CS 294-5: Statistical Natural Language Processing

Text Clustering, EM
Lecture 6: 9/19/05
Guest Lecturer:
Teg Grenager, Stanford University

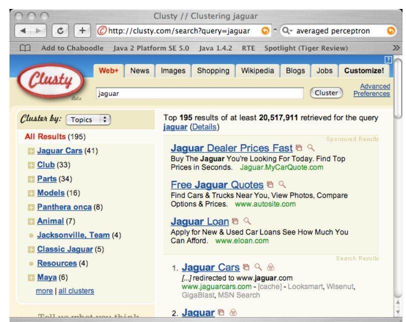
Overview

- So far: Classification
  - Applications: text categorization, language identification, word sense disambiguation
  - Generative models: Naïve Bayes
  - Discriminative models: maximum entropy models (a.k.a. logistic regression)
  - “Supervised” learning paradigm
- Today: Clustering
  - “Unsupervised” learning: no class labels to learn from
  - Magic: discovers hidden patterns in the data
  - Useful in a range of NLP tasks: IR, smoothing, data mining, exploratory data analysis
- Please interrupt me (I hear you’re good at that!)

Ambiguous web queries

- Web queries are often truly ambiguous:
  - jaguar
  - NLP
  - paris hilton
- Seems like word sense ambiguation should help
  - Different senses of jaguar: animal, car, OS X…
- In practice WSD doesn’t help for web queries
  - Disambiguation is either impossible (“jaguar”) or trivial (“jaguar car”)
- Better to let the user decide
- “Cluster” the results into useful groupings

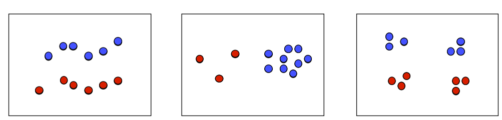
Demo: Meet “Clusty”



How’d they do that?

- Text categorization
  - Label data and build a MaxEnt classifier for every major disambiguation decision
  - Expensive, impractical for open domain
- Many clustering methods have been developed
  - Most start with a pairwise distance function
  - Most can be interpreted probabilistically (with some effort)
  - Axes: flat / hierarchical, agglomerative / divisive, incremental / iterative, probabilistic / graph theoretic / linear algebraic
- Our focus: “model-based” vs. “model-free”
  - **Model-Free:** Define a notion of “page similarity”, and put similar things together in clusters (heuristic, agglomerative)
  - **Model-Based:** Define a generative probabilistic model over the pages and their clusters, and search for parameters which maximize data likelihood (probabilistic, generative)

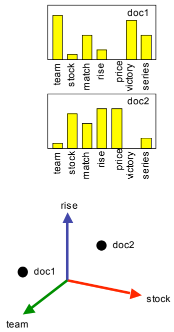
Point Clustering



- Task: group points into clusters
- Here we illustrate with simple two-dimensional point examples
- Warning: quite different from text clustering
  - Featural representations of text will typically have a large number of dimensions (10^3 - 10^6)
  - Euclidean distance isn’t necessarily the best distance metric for featural representations of text

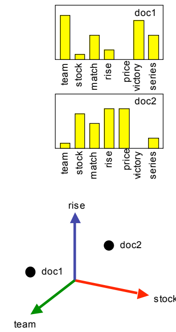
Two Views of Documents

- Probabilistic
  - A document is a collection of words sampled from some distribution, an empirical distribution
  - Correlations between words flows through hidden model structure
  - Distance: divergences
- Vector Space
  - A document is a point in a high-dimensional vector space
  - Correlations between words reflects low rank of valid document subspace
  - Distance: Euclidean / cosine



High-Dimensional Data

- Both of these pictures are totally misleading!
  - Documents are zero in almost all axes
  - Most document pairs are very far apart (i.e. not strictly orthogonal, but only share very common words and a few scattered others)
  - In classification terms: virtually all document sets are separable, for most any classification



Model-Based Clustering

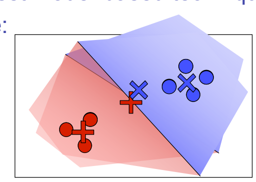
- Document clustering with probabilistic models:

Unobserved (C)	Observed (X)
c_1	LONDON -- Soccer team wins match…
c_2	NEW YORK – Stocks close up 3%…
c_2	Investing in the stock market has…
c_1	The first game of the world series…

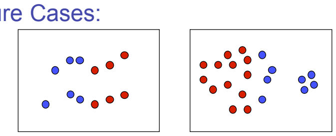
Find C and θ to maximize $P(X,C|\theta)$

k-Means Clustering

- The simplest model-based technique
- Procedure:



- Failure Cases:

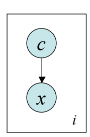


Mixture Models

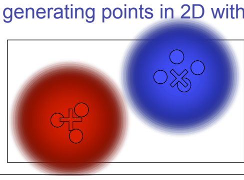
- Consider models of the form:

$$\mathrm{P}(\mathbf{x},\mathbf{c}) = \prod_i \mathrm{P}(c_i)\mathrm{P}(x_i|c_i)$$

The observed data instances | The clusters they belong to | Prior probability of cluster *i* | Prob of cluster generating data instance *i*



- Example: generating points in 2D with Gaussian

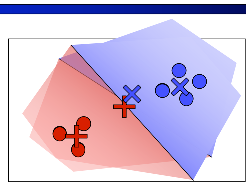


Learning with EM

$$\mathrm{P}(\mathbf{x},\mathbf{c}) = \prod_i \mathrm{P}(c_i)\mathrm{P}(x_i|c_i)$$

- Recall that in supervised learning, we search for model parameters which maximize data likelihood
  - Not guaranteed to work well, but it's a reasonable thing to do and we know how to do it
  - Maximum likelihood estimation is trivial in a generative model: can compute in closed form from data counts
- Can we do that here?
  - We could if we knew the cluster labels c_i
- Iterative procedure (Expectation-Maximization):
  1. Guess some initial parameters for the model
  2. Use model to make best guesses of c_i (E-step)
  3. Use the new complete data to learn better model (M-step)
  4. Repeat steps 2 and 3 until convergence

k-Means is Hard EM



- Iterative procedure (Expectation-Maximization):
  1. Guess some initial parameters for the model
  2. Use model to make best guesses of c_i (E-step)
  3. Use the new complete data to learn better model (M-step)
  4. Repeat steps 2 and 3 until convergence

EM in Detail

$$P(\mathbf{x}, \mathbf{c}) = \prod_i P(c_i) P(x_i | c_i)$$

- Expectation step
  - Using current model parameters, do probabilistic inference to compute the probability of the cluster labels c

$$Q_i^{(t)}(c_i) := P_{\theta^{(t)}}(c_i | x_i) = \frac{P_{\theta^{(t)}}(c_i) P_{\theta^{(t)}}(x_i | c_i)}{\sum_{c_i} P_{\theta^{(t)}}(c_i) P_{\theta^{(t)}}(x_i | c_i)}$$

  - These Q's can viewed as "soft completions" of the data
  - Note: k-Means approximates this Q function with the max
- Maximization step
  - Compute the model parameters which maximize the log likelihood of the "completed" data (can do in closed form)

$$\theta^{(t+1)} = \arg\max_\theta \sum_i \sum_{c_i} Q_i^{(t)}(c_i) \log P_\theta(x_i, c_i)$$

EM Properties

- EM is a general technique for learning anytime we have incomplete data (x,y)
  - Convenience Scenario: we want P(x), including y just makes the model simpler (e.g. mixing weights)
  - Induction Scenario: we actually want to know y (e.g. clustering)
  - You'll see it again in this course!
- Each step of EM is guaranteed to increase data likelihood - a hill climbing procedure
- Not guaranteed to find global maximum of data likelihood
  - Data likelihood typically has many local maxima for a general model class and rich feature set
  - Many "patterns" in the data that we can fit our model to…

EM Monotonicity Proof

$$\ell(\theta^{(t)}) = \sum_i \log P_{\theta^{(t)}}(x_i) = \sum_i \log \sum_{c_i} P_{\theta^{(t)}}(x_i, c_i)$$

$$\geq \sum_i \log \sum_{c_i} Q_i^{(t-1)}(c_i) \frac{P_{\theta^{(t)}}(x_i, c_i)}{Q_i^{(t-1)}(c_i)}$$

Multiply by 1

$$\geq \sum_i \sum_{c_i} \log Q_i^{(t-1)}(c_i) \frac{P_{\theta^{(t)}}(x_i, c_i)}{Q_i^{(t-1)}(c_i)}$$

Jensen's inequality for concave function f: f(E[x]) ≥ E[f(x)]

$$\geq \sum_i \sum_{c_i} \log Q_i^{(t-1)}(c_i) \frac{P_{\theta^{(t-1)}}(x_i, c_i)}{Q_i^{(t-1)}(c_i)}$$

We had chosen $\theta^{(t)}$ to be the max, so any other θ is worse.

$$= \sum_i \log \sum_{c_i} Q_i^{(t-1)}(c_i) \frac{P_{\theta^{(t-1)}}(x_i, c_i)}{Q_i^{(t-1)}(c_i)} = \ell(\theta^{(t-1)})$$

Uhoh! Jensen's would go the wrong way!

where $Q_i^{(t-1)}(c_i) := P_{\theta^{(t-1)}}(c_i | x_i)$

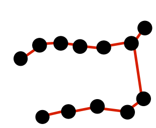
EM For Text Clustering

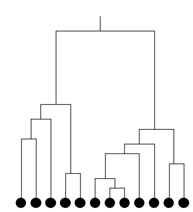
$$P(\mathbf{x}, \mathbf{c}) = \prod_i P(c_i) P(x_i | c_i)$$

- Remember, we care about documents, not points
- How to model probability of a document given a class?
  - Probabilistic: Naïve Bayes $P(x_i | c_i) = \prod_j P(w_{ij} | c_i)$
    - Doesn't represent differential feature weighting
  - Vector Space: Gaussian $P(x_i | c_i) = P(\mathbf{f}(x_i) | c_i) \sim \mathcal{N}(\mu, \Sigma)$
    - Euclidean distance assumption isn't quite right

Agglomerative Clustering

- Most popular heuristic clustering methods
- Big idea: pick up similar documents and stick them together, repeat
- Point Example (single link):

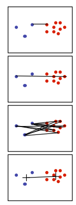




- You get a cluster hierarchy for free

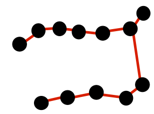
Agglomerative Choices

- Choice of distance metric between instances:
  - Euclidean distance (L2-norm) - equivalent to vector space model
  - KL-divergence - equivalent to probabilistic model
- Choice of distance metric between clusters:
  - Single-link: distance between closest instances in clusters
  - Complete-link: distance between furthest instances in clusters
  - Average-link: average distance between instances in clusters
  - Ward's method: difference between sum squared error to centroid of combined cluster and separate clusters

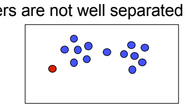


Single-Link Clustering

- Procedure:



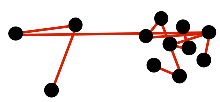
- Failure Cases
  - Fails when clusters are not well separated (often!)



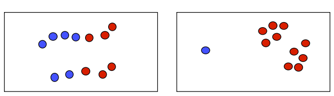
- Model Form:
  - Corresponds to fitting a model where instances in each cluster were generated by a *random walk* though the space

Complete-Link Clustering

- Procedure:

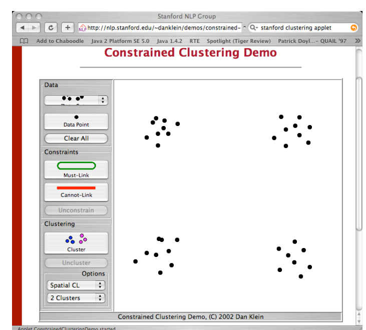


- Failure Cases
  - Fails when clusters aren't spherical, or of uniform size



- Model Form
  - Corresponds to fitting a model where instances in each cluster are generated in *uniform spheres* around a centroid

Clustering Demo



Clustering Method Summary

- Agglomerative methods:
  - Pro: easy to code
  - Pro: you get a hierarchy of clusters for free
  - Pro/Con: you don't have to explicitly propose a model (but your distance metrics imply one anyway)
  - Con: runtime > n^2, which becomes prohibitive
- Model-based methods:
  - Pro/Con: you're forced to propose an explicit model
  - Pro: usually quick to converge
  - Con: very sensitive to initialization
  - Con: how many clusters?

Clustering vs. Classification

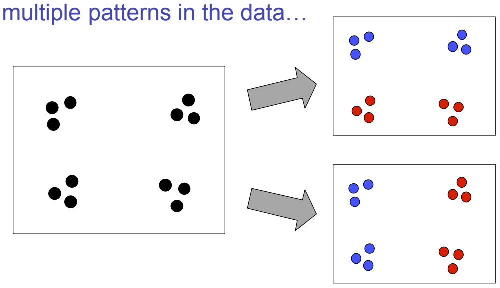
- Classification: we specify which pattern we want, features uncorrelated with pattern are idle

P(w|sports)	P(w|politics)	P(w|headline)	P(w|story)
the 0.1	the 0.1	the 0.05	the 0.1
game 0.02	game 0.005	game 0.01	game 0.01
win 0.02	win 0.01	win 0.01	win 0.01

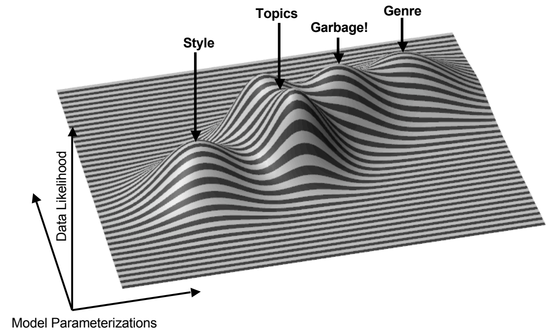
- Clustering: clustering procedure locks on to whichever pattern is most salient
  - P(content words | class) will learn topics
  - P(length, function words | class) will learn style
  - P(characters | class) will learn "language"

Multiple Patterns

- Even with the same model class, there are multiple patterns in the data…

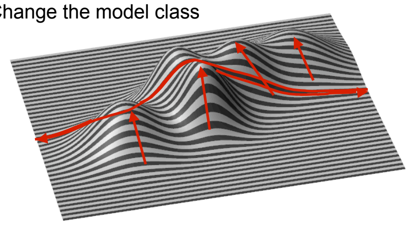


Multiple Patterns

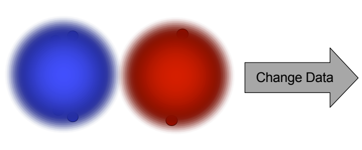


Multiple Patterns

- Ways to deal with it
  - Change the data itself
  - Change the search procedure (including smart initialization)
  - Change the model class



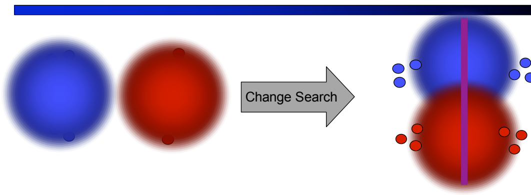
Multiple Patterns



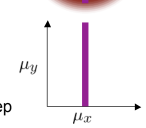
1D Projection

- Examples:
  - Remove stopwords from documents
  - Use dimensionality reduction techniques to change featural representation

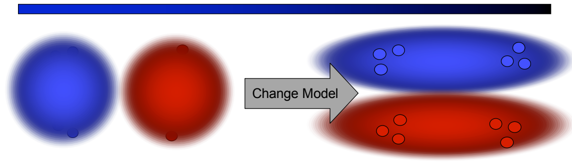
Multiple Patterns



- Examples:
  - Smart initialization of the search
  - Search a subspace by only reestimating some of the model parameters in the M-step



Multiple Patterns



- Examples:
  - Add heuristic feature weighting such as inverse document frequency (IDF)
  - Add a hierarchical emission model to Naïve Bayes
  - Limit the form of the covariance matrix in a Gaussian

Clustering Problems

- There are multiple patterns in the data, basic approach will just give you the most salient one
- Relationship between the data representation and the model class is complex and not well understood
- Data likelihood isn't usually what you want to maximize
- Can't find the global maximum anyway

Practical Advice

- What can go wrong:
  - Bad initialization (more on this later)
  - Bad interaction between data representation and model bias
  - Can learn some salient pattern that is not what you wanted
- What can you do?
  - Get used to disappointment
  - Look at errors!
  - Understand what the model family can (and can't) learn
  - Change data representation
  - Change model structure or estimators
  - …or change objective function [Smith and Eisner, ACL 05]

Semi-Supervised Learning

- A middle ground: semi-supervised methods
  - Use a small labeled training set and a large unlabeled extension set
  - Use labeled data to lock onto the desired patterns
  - Use unlabeled data to flesh out model parameters
- Some approaches
  - Constrained clustering
  - Self-training
  - Adaptation / anchoring
- Also: active learning

Summary

- Clustering
  - Clustering is cool
  - It's easy to find the most salient pattern
  - It's quite hard to find the pattern you want
  - It's hard to know how to fix when broken
  - EM is a useful optimization technique you should understand well if you don't already
- Next time: Part of speech tagging