Classification key concepts

- **Classifiers**: Naive Bayes, perceptron, P-A, logistic regression

Classification key concepts

- **Classifiers**: Naive Bayes, perceptron, P-A, logistic regression
- **Linear decision rule**

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x}, y)$$

Classification key concepts

- **Classifiers**: Naive Bayes, perceptron, P-A, logistic regression
- **Linear decision rule**

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \boldsymbol{\theta}^{\top} \mathbf{f}(\mathbf{x}, y)$$

- **How to set θ?**
  - Criteria: joint likelihood, 0/1 loss, hinge loss, conditional likelihood
  - Optimization: online vs batch
  - Smoothing, regularization, averaging

Classification key concepts

- **Classifiers**: Naive Bayes, perceptron, P-A, logistic regression
- **Linear decision rule**

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \boldsymbol{\theta}^{\top} \mathbf{f}(\mathbf{x}, y)$$

- **How to set θ?**
  - Criteria: joint likelihood, 0/1 loss, hinge loss, conditional likelihood
  - Optimization: online vs batch
  - Smoothing, regularization, averaging
- **What is in $\mathbf{f}(\mathbf{x}, y)$?**
  - Bag-of-words features
  - N-grams, suffixes, prefixes, etc...

Classification key concepts

- **Classifiers**: Naive Bayes, perceptron, P-A, logistic regression
- **Linear decision rule**

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x}, y)$$

- **How to set θ?**
  - Criteria: joint likelihood, 0/1 loss, hinge loss, conditional likelihood
  - Optimization: online vs batch
  - Smoothing, regularization, averaging
- **What is in $\mathbf{f}(\mathbf{x}, y)$?**
  - Bag-of-words features
  - N-grams, suffixes, prefixes, etc...
- **How to find the best y?** What if $\mathcal{Y}$ is too big to search over?

Classifier comparison

- **Naive Bayes**: easy to implement, fast to learn, probabilistic, restrictive independence assumption
- **Perceptron**: easy to implement, pretty fast to learn, not probabilistic, thrashing when not instances are not linearly separable
- **Passive-aggressive**: ibid, better behavior when data is not separable
- **Logistic regression**: harder to implement, can be slower to learn, probabilistic and discriminative, easy to regularize

What if...

- We assume **training data** $\{\mathbf{x}_i, y_i\}$
- What if we don't have the labels?
  Can we learn anything from unlabeled data?
- What if we have just a few labels?
  Can unlabeled data help?

Motivation: WSD

- Semcor has 60 labeled instances of the word concern as a noun.
- A context-based classifier would need thousands of bag-of-words features.

Motivation: WSD

- Semcor has 60 labeled instances of the word concern as a noun.
- A context-based classifier would need thousands of bag-of-words features.
- But suppose we identified two word groups:
  - services, produces, banking, pharmaceutical, energy electronics
  - said, dilemma, over, in, with, had

Motivation

Nigam *et al.*(1999):

> *... after a person read and labeled 1000 articles (from UseNet), a learned classifier achieved a precision of about 50% when making predictions for only the 10% of documents about which it was most confident. Most users of a practical system, however, would not have the patience to label a thousand articles... one would obviously prefer algorithms that can provide accurate classifications after hand-labeling only a few dozen articles.*

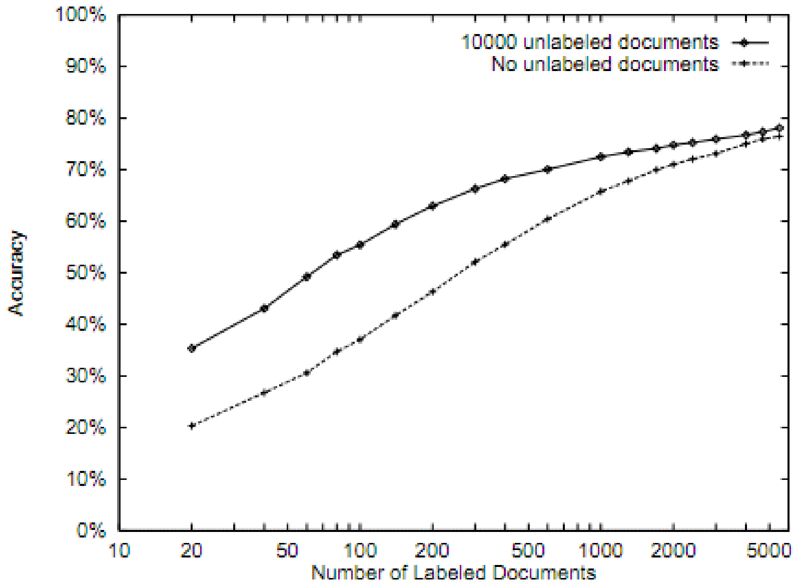
Learning from unlabeled examples?

- Unlabeled data can improve learning by giving a better idea of the underlying shape of the data.

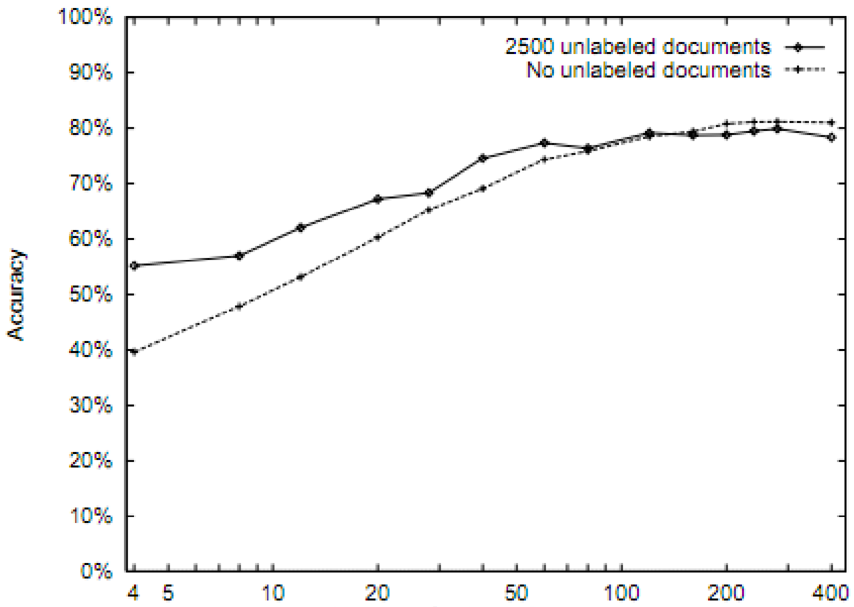
Learning from unlabeled examples?

- Unlabeled data can improve learning by giving a better idea of the underlying shape of the data.
- Nigam *et al.* augment Naive Bayes to include both labeled and unlabeled examples.
    - For the unlabeled examples, they maintain a distribution $q(y_i)$.
    - For the labeled example, y_i is known.
    - The algorithm alternates between updating μ, ϕ and $q(y_i)$ for the unlabeled examples

Accuracy on 20 Newsgroups



Accuracy on WebKB



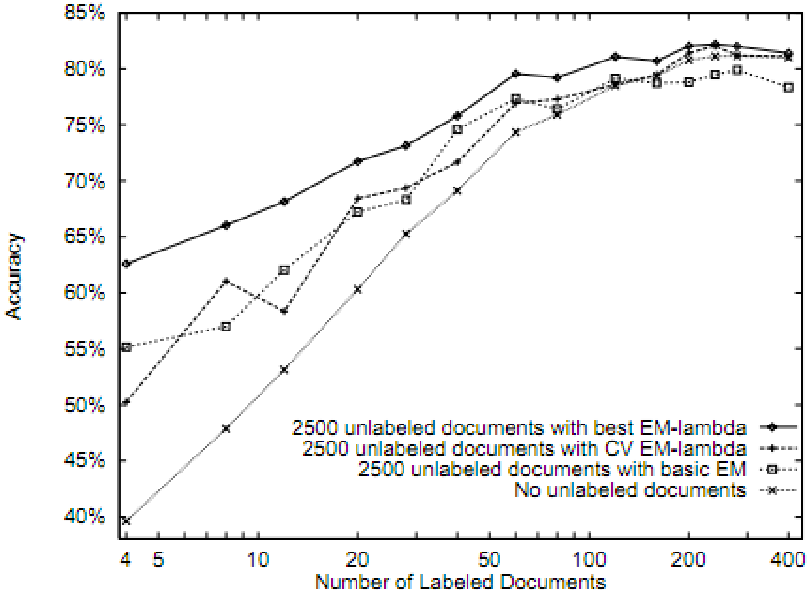
Downweighting unlabeled data

For unlabeled documents, we're just guessing the label.
Maybe they should count less.

$$\log P(\mathbf{x}^{(\ell)}, \mathbf{x}^{(u)}, \mathbf{y}^{(\ell)}) = \log P(\mathbf{x}^{(\ell)}, \mathbf{y}^{(\ell)}) + \lambda \sum_{y} \log P(\mathbf{x}^{(u)}, y) \tag{1}$$

- When $\lambda = 0$, it's supervised classification.
- When $\lambda = 1$, it's standard EM.

Accuracy on WebKB with downweighting



Multiple components per class

Naive Bayes assumes one “component” ϕ per class.

- Suppose we are classifying “baseball” vs “other.”
- There are many ways to **not** write about baseball.
  Why not have many possible components?

Multiple components per class

With EM, we can treat each class as a mixture of components.

- Assume there are k components per class $y \in \mathcal{Y}$
- Assume a distribution $P(c|y)$, where $P(c|y) = 0$ for components c not associated with the class y.

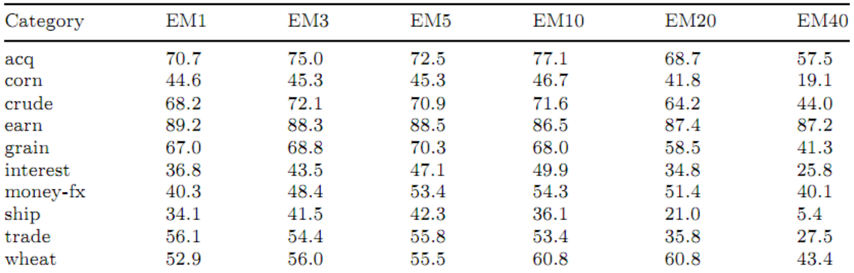
$$\begin{aligned} P(\mathbf{x}_i, y_i) &= \sum_c P(\mathbf{x}_i, y_i, c) \\ &= \sum_c P(\mathbf{x}_i|c)P(c|y_i)P(y_i) \end{aligned}$$

Multiple components per class

The component for each document, c_i, is called a **latent variable**.

- We perform **inference** over latent variables, computing the distribution $q_{C_i}(c) = P(c|\mathbf{x}_i, y_i)$.
- This is part of the E-step in EM.
- The ability to incorporate latent variables is a major advantage of probabilistic models.

Multiple components per class



Category	EM1	EM3	EM5	EM10	EM20	EM40
acq	70.7	75.0	72.5	77.1	68.7	57.5
corn	44.6	45.3	45.3	46.7	41.8	19.1
crude	68.2	72.1	70.9	71.6	64.2	44.0
earn	89.2	88.3	88.5	86.5	87.4	87.2
grain	67.0	68.8	70.3	68.0	58.5	41.3
interest	36.8	43.5	47.1	49.9	34.8	25.8
money-fx	40.3	48.4	53.4	54.3	51.4	40.1
ship	34.1	41.5	42.3	36.1	21.0	5.4
trade	56.1	54.4	55.8	53.4	35.8	27.5
wheat	52.9	56.0	55.5	60.8	60.8	43.4

- This can help, but it is sensitive to choosing the right number of components per class.
- With too many components, we will **overfit** — "memorizing" the training data.

Summary

- As a probabilistic model, Naive Bayes can go beyond just supervised classification.
- Expectation maximization allows us to handle **missing data**
  - In clustering and semi-supervised learning,
    the label y_i is missing.
  - In multi-component modeling, the component c_i is missing.
- There are lots of other ways to do semi-supervised learning.
  We'll talk about them towards the end of the course.