Learning in Graphical Models

- Problem Dimensions
  - Model
    - Bayes Nets
    - Markov Nets
  - Structure
    - Known
    - Unknown (structure learning)
  - Data
    - Complete
    - **Incomplete (missing values or hidden variables)**

Expectation-Maximization

- Last time:
  - Basics of EM
  - Learning a mixture of Gaussians (k-means)

- This time:
  - Short story justifying EM
    - Slides based on lecture notes from Andrew Ng
  - Applying EM for semi-supervised document classification

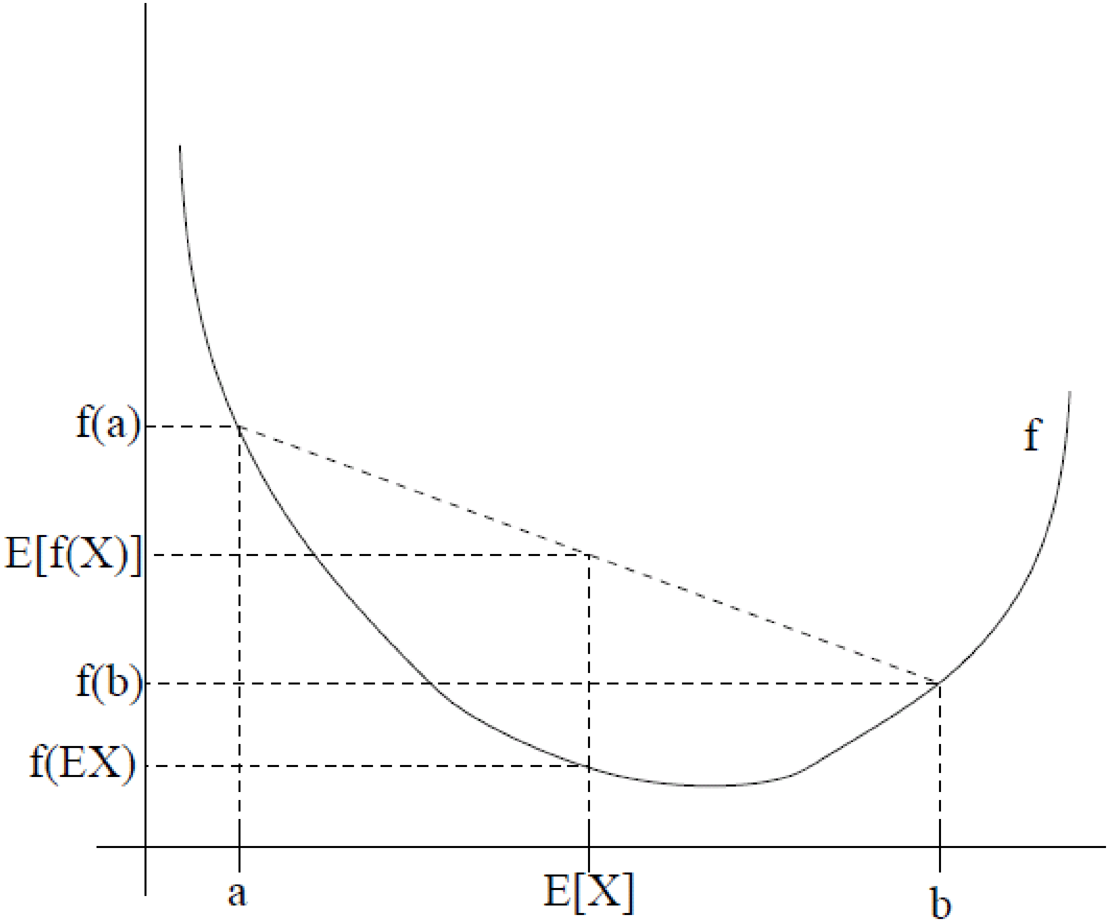
  - Homework #4

10,000 foot level EM

- Guess some parameters, then
  - Use your parameters to get a distribution over hidden variables
  - Re-estimate the parameters as if your distribution over hidden variables is correct
- Seems magical. When/why does this work?

Jensen's Inequality

- For f convex, $\boldsymbol{E}[f(X)] >= f(\boldsymbol{E}[X])$



Maximizing likelihood

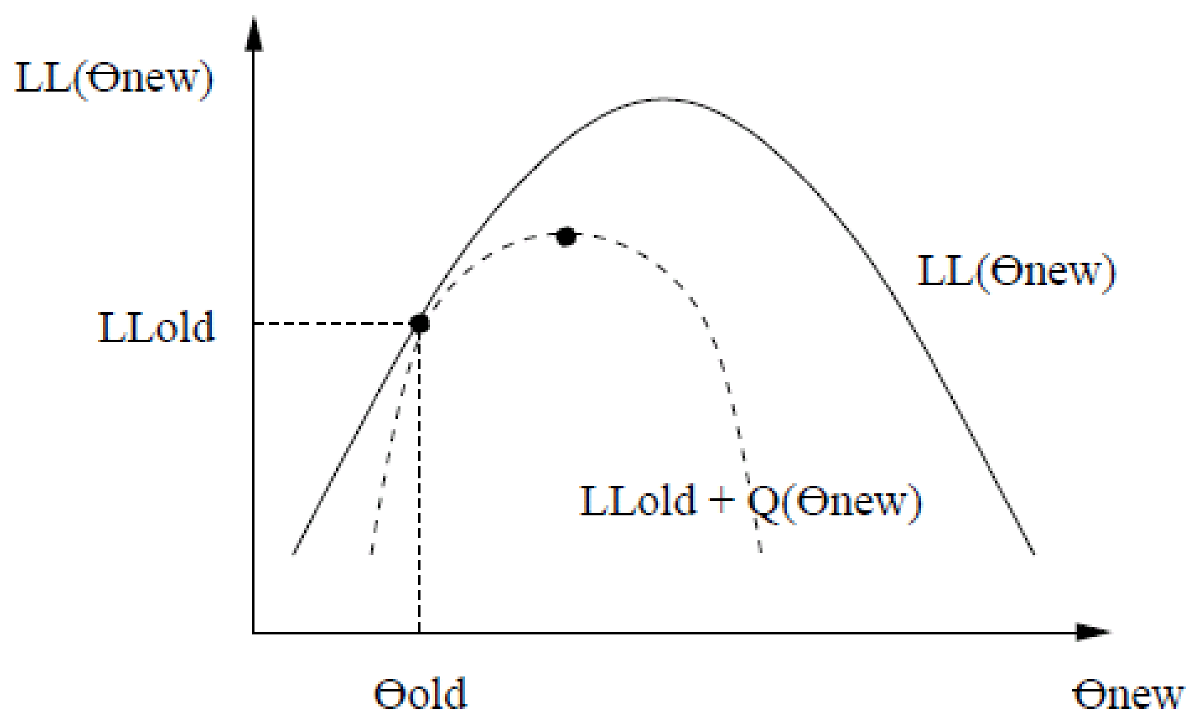
- $x^{(i)}$ = data, $z^{(i)}$ = hidden vars, θ = parameters

$$
\begin{aligned}
\sum_i \log p(x^{(i)};\theta) &= \sum_i \log \sum_{z^{(i)}} p(x^{(i)}, z^{(i)};\theta) \\
&= \sum_i \log \sum_{z^{(i)}} Q_i(z^{(i)}) \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})} \\
&\geq \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})}
\end{aligned}
$$

- This lower bound is easier to maximize, but
  - What is Q?   What good is maximizing a lower bound?

What do we use for *Q*?

- EM: Given a guess θ_{old} for θ, improve it
- Idea: choose *Q* such that our lower bound equals the true log likelihood at θ_{old}:



Ensure the bound is tight at θ_{old}

- When does Jensen's inequality hold exactly?

Ensure the bound is tight at θ_{old}

- When does Jensen's inequality hold exactly?
- Sufficient that

$$\log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

  be constant with respect to $z^{(i)}$

- Thus, choose $Q(z^{(i)}) = p(z^{(i)} \mid x^{(i)} ; \theta_{\text{old}})$

Putting it together

(E-step) For each i, set

$$Q_i(z^{(i)}) := p(z^{(i)}|x^{(i)};\theta).$$

(M-step) Set

$$\theta := \arg\max_\theta \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \theta)}{Q_i(z^{(i)})}$$

For exponential family

- *E* step:
  - Use θ_n to estimate **expected** sufficient statistics over **complete** data
- *M* step
  - Set θ_{n+1} = ML parameters given sufficient statistics
    - (Or MAP parameters)

EM in practice

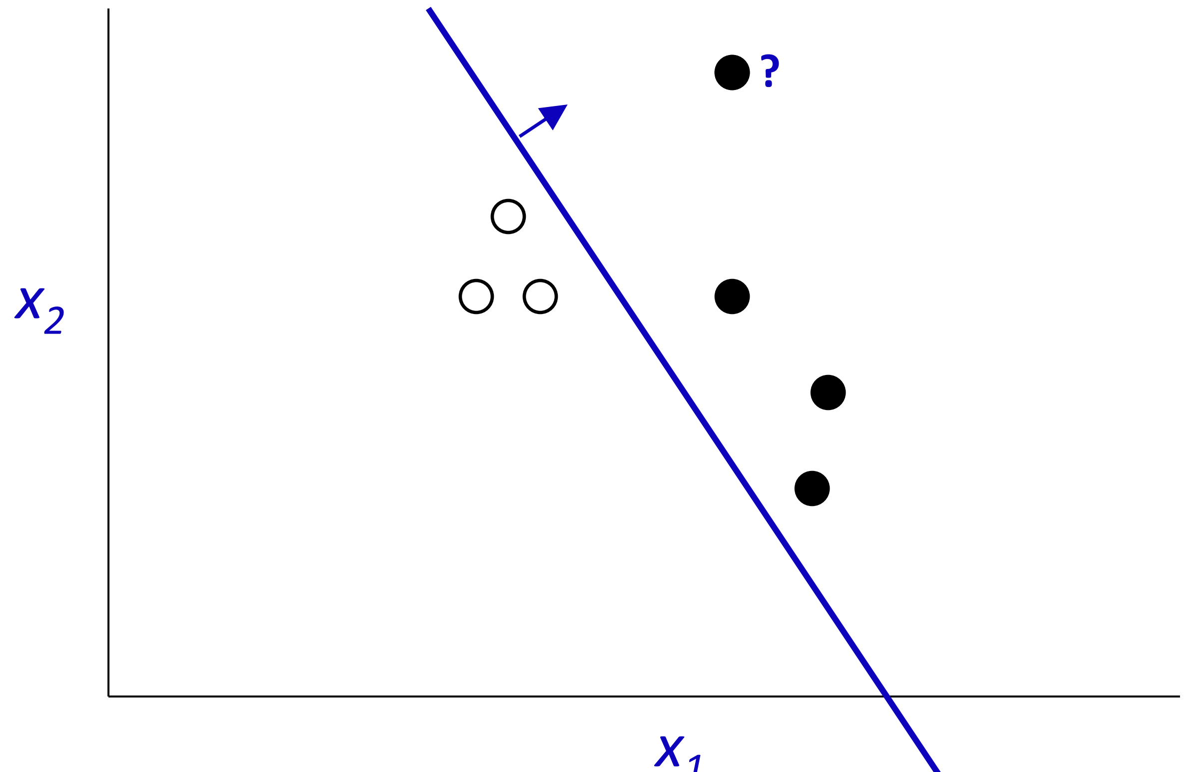
- Local maxima
  - Random re-starts, simulated annealing…
- Variants
  - Generalized EM: increase (not nec. maximize) likelihood in each step
  - Approximate E-step (e.g. sampling)

Semi-supervised Learning

- Unlabeled data abounds in the world
  - Web, measurements, etc.
- *Labeled* data is expensive
  - Image classification, natural language processing, speech recognition, etc. all require large #s of labels
- Idea: use unlabeled data to help with learning

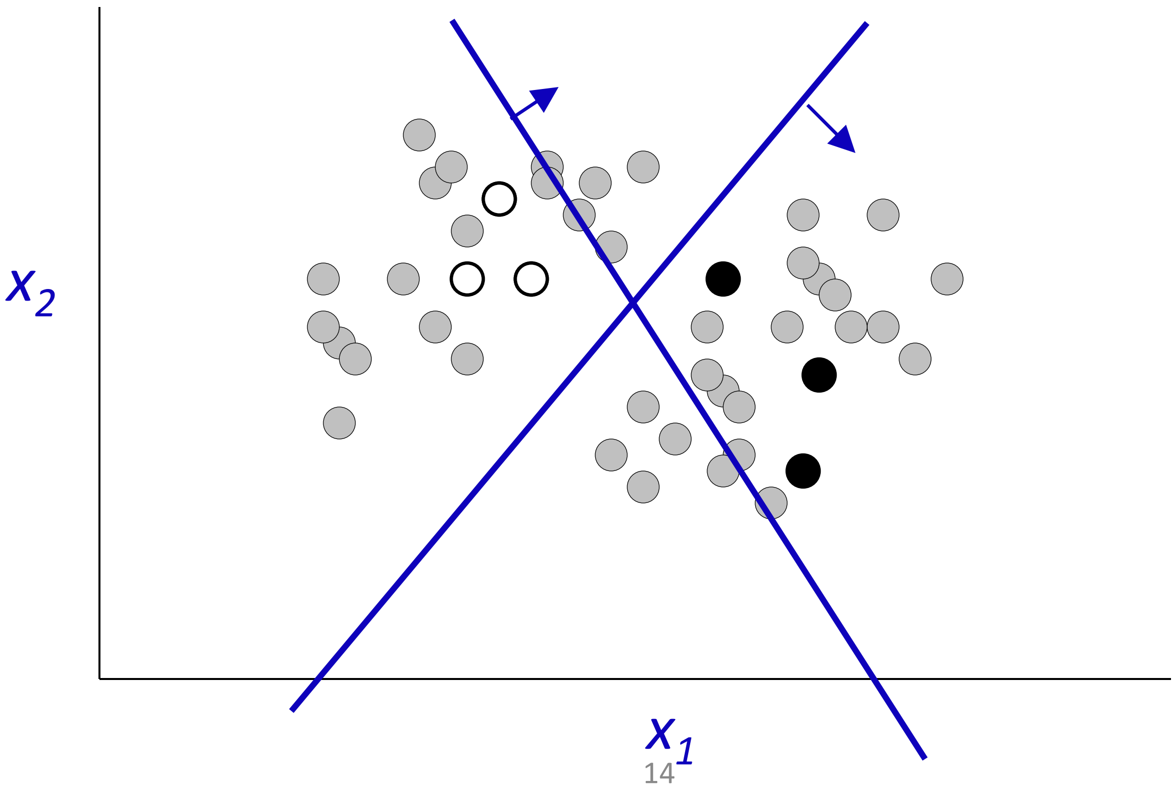
Supervised Learning

Learn function from $\mathbf{x} = (x_1, \ldots, x_d)$ to $y \in \{0, 1\}$
given labeled examples $(\mathbf{x}, y)$



Semi-Supervised Learning (SSL)

Learn function from $\mathbf{x} = (x_1, \ldots, x_d)$ to $y \in \{0, 1\}$
given labeled examples $(\mathbf{x}, y)$
and unlabeled examples $(\mathbf{x})$



SSL in Graphical Models

- Graphical Model describes how data (***x***, y) is generated
- Missing Data: *y*
- So use EM

Example: Document classification with Naïve Bayes

$$\mathrm{P}(x_i|\theta) = \sum_{j \in [M]} \mathrm{P}(c_j|\theta)\mathrm{P}(x_i|c_j;\theta).$$

- x_i = count of word *i* in document
- c_j = document class (sports, politics, etc.)
- x_{it} = count of word *i* in docs of class *t*

$$\mathrm{P}(x_i|\theta) \propto \mathrm{P}(|x_i|) \sum_{j \in [M]} \mathrm{P}(c_j|\theta) \prod_{w_t \in \mathcal{X}} \mathrm{P}(w_t|c_j;\theta)^{x_{it}}$$

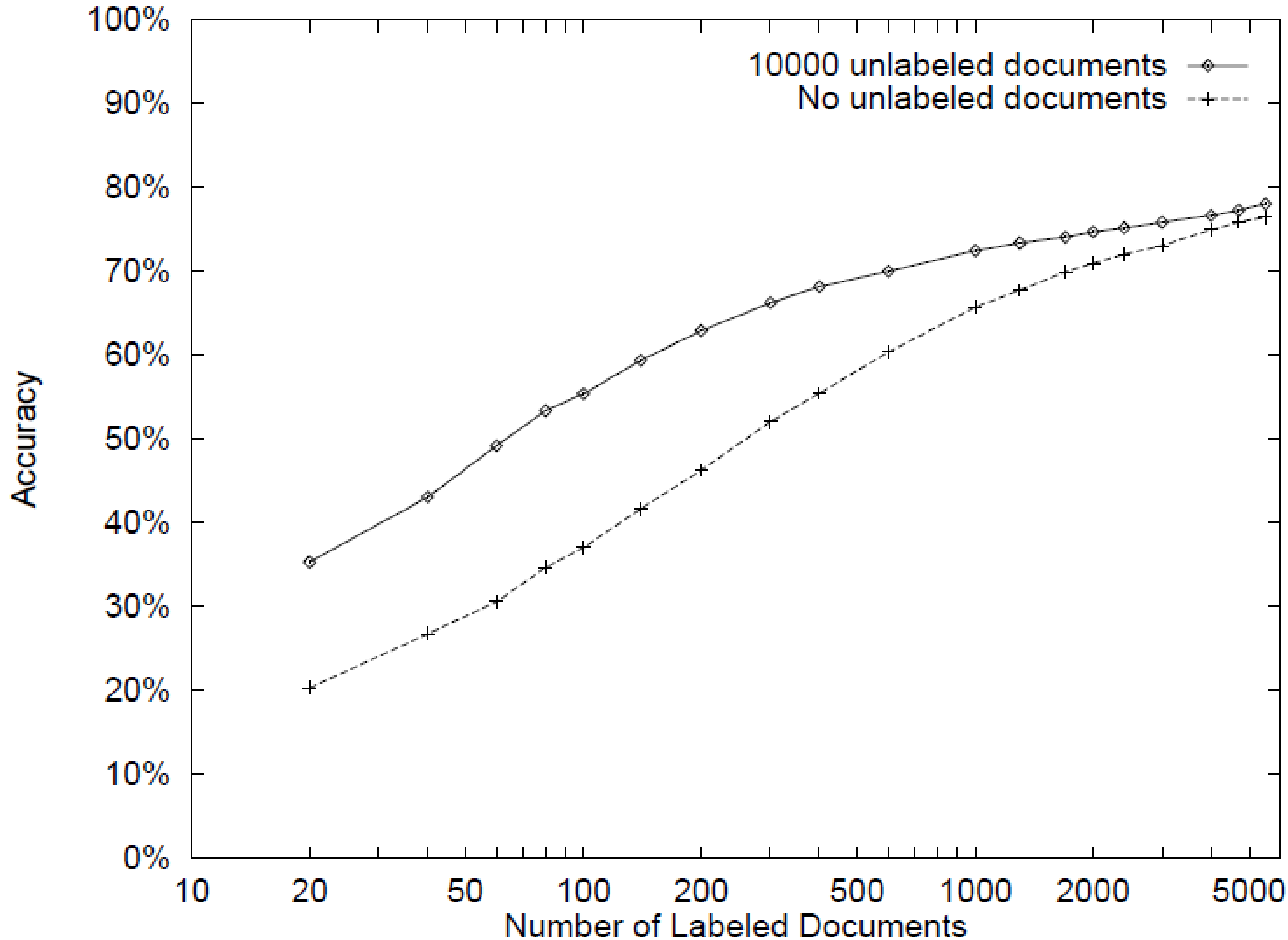
- M classes, $W = |\mathcal{X}|$ words

  (from *Semi-supervised Text Classification Using EM*, Nigam, et al.)

Semi-supervised Training

- Initialize θ ignoring missing data
- E-step:
  - $E[x_{it}]$ = count of word *i* in docs of class *t* in training set
    + E_θ[count of word *i* in docs of class *t* in unlabeled data]
  - $E[\#c_t]$ = count of docs in class t in training
    + E_θ[count of docs of class *t* in unlabeled data]
- M-step:
  - Set θ according to expected statistics above, I.e.:
    - $P_\theta(w_t \mid c_t) = (E[x_{it}] + 1) / (W + \Sigma_i E[x_{it}])$
    - $P_\theta(c_t) = (E[\#c_t] + 1) / (\#\text{words} + M)$

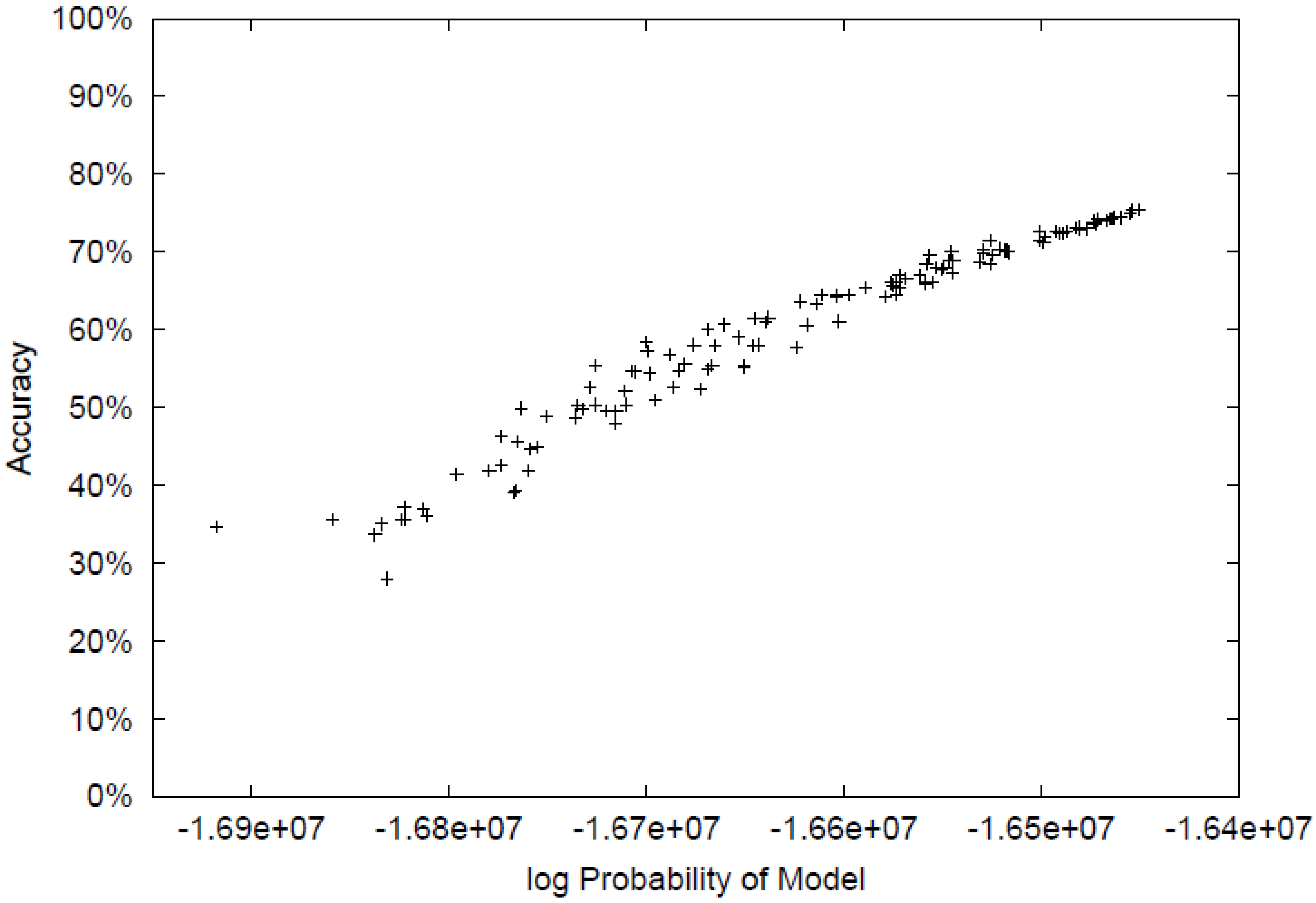
Semi-supervised Learning



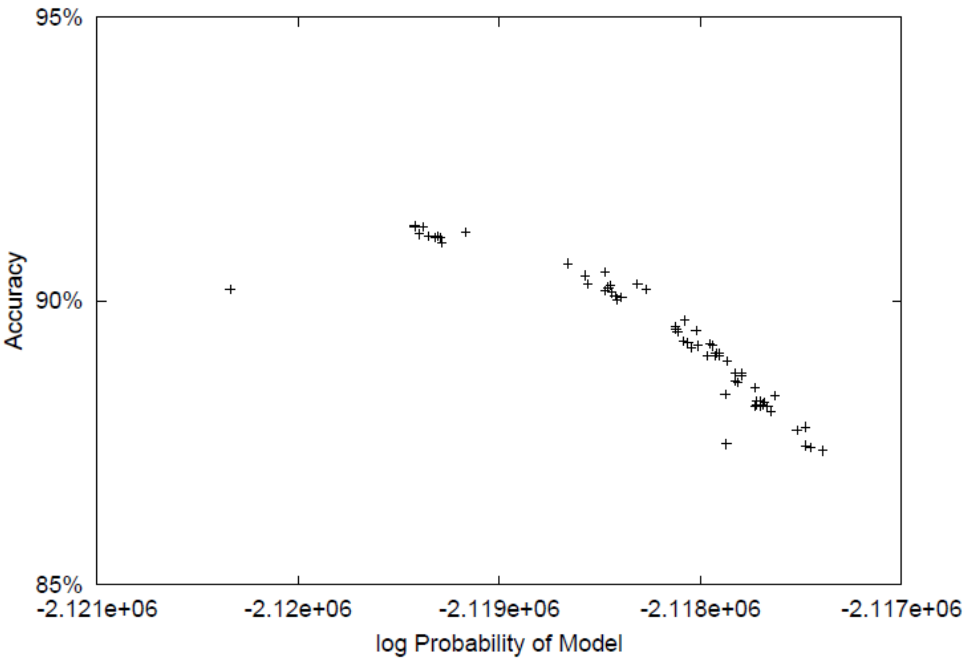
When does semi-supervised learning work?

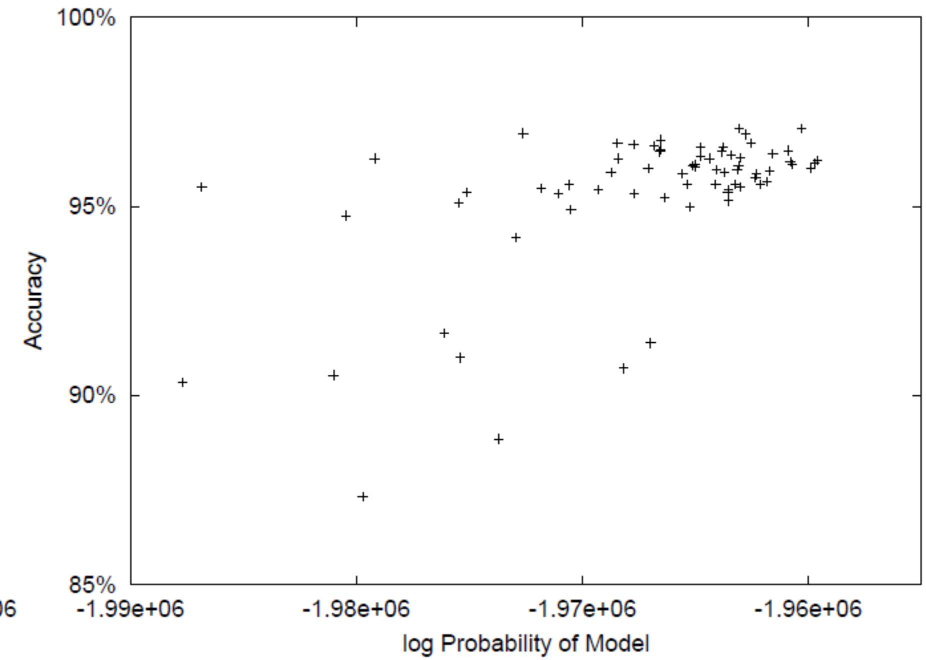
- When a better model of $P(\boldsymbol{x})$ -> a better model of $P(y \mid \boldsymbol{x})$
- Can't use purely *discriminative* models
- Accurate modeling assumptions are key
  - Consider: *negative* class

Good example



Issue: negative class





Negative

- NB*, EM* represent the negative class with the optimal number of model classes (c_i's)

Category	NB1	EM1	NB*	EM*
acq	86.9	81.3	88.0 (4)	**93.1** (10)
corn	94.6	93.2	96.0 (10)	**97.2** (40)
crude	94.3	94.9	95.7 (13)	**96.3** (10)
earn	94.9	95.2	**95.9** (5)	95.7 (10)
grain	94.1	93.6	96.2 (3)	**96.9** (20)
interest	91.8	87.6	95.3 (5)	**95.8** (10)
money-fx	93.0	90.4	94.1 (5)	**95.0** (15)
ship	94.9	94.1	**96.3** (3)	95.9 (3)
trade	91.8	90.2	94.3 (5)	**95.0** (20)
wheat	94.0	94.5	96.2 (4)	**97.8** (40)

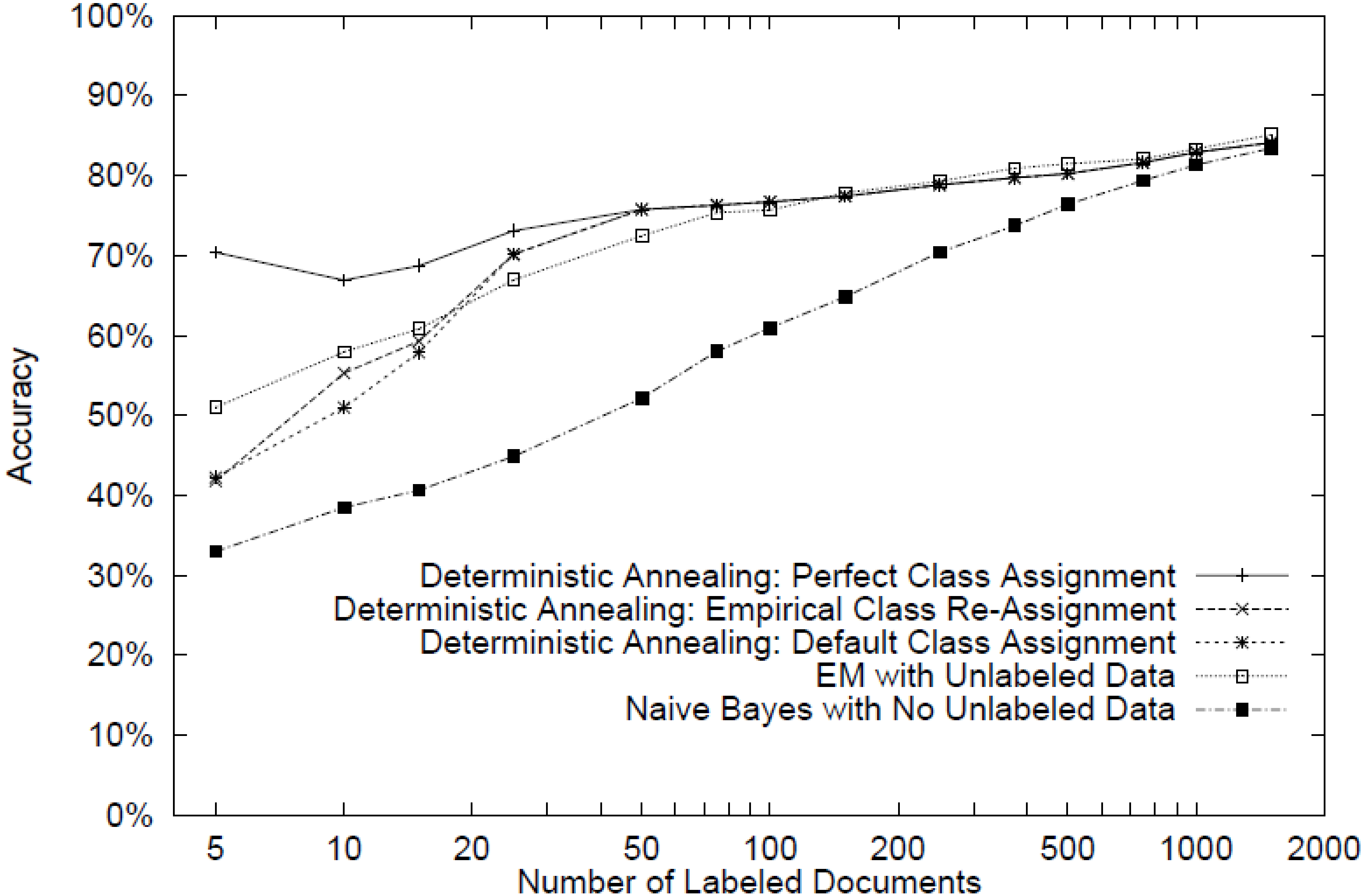
Problem: local maxima

- "Deterministic Annealing"

$$
\begin{aligned}
l(\theta|X,Y) \quad = \quad & \sum_{x_i \in X_u} \log \sum_{c_j \in [M]} [\mathrm{P}(c_j|\theta)\mathrm{P}(x_i|c_j;\theta)]^\beta \\
& + \sum_{x_i \in X_l} \log([\mathrm{P}(y_i = c_j|\theta)\mathrm{P}(x_i|y_i = c_j;\theta)]^\beta)
\end{aligned}
$$

- Slowly increase β
- Results: works, but can end up confusing classes (next slide)

Annealing performance



Homework #4 (1 of 3)

- What if we don’t know the target classes in advance?
- Example: Google Sets
- Wait until query time to run EM? Slow.
- Strategy: Learn a NB model in advance, obtain mapping from examples->”classes”
- Then at “query time” compare examples

Homework #4 (2 of 3)

- Classify noun phrases based on *context* in text
  - E.g. `___ prime minister`   `CEO of ___`
- Model noun phrases (NPs) as P(z | w):

P(z | Canada) =

z=1	2		N
0.14	0.01	…	0.06

- Experiment with different N
- Query time **input**: "seeds" (e.g., Algeria, UK)
  Output: ranked list of other NPs, using KL div.

Homework #4 (3 of 3)

- Code: written in Java
- You write ~5 lines
  - (important ones)
- Run some experiments

- Homework also has a few written exercises
  - Sampling

Road Map

- Basics of Probability and Statistical Estimation
- Bayesian Networks
- Markov Networks
- Inference
- Learning
  - Parameters, **Structure**, EM
- **HMMs**
- Something else?
  - Candidates: Active Learning, Decision Theory, Statistical Relational Models…
    Role of Probabilistic Models in the Financial Crisis?