Data Mining

Practical Machine Learning Tools and Techniques

Slides for Sections 4.8 and 6.6

(Skip incremental clustering and category utility)

Clustering

- Clustering techniques apply when there is no class to be predicted
- Aim: divide instances into "natural" groups
- As we've seen clusters can be:
  - disjoint vs. overlapping
  - deterministic vs. probabilistic
  - flat vs. hierarchical
- We'll look at a classic clustering algorithm called *k-means*
  - *k-means* clusters are disjoint, deterministic, and flat

General Setting

- Given a dataset D and a distance metric d(.,.), partition D into groups of "similar" items:
  - Items in a group are similar to each other;
  - Items in different groups are as dissimilar as possible.
- Similarity is based on the distance metric used.

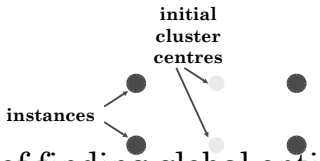
The *k*-means algorithm

To cluster data into *k* groups:(*k* is predefined)

1. Choose *k initial* cluster centers (for example, *k* random points)
2. repeat (until centers don't change)
   - Assign instances to clusters based on distance to cluster centers
   - Recompute centers by computing the *centroids* of clusters

Discussion

- Algorithm minimizes squared distance to cluster centers
- Result can vary significantly
  - based on initial choice of seeds
- Can get trapped in local minimum
  - Example:



- To increase chance of finding global optimum: restart with different random seeds
- Can we applied recursively with $k = 2$

Clustering: how many clusters?

- How to choose k in k-means? Possibilities:
  - Choose k that minimizes cross-validated squared distance to cluster centers
  - Apply k-means recursively with $k = 2$ and use stopping criterion
    - Seeds for subclusters can be chosen by seeding along direction of greatest variance in cluster (one standard deviation away in each direction from cluster center of parent cluster)
    - Implemented in algorithm called *X*-means

Hierarchical Clustering

- Repeat (until one cluster)

  {1. Put each item in a cluster by itself;
  2. Merge "closest" clusters}

- Distance between two clusters: (i) distance between closest points; (ii) average distance between all pairs of points in the two clusters; or (iii) maximum distance between any pair of points in the two clusters
- Another divisive strategy – start with the whole data and split.

Probability-based clustering

- Probabilistic perspective ⇒ seek the *most likely* clusters given the data
- Also: instance belongs to a particular cluster *with a certain probability*

Finite mixtures

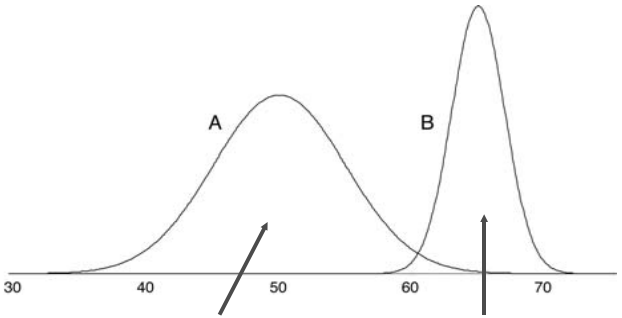
- Model data using a *mixture* of distributions
- One cluster, one distribution
  - governs probabilities of attribute values in that cluster
- *Finite mixtures* : finite number of clusters
- Individual distributions are normal (usually)
- Combine distributions using cluster weights

Two-class mixture model

data

A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	A	41
A	46	A	48	B	62	B	66	A	48		
A	45	A	49	A	43	B	65	B	64		
A	45	A	46	A	40	A	46	A	48		

model



μ_A=50, σ_A =5, p_A=0.6 μ_B=65, σ_B =2, p_B=0.4

Using the mixture model

- Probability that instance x belongs to cluster A:

$$Pr[A|x]=\frac{Pr[x|A]Pr[A]}{Pr[x]}=\frac{f(x;\mu_A,\sigma_A)p_A}{Pr[x]}$$

with

$$f(x;\mu,\sigma)=\frac{1}{\sqrt{(2\pi)}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Probability of an instance given the clusters:

$$Pr[x|\text{the\_clusters}]=\sum_i Pr[x|\text{cluster}_i]Pr[cluster_i]$$

Learning the clusters

- Assume:
  - we know there are k clusters
- Learn the clusters ⇒
  - determine their parameters
  - I.e. means and standard deviations
- Performance criterion:
  - *probability of training data given the clusters*
- EM algorithm
  - finds a local maximum of the likelihood

EM algorithm

- EM = Expectation-Maximization
  - Generalize *k*-means to probabilistic setting
- Iterative procedure:
  - E "expectation" step:
    Calculate cluster probability for each instance
  - M "maximization" step:
    Estimate distribution parameters from cluster probabilities
- Store cluster probabilities as instance weights
- Stop when improvement is negligible

Expectation Step

- Probability w that instance x belongs to cluster A:

$$w_i = Pr[A \mid x_i] = \frac{Pr[x_i \mid A]Pr[A]}{Pr[x_i]} = \frac{f(x_i; \mu_A, \sigma_A)p_A}{Pr[x_i]}$$

with

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{(2\pi)}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Probability of an instance given the clusters:

$$Pr[x_i] = \sum_j Pr[x_i \mid \text{cluster}_j] Pr[cluster_j]$$

Maximization Step

- Estimate parameters from weighted instances

$$\mu_A = \frac{w_1 x_1 + w_2 x_2 + \ldots + w_n x_n}{w_1 + w_2 + \ldots + w_n}$$

$$\sigma_A = \frac{w_1 (x_1 - \mu)^2 + w_2 (x_2 - \mu)^2 + \ldots + w_n (x_n - \mu)^2}{w_1 + w_2 + \ldots + w_n}$$

$$p_A = \sum_i w_i / n$$

- Stop when log-likelihood saturates

$$\sum_i \log(p_A Pr[x_i \mid A] + p_B Pr[x_i \mid B])$$