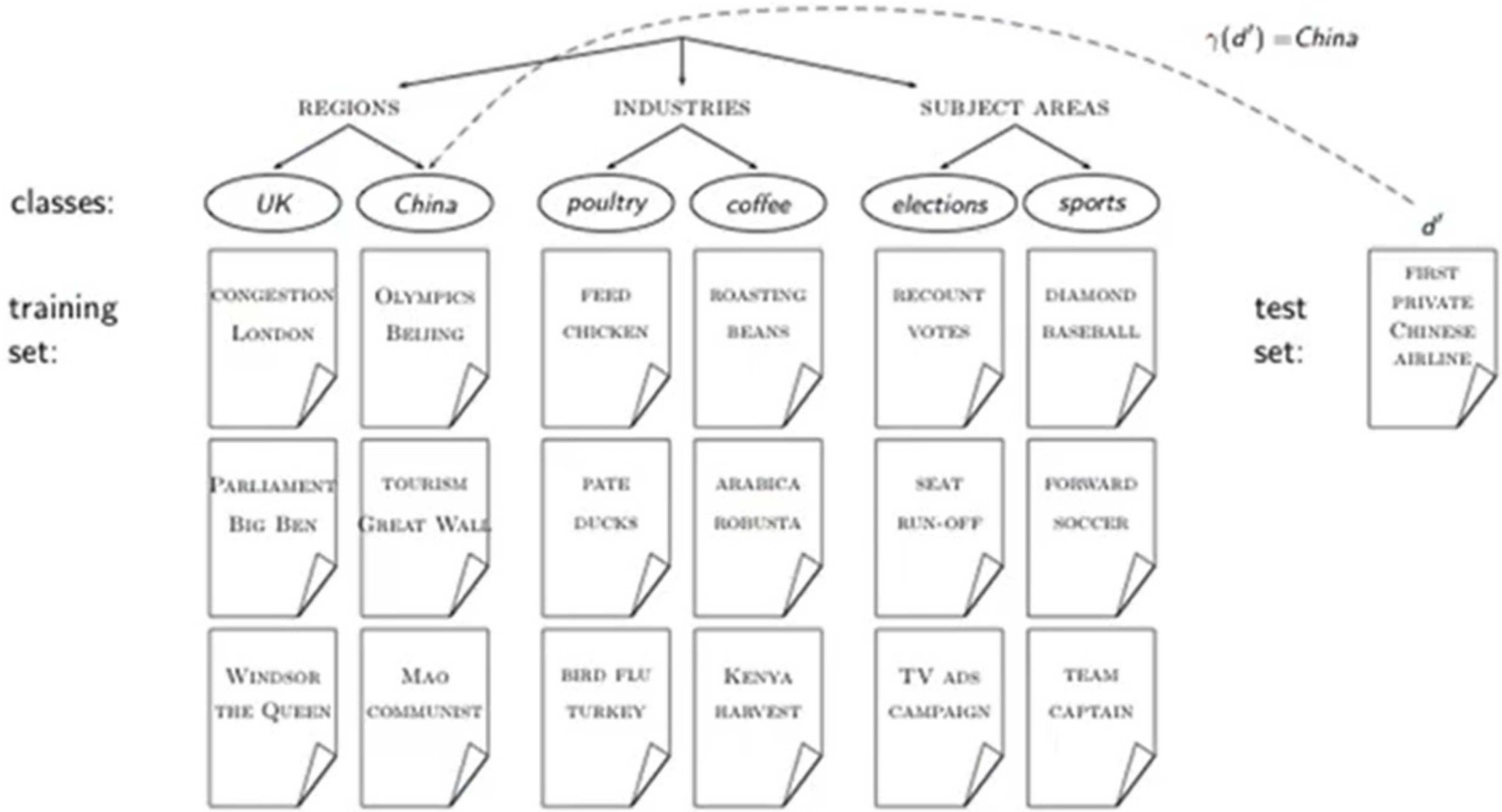
Document Classification

- **What is the role of document classification in information retrieval?**

Classification Methods (3): Supervised learning

- Given:
  - A document *d*
  - A fixed set of classes:
    $C = \{c_1, c_2,\ldots, c_J\}$
  - A training set *D* of documents each with a label in *C*
- Determine:
  - A learning method or algorithm which will enable us to learn a classifier γ
  - For a test document *d*, we assign it the class
  $\gamma(d) \in C$

Document classification



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 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}:

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

Parameter estimation – 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 positional independence assumption here:

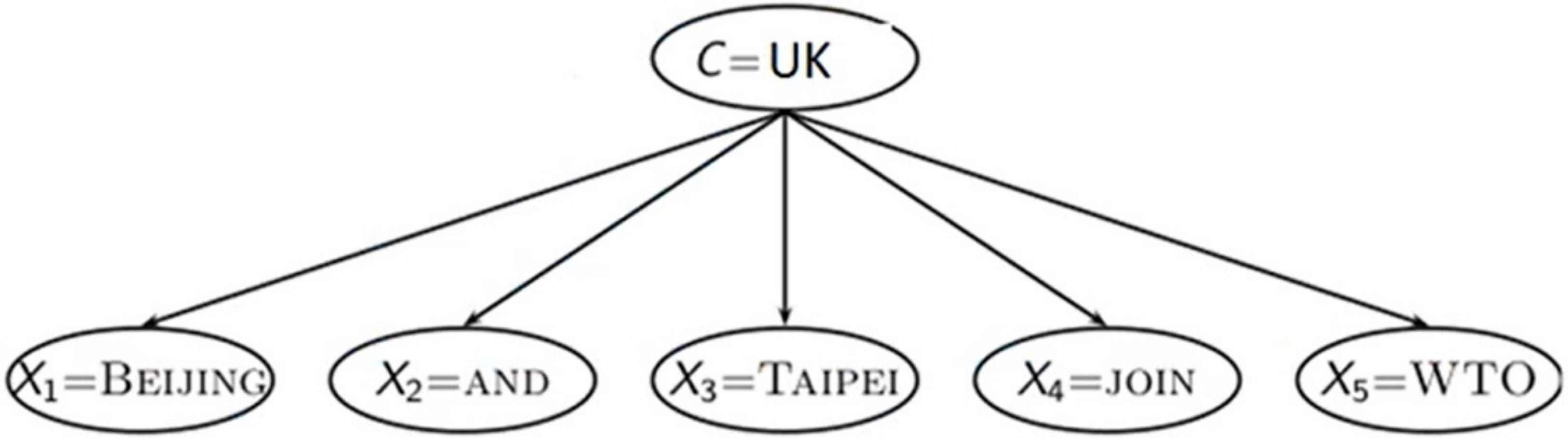
$$\hat{P}(t_{k_1}|c) = \hat{P}(t_{k_2}|c)$$

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.

The problem with maximum likelihood estimates: Zeros



- **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

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

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

- 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:)

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

$$\hat{P}(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|*c*) = (0+1) / (8+6) = 1/14

P(Japan|*c*) = (0+1) / (8+6) = 1/14

P(Chinese|*j*) = (1+1) / (3+6) = 2/9

P(Tokyo|*j*) = (1+1) / (3+6) = 2/9

P(Japan|*j*) = (1+1) / (3+6) = 2/9

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

Naive Bayes: Training

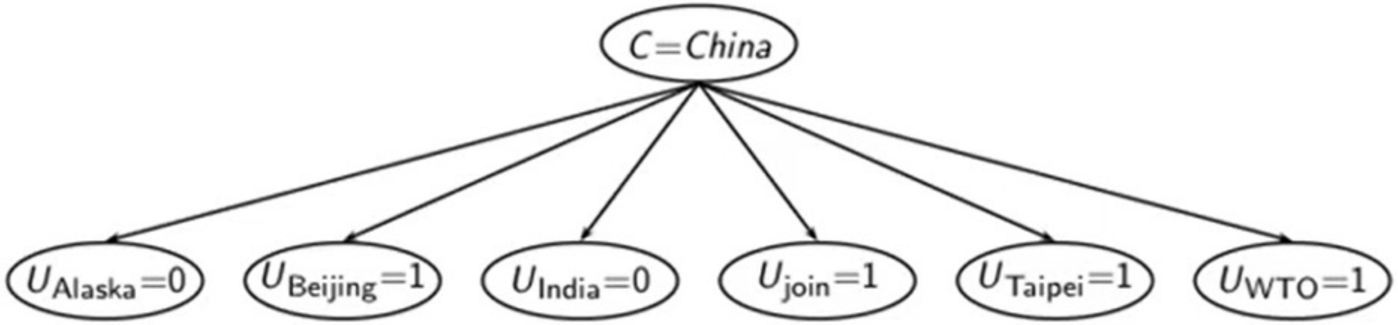
TrainMultinomialNB($\mathbb{C}, \mathbb{D}$)
$V \leftarrow$ ExtractVocabulary($\mathbb{D}$)
$N \leftarrow$ CountDocs($\mathbb{D}$)
for each $c \in \mathbb{C}$
do $N_c \leftarrow$ CountDocsInClass($\mathbb{D}, c$)
    $prior[c] \leftarrow N_c / N$
    $text_c \leftarrow$ ConcatenateTextOfAllDocsInClass($\mathbb{D}, c$)
    for each $t \in V$
    do $T_{ct} \leftarrow$ CountTokensOfTerm($text_c, t$)
    for each $t \in V$
    do $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
return $V, prior, condprob$

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

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

A different Naive Bayes model: Bernoulli model

- The Bernoulli model estimates P(t|c) as the fraction of documents of class c that contain term t
- The multinomial model estimates P (t|c) as the fraction of tokens or fraction of positions in documents of class c that contain term t
- The Bernoulli model considers the binary occurrence information for the terms in the test document ignoring the number of occurences of the term.
- The probability of non-occurrence of the terms of the vocabulary in the test document is also considered.



Algorithm

```
TrainBernoulliNB(ℂ, 𝔻)
V ← ExtractVocabulary(𝔻)
N ← CountDocs(𝔻)
for each c ∈ ℂ
do N_c ← CountDocsInClass(𝔻, c)
    prior[c] ← N_c / N
    for each t ∈ V
    do N_ct ← CountDocsInClassContainingTerm(𝔻, c, t)
        condprob[t][c] ← (N_ct + 1)/(N_c + 2)
return V, prior, condprob

ApplyBernoulliNB(ℂ, V, prior, condprob, d)
V_d ← ExtractTermsFromDoc(V, d)
for each c ∈ ℂ
do score[c] ← log prior[c]
    for each t ∈ V
    do if t ∈ V_d
          then score[c] += log condprob[t][c]
          else score[c] += log(1 − condprob[t][c])
return arg max_{c∈ℂ} score[c]
```

Test doc: “buy cheap dinner”

Doc id	cheap	buy	banking	dinner	the	class
1	0	0	0	0	1	Not spam
2	1	0	1	0	1	spam
3	0	0	0	0	1	Not spam
4	1	0	1	0	1	spam
5	1	1	0	0	1	spam
6	0	0	1	0	1	Not spam
7	0	1	1	0	1	Not spam
8	0	0	0	0	1	Not spam
9	0	0	0	0	1	Not spam
10	1	1	0	1	1	Not spam