A Survey of Clustering Methods for Heterogeneous Data

Christopher Daly
daly.ch@northeastern.edu
Northeastern University
Boston, MA, USA

ABSTRACT

When faced with a clustering problem, it's easy enough to find a package or library that works out of the box in R or Python, but it's not always clear if or why one algorithm is to be preferred over another. The problem is exacerbated in a mixed-data domain because there are fewer tools in general, and many more parameters and assumptions are in play, which, if violated, or estimated poorly, will make for an arbitrarily bad solution. Which ones work best, and what factors should be considered when choosing? A sampling of algorithms suitable for mixed data is examined.

1 INTRODUCTION

Clustering is a foundational problem in machine learning. It's generally an iterative process that seeks to identify data points that are more like each other than the rest of the population. Notions of similarity and distance lie at the heart of the various techniques used in clustering.

There are a variety of reasons one might turn to cluster analysis to understand a dataset. Increasingly, data is collated from a variety of sources and as such is increasingly heterogeneous (mixed) in nature. Given this reality, some questions arise immediately. Are all clustering methods equal? Why should I choose one over another? How many of these methods must I be conversant with to feel confident I'm getting the best results? What assumptions am I implicitly making when I choose an algorithm?

2 CLUSTERING BACKGROUND

At its root, clustering is the exercise of partitioning data according to some measure of association. It attempts to group the points together that are most similar and separate those examples that are least similar. How this is determined is a core element of any clustering algorithm, but becomes a thornier question in the mixed-data domain. What does similarity or distance mean when a feature can be continuous or categorical? If the distance metric compares continuous variables to continuous and categorical to categorical (as Gower does) how are those numbers combined at the end? Must they be weighted to avoid emphasising one group of features too heavily? And once these 'distances' are obtained, how are they used to obtain clusters?

Broadly speaking, the algorithms we look at can be grouped into distance-based or model-based methods. Distance-based methods (like PAM or HAC) use a distance calculation (like Gower) to calculate the similarity between any given examples. This stands in contrast to the model-based methods (like mixmod), which use a form of Gaussian mixture modeling to create a probabilistic model of the data, based on the assumption that the data is accurately modeled by a latent variable from a Multinomial distribution that in turn determines the Normal distribution that governs the continuous variables in that cluster.

3 DISCUSSION OF METHODS CONSIDERED

Because of the split nature of the data in the mixed domain, and the various ways available to compare, weight, and merge that data, there are a few different approaches employed by the algorithms in terms of how the data is parameterized. In certain cases, the data is split into continuous and categorical data frames, while in others, no distinction between categorical or continuous data is provided to the algorithm. In still other cases, a distance metric like Gower is employed, and the resulting dissimilarity matrix is provided directly. Gower was used in this study (the daisy method, and it accepts categorical and continuous columns in the same frame. (originally described in [Gower 1971])

K-Prototypes [Huang 1998] K-prototypes is a straight-forward implementation of k-means in the mixed data domain. It works by summing distances across categorical and continuous variables, using squared Euclidean distance for continuous variables and Hamming distance for categorical. The algorithm itself is capable of handling a mixed dataframe.

PAM (Partitioning around Medoids) K-Medoids was introduced by Kaufman et al [Kaufman and Rousseeuw 1987] It is quite similar to k-prototypes, but it introduces the idea of selecting representative examples as 'medoids', which differs from k-prototypes in that they are actual examples from data, as opposed to mean/mode constructs (centroids). PAM is capable of using arbitrary distance measures, but given the mixed domain in which we are operating, Gower was used here. PAM is the original implementation of the k-medoids method described when it was introduced. PAM also differs from k-prototypes in that it minimizes dissimilarity within the cluster. K-means/prototypes, on the other hand, minimizes total squared distance of all elements in a cluster from the centroid.

HAC (Hierarchical Ascendant Clustering: Hierarchical ascendant clustering an agglomerative hierarchical method. It starts with each example as its own cluster, and closest elements into the same clusters. It uses the Gower method for computing distance (because of the mixed data constraint) and it uses the Ward method to compute clusters. It's computationally intensive, and not suitable for large datasets. It does have the advantage of being able to deal with data in non-spherical clusters, as well as producing results in nicely readable dendograms that can be partitioned into arbitrary clusters.

Kamila [Foss and Markatou 2018] We're considering Kamila as a model method, but it's really sort of a hybrid of distance and model-based methods. The authors of the Kamila paper observe that the shortfalls of clustering arise in the partitional case because it is difficult to know how to correctly weight the contribution of

the continuous vs the categorical variables, and in the probabilistic case, the performance suffers if the underlying assumptions of distribution are violated. The Kamila method finds its way around the weights issue by using a KDE (kernel density estimate) and assuming spherical (generalizable to elliptical) clusters where the density is dependent only on the distance from a mean vector ([Foss and Markatou 2018] 3.2). Kamila is computationally expensive, but its calculation of the KDE for continuous variables allows it to relax the probabilistic assumptions associated with mixture models.

Mixmod([Lebret et al. 2015]) This method assumes that the data can be modeled with a latent variable assumed to be from a multinomial distribution, which in turn determines membership in a particular normal distribution. An EM process is followed to determine the appropriate parameters.

LCM (Latent Class Modeling) [Marbac and Sedki 2018] LCM (implemented in the VarSelLCM package in R) is an EM-based algorithm that is nearly identical to mixmod, excepting its variable and model selection capabilities. While extremely interesting, the selection features (which are implemented using information criterion like BIC, MICL, etc), are prohibitively long-running, at least for the large datasets examined here. The algorithm applies the selected information criteria to select from amongst the most relevant features and best-performing cluster sizes.

4 DATA

This paper was inspired by "Head-to-head comparison of clustering methods for heterogeneous data: a simulation-driven benchmark" [Preud'homme 2021]. The data generated by the authors is available online (https://mbi.loria.fr/clustering-of-mixed-data/). It was carefully crafted to provide representative continuous and categorical values, given a 'true' clustering. Thus, in each experiment, the predicted clusters could be compared to the actual. The data is grouped along design themes (see Fig. 1). For each theme, three iterations are performed, each modifying the chosen value. Each of these iterations contains a 1000 datasets.

(1) n (population size) varies
(2) k (cluster size) varies
(3) Relevance of continuous features varies
(4) Relevance of categorical features varies
(5) Number of continuous features varies while categorical features held constant at 2
(6) Number of continuous features varies while categorical features held constant at 4
(7) Number of continuous features varies while categorical features held constant at 8

Figure 1: Test numbers as used in plots

Figure 2 (taken from [Preud'homme 2021]) shows how the parameters were varied for each dataset. The basic testing themes are enumerated in Fig 1.

	1	2	3	4	5	6	7
General parameters							
Population size	**300/600/1200**	300	300	300	300	300	300
Number of clusters	6	**2/6/10**	6	6	6	6	6
Continuous variables							
Total number	4	4	**2/4/8**	4	4	4	10
Proportion of relevant variables	100%	100%	100%	100%	100%	100%	**20%/50%/90%**
Degree of relevance	Mild	Mild	Mild	**Low/mild/high**	Mild	Mild	Mild
Categorical variables							
Total number	4	4	**2 / 4 / 8**	4	4	10	4
Proportion of relevant variables	100%	100%	100%	100%	100%	**20%/50%/90%**	100%
Degree of relevance	Mild	Mild	Mild	Mild	**Low/mild/high**	mild	Mild

Table 2. Parameters of the simulated populations according to each tested scenario.

Figure 2: Table from [Preud'homme 2021] detailing how features varied by test

5 RESULTS

ARI is a metric used to compare clustering quality. 0 indicates a level of similarity that would be found by chance, while 1 indicates complete agreement.

Top Performers: Kamila, mixmod, and LCM were the top performers. Given that this test did not involve model or feature selection, LCM was prohibitively long-running. Its results were not significantly different than the other model-based methods, but its execution time (see Figure 10) was disqualifying. Kamila performed the best, but had the second longest running time. In Kamila's case at least, it was still broadly comparable to the others (as opposed to LCM). Still, it bears mentioning that Mixmod performed almost as well, and had the second fastest running time of all methods tested.

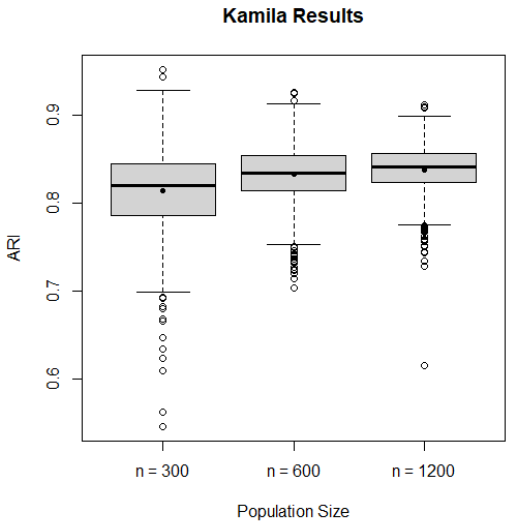

Figure 3: Kamila performed very well

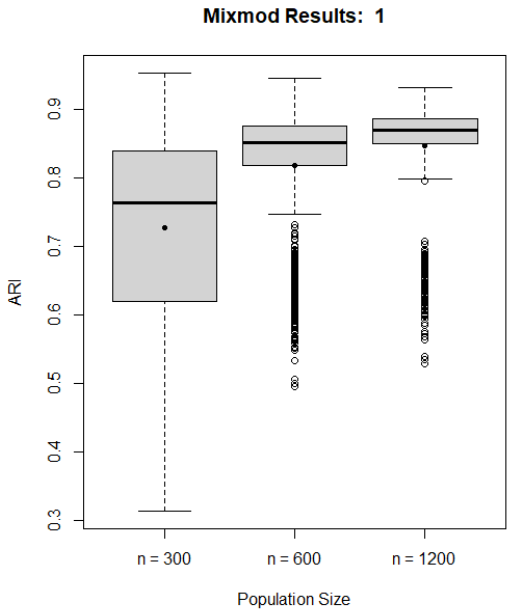

Figure 4: Mixmod performed only slightly worse than Kamila, but much faster

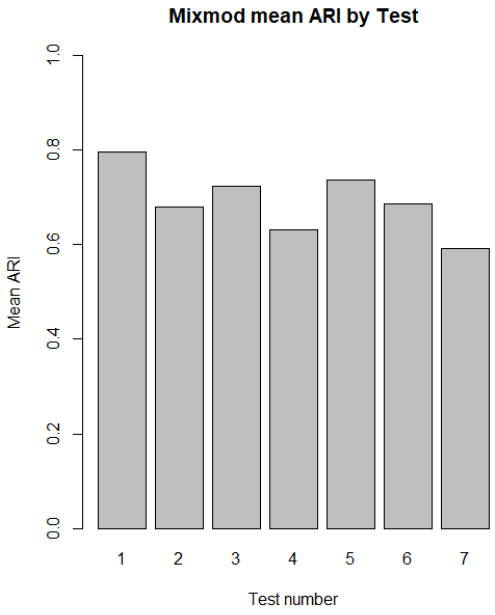

Figure 6: Mixmod was never far behind Kamila

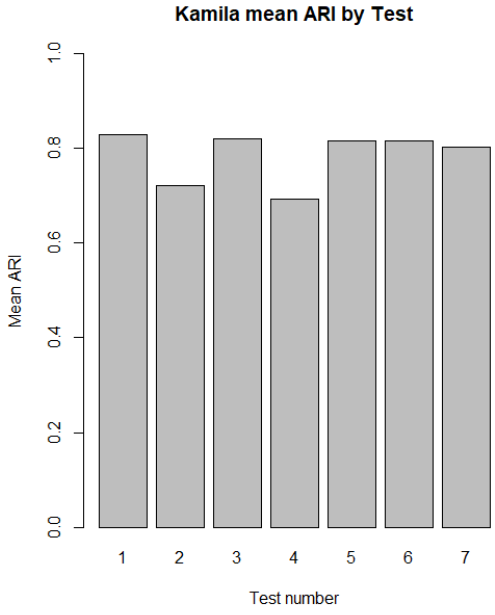

Figure 5: Kamila was a top performer across all tests

The mixture models were the clear winners. This is true to the point that there is no need to consider an alternative unless you find yourself in a situation where your data is particularly well-suited for another algorithm. Perhaps the dendogram produced by HAC is particularly useful for you, or maybe you know your data doesn't conform to the probabilistic assumptions implicit in mixture modeling. With enough time, you might choose to use Mixmod to obtain a baseline, and try to better it with Kamila.

For those in a hurry, use Mixmod. If you have a bit more time, Kamila will probably perform a little better.

Poor Performers: PAM and HAC were the worst performers.

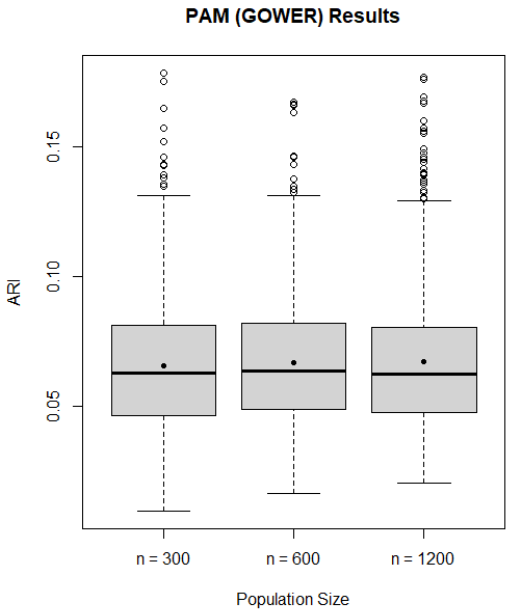

Figure 7: PAM

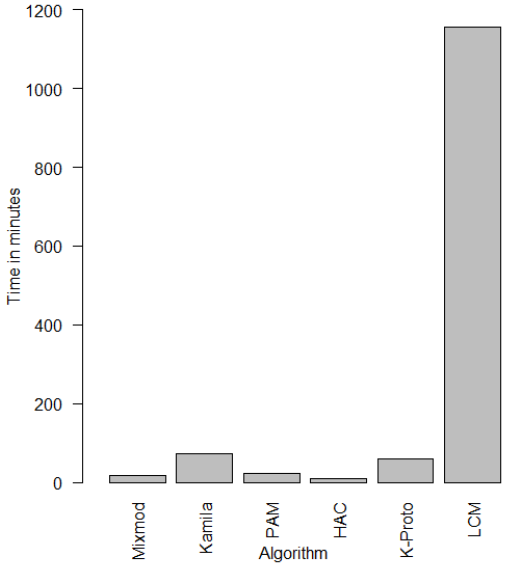

Figure 10: Execution time in minutes

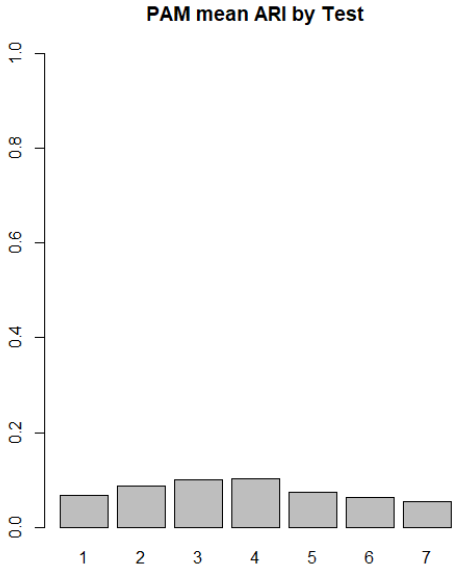

Figure 8: PAM performed surprisingly poorly

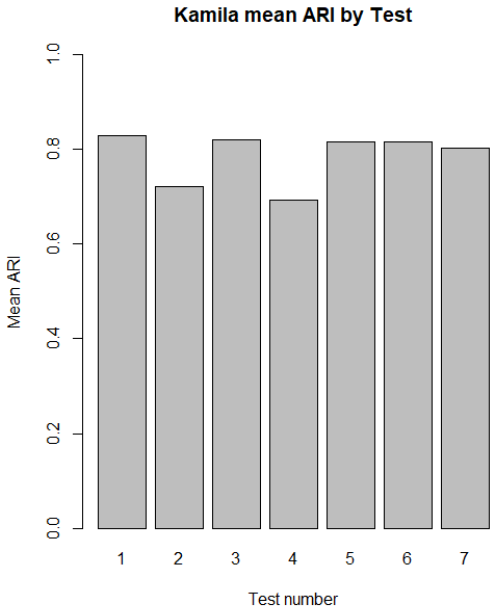

Figure 9: Kamila performed very well

6 CONCLUSION

The results themselves cluster into two groups, with Mixmod, Kamila, K-Proto, and LCM performing pretty well on the one hand, and PAM and HAC performing fairly atrociously on the other

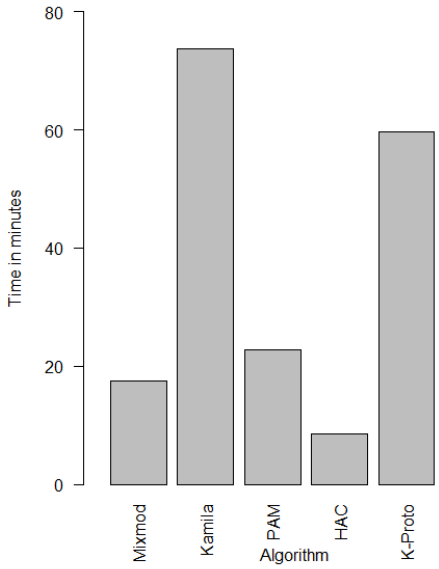

Figure 11: Execution time in minutes (without LCM)

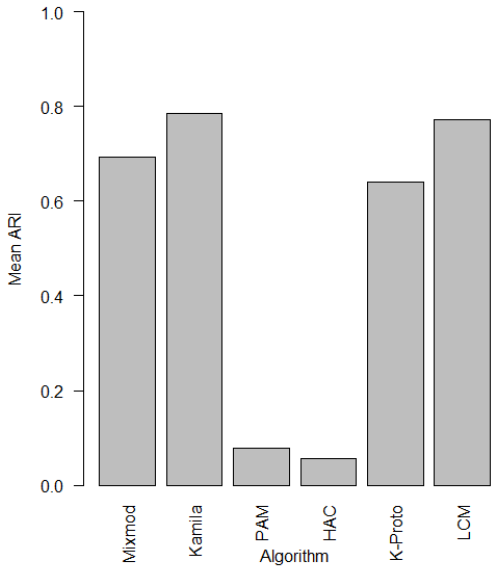

Figure 12: Overall performance by algorithm, across all tests

(Gower distance appearing to be the commonality there). Within the top performers, Kamila was the clear winner, but it was also the second worst in terms of running time. Mixmod, on the other hand, was the worst of that group, but had the second best running time. In our testing, Kamila came in with a mean ARI of about .79 to mixmod's .69. For your data, that might be enough to make a decision, but maybe the fact that kamila takes four times as long to run might militate in favor of at least a first run with mixmod to establish a baseline. It might even be that the results are good enough to stop there.

It was a bit surprising just how one sided the results were. The nuances of the performance of the various techniques are suitably subtle that more exploration is merited to determine if otherwise underperforming models might find their moment in the sun in certain niche cases. For instance, hierarchical ascendant clustering would certainly do much better in a sparse dataset with relatively few examples when compared to features. (verify, cite)

PAM and HAC were very ineffective, but for the distance models, K-Prototypes deserves special mention. It performed nearly as well as model-based methods across the board, and was relatively quick to execute. Depending on the use case, running K-Prototypes before running a more intensive model-based algorithm might be a really good idea. If you know your requirements ahead of time, it might be that the answer you get from an initial run of K-Prototypes is ' good enough'. It's certainly possible you could do better with a different method, but if the answer you have is sufficient, why bother?

The model-based methods (including Kamila) performed much better, with the perhaps most significant distinguishing factor being execution time. Mixmod is very fast, but doesn't perform as well as Kamila, which takes much longer to execute. LCM performed well enough, but given the extreme execution time, it's probably not worth it unless you're using it for its model/feature selection abilities.

REFERENCES

Alexander H. Foss and Marianthi Markatou. 2018. kamila: Clustering Mixed-Type Data in R and Hadoop. *Journal of Statistical Software* 83, 13 (2018), 1–44. https://doi.org/10.18637/jss.v083.i13

John C. Gower. 1971. A General Coefficient of Similarity and Some of Its Properties. *Biometrics* 27 (1971), 857.

Joshua Zhexue Huang. 1998. Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Mining and Knowledge Discovery* 2 (1998), 283–304.

Leonard Kaufman and Peter J. Rousseeuw. 1987. Clustering by means of medoids. , 405–416 pages.

Rémi Lebret, Serge Iovleff, Florent Langrognet, Christophe Biernacki, Gilles Celeux, and Gérard Govaert. 2015. Rmixmod: The R Package of the Model-Based Unsupervised, Supervised, and Semi-Supervised Classification Mixmod Library. *Journal of Statistical Software* 67, 6 (2015), 1–29. https://doi.org/10.18637/jss.v067.i06

Matthieu Marbac and Mohammed Sedki. 2018. VarSelLCM: an R/C++ package for variable selection in model-based clustering of mixed-data with missing values. *Bioinformatics* 35, 7 (09 2018), 1255–1257. https://doi.org/10.1093/bioinformatics/bty786 arXiv:https://academic.oup.com/bioinformatics/article-pdf/35/7/1255/48967518/bioinformatics_35_7_1255.pdf

Duarte K. Dalleau K. et al. Preud'homme, G. 2021. Head-to-head comparison of clustering methods for heterogeneous data: a simulation-driven benchmark. *Nature Scientific Reports* (2021). https://doi.org/10.1038/s41598-021-83340-8

7 ACKNOWLEDGEMENTS

Thank you to Professor Uzair Ahmad for his guidance and patience.

A APPENDIX

A.1 Code

The code used for this paper can be found at https://github.com/c-daly/MixedClusteringSurvey. This includes plots, which are located in the plots directory.

A.2 Selected Cluster Visualizations (using T-SNE)

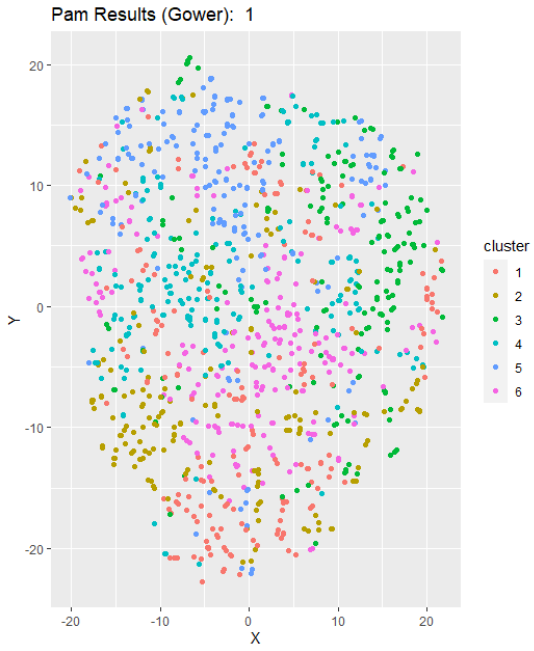

Figure 13: PAM was really just terrible

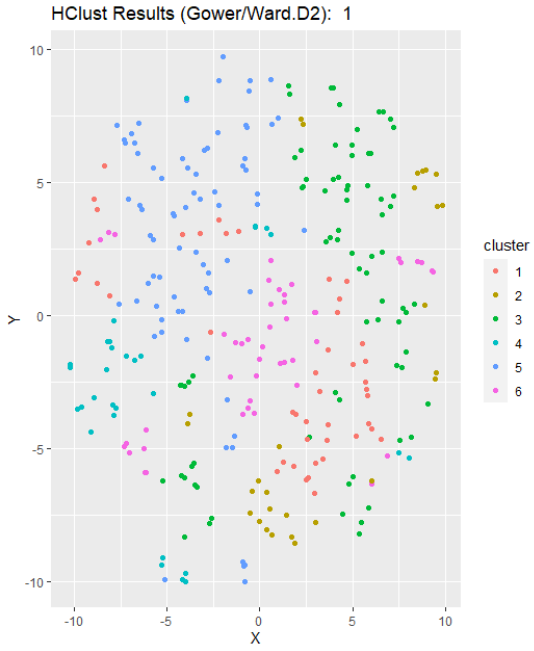

Figure 14: So was HAC

Figure 15: Kamila was a top performer

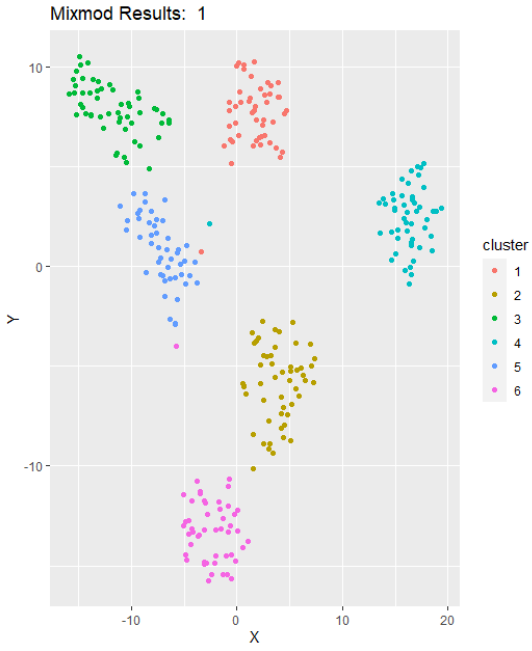

Figure 16: Mixmod also performed very well