Lec Mon

Monday, November 26, 2018 4:06 PM

Today: Unsupervised Learning

Clustering

Given data $\{x_n\}_{n=1}^N$ and some integer K, we would like to separate the data into prototypes (or centroids) of clusters $\{\mu_k\}_{k=1}^K$

eg

- Identifying communities within social networks
- Finding topics in news stories
- Grouping similar sequences into gene families

K-means

- guess locations of centroids (often wrong, initially), assign data points based on distance to centroids
- then update centroids to average positions of each cluster
- then update the clusters based on distance again
- then update centroids
- etc

Intuition: data points assigned to cluster k should be close to μ_k, the prototype

Distortion measure (objective/cost function)

$$J(\{r_{nk}\},\{\mu_k\}) = \sum_n \sum_k r_{nk}\|x_n - \mu_k\|_2^2$$

where $r_{nk} \in \{0,1\}$

$r_{nk} = 1 \; iff \; A(x_n) = k$

K-means objective is to find $\{r_{nk}\},\{\mu_k\}$ that minimize objective function

minimization is NP-hard (it's not convex)

so we'll rely on a heuristic:

Lloyd's algorithm

- often just called the K-means algorithm

1. initialize $\{\mu_k\}$ to some values
2. Assume the current value of $\{\mu_k\}$ fixed, minimize J over $\{r_{nk}\}$, which leads to the following cluster assignment rule:
   a. $r_{nk} = 1 \; if \; k = argmin_j \left\|x_n - \mu_j\right\|_2^2$
   b. $r_{nk} = 0 \; otherwise$
3. assume current value of $\{r_{nk}\}$ fixed, minimize J over $\{\mu_k\}$, which leads to the following rule to update the prototypes of the clusters
   a. $\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$
4. stop if the objective function J stays the same or return to step 2

Application: vector quantization

lossy compression of image - clustering pixels and vector quantizing them to produce an image with pixels of limited colors (K colors)

- Does the K-means algorithm converge?
  - yes
- How long does it take to converge?
  - in the worst case, exponential in number of data points
  - in practice, very quick

Properties of K-means algorithm
- how good is the K-means solution?
  - converges to a local minimum
  - solution depends on initialization
  - in practice, run many times with different initializations and pick the best
  - K-means++ is a neat approximation algorithm that has theoretical guarantees on the final value of the objective
    - still no guarantee we reach the global minimum, but we'll get close

Other practical issues
- choosing K
  - increasing K always decreases optimal value of K-means objective
    - analogous to overfitting in supervised learning
  - a heuristic - pick the k that has a relatively low optimal value but isn't too high
  - information criteria that effectively regularize more complex models

K-medoids
- K-means is sensitive to outliers
- in some applications we want the prototypes to be one of the points
- leads to K-medoids

Algorithm:
1. initialize $\{\mu_k\}$ by randomly selecting K of the N points
2. Assume the current value of $\{\mu_k\}$ fixed, minimize J over $\{r_{nk}\}$, which leads to the following cluster assignment rule:
   a. $r_{nk} = 1 \; if \; k = argmin_j \left\| x_n - \mu_j \right\|_2^2$
   b. $r_{nk} = 0 \; otherwise$
3. assume current value of $\{r_{nk}\}$ fixed, update the prototype of cluster K. in K-mediods, the prototype for a cluster is the data point that is closest to all other data points in the cluster
   a. $k^* = argmin_{mr_{mk}=1} \sum_n r_{nk} \|x_n - x_m\|_2^2$
   b. $\mu_k = x_k^*$
4. stop if the objective function J stays the same or return to step 2

What about a probabilistic interpretation of clustering?
Gaussian mixture models - we'll use a mixture of multiple Gaussian models (normal distributions) on our data
formal definition: Gaussian mixture model of K Gaussians has the following density function for x

$$p(x) = \sum_{k=1}^{K} \omega_k N(x|\mu_k, \Sigma_k)$$

where μ_k and Σ_k are mean and covariance matrices of kth component

GMM as marginal distribution of a joint distribution
Consider the following joint distribution p(x, z) = p(z)p(x|z)
where z is a discrete random variable taking values between 1 and K
Denote $\omega_k = p(z = k)$
now assume the conditional distributions are Gaussian distributions
$p(x|z = k) = N(x|\mu_k, \Sigma_k)$
then the marginal distribution of x is

$$p(x) = \sum_{k=1}^{K} \omega_k N(x|\mu_k, \Sigma_k)$$

Parameter estimation for Gaussian mixture models
Assume we have labels z. We have complete data D' and incomplete data D. given D', the maximum likelihood estimation of the θ is given by

$$\theta = argmax\ logP(D') = \sum_n \log p(x_n, z_n)$$

The complete likelihood is decomposable

$$\sum_k \sum_{n:z_n=k} \log p(z_n) p(x_n|z_n)$$

where we have grouped data by its values z_n. Let's introduce a binary variable γ_{nk} to indicate whether z_n = k

$$\sum_k \sum_n \gamma_{nk} \log p(z = k) p(x_n|z = k)$$

… some more math that is hard to write down (see slides)

Back to incomplete data
We can guess z_n using its posterior probability. Compute this with Bayes' Theorem.
In order to do this, we assume we know the parameters θ

Estimation with soft γ_{nk}
define $\gamma_{nk} = p(z_n = k|x_n)$

- recall that γ_{nk} binary
- "soft" assignment of x_n to kth component
- each x_n is assigned to a component fractionally according to $p(z_n=k|x_n)$
- now we get the same expression for MLE as before
- but we're cheating by using θ

Iterative procedure

1. initialize theta
2. compute γ_{nk}
3. recompute theta
4. repeat