Clustering

Clusterig overview

The goal in clustering is to take unlabeled data $(x_i)_{i=1}^n$, and produce a partition so that similar data is put in the same group, and dissimilar data is put in different groups.

Remark. Before giving more details, note *why* we are doing clustering:

- So far the course made made it seems like machine learning is only classification and regression (and all data comes with labels $(y_i)_{i=1}^n$), which is far from true.
- Just as with classification, even the easiest forms of the problems here are NP-hard. But here we can point out nice algorithms that achieve good guarantees without convexity. (Originally I planned to prove these theorems, but we ran out of time.)

There are many ways to formalize the search for a partition. There are two broad classes:

- Goal is **recovery**: there are some "true" underlying clusters, and we would like an algorithm which discovers this structure.
- Goal is **fit/prediction**: we are happy so long as we find *any* set of clusters which satisfies the above goals (similar are together, dissimilar are not together).

Remark. Before explaining this further, note that we had this same issue with classification. *[Note to future matus: we should have discussed it then! Whole course is designed around prediction, which is my own bias, which thus should be de-emphasized...]* Consider the problem of least squares, which we've only formulated as minimization of

$$w \mapsto \frac{1}{2}\|Xw - Y\|_2^2.$$

- **Recovery version.** We assume that there is a *true* $\bar{w}$, and we observe pairs $(x, y) := (x, x^\top w + \epsilon)$, where x and ϵ are drawn from some distribution, the latter being zero mean noise. If we consider the optimization problem over the distrbution, we are looking for a minimum to

  $$w \mapsto \mathbb{E}((x^\top w - y)^2/2),$$

  so for instance if $\mathbb{E}(xx^\top)$ is invertible, we get $\bar{w} = \mathbb{E}(xx^\top)^{-1}\mathbb{E}(x^\top y)$. But without invertibability, there are many minima to this equation; in particular, we can no longer uniquely recover a good guess for $\bar{w}$, and the recovery problem is impossible.
- **Prediction version.** For the prediction version, we just use pseuoinverse $\mathbb{E}(xx^\top)^\dagger$, which still minimizes the error.

In clustering, this distinction leads to similar complications.

Example. Suppose data is uniformly distributed on a circle. *[Picture drawn in class.]* Therefore if someone claims that some fixed partition/clustering is good, then any rotation of that clustering should also be good (because the data is invariant to rotations). So the fit/prediction problem is easy, but the recovery problem is impossible.

Remark. As discussed in class, the choice between a recovery and predicion setup is problem-dependent. For some problems, it's really crucial to have some value of a key underlying parameter, and there are natural reasonable modeling assumptions.

Example. *[In class we also discussed mixtures of Gaussians. One interesting case that arises is distinguishing between a single wide gaussian, and a mixture of k tightly packed narrow gaussians; information-theoretically, 2^k samples are required here for recovery, but of course the prediction/fit problem is trivial. (Ankur Moitra 2010, Figure 1)]*

k-means

To make things concrete, we'll discuss the problem of k-means: we are given data $X := (x_1, \dots, x_n)$, and are expected to find k centers $S := (c_1, \dots, c_k)$ in order to keep the cost $\phi_X(S)$ small, defined as

$$\phi_X(S) := \sum_{x \in X} \min_{c \in S} \|x - c\|_2^2.$$

Remarks.

- These centers induce a partition: associate each point $x \in X$ with a closest center $S(x) := \text{argmin}_{c \in S} \|x - c\|_2$, with ties broken in some arbitrary but consistent manner (i.e., $S(x)$ is deterministic given S and x). The corresponding partition on $\mathbb{R}^d$ has measure zero on the boundaries, which correspond to the places where ties are broken.
- We'll consider a **prediction/fit** setup: we just need low cost, it does not matter if we recover some "true" partition.
- If k is chosen too small, then this cost will be force to put dissimilar items together. If k is chosen too small, then it is forced to put similar items in different groups. If the data X has some clear structure for some specific k, then this choice will lead to the goal of similar items together, dissimilar apart. But we won't discuss this scenario or the choice of k today.
- We could also formulate this as a statistical problem, where we want the quantity

  $$\mathbb{E}_x(\min_{c \in S} \|x - c\|^2)$$

  to be small, or similar we want to do well on future examples, which we penalize according to

  $$x \mapsto \min_{c \in S} \|x - c\|^2.$$

  We won't pursue there, we'll only talk about the task of minimizin $\phi_X(S)$ over some fixed finite set X.
- This problem is NP-hard when $d = 2$, and it's NP-hard to approximation. *[note to future self, some refs.]*

The standard algorithm is **Lloyd's method.** *[note to future matus, give the proper reference.]*

1. Start with some initial centers $S_0 := (c_1^{(0)}, \dots, c_k^{(0)})$.
2. For $i \in \{1, \dots, t\}$:
   a. Find a corresponding partition $(A_1, \dots, A_k)$, breaking ties in a consistent way (i.e., set $A_j := \{x \in X : S_{i-1}(x) = c_j^{(i-1)}\}$).
   b. Set $c_j^{(i)} := \text{mean}(A_j)$ for all $j \in \{1, \dots, k\}$, and $S_i := (c_1^{(i)}, \dots, c_k^{(i)})$.

Remarks.

- This algorithm converges (because there are only finitely many partitions, and a step only makes a change if cost is decreased, thus if two consecutive iterations have the same cost, the cost stays the same forever). However, the convergence can be exponentially slow in the other problem parameters *[note to matus: citation needed.]*. In practice, however, the algorithm seems fast.
- This algorithm can be unboundedly bad when compared with the optimal choice. *[picture drawn in class.]* Suppose X consists of 4 points, the corners of a rectangle. The optimum is always to pick the centers of the two shorter sides; however, the preceding algorithm can get stuck both there, and at the centers of the longer sides. Note however that this configuration is unstable to random perturbations of the data and random initializations of the centers.

A very nice algorithm for this problem is `kmeans++`. (The name was given in the *second* paper on the algorithm Arthur and Vassilvitskii (2007); the first paper was Ostrovsky et al. (2006).)

1. Choose the first center c_1 uniformly at random from X
2. For $i \in \{2, \ldots, k'\}$.
    a. Choose centers c_i randomly from X according to the following distribution: $x \in X$ is choosen proportionally to $\min_{j \leq i-1} \|x - c_j\|^2$.

There are a variety of guarantees for this method:

- If $k' = k$, then in expected value of $\phi_X(\{c_1, \ldots, c_k\})$ is $8 \ln(k)$ times the optimal cost. *[add citation..]*
- If $k' = \mathcal{O}(k/\epsilon^5)$, then with high probability the method outputs k' centers with cost at most $(2 + \epsilon)$ times the optimal cost with k centers. *[add citation..]*

References

Ankur Moitra, Greg Valiant. 2010. "Settling the Polynomial Learnability of Mixtures of Gaussians." In *FOCS*.

Arthur, D., and S. Vassilvitskii. 2007. "`k-means++`: The Advantages of Careful Seeding." In *SODA*.

Ostrovsky, Rafail, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. 2006. "The Effectiveness of Lloyd-Type Methods for the k-Means Problem." In *FOCS*.