CS 181 Spring 2019 Section 7

Clustering

1 Motivation

We now move onto **unsupervised learning**, where the objective is to learn the structure of unlabeled data. In other words, we are looking for groups, or **clusters** among the data. Clustering algorithms are useful not only for finding groups in data, but also to extract features of the data that summarize the most important information about the data in a compressed way.

For most clustering algorithms, we need some kind of a metric that we can use to specify the notion of "distance" between the data points. If, for example, the points $\mathbf{x}$ and $\mathbf{x'}$ live in some Euclidean space $\mathbb{R}^m$, then the natural choice of such metric is the l_2 distance:

$$||\mathbf{x} - \mathbf{x'}|| = \sqrt{\sum_{i=1}^{n}(x_i - x_i')^2}$$

Now that the metric is well-defined, the next thing we need to do is to decide how many groups we want. Sometimes you know the ideal number of groups in advance (*e.g.* clustering alphabet). Other times, you need to decide if you'd like a more compressed representation with more information loss by having the number of groups small, or a less compressed representation with less information loss by having the number of groups large.

Suppose our data set is $\{\mathbf{x}_i\}_{i=1}^n$, then our objective is to find the ideal assignment of the data set to the clusters, by assigning to each of the n data points, a binary **responsibility vector** $\mathbf{r}_i$, which is all zeros except one component, which corresponds to the assigned cluster.

2 K-Means Algorithm

The idea is to represent each cluster by the point in data space that is the average of the data assigned to it. For some choice of K, the K-Means Algorithm (also called Lloyd's algorithm) keeps doing the following: iterate through the data points and update the responsibility vectors according to their closest means, and iterate over the mean vectors $\{\boldsymbol{\mu}_k\}_{k=1}^K$ and change them to be the mean of the examples currently assigned the cluster. We repeat this until convergence: until none of the responsibility vectors change.

2.1 Derivation

We begin by defining a loss function:

$$\mathcal{L}(\{\mathbf{r}_i\}_{i=1}^n, \{\boldsymbol{\mu}_k\}_{k=1}^K) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

and the K-Means Algorithm minimizes this via coordinate descent. We first choose r_i such that:

$$r_{ik} = \begin{cases} 1 & \text{if } k = argmin_{k'}||\mathbf{x}_i - \boldsymbol{\mu}_{k'}|| \\ 0 & \text{otherwise} \end{cases}$$

Having fixed everything else, we see that it is equivalent, for a given k, to minimize the square loss:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\mu}_k) &= \sum_{i=1}^{n} r_{ik}||\mathbf{x}_i - \boldsymbol{\mu}_k||^2 \\ &= \sum_{i=1}^{n} r_{ik}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T(\mathbf{x}_i - \boldsymbol{\mu}_k) \end{aligned}$$

Taking the derivative and setting it to zero,

$$\begin{aligned} \frac{\partial \mathcal{L}(\boldsymbol{\mu}_k)}{\partial \boldsymbol{\mu}_k} &= -2\sum_{i=1}^{n} r_{ik}(\mathbf{x}_i - \boldsymbol{\mu}_k) = 0 \\ \boldsymbol{\mu}_k &= \frac{\sum_{i=1}^{n} r_{ik}\mathbf{x}_i}{\sum_{i=1}^{n} r_{ik}} \end{aligned}$$

2.2 Number of Clusters

There is not an especially well justified method to choose the number of clusters when using K-means. One approach is to plot K vs the objective criterion, and look for a "knee" or "kink" where progress slows down.

An advanced method is to use the "gap statistic". But this is out of scope for the course.

2.3 Notes

Lloyd's algorithm finds a local optimal solution. Finding the global optimal is NP-hard. A common strategy is to use random restarts. More recently, an algorithm called **K-Means++** has enjoyed popular usage as an alternative to random initialization. This is

out of scope for the course, but the basic idea is to randomly select some of the data to be the first cluster centers. This is done by iteratively adding cluster centers, sampling them in proportion to the squared distance of each example from its nearest cluster center. Thus K-Means++ tends to favor points that are distant from the existing centers and produce a more diverse set of centers.

It is generally a good idea to **standardize** the data to account for unsatisfying result due to dimension mismatch. Lastly, when for the metric we are using for the given data set, a "mean" does not make sense, we might instead use a **K-Medoids Algorithm**. This algorithm requires the cluster centers to be a data point in the data set.

3 Hierarchical Agglomerative Clustering

Hierarchical clustering constructs a tree over the data, where the leaves are individual data items, while the root is a single cluster that contains all of the data. When drawing the dendrogram, for the clustering to be valid, the distances between the two groups being merged should be monotonically increasing. The main decision in using HAC is what the distance criterion should be between groups.

3.1 The Min-Linkage Criterion

For two groups indexed by i and i', the idea is to merge groups based on the shortest distance over all possible pairs:

$$DIST_{\min}(\{\mathbf{x}_i\}_{i=1}^{n}, \{\mathbf{x}_{i'}\}_{i'=1}^{n'}) = \min_{i,i'} ||\mathbf{x}_i - \mathbf{x}_{i'}||.$$

3.2 The Max-Linkage Criterion

For two groups indexed by i and i', the idea is to merge groups based on the largest distance over all possible pairs:

$$DIST_{\max}(\{\mathbf{x}_i\}_{i=1}^{n}, \{\mathbf{x}_{i'}\}_{i'=1}^{n'}) = \max_{i,i'} ||\mathbf{x}_i - \mathbf{x}_{i'}||$$

3.3 The Average-Linkage Criterion

For two groups indexed by i and i', the idea is to average over all possible pairs between the groups:

$$DIST_{\text{avg}}(\{\mathbf{x}_i\}_{i=1}^{n}, \{\mathbf{x}_{i'}\}_{i'=1}^{n'}) = \frac{1}{nn'} \sum_{i=1}^{n} \sum_{i'=1}^{n'} ||\mathbf{x}_i - \mathbf{x}_{i'}||$$

3.4 The Centroid-Linkage Criterion

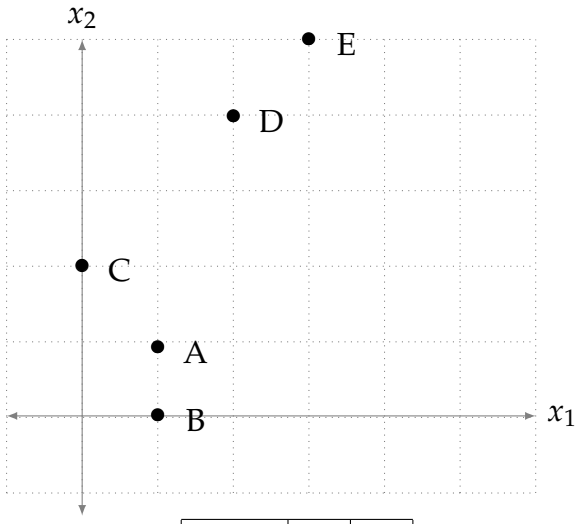
For two groups indexed by i and i', the idea is to look at the difference between the groups' centroids:

$$DIST_{\text{cent}}(\{\mathbf{x}_i\}_{i=1}^{n}, \{\mathbf{x}_{i'}\}_{i'=1}^{n'}) = || \left(\frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \right) - \left(\frac{1}{n'} \sum_{i'=1}^{n'} \mathbf{x}_{i'} \right) ||$$

4 Practice Problems

1. **K-Means (Di Cook)**

Use K-means to cluster these examples in $\mathbb{R}^2$, looking for $K = 2$ clusters. Suppose that points A and C are randomly selected as the initial means.



Point	x_1	x_2
A	1	1
B	1	0
C	0	2
D	2	4
E	3	5

2. **Convergence of K-Means (Bishop 9.1)**

Consider Lloyd's algorithm for finding a K-Means clustering data, i.e., minimizing

$$\mathcal{L}(\{\mathbf{r}_i\}_{i=1}^n, \{\boldsymbol{\mu}_k\}_{k=1}^K) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} ||\mathbf{x}_i - \boldsymbol{\mu}_k||.$$

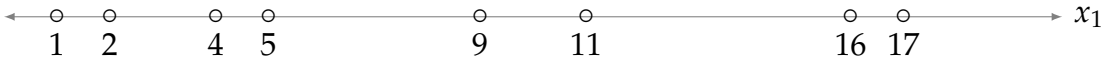
Show that as a consequence of there being a finite number of possible assignments for the set of responsibilities r_{ik}, and that for each such assignment there is a unique optimum for the means $\{\boldsymbol{\mu}_k\}_{k=1}^K$, the K-Means algorithm must converge after a finite number of iterations.

3. **K-means and HAC**

What are three important differences between K-means and HAC?

4. **Min-Linkage and Max-Linkage Criterion**

Assume the following examples lie in $\mathbb{R}$. Each example is initially in its own cluster.



$\{1\}\{2\}\{4\}\{5\}\{9\}\{11\}\{16\}\{17\}$

(a) Using the Min-Linkage Criterion for the HAC Algorithm, what is the clustering sequence? Draw the dendrogram.

(b) Using the Max-Linkage Criterion for the HAC Algorithm, what is the clustering sequence? Draw the dendrogram.

5. **Clustering Complexity**

What is the "big-O" complexity of HAC? What is the "big-O" complexity of K-means? Compare these.

6. **Scaling to large dimensions.**

Explain the 'curse of dimensionality' and how it is related to HAC.