INFO 4300 / CS4300 Information Retrieval

slides adapted from Hinrich Schütze's, linked from http://informationretrieval.org/

IR 25/25: Text Classification and Exam Overview

Paul Ginsparg

Cornell University, Ithaca, NY

1 Dec 2011

Administrativa

Assignment 4 due Fri 2 Dec (extended to Sun 4 Dec).

Final examination:

Wed, 14 Dec, from 7:00-9:30 p.m., in Upson B17

Office Hours:

Fri 2 Dec 11–12 (+ Saeed 3:30–4:30),
Wed 7 Dec 1–2,
Fri 9 Dec 1–2,
Wed 14 Dec 1–2

Overview

Outline

Discussion 5

More Statistical Methods:
Peter Norvig, "How to Write a Spelling Corrector"
`http://norvig.com/spell-correct.html`

(Recall also the above video assignment for 25 Oct:
`http://www.youtube.com/watch?v=yvDCzhbjYWs`
"The Unreasonable Effectiveness of Data", given 23 Sep 2010.)

Additional related reference:
`http://doi.ieeecomputersociety.org/10.1109/MIS.2009.36`
A. Halevy, P. Norvig, F. Pereira,
The Unreasonable Effectiveness of Data,
Intelligent Systems Mar/Apr 2009 (copy at readings/unrealdata.pdf)

A little theory

Find the correction c that maximizes the probability of c given the original word w:

$$\operatorname{argmax}_c \; P(c|w)$$

By Bayes' Theorem, equivalent to $\operatorname{argmax}_c \; P(w|c)P(c)/P(w)$. $P(w)$ the same for every possible c, so ignore, and consider:

$$\operatorname{argmax}_c \; P(w|c)P(c) \; .$$

Three parts :

- $P(c)$, the probability that a proposed correction c stands on its own. The language model: "how likely is c to appear in an English text?" ($P($"the"$)$ high, $P($"zxzxzxzyyy"$)$ near zero)
- $P(w|c)$, the probability that w would be typed when author meant c. The error model: "how likely is author to type w by mistake instead of c?"
- argmax_c, the control mechanism: choose c that gives the best combined probability score.

Example

$w =$ "thew"

- two candidate corrections $c =$ "the" and $c =$ "thaw".
- which has higher $P(c|w)$?
- "thaw" has only small change "a" to "e"
- "the" is a very common word, and perhaps the typist's finger slipped off the "e" onto the "w".

To estimate $P(c|w)$, have to consider both the probability of c and the probability of the change from c to w

Complete Spelling Corrector

```
import re, collections

def words(text): return re.findall('[a-z]+', text.lower())

def train(features):
    model = collections.defaultdict(lambda: 1)
    for f in features:
        model[f] += 1
    return model

NWORDS = train(words(file('big.txt').read()))

alphabet = 'abcdefghijklmnopqrstuvwxyz'
```

```
def edits1(word):
    s = [(word[:i], word[i:]) for i in range(len(word) + 1)]
    deletes = [a + b[1:] for a, b in s if b]
    transposes = [a + b[1] + b[0] + b[2:] for a, b in s if len(b)>1]
    replaces = [a + c + b[1:] for a, b in s for c in alphabet if b]
    inserts = [a + c + b for a, b in s for c in alphabet]
    return set(deletes + transposes + replaces + inserts)

def known_edits2(word):
    return set(e2 for e1 in edits1(word) for e2 in edits1(e1) if e2 in NWORDS)

def known(words): return set(w for w in words if w in NWORDS)

def correct(word):
    candidates = known([word]) or known(edits1(word))
                                or known_edits2(word) or [word]
    return max(candidates, key=NWORDS.get)
```

(For word of length n: n deletions, n-1 transpositions, 26n alterations, and 26(n+1) insertions, for a total of 54n+25 at edit distance 1)

Improvements

language model $P(c)$: need more words. add -ed to verb or -s to noun, -ly for adverbs

bad probabilities: wrong word appears more frequently?(didn't happen)

error model $P(w|c)$: sometimes edit distance 2 is better
('adres' to 'address', not 'acres')
or wrong word of many at edit distance 1
(in addition better error model permits adding more obscure words)
allow edit distance 3?

best improvement:
look for context ('they where going', 'There's no there thear')
$\Rightarrow$ Use n-grams
(See Whitelaw et al. (2009), "Using the Web for Language Independent Spellchecking and Autocorrection": Precision, recall, F1, classification accuracy)

Outline

More Data

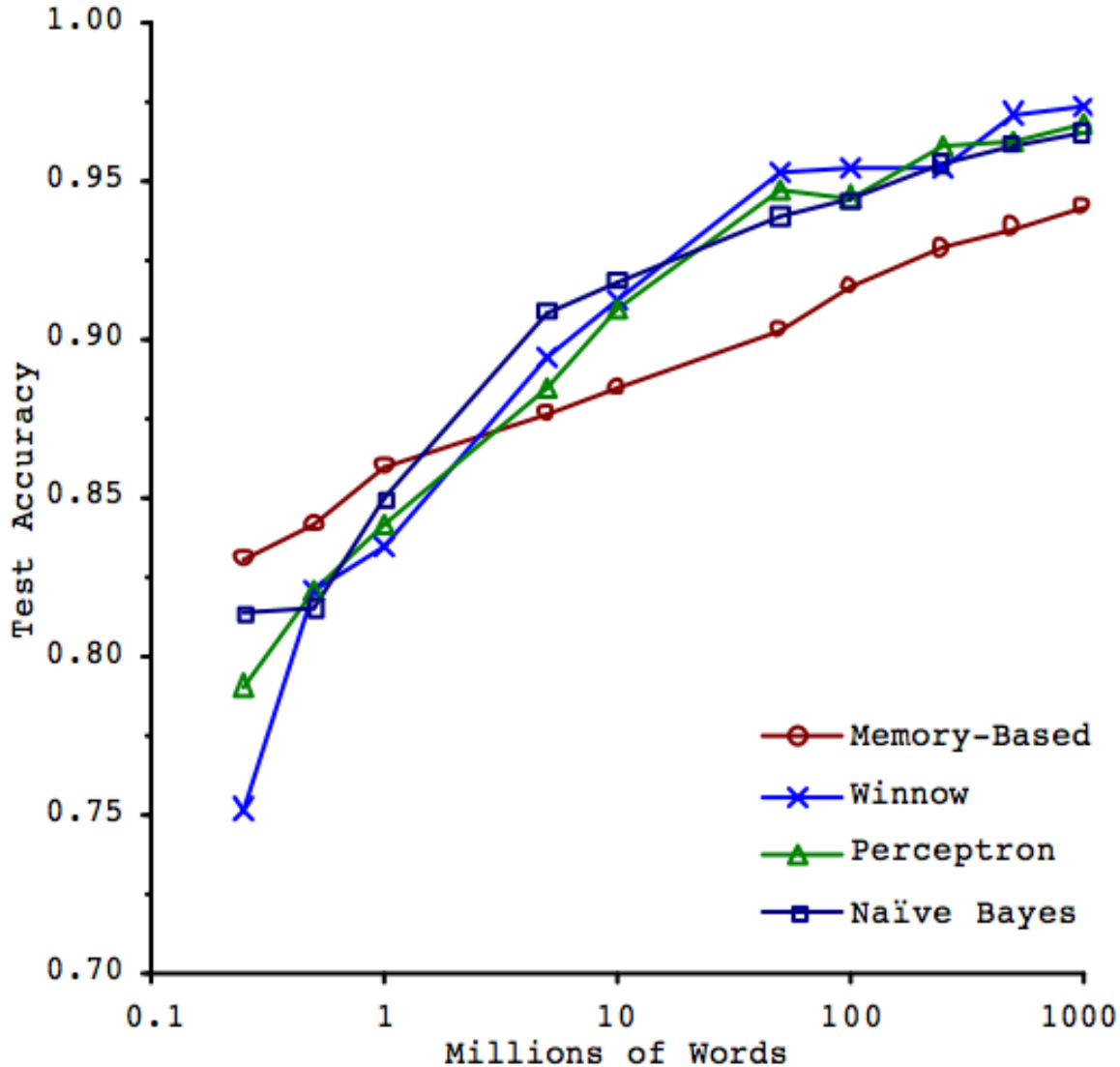


Figure 1. Learning Curves for Confusion Set Disambiguation

http://acl.ldc.upenn.edu/P/P01/P01-1005.pdf

Scaling to Very Very Large Corpora for Natural Language Disambiguation

M. Banko and E. Brill (2001)

More Data for this Task

http://acl.ldc.upenn.edu/P/P01/P01-1005.pdf
Scaling to Very Very Large Corpora for Natural Language Disambiguation
M. Banko and E. Brill (2001)

The amount of readily available on-line text has reached hundreds of billions of words and continues to grow. Yet for most core natural language tasks, algorithms continue to be optimized, tested and compared after training on corpora consisting of only one million words or less. In this paper, we evaluate the performance of different learning methods on a prototypical natural language disambiguation task, confusion set disambiguation, when trained on orders of magnitude more labeled data than has previously been used. We are fortunate that for this particular application, correctly labeled training data is free. Since this will often not be the case, we examine methods for effectively exploiting very large corpora when labeled data comes at a cost.

(Confusion set disambiguation is the problem of choosing the correct use of a word, given a set of words with which it is commonly confused. Example confusion sets include: {principle , principal}, {then , than}, {to , two , too} , and {weather,whether}.)

Segmentation

- nowisthetimeforallgoodmentocometothe
- Probability of a segmentation = P(first word) $\times$ P(rest)
- Best segmentation = one with highest probability
- P(word) = estimated by counting

Trained on 1.7B words English, 98% word accuracy

Spelling with Statistical Learning

- Probability of a spelling correction, c = P(c as a word) × P(original is a typo for c)
- Best correction = one with highest probability
- P(c as a word) = estimated by counting
- P(original is a typo for c) = proportional to number of changes

Similarly for speech recognition, using language model $p(c)$ and acoustic model $p(s|c)$
(Russel & Norvig, "Artificial Intelligence", section 24.7)

Google Sets

Given "lion, tiger, bear" find:

bear, tiger, lion, elephant, monkey, giraffe, dog, cat, snake, horse, zebra, rabbit, wolf, dolphin, dragon, pig, frog, duck, cheetah, bird, cow, cotton, hippo, turtle, penguin, rat, gorilla, leopard, sheep, mouse, puppy, ox, rooster, fish, lamb, panda, wood, musical, toddler, fox, goat, deer, squirrel, koala, crocodile, hamster

(using co-occurrence in pages)

And others

- Statistical Machine Translation
  - Collect parallel texts ("Rosetta stones"), Align (Brants, Popat, Xu, Och, Dean (2007), "Large Language Models in Machine Translation")
- Canonical image selection from the web (Y. Jing, S. Baluja, H. Rowley, 2007)
- Learning people annotation from the web via consistency learning (J. Yagnik, A. Islam, 2007)
  (results on learning from a very large dataset of 37 million images resulting in a validation accuracy of 92.68%)
- fill in occluded portions of photos (Hayes and Efros, 2007)

Outline

To avoid zeros: Add-one smoothing

- Add one to each count to avoid zeros:

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

- B is the number of different words (in this case the size of the vocabulary: $|V| = M$)

Example: Parameter estimates

Priors: $\hat{P}(c) = 3/4$ and $\hat{P}(\overline{c}) = 1/4$
Conditional probabilities:

$$\begin{aligned}
\hat{P}(\textsc{Chinese}|c) &= (5+1)/(8+6) = 6/14 = 3/7 \\
\hat{P}(\textsc{Tokyo}|c) = \hat{P}(\textsc{Japan}|c) &= (0+1)/(8+6) = 1/14 \\
\hat{P}(\textsc{Chinese}|\overline{c}) &= \\
\hat{P}(\textsc{Tokyo}|\overline{c}) = \hat{P}(\textsc{Japan}|\overline{c}) &= (1+1)/(3+6) = 2/9
\end{aligned}$$

The denominators are $(8+6)$ and $(3+6)$ because the lengths of $text_c$ and $text_{\overline{c}}$ are 8 and 3, respectively, and because the constant B is 6 since the vocabulary consists of six terms.

Exercise: verify that $\hat{P}(\textsc{Chinese}|c) + \hat{P}(\textsc{Beijing}|c) + \hat{P}(\textsc{Shanghai}|c) + \hat{P}(\textsc{Macao}|c) + \hat{P}(\textsc{Tokyo}|c) + \hat{P}(\textsc{Japan}|c) = 1$
and $\hat{P}(\textsc{Chinese}|\overline{c}) + \hat{P}(\textsc{Beijing}|\overline{c}) + \hat{P}(\textsc{Shanghai}|\overline{c}) + \hat{P}(\textsc{Macao}|\overline{c}) + \hat{P}(\textsc{Tokyo}|\overline{c}) + \hat{P}(\textsc{Japan}|\overline{c}) = 1$

Naive Bayes: Analysis

(See also D. Lewis (1998) "Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval")

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .
- . . . and state the assumptions we make in that derivation explicitly.

Derivation of Naive Bayes rule

We want to find the class that is most likely given the document:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} P(c|d)$$

Apply Bayes rule $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \frac{P(d|c)P(c)}{P(d)}$$

Drop denominator since $P(d)$ is the same for all classes:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} P(d|c)P(c)$$

Too many parameters / sparseness

$$
\begin{aligned}
c_{\text{map}} &= \arg\max_{c \in \mathbb{C}} P(d|c)P(c) \\
&= \arg\max_{c \in \mathbb{C}} P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)P(c)
\end{aligned}
$$

- There are too many parameters $P(\langle t_1, \ldots, t_k, \ldots, t_{n_d} \rangle | c)$, one for each unique combination of a class and a sequence of words.
- We would need a very, very large number of training examples to estimate that many parameters.
- This is the problem of data sparseness.

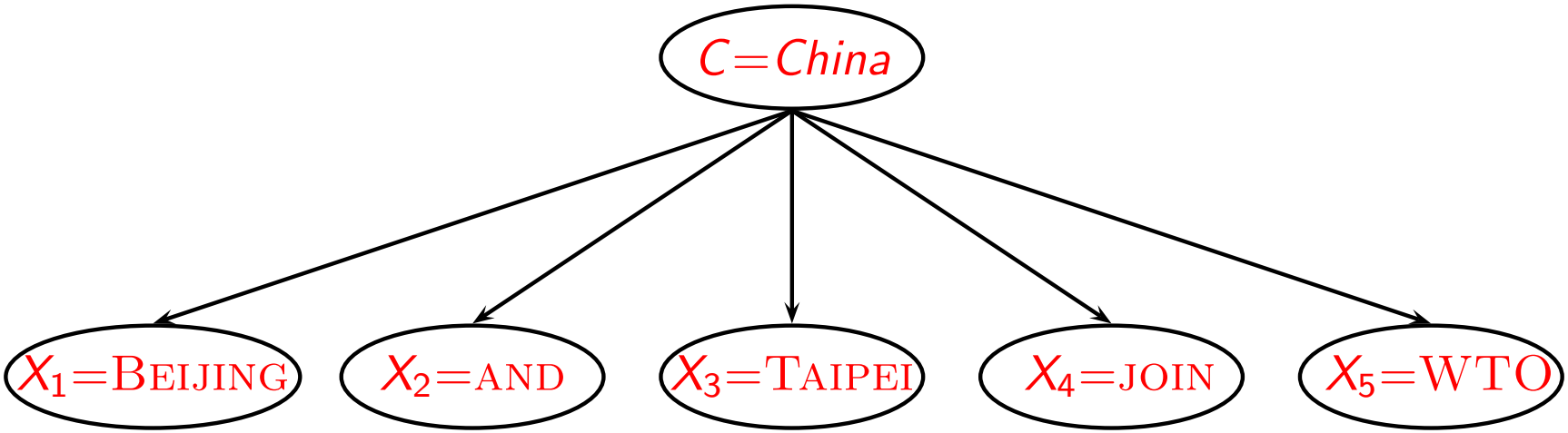
Naive Bayes conditional independence assumption

To reduce the number of parameters to a manageable size, we make the Naive Bayes conditional independence assumption:

$$P(d|c) = P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

We assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(X_k = t_k|c)$.
Recall from earlier the estimates for these priors and conditional probabilities: $\hat{P}(c) = \frac{N_c}{N}$ and $\hat{P}(t|c) = \frac{T_{ct}+1}{(\sum_{t' \in V} T_{ct'})+B}$

Generative model



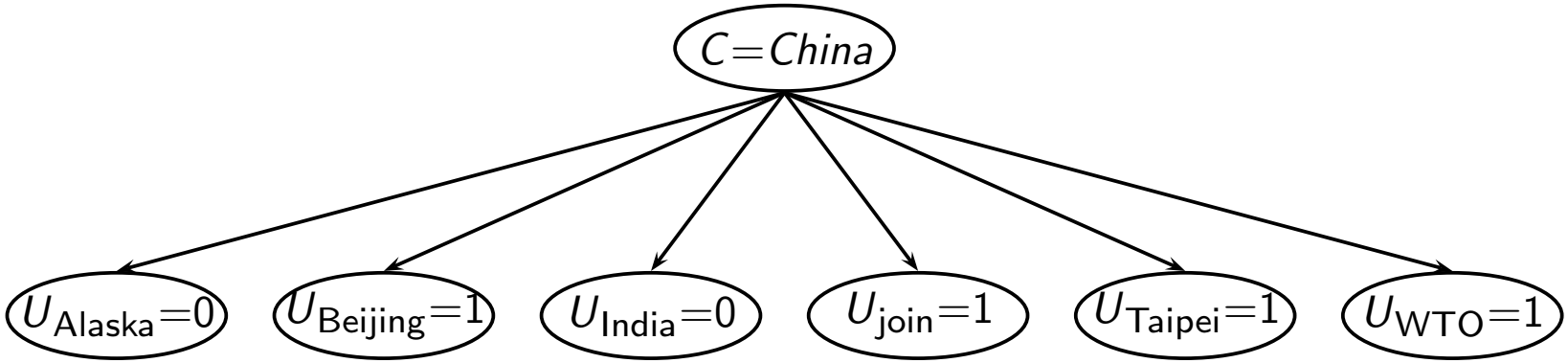
$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$

- Generate a class with probability $P(c)$
- Generate each of the words (in their respective positions), conditional on the class, but independent of each other, with probability $P(t_k|c)$
- To classify docs, we "reengineer" this process and find the class that is most likely to have generated the doc.

Second independence assumption

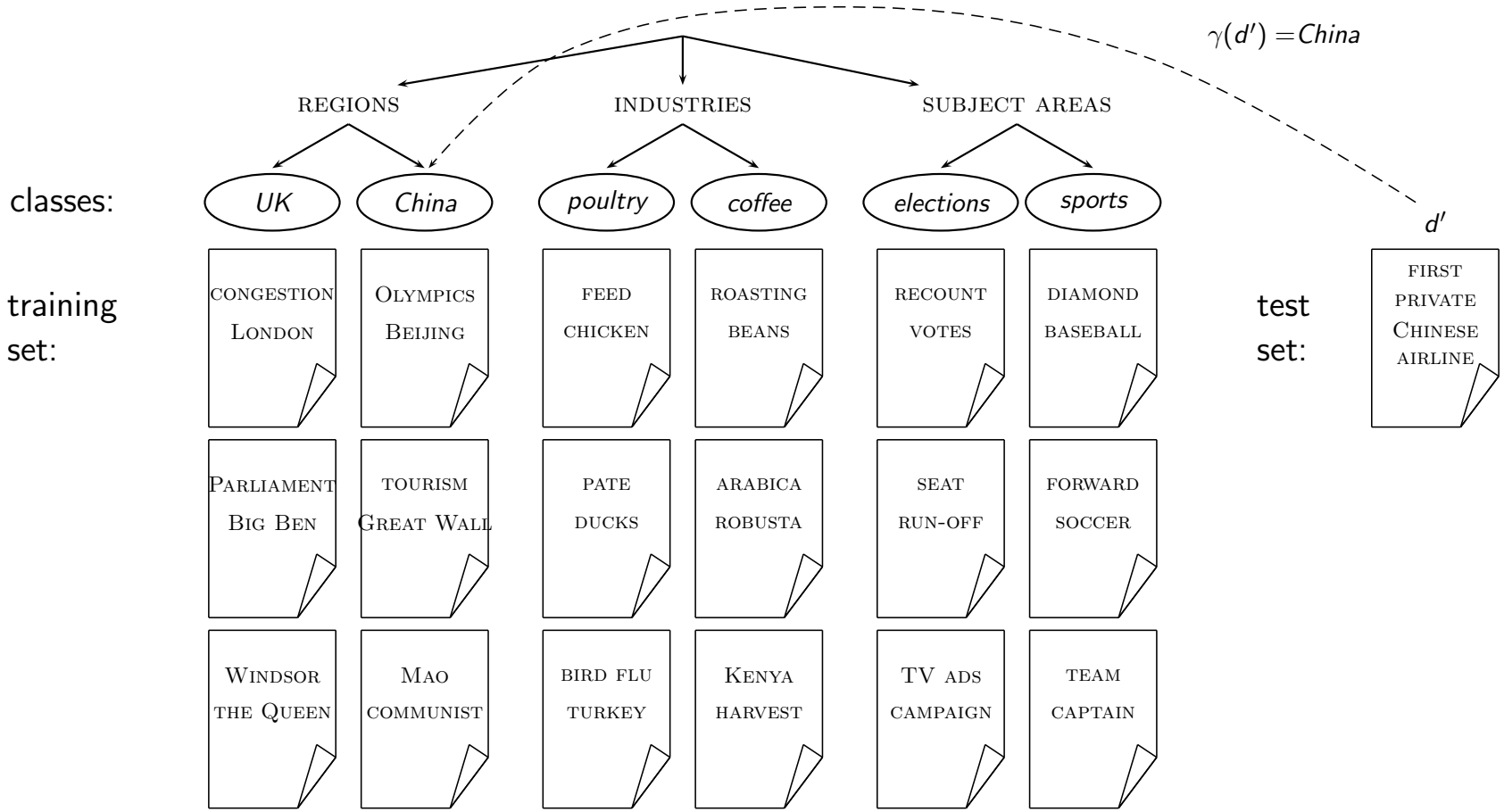
- $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- For example, for a document in the class *UK*, the probability of generating QUEEN in the first position of the document is the same as generating it in the last position.
- The two independence assumptions amount to the bag of words model.

A different Naive Bayes model: Bernoulli model



Outline

Evaluation on Reuters



Example: The Reuters collection

symbol	statistic	value
N	documents	800,000
L	avg. # word tokens per document	200
M	word types	400,000
	avg. # bytes per word token (incl. spaces/punct.)	6
	avg. # bytes per word token (without spaces/punct.)	4.5
	avg. # bytes per word type	7.5
	non-positional postings	100,000,000

type of class	number	examples
region	366	UK, China
industry	870	poultry, coffee
subject area	126	elections, sports

Evaluating classification

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- It's easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
- Measures: Precision, recall, F_1, classification accuracy

Naive Bayes vs. other methods

(a)

	NB	Rocchio	kNN	SVM
micro-avg-L (90 classes)	80	85	86	89
macro-avg (90 classes)	47	59	60	60

(b)

	NB	Rocchio	kNN	trees	SVM
earn	96	93	97	98	98
acq	88	65	92	90	94
money-fx	57	47	78	66	75
grain	79	68	82	85	95
crude	80	70	86	85	89
trade	64	65	77	73	76
interest	65	63	74	67	78
ship	85	49	79	74	86
wheat	70	69	77	93	92
corn	65	48	78	92	90
micro-avg (top 10)	82	65	82	88	92
micro-avg-D (118 classes)	75	62	n/a	n/a	87

Evaluation measure: F_1 Naive Bayes does pretty well, but some methods beat it consistently (e.g., SVM).

See Section 13.6

Outline

Violation of Naive Bayes independence assumptions

- The independence assumptions do not really hold of documents written in natural language.
- Conditional independence:

$$P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- Positional independence: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$
- Exercise
  - Examples for why conditional independence assumption is not really true?
  - Examples for why positional independence assumption is not really true?
- How can Naive Bayes work if it makes such inappropriate assumptions?

Why does Naive Bayes work?

- Naive Bayes can work well even though conditional independence assumptions are badly violated.
- Example:

	c_1	c_2	class selected
true probability $P(c\|d)$	0.6	0.4	c_1
$\hat{P}(c)\prod_{1\leq k\leq n_d}\hat{P}(t_k\|c)$	0.00099	0.00001	
NB estimate $\hat{P}(c\|d)$	0.99	0.01	c_1

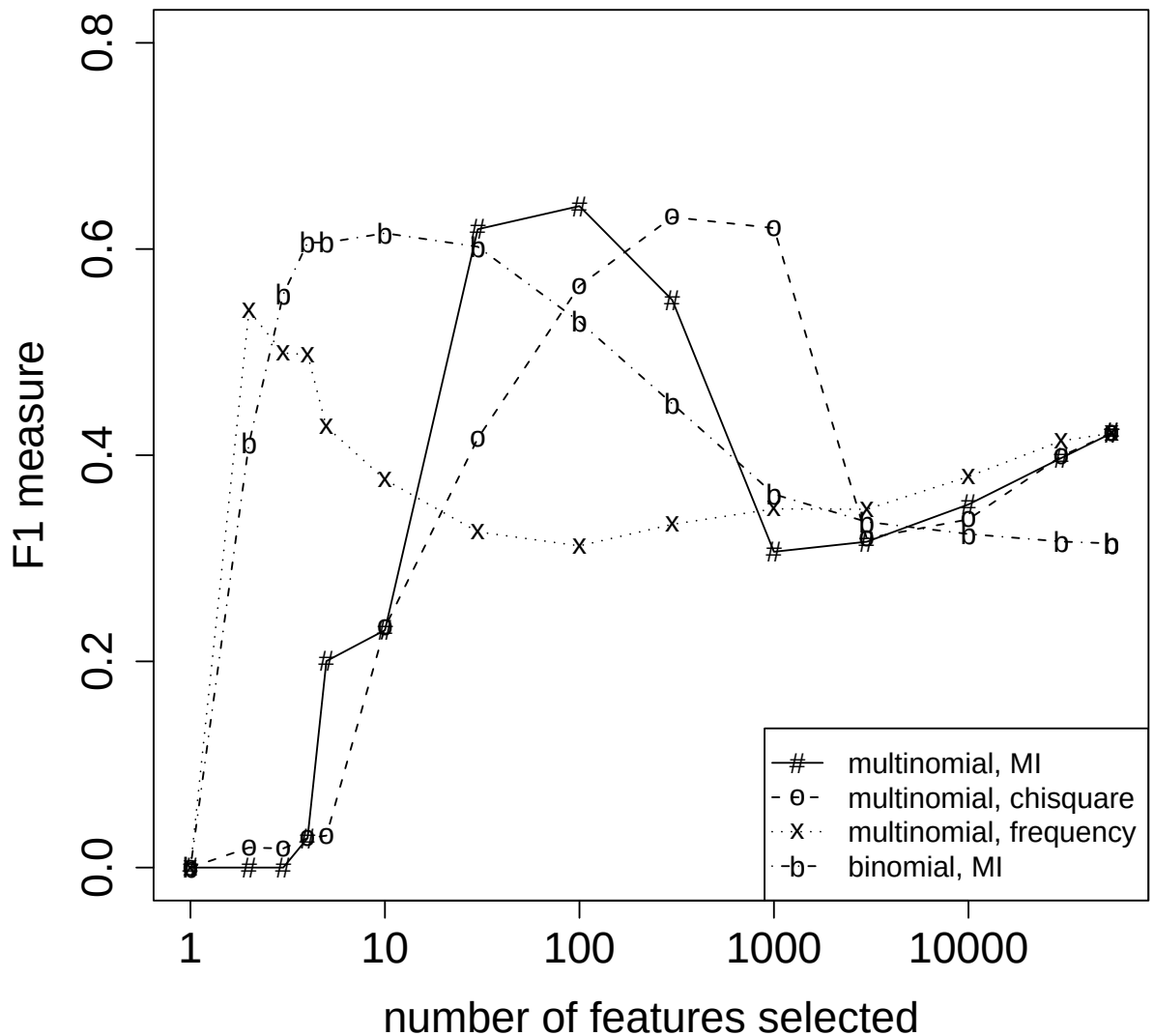
- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Correct estimation $\Rightarrow$ accurate prediction.
- But not vice versa!

Naive Bayes is not so naive

- Naive Bayes has won some bakeoffs (e.g., KDD-CUP 97)
- More robust to nonrelevant features than some more complex learning methods
- More robust to concept drift (changing of definition of class over time) than some more complex learning methods
- Better than methods like decision trees when we have many equally important features
- A good dependable baseline for text classification (but not the best)
- Optimal if independence assumptions hold (never true for text, but true for some domains)
- Very fast
- Low storage requirements

Naive Bayes: Effect of feature selection

Improves performance of text classifiers



(multinomial = multinomial Naive Bayes)

Feature selection for Naive Bayes

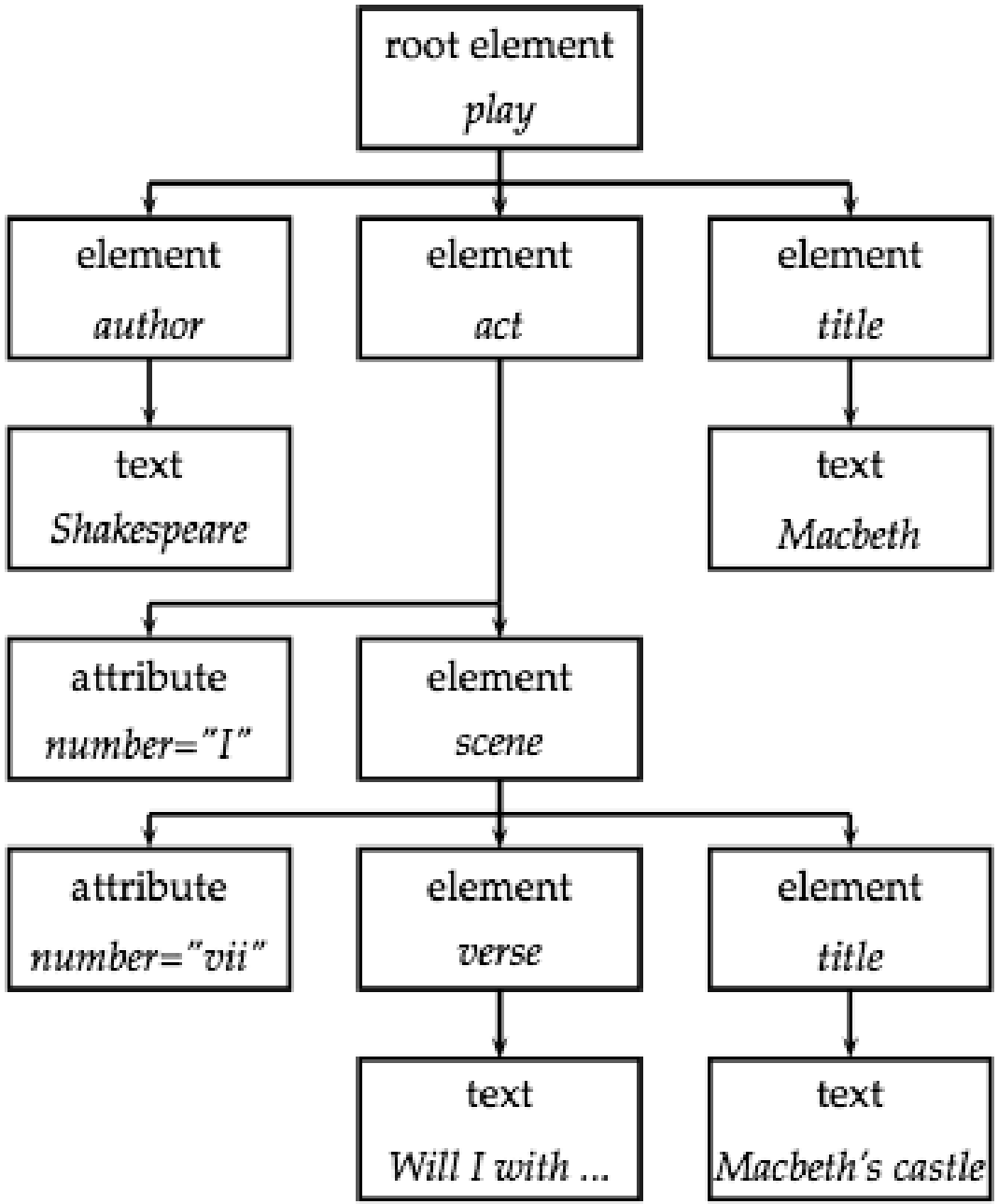
- In general, feature selection is necessary for Naive Bayes to get decent performance.
- Also true for most other learning methods in text classification: you need feature selection for optimal performance.

Outline

XML markup

⟨play⟩
⟨author⟩Shakespeare⟨/author⟩
⟨title⟩Macbeth⟨/title⟩ ⟨act number="I"⟩
⟨scene number="vii"⟩
⟨title⟩Macbeths castle⟨/title⟩
⟨verse⟩Will I with wine and wassail ...⟨/verse⟩
⟨/scene⟩
⟨/act⟩
⟨/play⟩

XML Doc as DOM object



Outline

Definition of *information retrieval* (from Lecture 1)

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

Three scales (web, enterprise/inst/domain, personal)

“Plan” (from Lecture 1)

- Search full text: basic concepts
- Web search
- Probabalistic Retrieval
- Interfaces
- Metadata / Semantics

IR $\Leftrightarrow$ NLP $\Leftrightarrow$ ML

Prereqs: Introductory courses in data structures and algorithms, in linear algebra and in probability theory

1st Half

Searching full text: dictionaries, inverted files, postings, implementation and algorithms, term weighting, Vector Space Model, similarity, ranking

Word Statistics

MRS: 1 Boolean retrieval
MRS: 2 The term vocabulary and postings lists
MRS: 5 Index compression
MRS: 6 Scoring, term weighting, and the vector space model
MRS: 7 Computing scores in a complete search system

1st Half, cont'd

Evaluation of retrieval effectiveness
MRS: 8. Evaluation in information retrieval

Latent semantic indexing
MRS: 18. Matrix decompositions and latent semantic indexing

Discussion 2 IDF
Discussion 3 Latent semantic indexing

Midterm

- 1) term-document matrix, VSM, tf.idf
- 2) Recall/Precision
- 3) LSI
- 4) Word statistics (Heap, Zipf)

2nd Half

MRS: 9 Relevance feedback and query expansion
MRS: 11 Probabilistic information retrieval

Web Search: anchor text and links, Citation and Link Analysis, Web crawling

MRS: 19 Web search basics
MRS: 21 Link analysis

2nd Half, cont'd

Classification, categorization, clustering

MRS: 13 Text classification and Naive Bayes
MRS: 14 Vector space classification
MRS: 16 Flat clustering
MRS: 17 Hierarchical clustering

(Structured Retrieval MRS: 10 XML Retrieval)

Discussion 4 Google
Discussion 5 Statistical Spell Correction

Final Exam: these topics, probably 4 questions

- issues in personal/enterprise/webscale searching, recall/precision, and how related to info/nav/trans needs
- issues for modern search engines... (e.g., w.r.t. web scale, tf.idf? recall/precision?)
- web indexing and retrieval: link analysis, PageRank
- clustering: flat, hierarchical (k-means, agglomerative, similarity dendrograms): evaluation of clustering, measures of cluster similarity (single/complete link, average, group average)
- cluster labeling, feature selection
- recommender systems, adversarial IR
- types of text classification (curated, rule-based, statistical), e.g., naive Bayes
- Vector space classification (rocchio, kNN)

Wed, 14 Dec, from 7:00-9:30 p.m., in Upson B17