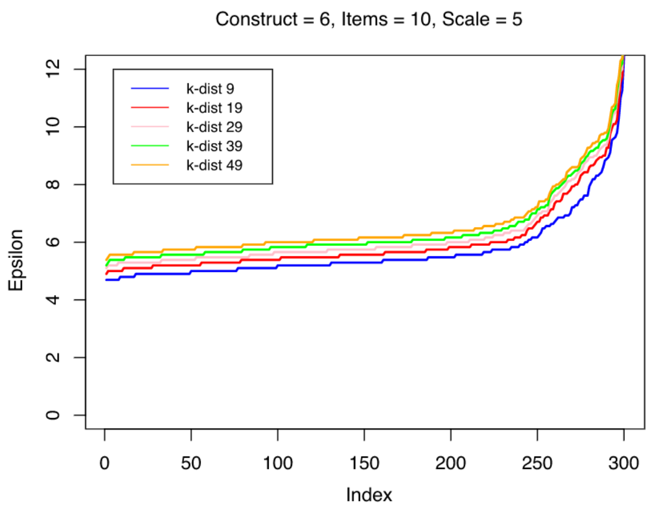

Figure 8: k-distance plot of simulation 2.

We use the same method as in simulation 1 in order to determine the optimal value for k in k-means. Figure 9 shows the WCSS for $k = 1, \ldots, 10$. The elbow seems to start forming at $k = 3$. To make sure that we find the optimal k, we run k-means on $k = 3, 4$ and choose the value for k that maximises the silhouette score.

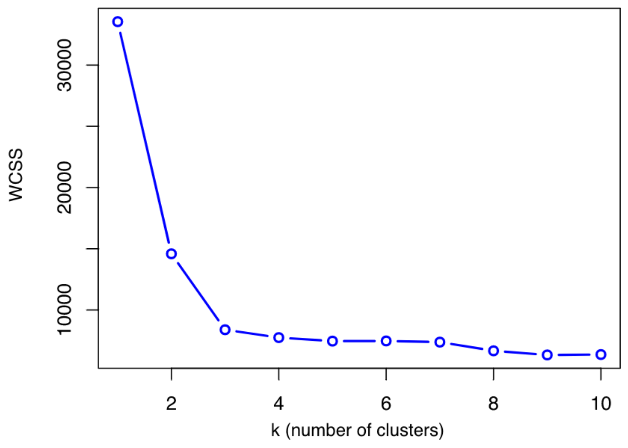

Figure 9: WCSS plot for simulation 2.

4.4 Results simulation 2

In this section, we discuss the results of the second simulation. This simulation aims to replicate a more realistic scenario mirroring the size, correlation, and item scale size of a real-world survey by Schroeders et al. (2021). The simulation generates regular observations similarly to the first simulation, while noise is modelled as careless responding and sampled from the real-world data. The next Section, 4.4.1, shows the performance of DBSCAN and compares it to our baseline k-means while varying the number of careless respondents. In Section 4.4.2 we discuss the effect of different assumptions on when one should label an observation as noise. Lastly, in Section 4.4.3 we dive deeper into the effect of the portion of careless responses per respondent on the effectiveness of DBSCAN.

4.4.1 Performance

We define noise as careless respondents and vary the amount from 0% to 40%. According to Schroeders et al. (2021), the amount of careless respondents varies anywhere from 3% to 46%. Moreover, in a survey held by Brühlmann et al. (2020), 46% of respondents self-reported to have answered one or multiple questions carelessly. In Table 7 the results of the simulation study are shown in terms of the ARI. We see that DBSCAN and k-means both perform equally well when there are no careless respondents in the data. When adding noise, we see a considerable drop in the ARI of both methods. However, DBSCAN outperforms k-means by 0.04 to 0.08 in every scenario where noise is included.

Table 7: Simulation 2 ARI (95% confidence interval) varying the amount of noise in the data.

Noise	0%	10%	20%	30%	40%
DBSCAN	1.00(±0.00)	0.90(±0.00)	0.75(±0.00)	0.59(±0.00)	0.42(±0.00)
k-means	1.00(±0.00)	0.84(±0.00)	0.67(±0.00)	0.53(±0.00)	0.38(±0.00)

4.4.2 Cut-off value

One could argue what the cut-off value, in terms of the percentage of items answered carelessly, should be for a respondent to be labelled as noise. In the previous section, we classify respondents that committed any form of careless responding as noise. At the same time, other research suggests that a cut-off value of 4 (50%) should be used to identify problematic observations (Maniaci et al., 2014). To assess how the results change in such a case we re-run the simulation but now with a cut-off value of 4 for a respondent to be labelled as noise. This means that in case of 40% we label 9.2% as noise (23% of 40%), which consist of respondents having answered between 50% and 83% of items carelessly (Table 9). At the same time, the other 30.8% that respond carelessly to less than half of the items keep their original label. The results are shown in Table 8. Both methods perform significantly better compared to classifying all careless respondents as noise. We see that DBSCAN and k-means perform equally well, both in case of no noise and 10% noise. DBSCAN starts outperforming k-means when increasing the amount of noise, having an ARI of 0.02 higher at 20% and 0.04 at 40%.

Table 8: Simulation 2 ARI (95% confidence interval) varying the amount of noise in the data, using 50% as cut-off.

Noise	0%	10%	20%	30%	40%
DBSCAN	1.00(±0.00)	0.96(±0.00)	0.94(±0.00)	0.92(±0.00)	0.90(±0.00)
k-means	1.00(±0.00)	0.96(±0.00)	0.92(±0.00)	0.90(±0.00)	0.86(±0.00)

While Maniaci et al. (2014) suggests 50% (4) is a good cut-off value, there is no consensus in literature. To illustrate how the results change when different assumptions on an appropriate cut-off value are made, we run the same simulation varying the cut-off value from 17% to 83%. We simulate with 40% noise resulting in different percentages of careless respondents labelled as actual noise depending on the cut-off value (Table 9).

Table 9: Percentage of careless respondents actually labelled as noise depending on the cut-off value. Simulating with 40% noise.

Cut-off value	17%	33%	50%	67%	83%
% labelled as noise	40.0%	19.6%	9.2%	2.8%	0.4%

The results of the simulation are displayed in Table 10. Comparing the ARI of DBSCAN to k-means, we see that until a cut-off value of 50%, DBSCAN outperforms k-means, but after that, k-means performs better. Hence, if one would want to assume a cut-off value greater or equal to 67%, our results argue k-means is the more suitable choice. While below and equal to 50%, DBSCAN is the better choice, producing clusters closest to expectation at the 50% mark.

Table 10: ARI (95% confidence interval) for k-means and DBSCAN varying cut-off value. Simulating with 40% noise.

Cut-off value	17%	33%	50%	67%	83%
DBSCAN	0.42(±0.00)	0.83(±0.00)	0.90(±0.00)	0.85(±0.01)	0.83(±0.01)
k-means	0.38(±0.00)	0.69(±0.00)	0.86(±0.00)	0.96(±0.00)	0.98(±0.00)

4.4.3 Percentage of careless responses

In this simulation study, we designed noise as observations that answer a certain portion of items carelessly (Section 4.3.2). The previous sections indicated that increasing the cut-off value helps the performance of DBSCAN, peaking at a cut-off value of 50% (Table 10).

To assess how the portion of careless responded items affects the performance of DBSCAN in more detail, we run simulations with a fixed portion of careless responses per careless respondent. We simulate with 20% noise while varying the number of items in this noise being answered carelessly from 10% to 50%. Looking at Table 11 we see that DBSCAN struggles to identify noise when only a small portion of items are answered carelessly, having an ARI index of 0.68 when 10% of items are answered carelessly. Increasing at 30%, to 0.76 and reaching 0.94 at 50%. The results suggest that DBSCAN does not reliably label observations with a small portion of careless responses as noise. This can either be positive or negative depending on the cut-off assumptions one would make when a respondent should be labelled as noise.

Table 11: ARI (95% confidence interval) based on 20% noise, varying the number of careless responses in noise observations.

Careless items	10%	20%	30%	40%	50%
DBSCAN	0.68(±0.00)	0.69(±0.00)	0.76(±0.01)	0.86(±0.01)	0.94(±0.01)
k-means	0.68(±0.00)	0.68(±0.00)	0.68(±0.00)	0.68(±0.00)	0.68(±0.00)

The results are in line with the results in Section 4.4.2, that DBSCAN performs better when the user assumes a higher cut-off value. Though, we also noted that DBSCAN performance is best at a cut-off of 50% and drops off when becoming higher. To find out why this is we take a deeper look into the clustering outcome of DBSCAN with 40% noise. Table 12 shows the percentage of observations labelled as noise by DBSCAN for each of the careless responding groups. Meaning every column represents a subset of the noise based on their respective percentage of careless responses (Section 4.3.2). The results show that DBSCAN only labels 1% of observations as noise that responded carelessly to 17% of items, increasing to 79% at 50% and reaching 99% at 83%. Hence, it explains why DBSCAN peaks at a cut-off value of 50% as from this point on, DBSCAN labels the majority of observations as noise.

Table 12: Percentage of observations labelled as noise (95% confidence interval) by DBSCAN per careless group. Simulating with 40% noise.

Careless group	17%	33%	50%	67%	83%
DBSCAN	0.01(±0.00)	0.37(±0.03)	0.79(±0.02)	0.95(±0.02)	0.99(±0.02)

5 Empirical data

In the previous sections we described two simulation studies that assess the theoretical properties of DBSCAN. In this section we discuss the performance of DBSCAN on a real-world data set to illustrate the performance in practice. Section 5.1 outlines the data and parameter settings used. Section 5.2 goes over the results.

5.1 Design

This section goes over the setup of the practical example. We discuss the data used in Section 5.1.1. Next, Section 5.1.2 goes over how we generate artificial noise. Lastly, Section 5.1.3 outlines the parameter selection.

5.1.1 Data

We use the American National Election Studies (ANES) survey results[2] of April 2020. This survey was conducted in the United States before the 2020 elections in November. The survey aims to study the political opinion of respondents based on questions on several important topics like #MeToo, immigration, impeachment, and the pandemic. The survey contains 470 questions and has 3080 respondents. Questions range from topics about the candidate's profile, traits, emotions to political opinion. The questions are close-ended and different scales are used throughout the survey. For this research, we take a subset of the questions that try to measure the opinion of, at that time, president Trump. Moreover, we only use questions that are answered on a 7-point rating scale. This results in a data set with 7 questions that try to measure the respondent's opinion on Trump as the president, shown in Table 13. The question: *"If the 2020 presidential election were between Donald Trump for the Republicans and Joe Biden for the Democrats, would you vote for Donald Trump, Joe Biden, someone else, or probably not vote?"* is asked at the beginning of the survey. We use respondents who intended to either vote Trump or Biden. Furthermore, the survey has two versions with different questions and our questions are from version 1. This results in a sample size of $n = 1258$, containing 615 Trump and 643 Biden voters.

[2] https://electionstudies.org/data-center/2020-exploratory-testing-survey/

Table 13: Sample of survey questions with their scale.

Question	Scale
Q1: Do you approve, disapprove, or neither approve nor disapprove of the way Donald Trump is handling his job as president?	(1) Approve extremely strongly ... (7) Disapprove extremely strongly
Q2: Do you approve, disapprove, or neither approve nor disapprove of the way Donald Trump is handling relations with foreign countries?	(1) Approve extremely strongly ... (7) Disapprove extremely strongly
Q3: Do you approve, disapprove, or neither approve nor disapprove of the way Donald Trump is handling immigration?	(1) Approve extremely strongly ... (7) Disapprove extremely strongly
Q4: Do you approve, disapprove, or neither approve nor disapprove of the way Donald Trump is handling the economy?	(1) Approve extremely strongly ... (7) Disapprove extremely strongly
Q5: Do you approve, disapprove, or neither approve nor disapprove of the way Donald Trump is handling the coronavirus (COVID-19) outbreak?	(1) Approve extremely strongly ... (7) Disapprove extremely strongly
Q6: Do you favor, oppose, or neither favor nor oppose the U.S. House of Representatives' decision in December of last year to impeach President Trump?	(1) Favor a great deal ... (7) Oppose a great deal
Q7: Do you favor, oppose, or neither favor nor oppose the U.S. Senate's decision in February to acquit President Trump of the impeachment charges and thus let him remain in office?	(1) Favor a great deal ... (7) Oppose a great deal

5.1.2 Noise

Next to the possibly existing noise in the data, we generate artificial noise to assess the method's robustness. We use a combination of the methods that we used to generate noise in simulations 1 and 2. In simulation 2, we discussed that noise in surveys often occurs because of careless responding and that the severity of responding varies between them. We use this same concept and generate noise that is split into groups that differ in the number of items answered carelessly (severity), in accordance to Table 6. Moreover, we discussed that humans are not capable of answering randomly, and therefore we sampled from a survey that was designed to be answered carelessly. However, this survey is based on questions with a 7-point scale, which does not match the 5-point scale in the real-world data set from which we sampled careless responses. Therefore, we simulate the careless responses in the same way as in simulation 2 (Section 4.3.2), but instead of sampling careless responses we use random responses. This results in noise that has varying portions of real and random responses based on the group sizes and severity of careless responding, shown in Table 6.

5.1.3 Parameter selection

Similarly to simulations 1 and 2, we use the elbow method to find an appropriate range of combinations for $MinPts$ and ϵ to evaluate DBSCAN on. Like in Section 4.1.4 we use a set of $MinPts = 10, 20, 30, 40, 50$. Again, we plot the distance to the k-nearest neighbour with values for $k = MinPts - 1$, shown in Figure 10. Judging from the figure we notice an elbow point between 2.0 and 5.0. To cover the full range, we use a range from 2.0 to 5.0 for ϵ with 0.1 increments.

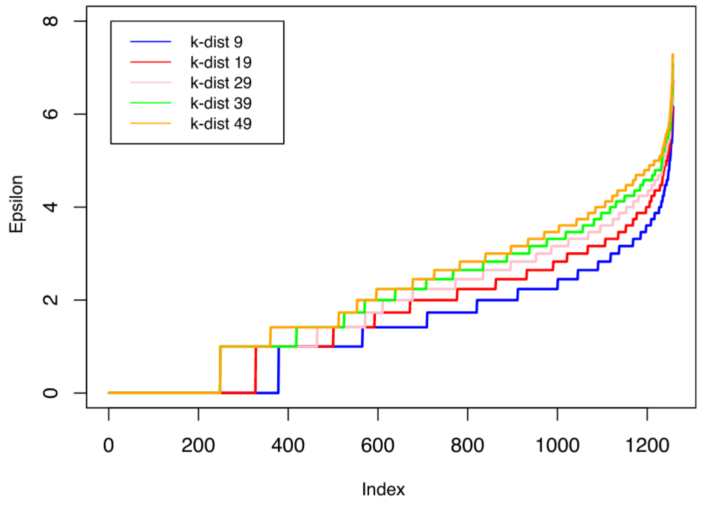

Figure 10: k-distance plot of the real-world data.

We use the same method as in both simulations in order to determine the optimal value for k in k-means. Figure 11 shows the WCSS for $k = 1, \ldots, 10$. The elbow starts to form around $k = 2$. We, therefore, run k-means on $k = 2, 3$ and choose the value for k that maximises the silhouette score.

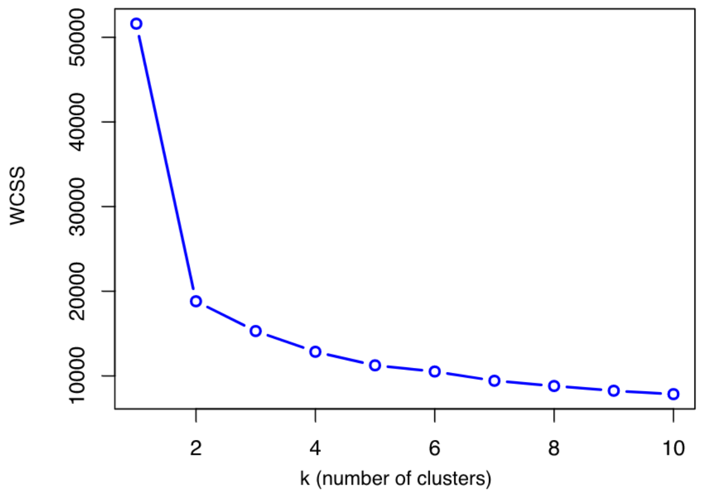

Figure 11: WCSS plot of real-world data.

5.2 Results

In this section we discuss the performance of DBSCAN in comparison with k-means on the ANES survey data set. As outlined in Section 5.1.1 we use a sample of a survey that aims to measure the political opinion of respondents in the United States. Section 5.2.1 covers and compares the general performance of both methods on the data set. Next, Section 5.2.2 analyses the robustness by adding artificial noise to the data.

5.2.1 Performance

We run the methods on the data set with different parameters outlined in Section 5.1.3. With both methods, we find two clusters when using the optimal parameters. With k-means the formed clusters each contain 56% and 44% of observations, respectively. DBSCAN labels 10% of observations as noise and forms two clusters with 55% and 35% of observations each. For both methods, we show the boxplots of each cluster to see how the clusters differ in terms of assessment of the questions. The results are shown in Figure 12. From the plots we can see that both DBSCAN and k-means form two clusters that are opposites of each other. Cluster 1 consists of respondents clearly approving of Trump's handling as a president, while cluster 2 clearly disapproves.

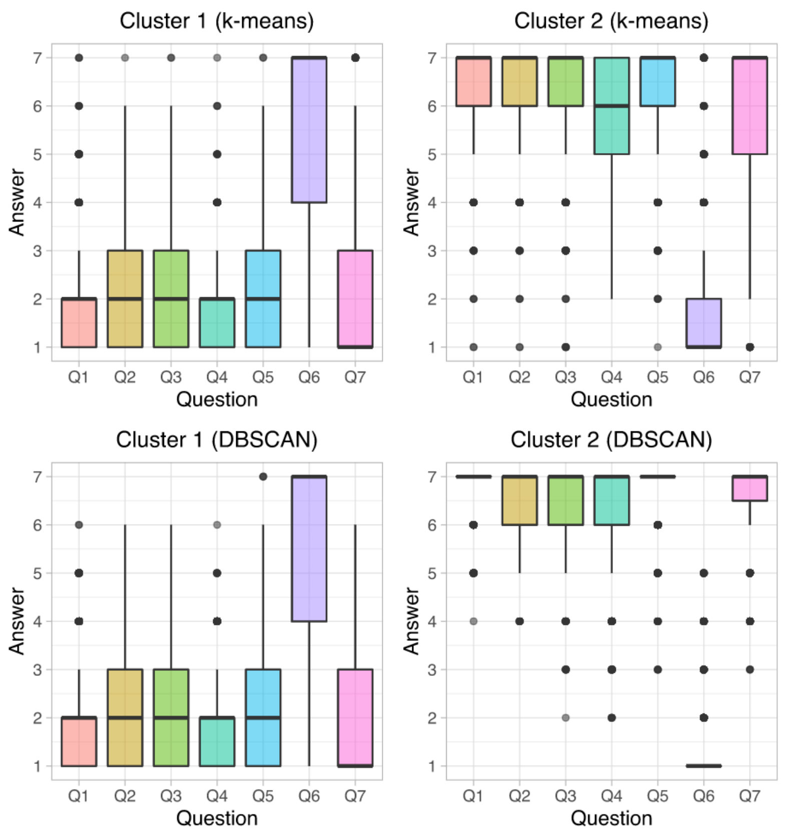

Figure 12: Boxplot of clustering results of DBSCAN and k-means.

Comparing the clusters formed by k-means and DBSCAN, we see that they are very similar, but DBSCAN has slightly smaller quartiles, especially in cluster 2. This is because DBSCAN labelled 10% of observations as noise which by design deviate from the mean of the cluster. Figure 13 shows the boxplot of the respondents labelled as noise. The plot shows larger quartiles compared to the boxplots of the actual clusters, with means leaning more towards the disapprove and oppose options, indicating it contains answers with varying opinions. Having 10% of noise is within range of the amount of noise typically seen in questionnaire data (Section 2.3).

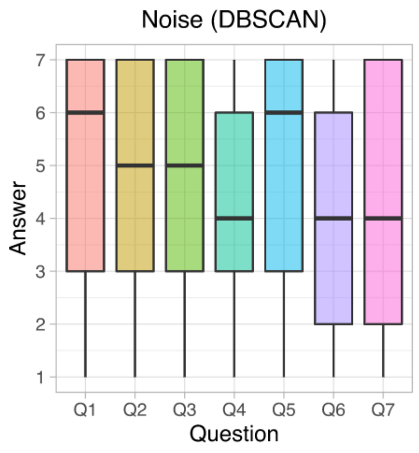

Figure 13: Boxplot of DBSCAN noise observations.

To evaluate the quality of the formed clusters, we use the silhouette score. As discussed in Section 3.3.2 it is tricky to directly compare the performance of two different methods using the silhouette score. We, therefore, use this as an indication. The silhouette score is calculated based on the average silhouette coefficient of all observations. However, DBSCAN labels a group of observations as noise which have a very negative silhouette coefficient when interpreted as a cluster because the noise is, by design, not close to each other. Therefore we look at the silhouette scores of the individual clusters produced by each method. Looking at the results shown in Table 14 one should keep in mind that DBSCAN labelled 10% of observations as noise. According to these results, DBSCAN creates higher quality clusters as the silhouette score for clusters 1 and 2 are 0.01 and 0.12 higher compared to k-means.

Table 14: Table silhouette scores (95% confidence interval).

k-means		DBSCAN	
Cluster 1	Cluster 2	Cluster 1	Cluster 2
0.78(±0.00)	0.80(±0.00)	0.79(±0.00)	0.92(±0.00)

5.2.2 Noise

We mentioned in the previous section that DBSCAN labelled 10% of observations as noise. As we do not control the data generating process, it is difficult to conclude that DBSCAN is more robust in this real-world scenario based on these results. Therefore, we re-run the methods on the data while adding between 0% and 40% artificial noise that simulates careless responding. The noise is modelled in accordance with the design in Section 5.1.2. The results are displayed in Table 15.

Table 15: Table silhouette scores (95% confidence interval).

	k-means		DBSCAN	
Noise	Cluster 1	Cluster 2	Cluster 1	Cluster 2
0%	0.78(±0.00)	0.80(±0.00)	0.79(±0.00)	0.92(±0.00)
10%	0.76(±0.00)	0.78(±0.00)	0.79(±0.00)	0.91(±0.00)
20%	0.75(±0.00)	0.76(±0.00)	0.77(±0.00)	0.89(±0.00)
30%	0.73(±0.00)	0.74(±0.00)	0.76(±0.00)	0.87(±0.00)
40%	0.71(±0.00)	0.73(±0.00)	0.75(±0.00)	0.86(±0.00)

The results show that k-means has a clear downtrend in silhouette score, losing between 0.01 and 0.02 for every 10% of noise added to the data. We see that cluster 1 goes from 0.78 at 0% noise to 0.71 at 40%, while cluster 2 goes from 0.80 to 0.73, losing 0.07 in both cases. DBSCAN goes from 0.79 to 0.75 in cluster 1 and from 0.92 to 0.86 in cluster 2, losing 0.04 and 0.06. Again we should note that DBSCAN classifies 10% to 19% of items as noise, which makes it difficult to compare absolute scores between the methods. However, we can compare the change in performance of each method when adding noise. The results indicate that DBSCAN has superior robustness, losing 0.03 less in cluster 1, and 0.01 in cluster 2 compared to k-means when adding 40% noise.

6 Conclusion

This paper investigates whether DBSCAN is a suitable method to use on rating scale data. We compare DBSCAN to the commonly used k-means clustering method in two simulation studies and one practical example. The first simulation assesses what dimensionality in the data is needed in order for DBSCAN to work and looks at its robustness against random responses. In the second simulation we look at the robustness of DBSCAN specifically to careless responding, which is the most common type of noise in rating scale data. Furthermore, we look into how the performance is affected when different assumptions are made on when a careless respondent should be labelled as noise. Finally, we look at the performance of DBSCAN on a practical example that aims to cluster political opinions in the United States.

The results of the first simulation show that k-means outperforms DBSCAN in every scenario when there is no noise in the data. DBSCAN performs poorly when using a small number of items and item scale size. Most notably, when using 1 valid construct and a 5-point scale. In that scenario, the ARI of DBSCAN is between 0.24 and 0.14 lower than k-means, depending on the number of items used. Increasing the size of the item scale and number of items improves the performance of DBSCAN and decreases the performance deficit to k-means, reaching a difference of 0.01 in terms of the ARI when using 7 items with an 11-point scale. When simulating with 5% noise, DBSCAN starts outperforming k-means when the data has at least 5 items with at least a 7-point scale. Moreover, the method also performs better on data with multiple constructs regardless of the number of items or item scale.

In the second simulation, we generate data based on an actual survey's size and correlation structure. Moreover, we use real careless responses sampled from this same survey as noise. The noise is modelled with different severity of careless responding, ranging from answering 17% to 83% of items carelessly. The results indicate that DBSCAN is more robust to this type of noise than k-means. There is no consensus in literature on the percentage of items a respondent has to answer carelessly to be labelled as noise (cut-off value). In our simulation study, we assess the effect of varying this cut-off value on the performance of DBSCAN. From the results, we conclude that DBSCAN is a suitable choice when the user assumes a cut-off value $\leq 50\%$, but gets outperformed by k-means when the cut-off value is $\geq 67\%$. Furthermore, DBSCAN performs best when using a cut-off value of 50%.

This research does not find conclusive evidence of DBSCAN performing better than k-means on real-world data. Based on the silhouette score of individual clusters, DBSCAN performs better. However, it is difficult to draw conclusions from these results as we have no control over the data generating process and comparing two different methods based on internal evaluation measures is not trivial.

The main finding of this paper is that DBSCAN can be a suitable method to use on rating scale data when the variability in the data is big enough and the user is suspecting the data to contain noise. We argue that the method can be used if the data consist of at least 5 items with at least a 7-point scale when using 1 construct and any number of items and scale when using multiple constructs. In these cases, the performance of DBSCAN is slightly worse than k-means when there is no noise but better when 5% or more noise is present. Furthermore, DBSCAN shows robustness to noise modelled as careless responding. Our simulation indicates that DBSCAN is most suitable when the user assumes a careless respondent should be labelled as noise when it responds to more than half of the items carelessly.

Further research could focus on the application of DBSCAN on empirical data. Our analysis gave indications of the usefulness of DBSCAN in practice. However, it lacked strong evidence of being a superior method to k-means. It would be useful to evaluate how accurate DBSCAN labels noise so its cluster performance can be compared fairly. A starting point could be the work of Niessen et al. (2016). The authors investigate several existing methods that aim to detect careless respondents. Further research could perform a study that compares the observations DBSCAN labels as noise to the results of the suggested detection methods.

However, Niessen et al. (2016) also mentions that all methods in their study have relatively low sensitivity. Therefore, ideally, a study could be performed where a questionnaire on a strongly opinionated topic is held, and part of the respondents are tasked to respond carelessly in various amounts, similar to the study of Schroeders et al. (2021).

Next to that, further research could investigate using different similarity measures. In this research, we assumed the data to be of interval scale. However, it is debatable whether this assumption is proper for the empirical data. Other distance measures that are suitable for ordinal data can be considered, like the general distance measure (GDM) introduced by Walesiak (1999) or the more recently introduced entropy-based distance measure by Zhang et al. (2019).

7 Limitations

One of the limitations of this research is the lack of relevant available data. We notice that data of more extensive surveys with a substantial number of questions and respondents are hardly available. This is likely because this type of data is primarily used internally, and relevant literature rarely publishes its data.

Secondly, it is difficult to compare the performance of the two different methods on real-world data. This is because we do not know how much noise there actually is in the data. This has implications for interpreting the internal performance metrics, in our case the silhouette score. The silhouette score is often used in literature but has the limitation that it is biased toward methods that form convex shaped clusters. Moreover, in the case of DBSCAN, it treats the observations labelled as noise as a cluster while it is actually not. This is an issue as the noise is by design not dense, and hence when seen as a cluster, it will get a very negative silhouette score. However, excluding them from the score is also debatable as it partly defeats the purpose of the metric and we do not know if the method correctly classified the observation as noise. There are alternatives like the density-based clustering validation (DBCV) metric introduced by Moulavi et al. (2014). Yet, there is no research that shows that this metric is suitable for the validation of methods that are not density-based. Nevertheless, we tried to compute the DBCV for our resulting clusters to give the reader an extra reference point. However, computation time of the original implementation[3] is very long, even after implementing improvements ourselves. Alternatively there is a more efficient implementation in the *hdbscan*[4] package. Unfortunately, we got different scores when testing the two implementations on the same clustering results.

Lastly, the results of DBSCAN in this study are based on parameters that are tuned with the silhouette score. This is common practice and reflects how one could determine the parameters in a real-world setting. However, other methods to determine the optimal parameter exist, and results will likely differ if another method is used.

[3]https://github.com/christopherjenness/DBCV

[4]https://github.com/scikit-learn-contrib/hdbscan

References

Baumgartner, H., & Steenkamp, J.-B. E. (2001). Response styles in marketing research: A cross-national investigation. *Journal of marketing research*, *38*(2), 143–156. https://doi.org/10.1509/jmkr.38.2.143.18840

Bowling, N. A., Gibson, A. M., Houpt, J. W., & Brower, C. K. (2021). Will the questions ever end? person-level increases in careless responding during questionnaire completion. *Organizational Research Methods*, *24*(4), 718–738. https://doi.org/10.1177/1094428120947794

Brühlmann, F., Petralito, S., Aeschbach, L. F., & Opwis, K. (2020). The quality of data collected online: An investigation of careless responding in a crowdsourced sample. *Methods in Psychology*, *2*, 100022. https://doi.org/10.1016/j.metip.2020.100022

Coretto, P., & Hennig, C. (2016). Robust improper maximum likelihood: Tuning, computation, and a comparison with other methods for robust gaussian clustering. *Journal of the American Statistical Association*, *111*(516), 1648–1659. https://doi.org/10.1080/01621459.2015.1100996

Credé, M. (2010). Random responding as a threat to the validity of effect size estimates in correlational research. *Educational and Psychological Measurement*, *70*(4), 596–612. https://doi.org/10.1177/0013164410366686

Cuesta-Albertos, J. A., Gordaliza, A., & Matrán, C. (1997). Trimmed *k*-means: An attempt to robustify quantizers. *The Annals of Statistics*, *25*(2), 553–576. https://doi.org/10.1214/aos/1031833664

Davé, R. N., & Krishnapuram, R. (1997). Robust clustering methods: A unified view. *IEEE Transactions on fuzzy systems*, *5*(2), 270–293. https://doi.org/10.1109/91.580801

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *kdd*, *96*(34), 226–231.

Figurska, M., Stańczyk, M., & Kulesza, K. (2008). Humans cannot consciously generate random numbers sequences: Polemic study. *Medical hypotheses*, *70*(1), 182–185. https://doi.org/10.1016/j.mehy.2007.06.038

Fritz, H., Garcia-Escudero, L. A., & Mayo-Iscar, A. (2012). Tclust: An r package for a trimming approach to cluster analysis. *Journal of Statistical Software*, *47*, 1–26. https://doi.org/10.18637/jss.v047.i12

Garcia-Escudero, L. A., Gordaliza, A., Matrán, C., & Mayo-Iscar, A. (2010). A review of robust clustering methods. *Advances in Data Analysis and Classification*, *4*(2), 89–109. https://doi.org/10.1007/s11634-010-0064-5

Guha, S., Rastogi, R., & Shim, K. (2000). Rock: A robust clustering algorithm for categorical attributes. *Information systems*, *25*(5), 345–366. https://doi.org/10.1016/S0306-4379(00)00022-3

Harpe, S. E. (2015). How to analyze likert and other rating scale data. *Currents in pharmacy teaching and learning*, *7*(6), 836–850. https://doi.org/10.1016/j.cptl.2015.08.001

Hartigan, J. A., & Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, *28*(1), 100–108. https://doi.org/10.2307/2346830

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*(1), 193–218. https://doi.org/10.1007/BF01908075

Jamieson, S. (2004). Likert scales: How to (ab) use them? *Medical education*, *38*(12), 1217–1218. https://doi.org/10.1111/j.1365-2929.2004.02012.x

Johnson, T., Kulesa, P., Cho, Y. I., & Shavitt, S. (2005). The relation between culture and response styles: Evidence from 19 countries. *Journal of Cross-cultural psychology*, *36*(2), 264–277. https://doi.org/10.1177/0022022104272905

Kodinariya, T. M., & Makwana, P. R. (2013). Review on determining number of cluster in k-means clustering. *International Journal*, *1*(6), 90–95.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology.*

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, *1*(14), 281–297.

Maniaci, M. R., & Rogge, R. D. (2014). Caring about carelessness: Participant inattention and its effects on research. *Journal of Research in Personality*, *48*, 61–83. https://doi.org/10.1016/j.jrp.2013.09.008

Moulavi, D., Jaskowiak, P. A., Campello, R. J., Zimek, A., & Sander, J. (2014). Density-based clustering validation. *Proceedings of the 2014 SIAM international conference on data mining*, 839–847. https://doi.org/10.1137/1.9781611973440.96

Niessen, A. S. M., Meijer, R. R., & Tendeiro, J. N. (2016). Detecting careless respondents in web-based questionnaires: Which method to use? *Journal of Research in Personality*, *63*, 1–11. https://doi.org/10.1016/j.jrp.2016.04.010

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Sander, J., Ester, M., Kriegel, H.-P., & Xu, X. (1998). Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data mining and knowledge discovery*, *2*(2), 169–194. https://doi.org/10.1023/A:1009745219419

Schoonees, P. C., Van de Velden, M., & Groenen, P. J. (2015). Constrained dual scaling for detecting response styles in categorical data. *psychometrika*, *80*(4), 968–994. https://doi.org/10.1007/s11336-015-9458-9

Schroeders, U., Schmidt, C., & Gnambs, T. (2021). Detecting careless responding in survey data using stochastic gradient boosting. *Educational and psychological measurement*, *82*(1), 29–56. https://doi.org/10.1177/00131644211004708

Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). Dbscan revisited, revisited: Why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, *42*(3), 1–21. https://doi.org/10.1145/3068335

Slonim, N., Aharoni, E., & Crammer, K. (2013). Hartigan's k-means vs. lloyd's k means–is it time for a change? *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI).*

Takagishi, M., van de Velden, M., & Yadohisa, H. (2019). Clustering preference data in the presence of response-style bias. *British Journal of Mathematical and Statistical Psychology*, *72*(3), 401–425. https://doi.org/10.1111/bmsp.12170

Van Vaerenbergh, Y., & Thomas, T. D. (2013). Response styles in survey research: A literature review of antecedents, consequences, and remedies. *International Journal of Public Opinion Research*, *25*(2), 195–217. https://doi.org/10.1093/ijpor/eds021

Walesiak, M. (1999). Distance measure for ordinal data. *No 2*(8), 167–173.

Wu, H., & Leung, S.-O. (2017). Can likert scales be treated as interval scales?—a simulation study. *Journal of Social Service Research*, *43*(4), 527–532. https://doi.org/10.1080/01488376.2017.1329775

Zhang, Y., & Cheung, Y.-m. (2020). An ordinal data clustering algorithm with automated distance learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(04), 6869–6876. https://doi.org/10.1609/aaai.v34i04.6168

Zhang, Y., Cheung, Y.-M., & Tan, K. C. (2019). A unified entropy-based distance metric for ordinal-and-nominal-attribute data clustering. *IEEE transactions on neural networks and learning systems*, *31*(1), 39–52. https://doi.org/10.1109/TNNLS.2019.2899381