Singapore University of Technology and Design
Information Systems Technology and Design
50.007 Machine Learning

Clustering: K-Means

In our previous lectures, we considered supervised learning scenarios, where we have access to both examples and the corresponding target labels or responses: $\{(x^{(i)}, y^{(i)}), i = 1, \ldots, n\}$. The goal was to learn a mapping from examples to labels that would work well on (generalize to) yet unseen examples. In contrast, here we have only examples $S_n = \{x^{(i)}, i = 1, \ldots, n\}$. What is the learning task now? The goal of *unsupervised learning* is to uncover useful structure in the data S_n such as identify groups or clusters of similar examples.

Clustering is one of the key problems in exploratory data analysis. Examples of clustering applications include mining customer purchase patterns, modeling language families, or grouping search results according to topics. Clustering can be also used for data compression. For example, consider vector quantization for image compression. A typical image consists of 1024*1024 pixels, where each pixel is represented by three integers ranging from 0 to 255 (8 bits), encoding red, green, and blue intensities of that point in the image. As a result, we need 24 bits to store each pixel, and the full image requires about 3MB of storage. One way to compress the image is to select only a few representative pixels and substitute each pixel with the closest representative. For example, if we use only 32 colors (5 bits), then we will need a codebook of 32 representative pixels (points in the red-green-blue color space). Now, instead of requiring 24bits for each pixel, we only use 5bits to store the identity of the closest representative pixel in our codebook. In addition, we need to store the codebook itself which has 32 points in the color space. Without compressing these further, each of the 32 representative pixels would require 24 bits to store. Taken together, the compressed image would require 640KB.

A bit more formally, the clustering problem can we written as:

Input: training set $S_n = \{x^{(i)}, i = 1, \ldots, n\}$, where $x^{(i)} \in R^d$, integer k
Output: a set of clusters $C_1, \ldots, C_k$.

For example, in the context of the previous example, each cluster is represented by one pixel (a point in color space). The cluster as a set would then consist of all the points in the color space that are closest to that representative. This example highlights the two ways of specify the output of the clustering algorithm. We can either return the groups (clusters) as sets, or we can return the representatives[1] that implicitly specify the clusters as sets. Which view is more appropriate depends on the clustering problem. For example, when clustering news, the output can be comprised of groups of articles

[1] In the clustering literature, the terms "representative", "center" and "exemplar" are used interchangeably.

about the same event. Alternatively, we can describe each cluster by its representative. In the news example, we may want to select a single news story for each event cluster. In fact, this output format is adopted in Google News.

Note that we have yet to specify any criterion for selecting clusters or the representatives. To this end, we must be able to compare pairs of points to determine whether they are indeed similar (should be in the same cluster) or not (should be in a different cluster). The comparison can be either in terms of similarity such as *cosine similarity* or dissimilarity as in Euclidean distance. Cosine similarity is simply the angle between two vectors (elements):

$$cos(x^{(i)}, x^{(j)}) = \frac{x^{(i)} \cdot x^{(j)}}{\|x^{(i)}\|\|x^{(j)}\|} = \frac{\sum_{l=1}^{d} x_l^{(i)} x_l^{(j)}}{\sqrt{\sum_{l=1}^{d} (x_l^{(i)})^2}\sqrt{\sum_{l=1}^{d} (x_l^{(j)})^2}} \tag{1}$$

Alternatively, we can focus on dissimilarity as in pairwise distance. In this lecture, we will primarily use squared Euclidian distance

$$dist(x^{(i)}, x^{(j)}) = \|x^{(i)} - x^{(j)}\|^2 = \sum_{l=1}^{d} (x_l^{(i)} - x_l^{(j)})^2 \tag{2}$$

but there are many alternatives. For example, in the literature, you may often encounter l_1 distance

$$dist(x^{(i)}, x^{(j)}) = \|x^{(i)} - x^{(j)}\|_1 = \sum_{l=1}^{d} |x_l^{(i)} - x_l^{(j)}| \tag{3}$$

The choice of which distance metric to use is important as it will determine the type of clusters you will find. A reasonable metric or similarity is often easy to find based on the application. Another issue with the metric is that the available clustering algorithms such as k-means discussed below may rely on a particular metric.

Once we have the distance metric, we can specify an objective function for clustering. In other words, we specify the cost of choosing any particular set of clusters or their representatives (a.k.a. centroids). The "optimal" clustering is then obtained by minimizing this cost. The cost is often cast in terms of *distortion* associated with individual clusters. For instance, the cost – distortion – associated with cluster C could be the sum of pairwise distances within the points in C, or the diameter of the cluster (largest pairwise distance). We will define the distortion here slightly differently: the sum of (squared) distances from each point in the cluster to the corresponding cluster representative z. For cluster C with centroid z, the distortion is defined as $\sum_{i \in C} \|x^{(i)} - z\|^2$. The cost of clustering $C_1, C_2, \ldots, C_k$, is simply the sum of costs of individual clusters:

$$cost(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)}) = \sum_{j=1\ldots k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2 \tag{4}$$

Our goal is to find a clustering that minimizes this cost. Note that the cost here depends on both the clusters and how the representatives (centroids) are chosen for each cluster. It seems unnecessary to have to specify both clusters and centroids and, indeed, one will imply the other. We will see this below. Note also that we only consider valid clusterings $C_1, \ldots, C_k$, those that specify a partition of the indexes $\{1, \ldots, n\}$. In other words, each point must belong to one and only one cluster.

1 K-means

We introduced two ways to characterize the output of a clustering algorithm: the cluster itself or the corresponding representative (centroid). For some cost functions these two representations are interchangeable: knowing the representatives, we can compute the corresponding clusters and vice versa. In fact, this statement holds for the cost function introduced above. We can define clusters by their representatives:

$$C_j = \{i \in \{1, \ldots, n\} \; s.t. \; the \; closest \; representative \; of \; x^{(i)} \; is \; z^{(j)}\} \tag{5}$$

These clusters define an optimal clustering with respect to our cost function for a fixed setting of the representatives $z^{(1)}, \ldots, z^{(k)}$. In other words,

$$\begin{aligned} cost(z^{(1)}, \ldots, z^{(k)}) &= \min_{C_1,\ldots,C_k} cost(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)}) && (6) \\ &= \min_{C_1,\ldots,C_k} \sum_{j=1\ldots k} \sum_{i \in C_j} \|x^{(i)} - z^{(j)}\|^2 && (7) \\ &= \sum_{i=1,\ldots,n} \min_{j=1,\ldots,k} \|x^{(i)} - z^{(j)}\|^2 && (8) \end{aligned}$$

where in the last expression we are simply assigning each point to its closest representative (as we should). Geometrically, the partition induced by the centroids can be visualized as as Voronoi partition of R^d, where R^d is divided into k convex cells. The cell is the region of space where the corresponding centroid z is the closest representative. See Figure 1.

The K-means algorithm

Now, given an optimization criterion, we need to find an algorithm that tries to minimize it. Directly enumerating and selecting the best clustering out of all the possible clusterings is prohibitively expensive. We will instead rely here on an approximate method known as the k-means algorithm. This algorithm alternatingly finds best clusters for centroids, and best centroids for clusters. The iterative algorithm is given by

1. Initialize centroids $z^{(1)}, \ldots, z^{(k)}$

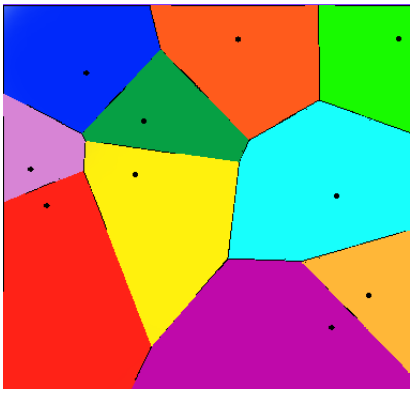

Figure 1: An example of Voronoi diagram

2. Repeat until there is no further change in cost
   (a) for each j=1,...,k: $C_j = \{i \text{ s.t. } x^{(i)} \text{ is closest to } z^{(j)}\}$
   (b) for each j=1,...,k: $z^{(j)} = \frac{1}{|C_j|} \sum_{i \in C_j} x^{(i)}$ (cluster mean)

Each iteration requires $\mathcal{O}(kn)$ operations.

Convergence The k-means algorithm does converge albeit not necessarily to a solution that is optimal with respect to the cost defined above. However, each iteration of the algorithm necessarily lowers the cost. Given that the algorithm alternates between choosing clusters and centroids, it will be helpful to look at

$$cost(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)}) \tag{9}$$

as the objective function for the algorithm. Consider $(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)})$ as the starting point. In the first step of the algorithm, we find new clusters $C'_1, C'_2, \ldots, C'_k$ corresponding to fixed centroids $z^{(1)}, \ldots, z^{(k)}$. These new clusters are chosen such that

$$\begin{aligned} cost(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)}) &\overset{(a)}{\geq} \min_{C_1,\ldots,C_k} cost(C_1, C_2, \ldots, C_k, z^{(1)}, \ldots, z^{(k)}) \quad (10) \\ &= cost(C'_1, C'_2, \ldots, C'_k, z^{(1)}, \ldots, z^{(k)}) \quad (11) \end{aligned}$$

This is because $C'_1, C'_2, \ldots, C'_k$ are clusters induced from assigning each point to its closest centroid. No other clusters can achieve lower cost for these centroids. The inequality (a) is equality only when the algorithm converges. In the 2nd step of the algorithm, we fix the new clusters $C'_1, C'_2, \ldots, C'_k$ and find new centroids $z'^{(1)}, \ldots, z'^{(k)}$ such that

$$\begin{aligned} cost(C'_1, C'_2, \ldots, C'_k, z^{(1)}, \ldots, z^{(k)}) &\overset{(b)}{\geq} \min_{z^{(1)},\ldots,z^{(2)}} cost(C'_1, C'_2, \ldots, C'_k, z^{(1)}, \ldots, z^{(k)}) \quad (12) \\ &= cost(C'_1, C'_2, \ldots, C'_k, z'^{(1)}, \ldots, z'^{(k)}) \quad (13) \end{aligned}$$

where the inequality in (b) is tight only when the centroids are already optimal for the given clusters, i.e., when the algorithm has converged. The problem of finding the new centroids decomposes across clusters. In other words, we can find them separately for each cluster. In particular, the new centroid $z'^{(j)}$ for a fixed cluster C'_j is found by minimizing

$$\sum_{i \in C'_j} \|x^{(i)} - z^{(j)}\|^2 \tag{14}$$

with respect to $z^{(j)}$. The solution to this is the mean of the points in the cluster

$$z'^{(j)} = \frac{1}{|C'_j|} \sum_{i \in C'_j} x^{(i)} \tag{15}$$

as specified in the k-means algorithm.

Now, taken together, (a) and (b) guarantee that the k-means algorithm monotonically decreases the objective function. As the cost has a lower bound (non-negative), the algorithm must converge. Moreover, after the first step of each iteration, the resulting cost is exactly Eq.(8), and it too will decrease monotonically.