Administrative

Assignment 8 back

Final project status reports due Wednesday

Next class: skim the papers

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

Hard clustering isn't always right

Assumes clusters are spherical

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 (soft clustering)

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**

$p(\text{red}) = 0.8$
$p(\text{blue}) = 0.2$

$p(\text{red}) = 0.9$
$p(\text{blue}) = 0.1$

---

**EM clustering**

Start with some initial cluster centers

**Iterate:**
- **soft assign** points to each cluster
  
  Calculate: $p(x; \theta_j)$
  the probability of each point belonging to each cluster

- recalculate the cluster centers
  
  Calculate new cluster parameters, $\theta_j$
  maximum likelihood cluster centers given the current soft clustering

---

**EM example**

Start with some initial cluster centers

---

**Step 1: soft cluster points**

Which points belong to which clusters (soft)?
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...
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

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

Covariance determines the shape of these contours

Step 1: soft cluster points

- soft assign points to each cluster
  Calculate: $p(x; \theta)$
  the probability of each point belonging to each cluster

How do we calculate these probabilities?
Step 1: soft cluster points

- soft assign points to each cluster
  Calculate: \( p(x; \theta_c) \)
  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, \( \theta \),
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:

\[
 f(x; \sigma, \theta) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\theta)^2}{2\sigma^2}}
\]
Step 2: recalculate centers

Recalculate centers:
Calculate $\theta_c$, maximum likelihood cluster centers given the current soft clustering.

How do we deal with “soft” data points?

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.

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

**Maximization:** Given the probabilistic assignment of all the points, estimate a new model, $\theta_c$.

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.

$$p(x; \theta_c)$$

Recalculate centers as the mean of the points in a cluster.

Maximization: Given the probabilistic assignment of all the points, estimate a new model, $\theta_c$. 


**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, $\theta$. 

**Iterate**: each iteration increases the likelihood of the data and is 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, $\theta'$
- Arbitrarily, randomly, or with a small set of training examples.

Use the model $\theta'$ to obtain another model $\theta$ such that

\[
\sum \log p_y(d) > \sum \log p_y(d) \quad \text{i.e. better models data (increased log likelihood)}
\]

Let $\theta' = \theta$ 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!
In machine translation, we train from pairs of translated sentences. Often useful to know how the words align in the sentences.

Use EM:
- learn a model of $P(\text{french-word} \mid \text{english-word})$

“la” and “the” observed to co-occur frequently, so $P(\text{la} \mid \text{the})$ is increased.

“house” co-occurs with both “la” and “maison”, but $P(\text{maison} \mid \text{house})$ can be raised without limit, to 1.0, while $P(\text{la} \mid \text{house})$ is limited because of “the” (pigeonhole principle)
Finding Word Alignments

settling down after another iteration

… la maison … la maison bleue … la fleur …
… the house … the blue house … the flower …

Inherent hidden structure revealed by EM training!
For details, see
- “A Statistical MT Tutorial Workbook” (Knight, 1999).
- 37 easy sections, final section promises a free beer.
- Software: GIZA++

Statistical Machine Translation

P(maison | house) = 0.411
P(maison | building) = 0.027
P(maison | manson) = 0.020

Estimating the model from training 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
What Is A Good Clustering?

Internal criterion: A good clustering will produce high quality clusters in which:
- the intra-class (that is, intra-cluster) similarity is high
- the inter-class similarity is low

How would you evaluate clustering?

Common approach: use labeled data

Use data with known classes
- For example, document classification data

Purity, the proportion of the dominant class in the cluster

Cluster I: Purity = \( \frac{\max(3, 1, 0)}{4} = \frac{3}{4} \)
Cluster II: Purity = \( \frac{\max(1, 4, 1)}{6} = \frac{1}{6} \)
Cluster III: Purity = \( \frac{\max(2, 0, 3)}{5} = \frac{3}{5} \)

Overall purity?

Cluster I: Purity = \( \frac{\max(3, 1, 0)}{4} \)
Cluster II: Purity = \( \frac{\max(1, 4, 1)}{6} \)
Cluster III: Purity = \( \frac{\max(2, 0, 3)}{5} \)

Overall purity:

Cluster average:
\[
\frac{3 \times \frac{3}{4} + 4 \times \frac{1}{6} + 3 \times \frac{3}{5}}{3} = 0.672
\]

Weighted average:
\[
\frac{3 \times \frac{3}{4} + 6 \times \frac{1}{6} + 5 \times \frac{3}{5}}{15} = \frac{3 + 4 + 3}{15} = 0.667
\]

Overall purity?
Purity issues...

Purity, the proportion of the dominant class in the cluster

Good for comparing two algorithms, but not understanding how well a single algorithm is doing, why?
- Increasing the number of clusters increases purity.

Purity isn’t perfect

Which is better based on purity?
Which do you think is better?
Ideas?

Common approach: use labeled data

Average entropy of classes in clusters

\[ \text{entropy(cluster)} = - \sum_i p(\text{class}_i) \log p(\text{class}_i) \]

where \( p(\text{class}_i) \) is proportion of class \( i \) in cluster.

Common approach: use labeled data

Average entropy of classes in clusters

\[ \text{entropy(cluster)} = - \sum_i p(\text{class}_i) \log p(\text{class}_i) \]
Common approach: use labeled data

Average entropy of classes in clusters

$$\text{entropy}(\text{cluster}) = - \sum p(\text{class}_i) \log p(\text{class}_i)$$

$$-0.5 \log 0.5 - 0.5 \log 0.5 = 1$$

$$-0.5 \log 0.5 - 0.25 \log 0.25 - 0.25 \log 0.25 = 1.5$$

Where we’ve been!

How many slides?

1,385 slides

Where we’ve been!

Our ML suite:

How many classes?

29 classes

How many lines of code?

2951 lines of code
Where we’ve been!

Our ML suite:
- Supports 7 classifiers
  - Decision Tree
  - Perceptron
  - Average Perceptron
  - Gradient descent
  - 2 loss functions
  - 2 regularization methods
  - K-NN
  - Naïve Bayes
  - 2 layer neural network
- Supports two types of data normalization
  - Feature normalization
  - Example normalization
- Supports two types of meta-classifiers
  - OVA
  - AWA

Where we’ve been!

Hadoop!
- 532 lines of hadoop code in demos

Where we’ve been!

Geometric view of data
Model analysis and interpretation (linear, etc.)
Evaluation and experimentation
Probability basics
Regularization (and priors)
Deep learning
Ensemble methods
Unsupervised learning (clustering)