Lecture Node for K-means

Zhi Jing, 18342040
School of Data and Computer Science
Sun Yat-sen University
jingzh5@mail2.sysu.edu.cn

Abstract

In this work, I thoroughly explain the K-means clustering algorithm and its application, as well as its variants in accordance with more recent research achievement. Their efficacy is investigated based on experimental analysis, in the area of computer vision specifically, with a thorough comparison. Also, in this paper, an introduction of the models hidden behind k-means is provided, like GMM, EM, which is expected to give you an alternative view of K-means.

As is acknowledged, K-means algorithm is considered one of the most popular algorithms in the research community. And because it has been decades since it came out to world(Lloyd, 1982)[20], K-means suffers from certain limitations, even fundamental drawbacks, when dealing with more sophisticated data and tasks, like center initialization problem leading to unexpected convergence. Also, the number of clusters is required to be defined beforehand, leading to different cluster shapes and outlier effects. What this paper provides is a structured overview of research conducted on the K-means to tackle with shortcomings mentioned above, outlining a thorough understanding of the k-means algorithm through multiple perspectives along with its different research directions and applications.

1. Introduction

K-means algorithm is a method of clustering, originated from signal processing, and now is widely used in Machine Learning (ML), which is, generally, comprised of two major categories, supervised and unsupervised learning. In supervised learning, it is designed to learn a function mapping a given input to an output based on the data of input–output pairs, which means that supervised learning requires the availability of data labeled with the desired output value. However, in most cases, it can be expensive, sometimes challenging, to obtain a large amount of labeled data. Unsupervised learning thus becomes popular and often used when facing a lack of labeled data because it requires no prior knowledge of the data by making assumptions or reasonable inferences. And on top of that, a method called semi-supervised learning is proposed in the recent research to compensate the inevitable accuracy loss that unsupervised learning suffers from due to the inadequacy of data by adding proper amount of labeled data to derive big accuracy improvements. And this paper focuses a thorough discussion on K-means algorithm, which falls into the category of unsupervised learning of clustering.

Clustering algorithms are designed to exploit the underlying structure of the data distribution and define rules for grouping the data with similar characteristics, making unlabeled data into meaningful data into groups of clusters and identifying hidden patterns from large data collections, without any use of prior knowledge about the datasets. Each cluster consists of similar data instances that are dissimilar enough to the instances from other clusters under an ideal clustering scenario. The key to such measure is the data itself having a hidden driving pattern and objective of the algorithm. And to further understand such hidden pattern, I provide other methods to describe and discover them on top of the discussion of K-means itself.

Many data-driven applications need clustering and it is used in many areas include image segmentation, information retrieval, document classification, associate rule mining. It is also studied in statistics, pattern recognition, computational geometry, bioinformatics, optimization, image processing, and in a variety of other fields. Table 1 shows the recent applications on K-means clustering [21] in different application domains. And some of these methods are discussed in the following sections.

Furthermore, after a review of K-means, in order to study the true nature of why K-means algorithm can be applied on data clustering, I introduce the latent variable view of mixture distributions in which the discrete latent variables can be interpreted as defining assignments of data points to specific components of the mixture. A general technique for finding maximum likelihood estimators in latent variable models is the expectation-maximization (EM) algorithm, where we can easiy see that K-means algorithm corresponds to a particular non-probabilistic limit of EM

Application Area	Algorithm Name
Multiview data	k-means
Data Summarization	Modified x-means
Endpoint detection	k-means for realtime detection
Big data	Privacy preserving k-means
Mobile health	k-means implemented with CORDIC
DDoS detection	Semi-supervised k-means algorithm with hybrid feature
Image compression	k-means cuckoo optimization
Sound source angle estimation	Neural network based on global k-means
Social tags	k-means based on latent semantic analysis
Sensing for IGBT current	k-means with neural network
Image segmentation	Kernel k-means Nystrom approximation
Computational complexity	Multiple kernel k-means with late fusion
Optimization	Non alternating stochastic k-means
Text processing	Vanilla k-means
High dimensional data processing	Fast adaptive k-means (FAKM)
Adaptive clustering	Fuzzy k-means with S-distance
Mobile storage positioning	Potential k-means
Load pattern	Hierarchical k-means (H-Kmeans)
Wireless sensor networks	Distributed k-means and fuzzy c-means
Partial multiview data	Weighted k-means
Wind power forecasting	k-means with bagging neural network
Image processing	A hybrid parallelization of k-means algorithm
Shape recognition	Fuzzy k-means clustering ensemble (FKMCE)
Signal processing	Compressive k-means clustering (CKM)
Face detection	Symmetry-based version of k-means (SBKM)

Table 1. Example of a short caption, which should be centered.

applied to mixtures of Gaussians. Therefore, we shall get a full vision of the true nature of K-means.

2. Models

In section 2, I introduce and describe the theoretical principle of the models and the connection between them at last.

2.1. K-means

We shall begin by the problem of identifying clusters of data points in a multidimensional space. Suppose we have a data set $\{X_1, ..., X_N\}$ consisting of N observations of a random D-dimensional Euclidean variable x. The goal is to partition the data set into some certain number K of clusters. Here, We shall suppose, for the moment, that the value of K is given. It is worth noting that intuitively we might think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster. We can formalize this notion by first introducing a set of D-dimensional vectors $\boldsymbol{\mu}_k$, where $k = 1, ..., K$, in which $\boldsymbol{\mu}_k$ is a prototype associated with the k-th cluster. As we shall see shortly, we can think of the $\boldsymbol{\mu}_k$ as representing the centers of the clusters. Our goal is then to find an assignment of data points to clusters, as well as a set of vectors $\{\boldsymbol{\mu}_k\}$, such that the sum of the squares of the distances of each data point to its closest vector $\boldsymbol{\mu}_k$, is a minimum. It is convenient at this point to define some notation to describe the assignment of data points to clusters. For each data point X_n, we introduce a corresponding set of binary indicator variables $r_{nk} \in \{0, 1\}$, where $k = 1, ..., K$ describing which of the K clusters the data point X_n is assigned to, so that if data point X_n is assigned to cluster k then $r_{nk} = 1$, and $r_{nj} = 0$ for $j \neq k$. This is known as the 1-of-K coding scheme. We can then define an objective function, sometimes called a distortion measure, given by

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| \boldsymbol{x}_n - \boldsymbol{\mu}_k \right\|^2 \tag{1}$$

which represents the sum of the squares of the distances of each data point to its assigned vector $\boldsymbol{\mu}_k$. Our goal is, then, to find values for the $\{r_{nk}\}$ and the $\{\boldsymbol{\mu}_k\}$ so as to minimize J.

This issue is addressed through an iterative procedure where each iteration involves two successive steps corresponding to successive optimizations with respect to the r_{nk} and the $\boldsymbol{\mu}_k$. First we choose some initial values for $\boldsymbol{\mu}_k$.

Then in the first phase we minimize J with respect to the r_{nk}, keeping the $\boldsymbol{\mu}_k$ fixed. In the second phase we minimize J with respect to the $\boldsymbol{\mu}_k$, keeping r_{nk} fixed. This two-stage optimization is then repeated until convergence. We shall see that these two stages of updating r_{nk} and updating $\boldsymbol{\mu}_k$ correspond respectively to the E (expectation) and M (maximization) steps of the EM algorithm, and to emphasize this we shall use the terms E step and M step in the context of the K-means algorithm.

Let's then see the mathematical principle behind these 2 steps. Consider first the determination of the r_{nk}. Because J in (1) is a linear function of r_{nk}, this optimization can be performed easily to give a closed form solution. The terms involving different n are independent and so we can optimize for each n separately by choosing r_{nk} to be 1 for whichever value of k gives the minimum value of $\|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2$. In other words, we simply assign the n data point to the closest cluster centre. More formally, this can be expressed as

$$r_{nk} = \begin{cases} 1 & if\ k = argmin_j \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2 \\ 0 & otherwise \end{cases} \tag{2}$$

Now consider the optimization of the $\boldsymbol{\mu}_k$ with the r_{nk} held fixed. The objective function J is a quadratic function of $\boldsymbol{\mu}_k$, and it can be minimized by setting its derivative with respect to $\boldsymbol{\mu}_k$ to zero giving

$$2\sum_{n=1}^{N} r_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0 \tag{3}$$

which we can easily solve for $\boldsymbol{\mu}_k$ to give

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\boldsymbol{x}_n}{\sum_n r_{nk}} \tag{4}$$

And we can see that in this expression, the denominator is equal to the number of points assigned to cluster k, and so this result has a simple interpretation, namely set $\boldsymbol{\mu}_k$ equal to the mean of all of the data points $\boldsymbol{x}_n$ assigned to cluster k. And this is the reason why the procedure is known as the K-means algorithm. The two phases of re-assigning data points to clusters and re-computing the cluster means are repeated in turn until there is no further change in the assignments (or until some maximum number of iterations is exceeded). Because each phase reduces the value of the objective function J, convergence of the algorithm is assured.

However, it may converge to a local rather than global minimum of J. The convergence properties of the K-means algorithm were studied by MacQueen(1967). Also, a direct implementation of the K-means algorithm as discussed here can be relatively slow, because in each E step it is necessary to compute the Euclidean distance between every prototype vector and every data point. As we shall see, these problems are addressed in the discussion in the following subsection. Moreover, useing squared Euclidean distance as the measure of dissimilarity between a data point and a prototype vector will not only has this limit on the type of data variables that can be considered, but also make the determination of the cluster means non-robust to outliers. We can generalize the K-means algorithm by introducing a more general dissimilarity measure V(x,x') between two vectors x and x' and then minimizing the following distortion measure

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk} V(\boldsymbol{x}_n, \boldsymbol{\mu}_k) \tag{5}$$

which gives the $K - medoids$ algorithm. Accordingly, E step and M step need some adjustment, where E step is to assign each data point to the cluster for which the dissimilarity to the corresponding prototype is smallest and M step, more complex than the former, is common to restrict each cluster prototype to be equal to one of the data vectors assigned to that cluster. The computational cost of E step is $O(KN)$, as is also the case for the standard K-means algorithm, while the cost of M step is $O(N_k^2)$ after adjusting.

Here, we conclude our discussion on K-means itself in this section. One notable feature of the K-means algorithm is that at each iteration, every data point is assigned uniquely to one, and only one, of the clusters. Whereas some data points will be much closer to a particular center $\boldsymbol{\mu}_k$ than to any other centre, there may be other data points that lie roughly midway between cluster centres. In the latter case, it is not clear that the hard assignment to the nearest cluster is the most appropriate. We shall see in the following section that by adopting a probabilistic approach, we obtain 'soft' assignments of data points to clusters in a way that reflects the level of uncertainty over the most appropriate assignment.

2.2. Special techniques and typical variants of K-means

We continue our discussion on K-means by showing some special techniques and variants of K-means in this section.

2.2.1 optimization

So far, we have considered a batch version of K-means in which the whole data set is used together to update the prototype vectors. We can also derive an on-line stochastic algorithm[21] by applying the Robbins-Monro[29] procedure to the problem of finding the roots of the regression function given by the derivatives of J in (1) with respect to $\boldsymbol{\mu}_k$. This leads to a sequential update in which, for each data

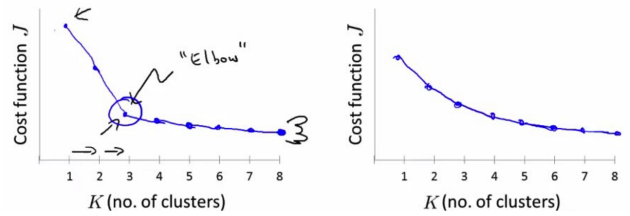

Figure 1. Examples of Elbow point choosing

point $\boldsymbol{x}_n$ in turn, we update the nearest prototype $\boldsymbol{\mu}_k$ using

$$\boldsymbol{\mu}_k^{new} = \boldsymbol{\mu}_k^{old} + \eta_n(\boldsymbol{x}_n - \boldsymbol{\mu}_k^{old}) \tag{6}$$

where η_n is the learning rate parameter, which is typically made to decrease monotonically as more data points are considered.

Also, various schemes have been proposed for speeding up the K-means algorithm, some of which are based on precomputing a data structure such as a tree such that nearby points are in the same subtree (Ramasubramanian and Paliwal, 1990[27]; Moore, 2000[23]). Other approaches make use of the triangle inequality for distances, thereby avoiding unnecessary distance calculations (Hodgson, 1998[14]; Elkan, 2003[11]).

2.2.2 Choosing the value of k

Then, let's see the technique of properly choosing the value of K as I left undecided in the previous discussion.

Although this choosing procedure is strongly related to the very purpose of using K-means in the first place, we are here just to consider this problem under a general situation with no specific context. An universal but usually not so effective choice of K is the root of $n/2$, namely $K = \sqrt{n/2}$.

Another method to choose a proper K is the $Elbow\ Method$[17]. The idea is that Start with K=2, and keep increasing it in each step by 1, calculating your clusters and the cost that comes with the training. At some value for K the cost drops dramatically, and after that it reaches a plateau when you increase it further. This is the K value you want. It works on the principal that after a certain number of K clusters, the difference in SSE (Sum of Squared Errors) starts to decrease and diminishes gradually. Here, the WCSS(Within Cluster Sum of Squared errors) metric is used as an indicator of the same. Hence, the K value, specifies the number of clusters. The first picture of Figure 1 shows clearly the elbow point. However, sometimes the data gives no sign of a clear elbow point as shown in the second picture of Figure 1. A algebra expression of $Elbow\ Method$ is provided:

$$k = argmin_K \sum_{i=1}^{K} \sum W(c_i)^2 \tag{7}$$

where C_i is the i-th cluster and $W(c_i)$ is the within-cluster variation.

In addition, we shall talk about another method to choose k, using the Silhouette. A number of approaches utilize indexes comparing within-cluster distances with between cluster distances: the greater the difference the better the fit; many of them are mentioned in Milligan and Cooper [12]. Two of the indexes are: (a) the point-biserial correlation, that is, the correlation coefficient between the entity-to-entity distance matrix and the binary partition matrix assigning each pair of the entities 1, if they belong to the same cluster, and 0, if not, and (b) its ordinal version proposed by Hubert and Levin [15]. A well-balanced coefficient, the silhouette width, which has shown good performance in experiments, was introduced by Kaufman and Rousseeuw [16, 26]. The concept of silhouette width involves the difference between the within-cluster tightness and separation from the rest. Specifically, the silhouette width s(i) for entity i ∈ I is defined as

$$s(i) = \frac{b(i) - a(i)}{\max a(i), b(i)} \tag{8}$$

Where $a(i)$ is the average distance between i and all other entities of the cluster to which i belongs and $b(i)$ is the minimum of the average distances between i and all the entities in each other cluster. The silhouette width values lie in the range from -1 to 1. If the silhouette width value for an entity is about zero, it means that that the entity could be assigned to another cluster as well. If the silhouette width value is close to -1, it means that the entity is wrongly classified. If all the silhouette width values are close to 1, it means that the set I is well clustered. A clustering can be characterized by the average silhouette width of individual entities. The largest average silhouette width, over different K, indicates the best number of clusters. There are other methods to determine the value of K as well, like Information Criterion Approach[1, 2, 8, 28, 31], Cross-validation[32, 18, 4, 19, 3, 34].

2.2.3 Fuzzy C Means

Although there are a lot of them as I list on the Table 1, I want to specify on one particular variant of K-means, Fuzzy C Means (FCM)[10, 6, 5], as you might have heard of it before. Don't get confused, the 'c' of FCM here has exactly the same meaning of 'k' of K-means.

In fuzzy clustering the objects are assigned to K fuzzy clusters $\hat{S}_k(k = 1, ..., K)$ with a corresponding degree of membership for each cluster. Its objective is to minimize the weighted sum of Euclidean distances between the objects i and the means $\boldsymbol{\mu}_k$ of the corresponding clusters k. The weights[25] are derived from the membership degrees of

the objects to the clusters, as shown below:

$$w = min \sum_{i=1}^{N} \sum_{k=1}^{K} \lambda_{i,k}^{m} \left\| \boldsymbol{x}_n - \boldsymbol{\mu}_k \right\|^2 \tag{9}$$

with $\lambda_{i,k}$ the membership degree of object i to cluster k:

$$\lambda_{i,k} = \frac{1}{\sum_{j=1}^{K} (\frac{d(\boldsymbol{x}_i, \boldsymbol{\mu}_k)}{d(\boldsymbol{x}_j, \boldsymbol{\mu}_k)})^{\frac{2}{m-1}}} \tag{10}$$

and $m \in (1, \infty)$ the fuzzifier parameter which defines the clusters' fuzziness. For $m \to \infty$, all membership degrees converge towards $1/K$, the most fuzziest result possible. For $m \to 1$ each membership degree $\lambda_{i,k}$ converges towards 0 or to 1, i.e. fuzzy c-means provides hard results as in classic k-means. Selecting an appropriate m is challenging and depends on the given data as well as on the expected results. A still well accepted compromise is to set $m = 2.0$ [24]. However, studies also show that this selection is not appropriate for some applications. Further methods that address the selection of m have been discussed by Yu *et al*. [36] and Wu [35]. The membership degrees $\lambda_{i,k}$ show how representative an object i is for cluster k with $\lambda_{i,k} \in [0, 1] \forall i, k$. A membership degree of 0 indicates "not representative at all" while a membership degree of 1 indicates that the object represents the cluster perfectly. The sum of the membership degrees of an object i to all clusters is required to equal 1 in order to assure the algorithm's convergence:

$$\sum_{k=1}^{K} \lambda_{i,k} = 1 \; \forall i = 1, ..., N \tag{11}$$

Overlapping classes constitute a main application area of fuzzy clustering. Furthermore, fuzzy membership degrees are very suitable when, e.g., in complex decision processes, information from different sources has to be aggregated to arrive at a final decision. Optionally, after performing fuzzy c-means, the results can be defuzzified in a final step of the analysis when hard decisions are required.

2.3. An alternative view

2.3.1 GMM and EM algorithm

In this section, we shall begin our discussion on Gaussian Mixture Model (GMM) and Expectation-Maximization (EM) algorithm, where I introduce a view of latent variables[7, 13]. I want to start at the formulation of Gaussian mixtures in terms of discrete latent variables. This will provide us with a deeper insight into this important distribution, and will also serve to motivate the expectation-maximization algorithm. First, the Gaussian mixture distribution can be written as a linear superposition of Gaussians in the form:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k N(x | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{12}$$

where $\boldsymbol{\mu}_k$ is the mean of the k-th Gaussian distribution and $\boldsymbol{\Sigma}_k$ is the covariance and parameters π_k are called $mixing\ coefficients$.

Let us introduce a K-dimensional binary random variable z having a 1-of-K representation in which a particular element z_k is equal to 1 and all other elements are equal to 0. The values of z_k therefore satisfy $z_k \in \{0, 1\}$ and $\sum_k z_k = 1$, and we see that there are K possible states for the vector z according to which element is nonzero. Due to the restriction of this assignment, I would directly propose that the marginal distribution of x is also a Gaussian mixture of the form (12). So is to say, if we have several observations $x_1, ..., x_N$, then, because we have represented the marginal distribution in the form $p(x) = \sum_z p(\boldsymbol{x}, \boldsymbol{z})$, it follows that for every observed data point x_n there is a corresponding variable z_n, namely the latent variable. The prove of this proposal can be found in [7].

Therefore, we have found an equivalent formulation of the Gaussian mixture involving an explicit latent variable. And this result is amazing though it might not be so obvious for the moment. Let me give a hint: we are now able to work with the joint distribution $p(x, z)$ instead of the marginal distribution $p(x)$ and this will lead to significant simplifications, most notably through the introduction of the expectation-maximization (EM) algorithm.

Also, Another quantity that will play an important role is the conditional probability of z given x. We shall use $\gamma(z_k)$ to denote $p(z_k = 1|x)$, whose value can be found using Bayes' theorem:

$$\gamma(z_k) = \frac{\pi_k N(x | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j N(x | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{13}$$

So let us continue to the EM algorithm. An elegant and powerful method for finding maximum likelihood solutions for models with latent variables is called the expectation-maximization algorithm, or EM algorithm (Dempster *et al*. [9]; McLachlan and Krishnan [22]). We shall motivate the EM algorithm by giving a relatively informal treatment in the context of the Gaussian mixture model. I emphasize, however, that EM has broad applicability, and indeed it can be encountered in the context of a variety of different models though not in this paper.

Let us begin by writing down the conditions that must be satisfied at a maximum of the likelihood function. Setting the derivatives of $\ln p(X|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ in (25) with respect to the means $\boldsymbol{\mu_k}$ of the Gaussian components to zero, we obtain

$$0 = -\sum_{n=1}^{N} \frac{\pi_k N(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j N(\boldsymbol{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k (\boldsymbol{x}_n - \boldsymbol{\mu}_k) \tag{14}$$

And then, multiplying by $\boldsymbol{\Sigma^{-1}}$, which we assume to be non-

singular just for convenience and rearranging we obtain

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}\boldsymbol{x}_n) \tag{15}$$

Equivalently we also have

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{16}$$

We can interpret N_k as the effective number of points assigned to cluster k. If we set the derivative of $\ln p(X|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to $\boldsymbol{\Sigma}_k$ to zero, and follow a similar line of reasoning, making use of the result for the maximum likelihood solution for the covariance matrix of a single Gaussian, we get

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \tag{17}$$

which has the same form as the corresponding result for a single Gaussian fitted to the data set, but again with each data point weighted by the corresponding posterior probability and with the denominator given by the effective number of points associated with the corresponding component. Finally, we maximize $\ln p(X|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to the mixing coefficients π_k. Here we must take account of the constraint $\sum_{k=1}^{K} \pi_k = 1$, which requires the mixing coefficients to sum to one. This can be achieved using a Lagrange multiplier λ and maximizing the following quantity

$$\ln p(X|\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \lambda(\sum_{k=1}^{K} \pi_k - 1) \tag{18}$$

which gives

$$\begin{aligned} 0 &= \sum_{n=1}^{N} \frac{N(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j N(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \\ &= \sum_{n=1}^{N} \frac{N(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\gamma(z_{nk})} + \lambda \end{aligned} \tag{19}$$

If we now multiply both sides by π_k and sum over k, we find λ = -N. Using this to eliminate λ and rearranging we obtain

$$\pi_k = \frac{N_k}{N} \tag{20}$$

so that the mixing coefficient for the k-th component is given by the average responsibility which that component takes for explaining the data points.

These results do suggest a simple iterative scheme for finding a solution to the maximum likelihood problem, which as we shall see turns out to be an instance of the EM algorithm for the particular case of the Gaussian mixture model. We first choose some initial values for the means, covariances, and mixing coefficients. Then we alternate between the following two updates that we shall call the E step and the M step. Here is the reason. In the expectation step, or E step, we use the current values for the parameters to evaluate the posterior probabilities, or responsibilities, given by $\gamma(r_{nk})$. We then use these probabilities in the maximization step, or M step, to re-estimate the means, covariances, and mixing coefficients using the results (17)(20)(23). To be more specific, we first evaluate the new means using (17) and then use these new values to find the covariances using (20), in keeping with the corresponding result for a single Gaussian distribution. In practice, the algorithm is deemed to have converged when the change in the log likelihood function, or alternatively in the parameters, falls below some threshold. Now we have a clear vision of EM on GMM.

2.3.2 Review of connections between EM, GMM and K-means

As the discussion above, we know what is GMM and EM in the context of GMM through my introduction. These mathematical proof might be dull yet valuable and can provide us with an alternative view of K-means. Recall that the two stages of K-means, which is updating r_{nk} and updating $\boldsymbol{\mu}_k$ correspond respectively to the E (expectation) and M (maximization) steps of the EM algorithm as I wrote in section 2.1. Let us see exactly how these 2 models react to each other. Comparison of the K-means algorithm with the EM algorithm for Gaussian mixtures shows that there is a close similarity. Whereas the K-means algorithm performs a hard assignment of data points to clusters, in which each data point is associated uniquely with one cluster, the EM algorithm makes a soft assignment based on the posterior probabilities. In fact, we can derive the K-means algorithm as a particular limit of EM for Gaussian mixtures as follows. Consider a Gaussian mixture model in which the covariance matrices of the mixture components are given by $\epsilon\boldsymbol{I}$, where ϵ is a variance parameter that is shared by all of the components, and $\boldsymbol{I}$ is the identity matrix, so that

$$p(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) = \frac{1}{(2\pi\epsilon)^{1/2}} \exp\{-\frac{1}{2\epsilon} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2\} \tag{21}$$

We now consider the EM algorithm for a mixture of K Gaussians of this form in which we treat ϵ as a fixed constant, instead of a parameter to be re-estimated. From $\gamma(z_{nk})$ the posterior probabilities, or responsibilities, for a particular data point xn, are given by

$$\gamma(z_{nk}) = \frac{\pi_k \exp\{-\frac{1}{2\epsilon} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2\}}{\sum_j \pi_j \exp\{-\frac{1}{2\epsilon} \left\|\boldsymbol{x}_n - \boldsymbol{\mu}_j\right\|^2\}} \tag{22}$$

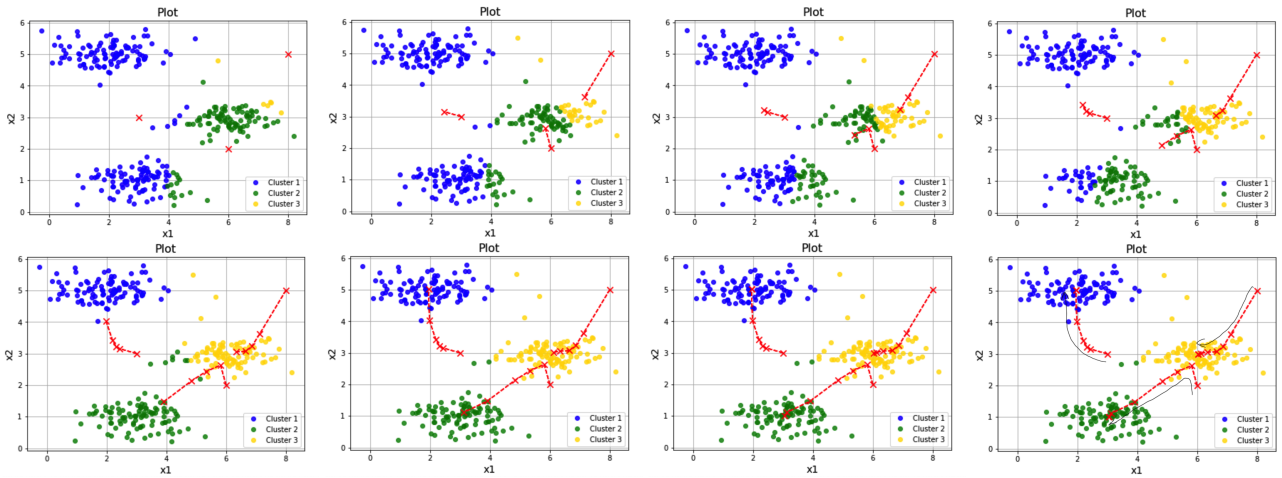

Figure 2. All 8 iterations of K-means on data point with k = 3

If we consider the limit $\epsilon \to 0$, we see that in the denominator the term for which $\|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2$ is smallest will go to zero most slowly, and hence the responsibilities $\gamma(z_{nk})$ for the data point x_n all go to zero except for term j, for which the responsibility $\gamma(z_{nk})$ will go to unity. Note that this holds independently of the values of the π_k so long as none of the π_k is zero. Thus, in this limit, we obtain a hard assignment of data points to clusters, just as in the K-means algorithm, so that $\gamma(z_{nk}) \to r_{nk}$ where r_{nk} is defined in section 2.1. Each data point is thereby assigned to the cluster having the closest mean. The EM re-estimation equation for the μ_k, given by (17), then reduces to the K-means result (4). Note that the re-estimation formula for the mixing coefficients (23) simply resets the value of π_k to be equal to the fraction of data points assigned to cluster k, although these parameters no longer play an active role in the algorithm. Finally, in the limit $\epsilon \to 0$ the expected complete-data log likelihood, becomes

$$\mathbb{E}_Z[\ln p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \pi)] \to -\frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|^2 + const \tag{23}$$

Thus we see that in this limit, maximizing the expected complete-data log likelihood[30] is equivalent to minimizing the distortion measure J for the K-means algorithm given by (1). The proof of this part is provided at [9]. Note that the K-means algorithm does not estimate the covariances of the clusters but only the cluster means. A hard-assignment version of the Gaussian mixture model with general covariance matrices, known as the elliptical K-means algorithm, has been considered by Sung and Poggio[33]. It is also worth noting that the K-means algorithm itself is often used to initialize the parameters in a Gaussian mixture model before applying the EM algorithm.

3. Experiment

In this section, my illustration continues on K-means with experiment where we can see the evaluation of k-means representative variants. I carried 2 experiment with one on directly illustrating K-means showing the history of iterations and another one on clustering a picture. Thus, we shall clearly see how K-means performs.

3.1. Datasets

For the first experiment, I used the distribution locations of zebra (link `https://github.com/VarusJ/cv_midterm/blob/main/data/data.mat`) and ran the k-means code to find out fountains clusters. And for the other experiment, I used a picture of Captain America downloaded from the internet. The original data is visualized, shown in Figure 4. Setting the value of k to 3 and result of the first iteration along with the final result can be found in Figure 4.

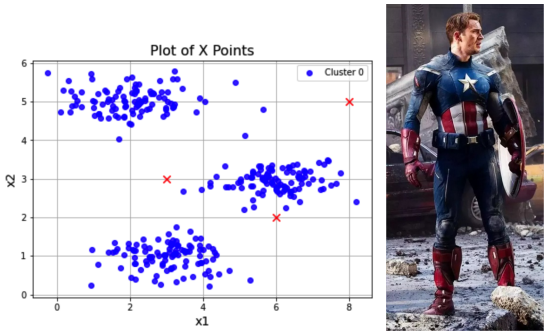

Figure 3. visualization of the original data

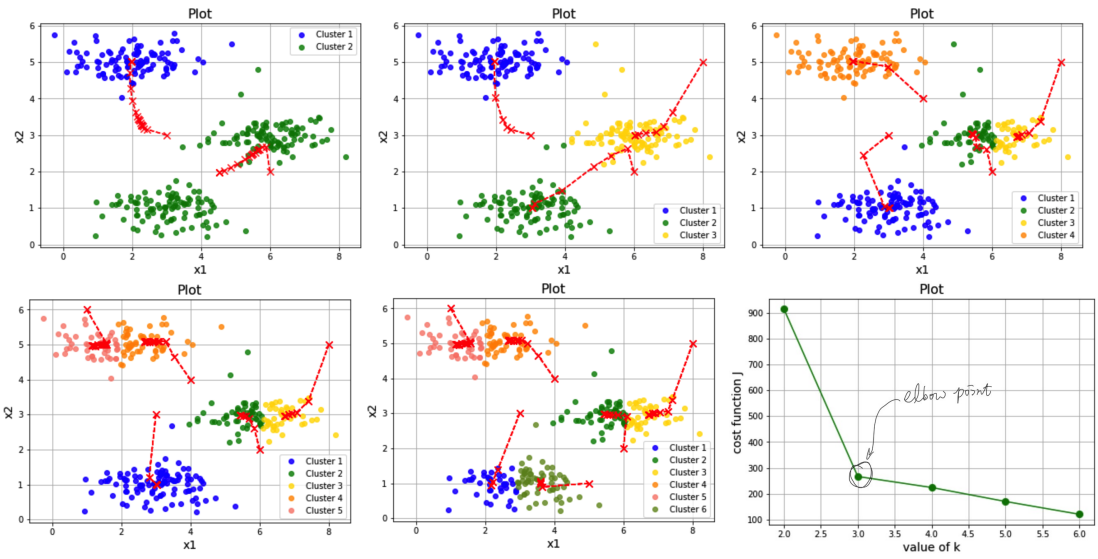

Figure 4. The result of K-means with k increasing from 2 to 6 and a k-J plot showing elbow point

3.2. Result

For the first experiment, the K-means algorithm is illustrated using the data set described above, where I have chosen K = 3, and so the assignment of each data point to the nearest cluster centre can be easily observed. Also, I offer another experiment with an increasing value of k so that you could get a better understanding of section 2.2.2, where you can easily find the elbow point and the reason why the former part of this experiment set k to be 3.

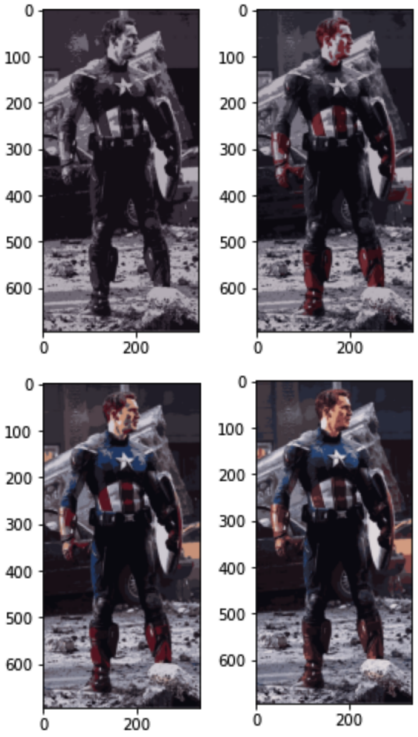

Figure 5. K-means on image segementation where k = 4, 8, 12, 16

For the second experiment, I mean to exhibit an application of K-means algorithm in the area of Computer Vision despite there are many other applications listed in Table 1. As an illustration, we consider image segmentation and image compression. The goal of segmentation is to partition an image into regions each of which has a reasonably homogeneous visual appearance or which corresponds to objects or parts of objects. Each image pixel is a point in a 3-dimensional space and our segmentation algorithm simply treats each pixel in the image as a separate data point. I illustrate the result of running K-means to convergence, for $K = 4, 8, 12, 16$, by re-drawing the image replacing each pixel vector with the R, G, B intensity triplet given by the centre μ_k to which that pixel has been assigned. Results for various values of K are shown in Figure 5.

4. Conclusion

This paper focuses on the k-means algorithm and its variants, along with the models hidden behind, illustrated with experimental analysis. For K-means, I first introduce its application to give you a broad idea of K-means' capability, with its mathematical principles and proof provided later. Then, I continue our discussion by the techniques and a important variant, Fuzzy C Means. Finally I give an introduction of GMM and EM algorithm so that it can be seen that K-means is a special case of EM based on the context of GMM. For future work, combining with graph information, which is also my research area, might lead to more result.

References

[1] Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[2] Barron, Andrew, Rissanen, and Jorma. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 1998.

[3] Shai Ben-David, Ulrike Von Luxburg, and Dávid Pál. A sober look at clustering stability. In *International Conference on Computational Learning Theory*, pages 5–19. Springer, 2006.

[4] Asa Ben-Hur, Andre Elisseeff, and Isabelle Guyon. A stability based method for discovering structure in clustered data. In *Biocomputing 2002*, pages 6–17. World Scientific, 2001.

[5] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013.

[6] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[8] Hamparsum Bozdogan. Model selection and akaike's information criterion (aic): The general theory and its analytical extensions. *Psychometrika*, 52:345–370, 02 1987.

[9] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[10] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.

[11] Charles Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th international conference on Machine Learning (ICML-03)*, pages 147–153, 2003.

[12] Glenn, W., MilliganMartha, C., and Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 1985.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[14] Michael E. Hodgson. Reducing the computational requirements of the minimum-distance classifier. *Remote Sensing of Environment*, 25(1):117 – 128, 1988.

[15] Lawrence J. Hubert and Joel R. Levin. A general statistical framework for assessing categorical clustering in free recall. *Psychological Bulletin*, 83(6):1072–1080, 1976.

[16] Leonard Kaufman and Peter J. Rousseeuw. Finding groups in data: An introduction to cluster analysis. *Journal of the American Statal Association*, 1990.

[17] Trupti M Kodinariya and Prashant R Makwana. Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95, 2013.

[18] Abba M Krieger and Paul E Green. A cautionary note on using internal cross validation to select the number of clusters. *Psychometrika*, 64(3):341–353, 1999.

[19] Tilman Lange, Volker Roth, Mikio L Braun, and Joachim M Buhmann. Stability-based validation of clustering solutions. *Neural computation*, 16(6):1299–1323, 2004.

[20] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[21] J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[22] Geoffrey J McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

[23] Andrew Moore. The anchors hierarchy: Using the triangle inequality to survive high dimensional data. In *In Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 397–405. AAAI Press, 2000.

[24] N. R. Pal and J. C. Bezdek. On cluster validity for the fuzzy c-means model. *IEEE Transactions on Fuzzy Systems*, 3(3):P.370–379, 1995.

[25] Georg Peters, Fernando Crespo, Pawan Lingras, and Richard Weber. Soft clustering–fuzzy and rough approaches and their extensions and derivatives. *International Journal of Approximate Reasoning*, 54(2):307–322, 2013.

[26] Katherine Pollard and Mark Laan. A method to identify significant clusters in gene expression data. *Mark J. van der Laan*, 2, 01 2002.

[27] V. Ramasubramanian and K. K. Paliwal. A generalized optimization of the k-d tree for fast nearest-neighbour search. In *Fourth IEEE Region 10 International Conference TENCON*, pages 565–568, 1989.

[28] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[29] Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 09 1951.

[30] Paul Scheet and Matthew Stephens. A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. *The American Journal of Human Genetics*, 78(4):629–644, 2006.

[31] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[32] Padhraic Smyth. Clustering using monte carlo cross-validation. In *Kdd*, volume 1, pages 26–133, 1996.

[33] K-K Sung and Tomaso Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):39–51, 1998.

[34] Junhui Wang. Consistent selection of the number of clusters via crossvalidation. *Biometrika*, 97(4):893–904, 2010.

[35] Kuo Lung Wu. Analysis of parameter selections for fuzzy c -means. *Pattern Recognition*, 45(1):407–415, 2012.

[36] Jian Yu, Qiansheng Cheng, and Houkuan Huang. Analysis of the weighting exponent in the fcm. *IEEE Transactions on Systems Man and Cybernetics Part B*, 34(1):634–639, 2004.