Central European Journal of Operations Research
https://doi.org/10.1007/s10100-022-00824-2

ORIGINAL PAPER

A fair-multicluster approach to clustering of categorical data

Carlos Santos-Mangudo[1] · **Antonio J. Heras**[1]

Accepted: 12 October 2022

Abstract
In the last few years, the need of preventing classification biases due to race, gender, social status, etc. has increased the interest in designing fair clustering algorithms. The main idea is to ensure that the output of a cluster algorithm is not biased towards or against specific subgroups of the population. There is a growing specialized literature on this topic, dealing with the problem of clustering numerical data bases. Nevertheless, to our knowledge, there are no previous papers devoted to the problem of fair clustering of pure categorical attributes. In this paper, we show that the Multicluster methodology proposed by Santos and Heras (Interdiscip J Inf Knowl Manag 15:227–246, 2020. https://doi.org/10.28945/4643) for clustering categorical data, can be modified in order to increase the fairness of the clusters. Of course, there is a trade-off between fairness and efficiency, so that an increase in the fairness objective usually leads to a loss of classification efficiency. Yet it is possible to reach a reasonable compromise between these goals, since the methodology proposed by Santos and Heras (2020) can be easily adapted in order to get homogeneous and fair clusters.

Keywords Clustering · Fairness · Fair clustering · Categorical data

1 Introduction

Cluster Analysis can be defined as a set of techniques for finding homogeneous subsets (clusters) in a given dataset. The clusters should be homogeneous, in the sense that the elements within each subset should be quite similar to each other. Also, elements belonging to different clusters should be quite different. In other words, elements in

✉ Carlos Santos-Mangudo
casant01@ucm.es

Antonio J. Heras
aheras@ccee.ucm.es

[1] Financial and Actuarial Economics and Statistics Department, Complutense University of Madrid, Campus de Somosaguas, S/N, 28223 Pozuelo de Alarcón, Spain

Published online: 08 November 2022

the same cluster should show a high similarity, and elements belonging to different subsets should show a low similarity.

Different induction principles lead to a number of clustering techniques. According to Fraley and Raftery (1998), clustering techniques can be classified into hierarchical and partitioned methods. Han et al. (2012) classifies them into density-based, model-based and grid-based methods.

There is an extensive literature on this subject. Among the most frequently used algorithms for cluster analysis, we can mention, the CURE and ROCK algorithms (Guha et al. 2000, 2001), the K-Modes algorithm (Huang 1997a, b, 1998, 2009), the K-Prototypes algorithm (Huang 2005; Ji et al. 2020), the K-Means algorithm (McQueen 1967), the DBSCAN algorithm (Pietrzykowski 2017; Zhu et al. 2013) or the IDCUP algorithm (Altaf et al. 2020).

A data set can be considered as a matrix where the rows are the observations, individuals or elements, and the columns are the features, attributes or traits associated to these elements. Many well-known clustering algorithms, such as the popular K-Means, only work with numerical datasets, where all the attributes are numerically measured. K-Means (Forgy 1965; McQueen 1967) associates the clusters to their average values (centers of gravity) and assigns the elements to their nearest clusters; the algorithm then calculates the new centers of gravity and reallocates the elements to the new clusters. These steps shall be repeated until no more changes are observed or a maximum number of iterations is reached.

In some datasets, however, we find categorical data, with non-numerical attributes, and K-Means no longer works. The widely used K-Modes algorithm (Huang 1997a, b, 1998) is based on similar ideas and is adapted to work with categorical data. Instead of centers of gravity and Euclidean distances, K-Modes uses "centroids" defined from the modes of the categorical attributes, and measures of "dissimilarity" to quantify the distances between them.

The final results given by both K-Means and K-Modes often depend on the selection of the initial "seeds". Since this selection process usually involves some randomization scheme, the results can be instable, i.e. running an algorithm several times on the same dataset can lead to different final allocations. Some solutions for this problem have been suggested in the literature: see Ahmad and Dey (2007a, b), Cao et al. (2009), Gan et al. (2005), Jiang et al. (2016), Khan and Ahmad (2012, 2013, 2015), Ng and Wong (2002), Sajidha et al. (2018), Santos and Heras (2020). It is also worth mentioning K-Means++ (Arthur and Vassilvitskii 2007), an important variation of K-Means, that improves the running time of Lloyd's algorithm and the quality of the final solution. Moreover, it is implemented in most numerical packages, e.g.: scikit-learn or Matlab.

Besides the classification efficiency and the stability of the results, a new problem has received a lot of attention in the last years. Classification algorithms are increasingly applied to many important economic and social problems, such as prediction of criminal behaviour, screening of job applicants, mortgage approvals, marketing research or insurance rating, among many others. Human supervision of many decision making processes is progressively being replaced by automated data analysis, and there is a growing concern in our societies about the lack of human control of the outcomes.

For instance, an important potential problem is that the output of the algorithms could unreasonably harm or benefit some groups of people that share sensitive attributes, related to gender, race, religion, social status, etc. These discrimination problems are often unintended, due to the complexity of the algorithmic processing of huge amounts of data. As a consequence, the need to prevent these classification biases related to sensitive attributes has increased the interest in designing fair clustering algorithms. The meaning of "fairness" in this case is to ensure that the outputs of the algorithms are not biased towards or against specific subgroups of the population.

The literature on the issue of fair clustering is extensive: see, among others, Abraham et al. (2020), Chierichetti et al (2017), Chen et al. (2019), Esmaeili et al. (2020), Kleindessner et al. (2019) and Ziko et al. (2019). However, all these papers have studied the numerical case. To our knowledge, there are no previous papers devoted to the problem of fair clustering of pure categorical datasets.

In this paper, we put forward a modification of the Multicluster methodology proposed by Santos and Heras (2020) for clustering categorical data, in order to reach a compromise between fairness and classification efficiency. As we shall see, the output of the proposed algorithm combines a total stability with a high degree of fairness and efficiency.

The outline of the paper is as follows: in the first section (Fair clustering of categorical data) we give a brief description of the main ideas of the paper. In the second section (Methods) we explain how the fair clustering algorithm operates. In the third section (Experimental Results), several well-known real databases are used to illustrate the application of the methodology, showing good results in terms of clustering efficiency and fairness. Concluding remarks are presented in the last sections (Discussion and Conclusions).

2 Fair clustering of categorical data

Santos and Heras (2020) have proposed a new methodology for clustering categorical data, based on the so-called "multiclusters". Each multicluster is associated to a non-empty combination of the attributes of the data set, so that the objects belonging to it show a total coincidence in the values of their attributes. However, since the number of multiclusters may be excessive, it is often required to reduce it, in order to reach the desired (usually small) number of final clusters. For this purpose, the algorithm takes the biggest clusters as "seeds" and associates them to the smaller clusters, taking into account the similarities between their attributes. This way, those Multiclusters showing a great number of coincidences between their attributes will be eventually tied together, giving rise to greater clusters sharing many (not all) of their attributes. The process ends when the desired number of final clusters is reached.

In this paper we show that this clustering algorithm for categorical data can easily be adapted to getting not only efficient but also fair clusters. Following previous works on fair clustering for numerical data (Chierichetti et al. 2017), we assume a protected attribute in the database, such as gender or ethnicity.

Under the legal doctrine of Disparate Impact, a decision making process is considered discriminatory or unfair if it has a disproportionately adverse impact on the

protected classes (Barocas and Selbst 2016). Unlike the doctrine of Disparate Treatment, Disparate Impact is not concerned with intent or motivations, it only focuses on the outcomes. Under this doctrine, a clustering algorithm is fair if it leads to a set of fair clusters, and a cluster is fair if it has a proper representation of the values of the protected attribute: for instance, 50% males and 50% females, if the protected attribute is Gender. Notice, however, that the desired proportions of the values of that attribute are not necessarily identical in all cases: if the gender proportions in the dataset are highly unbalanced, forcing an equal representation of males and females in the final clusters may lead to unreasonable proportions of other attributes. For this reason, the desired proportions can also be defined as the proportions of the protected attribute in the dataset. If the gender ratio in the dataset is, for instance, 30–70%, then it should be the same or quite similar in the final clusters.

The Multicluster algorithm can be modified in order to increase the fairness of the obtained clusters. Of course, there is a trade-off between fairness and efficiency, so that, if we want to increase the fairness, we have to give up some classification efficiency. Yet it is possible to reach a reasonable compromise between these goals. The idea is to add a new step in the algorithm, in which we link two clusters when the distribution of the protected attribute after linking the clusters is closer to the desired distribution. This procedure is repeated until the desired number of clusters is reached.

3 Methods

In this section we explain how the "Fair Multicluster" algorithm for categorical data works. We assume the existence of a protected attribute in the data set, and also of desired ratios between its values. The goal of the algorithm is to split the total data base into a set of homogeneous and fair clusters: homogeneous, because each of them must contain only similar observations; and fair, because the proportions of the values of the protected attribute must be close to the desired proportions.

The algorithm works as follows:

Step 1

1. We identify the clusters for each single attribute with its different categorical values. For example, if a given attribute only has two values A and B, these are also the clusters associated to that criterion.
2. We merge all the possible single-attribute clusters in order to get the initial set of "Multiclusters". For example, if there are only two attributes with values A, B and C, D, E, respectively, then there will be six "Multiclusters": AC, AD, AE, BC, BD and BE. Notice that all the elements belonging to a given Multicluster show a total coincidence of the values of their attributes. This initial set of Multiclusters gives us the maximum number of clusters, which may be large. However, in real examples many of them are usually empty, so that the number of non-empty Multiclusters is much more reduced.

Step 2

1. For every couple of clusters, we compute the number of coincidences between their attributes. For example, the number of coincidences between AC and AD is one (A), and the number of coincidences between AC and BD is zero. This information is shown in the so-called Coincidence Matrix.
2. For every row of the Coincidence Matrix obtained before, select the column with the highest number of coincidences and merge the respective Multiclusters. The elements belonging to these new and bigger clusters share many (but not all) attributes. When two or more columns can be selected, we can break the tie by means of the Fleiss' Kappa coefficient (Fleiss et al. 1969, 2003; Fleiss 1971), a widely used measure of the degree of similarity between objects with categorical attributes. Notice that, if we have already compared cluster "A" and cluster "B", we don't need to further compare cluster "B" and cluster "A". For this reason, in this procedure we only need to work with the upper triangle of the matrix.

Step 3

1. We form a table with the optimal clusters obtained in the previous step, ranked in increasing order according to their size. For every row (cluster) of the table, we link it with other row (cluster) of the same table such that the resulting ratios of the values of the protected attribute are the closest to the desired ratios. This way, we obtain a new set of bigger clusters with a distribution of the protected attribute closer to the desired distribution.
2. We repeat the previous step until the predefined number of desired clusters is reached. The output of the algorithm is a set of clusters with a high degree of homogeneity and fairness.

To illustrate the methodology, the "German Credit" database from UCI Machine Learning Repository (Dua and Graff 2019) has been used as an unsupervised dataset; we work with a random sample of 20 observations and 9 categorical attributes, which we show in Table 1.

The first step of the algorithm is the calculation of the clusters for every single attribute, which correspond to their different values. Table 2 shows the distribution of clusters for each attribute, obtained in step 1.1. We choose Gender as protected attribute, with two values, Male (M) and Female (F). To ensure the reproducibility of the analysis, we rank the values of the attributes in increasing order according to their size.

In the step 1.2, we combine all the possible single-attribute clusters in order to get the initial set of multiclusters. The maximum number of multiclusters obtained this way can be very high, since it is the product of the number of clusters for every attribute of the dataset. In our case, the maximum number of multiclusters will be $4*4*7*5*5*4*2*3*4*4*4*3*3*3*4*2*2*1 = 464.486.400$. However, almost all of them are empty. Actually, there are only 20 nonempty multiclusters, which are shown in Table 3.

In Table 3, each multicluster contains only one single observation. To identify the multiclusters, we use the numbers associated to the values of the attributes in Table

Table 1 A sample of 20 observations from the German Credit dataset

Observation	Status account	Credit history	Purpose	Savings	Employment	Installment	Gender	Debtors	Residence	Property	Others	Housing	Credits	Job	Number people	Telephone	Foreign
775	A13	A34	A40	A63	A71	2	M	A101	4	A124	A141	A153	3	A171	1	A191	A201
204	A11	A32	A48	A61	A74	4	M	A101	4	A122	A143	A151	1	A173	1	A191	A201
699	A14	A34	A43	A61	A73	4	M	A101	2	A123	A143	A152	2	A173	1	A191	A201
250	A14	A32	A43	A61	A71	3	F	A102	4	A121	A143	A151	1	A173	1	A191	A201
593	A14	A32	A49	A64	A75	4	F	A101	4	A121	A141	A152	1	A172	1	A191	A201
661	A13	A32	A43	A61	A73	3	M	A101	4	A121	A143	A151	1	A173	1	A191	A201
463	A12	A32	A42	A61	A72	3	F	A101	1	A121	A143	A151	1	A174	1	A191	A201
828	A14	A30	A49	A61	A73	2	M	A101	2	A123	A142	A152	2	A173	2	A191	A201
845	A14	A32	A42	A61	A75	4	M	A101	4	A122	A141	A152	3	A173	2	A192	A201
11	A12	A32	A40	A61	A72	3	F	A101	1	A123	A143	A151	1	A173	1	A191	A201
896	A14	A33	A41	A65	A74	3	M	A101	2	A123	A142	A152	1	A174	2	A192	A201
379	A12	A32	A40	A61	A75	4	M	A101	2	A124	A143	A153	1	A174	1	A192	A201
989	A11	A32	A41	A61	A71	4	M	A101	2	A124	A143	A153	1	A174	1	A192	A201
323	A11	A32	A41	A61	A74	2	M	A101	1	A124	A143	A153	1	A174	1	A192	A201
716	A14	A34	A41	A65	A75	1	M	A101	4	A123	A143	A152	2	A173	1	A191	A201
619	A12	A32	A42	A62	A73	2	F	A102	4	A123	A143	A151	1	A173	1	A191	A201
908	A12	A32	A46	A65	A73	2	M	A101	2	A123	A143	A152	1	A173	1	A191	A201
955	A11	A32	A40	A61	A73	4	F	A103	4	A122	A143	A152	1	A173	1	A192	A201
209	A11	A32	A49	A61	A73	2	M	A101	2	A123	A142	A152	1	A172	1	A191	A201
304	A11	A34	A40	A61	A74	4	M	A102	3	A122	A143	A152	2	A173	1	A192	A201

Table 2 Cluster distribution of the attributes

Status account	Frecuency	Cluster
A13	2	1
A12	5	2
A11	6	3
A14	7	4

Credit history	Frecuency	Cluster
A30	1	1
A33	1	2
A34	4	3
A32	14	4

Purpose	Frecuency	Cluster
A46	1	1
A48	1	2
A42	3	3
A43	3	4
A49	3	5
A41	4	6
A40	5	7

Savings	Frecuency	Cluster
A62	1	1
A63	1	2
A64	1	3
A65	3	4
A61	14	5

Employment	Frecuency	Cluster
A72	2	1
A71	3	2
A74	4	3
A75	4	4
A73	7	5

Installment	Frecuency	Cluster
1	1	1
3	5	2
2	6	3
4	8	4

Gender	Frecuency	Cluster
F	6	1
M	14	2

Debtors	Frecuency	Cluster
A103	1	1
A102	3	2
A101	16	3

Residence	Frecuency	Cluster
3	1	1
1	3	2
2	7	3
4	9	4

Property	Frecuency	Cluster
A121	4	1
A122	4	2
A124	4	3
A123	8	4

Others	Frecuency	Cluster
A141	3	1
A142	3	2
A143	14	3

Housing	Frecuency	Cluster
A153	4	1
A151	6	2
A152	10	3

Credits	Frecuency	Cluster
3	2	1
2	4	2
1	14	3

Job	Frecuency	Cluster
A171	1	1
A172	2	2
A174	5	3
A173	12	4

Number people	Frecuency	Cluster
2	3	1
1	17	2

Telephone	Frecuency	Cluster
A192	7	1
A191	13	2

Table 3 20 nonempty multiclusters

Observation	Status account	Credit history	Purpose	Savings	Employment	Installment	Gender	Debtors	Residence	Property	Others	Housing	Credits	Job	Number people	Telephone	Multicluster
775	1	3	7	2	2	3	2	3	4	3	1	1	1	1	2	2	1372232343111122
204	3	4	2	5	3	4	2	3	4	2	3	2	3	4	2	2	3425342342323422
699	4	3	4	5	5	4	2	3	3	4	3	3	2	4	2	2	4345542334332422
250	4	4	4	5	2	2	1	2	4	1	3	2	3	4	2	2	4445221241323422
593	4	4	5	3	4	4	1	3	4	1	1	3	3	2	2	2	4453441341133222
661	1	4	4	5	5	2	2	3	4	1	3	2	3	4	2	2	1445522341323422
463	2	4	3	5	1	2	1	3	2	1	3	2	3	3	2	2	2435121321323322
828	4	1	5	5	5	3	2	3	3	4	2	3	2	4	1	2	4155532334232412
845	4	4	3	5	4	4	2	3	4	2	1	3	1	4	1	1	4435442342131411
11	2	4	7	5	1	2	1	3	2	4	3	2	3	4	2	2	2475121324323422
896	4	2	6	4	3	2	2	3	3	4	2	3	3	3	1	1	4264322334233311
379	2	4	7	5	4	4	2	3	3	3	3	1	3	3	2	1	2475442333313321
989	3	4	6	5	2	4	2	3	3	3	3	1	3	3	2	1	3465242333313321
323	3	4	6	5	3	3	2	3	2	3	3	1	3	3	2	1	3465332323313321
716	4	3	6	4	4	1	2	3	4	4	3	3	2	4	2	2	4364412344332422
619	2	4	3	1	5	3	1	2	4	4	3	2	3	4	2	2	2431531244323422
908	2	4	1	4	5	3	2	3	3	4	3	3	3	4	2	2	2414532334333422
955	3	4	7	5	5	4	1	1	4	2	3	3	3	4	2	1	3475541142333421
209	3	4	5	5	5	3	2	3	3	4	2	3	3	2	2	2	3455532334233222
304	3	3	7	5	3	4	2	2	1	2	3	3	2	4	2	1	3375342212332421

2. Notice that the variables in Table 1 have the original labeling given in the dataset German_Credit: for instance, the values of the attributes of the first observation (775) are labeled as A13 (for the attribute "Status Account"), A34 ("Credit History"), A40 ("Purpose"), etc. To simplify the notation, in Table 2 these labels are substituted by numbers: according to Table 2, A13 will be "1", A34 will be "3", A40 will be "7", etc. In Table 3 we label the Multiclusters with the numeric values attached to the values of their attributes in Table 2. Following this rule, the Multicluster containing observation 775, for example, will be labeled as "1372232343111122".

According to the information given in Table 3, we build the Coincidence Matrix (Table 4):

In order to reduce the number of clusters, we merge those Multiclusters that share the highest number of values of attributes. For each row, when there is only one column showing the highest value of coincidences, we merge the corresponding clusters. That is, we merge the clusters associated to that row and to the column corresponding to the highest value. This is the situation shown in Table 5, built from the second row of the Coincidence Matrix: in this case, the Multiclusters 1445522341323422 and 4445221241323422 should be merged, because they share the values of 12 attributes.

When there are several columns with the highest value, we break the tie by means of the Fleiss-Kappa coefficient (Fleiss et al. 1969, 2003; Fleiss 1971). For example, in Table 6, built from the first row of the Coincidence Matrix, we find five columns with 6 coincidences. In this case, the Multiclusters 1372232343111122 and 3465332323313321 should be merged, because they get the highest value of the Kappa-Fleiss coefficient (0.957525773195876). If there are several Multiclusters having the same highest Kappa concordance value, the first of them should be selected following the top-down methodology.

Once the process explained before has been executed for all rows included in the Coincidence Matrix (Table 4), we obtain the Optimal Multiclusters Table (Table 7), with 11 nonempty optimal Multiclusters. Of course, the final number of clusters could be less than 11, if desired. Further details about Step 2 of the algorithm can be found

Table 4 Coincidence Matrix between multiclusters

Multicluster	Freq	1372232343111122	1445522341323422	2414532334333422	2431531244323422	2435121321323322	2475121324323422	2475442333313321	3375342212332421	3425342342323422	3455532334233222	3465242333313321	3465332323313321	3475541142333421	4155532334232412	4264322334233311	4345542334332422	4364412344332422	4435442342131411	4445221241323422	4453441341133222
1372232343111122	1		6	5	4	3	4	6	4	5	5	6	6	3	4	2	5	6	5	4	5
1445522341323422	1			9	9	10	10	7	5	11	8	7	7	8	6	4	9	7	6	12	7
2414532334333422	1				10	7	9	8	5	8	11	7	7	7	9	7	10	9	5	6	6
2431531244323422	1					9	10	5	4	8	7	4	5	8	5	2	6	6	4	10	6
2435121321323322	1						13	8	3	8	6	7	8	6	3	4	5	4	4	10	7
2475121324323422	1							8	5	9	7	6	7	8	5	4	7	6	4	10	6
2475442333313321	1								7	8	7	13	11	8	4	6	7	5	7	5	6
3375342212332421	1									9	5	7	7	10	5	4	9	7	7	5	3
3425342342323422	1										8	9	9	10	5	4	8	7	8	9	7
3455532334233222	1											8	8	7	11	7	9	6	5	5	8
3465242333313321	1												13	8	4	7	7	5	6	6	5
3465332323313321	1													7	4	7	5	5	5	5	4
3475541142333421	1														4	3	7	5	8	8	7
4155532334232412	1															8	11	8	7	4	5
4264322334233311	1																6	7	6	3	4
4345542334332422	1																	11	7	7	6
4364412344332422	1																		7	6	7
4435442342131411	1																			5	8
4445221241323422	1																				8
4453441341133222	1																				17

Table 5 An example of multicluster association with only one coincidence

Multicluster	Freq	1372232343111122	1445522341323422	2414532334333422	2431531244323422	2435121321323322
				9	9	10
Multicluster	Freq	2475121324323422	2475442333313321	3375342212332421	3425342342323422	3455532334233222
		10	7	5	11	8
Multicluster	Freq	3465242333313321	3465332323313321	3475541142333421	4155532334232412	4264322334233311
		7	7	8	6	4
Multicluster	Freq	4345542334332422	4364412344332422	4435442342131411	4445221241323422	4453441341133222
		9	7	6	**12**	7

Table 6 An example of multicluster association with more than one coincidence

Multicluster	Freq	1372232343111122	1445522341323422	2414532334333422	2431531244323422	2435121321323322
1372232343111120	1		**6**	5	4	3
Multicluster	Freq	2475121324323422	2475442333313321	3375342212332421	3425342342323422	3455532334233222
1372232343111120	1	4	**6**	4	5	5
Multicluster	Freq	3465242333313321	3465332323313321	3475541142333421	4155532334232412	4264322334233311
1372232343111120	1	**6**	**6**	3	4	2
Multicluster	Freq	4345542334332422	4364412344332422	4435442342131411	4445221241323422	4453441341133222
1372232343111120	1	5	**6**	5	4	5

Table 7 Optimal multiclusters table

Optimal multiclusters	Frequency
3455532334233220	1
3465242333313320	1
4155532334232410	1
4345542334332420	1
2475121324323420	2
3465332323313320	2
4364412344332420	2
4435442342131410	2
4445221241323420	2
3475541142333420	3
4453441341133220	3

in Santos and Heras (2020). Notice that the clusters with equal frequency in Table 7 are lexicographically sorted.

In the last step of the algorithm (Step 3), we focus in the fairness objective. We have chosen as the desired ratio of the attribute Gender the relative initial proportions of its values in the dataset, 30% (for Female) and 70% (for Male). Then, for every row (multicluster) of Table 7 and beginning from the first one, we calculate the (Euclidean) distance between the observed ratios of the protected attribute and the desired ratios, after joining it to any of the other following rows (multiclusters). That is, for every row (the "Transmitter" multicluster), we select each one of the following rows (the "Receiver" multicluster), join the elements of both "Transmitter" and "Receiver" multiclusters to form a bigger cluster and calculate the (Euclidean) distance between the ratios of the protected attribute in the new bigger cluster and the desired ratios (30%, 70%). The process is repeated with all the following rows, and we finally join those rows (multiclusters) such that the ratios of the new cluster are closest to the desired ratios.[1]

For example, taking the second row in Table 7 (3465242333313320) as Transmitter, the minimum distance to the desired ratios (0.0471) is reached by joining it to the Receiver multicluster located in the eighth row (4435442342131410): see Table 8 for the details. Joining both rows in a new Table, the procedure is repeated until a predetermined number of clusters (k) is reached.

Table 9 shows the distribution of the protected attribute with two final clusters (k = 2), with a total fairness ratio[2] of 96%

[1] Of course, if the desired number of clusters is reached after step 2, then step 3 cannot be implemented. Nevertheless, this theoretical situation hardly occurs in practice, because in real world examples the number of rows of the Optimal Multicluster Table is usually much larger than the predefined desired number of clusters.

[2] The Fairness Ratio measures the distance between the desired distribution of the protected attribute and its final distribution; the formal definition is given below, in the section Evaluation Metrics.

Table 8 An example of calculation of the distances between a multicluster transmitter (2nd row of Table 7) and the receivers

Optimal Multicluster (Transmitter)
3465242333313320

Optimal Multiclusters (Receiver)	Distance
3465242333313320	na
4155532334232410	0,4243
4345542334332420	0,4243
2475121324323420	0,5185
3465332323313320	0,4243
4364412344332420	0,4243
4435442342131410	**0,0471**
4445221241323420	0,4243
3475541142333420	0,0707
4453441341133220	0,2828

Table 9 Observed and desired cluster distributions

Observed distribution "GENDER" (proposed algorithm)	Final clusters		Desired distribution "GENDER" (in dataset)	%
	1	2		
Total achieved per Cluster	98%	93%	Female	30
Overall total achieved	96%		Male	70

4 Experimental results

4.1 Datasets used for evaluation

Table 10 shows the categorical databases that are used for the evaluation of the clustering efficiency of the algorithm. In all cases there is a response variable, defined as the real cluster in which every observation is placed, which is known in advance but not used as an input of the algorithm. This omitted information can be used to evaluate the clustering efficiency, by contrasting the real classification of the observations to that given by the algorithm (see, among others, Yu et al. (2018), and Zhu and Ma (2018)).

As for the evaluation of the fairness of the classification, we measure the distance between the desired distribution of the protected attribute and its final distribution in the clusters given by the algorithm. In all the examples we choose the initial proportions of the values of the protected attribute in the data set as desired proportions to be approached in the final clusters. In other terms, the proportions of the values of the protected attribute in the final clusters (the output of the algorithm) should be close

Table 10 The datasets used in the experimental analysis

Name of dataset	Number of observations	Number of categorical attributes	Type of fairness attribute	Number of different values of Fairness attribute	Number of "k" final $clusters$[5]
$Absenteeism$[1]	8336	6	Gender	2	6
$BankMarketing$[3]	4521	11	Marital status	3	2
$CarsInsurance$[4]	3637	5	Gender	2	2
$GermanCreditFC1$[3]	1000	17	Gender	2	2
$GermanCreditFC2$[3]	1000	17	Marital status	3	2
$HumanResourcesFC1$[1]	292	19	Gender	2	4
$HumanResourcesFC2$[1]	292	19	Marital status	3	4
$HRIBM$[1]	1470	23	Gender	2	3
$CensusIncomeFC1$[2]	45,222	8	Gender	2	2
$CensusIncomeFC2$[2]	10,000	8	Race	4	2

[1] Kaggle Data Repository
[2] UCI Machine Learning Repository (Dua and Graff 2019)
[3] Machine Learning Data Repository
[4] Macquarie Australia University Data Repository
[5] This column shows the desired number (k) of final clusters selected by the decision-maker

to their initial observed proportions in the whole data set. Of course, any alternative desired distribution could be selected.

4.2 Evaluation metrics

Many measures of the degree of similarity between different partitions of the same data set have been proposed in the literature: see, among others, Dom (2012), Headden et al. (2008), Meilâ (2007), Reichart and Rappoport (2009), Rosenberg and Hirschberg (2007), Vinh et al. (2010), Wagner and Wagner (2007), Walker and Ringger (2008). We have selected four well-known measures of the similarity between two partitions P and R of a given data set. In our applications, P will be the output of the clustering algorithm, and R the "real" partition observed in the data set.

(I) "Fowlkes-Mallows index" (Fowlkes and Mallows 1983). High values of the Fowlkes–Mallows index indicate a great similarity between the clusters. It is defined as:

$$\boldsymbol{FMI} = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}$$

where

- TP as the number of pairs of points that are in the same cluster in both P and R.
- FP as the number of pairs of points that are in the same cluster in P but not in R
- FN as the number of pairs of points that are in the same cluster in R but not in P
- TN as the number of pairs of points that are in different clusters in both P and R

(II) "Maximum-Match Measure" (Meilâ and Heckerman 2001) is defined as

$$\boldsymbol{MMM} = \frac{1}{n}\sum_{i=1}^{k} max_j m_{ij}$$

where m_{ij} is the number of observations belonging to both clusters P_i and R_j and n is the total number of observations in the data set.

(III) "Normalized Variation of Information Measure" (Reichart and Rappoport 2009) is a normalized version of the VI-Variation of Information measure (Meila 2007); it is defined as:

$$\boldsymbol{NVI} = \begin{cases} \frac{H(P|R)+H(R|P)}{H(P)} & H(P) \neq 0 \\ H(R) & H(P) = 0 \end{cases}$$

where H(P) and H(R) are the entropies of the partitions P and R, and H(P|R) and H(R|P) are their conditional entropies.

"Overlap coefficient" (Vijaymeena and Kavitha 2016) also known as Szymkiewicz-Simpson coefficient, is a similarity measure based on the concept of the overlap between sets. Given two finite sets P and R, the overlap between them is defined as the size of the intersection divided by the smallest size of the two sets:

$$\boldsymbol{OI} = \frac{P \cap R}{\min(|P|, |R|)}$$

For the evaluation of fairness, we use the Euclidean distance between the desired distribution of the protected attribute and its final distribution in the clusters given by the algorithm:

$$Fairness\ ratio = \frac{\sum_{i=1}^{i=k}(1 - euclidean\ distance(Observed_i; Desired)}{number\ of\ clusters(k)}$$

4.3 Performance results

Table 11 shows the clustering efficiency of three algorithms (Multicluster, Fair-Multicluster and K-Modes) for the data sets in Table 10, measured by means of the Fowlkes-Mallows measure, the Maximum-Match measure, the Normalized Variation of Information measure and the Overlap measure. The highest performances are shown

Table 11 Comparison of classification efficiencies

Dataset	K-multicluster (Santos and Heras 2020)				Proposed algorithm fair-multicluster				K-modes			
	FMI	MMM	NVI	OI	FMI	MMM	NVI	OI	FMI	MMM	NVI	OI
Absenteeism	**0.657**	**0.655**	0.960	0.943	0.432	0.288	**0.972**	**0.957**	0.501	0.352	0.964	0.947
Bank Marketing	**0.657**	0.643	1.000	0.796	0.636	0.562	**1.000**	**0.796**	0.679	**0.711**	0.998	0.789
CARS_Insurance	**0.859**	**0.966**	**1.000**	**0.933**	0.732	0.801	**1.000**	0.789	0.643	0.617	0.999	0.787
German Credit FC1	0.546	0.601	0.997	0.576	**0.690**	**0.906**	**0.999**	**0.826**	0.643	0.804	0.994	0.699
German Credit FC2	**0.753**	**0.988**	**1.000**	**0.977**	0.715	0.936	**1.000**	0.881	0.543	0.582	0.997	0.577
Human Resources FC1	**0.473**	**0.490**	0.971	0.620	0.438	0.435	**0.988**	**0.643**	0.405	0.353	0.979	0.623
Human Resources FC2	**0.473**	**0.490**	0.971	0.620	0.421	0.404	**0.977**	**0.637**	0.412	0.360	0.993	0.624
HR IBM	**0.618**	**0.851**	0.996	**0.740**	0.542	0.716	**0.999**	0.559	0.437	0.430	0.983	0.532
Census Income FC1	**0.672**	0.774	0.973	0.685	**0.672**	**0.812**	**0.996**	**0.707**	0.578	0.630	0.911	0.626
Census Income FC2	**0.692**	**0.809**	0.976	**0.724**	0.567	0.569	**0.999**	0.629	0.582	0.674	0.933	0.618

Table 12 Final clustering distribution of fair-multicluster algorithm

	Divorced (%)	Married (%)	Single (%)	Observed
Cluster 1	16.3	39.5	44.2	(0.163, 0.395, 0.442)
Cluster 2	16.4	40.3	43.3	(0.164, 0.403, 0.433)
Cluster 3	13.9	43.1	43.1	(0.139, 0.431, 0.431)
Cluster 4	11.8	41.8	46.4	(0.118, 0.418, 0.464)

by bold-faced numbers. We conclude that Multicluster and Fair-Multicluster outperform K-Modes in most cases.

To better understand the proposed fairness measure, we give a detailed calculation of its value for the "*Human Resources FC2*" data set. The elements of this dataset have 3 different values of the Fairness or protected attribute (Marital Status): Divorced (14%), Married (41%) and Single (45%). Therefore, the "desired" distribution of this attribute will be (0.14, 0.41, 0.45).

Table 12 shows the final distributions of this attribute in each of the four clusters given by the Fair-Multicluster algorithm:

Then, we can calculate the Fairness measure as the average of the distances between the observed vectors and desired vectors for each final cluster:

$$Fairnessratio = \frac{\sum_{i=1}^{i=k}(1 - euclideandistance(Observed_i; Desired)}{numberofclusters(k)} = 0,981 \sim 98\%$$

Table 13 shows the Fairness measures of the final clusters given by the three algorithms. We conclude that, concerning the Fairness measure, Fair-Multicluster largely outperforms Multicluster and K-Modes in all cases.

Table 13 Comparative of fairness classification

Dataset	K-multicluster (Santos and Heras 2020)	Proposed algorithm fair-multicluster	K-modes
Absenteeism	0.678	**0.98**	0.680
Bank Marketing	0.905	**0.99**	0.958
CARS_Insurance	0.667	**0.99**	0.551
German Credit FC1	0.933	**0.99**	0.958
German Credit FC2	0.764	**1.00**	0.917
Human Resources FC1	0.737	**1.00**	0.792
Human Resources FC2	0.689	**0.97**	0.524
HR IBM	0.754	**0.97**	0.989
Census Income FC1	0.946	**0.93**	0.497
Census Income FC2	0.960	**0.99**	0.968

On the basis of the results obtained before and shown in Tables 11 and 13, we conclude that the proposed Fair-Multicluster algorithm has an excellent performance in terms of the fairness measure (as expected), while at the same time it outperforms the well-known K-Modes algorithm in terms of classification efficiency. We also conclude that the K-Multicluster algorithm often gets better results in terms of this last objective. Actually, the figures in both Tables allow comparing the performances of the K-Multicluster and Fair-Multicluster algorithms, thus giving a numerical evaluation of the trade-off between efficiency and fairness: considering, for instance, the CARS_Insurance dataset, the efficiency ratios are (FMI = 0.859, MMM = 0.966, NVI = 1.000, OI = 0.933) for the K-Multicluster algorithm and (three of them) decrease to (FMI = 0.732, MMM = 0.801, NVI = 1.000, OI = 0.789) for the Fair-Multicluster algorithm, while at the same time the fairness ratio increases from 0.667 to 0.99.

5 Discussion

The key ideas behind the Fair-Multicluster algorithm are easy to understand in intuitive terms. Perhaps the main contribution is the way it combines the initial multiclusters in order to reach a compromise between the opposite goals of clustering efficiency and fairness: on the one side, Step 2 merges similar clusters, trying to get highly homogeneous clusters in the final classification; on the other side, merging clusters in Step 3 is looking for a fair distribution of the values of the protected attribute. In other terms, repeating Step 2 increases the efficiency of the final cluster classification, while repeating Step 3 increases the fairness. Since efficiency and fairness often go in opposite directions (improving one of them usually has the consequence of worsening the other), we have to predefine some compromise between them. In practise, this compromise can be achieved by selecting the number of iterations of Step 2 before starting Step 3. Actually, working with small and medium size databases, we have seen that it is usually enough one single repetition of Step 2 in order to reach a reasonable value of the efficiency. For example, working with the German Credit file, we have got a significant improvement of the efficiency only after 5 iterations of Step 2, with the unfortunate consequence of a great loss of fairness. For this reason, in this paper we have worked with only one iteration of Step 2 in the German Credit example, obtaining good values of both efficiency and fairness. Nevertheless, when working with bigger files, it may be necessary to perform several experiments to find the optimal number of Step 2 iterations. Of course, this procedure can be very time-consuming, and the high computing time is perhaps the main drawback of the proposed Fair-Multicluster algorithm.

6 Conclusions and future work

Assuming the existence of a protected attribute such as race, gender or social status, in this paper we propose a clustering algorithm for finding homogeneous and fair clusters. The clusters should be homogeneous, that is, formed by similar elements, and should also be fair, not biased towards or against specific subgroups of the population. Of

course, there is a trade-off between fairness and efficiency, so that an increase in the fairness objective usually leads to a loss of classification efficiency. Yet the so-called Fair-Multicluster algorithm reaches a reasonable compromise between these goals. This algorithm can be considered as an adaptation of the K-Multicluster algorithm proposed by Santos and Heras (2020) for clustering categorical data bases, an algorithm which can be easily modified in order to get homogeneous and fair clusters.

The high performance of the Fair-Multicluster algorithm has been checked by comparing it with the Multicluster and the well-known K-Modes algorithms. Their classification efficiencies and fairness have been calculated in ten categorical data bases, using four well-known measures of efficiency and a measure of fairness based on the distance between the final distribution of the protected attribute and its desired distribution. As for the classification efficiency, Table 11 shows that both K-Multicluster and Fair-Multicluster algorithms outperform K-Modes in most cases. With respect to the fairness objective, Table 13 shows the highest performance in all cases of the Fair-Multicluster algorithm, reaching scores close to 100% in many cases. Besides, unlike K-Modes, the output of the Fair-Multicluster algorithm is not affected by randomness.[3] Replicability, classification efficiency and fairness are the major benefits of the proposed Fair-Multicluster algorithm.

Among the future developments of this methodology, we highlight its application to mixed data sets with both quantitative and qualitative attributes, and/or to data sets with several (more than one) protected attributes.

Authors contributions In our work, we address the issue of designing a fair algorithm for clustering categorical data. It is well known the great importance of the algorithms in our world today. It is often said that algorithms rule the world, with applications in many important fields such as finance, insurance, marketing, medicine, criminal justice, etc. For that reason, there is a growing interest in designing fair algorithms, that is, algorithms not biased towards or against specific subgroups of the population, like to prevent gender discrimination, we think that our paper could be of interest for publication in your journal.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Declarations

Conflict of interest We have no conflicts of interest to disclose.

Ethics approval We confirm that this work is original and has not been published elsewhere, nor is it currently under consideration for publication elsewhere.

[3] The obtained results are replicable, because the output of the algorithm only depends on the initial cluster ordering.

References

Abraham SSPD, Sundaram S. S (2020) Fairness in clustering with multiple sensitive attributes. In: Advances in database technology—EDBT, pp 287–298. arXiv:1910.05113

Ahmad A, Dey L (2007a) A method to compute distance between two categorical values of same attribute in unsupervised learning for categorical data set. Pattern Recognit Lett 28(1):110–118. https://doi.org/10.1016/j.patrec.2006.06.006

Ahmad A, Dey L (2007b) A k-mean clustering algorithm for mixed numeric and categorical data. Data Knowl Eng 63(2):503–527. https://doi.org/10.1016/j.datak.2007.03.016

Altaf S, Waseem Waseem M, Kazmi L (2020) IDCUP algorithm to classifying arbitrary shapes and densities for center-based clustering performance analysis. Interdiscip J Inf Knowl Manag 15:91–108. https://doi.org/10.28945/4541

Barocas S, Selbst AD (2016) Big data's disparate impact. Calif Law Rev 104(3):671–732. https://doi.org/10.2139/ssrn.2477899

Cao F, Liang J, Bai L (2009) A new initialization method for categorical data clustering. Expert Syst Appl 36(7):10223–10228. https://doi.org/10.1016/j.eswa.2009.01.060

Chen X, Fain B, Lyu L, Munagala K (2019) Proportionally fair clustering. In: 36th international conference on machine learning, ICML, pp 1782–1791. arXiv:1905.03674

Chierichetti F, Kumar R, Lattanzi S, Vassilvitskii S (2017) Fair clustering through fairlets. In: Advances in neural information processing systems, pp 5030–5038. arXiv:1802.05733

Dom BE (2012) An information-theoretic external cluster-validity measure. arXiv:1301.0565

Dua D, Graff C (2019) UCI machine learning repository. University of California, School of Information and Computer Science, Irvine

Esmaeili SA, Brubach B, Tsepenekas L, Dickerson JP (2020) Probabilistic fair clustering. arXiv:2006.10916

Fleiss JL (1971) Measuring nominal scale agreement among many raters. Psychol Bull 76(5):378–382. https://doi.org/10.1037/h0031619

Fleiss JL, Cohen J, Everitt BS (1969) Large sample standard errors of kappa and weighted kappa. Psychol Bull 72(5):323–327. https://doi.org/10.1037/h0028106

Fleiss JL, Levin B, Paik MC (2003) Statistical methods for rates and proportions. In: Statistical methods for rates and proportions. https://doi.org/10.1002/0471445428

Forgy EW (1965) Cluster analysis of multivariate data: efficiency versus interpretability of classification. Biometrics 21:768–780

Fowlkes EB, Mallows CL (1983) A Method for comparing two hierarchical clusterings. J Am Stat Assoc 78(383):553. https://doi.org/10.2307/2288117

Fraley C, Raftery AE (1998) How many clusters? Which clustering method? Answers via model-based cluster analysis. Comput J 41(8):578–588. https://doi.org/10.1093/comjnl/41.8.578

Gan G, Yang Z, Wu J (2005) A genetic k-modes algorithm for clustering categorical data. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), vol 3584 LNAI, pp 195–202. https://doi.org/10.1007/11527503_23

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

Guha S, Rastogi R, Shim K (2001) Cure: an efficient clustering algorithm for large databases. Inf Syst 26(1):35–58. https://doi.org/10.1016/S0306-4379(01)00008-4

Han J, Kamber M, Pei J (2012) Data mining: concepts and techniques. In: Morgan Kaufmann series in data management systems, 3rd edn. Elsevier

Headden WP, McClosky D, Charniak E (2008) Evaluating unsupervised part-of-speech tagging for grammar induction. In: Coling 2008—22nd international conference on computational linguistics, proceedings of the conference, vol 1, pp 329–336. https://doi.org/10.3115/1599081.1599123

Huang Z (1998) Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Min Knowl Discov 2(3):283–304. https://doi.org/10.1023/A:1009769707641

Huang Z, Ng MK, Rong H, Li Z (2005) Automated variable weighting in k-means type clustering. IEEE Trans Pattern Anal Mach Intell 27(5):657–668. https://doi.org/10.1109/TPAMI.2005.95

Huang Z (1997a) A fast clustering algorithm to cluster very large categorical data sets in data mining. In: Research issues on data mining and knowledge discovery, pp 1–8
Huang Z (1997b) Clustering large data sets with mixed numeric and categorical values. In: Proceedings of the first Pacific-Asia knowledge discovery and data mining conference, Singapore, World Scientific, pp 21–34
Huang JZ (2009) Clustering categorical data with k-modes. In: Encyclopedia of data warehousing and mining, second edition, pp 246–250. https://doi.org/10.4018/978-1-60566-010-3.ch040
Ji J, Pang W, Li Z, He F, Feng G, Zhao X (2020) Clustering mixed numeric and categorical data with cuckoo search. IEEE Access 8:30988–31003. https://doi.org/10.1109/ACCESS.2020.2973216
Jiang F, Liu G, Du J, Sui Y (2016) Initialization of K-modes clustering using outlier detection techniques. Inf Sci 332:167–183. https://doi.org/10.1016/j.ins.2015.11.005
Khan SS, Ahmad A (2013) Cluster center initialization algorithm for K-modes clustering. Expert Syst Appl 40(18):7444–7456. https://doi.org/10.1016/j.eswa.2013.07.002
Khan SS, Ahmad A (2012) Cluster center initialization for categorical data using multiple attribute clustering. MultiClust@ SDM, 3–10
Khan SS, Ahmad A (2015) Computing initial points using density based multiscale data condensation for clustering categorical data. In: International conference on applied artificial intelligence, ICAAI
Kim B (2017) A fast K-prototypes algorithm using partial distance computation. Symmetry 9(4):58. https://doi.org/10.3390/sym9040058
Kleindessner M, Awasthi P, Morgenstern J (2019) Fair k-center clustering for data summarization. In: 36th international conference on machine learning, ICML 2019, pp 5984–6003. arXiv:1901.08628
McQueen JB (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol 1, pp 281–297
Meilă M (2007) Comparing clusterings—an information based distance. J Multivar Anal 98(5):873–895. https://doi.org/10.1016/j.jmva.2006.11.013
Meilâ M, Heckerman D (2001) An experimental comparison of model-based clustering methods. Mach Learn. https://doi.org/10.1023/A:1007648401407
Ng MK, Wong JC (2002) Clustering categorical data sets using tabu search techniques. Pattern Recognit 35(12):2783–2790. https://doi.org/10.1016/S0031-3203(02)00021-3
Pietrzykowski M (2017) Local regression algorithms based on centroid clustering methods. Procedia Comput Sci 112:2363–2371. https://doi.org/10.1016/j.procs.2017.08.210
Reichart R, Rappoport A (2009) The NVI clustering evaluation measure. https://doi.org/10.5555/1596374.1596401
Rosenberg A, Hirschberg J (2007) V-measure: a conditional entropy-based external cluster evaluation measure, pp 410–420
Sajidha SA, Chodnekar SP, Desikan K (2018) Initial seed selection for K-modes clustering—a distance and density based approach. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2018.04.013
Santos MC, Heras A (2020) A multicluster approach to selecting initial sets for clustering of categorical data. Interdiscip J Inf Knowl Manag 15:227–246. https://doi.org/10.28945/4643
Vijaymeena MK, Kavitha K (2016) A Survey on similarity measures in text mining. Mach Learn Appl Int J 3(1):19–28. https://doi.org/10.5121/mlaij.2016.3103
Vinh NX, Epps J, Bailey J (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. J Mach Learn Res. https://doi.org/10.5555/1756006.1953024
Wagner S, Wagner D (2007) Comparing clusterings—an overview. Technical report 2006-04
Walker DD, Ringger EK (2008) Model-based document clustering with a collapsed Gibbs sampler. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, pp 704–712. https://doi.org/10.1145/1401890.1401975
Yu SS, Chu SW, Wang CM, Chan YK, Chang TC (2018) Two improved k-means algorithms. Appl Soft Comput J 68:747–755. https://doi.org/10.1016/j.asoc.2017.08.032
Zhu E, Ma R (2018) An effective partitional clustering algorithm based on new clustering validity index. Appl Soft Comput J 71:608–621. https://doi.org/10.1016/j.asoc.2018.07.026
Zhu L, Lei JS, Bi ZQ, Yang J (2013) Soft subspace clustering algorithm for streaming data. Ruan Jian Xue Bao/j Softw 24(11):2610–2627. https://doi.org/10.3724/SP.J.1001.2013.04469
Ziko IM, Granger E, Yuan J, Ayed IB (2019) Variational fair clustering. arXiv:1906.08207