Clustering of categorical and numerical data

Lukáš Janásek and Luu Danh Tiep

March 2020

1 Introduction

The aim of this project is to replicate a study conducted in paper 'Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values' Huang 1998. In the paper, the author describes the clustering of categorical and numerical data and suggests a new method for clustering mixed data - data with both categorical and numerical variables. In our project, we aim to replicate the clustering analysis. The paper used two data sets, soy bean and credit approval to demonstrate k-modes and k-proto algorithm respectively. For both algorithms, the goal is to evaluate their clustering 'performance' with respect to their initialization method. Both algorithms converge to local minimum and they are thus dependent on their initialization. Since we know the original class of each observation, we measure the 'performance' as accuracy of the clusters with respect to the real classes given in the data. For k-proto algorithm we also investigate the effect of parameter γ that balances distances of categorical and numerical variables on the accuracy. To be able to replicate the results, we have written our own functions replicating the two algorithms described in the paper - 'k-modes' (clustering of only categorical data) and 'k-proto' (clustering of mixed-type data). Consequently, we use the functions on the two data sets (soybean and credit) used by the authors.

The structure of our project is as follows: Firstly, we describe the algorithms used in the paper. Secondly, we describe the data. Thirdly, we replicate the results of the paper. In the first part, we apply the already available functions 'klar::k-modes' and 'clustMixType::kproto' without possibility to determine the initialization method. In the second part, we use our two manually written functions with both initialization methods following the analysis in the paper. [1]

2 Description of algorithms

2.1 k-modes

The k-modes algorithm is designed for clustering data with only categorical variables. The algorithm is very similar to k-means algorithm except for the modification of dissimilarity measure. Instead of Euclidean distance the algorithm uses Hamming distance as a measure of dissimilarity. The Hamming distance for two observations is defined as a number of components for which the categories between the two observations differ. Formally:

$$d(X, Y) = \sum_{i=1}^{n} \delta(x_i, y_i)$$

where $\delta(x_i, y_i) = 0$ if $x_i = y_i$ and 1 otherwise.

The algorithm is given in the following steps:

1. Select k data points each representing one of the k centers.
2. Each data point assign to the closest cluster center based on the Hamming distance.
3. Update the center of clusters. The new center of the cluster is given as a mode of all data points belonging to the cluster in each dimension.

[1] Please refer to the github link for the functions (Janásek 2020)

4. Repeat steps 2 and 3 until there is no data point changing its cluster.

As the author states, the algorithm is sensitive to the initialization of clusters at the beginning of the algorithm. Hence, we also allowed for two possible methods of initialization.

1. Select first k distinct data points
2. For each dimension of observations (column in the dataset) select up to the k most frequent classes. Then assign the classes to clusters randomly such that there are no same clusters. Next, select data points that are closest to the clusters and use the data points as a new center of the cluster.

The k-modes algorithm is conducted by our first function 'kmodes_fit'. The function takes as an input dataset with categorical variables (it converts all variables to characters), the number of clusters, one of the initialization methods, and also the maximal number of iterations to control for the execution time of the algorithm. The function returns a list containing a vector of cluster labels (from 1 up to the number of clusters) and a matrix of cluster centers (each row representing one cluster center). We also provided 'kmodes_predict' function that assigns labels to a new dataset based on the Hamming distance from already fitted centers of clusters and 'kmodes_measures' which returns within sum of square, total sum of squares and Calinski-Harabasz index.

2.2 k-proto

K-proto algorithm is based on the same steps as the k-modes algorithm except for the distance measure. Since the data contain both numerical and categorical feature, the resulting distance measure is a mixture of squared Euclidean and Hamming distance. For k numerical attributes $A_1^n, A_2^n, ..., A_k^n$ and l categorical attributes $A_{k+1}^c, A_{k+2}^c, ..., A_{k+l}^c$ of variables X and Y the author suggests a form:

$$d_{mixed}(X,Y) = \sum_{i=1}^{k}(x_i - y_i)^2 + \gamma \sum_{j=k+1}^{l+k} \delta(x_j, y_j)$$

where γ is a weight set to balance the two distance types. The choice of γ depends on the scaling of numerical variables and the size of dimension of data. We followed the author's choices of γ in the following analysis. The steps of the algorithm are

1. Select k data points each representing one of k clusters center
2. Each data point assigned to the closest cluster center based on the mixed distance.
3. Update the center of clusters. The new center of the cluster is given as a mode of all data points belonging to the cluster in each categorical attribute and a sample mean of each numerical attribute.
4. Repeat steps 2 and 3 until there is no data point changing its cluster.

K-proto algorithm is conducted by our function 'kproto_fit'. The function takes as an input dataset with categorical variables (characters) and numerical variables, number of clusters, one of the initialization methods, γ, and maximal number of iterations. The function returns a list containing a vector of cluster labels (from 1 up to the number of clusters) and a matrix of cluster centers (each row representing one cluster center). We also provided 'kproto_predict' function that assigns labels to a new dataset based on the mixed distance and γ (that must be the same as during fitting the clusters) from already fitted centers of clusters and also 'kproto_measures' returning within sum of square, total sum of squares and Calinski-Harabasz index.

3 Dataset description

In this paper, we are using two real-world data set.

3.1 Soybean disease data set

The first data set is the soybean data set. It has been frequently used to test conceptual clustering algorithms that have been done in the past Michalski and Stepp 1983; Fisher 1987. This data set was chosen as all its value can be considered as categorical variables. The data set we are working with have been updated and consists of 683 observations with 35 attributes and 1 class column. We cleaned the data so that we omit observations with NA values in some of the rows. We also select four chosen soy classes, that were also inspected in the previous work and we obtained 80 observations with each soy class being represented by 20 observations. We then investigate the uniqueness of each attribute and removed attributes, that have only one unique value. As a result, we obtained 80 observations with 21 variables to work with. This data set will be used for k-modes clustering.

3.2 Credit approval data set

The second data set is the credit approval data set. This data set has been used in the work of Quinlan. This data has 690 observations with 15 attributes and 1 class column. Removing NA values, the data set has 653 observations. We also inspected the uniqueness of 15 attributes and we see that all of them have at least 2 unique values, therefore the data set that will be used for k-proto clustering has 653 observations with 15 attributes. The numerical values of the dataset were also re-scaled.

4 Replication results

The study Huang 1998 conducted the research as follows: From the base data set, 100 data sets were created by reordering the items. Then, the last column, which indicated the class was removed and such data frame were parsed into the function. Then the accuracy was calculated as the fraction of correctly clustered observations divided by the total number of observations. To assign each cluster with its respective class, we used the Hungarian algorithm, which will provide us with the highest accuracy measure.

4.1 Replication using provided packages

4.1.1 Soy Bean Data Set

We used klar::kmodes [2] provided to replicate the paper. However, we were unable to do it as there is no possibility to select the initialization method, that the function shall follow. However, we could still conduct some clustering.

We used the plot of total within difference to inspect the optimal number of clusters. In our case, the number should be 4 as we have 4 disease classes. According to the elbow plot (see Figure 1, the flex point is not clear but we can see that the most significant flex point is at point 6, which is slightly off from the actual 4 clusters. This might mean that there exist diseases, which have quite significant different attributes even within one disease itself.

Running the 100 randomly reordered data sets, we obtained the histogram of the accuracy depicted in fig. 2. From the histogram, we can see that most of the accuracy lies between 0.90 and 0.95, and the second most lies between 0.60 and 0.65. All of them are above 0.25, which is a random guess probability of accuracy. The mean value is 0.82. The source of the variance of the accuracy is the fact that that algorithm converges always to the local optimum.

As mentioned above, using the klaR::kmodes function, we are unable to distinguish the method used thus we are not able to fully replicate the results. This process, however, can be done with the manually written function **kmodes_fit** as we will further show in subsubsection 4.2.1.

4.1.2 Credit Approval Data Set

We tried to use clustMixType::kproto [3] provided to replicate the paper. However, similarly to k-modes, we were unable to fully replicate the paper as we were unable to select the method chosen. Thus we were reliant

[2] This notation means that the function name is kmodes from the package klar

[3] This notation means that the function name is kproto from the packge clustMixType

Figure 1: Elbow plot to determine the optimal number of modes

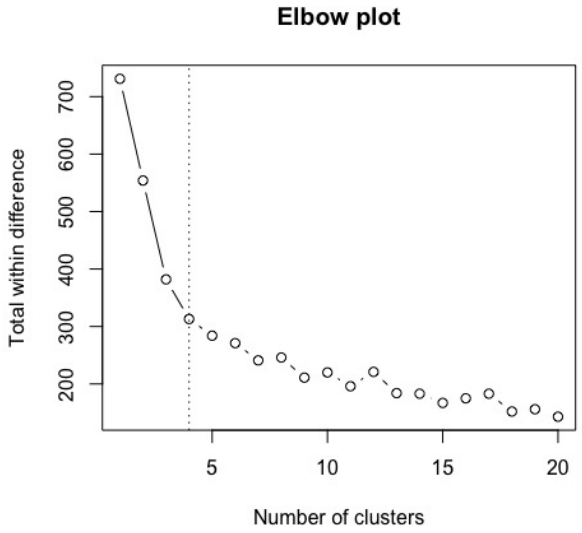


Figure 2: Histogram of accuracy using klaR::kmodes

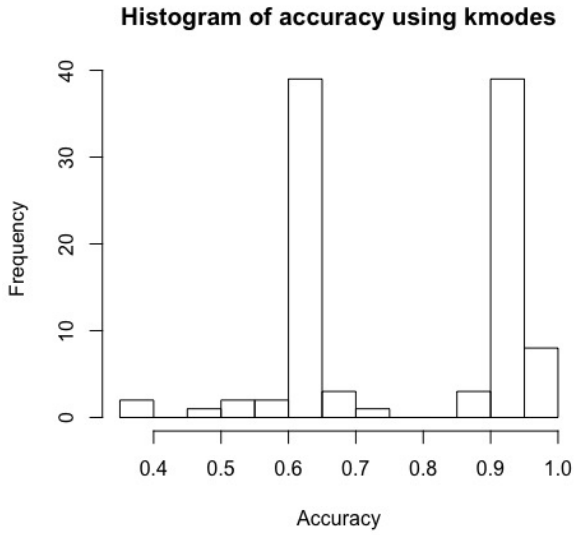


Table 1: Number of results with given Accuracy and CH

Accuracy	CH
1	104(2)
0.99	
0.98	
0.97	
0.96	104(1)
0.95	102(3), 109(7), 110(1)
0.94	104(1), 105(1), 108(3), 110(8)
0.93	103(1), 104(2), 104(2)
0.92	
0.91	103(1), 104(2), 105(2)
0.90	102)3)
≤ 0.89	26 - 83 (58)

on which method the package specified for us.

Using the provided setup, we obtain following results with respect to the chosen γ: The highest value of the accuracy obtained is 0.83. It seems that the accuracy is constantly getting visibly higher with each gamma added according to Table 2, which indicates that the dataset is dominated by categorical variables.

Table 2: Number of results with given Accuracies with respect to γ using clustMixType::kproto

	Accuracy	0.5	0.7	0.9	1	1.1	1.2	1.3	1.4
1	0.83								
2	0.82								
3	0.81								
4	0.80								5
5	0.79								49
6	0.78					2		26	20
7	0.77					2	13	70	25
8	0.76					9	58		
9	0.75				41	75	24		
10	0.74				22	5			
11	0.73			75	24		2		
12	0.72		13	19		4			
13	≤ 0.71	100	87	6	13	3	3	4	1

4.2 Replication using manually written functions

As mentioned in subsubsection 4.1.1 and subsubsection 4.1.2, in both cases provided packages did not have the option to choose which initialization method to use. By manually writing our own function, we are able to select which initialization method we want to use as described in section 2. Therefore we are able to fully replicate the results and do a comparison. The process is as follows: Running the written function assigns each observation to a cluster. Then using the Hungarian algorithm, each cluster is assigned to a disease so that the accuracy measure (which is the number of correctly clustered observation divided by the number of total observation) is maximized.

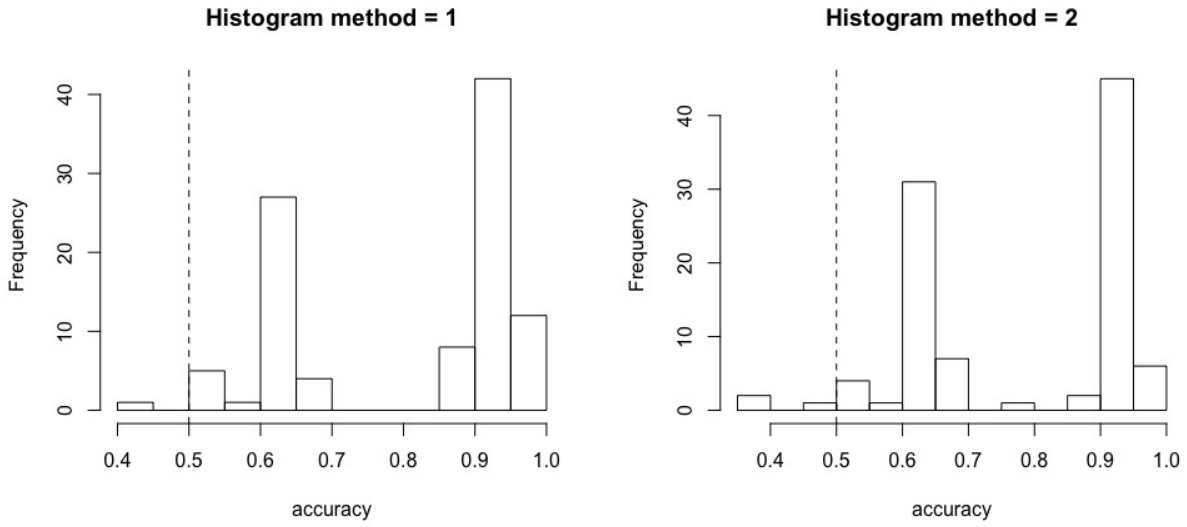
4.2.1 Soy Bean Data Set

We see that both methods yield similar results, however, the accuracy results from the method 1 still edges the method 2 a bit as its mean is higher and its minimal value did not dip so low.

Table 3: Comparison of results with distinct method while using manually written function on soy bean data

	Method = 1	Method = 2
Min.	0.438	0.360
1st Qu.	0.634	0.620
Median	0.912	0.910
Mean	0.816	0.787
3rd Qu.	0.938	0.940
Max.	1	1

Figure 3: Comparison of histogram of accuracy using manually written function on soy bean data



(a) Histogram using manual::kmodes_fit method = 1

(b) Histogram using manual::kmodes_fit method = 2

Evaluating the accuracy distribution Table 4 and histogram Figure 3, we can deduce that the second method is slightly superior.

Instead of using the costs of the clustering for comparison, we are using Calinski-Habaresz Index (CH) which is defined as

$$CH = \frac{n-k}{k-1} \times \frac{TSS - WSS}{WSS}$$

and a good clustering is going to maximize this index. As we see in Table 5, the lower accuracy range has lower CH index than the high accuracy range, which is a good indicator. In comparison between the two method, we might conclude that method 2 is slightly better as it is maximizing the CH index much more.

4.2.2 Credit Approval Data Set

In Table 6 and in Table 7, the accuracy is getting marginally better in the middle of the spectrum of the accuracy while at the top of the accuracy spectrum, it is stagnating or in some cases decreasing thus there

Table 4: Comparison of the accuracy distribution of manually written function on credit approval dataset

Accuracy	Method = 1	Method = 2
1	0	1
0.99	1	0
0.98	0	4
0.97	0	0
0.96	6	1
0.95	9	9
0.94	15	23
0.93	0	0
0.92	6	3
0.91	9	10
0.90	3	3
≤ 0.89	50	53

Table 5: Comparison of number of results with given Accuracy and CH of using manually written function on credit approval dataset

Accuracy	Method = 1	Method = 2
1		104(1)
0.99		
0.98		101(1)
0.97		
0.96	104(2)	101(1)
0.95	107(1), 109(1), 110(2)	101(1), 102(1), 103(1), 109(3), 110(3)
0.94	105(2), 108(5), 110(6)	103(4), 105(2), 108(8), 110(7)
0.93		
0.92	104(1), 105(1), 109(2)	108(1), 109(1)
0.91	103(4), 105(1)	101(1), 102(1), 103(5), 105(3)
0.90		
≤ 0.89	26 - 97(72)	26 - 83 (52)

is no clear distinction of the γ breakpoint. If we were to compare the overall accuracy of both methods, it is very hard to state which one perform better but one might say that the first method was slightly better as it was able to produce higher number of highly accurate clustering.

Table 6: Number of results with given accuracy with respect to γ using manually written function on credit approval dataset with method 1

	Accuracy	0	0.5	0.7	0.9	1	1.1	1.2	1.3	1.4
1	0.83			5	7	6	2	2	2	4
2	0.82			35	4	27	5	5	5	4
3	0.81		18	48	28		21	15	22	14
4	0.80		72		40	36	27	39	26	39
5	0.79		2		2	14	17	18	15	14
6	0.78									
7	0.77									
8	0.76									
9	0.75									2
10	0.74					1	1	1	4	1
11	0.73									
12	0.72									
13	≤ 0.71	100	8	12	19	16	27	20	26	22

Table 7: Number of results with given accuracy with respect to γ using manually written function on credit approval dataset with method 2

	Accuracy	0	0.5	0.7	0.9	1	1.1	1.2	1.3	1.4
1	0.83			2	4	3	3	4	4	6
2	0.82			33	2	22	3	3	1	
3	0.81		23	48	26		19	11	22	16
4	0.80		68	2	34	27	29	26	23	29
5	0.79		4		5	15	7	16	13	14
6	0.78									
7	0.77									
8	0.76									
9	0.75									1
10	0.74					1	2	2	4	3
11	0.73				1			1		
12	0.72									
13	≤ 0.71	100	5	15	28	32	37	37	33	31

4.3 Comparison of both results with the findings of the base paper

4.3.1 Soy Dataset

As we have came to conclusion in subsubsection 4.2.1, that method 2 is slightly superior to method 1 which is in line with what was found in the base paper (Huang 1998).

4.3.2 Credit Data Set

In comparison with the base paper, in our manual results, we were unable to determine the breakpoint γ (as shown in subsubsection 4.2.2), by which the accuracy was not increasing anymore. We see that after

substantial change in accuracy when $\gamma = 0.7$, the next γ values offered marginal increase (Table 6, Table 7) and in the out-of-the-box results, the accuracy seems to be still rising even after $\gamma = 1.4$ (Table 2). Meanwhile in the base paper, when $\gamma > 1.2$, the accuracy showed substantial accuracy decrease.

Using clustMixType::kproto, we were not able to reach the accuracy height as using manual::kproto_fit (0.83). Unlike the paper outlined, where for the highest accuracy value the second method was clearly superior, in our case, it was in favor of the first method in ratio 38:34.

4.3.3 Scalability test

As we were unable to obtain appropriate data set for the scalability test, which would allow us to reproduce the findings in this part, we have skipped this part, unfortunately.

5 Conclusion

In this work, we replicated the results of the Extensions to the k-means algorithm for clustering large data sets with categorical values (Huang 1998). We tried using solely packages that are provided freely (clustMixType and kmodes) however, they were insufficient as we were not able to set up the method of initial clustering, therefore we wrote our own function (kmodes_fit and kproto_fit) to help us replicating the results. Furthermore, we use Calinski-Habaresz Index to evaluate the performance of the kmodes function instead of using the cost function as it was described in the base paper.

Our findings regarding the soy bean dataset coincides with the findings of the base paper as we found that the second method is superior in terms of accuracy and cost to the first method.

Regarding the credit approval dataset, the conclusion is not clear cut as it was found in the base paper. The base paper found a distinctive γ, until which the accuracy was rising and then after which the accuracy was steadily decreasing. It was not like that in our case. Also we are unable to declare easily, which method is superior as both were very similar in terms of the accuracy.

We did not have a chance to replicate the scalability test as we were not able to obtain the same or suitable dataset for the analysis to be conducted. We would therefore recommend to replicate the test again in the further work.

References

[1] Zhexue Huang. "Extensions to the k-means algorithm for clustering large data sets with categorical values". In: *Data mining and knowledge discovery* 2.3 (1998), pp. 283–304.

[2] Lukáš Janásek. *Categorical and mixed data clustering*. 2020. URL: `https://github.com/OrangePiano/main`.

[3] Ryszard S Michalski and Robert E Stepp. "Learning from observation: Conceptual clustering". In: *Machine learning*. Springer, 1983, pp. 331–363.

[4] Douglas H Fisher. "Knowledge acquisition via incremental conceptual clustering". In: *Machine learning* 2.2 (1987), pp. 139–172.