Clustering and Density Estimation

Muchang Bahng

Spring 2025

Contents

Clustering and dimensionality reduction is used for many purposes, such as preprocessing data, visualizing it, or encoding it in a sparser, more efficient way. Sometimes they are used synonymously because in the backened, many dimensionality reduction algorithms use some form of clustering, so I also group them together.

In the clustering problem, we are given a training set of *unlabeled* data

$$\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\} \tag{1}$$

and want to group the data into a few cohesive clusters.

1 K Means Clustering

We will start off with conceptually the simplest form of clustering, although not the earliest developed. K-means was published at 1967 in [Mac67], but has been discovered in the 1950s. The general idea is that given an integer K, we want to find K *centroids* that provide a good approximation of where the clusters are in the data. We can quantify what a good approximation is by taking the distance from a sample to its closest cluster centroid.

Definition 1.1 (K Means Clustering)

A **K-means clustering** model is an parameteric unsupervised model with parameters $\{\mu_1, \ldots, \mu_K\}$ representing clusters that each sample belongs to. The cluster that sample x belongs to is

$$\text{cluster}(x) = \operatorname*{argmin}_{\mu_k} d(x, \mu_k) \tag{2}$$

Usually, we let d be the L^2 metric in Euclidean space.

Theorem 1.1 (Risk)

The expected risk of K-means is

$$R(\mu_1, \ldots, \mu_K) = \mathbb{E}_x \left[\min_{k \in [K]} \|x - \mu_k\|^2 \right] = \int \min_{k \in [K]} \|x - \mu_k\|^2 \, dx \tag{3}$$

and our empirical risk for a dataset $\mathcal{D} = \{x^{(i)}\}_{i=1}^n$ is therefore

$$\hat{R}(\mu_1, \ldots, \mu_K) = \frac{1}{n} \sum_{i=1}^n \min_{k \in [K]} \|x^{(i)} - \mu_k\|^2 \tag{4}$$

1.1 NP-Hardness

So we have reduced this model into an optimization problem of the appropriate risk. Let's try and analyze how hard this is.

Theorem 1.2 ()

For a fixed dimension d and number of clusters K, we can minimize the empirical risk in $O(n^{dk+1})$.

Theorem 1.3 ()

For any fixed d (even $d = 2$), minimizing the empirical risk over all K is NP-hard.

Therefore, we must rely on approximate algorithms.

1.2 Lloyd's Algorithm

Great, we have an almost-everywhere differentiable function, which can be solved with gradient methods like SGD or coordinate descent. The problem is that this is not necessarily convex.

Theorem 1.4 (Convergence of Coordinate Descent)

Now that convergence is guaranteed, constructing the algorithm is straightforward. In fact, it is precisely coordinate descent!

Algorithm 1.1 (Lloyd's Algorithm)

The algorithm intuitively initializes the centroids randomly, and then moves them to the center of each cluster through the two stage process:

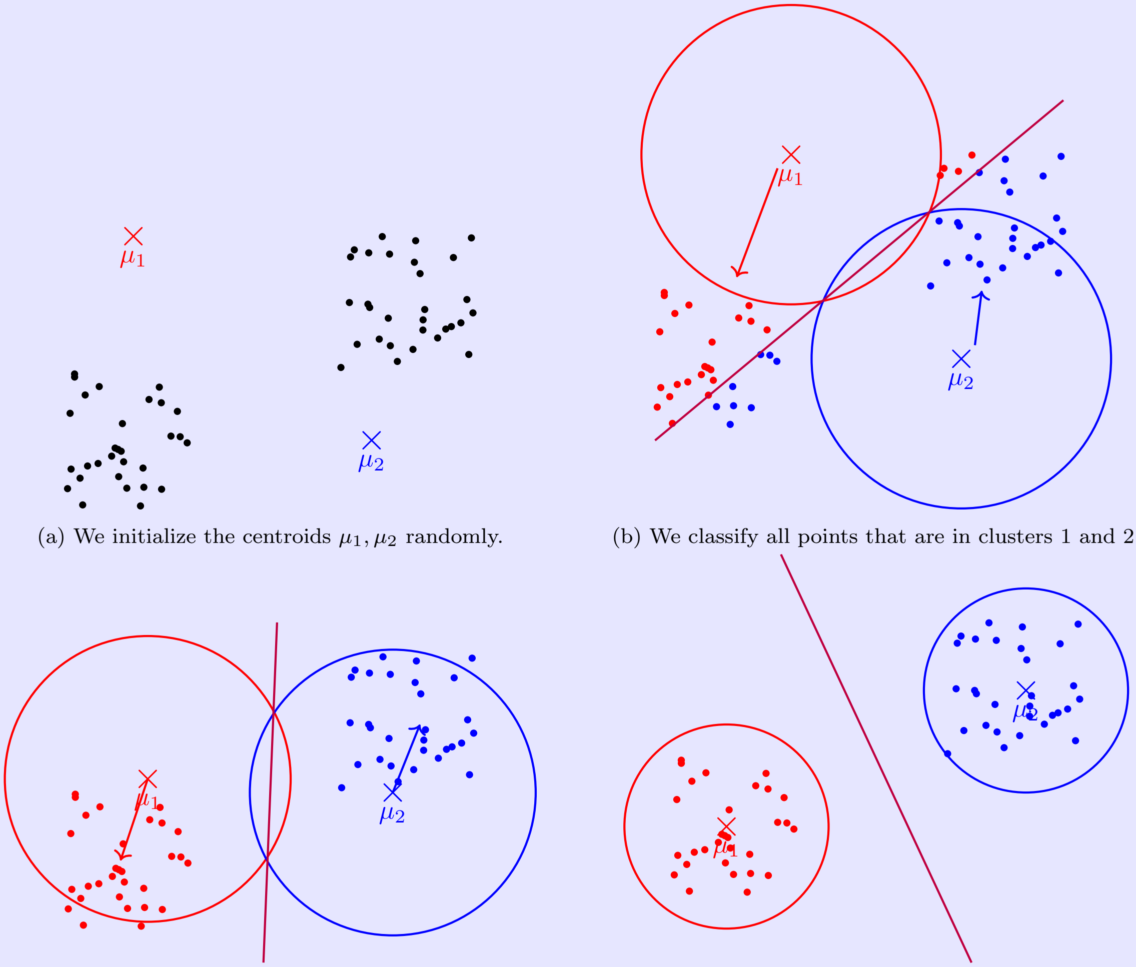
1. Assigning each training sample $x^{(i)}$ to the closest cluster centroid μ_k.
2. Move each training cluster centroid μ_k to the mean of the points assigned to it.

Algorithm 1 K-Means Clustering

```
procedure KMeans($\mathbf{X}, K$)
Require: Dataset $\mathbf{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$ where $x^{(i)} \in \mathbb{R}^d$, number of clusters $K$
Ensure: Cluster centroids $\mu_1, \mu_2, \ldots, \mu_K$ and cluster assignments $c^{(1)}, c^{(2)}, \ldots, c^{(n)}$
    Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^d$ randomly
    repeat
        for $i \leftarrow 1$ to $n$ do
            $c^{(i)} \leftarrow \arg\min_j ||x^{(i)} - \mu_j||^2$
        end for
        for $j \leftarrow 1$ to $K$ do
            $\mu_j \leftarrow \frac{\sum_{i=1}^n \mathbf{1}\{c^{(i)}=k\} x^{(i)}}{\sum_{i=1}^n \mathbf{1}\{c^{(i)}=k\}}$
        end for
    until convergence
    return $\mu_1, \mu_2, \ldots, \mu_K, c^{(1)}, c^{(2)}, \ldots, c^{(n)}$
end procedure
```

Example 1.1 (K-Means Walkthrough)

Let us walk through how the centroids evolve visually on a toy dataset.



(a) We initialize the centroids μ_1, μ_2 randomly.

(b) We classify all points that are in clusters 1 and 2.

(c) We update the centroids and classify again.

(d) We have convergence.

Figure 1: A walkthrough for K-means clustering for $K = 2$ in $\mathbb{R}^2$.

1.3 Concentration Bounds

We can bound the supremum of the expected and exmpirical risk either through the VC dimension or directly with the Rademacher complexity. They both give different bounds, which have advantages and disadvantages.

Theorem 1.5 ()

Let us work over a compact domain. Given that C^* is the true risk minimizer and $\hat{C}$ is our empirical risk minimizer,

$$C^* = \operatorname*{argmin}_{\mu_1, \ldots, \mu_K} R(C), \qquad \hat{C} = \operatorname*{argmin}_{\mu_1, \ldots, \mu_K} \hat{R}(C) \tag{5}$$

we have

$$\mathbb{E}\left[|R(C^*) - R(\hat{C}) \right] \leq \sqrt{\frac{K(d+1)\log n}{n}} \tag{6}$$

Proof.

This is proved using VC dimension.

Let's parse this. So the more clusters you're trying to find—the bigger the K—the worse the bound is. More disturbing is the dimension d. If d is big, then this is not a very good bound.

Theorem 1.6 (2008 Gao, Deroit?, Lugacy?)

Let us work over a compact domain. Given that C^* is the true risk minimizer and $\hat{C}$ is our empirical risk minimizer,

$$C^* = \underset{\mu_1, \ldots, \mu_K}{\text{argmin}}\, R(C), \qquad \hat{C} = \underset{\mu_1, \ldots, \mu_K}{\text{argmin}}\, \hat{R}(C) \tag{7}$$

we have

$$\mathbb{E}\left[|R(C^*) - R(\hat{C}) \right] \leq \frac{K}{n} \tag{8}$$

Proof.

This is proved directly using Rademacher complexity.

The advantage of this is that this bound is dimensionless. This is even true in infinite dimensional Hilbert spaces, which is useful when clustering functions.

Now just because the risks are close it does not mean that the clusters are close. You can have two sets of clusters $\{\mu_k\}, \{\mu'_k\}$ that have similar risk but they can be far apart from each other. To control this we need extra assumptions.

1.4 Choosing Number of Clusters K

Note that the performance of K means really depends on a good choice of K. Think about what would have happened if we set $K = 5$ in the walkthrough above. Let's study the impact of changing K a bit more.

Theorem 1.7 (Expected Risk Decreases as K Increases)

If $K < K'$, then

$$R(\mu_1, \ldots, \mu_K) \leq R(\mu_1, \ldots, \mu_{K'}) \tag{9}$$

Proof.

2 K Means ++

Regular K means doesn't have good convergence.

3 Multi-Dimensional Scaling

Again, we want to reduce our dimension, but the goal is slightly different from PCA.

Definition 3.1 (Multi-Dimensional Scaling)

Given our data $X \in \mathbb{R}^d$, we want to construct a linear map $T : \mathbb{R}^d \to \mathbb{R}^k$ such that it preserves the pairwise differences between the data points. That is, we want to minimize the following loss function

$$\min_T \sum_{i \neq j} \big(d_{\mathbb{R}^k}(T(x_i), T(x_j)) - d_{\mathbb{R}^d}(x_i, x_j) \big) \tag{10}$$

where d_V is a distance metric in the space V.

Note that we can easily modify this formulation to preserve other structures, such as dot products, weights between distances, or different types of metrics in each space. It turns out that when the distance metric is the Euclidean L2 distance, then the solution to this linear map turns out to be PCA. This may be a more intuitive way to think about PCA, since we're trying to preserve the pairwise distances between the data points.

Theorem 3.1 (Equivalence of Classical MDS and PCA)

If the distance metric is the Euclidean L2 distance, then the solution to the MDS problem is equivalent to PCA. That is,

$$T_k = \operatorname*{argmin}_T \sum_{i \neq j} \big(||T(x_i) - T(x_j)||^2 - ||x_i - x_j||^2 \big) \tag{11}$$

Generally, if you don't use classical MDS, then you will get a different answer than PCA and there doesn't exist a closed form solution, so you'll have to minimize this numerically.

Example 3.1 (Non Classical MDS)

The loss

$$\sum_{i \neq j} \big(||T(x_i) - T(x_j)|| - ||x_i - x_j|| \big)^2 \tag{12}$$

does not give the same solution as PCA.

4 Isomap

Isomap is a bit different in the way that it tries to capture more of the global structure of the data, which brings advantages and disadvantages. It is simply a modification of MDS but with geodesic distances.

You start off with the point cloud, but with every point, x_i, you find the local neighborhood N_i and you make a weighted graph over the whole dataset in the high dimensional space. Then, the distance between any two arbitrary points is the shortest weighted sum of the path between them, calculated by Dijkstra's algorithm. Intuitively, this is an approximation of the geodesic distance, denoted d_G, between these two points on a manifold.

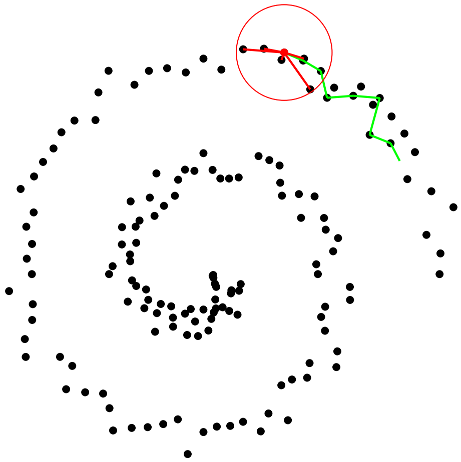


Figure 2: The classical example is the spiral manifold. The data lies in this manifold, and the geodesic distance helps us gain an accurate distance metric within this data.

Definition 4.1 (Isomap)

Then, we simply do Isomap by minimizing

$$\min_T \sum_{i \neq j} \big(d_{\mathbb{R}^k}(T(x_i), T(x_j)) - d_G(x_i, x_j) \big) \tag{13}$$

The problem with this is that it is very sensitive to noise. For example, if we had a few points lying between the spirals, then the geodesic distance between the two spirals would be very small, and so the MDS would try to bring them closer together.

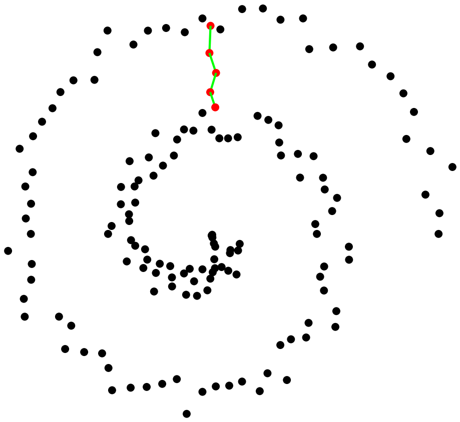


Figure 3: With extra noisy points (red), the geodesic distance can get corrupted.

To fix this, we use *diffusion maps*, which looks at all possible paths between two points and looks at some average of them, which increases robustness.

5 Local Linear Embedding

PCA and MDS are linear embedding methods. Let's move onto nonlinear ones. The first nonlinear models that we work with again use the idea of locality (remember kernel regression). You have data that is globally nonlinear, but if you look at a point and its local neighborhood around it, then it is approximately linear since we assume that it lives in some smooth manifold.

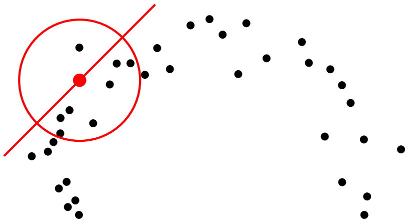


Figure 4: Local linear embedding assumes that the data is locally linear.

The concept of neighborhood can be defined in two ways. You can either just fix an ϵ and take the ϵ-ball around each point x_i. Or you can fix a k and take the k nearest neighbors of each point. The general idea of using kernel PCA is to take a local neighborhood of the data and construct some linear approximation of it.

6 t-SNE

7 UMAP

[MHM20].

References

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. *Berkeley Symp. on Math. Statist. and Prob*, 1967.

[MHM20] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.