Text Classification & Naive Bayes

Chapter 13 - IIR

SAPIENZA
UNIVERSITÀ DI ROMA

Fabrizio Silvestri

Quick Intro

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 proceed to decide whether this text is spam or ham?

SAPIENZA
UNIVERSITÀ DI ROMA

Text Classification: a (more) formal definition

- 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., spam vs. nonspam).
  - A training set D of labeled documents.
    - Each labeled document $(d, c) \in X \times C$

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

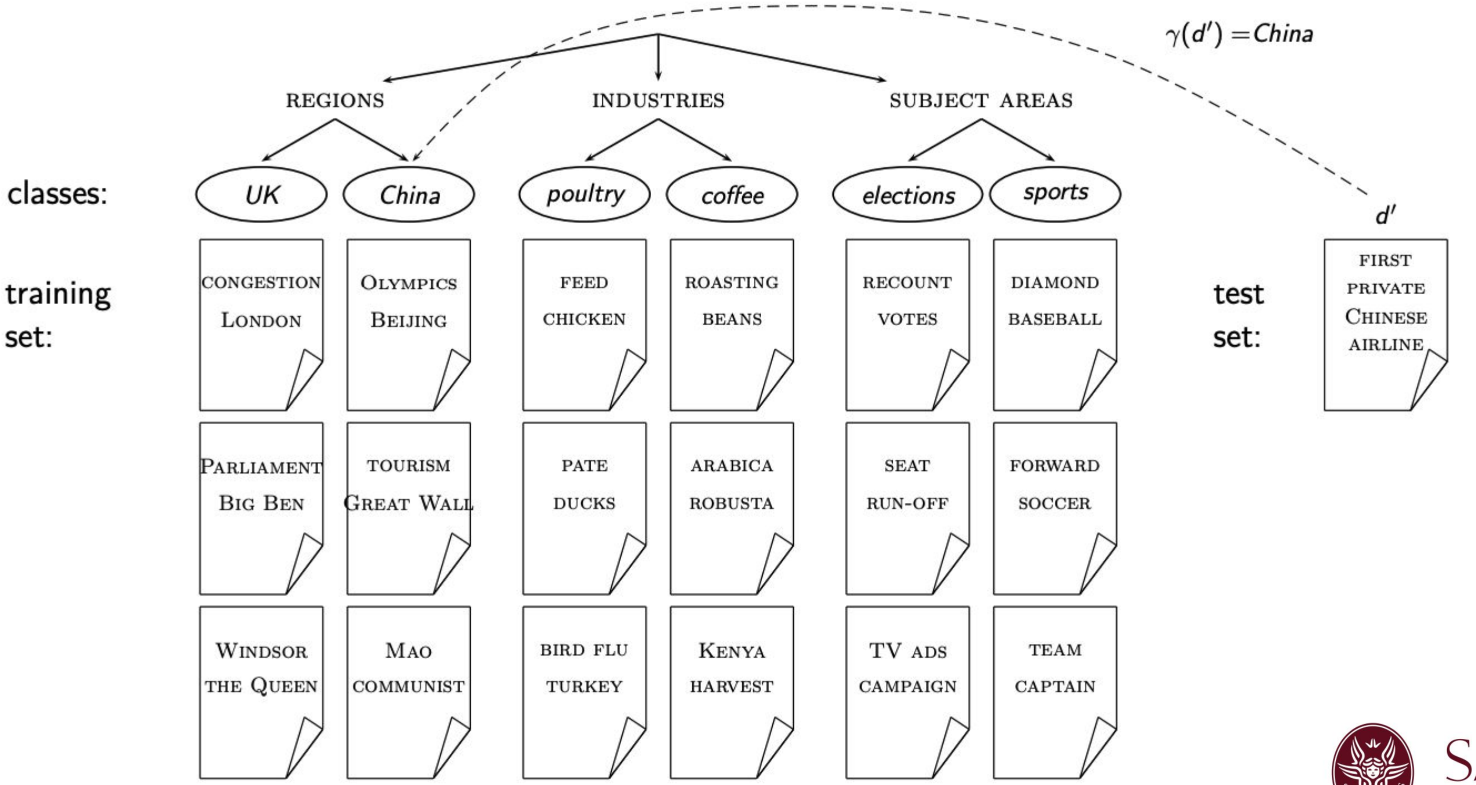
$$\mathbf{f : X \rightarrow C}$$

SAPIENZA
UNIVERSITÀ DI ROMA

Text Classification: Inference

- Given a representation for a document $d \in X$, determine the most appropriate class using the learnt function f
- In other words $C = f(d)$

An example: Topic Classification

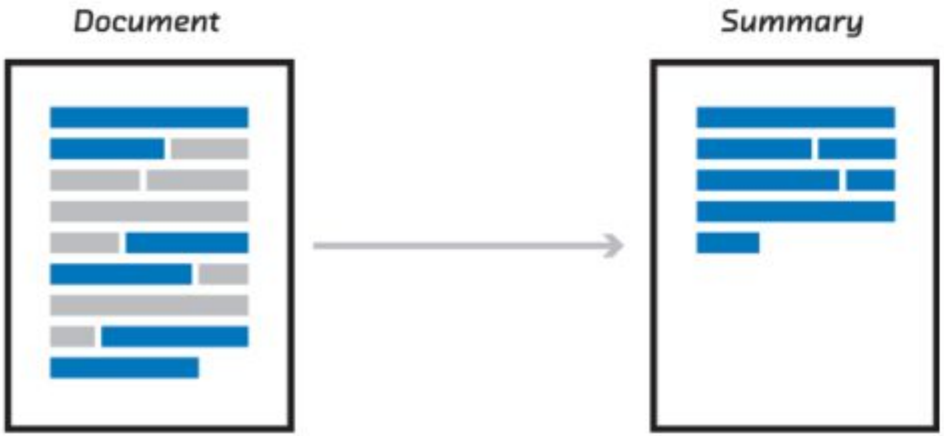


An example: Sentiment Analysis

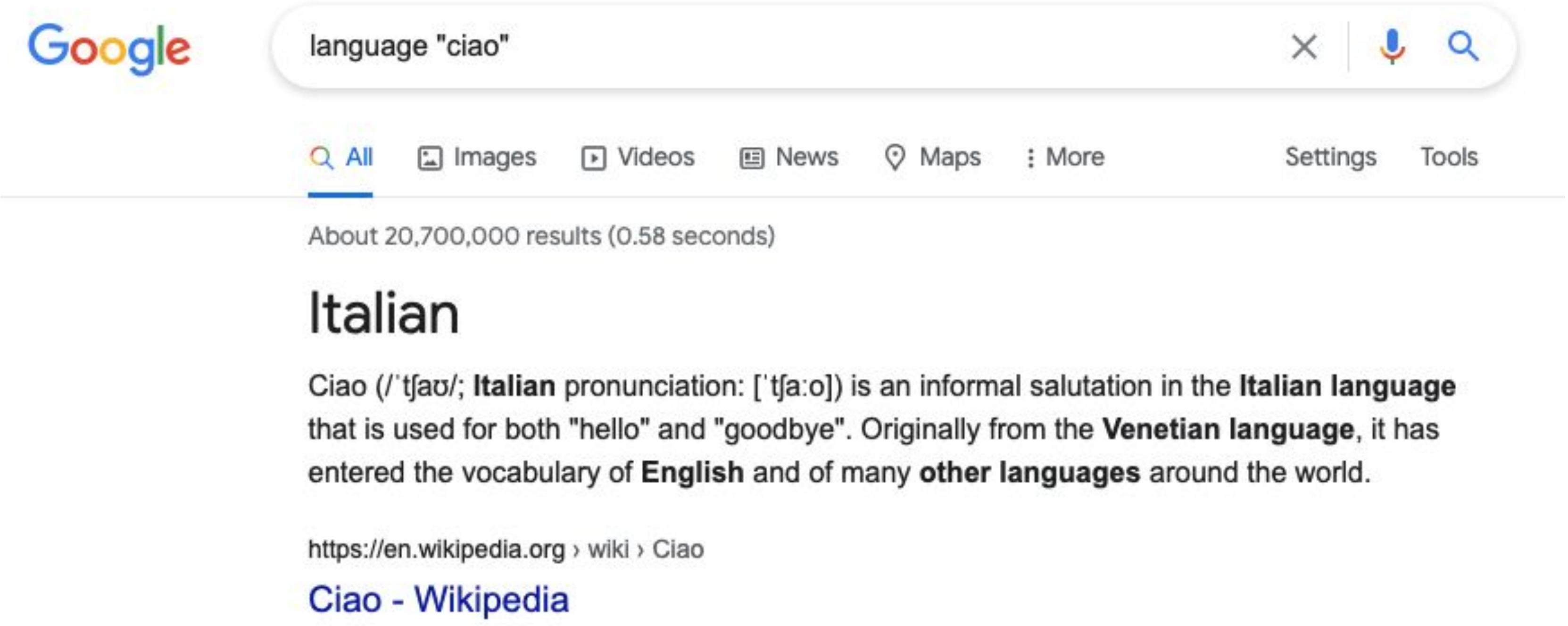
SAPIENZA
UNIVERSITÀ DI ROMA

An example: Misinformation Classification

An example: Automatic Summarization

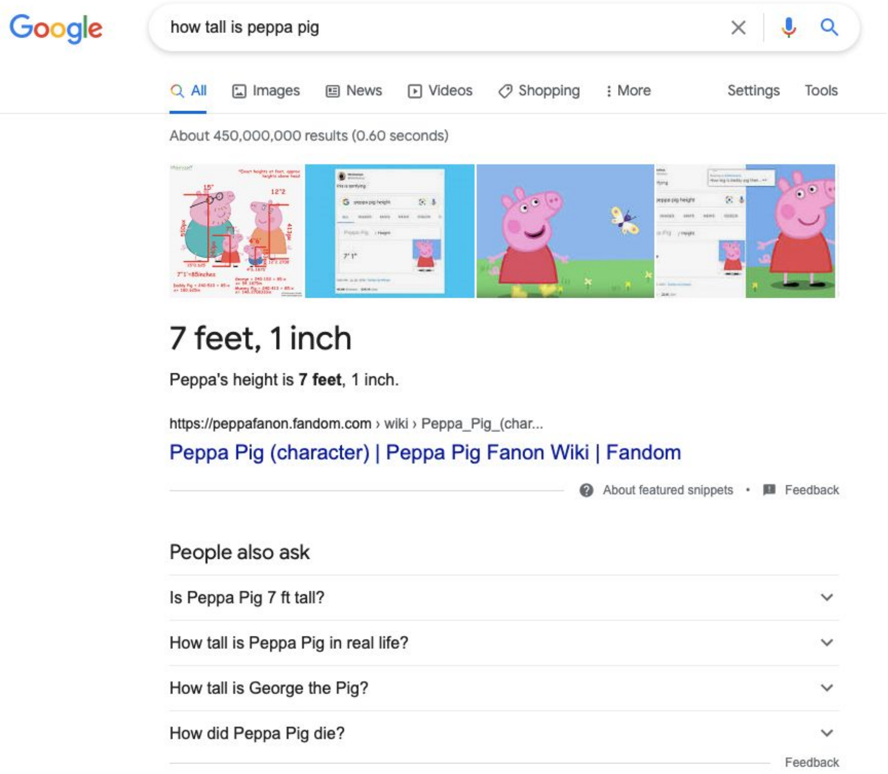


An example: Language Detection



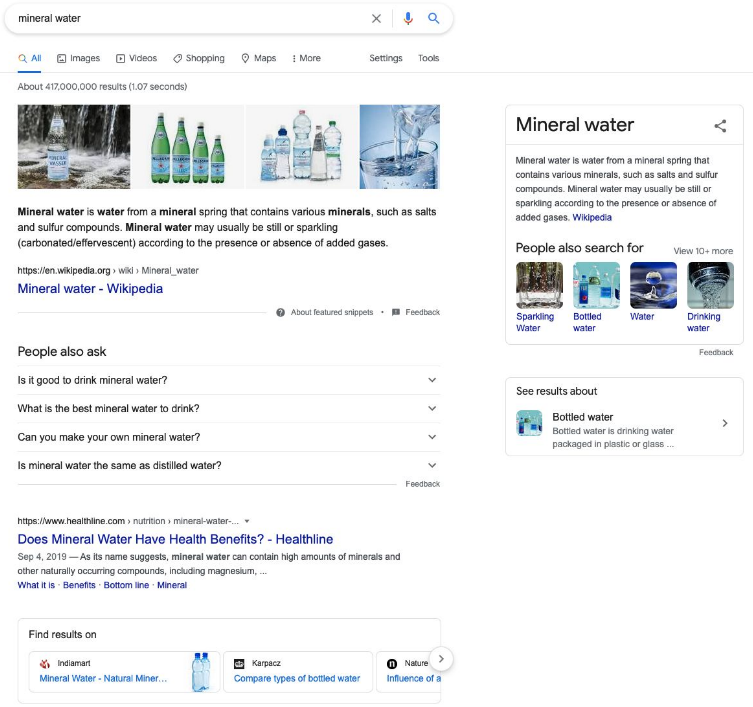
SAPIENZA
UNIVERSITÀ DI ROMA

An example: Question Detection

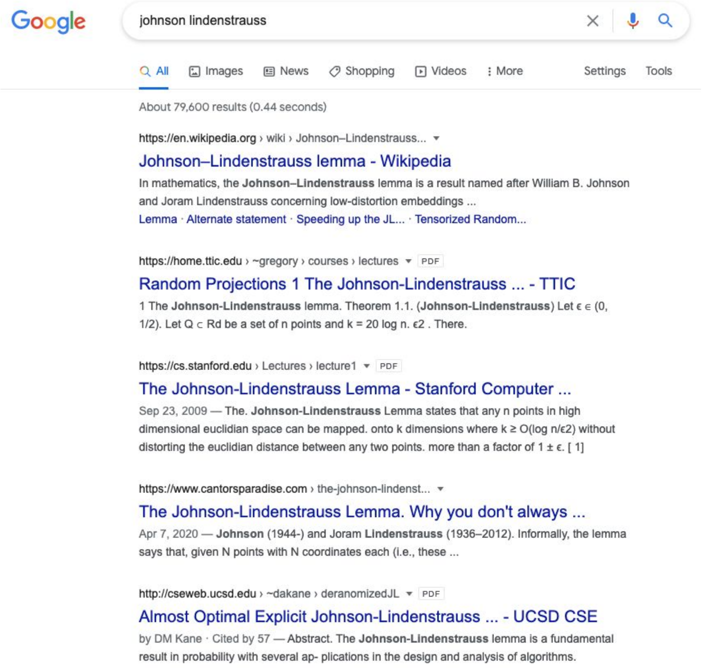


SAPIENZA
UNIVERSITÀ DI ROMA

An example: Vertical Selection



vs.



SAPIENZA
UNIVERSITÀ DI ROMA

Some Text Classification Methods

Rule Based

- E.g., Google Alerts is rule-based classification.
- There are IDE-type development environments 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.

Statistical Modeling

- This was our definition of the classification problem – text classification as a learning problem
  - (i) Supervised learning of a classification function **f** and
  - (ii) application of **f** to classifying new documents
- We will look at two methods for doing this: Naive Bayes and SVMs
- No free lunch: requires hand-classified training data
  - But this manual classification can be done by non-experts.

Naïve Bayes

What is a Naïve 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)
- $P(t_k \mid c)$ is the conditional probability of term t_k occurring in a document of class c
- $P(t_k \mid c)$ as a measure of how much evidence tk 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).

SAPIENZA
UNIVERSITÀ DI ROMA

Maximum A Posteriori (MAP) Classifier

- 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}:

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

Actually...

- 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:

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

Summarizing: Naïve Bayes Classifier

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

- Each conditional parameter log $\hat{P}(t_k \mid c)$ 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.

SAPIENZA
UNIVERSITÀ DI ROMA

Parameter Estimation: MLE

^

^

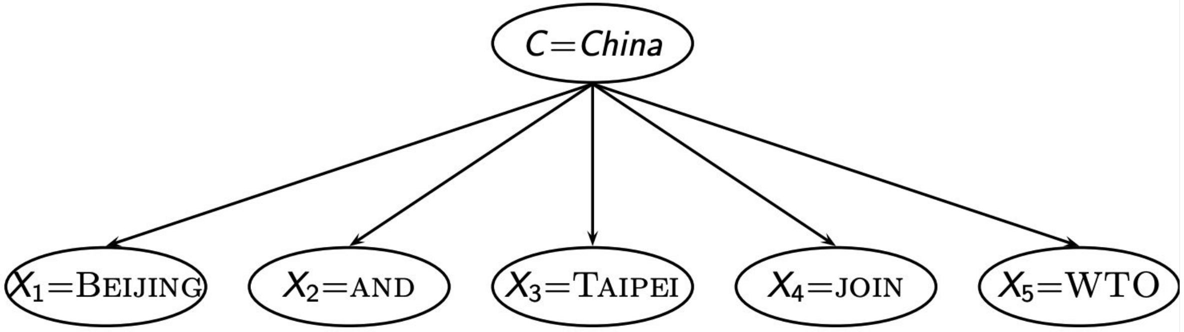
- Estimate parameters $P(c)$ and $P(t_k \mid c)$ from train data: How?
- Prior: $P(c) = N_c / N$
    - N_c : number of docs in class c; N: total number of docs
- Conditional probabilities:

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

- 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: $P(t_k \mid c) = P(t_k \mid c)$, independent of position.

SAPIENZA UNIVERSITÀ DI ROMA

How to deal with zeros?



$$
\begin{aligned}
P(China|d) \quad \propto \quad & P(China) \cdot P(\text{BEIJING}|China) \cdot P(\text{AND}|China) \\
& \cdot P(\text{TAIPEI}|China) \cdot P(\text{JOIN}|China) \cdot P(\text{WTO}|China)
\end{aligned}
$$

We will get P(China|d) = 0 for any document that contains WTO!

er occurs in class China in the train set:

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

Add 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 bins – in this case the number of different words or the size of the vocabulary |V | = M

Naïve Bayes: Summary

- 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

Naïve Bayes: Training

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

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

Naïve Bayes: Testing

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

$\quad W \leftarrow \textsc{ExtractTokensFromDoc}(V, d)$

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

$\quad \textbf{do } score[c] \leftarrow \log prior[c]$

$\qquad \textbf{for each } t \in W$

$\qquad \textbf{do } score[c] += \log condprob[t][c]$

$\quad \textbf{return } \arg\max_{c \in \mathbb{C}} score[c]$

Exercise: Parameter Estimation

	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	?

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

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

(B is the number of bins – in this case the number of different words or the size of the vocabulary $|V| = M$)

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

Solution

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

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

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 as the vocabulary consists of six terms.

Solution

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

Thus, the classifier assigns the test document to $c = \textit{China}$. 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.

Implementing NB

- NB from scratch
- Using scikit-learn

Naïve Bayes: Computational Complexity

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

- 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, **D**: training set, **V** : vocabulary, **C**: set of classes
- $\Theta(|\mathbf{D}|L_{ave})$ is the time it takes to compute all counts.
- $\Theta(|\mathbf{C}||\mathbf{V}|)$ is the time it takes to compute the parameters from the counts.
- Generally: $|\mathbf{C}||\mathbf{V}| < |\mathbf{D}|L_{ave}$
- 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.

Theoretical Analysis of Naïve Bayes

Properties of NB

- Now we want to gain a better understanding of the properties of Naive Bayes.
- We will formally derive the classification rule . . .
- . . . and make our assumptions explicit.

Sapienza
Università di Roma

Derivation of NB

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

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

- Apply Bayes rule P(c|d) = (1/P(d)) * P(d|c)P(c)

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

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

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

Data sparsity

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

- There are too many parameters $P(<t_1,\ldots,t_k, \ldots,t_n>|c)$, 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.

NB: conditional independence assumption

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

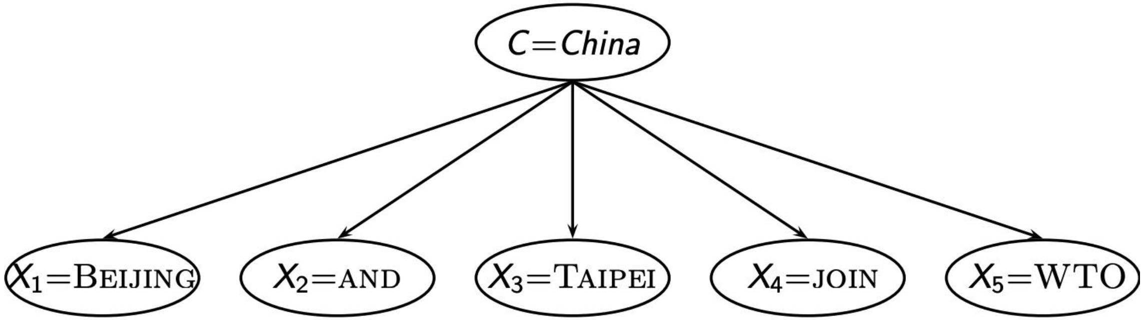
$$P(d|c) = P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \leq k \leq n_d} P(X_k = t_k | c)$$

- 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 conditional probabilities:

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

SAPIENZA
UNIVERSITÀ DI ROMA

NB as a Generative Model



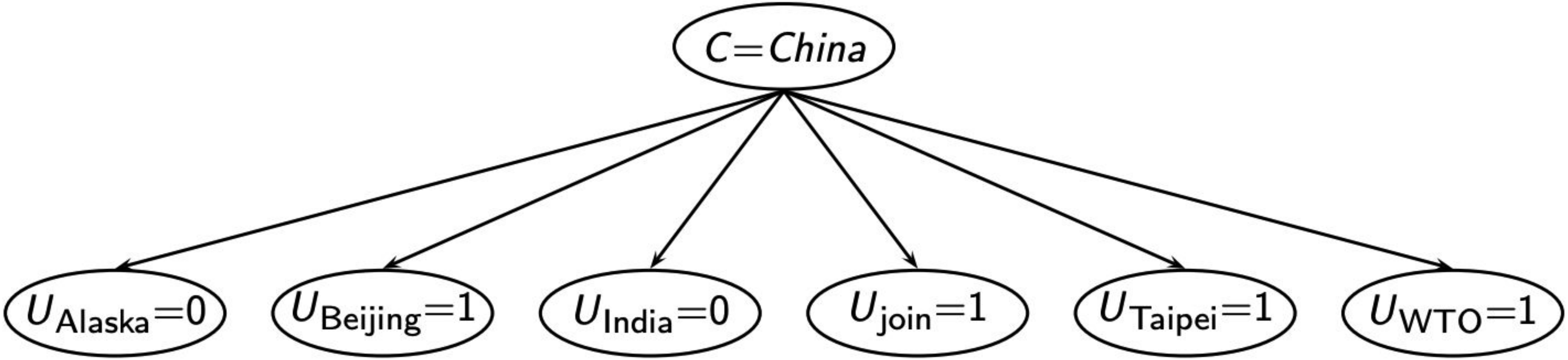
$$P(c|d) \propto P(c) \prod_{1<k<n_d} P(t_k|c)$$

- 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.

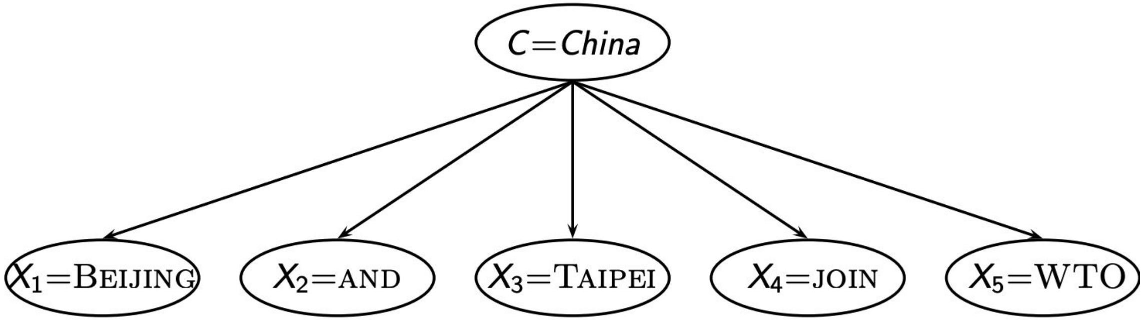
Naïve Bayes: The Second Independence Assumption

- $\hat{P}(X_{k_1} = t|c) = \hat{P}(X_{k_2} = t|c)$
- 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.

Bernoulli NB



Gaussian NB



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

- Each $P(t_k|c) \sim N(m, s)$
  - Normally distributed with parameter m and s estimated on the training set

Violations of the Independence Assumptions

- Conditional independence:
  - $P(\langle t_1, \ldots, t_{n_d} \rangle | c) = \prod_{1 \le k \le n_d} P(X_k = t_k | c)$
- Positional independence:
  - $\hat{P}(X_{k_1} = t | c) = \hat{P}(X_{k_2} = t | c)$
- The independence assumptions do not really hold of documents written in natural language.
- 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 NB 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

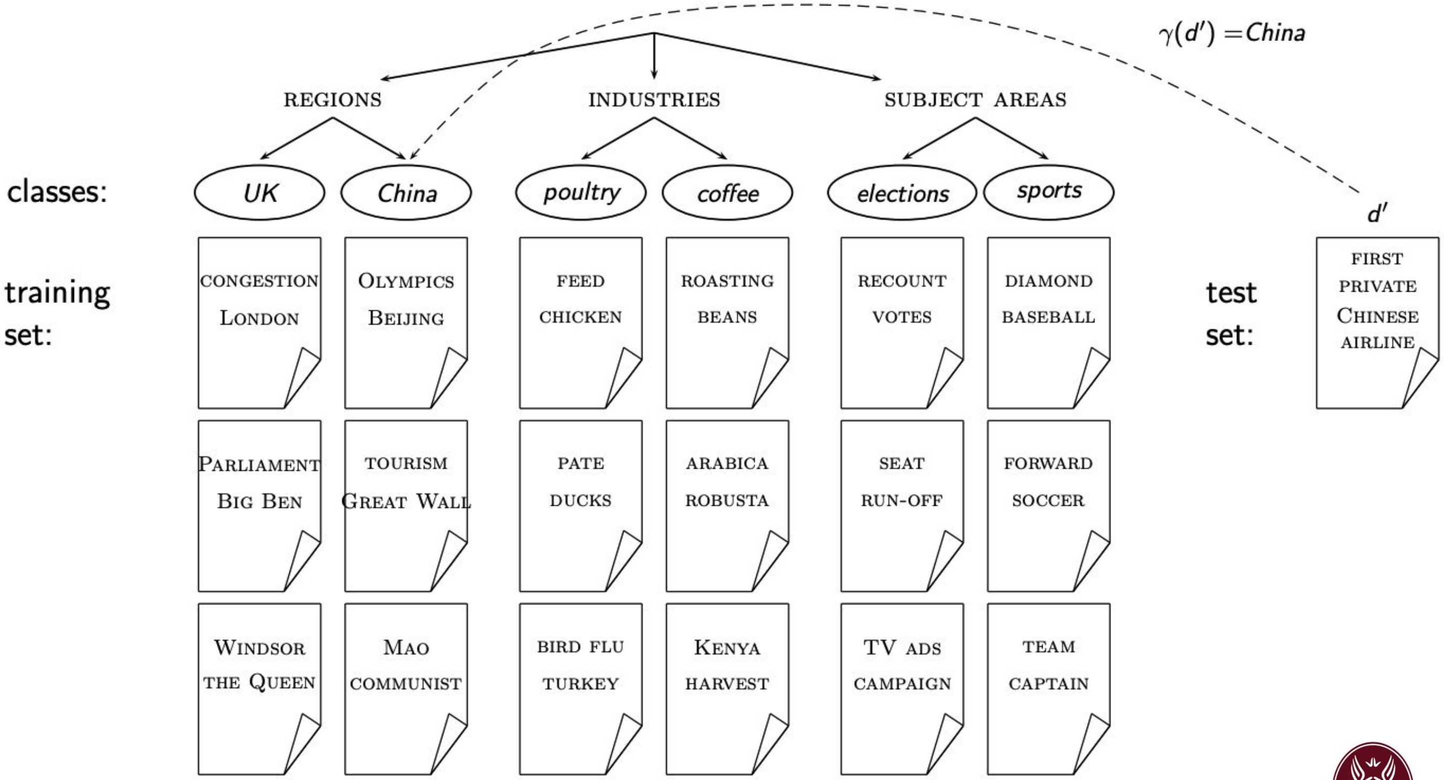
- Double counting of evidence causes underestimation (0.01) and overestimation (0.99).
- Classification is about predicting the correct class and not about accurately estimating probabilities.
- Naive Bayes is terrible for correct estimation . . .
  - but if often performs well at accurate prediction (choosing the correct class).

NB is it 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

Evaluation

Reuters Collection



The Reuters Collection

symbol	statistic	value
N	documents	800,000
L	avg. # word tokens per document	200
M	word types	400,000

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

An example of a Reuters Document

May 4, 2021
2:17 PM CEST

India

‘Last resort’: Desperate for oxygen, Indian hospitals go to court

4 minute read

Aditya Kalra

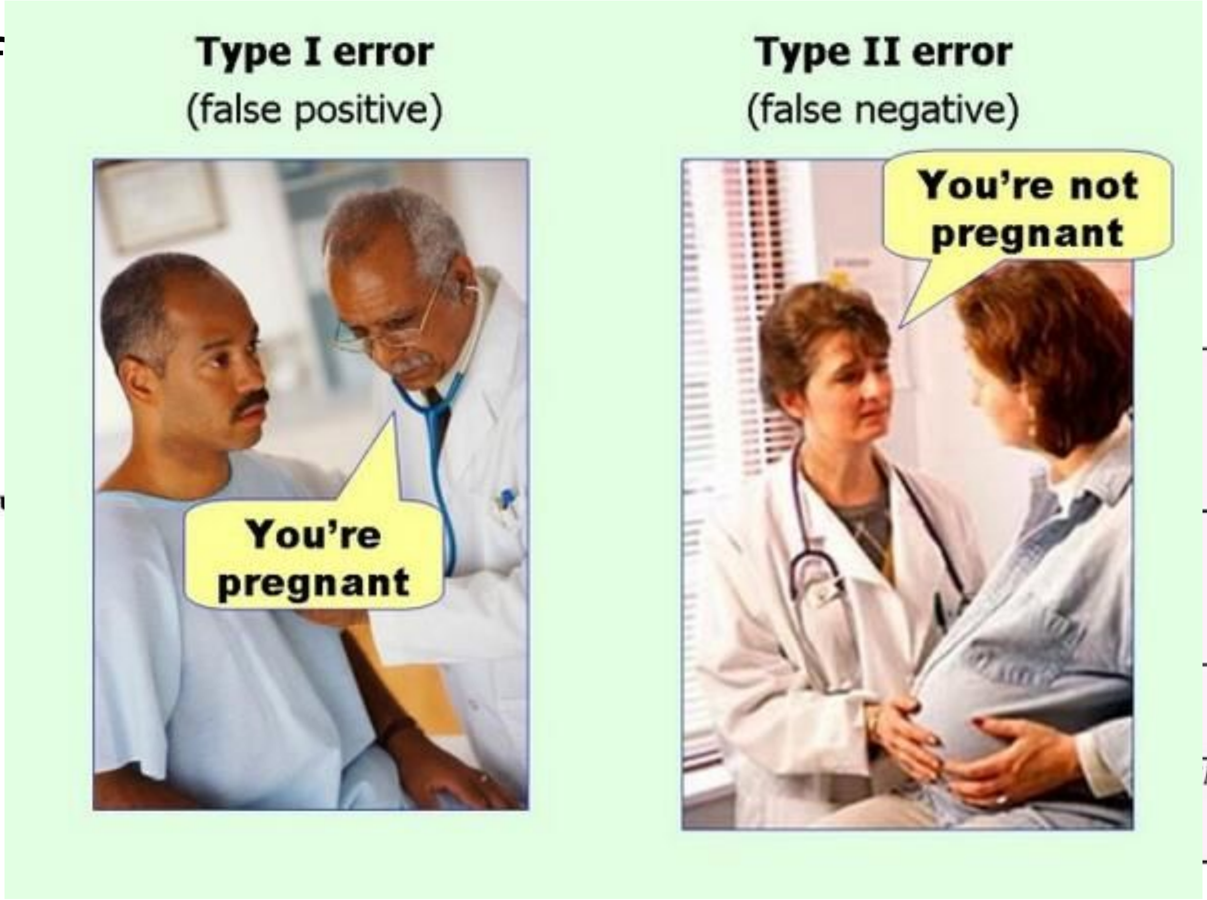
Patients suffering from the coronavirus disease (COVID-19) get treatment at the casualty ward in Lok Nayak Jai Prakash

A court in India’s capital New Delhi has become the last hope for many hospitals **struggling to get oxygen for COVID-19 patients** as supplies run dangerously short while government officials bicker over who is responsible.

Evaluating Classification

- Evaluation must be done on test data that are independent of the training data, i.e., training and test sets are disjoint.
- 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

The Conf



Actu

N)

F_1 Measure

- F1 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 F1 for aggregate TP, FP, FN

Takeaway Messages

- 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?