Randomized Algorithms.
Lecture Notes on Dimensionality Reduction for k-Means.
Kasper Green Larsen.

1 K-Means

The k-means clustering problem is a classic unsupervised learning problem. Here one is given a set of n points $x_1, \ldots, x_n$ in $\mathbb{R}^d$ and a parameter k. The goal is to select k cluster centers $c_1, \ldots, c_k$ such that one minimizes the sum of squared distances to the nearest cluster center, i.e. find $c_1, \ldots, c_k$ minimizing:

$$\sum_{i=1}^{n} \min_j \|x_i - c_j\|_2^2.$$

Let $c_1^*, \ldots, c_k^*$ be the optimal cluster centers and define sets $X_1, \ldots, X_k$ such that X_i contains all input points x_j such that c_i^* is the closest cluster center. It is well known that c_i^* must equal $\frac{1}{|X_i|}\sum_{x_j \in X_i} x_j$, i.e. the optimal choice of cluster center is the mean of the data points in the cluster. So for any partitioning of the points $x_1, \ldots, x_n$ into clusters $X_1, \ldots, X_k$, the optimal cost for that partitioning is:

$$\sum_{i=1}^{k} \sum_{x_j \in X_i} \left\| x_j - \frac{1}{|X_i|} \sum_{x_h \in X_i} x_h \right\|_2^2.$$

Computing the optimal k-means clustering is NP-hard, so one often runs a heuristic such as Lloyd's algorithm instead. Lloyd's algorithm starts by partitioning the input points into k clusters $X_1, \ldots, X_k$ in some way (different implementations have different initializations). One then repeats the following: Compute the optimal cluster centers $c_1, \ldots, c_k$ (as the mean of the points in the cluster). Then re-assign each point to the closest cluster center and repeat until no clusters change.

Lloyd's algorithm may get stuck in a local minimum and hence does not necessarily find the optimal clustering. The number of iterations before termination may vary on different initializations.

Letting t denote the number of iterations of Lloyd's algorithm, the total running time is $O(tndk)$ because in each iteration, we need to compute the new centers in $O(nd)$ time and reassign points in $O(ndk)$ time.

Johnson-Lindenstrauss. Let us see how Johnson-Lindenstrauss transforms may be used to speed up Lloyd's algorithm. Before running Lloyd's algorithm, perform a Johnson-Lindenstrauss transform such that all distances between points are preserved to within $(1 \pm \varepsilon)$. Let $A \in \mathbb{R}^{m \times d}$ be the embedding matrix such that each original point x_i is mapped to Ax_i. The new dimensionality is $m = O(\varepsilon^{-2} \lg n)$. Performing the transform (naively) takes $O(nd\varepsilon^{-2} \lg n)$ time, which may be much less than $O(tndk)$. Thereafter, running Lloyd's in the lower dimensional space takes time $O(tnk\varepsilon^{-2} \lg n)$ for a total running time of $O(n\varepsilon^{-2} \lg n(d + kt))$. Depending on d, t and k, this may or may not be faster than without dimensionality reduction. If one instead uses the Fast Johnson Lindenstrauss transform (next lecture), one can often embed in time $O(d \lg d)$, reducing the total time to $O(nd \lg d + tnk\varepsilon^{-2} \lg n)$.

Let us show that the clustering found on the dimensionality reduced data set is almost as good as if one runs in the original space. For this, we first need a technical lemma showing how the cost function may be rewritten:

Lemma 1. *For any set of points $x_1, \ldots, x_n$ and a partitioning of the points into clusters $X_1, \ldots, X_k$, the cost of the clustering satisfies*

$$\sum_{j=1}^{k} \sum_{x_i \in X_j} \left\| x_i - \frac{1}{|X_j|} \sum_{x_h \in X_j} x_h \right\|_2^2 = \frac{1}{2} \sum_{j=1}^{k} \frac{1}{|X_j|} \sum_{x_i \in X_j} \sum_{x_h \in X_j} \|x_i - x_h\|_2^2 .$$

As the proof consist of mostly uninteresting linear algebraic manipulations, we defer the proof to the end of the section.

The interesting thing about Lemma 1 is that it shows that the cost of a clustering can be written solely in terms of pairwise distances. Intuitively this means that the cost of all clusterings will be roughly preserved if all distances are roughly preserved. To see this formally, consider any clustering $X_1, \ldots, X_k$ on the dimensionality reduced data set $Ax_1, \ldots, Ax_n$ and assume its cost is μ. By Lemma 1, we get that the cost of the clustering equals:

$$\mu = \sum_{j=1}^{k} \sum_{x_i \in X_j} \left\| Ax_i - \frac{1}{|X_j|} \sum_{x_h \in X_j} Ax_h \right\|_2^2 = \frac{1}{2} \sum_{j=1}^{k} \frac{1}{|X_j|} \sum_{x_i \in X_j} \sum_{x_h \in X_j} \|Ax_i - Ax_h\|_2^2 .$$

Johnson-Lindenstrauss guarantees that each term $\|Ax_i - Ax_h\|_2^2$ is within $(1 \pm \varepsilon)\|x_i - x_h\|_2^2$, thus the cost of the same clustering *on the original data points* lies in the interval:

$$\frac{1}{2} \sum_{j=1}^{k} \frac{1}{|X_j|} \sum_{x_i \in X_j} \sum_{x_h \in X_j} (1 \pm \varepsilon) \|x_i - x_h\|_2^2 = (1 \pm \varepsilon) \frac{1}{2} \sum_{j=1}^{k} \frac{1}{|X_j|} \sum_{x_i \in X_j} \sum_{x_h \in X_j} \|x_i - x_h\|_2^2 = (1 \pm \varepsilon)\mu.$$

Since this holds for *all* clusterings, we can in particular conclude that if one finds a good clustering $X_1, \ldots, X_k$ on the low dimensional data set, say one with a cost of at most $(1+\gamma)\mu^*$ where μ^* is the optimal clustering in low dimensional space, then that clustering has a cost of at most $(1+\varepsilon)(1+\gamma)\mu^*$ on the original data set. Moreover, if we let ψ^* be the cost of the optimal clustering $X_1^*, \ldots, X_k^*$ on the original data set, then we know that the optimal clustering must have a cost of $\frac{1}{(1\pm\varepsilon)}\psi^* \geq \mu^*$ in low dimensional space. It follows that the clustering $X_1, \ldots, X_k$ has a cost of no more than $(1+\varepsilon)(1+\gamma)\frac{1}{1\pm\varepsilon}\psi^*$. For $\varepsilon \leq 1/2$, we have $1/(1-\varepsilon) = 1 + \varepsilon/(1-\varepsilon) \leq 1 + 2\varepsilon$ and $(1+\varepsilon)(1+2\varepsilon) = (1 + 3\varepsilon + 2\varepsilon^2) \leq 1 + 4\varepsilon$ and thus the cost is no more than $(1+4\varepsilon)(1+\gamma)\psi^*$. That is, if can find an approximately optimal clustering for the dimensionality reduced points, we have found an approximately optimal clustering for the original points.

Proof of Lemma 1.

Proof.

$$\sum_{j=1}^{k}\sum_{x_i\in X_j}\left\|x_i-\frac{1}{|X_j|}\sum_{x_h\in X_j}x_h\right\|_2^2 =$$

$$\sum_{j=1}^{k}\sum_{x_i\in X_j}\left(\|x_i\|_2^2-\frac{2}{|X_j|}\sum_{x_h\in X_j}\langle x_i,x_h\rangle+\frac{1}{|X_j|^2}\left\|\sum_{x_h\in X_j}x_h\right\|_2^2\right) =$$

$$\sum_{j=1}^{k}\left(\sum_{x_i\in X_j}\|x_i\|_2^2-\frac{2}{|X_j|}\langle\sum_{x_i\in X_j}x_i,\sum_{x_h\in X_j}x_h\rangle+\frac{1}{|X_j|}\|\sum_{x_h\in X_j}x_h\|_2^2\right) =$$

$$\sum_{j=1}^{k}\left(\sum_{x_i\in X_j}\|x_i\|_2^2-\frac{2}{|X_j|}\left\|\sum_{x_i\in X_j}x_i\right\|_2^2+\frac{1}{|X_j|}\left\|\sum_{x_h\in X_j}x_h\right\|_2^2\right) =$$

$$\sum_{j=1}^{k}\left(\sum_{x_i\in X_j}\|x_i\|_2^2-\frac{1}{|X_j|}\left\|\sum_{x_i\in X_j}x_i\right\|_2^2\right) =$$

$$\frac{1}{2}\sum_{j=1}^{k}\left(2\sum_{x_i\in X_j}\|x_i\|_2^2-\frac{2}{|X_j|}\left\|\sum_{x_i\in X_j}x_i\right\|_2^2\right) =$$

$$\frac{1}{2}\sum_{j=1}^{k}\left(\frac{2}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\|x_i\|_2^2-\frac{2}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\langle x_i,x_h\rangle\right) =$$

$$\frac{1}{2}\sum_{j=1}^{k}\left(\frac{1}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\left(\|x_i\|_2^2+\|x_h\|_2^2\right)-\frac{2}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\langle x_i,x_h\rangle\right) =$$

$$\frac{1}{2}\sum_{j=1}^{k}\frac{1}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\left(\|x_i\|_2^2+\|x_h\|_2^2-2\langle x_i,x_h\rangle\right) =$$

$$\frac{1}{2}\sum_{j=1}^{k}\frac{1}{|X_j|}\sum_{x_i\in X_j}\sum_{x_h\in X_j}\|x_i-x_h\|_2^2.$$

□