Lecture Notes on k-Means Clustering (I)

Technical Report · October 2013

DOI: 10.13140/2.1.3738.4480

1 author:

Christian Bauckhage
University of Bonn
366 PUBLICATIONS **5,647** CITATIONS

SEE PROFILE

Lecture Notes on k-Means Clustering (I)

Christian Bauckhage
B-IT, University of Bonn

This is the first in a series of lecture notes on k-means clustering, its variants, and applications. In this note, we study basic ideas behind k-means clustering and identify common pitfalls in its use.

k-Means Clustering

Although the k-means clustering algorithm is frequently applied in practice, it seems that many users are not familiar with the theory behind it. This is unfortunate, because the algorithm operates on certain implicit assumptions which, if ignored, may lead to seemingly unreasonable results. We analyze these principles, discuss when and how to apply k-means, and point out common pitfalls in its use.

FOR THE TIME BEING, we focus on the problem of clustering data that can be embedded in Euclidean vector spaces; extensions to more abstract settings will be studied later on.

Accordingly, let us assume a data set $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^m$ whose n elements $\boldsymbol{x}_j$ are m dimensional real-valued vectors that are to be clustered (see Fig. 1 for an example).

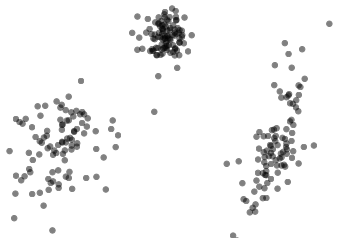


Figure 1: Didactic example of $n = 150$ data points $\boldsymbol{x}_j \in \mathbb{R}^2$ sampled from three bivariate Gaussian distributions.

WHENEVER WE SET OUT TO CLUSTER DATA, we are interested in detecting latent structures or regularities within a given sample. In practice, we must therefore decide how to characterize structures within a set of data and how to search for them?

The famous k-means algorithm arguably provides the most popular answers to these questions. It represents structures in terms of *subsets* of X and attempts to divide the data into k different *clusters* $C_i \subset X$ which meet the following criteria:

1. clusters should be pairwise disjoint, i.e. $C_i \cap C_j = \emptyset$ for $i \neq j$
2. clusters should cover the data such that $C_1 \cup C_2 \cup \ldots \cup C_k = X$
3. most importantly, data assigned to a cluster C_i should be *similar*.

While the first two criteria are well defined, the third one hinges on the rather intuitive notion of similarity. Indeed, there are many possible definitions of similarity and characteristic differences between clustering algorithms usually trace back to different notions of what it means to be similar.

The idea employed in k-means clustering is to represent each cluster C_i by means of a *centroid* $\boldsymbol{\mu}_i \in \mathbb{R}^m$ and to measure similarities in terms of Euclidean distances between centroids and data points. Hence, two data points are considered similar if their distances to a common centroid are small and a point $\boldsymbol{x}_j$ will be assigned to cluster C_i if its distance to $\boldsymbol{\mu}_i$ is less than that to any other centroid.

cluster centroids

THIS NOTION OF SIMILARITY reduces the problem of clustering to the problem of finding appropriate centroids. This, in turn, can be cast as the task of minimizing the following objective function

$$E(k) = \sum_{i=1}^{k} \sum_{\boldsymbol{x}_j \in C_i} \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_i \right\|^2 \tag{1}$$

where the minimization has to be carried out w.r.t. k centroids $\boldsymbol{\mu}_i$. Thus, minimizing (1) is to determine suitable centroids $\boldsymbol{\mu}_i$ such that, if the data are partitioned into corresponding clusters C_i, distances between data points and their closest cluster centroid become as small as possible.

Already at this point, we note that the objective in (1) can also be expressed in terms of *indicator variables* z_{ij} which register for any $\boldsymbol{x}_j$ whether it belongs to cluster C_i. That is, by defining

indicator variables

$$z_{ij} = \begin{cases} 1, \text{ if } \boldsymbol{x}_j \in C_i \\ 0, \text{ otherwise} \end{cases} \tag{2}$$

we find the above objective function to be equivalent to

$$E(k) = \sum_{i=1}^{k} \sum_{j=1}^{n} z_{ij} \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_i \right\|^2. \tag{3}$$

ALTHOUGH BOTH OBJECTIVES LOOK RATHER INNOCENT and formalize an intuitive problem, finding an optimal solution, i.e. finding global minimizers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$, is actually NP-hard[1]. Even for 2D data and settings where $k = 2$, it cannot be guaranteed that the best possible solution will be found in reasonable time. This is important to know! It tells us that **the k-means algorithm is merely a heuristic for dealing with a surprisingly hard problem**.

[1] D. Aloise, A. Deshapande, P. Hansen, and P. Popat. NP-Hardness of Euclidean Sum-of-Squares Clustering. *Machine Learning*, 75(2), 2009

The Classical k-means Algorithm

So, how does the k-means algorithm minimize the above objectives? In a nutshell, it realizes a greedy iterative update scheme. When started, say at iteration $t = 0$, it randomly initializes the parameters $\boldsymbol{\mu}_1^{(t)}, \boldsymbol{\mu}_2^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$. Given this initial guess, the data are then clustered accordingly. That is, the algorithm determines k clusters where

$$C_i^{(t)} = \left\{ \boldsymbol{x}_j \in X \,\middle|\, \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_i^{(t)} \right\|^2 \leq \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_l^{(t)} \right\|^2 \forall\, l \neq i \right\} \tag{4}$$

Once clusters have been determined, the algorithm updates the current estimates of the cluster centroids by computing

$$\boldsymbol{\mu}_i^{(t+1)} = \frac{1}{n_i} \sum_{\boldsymbol{x}_j \in C_i^{(t)}} \boldsymbol{x}_j \tag{5}$$

where $n_i = |C_i^{(t)}|$ denotes the number of elements in cluster C_i. As these updates may lead to a new clustering, the iteration counter is increased to $t = t + 1$ and steps (4) and (5) are repeated until the

---

$t \leftarrow 0$
done $\leftarrow$ False
randomly initialize $\boldsymbol{\mu}_1^t, \boldsymbol{\mu}_2^t, \ldots, \boldsymbol{\mu}_k^t$
while not done **do**
  // for each mean, update its cluster
  for $i = 1$ to k **do**
    $C_i^t = \left\{ \boldsymbol{x} \in X \,\middle|\, \|\boldsymbol{x} - \boldsymbol{\mu}_i^t\| \leq \|\boldsymbol{x} - \boldsymbol{\mu}_j^t\| \right\}$
  if $C_i^t \cap C_i^{t-1} = C_i^t \; \forall\, i$ **then**
    done $\leftarrow$ True

  // for each cluster, update its mean
  for $i = 1$ to k **do**
    $n_i = |C_i^t|$
    $\boldsymbol{\mu}_i^{t+1} = \frac{1}{n_i} \sum_{\boldsymbol{x} \in C_i^t} \boldsymbol{x}$
  if $\left\|\boldsymbol{\mu}_i^{t+1} - \boldsymbol{\mu}_i^t\right\| \leq \epsilon \; \forall\, i$ **then**
    done $\leftarrow$ True

  $t \leftarrow t + 1$
  if $t > t_{\max}$ **then**
    done $\leftarrow$ True

end while

Figure 2: Pseudo code for k-means.

assignment of data points to clusters does not change anymore or t exceeds a predefined number of iterations (see Fig. 3 for an example).

IT IS EASY TO SHOW that updating each centroid to the *sample mean* of its cluster will never increase the value of objectives in (1) or (3)[2]. Each iteration therefore improves on the previous result and k-means clustering usually converges quickly.

[2] We will prove this in a later note.

However, as the algorithm starts from a random initialization, it cannot be guaranteed to find the global minimum of its objective function. As it typically converges to a local minimum, **it is pivotal to run k-means several times so as to empirically determine good clusters**. Yet, even then, k-means may produce questionable results when seeded with inappropriate centroids. Intelligent initializations are the topic of ongoing research[3,4] and **arbitrary initialization are considered harmful; initial centroids should at least be selected among the available data points**.

[3] P.S. Bradley and U.M. Fayyad. Refining Initial Points for K-Means Clustering. In *Proc. ICML*, 1998

[4] D. Arthur and S. Vassilvitskii. k-means++: The Advantages of Careful Seeding. In *Proc. SODA*, 2007

Common Pitfalls in Practical Applications

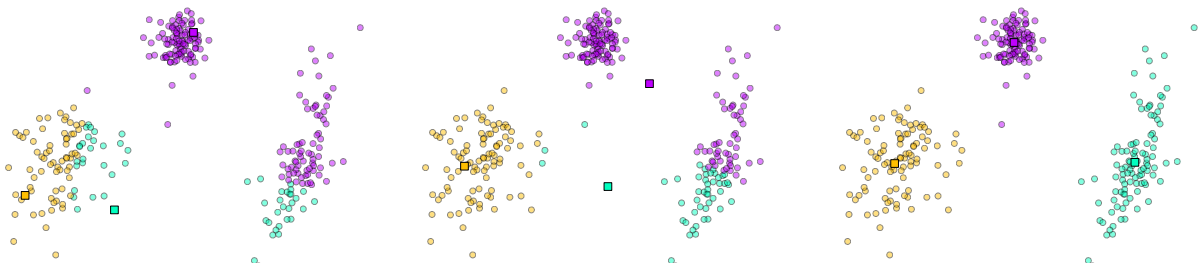
Before we continue to study further theoretical aspects of k-means clustering, it seems instructive to discuss when and how to apply the k-means algorithm in practice.

It can be shown that the k-means algorithm is tailored to locally Gaussian data[5]. Hence, **if k-means clustering is applied to data sets that are not composed of compact convex subsets, it is overly optimistic to expect the algorithm to identify reasonable clusters**.

[5] We will do this in a later note.

The following example illustrates what this may mean in practice. Figure 4(a) displays a set of 2D data which apparently consists of two distinct, linear clusters. Yet, if we set $k = 2$ and run k-means, the result will likely look like in Fig. 4(b). In particular, cluster centroids (□) will be rather far from actual data points but form the centers of two imaginary circles which each contain about half the data. This practical result therefore illustrates the tendency of the k-means algorithm to produce compact, blob-like clusters. In other words, for data such as in this example, the algorithm will easily determine clusters but its results will hardly agree with what is salient to humans.

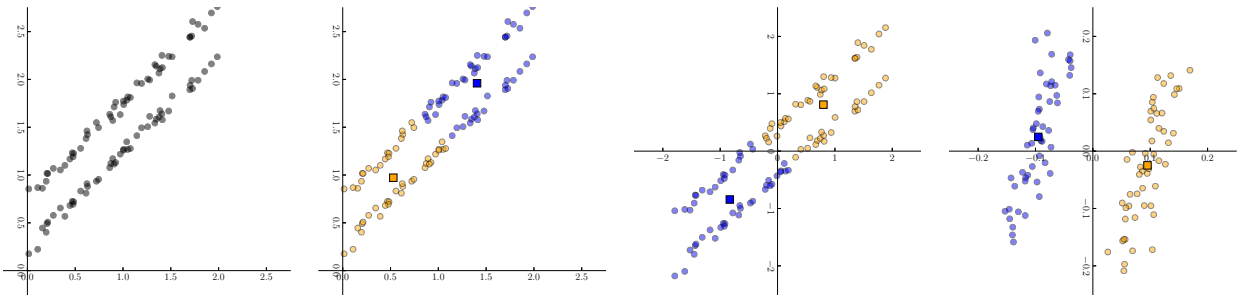
AS A REMEDY, the literature frequently suggests to normalize the



(a) initial centroids and clustering (b) situation after first update (c) situation upon convergence

Figure 3: k-means clustering applied to determine $k = 3$ clusters in the data (○) from Fig. 1; (a) initial centroids (□) and corresponding clustering, (b) situation after the first update, and (c) result upon convergence.

As k-means is tailored to Gaussian mixtures and since the initial choice of centroids was favorable, it took only four iterations to converge to this reasonable solution. However, there is no guarantee for k-means to always behave this way. In practice, one should therefore run it several times and base any further analysis on the result with the lowest value of the objective in Eq. (1).



(a) two apparent 2D clusters (b) clusters found by k-means (c) clusters after normalization (d) clusters after whitening

Figure 4: Example of the limitations of k-means clustering. As it implicitly assumes locally Gaussian data, the algorithm clusters data into compact convex subsets.

The obvious linear substructures in this example can not be uncovered by k-means. Generally, data needs to be whitened so that the algorithm stands a better chance of producing results close to human intuition. However, there is no guarantee for k-means to succeed in this regard.

data using $y_{dj} = (x_{dj} - \mu_d)/\sigma_d^2$ where x_{dj} is the dth component of vector $\boldsymbol{x}_j$ and μ_d and σ_d^2 denote the data's mean and variance along dimension d. Afterwards, the data will be of zero mean and unit variance and are commonly believed to be more amenable to k-means. But Fig. 4(c) shows that this is not necessarily the case. For our example, the transformation is too naïve to have a noticeable effect on the result; the data are simply not Gaussian enough for k-means to identify their "true" latent substructures.

THEREFORE, if we insist on applying the k-means algorithm to non-Gaussian data, we need better preprocessing. A reasonable idea is to *whiten* the data.

data whitening

Computing the sample mean $\boldsymbol{\mu} = \frac{1}{n}\sum_j \boldsymbol{x}_j$ and transforming the data to zero-mean $\boldsymbol{y}_j = \boldsymbol{x}_j - \boldsymbol{\mu}$ yields the covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{Y}\boldsymbol{Y}^T$. Using principal component analysis, $\boldsymbol{\Sigma}$ can be decorrelated so that $\boldsymbol{\Lambda} = \boldsymbol{U}^T\boldsymbol{\Sigma}\boldsymbol{U}$. Transformed vectors $\boldsymbol{z}_j = \boldsymbol{U}^T\boldsymbol{y}_j$ thus have a diagonal covariance matrix $\boldsymbol{\Lambda}$. Still, variances along different dimensions will usually differ; but since $\boldsymbol{\Lambda}^{-1/2}\boldsymbol{\Lambda}\boldsymbol{\Lambda}^{-1/2} = \boldsymbol{I}$, the transformation

$$\boldsymbol{w}_j = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{U}^T(\boldsymbol{x}_j - \boldsymbol{\mu}) \tag{6}$$

maps the data to vectors $\boldsymbol{w}_j$ for which $\frac{1}{n}\sum_j \boldsymbol{w}_j\boldsymbol{w}_j^T = \boldsymbol{I}$. That is, after whitening, variances in the directions of the principal axes of the data are identically 1. Geometrically, we can picture this transformation as mapping the data (close) to the surface of a sphere where possible clusters will be more separated than in the original representation.

Figure 4(d) shows the whitened data and illustrates the corresponding effect on k-means clustering. Based on this example, it appears that **proper preprocessing enhances the chances for the k-means algorithm to identify structures that are not necessarily Gaussian**. Nevertheless, there is still no guarantee for it to do so.

THERE IS YET ANOTHER PATHOLOGICAL SITUATION, namely the case of very high dimensional data where we are given n data points $\boldsymbol{x}_j \in \mathbb{R}^m$ but $n \ll m$.

HDLSS data

For *high dimension low sample size data* like these, the notion of similarity invoked by k-means may be useless, because as dimensions grow so do distances and it becomes very likely that all the given

data are equally far apart [6,7,8]. Hence, centroids determined by k-means clustering will be far from any data point and the decision of whether to assign a data point to a cluster will basically depend on minuscule random variations rather than on structural properties. Therefore, **for very high dimensional data, k-means clustering should be applied only after dimensionality reduction**.

Notes and References

The origins of the idea of k-means clustering can be traced back to work by Steinhaus[9] in the 1950s. The standard algorithm, that is the iterative algorithm discussed in this note, was first discussed by Lloyd in a Bell Labs technical report also in the 1950s but openly published only much later[10]. Finally, the now so commonly used term "k-means" was coined by MacQueen[11] in the 1960s.

For readers interested in deeper expositions than the ones in this and our subsequent notes on the topic, we recommend the rigorous and thorough accounts by Bishop[12] or MacKay[13].

[6] D.L. Donoho and J. Tanner. Neighborliness of Randomly Projected Simplices in High Dimensions. *PNAS*, 102(27), 2005

[7] P. Hall, J.S. Marron, and A. Neeman. Geometric Reprsentations of High Dimension, Low Sample Size Data. *J. Royal Statistical Society B*, 67(3), 2005

[8] F. Murtagh. The Remarkable Simplicity of Very High Dimensional Data: Applications of Model-based Clustering. *J. of Classification*, 26(3), 2009

[9] H. Steinhaus. Sur la Division des Corps Matériels en Parties. *Bulletin Polish Acad. of Sciences*, 4(12):801–804, 1957

[10] S.P. Lloyd. Least Squares Quantization in PCM. *IEEE Trans. on Information Theory*, 28(2), 1982

[11] J.B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. Berkeley Symp. on Mathematical Statistics and Probability*, 1967

[12] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006

[13] D.J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003