Lecture 22: Clustering

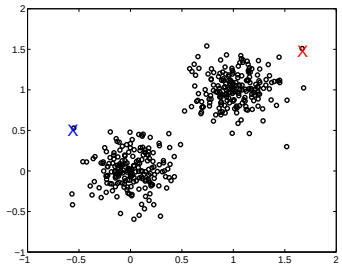
- Unsupervised learning - clustering
- K-means clustering
- Hierarchical clustering

Unsupervised learning

- In supervised learning, we have data in the form of pairs $\langle \mathbf{x}, y \rangle$, where $y = f(x)$. The goal is to approximate f
- In *unsupervised learning*, the data just contains x!
- The main goal is to find *structure* in the data
- Potential uses:
  - Visualization of the data
  - Data compression
  - Density estimation: what distribution generated data?
  - Novelty detection
- The definition of *ground truth* is often missing (no clear error function, like in supervised learning

  Usually some internal or external validation is needed

Example



- Suppose you wanted to transmit the coordinates of points drawn randomly from this set
- You are only allowed to send a limited number of bits per point (e.g., 1 or 2)
- Transmission will be *lossy*

  Loss = sum squared error between original and decoded coordinates
- We want to find an encoder/decoder that looses little information

Clustering

The original data is replaced by *clusters*, which are typically parameterized.

E.g. clusters could be Gaussian, represented by their mean and variance.

Kinds of clustering

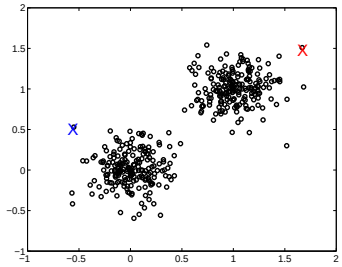
- Flat methods, e.g. K-means, mixture models
- Hierarchical methods (top-down or bottom-up)
- Other methods, e.g. spectral clustering, self-organizing maps etc.

Loss is one possible measure of success

If clustering is used as a pre-processing step, then the utility of the clusters for classification/action can be used to evaluate it as well.

K-means clustering

1. Pick a number of desired clusters, K
2. Guess K cluster center locations
3. Repeat:
   (a) Assign each example to its closest cluster center (thus, each center "owns" a set of points)
   (b) Re-compute the center location for each cluster, to be the mean of the examples assigned to that cluster



Questions

- What is K-means trying to optimize?
- Will it terminate?
- Will it find an optimal clustering?
- How should we choose the initial cluster centers?
- Can we automatically choose the number of centers?

Loss (distortion) function

- Going back to the encoding/decoding idea: the loss can be written as:

$$\sum_{i=1}^{d} (\mathbf{x_i} - \mathbf{c}_{encode(x_i)}$$

- What properties should the luster centers have to minimize this?
- $\mathbf{x_i}$ should be encoded by its closest center

  Otherwise, the loss could be reduced by replacing $encode(x_i)$ by the center closest to x_i.

Loss function analysis (continued)

- The partial derivative of the loss wrt the cluster centers has to be 0:

$$\frac{\partial Loss}{\partial c_j} = 0 \Longrightarrow c_j = \frac{1}{N(c_j)} \sum_{i} \in Ownedby(c_j)\mathbf{x_i}$$

  Hence, each center must be at the centroid (mean) of the points it owns!

- This is exactly K-means

Does the algorithm terminate?

- There is only a finite number of ways of partitioning d data points into K groups
- So there is only a finite number of possible configurations in which the cluster centers are at the centroids of the points they own
- If the configuration changes in an iteration, that the loss must have strictly improved
- Which means that with every configuration change, we get a configuration not seen before!
- Since there are only a finite number of configurations, the algorithm will terminate.

Does K-means yield the optimal configuration?

- Not necessarily!
- There are local minima
- The quality of the configuration depends on the starting centers!

Finding good configurations

- Be careful where you start
  - Place first center on top of a randomly chosen data point
  - Place second center on a data point as far away as possible from the first one
  - Place the i-th center as far away as possible from the closest of centers 1 through $i-1$
- Do random restarts (does this sound familiar?)

Choosing the number of clusters

- A difficult problem, ideas are floating around
- Minimum description length: minimize loss + complexity of the clustering
- E.g., Schwartz criterion: nr dimensions * nr centers * $\log d$

Common uses of K-means

- Often used in exploratory data analysis
- In one-dimension, it is a good way to discretize real-valued variables into non-uniform buckets
- Used in speech understanding to convert wave forms into one of k categories (vector quantization)

Hierarchical bottom-up clustering

Hierarchical agglomerative clustering:

1. Initially, each point constitutes a cluster
2. Find the two closest points (clusters) and merge them
3. Repeat previous step until we have a single cluster, containing all points

Two main requirements:

1. There must be a distance measure $d(x_i, x_j)$ between points
2. There must be a distance measure between clusters (cluster linkage)

Linkage: Measuring cluster distance

Given clusters C_k and C_l, there are several ways of measuring the distance:

- *Single linkage:*

$$d_{kl} = \min_{i \in C_k, j \in C_l} d(x_i, x_j)$$

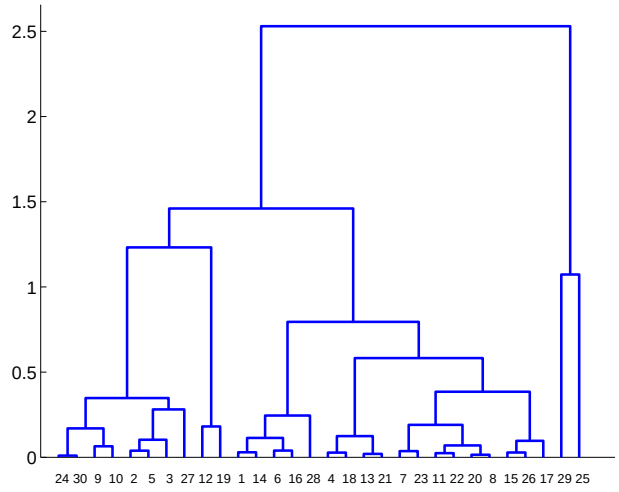
- *Average linkage:*

$$d_{kl} = \frac{1}{|C_k||C_l|} \sum_{i \in C_k, j \in C_l} d(x_i, x_j)$$

- *Centroid linkage:*

$$d_{kl} = d(\bar{x}_k, \bar{x}_l), \text{ where } \bar{x}_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$$

Example of bottom-up clustering



This representation is called a dendrogram

Comments

- It is nice to get a hierarchy instead of an unstructured collection of groups
- If you want to get K clusters, just cut out the $k-1$ longest links
- Not well-founded theoretically...