# Statistical NLP
## Spring 2007


## Lecture 4: Text Categorization

Dan Klein – UC Berkeley

# Overview

- So far: language models give P(s)
  - Help model fluency for various noisy-channel processes (MT, ASR, etc.)
  - N-gram models don't represent any deep variables involved in language structure or meaning
  - Usually we want to know something about the input other than how likely it is (syntax, semantics, topic, etc)
- Next: Naïve-Bayes models
  - We introduce a single new global variable
  - Still a very simplistic model family
  - Lets us model hidden properties of text, but only very non-local ones…
  - In particular, we can only model properties which are largely invariant to word order (like topic)

# Text Categorization

- Want to classify documents into broad semantic topics (e.g. politics, sports, etc.)

Democratic vice presidential candidate John Edwards on Sunday accused President Bush and Vice President Dick Cheney of misleading Americans by implying a link between deposed Iraqi President Saddam Hussein and the Sept. 11, 2001 terrorist attacks.

While No. 1 Southern California and No. 2 Oklahoma had no problems holding on to the top two spots with lopsided wins, four teams fell out of the rankings — Kansas State and Missouri from the Big 12 and Clemson from the Atlantic Coast Conference and Oregon from the Pac-10.

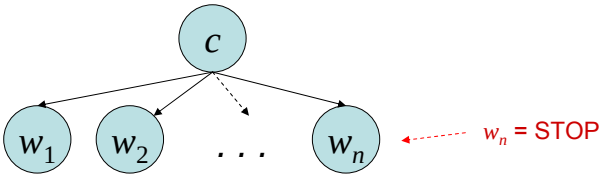
- Which one is the politics document? (And how much deep processing did that decision take?)
- One approach: bag-of-words and Naïve-Bayes models
- Another approach later…
- Usually begin with a labeled corpus containing examples of each class

# Naïve-Bayes Models

- Idea: pick a topic, then generate a document using a language model for that topic.
- Naïve-Bayes assumption: all words are independent given the topic.

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i P(w_i \mid c)$$

*We have to smooth these!*




- Compare to a unigram language model:

$$P(w_1, w_2, \ldots w_n) = \prod_i P(w_i)$$

# Using NB for Classification

- We have a joint model of topics and documents

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i P(w_i \mid c)$$

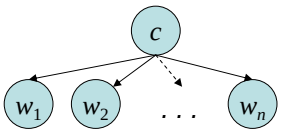
- Gives posterior likelihood of topic given a document

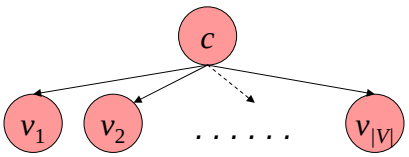
$$P(c \mid w_1, w_2, \ldots w_n) = \frac{P(c)\prod_i P(w_i \mid c)}{\sum_{c'}\left[P(c')\prod_i P(w_i \mid c')\right]}$$

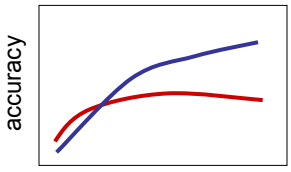
- What about totally unknown words?
- Can work shockingly well for textcat (especially in the wild)
- How can unigram models be so terrible for language modeling, but class-conditional unigram models work for textcat?
- Numerical / speed issues
- How about NB for spam detection?

# Two NB Formulations

- Two NB models for text categorization
  - The class-conditional unigram model, a.k.a. multinomial model
    - One node per word in the document
    - Driven by words which are present
    - Multiple occurrences, multiple evidence
    - Better overall – plus, know how to smooth
  - The binominal (binary) model
    - One node for each word in the vocabulary




    - Incorporates explicit negative correlations
    - Know how to do feature selection (e.g. keep words with high mutual information with the class variable)

# Example: Barometers

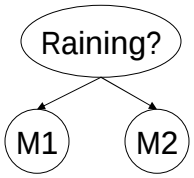
**Reality**

Raining

Sunny

P(+,+,r) = 3/8 P(-,-,r) = 1/8 P(+,+,s) = 1/8 P(-,-,s) = 3/8

**NB Model**




NB FACTORS:

- P(s) = 1/2
- P(+|s) = 1/4
- P(+|r) = 3/4

PREDICTIONS:

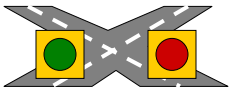
- P(r,+,+) = (½)(¾)(¾)
- P(s,+,+) = (½)(¼)(¼)
- P(r|+,+) = 9/10
- P(s|+,+) = 1/10

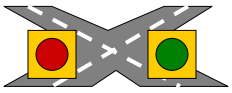
*Overconfidence!*

# Example: Stoplights

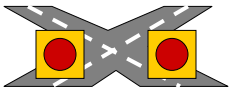
**Reality**

Lights Working



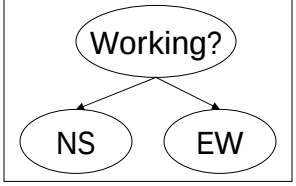


Lights Broken



P(g,r,w) = 3/7 P(r,g,w) = 3/7 P(r,r,b) = 1/7

**NB Model**




NB FACTORS:

- P(w) = 6/7
- P(r|w) = 1/2
- P(g|w) = 1/2
- P(b) = 1/7
- P(r|b) = 1
- P(g|b) = 0

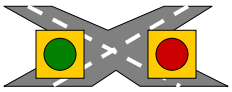
P(b|r,r) = 4/10 (what happened?)

# (Non-)Independence Issues

- Mild Non-Independence
  - Evidence all points in the right direction
  - Observations just not entirely independent
  - Results
    - Inflated Confidence
    - Deflated Priors
  - What to do? Boost priors or attenuate evidence

$$P(c, w_1, w_2, \ldots w_n) \text{ "=" } P(c)^{boost>1} \prod_i P(w_i \mid c)^{boost<1}$$

- Severe Non-Independence
  - Words viewed independently are misleading
  - Interactions have to be modeled
  - What to do?
    - Change your model!



# Language Identification

- How can we tell what language a document is in?

The 38th Parliament will meet on Monday, October 4, 2004, at 11:00 a.m. The first item of business will be the election of the Speaker of the House of Commons. Her Excellency the Governor General will open the First Session of the 38th Parliament on October 5, 2004, with a Speech from the Throne.

La 38e législature se réunira à 11 heures le lundi 4 octobre 2004, et la première affaire à l'ordre du jour sera l'élection du président de la Chambre des communes. Son Excellence la Gouverneure générale ouvrira la première session de la 38e législature avec un discours du Trône le mardi 5 octobre 2004.

- How to tell the French from the English?
  - Treat it as word-level textcat?
    - Overkill, and requires a lot of training data
    - You don't actually need to know about words!

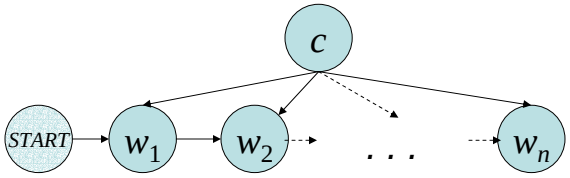
      Σύμφωνο σταθερότητας και ανάπτυξης
      Patto di stabilità e di crescita

  - Option: build a character-level language model

# Class-Conditional LMs

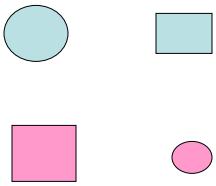
- Can have a topic variable for other language models

$$P(c, w_1, w_2, \ldots w_n) = P(c)\prod_i P(w_i \mid w_{i-1}, c)$$




- Could be characters instead of words, used for language ID (HW2)
- Could sum out the topic variable and use as a language model
- How might a class-conditional n-gram language model behave differently from a standard n-gram model?

# Clustering / Pattern Detection



Soccer team wins match

Stocks close up 3%

Investing in the stock market has …

The first game of the world series …

- Problem 1: There are many patterns in the data, most of which you don't care about.

# Clustering vs. Classification

- Classification: we specify which pattern we want, features uncorrelated with that 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: the clustering procedure locks on to whichever pattern is most salient, statistically
  - P(content words | class) will learn topics
  - P(length, function words | class) will learn style
  - P(characters | class) will learn "language"

# Model-Based Clustering

- Clustering with probabilistic models:

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

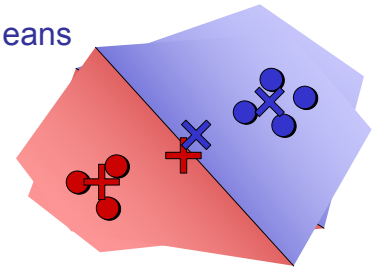
Build a model of the domain: $P(x, y, \theta)$

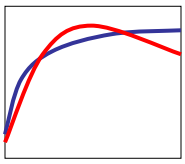
Often: find $\theta$ to maximize: $P(x|\theta) = \sum_y P(x, y|\theta)$

- Problem 2: The relationship between the structure of your model and the kinds of patterns it will detect is complex.

# Learning Models with EM

- Hard EM: alternate between
  - E-step: Find best "completions" Y for fixed $\theta$
  - M-step: Find best parameters $\theta$ for fixed Y
- Example: K-Means





- Problem 3: Data likelihood (usually) isn't the objective you really care about
- Problem 4: You can't find global maxima anyway

# EM for Naïve-Bayes

- First we calculate posteriors (completions):

$$P(y|x) = \frac{P(y)\prod_i P(x_i|y)}{\sum_{y'} P(y'\prod_i P(x_i|y')}$$

- Then we re-estimate parameters P(y), P(x|y) from the (fractionally) labeled data:

$$c(w,y) = \sum_{(x,y)\in D} P(y|x)\,[c(w \in x)]$$

- Can do this when some or none of the docs are labeled

# EM in General

- We'll use EM over and over again to fill in missing data
  - 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)
  - NLP differs from much of machine learning in that we often want to interpret or use the induced variables (which is tricky at best)
- General approach: alternately update y and $\theta$
  - E-step: make a guess at posteriors $P(y|x,\theta)$
    - This means scoring all completions with the current parameters
  - M-step: fit $\theta$ to these completions
    - This is usually the easy part – treat the completions as (fractional) complete data
  - Formally, we are maximizing a lower bound on the observed data likelihood
  - In general, we start with some noisy labelings and the noise adjusts into patterns based on the data and the model
  - We'll see lots of examples in this course
- EM is only locally optimal (why?)

# Heuristic Clustering?

- Many methods of clustering 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
- Examples:
  - Single-link agglomerative clustering
  - Complete-link agglomerative clustering
  - Ward's method
  - Hybrid divisive / agglomerative schemes

# Document Clustering

- Typically want to cluster documents by topic
- Bag-of-words models usually do detect topic
  - It's detecting deeper structure, syntax, etc. where it gets really tricky!
- All kinds of games to focus the clustering
  - Stopword lists
  - Term weighting schemes (from IR, more later)
  - Dimensionality reduction (more later)