CLUSTERING BEYOND K-MEANS

David Kauchak
CS 451 – Fall 2013

Administrative

Final project
- Presentations on Friday
  - 3 minute max
  - 1-2 PowerPoint slides. E-mail me by 9am on Friday
  - What problem you tackled and results
- Paper and final code submitted on Sunday

Final exam next week

K-means

Start with some initial cluster centers

Iterate:
- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

Problems with K-means

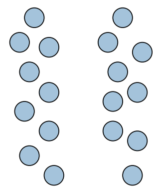
Determining K is challenging

Spherical assumption about the data (distance to cluster center)

Hard clustering isn't always right

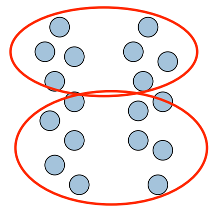
Greedy approach

Problems with K-means



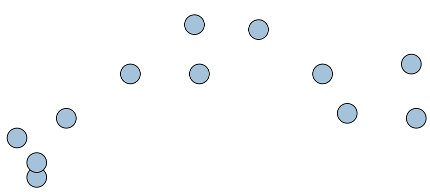
What would K-means give us here?

Assumes spherical clusters

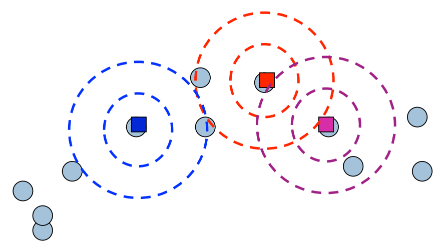


k-means assumes spherical clusters!

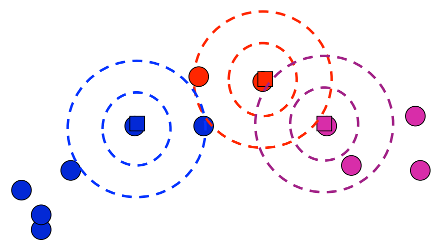
K-means: another view



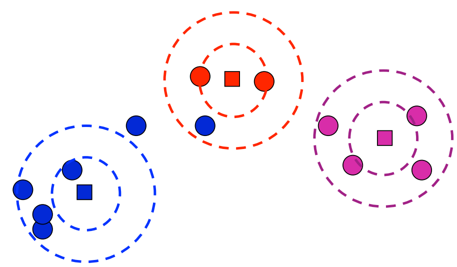
K-means: another view



K-means: assign points to nearest center



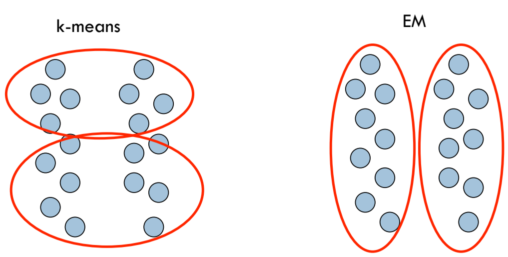
K-means: readjust centers



Iteratively learning a collection of spherical clusters

EM clustering: mixtures of Gaussians

Assume data came from a mixture of Gaussians (**elliptical data**), assign data to cluster with a certain *probability*



EM clustering

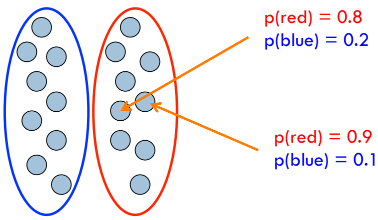
Very similar at a high-level to K-means

Iterate between assigning points and recalculating cluster centers

Two main differences between K-means and EM clustering:

1. We assume elliptical clusters (instead of spherical)
2. It is a "soft" clustering algorithm

Soft clustering



EM clustering

Start with some initial cluster centers

Iterate:

- soft assigned points to each cluster

  Calculate: $p(\theta_c | x)$

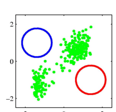
  the probability of each point belonging to each cluster

- recalculate the cluster centers

  Calculate new cluster parameters, θ_c

  maximum likelihood cluster centers given the current soft clustering

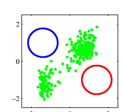
EM example



Start with some initial cluster centers

Figure from Chris Bishop

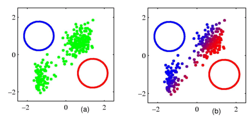
Step 1: soft cluster points



Which points belong to which clusters (soft)?

Figure from Chris Bishop

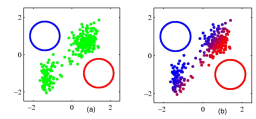
Step 1: soft cluster points



Notice it's a soft (probabilistic) assignment

Figure from Chris Bishop

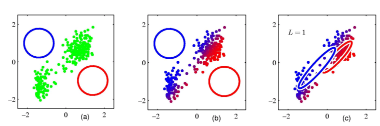
Step 2: recalculate centers



What do the new centers look like?

Figure from Chris Bishop

Step 2: recalculate centers



Cluster centers get a **weighted** contribution from points

Figure from Chris Bishop

keep iterating…

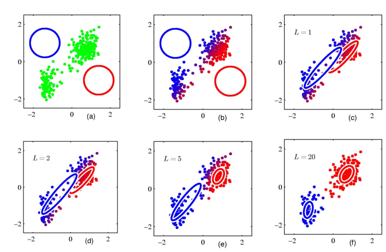


Figure from Chris Bishop

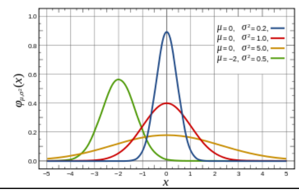
Model: mixture of Gaussians

How do you define a Gaussian (i.e. ellipse)?
In 1-D?
In M-D?

Gaussian in 1D

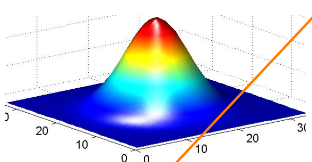
$$f(x;\sigma,\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

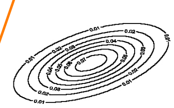
parameterized by the mean and the standard deviation/variance



Gaussian in multiple dimensions

$$N[x;\mu,\Sigma] = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma)}}\exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)]$$

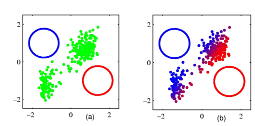




Covariance determines the shape of these contours

We learn the means of each cluster (i.e. the center) and the covariance matrix (i.e. how spread out it is in any given direction)

Step 1: soft cluster points



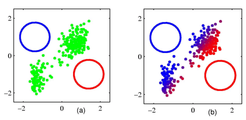
- soft assigned points to each cluster

Calculate: $p(\theta_c|x)$

the probability of each point belonging to each cluster

How do we calculate these probabilities?

Step 1: soft cluster points



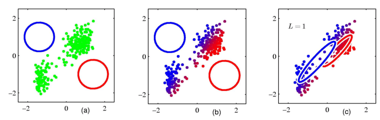
- soft assigned points to each cluster

Calculate: $p(\theta_c|x)$

the probability of each point belonging to each cluster

Just plug into the Gaussian equation for each cluster!
(and normalize to make a probability)

Step 2: recalculate centers



Recalculate centers:

calculate new cluster parameters, θ_c

maximum likelihood cluster centers given the current soft clustering

How do calculate the cluster centers?

Fitting a Gaussian

What is the "best"-fit Gaussian for this data?

10, 10, 10, 9, 9, 8, 11, 7, 6, …

Recall this is the 1-D Gaussian equation:

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

Fitting a Gaussian

What is the "best"-fit Gaussian for this data?

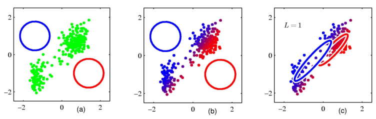
10, 10, 10, 9, 9, 8, 11, 7, 6, …

The MLE is just the mean and variance of the data!

Recall this is the 1-D Gaussian equation:

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

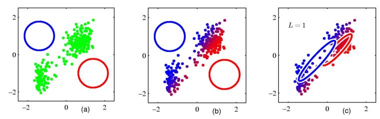
Step 2: recalculate centers



Recalculate centers:
- Calculate θ_c
- maximum likelihood cluster centers given the current ***soft clustering***

How do we deal with "soft" data points?

Step 2: recalculate centers



Recalculate centers:
- Calculate θ_c
- maximum likelihood cluster centers given the current ***soft clustering***

Use fractional counts!

E and M steps: creating a better model

EM stands for Expectation Maximization

Expectation: Given the current model, figure out the expected probabilities of the data points to each cluster

$p(\theta_c|x)$ What is the probability of each point belonging to each cluster?

Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

Just like NB maximum likelihood estimation, except we use fractional counts instead of whole counts

Similar to *k*-means

Iterate:
- Assign/cluster each point to closest center
  - Expectation: Given the current model, figure out the expected probabilities of the points to each cluster $p(\theta_c|x)$
- Recalculate centers as the mean of the points in a cluster
  - Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

E and M steps

Expectation: Given the current model, figure out the expected probabilities of the data points to each cluster

Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

Iterate:

each iterations increases the likelihood of the data and guaranteed to converge (though to a local optimum)!

EM

EM is a general purpose approach for training a model when you don't have labels

Not just for clustering!
- K-means is just for clustering

One of the most general purpose unsupervised approaches
- can be hard to get right!

EM is a general framework

Create an initial model, θ'
- Arbitrarily, randomly, or with a small set of training examples

Use the model θ' to obtain another model θ such that

$\sum_i \log P_\theta(data_i) > \sum_i \log P_{\theta'}(data_i)$ i.e. better models data (increased log likelihood)

Let θ' = θ and repeat the above step until reaching a local maximum
- Guaranteed to find a better model after each iteration

Where else have you seen EM?

EM shows up all over the place

Training HMMs (Baum-Welch algorithm)

Learning probabilities for Bayesian networks

EM-clustering

Learning word alignments for language translation

Learning Twitter friend network

Genetics

Finance

Anytime you have a model and unlabeled data!

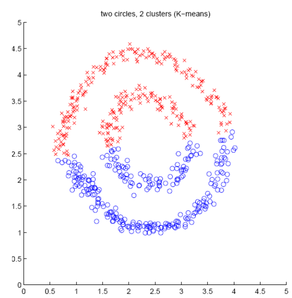
Other clustering algorithms

K-means and EM-clustering are by far the most popular for clustering

However, they can't handle all clustering tasks

What types of clustering problems can't they handle?

Non-gaussian data

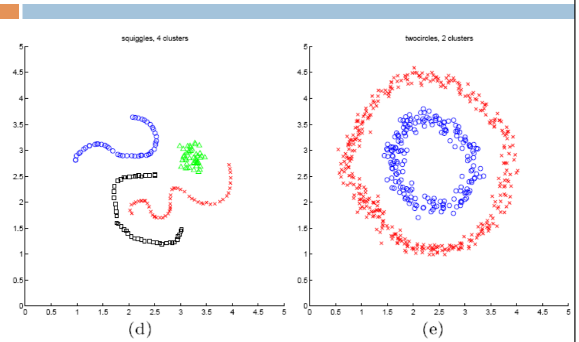


What is the problem?

Similar to classification: global decision (linear model) vs. local decision (K-NN)

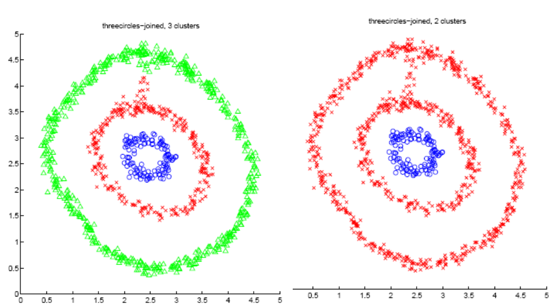
Spectral clustering

Spectral clustering examples



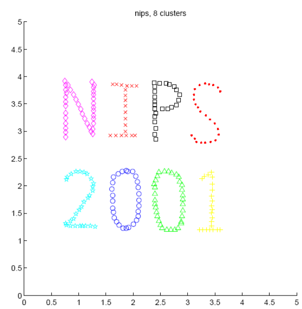
Ng et al On Spectral clustering: analysis and algorithm

Spectral clustering examples



Ng et al On Spectral clustering: analysis and algorithm

Spectral clustering examples



Ng et al On Spectral clustering: analysis and algorithm