Statistics and Probability Letters 84 (2014) 88–95

Contents lists available at ScienceDirect

Statistics and Probability Letters

journal homepage: www.elsevier.com/locate/stapro

On K-means algorithm with the use of Mahalanobis distances

Igor Melnykov [a,*,1], Volodymyr Melnykov [b,2]

[a] Nazarbayev University, Astana, 010000, Kazakhstan

[b] The University of Alabama, Tuscaloosa, AL 35487, USA

ARTICLE INFO

Article history:
Received 2 June 2013
Received in revised form 20 September 2013
Accepted 20 September 2013
Available online 27 September 2013

MSC:
62H30

Keywords:
K-means algorithm
Mahalanobis distance
Initialization

ABSTRACT

The K-means algorithm is commonly used with the Euclidean metric. While the use of Mahalanobis distances seems to be a straightforward extension of the algorithm, the initial estimation of covariance matrices can be complicated. We propose a novel approach for initializing covariance matrices.

1. Introduction

Cluster analysis is concerned with detecting relatively distinct data groups consisting of similar observations. The need in such data analysis has been seen in various areas of science. There are many clustering methods proposed in the literature that can be divided into two general groups: hierarchical methods with different merging or splitting rules called linkages (Ward, 1963) and partition-optimization algorithms such as K-means (Forgy, 1965; MacQueen, 1967) and K-medoids (Kaufman and Rousseuw, 1990). In this paper, we discuss some variations of the most well-known partition-optimization algorithm—K-means.

Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ be a set of observations in a p-dimensional space. Also, let the number of clusters K be pre-defined. The K-means algorithm minimizes the objective function given by $F(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = \sum_{k=1}^{K} \sum_{\boldsymbol{x}_i \in C_k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|^2$, where C_k denotes the kth cluster, $\bar{\boldsymbol{x}}_k = \frac{1}{n_k} \sum_{\boldsymbol{x}_i \in C_k} \boldsymbol{x}_i$ is the estimated center of the kth cluster, and n_k is the number of points currently in the kth cluster (Lozano et al., 1999). $\|\cdot\|$ denotes the norm used by the K-means algorithm. In particular, in the case of the Euclidean norm, $\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\| = \sqrt{(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)^T (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)}$. The K-means algorithm starts with K "seeds" supposedly representing cluster centers. Then, the rest of observations are assigned to the closest group centers. Based on this initial partition, cluster sample means need to be obtained and the procedure is repeated until a stable solution is found.

The K-means algorithm is very fast and due to its speed and ease of implementation has become one of the most popular clustering techniques. The algorithm is sensitive to the initial choice of cluster centers and there is an extensive research on

* Correspondence to: Colorado State University – Pueblo, Pueblo, CO 81001, USA. Tel.: +1 719 5492664.
E-mail address: igor.melnykov@colostate-pueblo.edu (I. Melnykov).

1 Igor Melnykov is an Associate Professor at the Department of Mathematics in Nazarbayev University (on leave from CSU-Pueblo).

2 Volodymyr Melnykov is an Assistant Professor at the Department of Information Systems, Statistics, and Management Science in the University of Alabama.

0167-7152/$ – see front matter
http://dx.doi.org/10.1016/j.spl.2013.09.026

initialization methods (Lozano et al., 1999; Erisoglu et al., 2011). In particular, it can be noted that the K-means algorithm experiences difficulties when the initial center "seeds" are chosen so that there are multiple "seeds" in the same cluster and some clusters are overlooked. A very common and the most simple way of dealing with this complication is to restart the algorithm several times picking the center "seeds" at random. This approach works well if the number of clusters is not too large and they are all fairly well represented. But even in the case of $n = 1500$ points spread among $K = 15$ clusters in equal proportions, the probability of having a "seed" in each cluster is approximately $3.2 \cdot 10^{-6}$. The need for the initialization techniques that would choose initial cluster centers in a more careful way rather than randomly is addressed, for instance, by Erisoglu et al. (2011). Proposed variations of the algorithm typically aim at finding better original cluster centers but do not concern the shape of detected clusters. In the mean time, the ability of a particular distance measure to accommodate clusters of certain configuration is another important aspect arising in the application of the K-means algorithm. The most widely used, Euclidean distance measure, works well for clusters with roughly spherical homogeneous covariance matrices. The use of the Mahalanobis distance $d^{(m)} = \sqrt{(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$, with $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ being the mean vector and covariance matrix, respectively, is beneficial when clusters are elliptical in shape, although there is an additional need to estimate the covariance matrix $\boldsymbol{\Sigma}$ (Gnanadesikan et al., 1993). Some other, less frequently used metrics, are discussed in Grabusts (2011).

In our paper, we develop a method that is geared towards the clusters of roughly elliptical shapes and uses the Mahalanobis metric as distance measure. While the use of Mahalanobis distances in this framework is relatively straightforward, it is not very popular. The main reason for that is related to the poor performance of the method when covariance matrices are not initialized properly. In Section 2, we develop an initialization procedure that aims to gather the information about the covariance matrices specifically for the purpose of using it subsequently by the K-means algorithm. Section 3 investigates the performance of the proposed variation of the method and compares it with two other approaches. In Section 4, we illustrate our proposed method on a celebrated *Iris* dataset. The paper is concluded with a discussion in Section 5.

2. Methodology

2.1. Proposed method

Before we describe the proposed algorithm thoroughly, its main idea will be outlined. As mentioned in Section 1, the K-means based on Mahalanobis distances will be employed. The necessary stage in such a variation of the algorithm is clusters' covariance matrix estimation. While it is trivial at each iteration of the K-means, finding good initial estimates of $\boldsymbol{\Sigma}_k$ for $k = 1, 2, \ldots, K$ is the most difficult step. In Section 3, we will illustrate the importance of initialization, without which the entire procedure fails miserably. Our initialization strategy involves identifying a group of points with a high concentration of neighbors that represents the "core" of the selected cluster. These points can be used to provide a very rough covariance matrix estimate. This estimate can be improved as discussed in the algorithm later. Then, it can be used for calculating Mahalanobis distances for the rest of the points. Presumably, the closest points should belong to the same cluster and the first observed jump in distances can be seen as an indication that points from a different cluster start being captured. This strategy can be repeated K times, each time eliminating the selected points to avoid repetitive selection of the same clusters. Now, the outlined idea will be considered in greater detail. The algorithm's key steps have names or numbers assigned to them. Steps 1–8 represent the initialization stage while Steps 9–11 implement the K-means algorithm.

Algorithm description:

1. *Initial setup.* For each point $\boldsymbol{x}_i$ in the dataset, compute the Euclidean distances $d_{i,j}$ to other points, $d_{i,j} = \sqrt{(\boldsymbol{x}_i-\boldsymbol{x}_j)^T(\boldsymbol{x}_i-\boldsymbol{x}_j)}$, $i, j = 1, 2, \ldots, n$, where $i \neq j$; then for each observation $\boldsymbol{x}_i$, compute the sum of w smallest distances $d_{i,j}$, *i.e.* for ordered distances $d_{i,(1)} \leq d_{i,(2)} \leq \cdots \leq d_{i,(n-1)}$, calculate $S_{i,w} = \sum_{j=1}^{w} d_{i,(j)}$. In our experiments, we considered $w = 20$.

   Set $k = 1$. Also, let M be the number of points not assigned to any cluster and set M equal to n, the total number of points.
2. *Setting the center of the kth cluster.* Randomly assign one of the points to be the proposed cluster center. The assignment probabilities are inversely proportional to the ranks $i_1, i_2, \ldots, i_M$, where $S_{i_1,w} \leq S_{i_2,w} \leq \cdots \leq S_{i_M,w}$. Thus, each probability is given by $\frac{1}{i_s}\left(\sum_{l=1}^{M}\frac{1}{i_l}\right)^{-1}$, $s = 1, 2, \ldots, M$.
3. Assuming that D is the minimum number of points possible in a cluster, assign D points nearest to the center to the current cluster C_k. In our experiments, we used $D = 20$.
4. Compute the estimates of the mean $\bar{\boldsymbol{x}}_k$ and covariance matrix $\hat{\boldsymbol{\Sigma}}_k$ using the points currently in the cluster.
5. Update the membership of points in the cluster according to a probability coverage criterion. We used the inequality $(\boldsymbol{x}_i-\bar{\boldsymbol{x}}_k)^T\hat{\boldsymbol{\Sigma}}_k^{-1}(\boldsymbol{x}_i-\bar{\boldsymbol{x}}_k) < \chi^2_{p,0.99}$ as the criterion, *i.e.*, all points satisfying this inequality are included in the kth cluster. Here, $\chi^2_{p,0.99}$ is the 99th percentile of the chi-square distribution with p degrees of freedom.
6. Repeat Steps 3, 4, and 5 several times to obtain a rough estimate of the covariance matrix. (We used five iterations of this process.)

7. *Finding the edge of the current cluster.* Compute the estimated Mahalanobis distances $\hat{d}_j^{(m)} = \sqrt{\left((\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k)^T \hat{\boldsymbol{\Sigma}}_k^{-1} (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_k)\right)}$, $j = 1, 2, \ldots, M$, for each of the M points remaining in consideration and form the ordered sequence $\hat{d}_{j_1}^{(m)} \leq \hat{d}_{j_2}^{(m)} \leq \cdots \leq \hat{d}_{j_M}^{(m)}$. For the adjacent elements of this sequence, define $\Delta_l = \ln(\hat{d}_{j_{l+1}}^{(m)}) - \ln(\hat{d}_{j_l}^{(m)})$, $l = 1, 2, \ldots, M-1$ and compute $r = \text{argmin}_l\{\Delta_l : \frac{\Delta_l - \mu_{\Delta_l}}{\sigma_{\Delta_l}} > k_{Cheb}\}$. If $\{\Delta_l : \frac{\Delta_l - \mu_{\Delta_l}}{\sigma_{\Delta_l}} > k_{Cheb}\} = \emptyset$, let $r = \arg_l(\max \Delta_l)$. (In experimental runs, we used $k_{Cheb} = 10$.) Declare all points $\boldsymbol{x}_{j_1}, \boldsymbol{x}_{j_2}, \ldots, \boldsymbol{x}_{j_r}$ to be members of the kth cluster and exclude them from further consideration. Reduce M by r.
8. If $k \leq K$, increase k by 1 and go back to Step 2.
9. Update all estimates $\bar{\boldsymbol{x}}_k$ and $\hat{\boldsymbol{\Sigma}}_k$, $k = 1, 2, \ldots, K$, using the points currently assigned to each cluster.
10. Using Mahalanobis distances, assign each of the n points to the cluster with the nearest center. Evaluate the solution using some specified criterion.
11. Repeat Steps 9–10 until a stable solution is reached or $n_k < D$ for some k.

Steps 3–6 provide the covariance matrix estimation. They aim at including more observations in the original cluster core before the calculation of Mahalanobis distances starts. These steps are very important as the performance of the entire strategy depends on the quality of covariance matrix estimation. It has to be noticed that several replications of Steps 3–5 can be recommended. However, one should not repeat it too many times to avoid capturing points from neighboring clusters when data groups overlap considerably.

The edge of the current cluster is determined with the use of the quantities Δ_l. Large values of Δ_l, or so-called peaks, correspond to the considerable changes in Mahalanobis distances that are expected to occur when we evaluate these distances for the points that are not consistent with the covariance matrix of the current cluster. We use log-transformed distances in Step 7 for a purpose that is two-fold. On the one hand, since $\ln(\hat{d}_{j_{l+1}}^{(m)}) - \ln(\hat{d}_{j_l}^{(m)}) = \ln\left(\frac{\hat{d}_{j_{l+1}}^{(m)}}{\hat{d}_{j_l}^{(m)}}\right) = \ln\left(1 + \frac{\hat{\delta}}{\hat{d}_{j_l}^{(m)}}\right)$, where $\hat{\delta} = \hat{d}_{j_{l+1}}^{(m)} - \hat{d}_{j_l}^{(m)}$, we evaluate the relative change in distances rather than the absolute change. On the other hand, logarithms make the peaks marking far away clusters less prominent, thus, making the identification of the edge for the current cluster more likely.

In Step 7, we employ Chebyshev's inequality to identify the most prominent peaks Δ_l. It should be noted that the distribution of differences Δ_l is usually extremely skewed to the right and has numerous outliers which complicates the task of identifying cluster edges using usual outlier detection methods. However, Chebyshev's inequality serves our purpose quite well due to its notorious conservativeness and the resulting ability to detect only the most prominent peaks. We initialized the algorithm with $k_{Cheb} = 10$ giving a 99% probability coverage for the inequality. Such a high value of k_{Cheb} carries a potential risk that no peaks will be identified if clusters overlap significantly. In this situation, we pick the tallest peak to mark the cluster edge and there is a possibility that two or more clusters will be lumped together. If this happens and upon the completion of the initialization part of the algorithm the number of identified clusters is below K, we reduce k_{Cheb} by 1 and repeat the initialization starting with Step 2. In the vast majority of cases that we tested, k_{Cheb} did not go below 6.

In the K-means algorithm using Euclidean distances, a stable solution is reached at a local minimum of the objective function $F(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ and no additional criterion besides the objective function is needed to evaluate the obtained solution. However, the use of Mahalanobis distances necessitates the implementation of a different approach to solution evaluation. For this purpose, we propose a criterion that has a close connection to the maximum likelihood estimator of the p-variate normal distribution. This natural relationship can be justified by the correspondence between the Mahalanobis distance and the Gaussian density kernel. In other words, we propose a criterion based on approximating the found clusters with p-dimensional normal densities. Based on a sample of size n_k, the largest value that can be reached by the likelihood function of the p-dimensional normal distribution is given by $(2\pi)^{-n_k p/2}(\det(\hat{\boldsymbol{\Sigma}}_k))^{-n_k/2}\exp(-n_k p/2)$. Then, based on the current classification vector $z_1, z_2, \ldots, z_n$ and the so-called classification likelihood given by $L(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k; k = 1, 2, \ldots, K) = \prod_{i=1}^n \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})$, where $\phi(\cdot)$ is the normal pdf, we obtain $\max\{\log L\} \propto -\sum_{k=1}^K \hat{n}_k \sum_{i=1}^p \log(\hat{\lambda}_{k,i})$, where $\hat{\lambda}_{k,i}$ are the eigenvalues of the covariance matrix $\hat{\boldsymbol{\Sigma}}_k$. Thus, letting $\mathcal{A} = -\sum_{k=1}^K \hat{n}_k \sum_{i=1}^p \log(\hat{\lambda}_{k,i})$, $\mathcal{A}$ is used as the solution quality criterion on Step 10. The classification that results in the highest observed value of $\mathcal{A}$ is reported as the best solution that was found.

It is possible that in the process of reaching a stable solution, the number of points in one of the clusters becomes smaller than D, the minimum number of points required in a cluster. In that case, the algorithm stops and the best solution obtained up to that point is reported.

2.2. Theoretical justification

Since the use of the K-means algorithm based on Mahalanobis distances has been justified in the introduction, we focus on studying the proposed initialization algorithm. Namely, we would like to know the conditions under which Step 7 of the algorithm is expected to produce a high peak.

Result 2.1. *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$ be independent identically distributed (iid) random variables with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Also, let $\boldsymbol{X}_{n+1}$ be an independent (from $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n$) random variable with mean $\boldsymbol{\mu}_\bullet$ and covariance matrix $\boldsymbol{\Sigma}_\bullet$. Then, for sufficiently high n, the peaks constructed at Step 7 of the proposed algorithm are expected to be higher than usual when $(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu}) > p - tr\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\}$.*

Proof. Using the facts that $\bar{\boldsymbol{X}}_n \xrightarrow{p} \boldsymbol{\mu}$ and $\hat{\boldsymbol{\Sigma}}_n^{-1}\boldsymbol{\Sigma} \xrightarrow{p} \boldsymbol{I}_p$, it is easy to show that the estimated Mahalanobis distance $\hat{d}_{n+1}^{(m)} = (\boldsymbol{X}_{n+1} - \bar{\boldsymbol{X}}_n)'\hat{\boldsymbol{\Sigma}}_n^{-1}(\boldsymbol{X}_{n+1} - \bar{\boldsymbol{X}}_n)$ converges in probability to the Mahalanobis distance $d_{n+1}^{(m)} = (\boldsymbol{X}_{n+1} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})$, *i.e.*, $\hat{d}_{n+1}^{(m)} \xrightarrow{p} d_{n+1}^{(m)}$. On the other hand,

$$\begin{aligned}
E\{d_{n+1}^{(m)}\} &= E\{(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})\} \\
&= \mathrm{tr}\{E\{(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})\}\} \\
&= \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}E\{(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})(\boldsymbol{X}_{n+1} - \boldsymbol{\mu})'\}\} \\
&= \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}[E\{\boldsymbol{X}_{n+1}\boldsymbol{X}_{n+1}'\} - E\{\boldsymbol{X}_{n+1}\}\boldsymbol{\mu}' - \boldsymbol{\mu}E\{\boldsymbol{X}_{n+1}'\} + \boldsymbol{\mu}\boldsymbol{\mu}']\} \\
&= \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}[\boldsymbol{\Sigma}_\bullet + \boldsymbol{\mu}_\bullet\boldsymbol{\mu}_\bullet' - \boldsymbol{\mu}_\bullet\boldsymbol{\mu}' - \boldsymbol{\mu}\boldsymbol{\mu}_\bullet' + \boldsymbol{\mu}\boldsymbol{\mu}']\} \\
&= \mathrm{tr}\{\boldsymbol{\Sigma}^{-1}\left[\boldsymbol{\Sigma}_\bullet + (\boldsymbol{\mu} - \boldsymbol{\mu}_\bullet)(\boldsymbol{\mu} - \boldsymbol{\mu}_\bullet)'\right]\} \\
&= \mathrm{tr}\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\} + (\boldsymbol{\mu} - \boldsymbol{\mu}_\bullet)'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \boldsymbol{\mu}_\bullet).
\end{aligned}$$

It can be noted that if $\boldsymbol{X}_{n+1}$ belongs to the same cluster as the other n points, $\boldsymbol{\mu}_\bullet = \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_\bullet = \boldsymbol{\Sigma}$. Then, it follows that $E\{d_{n+1}^{(m)}\} = \mathrm{tr}\{\boldsymbol{I}_p\} = p$. Therefore, for sufficiently large n, peaks are expected to be higher than usual when the condition $E\{d_{n+1}^{(m)}\} > p$ holds. In other form, this condition can be written as $(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu}) > p - \mathrm{tr}\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\}$. □

The condition $(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu}) > p - \mathrm{tr}\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\}$ provides considerable insight into the problem. The quantity $(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})$ represents the Mahalanobis distance between the center of the current cluster and the center of another data group. The term $\mathrm{tr}\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\}$ reflects the similarity between $\boldsymbol{\Sigma}_\bullet$ and $\boldsymbol{\Sigma}$ and is always positive as $\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}$ is a positive definite matrix. It is often the case that $\mathrm{tr}\{\boldsymbol{\Sigma}_\bullet\boldsymbol{\Sigma}^{-1}\} > p$ and higher peaks are expected to be observed even if $\boldsymbol{\mu}_\bullet = \boldsymbol{\mu}$. On the other hand, it is not impossible to encounter $\boldsymbol{\mu}_\bullet$ and $\boldsymbol{\Sigma}_\bullet$ such that the above condition does not hold. As an example, one can consider $\boldsymbol{\mu}_\bullet = \boldsymbol{\mu}$ and $\boldsymbol{\Sigma}_\bullet$ with the same orientation as in $\boldsymbol{\Sigma}$ but eigenvalues considerably higher than those in $\boldsymbol{\Sigma}$. In this case, one cluster "swallows" the other one and there is no surprise that we expect difficulties in identifying a point representing a foreign data group. On the contrary, when clusters do not overlap too much and their centers are not too close to each other, the proposed algorithm is expected to perform well. We can also note that since the logarithmic function is strictly increasing, the result stated is also valid if the *log*-function is applied to Mahalanobis distances as in Step 7 of the proposed algorithm.

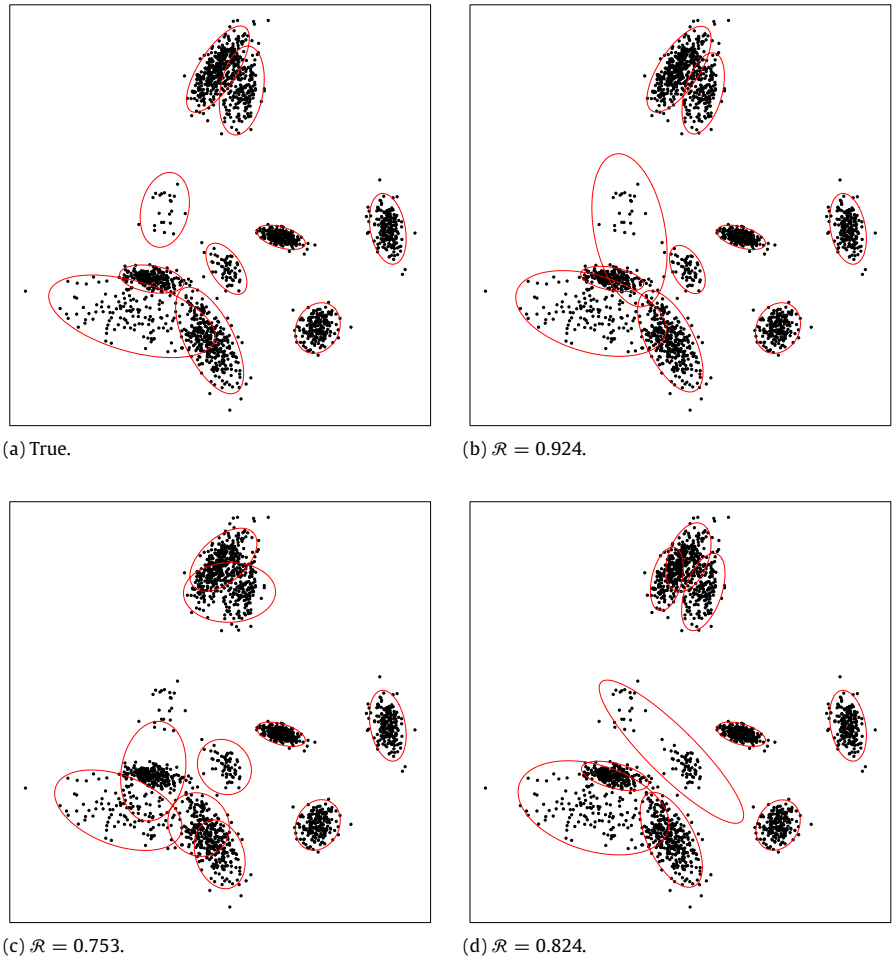
The obtained result is distribution-free. However, in those cases when the distribution is known, the procedure can be enhanced by incorporating this additional information. As an illustration, consider a popular setting with Gaussian clusters. In this case, the distribution of $d_{n+1}^{(m)}$ can be derived. Let $\boldsymbol{Z} \sim N_p(\underline{0}, \boldsymbol{I}_p)$, then

$$\begin{aligned}
d_{n+1}^{(m)} &= (\boldsymbol{X}_{n+1} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}_{n+1} - \boldsymbol{\mu}) \\
&\overset{d}{=} (\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\boldsymbol{Z} + \boldsymbol{\mu}_\bullet - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\boldsymbol{Z} + \boldsymbol{\mu}_\bullet - \boldsymbol{\mu}) \\
&= \left[\boldsymbol{Z} + \boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right]' \boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\left[\boldsymbol{Z} + \boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right] \\
&= \left[\boldsymbol{Z} + \boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right]' \boldsymbol{\Gamma}\boldsymbol{\Lambda}\boldsymbol{\Gamma}'\left[\boldsymbol{Z} + \boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right],
\end{aligned}$$

where $\boldsymbol{\Lambda} = diag\{\lambda_1, \lambda_2, \ldots, \lambda_p\}$ is the diagonal matrix of eigenvalues of the positive definite matrix $\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}$ and $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_p)$ is the corresponding matrix of eigenvectors, and then $\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}_\bullet^{\frac{1}{2}} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}\boldsymbol{\Gamma}'$ by spectral decomposition. For $\boldsymbol{W} = \boldsymbol{\Gamma}'\boldsymbol{Z} \sim N_p(\underline{0}, \boldsymbol{I}_p)$ and $\nu_j = \boldsymbol{\gamma}_j'\boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})$, we immediately obtain

$$\begin{aligned}
d_{n+1}^{(m)} &= \left[\boldsymbol{W} + \boldsymbol{\Gamma}'\boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right]' \boldsymbol{\Lambda}\left[\boldsymbol{W} + \boldsymbol{\Gamma}'\boldsymbol{\Sigma}_\bullet^{-\frac{1}{2}}(\boldsymbol{\mu}_\bullet - \boldsymbol{\mu})\right] \\
&= \sum_{j=1}^{p}\lambda_j(W_j + \nu_j)^2.
\end{aligned}$$

This implies that $d_{n+1}^{(m)}$ is distributed according to a linear combination of independent non-central χ_1^2 random variables with non-centrality parameters ν_j^2.



(a) True. (b) $\mathcal{R} = 0.924$. (c) $\mathcal{R} = 0.753$. (d) $\mathcal{R} = 0.824$.

Fig. 1. Classifications for the illustrative example: (a) true cluster composition, (b) proposed method, (c) K-means with Euclidean distances, (d) K-means with Mahalanobis distances and naive initialization. Red ellipses represent confidence regions for the obtained clusters. $\mathcal{R}$ represents the adjusted Rand index.

This result can be immediately used for finding approximate confidence intervals for $\hat{d}_{n+1}^{(m)}$ which might lead to an improved performance of the procedure in detecting boundaries of Gaussian clusters. While this idea can be helpful for dealing with Gaussian clusters in this particular framework, we would like to keep our algorithm as free from distributional forms as possible.

2.3. Illustrative example

In this section, we provide an example to show the strength of our algorithm and illustrate its use. We consider an example with $K = 10$ clusters, where $n = 2000$ and $p = 2$. The dataset was generated using the package MixSim (Melnykov et al., 2012). This package is designed to generate Gaussian mixtures according to the pre-specified level of complexity defined by the average or maximum pairwise overlap. The overlap between two p-dimensional Gaussian densities is defined as the sum of associated misclassification probabilities (Maitra and Melnykov, 2010). Besides K, n, and p, we specified the maximum pairwise overlap $\breve{\omega}$ equal to 0.1. The proportions of points among the components of the mixture were chosen unequal, varying between 0.01 and 0.30. This does not affect the performance of our suggested initialization method, but the K-means algorithm with random initialization often experiences difficulties "seeding" each of the clusters in such a situation. The ten clusters along with the ellipses representing their covariance matrices are shown in Fig. 1(a).

We picked two variations of the K-means algorithm to compare to our method. The first one carried out a random initialization and then used the Euclidean distances in the implementation of the algorithm. This is the most frequently used version of K-means. For the other method, we used the same random initialization strategy as for the regular K-means, but then used the Mahalanobis distances during the iterative stage of the algorithm. All three methods require different amounts of time computationally, therefore, we compared them by allowing them the same intervals of computer time. As a comparison measure, we used the adjusted Rand index ($\mathcal{R}$) that measures the agreement between two

classifications (Hubert and Arabie, 1985). In our case, these are the true and estimated partitionings. The value of the adjusted Rand index equal to 1 represents the complete agreement between two classifications, while $\mathcal{R}$ close to 0 shows the degree of agreement that would be expected by random chance.

As can be seen in Fig. 1(a), the example presents a challenging classification problem. There are substantial overlaps of several clusters that come in a variety of shapes and sizes. Our proposed method performed quite well (Fig. 1(b)) identifying all clusters correctly and showing $\mathcal{R} = 0.924$. The only visible difficulty encountered by the method was the estimation of the covariance matrix of a sparse cluster in the middle of the plot. The two heavily populated clusters at the top of the plot were distinguished quite well even though the large degree of overlap led to a few point misclassifications here and lowered the value of $\mathcal{R}$ somewhat. The regular K-means procedure with Euclidean distances yielded $\mathcal{R} = 0.753$. From Fig. 1(c), we observe that several elongated clusters were identified correctly by the algorithm. However, using Euclidean distances, the method experienced problems with the two clusters at the top misclassifying many points there. In addition, two clusters at the bottom left part of the plot were not detected correctly. Finally, the K-means with Mahalanobis distances and random initialization (Fig. 1(d)) showed $\mathcal{R} = 0.824$. With the advantage of using Mahalanobis distances, the method was able to identify some clusters more accurately than the regular K-means, but apparently the method stumbled on the initialization as the sparse cluster in the middle of the plot was never separated and this led to placing too many "seeds" in other clusters. This emphasizes the fact that the use of Mahalanobis distances by themselves is not sufficient even when modeling such a moderate number of clusters as $K = 10$ and should be coordinated with the initialization part of the algorithm.

3. Experimental validation

In this section, we consider a simulation study concerned with the performance of three different versions of the K-means algorithm. The first one is the standard algorithm based on Euclidean distances that was discussed in Section 1. The second variation of K-means relies on the calculation of Mahalanobis distances but assumes the same initialization as in the regular K-means. The last version of the algorithm is our proposed method that involves the Mahalanobis metric along with the initialization strategy proposed in Section 2.1. The results of the study are summarized in Table 1. 2- and 5-variate mixture models with 10 and 20 components were simulated by MixSim (Melnykov et al., 2012). The clustering complexity was reflected through the level of maximum overlap $\breve{\omega} = 0.001, 0.01, 0.1$, where the values 0.001 and 0.01 correspond to high and moderate separation between components (Maitra and Melnykov, 2010) and 0.1 implies that at least some mixture components overlap very substantially. Under each parameter setting, we generated 25 mixture models. From each obtained model, one dataset of size 500 (for $K = 10$) or 1000 (for $K = 20$) was simulated. For each simulated dataset, the three studied K-means methods were run and the adjusted Rand index was calculated. Table 1 provides the obtained median ($\mathcal{R}_{\frac{1}{2}}$) and interquartile range ($\mathcal{I}_{\mathcal{R}}$) of the index. Also, we report the number of times a particular method found the best solution compared to the solutions proposed by the two competitors ($\mathcal{B}$); we also provide the average rank ($\mathcal{P}$). Fig. 2 provides a visual display of the obtained results expressed in terms of the adjusted Rand index $\mathcal{R}$.

As we can see, the performance of the proposed procedure (called *Km-M+*) is very good. It is clearly the best performer for well- and moderately-separated clusters ($\breve{\omega} = 0.001, 0.01$) with the highest number of best solutions found $\mathcal{B}$ and the top rank $\mathcal{P}$ close to 1. The closest competitor is the regular K-means algorithm based on Euclidean distances (*Km-E*). However, the performance of the algorithm is somewhat worse for $K = 20$ than for $K = 10$. It is a well expected effect of initialization problems for the large number of clusters. In the mean time, our proposed method performs remarkably well for both levels of K. It also outperforms the K-means with the Mahalanobis distances and naive initialization (called *Km-M*) which clearly suggests that the effective initialization of covariance matrices is the cornerstone of the K-means based on Mahalanobis distances. The performance of our algorithm degrades when the overlap becomes high ($\breve{\omega} = 0.1$) and the number of dimensions increases ($p = 5$). One of the reasons for that is that it becomes increasingly difficult to estimate covariance matrices in such cases. This is a well-known problem often referred to as "the curse of dimensionality"; it implies that in higher dimensions, the separation between clusters has to be higher to observe good performance of clustering algorithms. As we can see, the other methods also degrade considerably for $p = 5$ and $\breve{\omega} = 0.1$. Overall, we can conclude that the performance of our approach is promising, very often better than that of K-means, and the initialization stage is crucial for the clustering success.

4. Application

As a test of our algorithm on real-life data, we consider its performance on the dataset *Iris* that was first analyzed by Fisher (1936). This dataset consists of 150 4-dimensional observations representing three different Iris species: *Iris Setosa*, *Iris Virginica*, and *Iris Versicolor*. Each class includes 50 data points. While one of the clusters (*Setosa*) is very well separated, the other two exhibit a substantial overlap, making this a challenging classification problem for many methods.

The K-means with Euclidean distances misclassifies 16 of the data points and offers a partition with $\mathcal{R} = 0.730$. Our proposed method along with *Km-M* obtain the same solution with just 5 misclassifications and the corresponding value of $\mathcal{R}$ equal to 0.904. The complete agreement between the two latter methods is to be expected as the initialization does not present many challenges with one well-separated cluster and equal group representations, but capturing the shape of the clusters is a more important task here. In summary, the proposed method showed that it was up to the task and handled the *Iris* dataset fairly well.

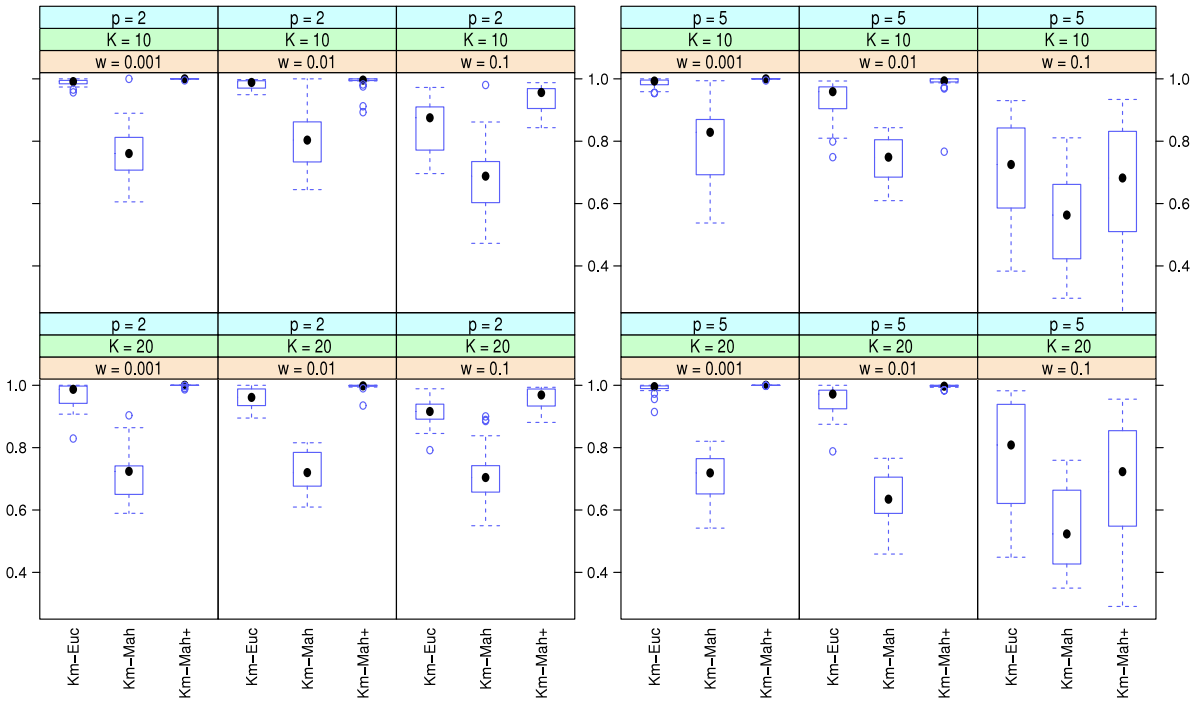


Fig. 2. Sample distributions of the adjusted Rand index $\mathcal{R}$ for the three variations of the K-means algorithm from the simulation study of Section 3. p represents dimensions, K stands for the number of clusters, and ω is the level of maximum overlap.

Table 1
Results of the simulation study for different number of clusters K and dimensions p; $\breve{\omega}$ is the maximum overlap between mixture components. $\mathcal{B}$ represents the number of times the procedure produced the highest adjusted Rand index, $\mathcal{P}$ shows the average rank based on the same index, $\mathcal{R}_{\frac{1}{2}}$ and $\mathcal{I}_{\mathcal{R}}$ are the corresponding median and interquartile range.

			$p = 2$			$p = 5$		
			$\breve{\omega} = 0.001$	$\breve{\omega} = 0.01$	$\breve{\omega} = 0.1$	$\breve{\omega} = 0.001$	$\breve{\omega} = 0.01$	$\breve{\omega} = 0.1$
$K = 10$	*Km-E*	$\mathcal{B}, \mathcal{P}$	5, 1.92	5, 1.90	0, 2.08	2, 2.00	1, 1.96	15, 1.44
		$\mathcal{R}_{\frac{1}{2}}$	0.991	0.988	0.875	0.993	0.959	0.725
		$\mathcal{I}_{\mathcal{R}}$	0.011	0.024	0.139	0.015	0.070	0.256
	Km-M	$\mathcal{B}, \mathcal{P}$	1, 2.96	2, 2.90	1, 2.84	0, 2.96	0, 3.00	1, 2.60
		$\mathcal{R}_{\frac{1}{2}}$	0.761	0.804	0.688	0.829	0.749	0.563
		$\mathcal{I}_{\mathcal{R}}$	0.105	0.128	0.132	0.177	0.120	0.239
	Km-M+	$\mathcal{B}, \mathcal{P}$	25, 1.12	22, 1.20	24, 1.08	25, 1.04	24, 1.04	9, 1.96
		$\mathcal{R}_{\frac{1}{2}}$	1.000	0.996	0.956	1.000	0.994	0.682
		$\mathcal{I}_{\mathcal{R}}$	0.000	0.005	0.064	0.000	0.010	0.322
$K = 20$	*Km-E*	$\mathcal{B}, \mathcal{P}$	3, 1.94	4, 1.90	6, 1.76	3, 1.94	1, 1.98	18, 1.28
		$\mathcal{R}_{\frac{1}{2}}$	0.987	0.961	0.916	0.996	0.972	0.809
		$\mathcal{I}_{\mathcal{R}}$	0.055	0.053	0.048	0.009	0.059	0.318
	Km-M	$\mathcal{B}, \mathcal{P}$	0, 3.00	0, 3.00	0, 3.00	0, 3.00	0, 3.00	0, 2.92
		$\mathcal{R}_{\frac{1}{2}}$	0.724	0.720	0.704	0.719	0.635	0.523
		$\mathcal{I}_{\mathcal{R}}$	0.092	0.109	0.085	0.113	0.116	0.237
	Km-M+	$\mathcal{B}, \mathcal{P}$	25, 1.06	24, 1.10	19, 1.24	25, 1.06	25, 1.02	8, 1.80
		$\mathcal{R}_{\frac{1}{2}}$	1.000	0.998	0.969	1.000	0.997	0.723
		$\mathcal{I}_{\mathcal{R}}$	0.000	0.003	0.055	0.000	0.004	0.306

5. Discussion

This paper presented a novel version of the K-means algorithm based on the Mahalanobis distance metric. While the use of Mahalanobis distances is not new in clustering framework, they are not commonly used due to the necessity to initialize data group covariance matrices. We proposed a strategy aiming at addressing this issue. The developed procedure is illustrated on a synthetic dataset; its performance was compared to that of two other versions of the K-means algorithm. A conducted simulation study proved viability of the proposed method, which outperformed the competitors in multiple situations. An application of the procedure to the time-honored *Iris* dataset was considered with good results.

Overall, we conclude that the use of Mahalanobis distances in the K-means algorithm is more flexible and should be preferred for clusters of elliptic shapes. At the same time, the use of Mahalanobis distances is not realistic without a proper covariance matrix initialization strategy. Our developed procedure addresses the above-mentioned challenges and demonstrates promising results.

Acknowledgment

This research was supported in part by the Seed Grant $K\Phi - 13/15$ of the Corporate Fund "Fund of Social Development" of Nazarbayev University.

References

Erisoglu, M., Calis, N., Sakallioglu, S., 2011. A new algorithm for initial cluster centers in k-means algorithm. Pattern Recognition Letters 32, 1701–1705.
Fisher, R.A., 1936. The use of multiple measurements in taxonomic poblems. Annals of Eugenics 7, 179–188.
Forgy, E., 1965. Cluster analysis of multivariate data: efficiency vs. interpretability of classifications. Biometrics 21, 768–780.
Gnanadesikan, R., Harvey, J.W., Kettenring, J.R., 1993. Mahalanobis metrics for cluster analysis. Sankhya, Series A 55, 494–505.
Grabusts, P., 2011. The choice of metrics for clustering algorithms, in: Proceedings of the 8th International Scientific and Practical Conference, pp. 70–76.
Hubert, L., Arabie, P., 1985. Comparing partitions. Journal of Classification 2, 193–218.
Kaufman, L., Rousseuw, P.J., 1990. Finding Groups in Data. John Wiley & Sons, New York.
Lozano, J.A., Pena, J.M., Larranaga, P., 1999. An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recognition Letters 20, 1027–1040.
MacQueen, J., 1967. Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium 1, 281–297.
Maitra, R., Melnykov, V., 2010. Simulating data to study performance of finite mixture modeling and clustering algorithms. Journal of Computational and Graphical Statistics 19, 354–376.
Melnykov, V., Chen, W.C., Maitra, R., 2012. MixSim: an R package for simulating data to study performance of clustering algorithms. Journal of Statistical Software 51, 1–25.
Ward, J.H., 1963. Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association 58, 236–244.