Introduction to

Information Retrieval

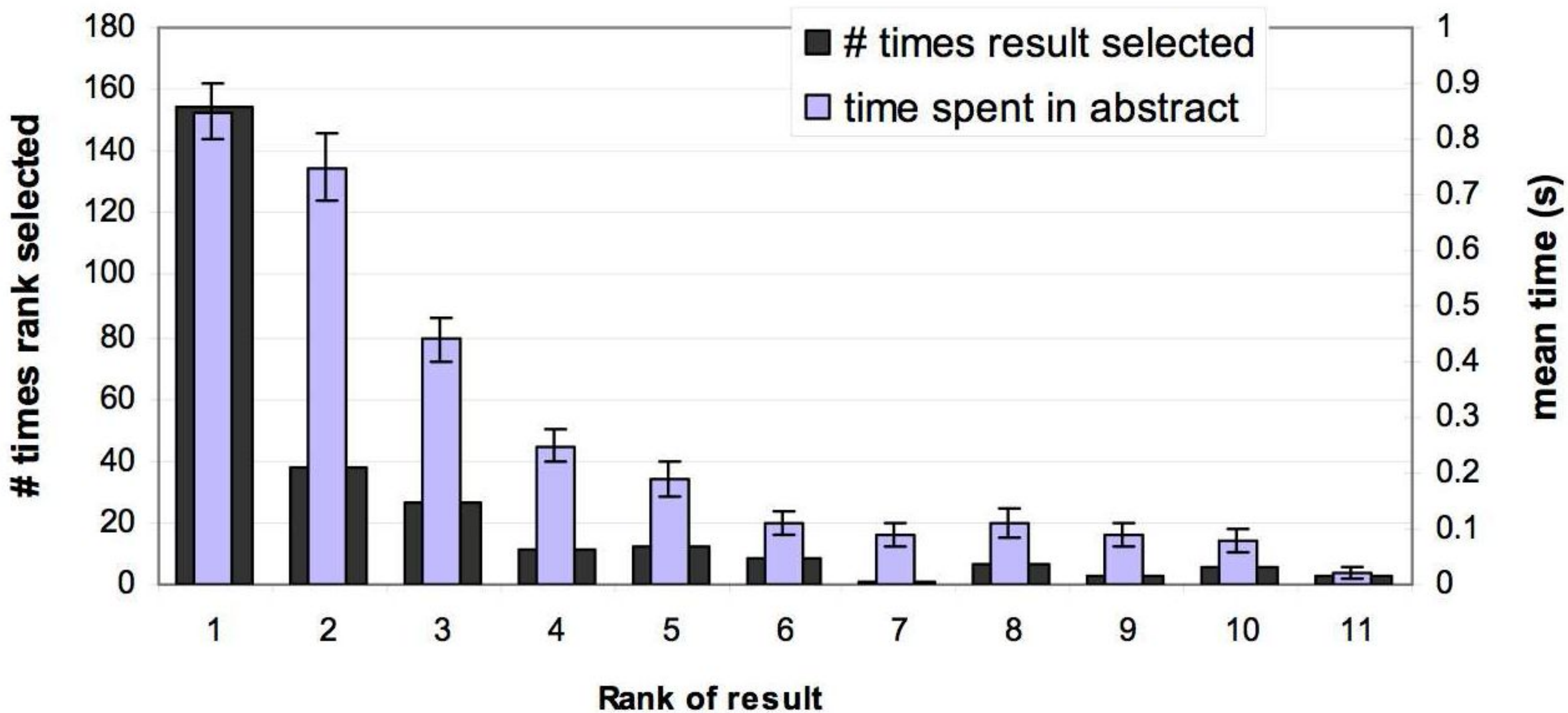
Hinrich Schütze and Christina Lioma

Lecture 13: Text Classification & Naive Bayes

Overview

Overview

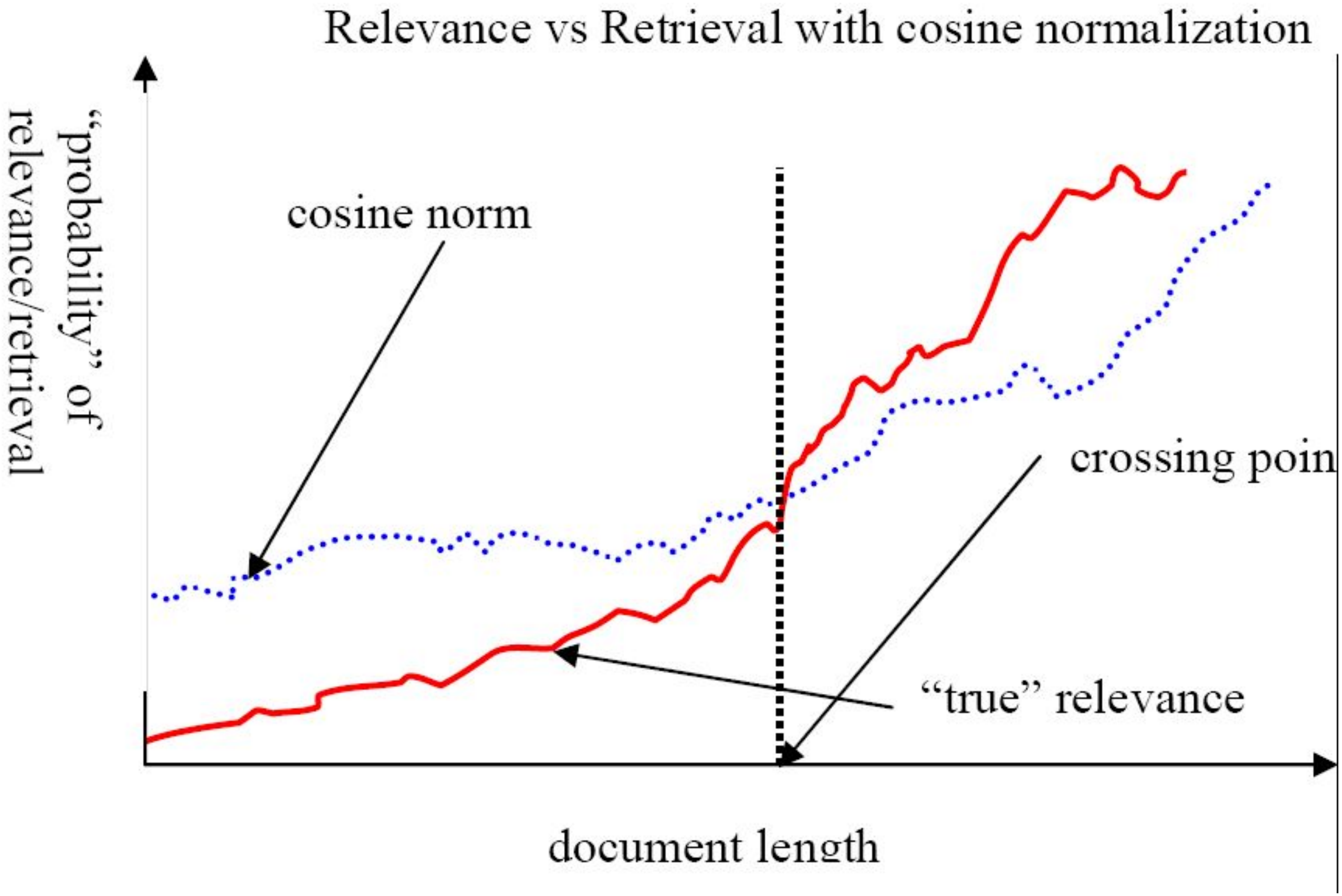
Looking vs. Clicking



- Users view results one and two more often / thoroughly
- Users click most frequently on result one

Google

Pivot normalization

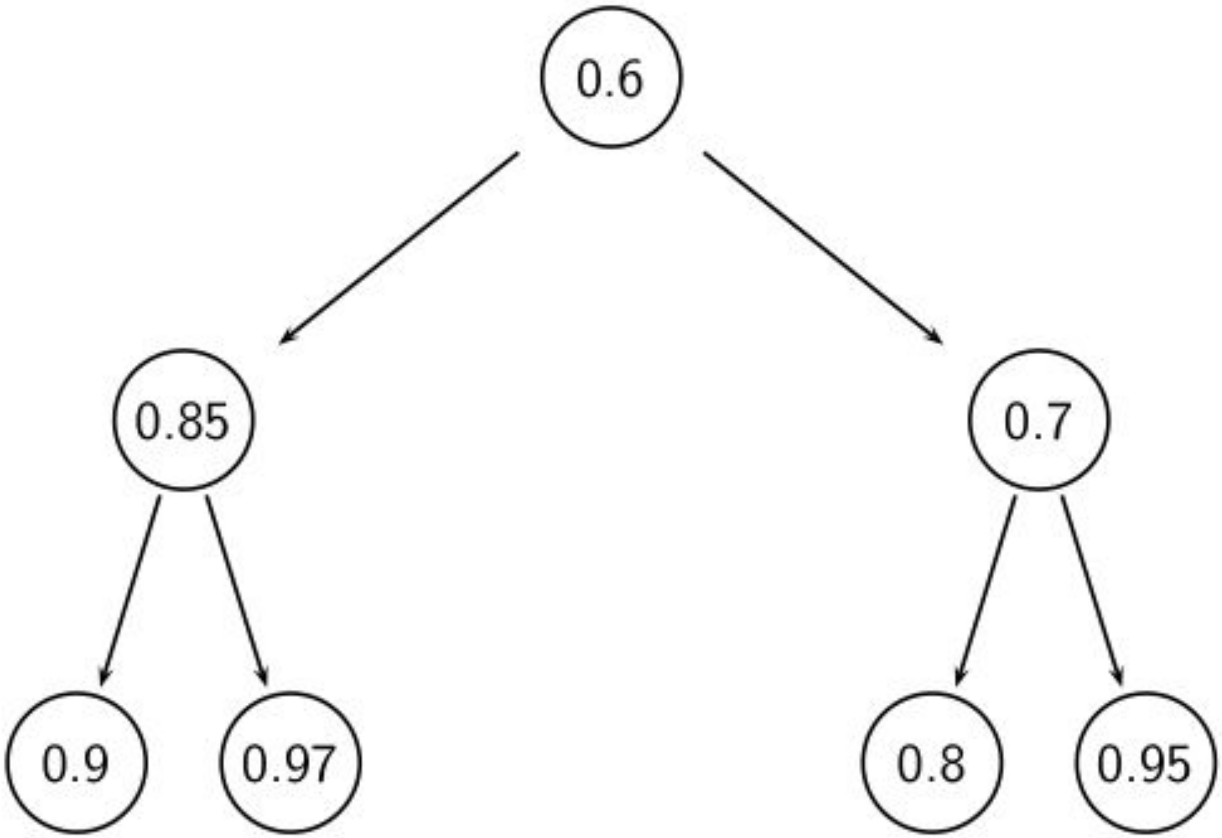


Source: Lillian Lee

Use min heap for selecting top *k* out of *N*

- Use a binary min heap
- A binary min heap is a binary tree in which each node's value is less than the values of its children.
- It takes $O(N \log k)$ operations to construct the k-heap containing the k largest values (where N is the number of documents).
- Essentially linear in N for small k and large N.

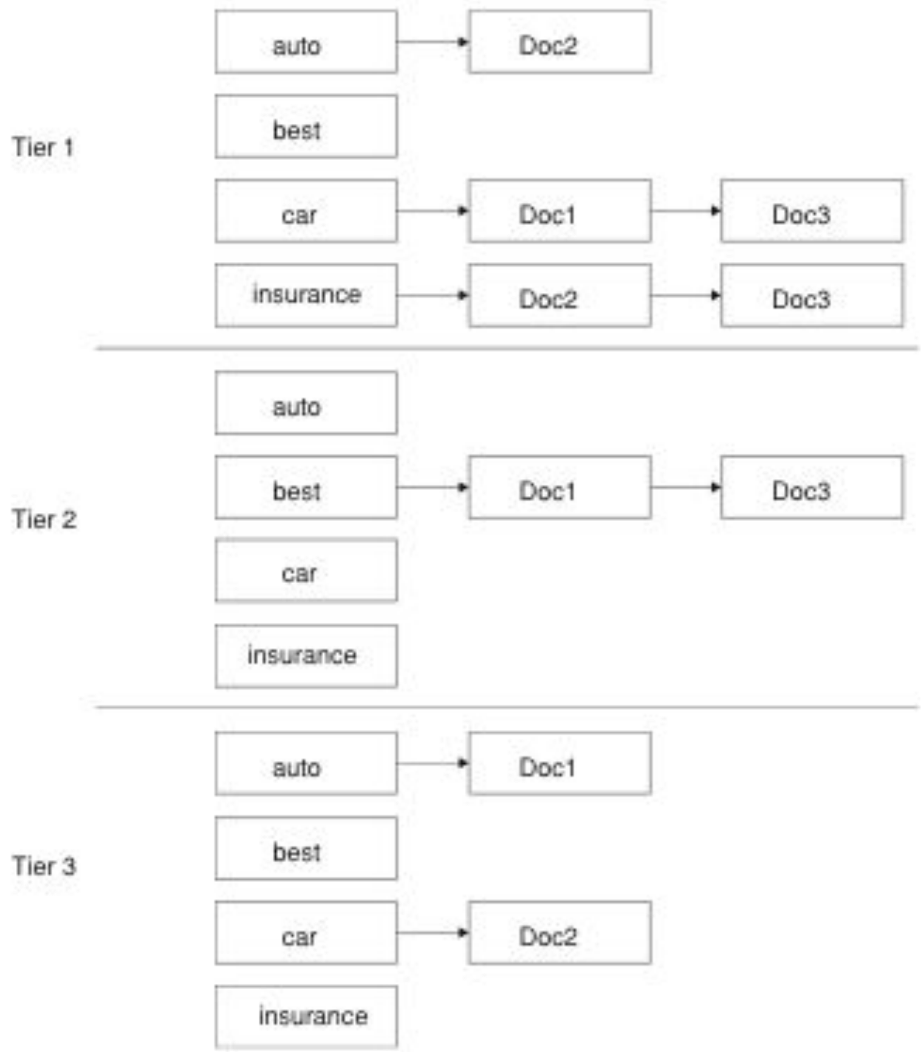
Binary min heap



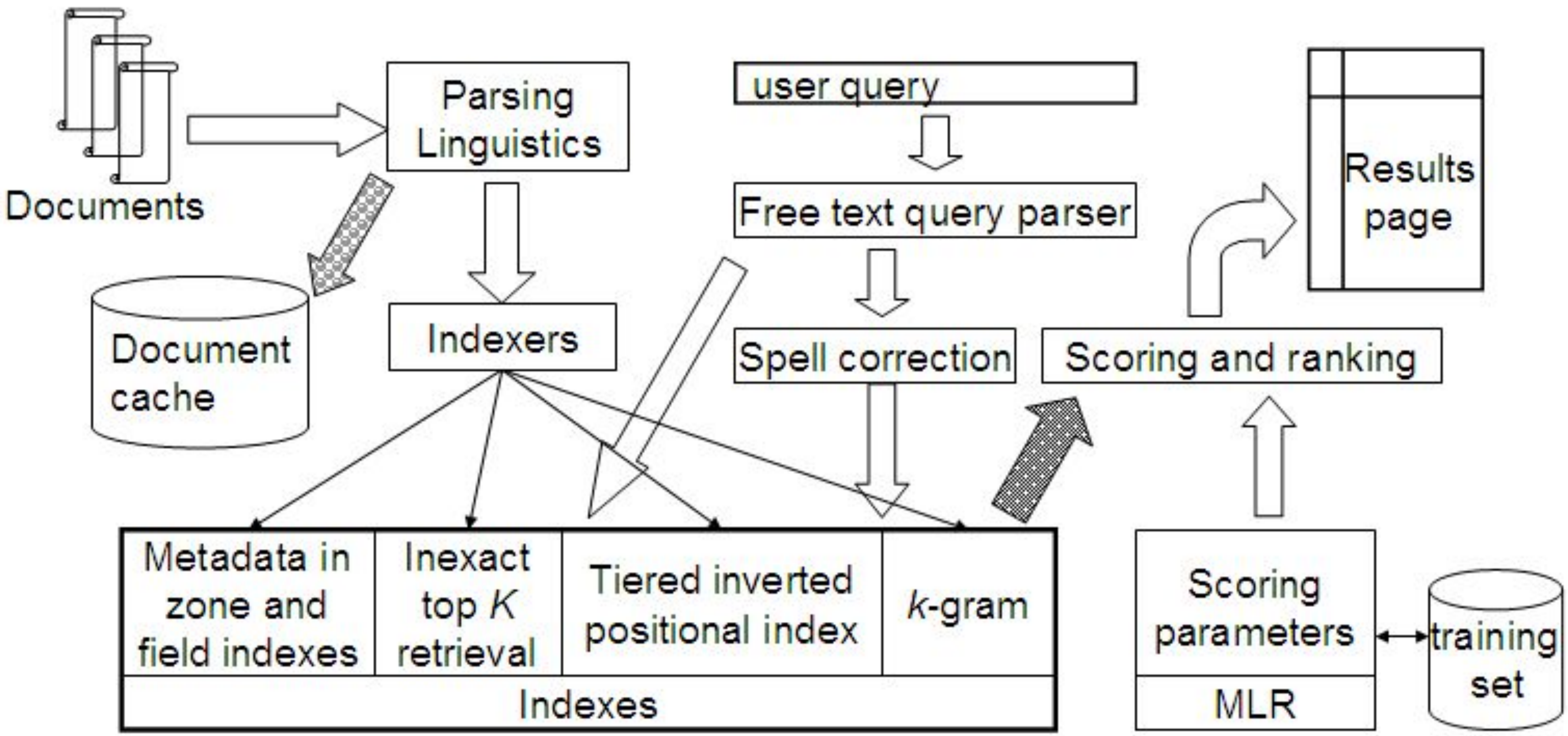
Heuristics for finding the top *k* most relevant

- Document-at-a-time processing
  - We complete computation of the query-document similarity score of document d_i before starting to compute the query-document similarity score of d_{i+1}.
  - Requires a consistent ordering of documents in the postings lists
- Term-at-a-time processing
  - We complete processing the postings list of query term t_i before starting to process the postings list of t_{i+1}.
  - Requires an accumulator for each document "still in the running"
- The most effective heuristics switch back and forth between term-at-a-time and document-at-a-time processing.

Tiered index



Complete search system



Take-away today

- Text classification: definition & relevance to information retrieval
- Naive Bayes: simple baseline text classifier
- Theory: derivation of Naive Bayes classification rule & analysis
- Evaluation of text classification: how do we know it worked / didn’t work?

Outline

1. Recap
2. Text classification
3. Naive Bayes
4. NB theory
5. Evaluation of TC

A text classification task: Email spam filtering

```
From: ''' <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay
Anyone can buy real estate with no money down
Stop paying rent TODAY !
There is no need to spend hundreds or even thousands for
similar courses
I am 22 years old and I have already purchased 6 properties
using the
methods outlined in this truly INCREDIBLE ebook.
Change your life NOW !
=================================================
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
=================================================
```

How would you write a program that would automatically detect and delete this type of message?

Formal definition of TC: Training

Given:

- A document space X
  - Documents are represented in this space – typically some type of high-dimensional space.
- A fixed set of classes $C = \{c_1, c_2, . . . , c_J\}$
  - The classes are human-defined for the needs of an application (e.g., relevant vs. nonrelevant).
- A training set D of labeled documents with each labeled document $<d, c> \in X \times C$

Using a learning method or learning algorithm, we then wish to learn a classifier Υ that maps documents to classes:

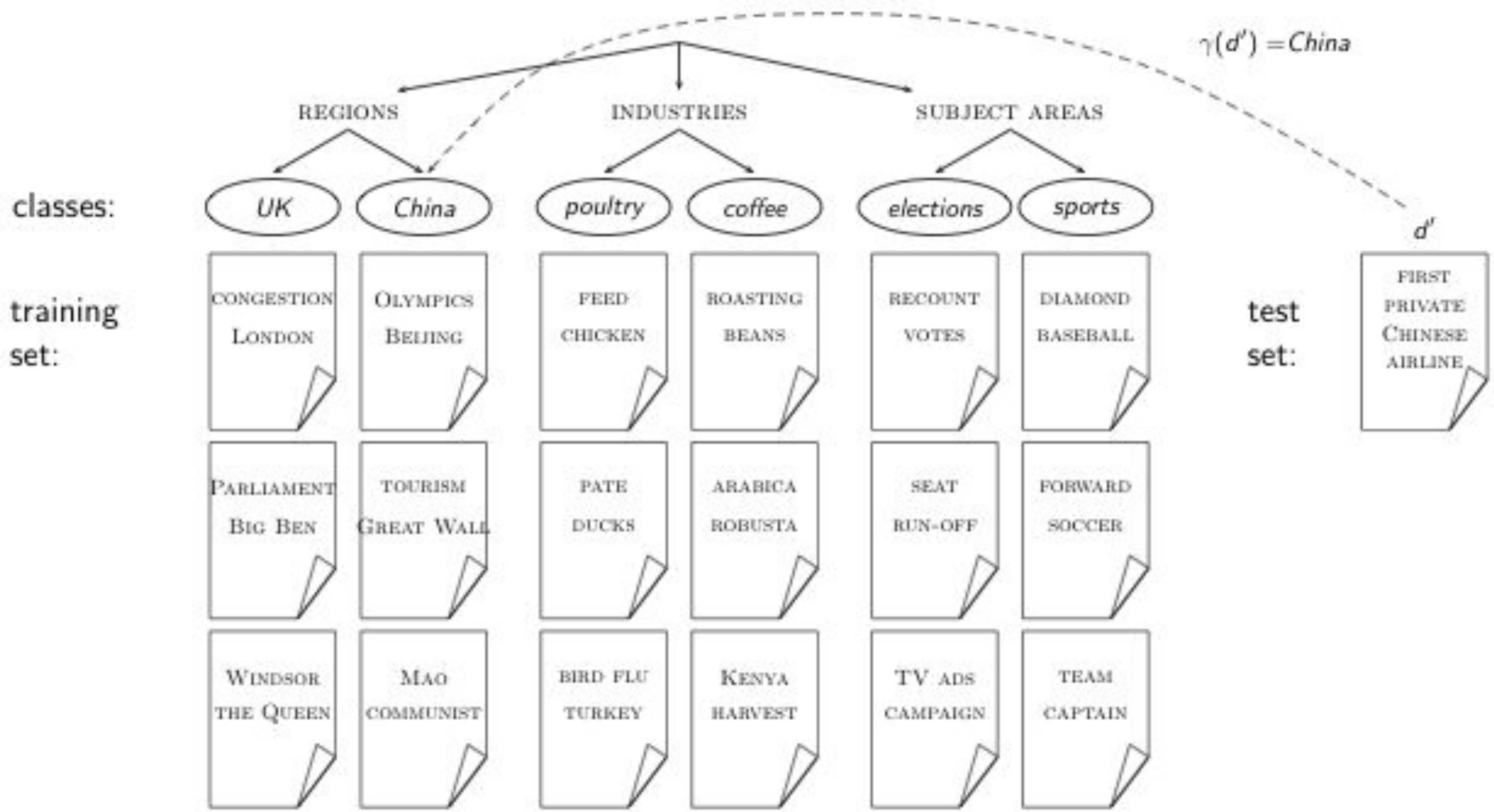
$$\Upsilon : X \rightarrow C$$

Formal definition of TC: Application/Testing

Given: a description $d \in X$ of a document Determine: $\Upsilon(d) \in C$,

that is, the class that is most appropriate for *d*

Topic classification



Exercise

- Find examples of uses of text classification in information retrieval

Examples of how search engines use classification

- Language identification (classes: English vs. French etc.)
- The automatic detection of spam pages (spam vs. nonspam)
- The automatic detection of sexually explicit content (sexually explicit vs. not)
- Topic-specific or *vertical* search – restrict search to a "vertical" like "related to health" (relevant to vertical vs. not)
- Standing queries (e.g., Google Alerts)
- Sentiment detection: is a movie or product review positive or negative (positive vs. negative)

Classification methods: 1. Manual

- Manual classification was used by Yahoo in the beginning of the web. Also: ODP, PubMed
- Very accurate if job is done by experts
- Consistent when the problem size and team is small
- Scaling manual classification is difficult and expensive.
- → We need automatic methods for classification.

Classification methods: 2. Rule-based

- Our Google Alerts example was rule-based classification.
- There are IDE-type development enviroments for writing very complex rules efficiently. (e.g., Verity)
- Often: Boolean combinations (as in Google Alerts)
- Accuracy is very high if a rule has been carefully refined over time by a subject expert.
- Building and maintaining rule-based classification systems is cumbersome and expensive.

A Verity topic (a complex classification rule)

comment line	# Beginning of art topic definition
top-level topic	art ACCRUE
topic definition modifiers	/author = "fsmith"
	/date = "30-Dec-01"
	/annotation = "Topic created by fsmith"
subtopictopic	* 0.70 performing-arts ACCRUE
evidencetopic	** 0.50 WORD
topic definition modifier	/wordtext = ballet
evidencetopic	** 0.50 STEM
topic definition modifier	/wordtext = dance
evidencetopic	** 0.50 WORD
topic definition modifier	/wordtext = opera
evidencetopic	** 0.30 WORD
topic definition modifier	/wordtext = symphony
subtopic	* 0.70 visual-arts ACCRUE
	** 0.50 WORD
	/wordtext = painting
	** 0.50 WORD
	/wordtext = sculpture
subtopic	* 0.70 film ACCRUE
	** 0.50 STEM
	/wordtext = film
subtopic	** 0.50 motion-picture PHRASE
	*** 1.00 WORD
	/wordtext = motion
	*** 1.00 WORD
	/wordtext = picture
	** 0.50 STEM
	/wordtext = movie
subtopic	* 0.50 video ACCRUE
	** 0.50 STEM
	/wordtext = video
	** 0.50 STEM
	/wordtext = vcr
	# End of art topic

Classification methods: 3. Statistical/Probabilistic

- This was our definition of the classification problem – text classification as a learning problem
- (i) Supervised learning of a the classification function ϒ and

  (ii) its application to classifying new documents
- We will look at a couple of methods for doing this: Naive Bayes, Rocchio, kNN, SVMs
- No free lunch: requires hand-classified training data
- But this manual classification can be done by non-experts.

Outline

The Naive Bayes classifier

- The Naive Bayes classifier is a probabilistic classifier.
- We compute the probability of a document *d* being in a class *c* as follows:

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

  - n_d is the length of the document. (number of tokens)
  - $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class *c*
  - $P(t_k|c)$ as a measure of how much evidence t_k contributes that *c* is the correct class.
  - $P(c)$ is the prior probability of *c*.
  - If a document's terms do not provide clear evidence for one class vs. another, we choose the c with highest $P(c)$.

Maximum a posteriori class

- Our goal in Naive Bayes classification is to find the "best" class.
- The best class is the most likely or maximum a posteriori (MAP) class c_{map}:

$$c_{\text{map}} = \underset{c \in \mathbb{C}}{\arg\max}\, \hat{P}(c|d) = \underset{c \in \mathbb{C}}{\arg\max}\ \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c)$$

Taking the log

- Multiplying lots of small probabilities can result in floating point underflow.
- Since log(*xy*) = log(*x*) + log(*y*), we can sum log probabilities instead of multiplying probabilities.
- Since log is a monotonic function, the class with the highest score does not change.
- So what we usually compute in practice is:

$$c_{\text{map}} = \underset{c \in \mathbb{C}}{\arg\max}\, [\log \hat{P}(c) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|c)]$$

Naive Bayes classifier

- Classification rule:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} [\log \hat{P}(c) + \sum_{1 \leq k \leq n_d} \log \hat{P}(t_k|c)]$$

- Simple interpretation:
  - Each conditional parameter log $\hat{P}(t_k|c)$ is a weight that indicates how good an indicator t_k is for c.
  - The prior log $\hat{P}(c)$ is a weight that indicates the relative frequency of c.
  - The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class.
  - We select the class with the most evidence.

Parameter estimation take 1: Maximum likelihood

- Estimate parameters $\hat{P}(c)$ and $\hat{P}(t_k|c)$ from train data: How?
- Prior:

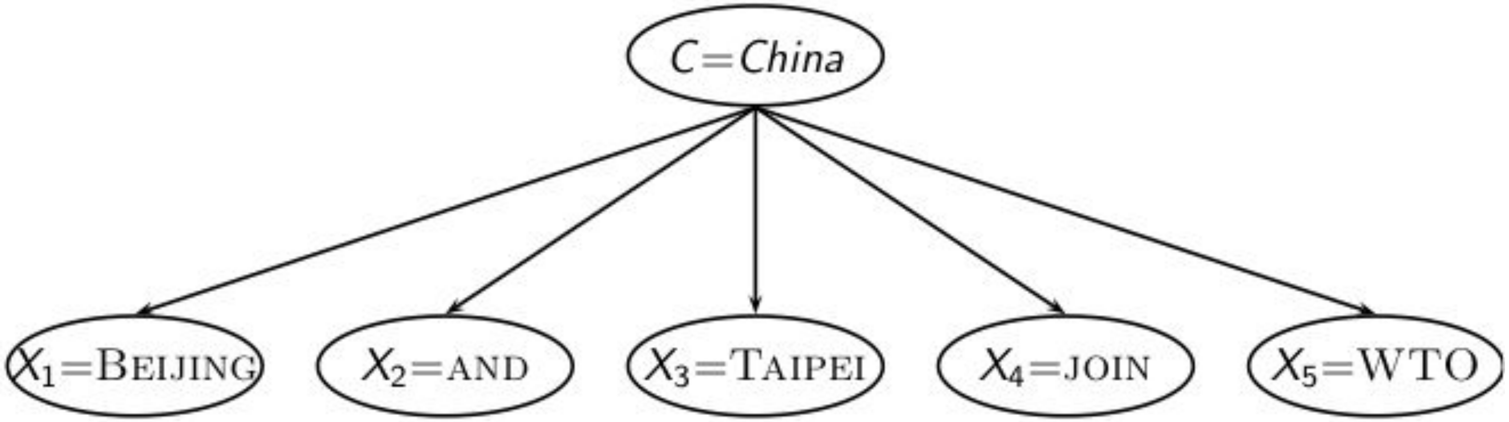
$$\hat{P}(c) = \frac{N_c}{N}$$

- N_c : number of docs in class c; N: total number of docs
- Conditional probabilities:

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

- T_{ct} is the number of tokens of t in training documents from class c (includes multiple occurrences)
- We've made a Naive Bayes independence assumption here: $\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$

The problem with maximum likelihood estimates: Zeros



$$P(China|d) \propto P(China) \cdot P(\texttt{BEIJING}|China) \cdot P(\texttt{AND}|China) \cdot P(\texttt{TAIPEI}|China) \cdot P(\texttt{JOIN}|China) \cdot P(\mathrm{WTO}|China)$$

- If WTO never occurs in class China in the train set:

$$\hat{P}(\mathrm{WTO}|China) = \frac{T_{China,\mathrm{WTO}}}{\sum_{t' \in V} T_{China,t'}} = \frac{0}{\sum_{t' \in V} T_{China,t'}} = 0$$

The problem with maximum likelihood estimates: Zeros (cont)

- If there were no occurrences of WTO in documents in class China, we'd get a zero estimate:

$$\hat{P}(\text{WTO}|\mathit{China}) = \frac{T_{\mathit{China},\text{WTO}}}{\sum_{t' \in V} T_{\mathit{China},t'}} = 0$$

- → We will get P(China|d) = 0 for any document that contains WTO!
- Zero probabilities cannot be conditioned away.

To avoid zeros: Add-one smoothing

- Before:

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

- Now: 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: $|V| = M$)

To avoid zeros: Add-one smoothing

- Estimate parameters from the training corpus using add-one smoothing
- For a new document, for each class, compute sum of (i) log of prior and (ii) logs of conditional probabilities of the terms
- Assign the document to the class with the largest score

Naive Bayes: Training

$\text{TrainMultinomialNB}(\mathbb{C}, \mathbb{D})$

$V \leftarrow \text{ExtractVocabulary}(\mathbb{D})$

$N \leftarrow \text{CountDocs}(\mathbb{D})$

for each $c \in \mathbb{C}$

do $N_c \leftarrow \text{CountDocsInClass}(\mathbb{D}, c)$

  $prior[c] \leftarrow N_c / N$

  $text_c \leftarrow \text{ConcatenateTextOfAllDocsInClass}(\mathbb{D}, c)$

  for each $t \in V$

  do $T_{ct} \leftarrow \text{CountTokensOfTerm}(text_c, t)$

  for each $t \in V$

  do $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$

return $V, prior, condprob$

Naive Bayes: Testing

ApplyMultinomialNB($\mathbb{C}, V, prior, condprob, d$)

$W \leftarrow$ ExtractTokensFromDoc(V, d)
for each $c \in \mathbb{C}$
do $score[c] \leftarrow \log prior[c]$
    for each $t \in W$
    do $score[c] += \log condprob[t][c]$
return $\arg\max_{c \in \mathbb{C}} score[c]$

Exercise

	docID	words in document	in $c = China$?
training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test set	5	Chinese Chinese Chinese Tokyo Japan	?

- Estimate parameters of Naive Bayes classifier
- Classify test document

Example: Parameter estimates

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

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

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 as the vocabulary consists of six terms.

Example: Classification

$$\hat{P}(c|d_5) \propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003$$
$$\hat{P}(\overline{c}|d_5) \propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001$$

Thus, the classifier assigns the test document to *c* = *China*. The reason for this classification decision is that the three occurrences of the positive indicator `CHINESE` in d_5 outweigh the occurrences of the two negative indicators `JAPAN` and `TOKYO`.

Time complexity of Naive Bayes

mode	time complexity
training	$\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}} + \lvert\mathbb{C}\rvert\lvert V\rvert)$
testing	$\Theta(L_{\mathrm{a}} + \lvert\mathbb{C}\rvert M_{\mathrm{a}}) = \Theta(\lvert\mathbb{C}\rvert M_{\mathrm{a}})$

- L_{ave}: average length of a training doc, L_a: length of the test doc, M_a: number of distinct terms in the test doc, $\mathbb{D}$: training set, V : vocabulary, $\mathbb{C}$: set of classes
- $\Theta(\lvert\mathbb{D}\rvert L_{\mathrm{ave}})$ is the time it takes to compute all counts.
- $\Theta(\lvert\mathbb{C}\rvert\lvert V\rvert)$ is the time it takes to compute the parameters from the counts.
- Generally: $\lvert\mathbb{C}\rvert\lvert V\rvert < \lvert\mathbb{D}\rvert L_{\mathrm{ave}}$
- Test time is also linear (in the length of the test document).
- Thus: Naive Bayes is linear in the size of the training set (training) and the test document (testing). This is optimal.

Outline

Naive Bayes: Analysis

- 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

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

- 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.

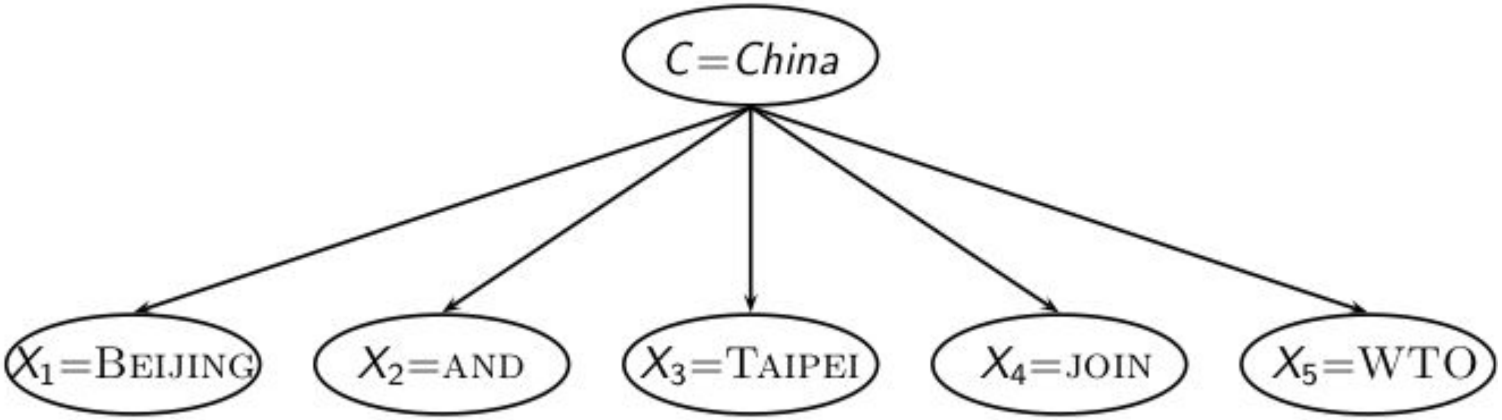
Naive Bayes conditional independence assumption

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 $P(X_k = t_k \mid c)$. 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



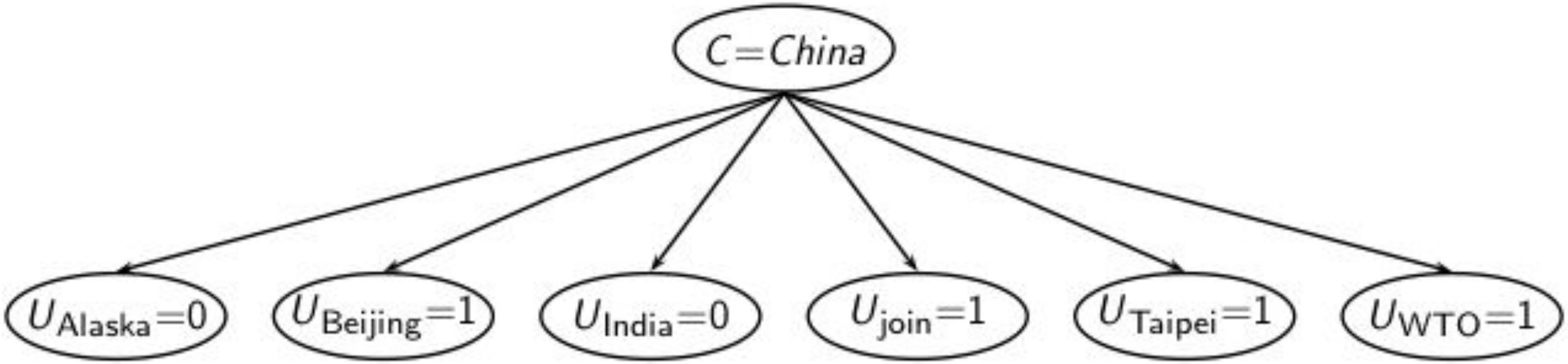
$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c)$$

- 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



Violation of Naive Bayes independence assumption

- 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!

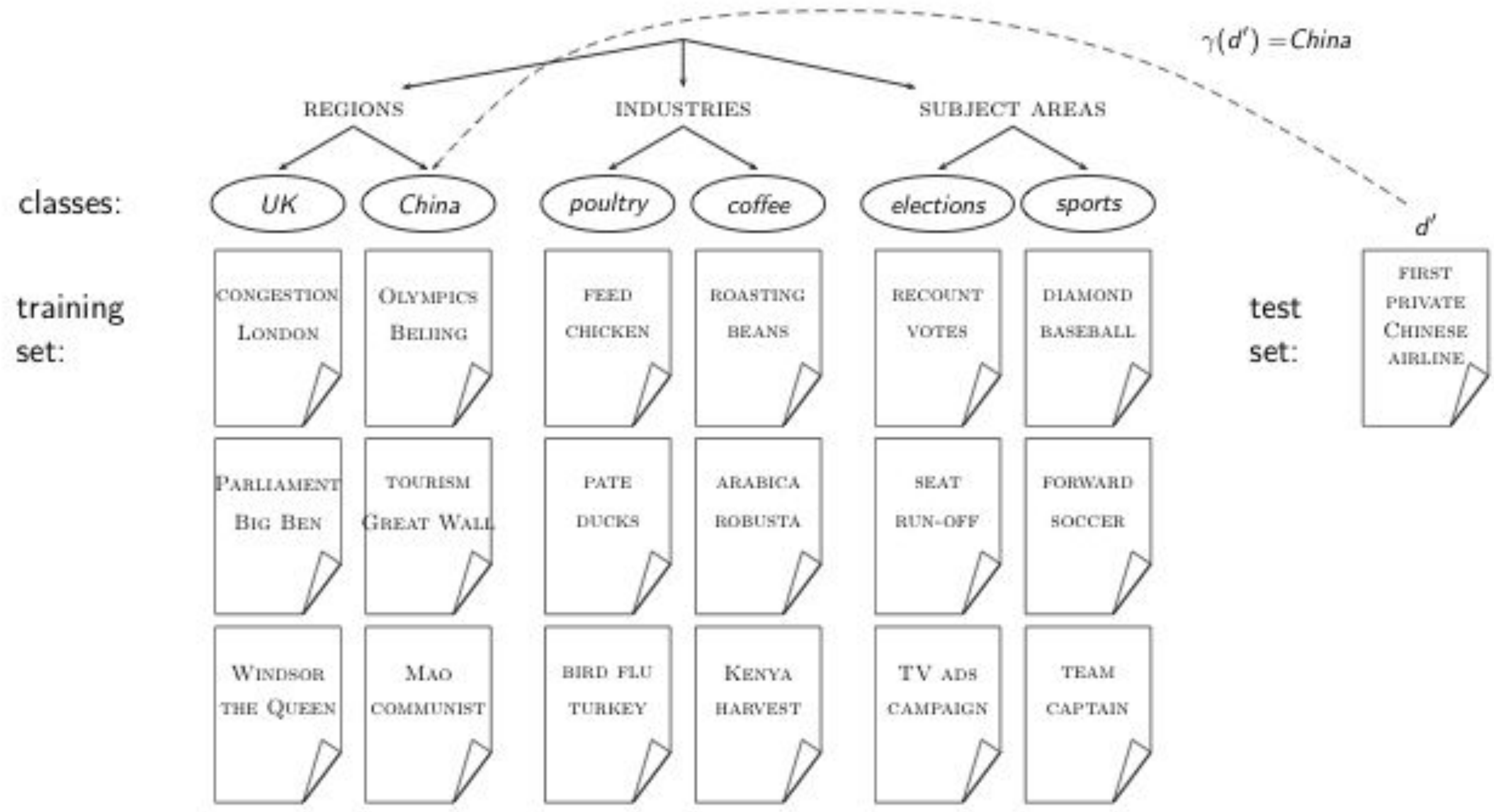
Naive Bayes is not so naive

- Naive 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

Outline

1. Recap
2. Text classification
3. Naive Bayes
4. NB theory
5. Evaluation of TC

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

A Reuters document

REUTERS

You are here: Home > News > Science > **Article**

Go to a Section: U.S. International Business Markets Politics Entertainment Technology Sports Oddly Enoug

Extreme conditions create rare Antarctic clouds

Tue Aug 1, 2006 3:20am ET

Email This Article | Print This Article | Reprints

[-] Text [+]

SYDNEY (Reuters) - Rare, mother-of-pearl colored clouds caused by extreme weather conditions above Antarctica are a possible indication of global warming, Australian scientists said on Tuesday.

Known as nacreous clouds, the spectacular formations showing delicate wisps of colors were photographed in the sky over an Australian

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

Precision *P* and recall *R*

	in the class	not in the class
predicted to be in the class	true positives (TP)	false positives (FP)
predicted to not be in the class	false negatives (FN)	true negatives (TN)

$$P = TP \,/\, (\, TP + FP)$$

$$R = TP \,/\, (\, TP + FN)$$

A combined measure: *F*

- F_1 allows us to trade off precision against recall.
- $$F_1 = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}} = \frac{2PR}{P+R}$$
- This is the harmonic mean of *P* and *R*: $\frac{1}{F} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R})$

Averaging: Micro vs. Macro

- We now have an evaluation measure (F_1) for one class.
- But we also want a single number that measures the aggregate performance over all classes in the collection.
- Macroaveraging
  - Compute F_1 for each of the C classes
  - Average these C numbers
- Microaveraging
  - Compute TP, FP, FN for each of the C classes
  - Sum these C numbers (e.g., all TP to get aggregate TP)
  - Compute F_1 for aggregate TP, FP, FN

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).

Take-away today

- Text classification: definition & relevance to information retrieval
- Naive Bayes: simple baseline text classifier
- Theory: derivation of Naive Bayes classification rule & analysis
- Evaluation of text classification: how do we know it worked / didn’t work?

Resources

- Chapter 13 of IIR
- Resources at `http://ifnlp.org/ir`
  - Weka: A data mining software package that includes an implementation of Naive Bayes
  - Reuters-21578 – the most famous text classification evaluation set (but now it's too small for realistic experiments)