AUTOMATIC CLASSIFICATION VIA PROBABILISTIC MODELS:
THE NAÏVE BAYES

DEEP LEARNING 2024/25

R. Basili

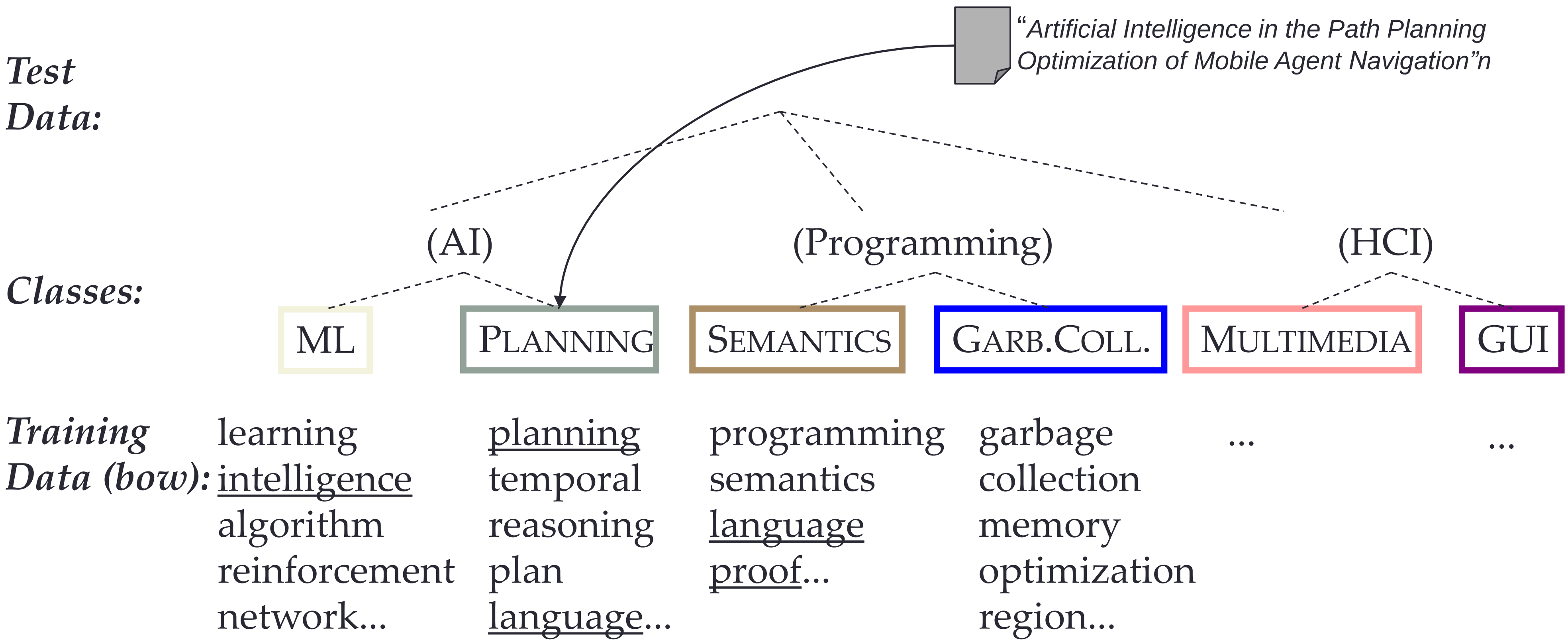
(*many slides from*: H. Schutze)

Università di Roma “Tor Vergata”
Email: basili@info.uniroma2.it

Agenda

- Document classification as a probabilistic inference
- Probabilistic Algorithms for Automatic Classification (AC)
  - Naive Bayes classification
  - Two models:
    - Univariate Binomial (FIRST UNIT)
    - Multinomial (Class Conditional Unigram Language Model) (SECOND UNIT)
- Some intuition on Parameter estimation
- The problem of Feature Selection
- Summary

Document Classification



(Note: in real life there is often a hierarchy; e.g. to Garb. Coll. with h(d) as a multiclassificatio function)

Text Categorization tasks: examples

- Labels are most often *topics* such as Yahoo-categories
  - e.g., "*finance*" "*sports*" "*news>world>asia>business*"
- Labels may be *genres*
  - e.g., "editorials" "movie-reviews" "news“
- Labels may be *opinion* (as in Sentiment Analysis)
  - e.g., “like”, “hate”, “neutral”
- Labels may be *domain-specific* **binary**
  - e.g., "interesting-to-me" : "not-interesting-to-me”,
    “spam” : “not-spam”,
    “contains adult language” :“doesn’t”,
    “is a fake” :“it isn’t”

Categorization/Classification

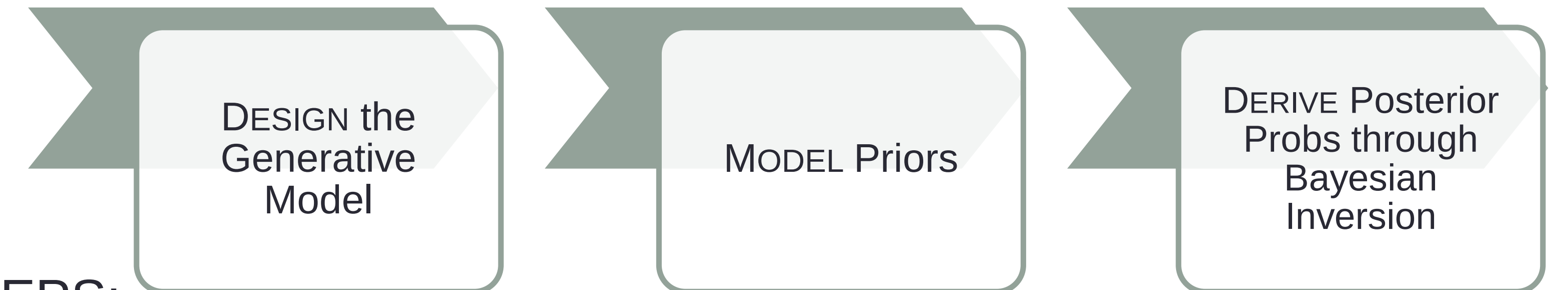
- Given:
  - A description of an instance, $x \in X$, where X is the *instance language* or *instance space*.
    - Issue: how to represent text documents.
  - A fixed set of categories:

    $C = \{c_1, c_2, \ldots, c_n\}$
- Determine:
  - The category of x: $h(x) \in C$ *(or 2^C),* where $h(x)$ is a *categorization function* whose domain is X that correspond to the classe(s) of (or subsets of the set) C, suitable for x.

- Learning problem:
  - We want to know how to build the categorization function *h* ("classifier").

Bayesian Methods

- Learning and classification methods based on probability theory:
  - Bayes theorem plays a critical role in probabilistic learning and classification.



- STEPS:
  - BUILD a generative model that approximates how data are produced
  - Use prior probability of each category when NO INFORMATION about an item is available.
  - PRODUCE, during categorization, the POSTERIOR PROBABILITY distribution over the possible categories given a description of an item

A document as a joint uncertain event

- In a relational DB, a tuple

  $$t=(t1, \ldots tn)$$

  is the **joint event** of the kind:

  $$(A1=t1 \wedge A2=t2 \wedge \ldots \wedge An=tn),$$

  where A*i* is the *i*-th attribute

A1	A2		Ak		An
t1	*t2*	…	*tk*	…	*tn*

- The probability of the tuple is the joint probability of all events, i.e.:

$$P(E_1 \wedge \ldots \wedge E_n) = P(A1=t1 \wedge A2=t2 \wedge \ldots \wedge An=tn)$$

A document as a joint uncertain event

- In a document, the basic event is (in line with the *bag-of-word* model) related to the occurrence of individual words
  - Notice how the *tf·idf* model itself is a probabilistic estimate
- Two modelling options for this estimate:
  - (**Dictionary oriented model**) The document *d* is a (random) selection of its words from the dictionary *V*, i.e. a binary choice over *V*
  - (**Document oriented model**) The document is a (random) selection of one word for each of its own positions, i.e. a multiple way choice from V for each position $W_1, \ldots, W_{|d|}$

Bayes' Rule

- Given an instance *X* and a category *C* the probability *P(C,X)* can be used as a joint event:

$$P(C,X) = P(C \mid X)P(X) = P(X \mid C)P(C)$$

- The following rule thus holds for every *X* and *C*:

$$\boxed{P(C \mid X) = \frac{P(X \mid C)P(C)}{P(X)}}$$

- What does *P(X|C)* means?

Maximum a posteriori Hypothesis

$$h_{MAP} \equiv \underset{h \in H}{\operatorname{argmax}}\, P(h \mid X)$$

$$= \underset{h \in H}{\operatorname{argmax}} \frac{P(X \mid h)P(h)}{P(X)} =$$

As *P(X)* is constant

$$= \underset{h \in H}{\operatorname{argmax}}\, P(X \mid h)P(h)$$

Maximum likelihood Hypothesis

If all hypotheses are a priori equally likely, we only need to consider the $P(D|h)$ term:

$$h_{ML} \equiv \underset{h \in H}{\operatorname{argmax}} P(X \mid h)$$

Naive Bayes Classifiers

Task: Classify a new instance document *D* based on a tuple of attribute values $D=(x_1, x_2, \ldots, x_n)$ into one of the classes $c_j \in C$

$$c_{MAP} = \operatorname{argmax}_{c_j \in C} P(c_j | x_1, x_2, \ldots, x_n) =$$

$$= \operatorname{argmax}_{c_j \in C} \frac{P(x_1, x_2, \ldots, xn | c_j) P(c_j)}{P(x_1, x_2, \ldots, xn)} =$$

$$= \operatorname{argmax}_{c_j \in C} P(x_1, x_2, \ldots, xn | cj) P(cj)$$

Problems to be solved to apply Bayes

- **Determine** the representation of documents as joint events

$$D=(x_1, x_2, \ldots, x_n)=(x^D_1, x^D_2, \ldots, x^D_n)$$

- **Determine** how x_i is related to the document content
- **Determine** how to estimate
  - $P(C_j)$ for the different classes $j=1, \ldots., k$
  - $P(x^D_i)$ for the different properties/features $i=1, \ldots, n$
  - $P(x^D_1, x^D_2, \ldots, x^D_n \mid C_j)$ for the different tuples and classes

- **Define the criteria to** select among the different classes

$$P(C_j \mid x^D_1, x^D_2, \ldots, x^D_n) \quad j=1, \ldots k$$

  - Argmax? Best *m* scores? Thresholds?

Problems to be solved to apply Bayes

- Determine the notion of document as the joint event

$$D=(x_1, x_2, \ldots, x_n)=(x^D_1, x^D_2, \ldots, x^D_n)$$

- Determine how x_i is related to the document content
- Determine how to estimate
  - $P(C_j)$ for the different classes $j=1, \ldots, k$
  - $P(x^D_i)$ for the different properties/features $i=1, \ldots, n$
  - $P(x^D_1, x^D_2, \ldots, x^D_n \mid C_j)$ for the different tuples and classes
- Define the law that select among the different

$$P(C_j \mid x^D_1, x^D_2, \ldots, x^D_n)\ j=1, \ldots k$$

  - Argmax? Best *m* scores? Thresholds?

Problems to be solved to apply Bayes

- Determine the notion of document as the joint event

$$D=(x_1, x_2, \ldots, x_n)=(x^D_1, x^D_2, \ldots, x^D_n)$$

- Determine how x_i is related to the document content
- IDEA: use words and their direct occurrences, as «signals» for the content
  - Words are individual outcomes of the test of picking randomly one token from the text
  - Random variables X can be used such that x_i represent $X=word_i$
  - Multiple Occurrences of words in texts trigger several successfu tests for the same word $word_i$; they augment the probability

$$P(x_i)=P(X=word_i)$$

Modeling the document content

- Variables *X* provide a description of a document *D* as they correspond to the outcome of a test
- *D* corresponds to the joint event of *one unique picking* of words $word_i$ from the vocabulary V, whose outcomes are
  - **Present** if $word_i$ occurrs in D
  - **Not present** if $word_i$ does not occur in D
- It is a *binary event*, like a picking a white or black ball from a urn
- The joint event is the «parallel» picking of the ball for every (urn, i.e.) $word_i$ in the dictionary, that is *one urn per word* is accessed

- Notice how *n* (i.e. the number of features) here becomes the size $|V|$ of the vocabulary *V*
- Each feature x_i models the presence or absence of $word_i$ in D, and can be written as $X_i=0$ or $X_i=1$

This is the basis for the so-called Multivariate binomial model!

Problems to be solved to apply Bayes

- Determine the notion of document as the joint event
  D=(x1, x2, …, xn)=(xD1, xD2, …, xDn)

- Determine how xi is related to the document content
- Determine how to estimate
  - $P(C_j)$ for the different classes $j=1, \ldots, k$
  - $P(x^D_i)$ for the different properties/features i=1, …, n
  - $P(x^D_1, x^D_2, \ldots, x^D_n \mid C_j)$ for the different tuples and classes

- Define the law that select among the different

  $$P(C_j \mid x^D_1, x^D_2, \ldots, x^D_n) \quad j=1, \ldots k$$

  - Argmax? Best *m* scores? Thresholds?

Naïve Bayes Classifier: Naïve Bayes Assumption

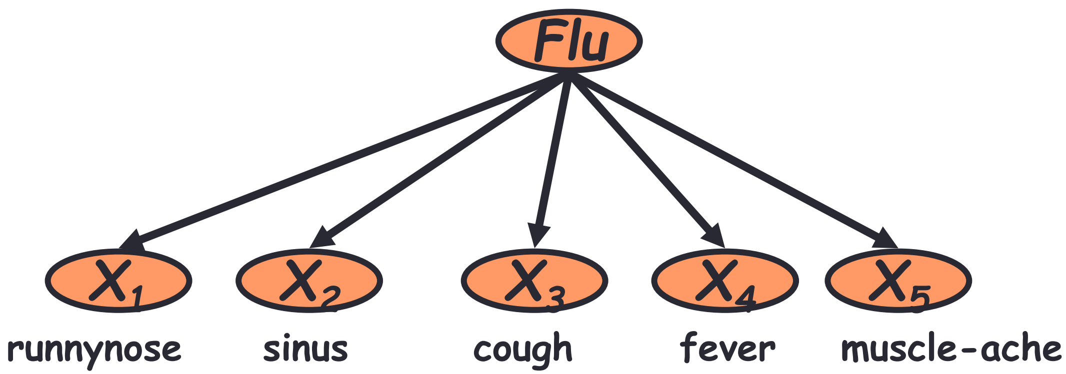
- $P(c_j)$
  - Can be estimated from the frequency of classes in the training examples.
- $P(x_1,x_2,\ldots,x_n|c_j)$
  - $O(|X|^n \bullet |C|)$ parameters
  - Could only be estimated if a very, very large number of training examples was available.

Naïve Bayes Conditional Independence Assumption:

Assume that the probability of observing the conjunction of attributes is equal to the product of the individual probabilities $P(x_i|c_j)$.

$\Longrightarrow$ $O(|X| \bullet |C|)$ parameters

The Naïve Bayes Classifier

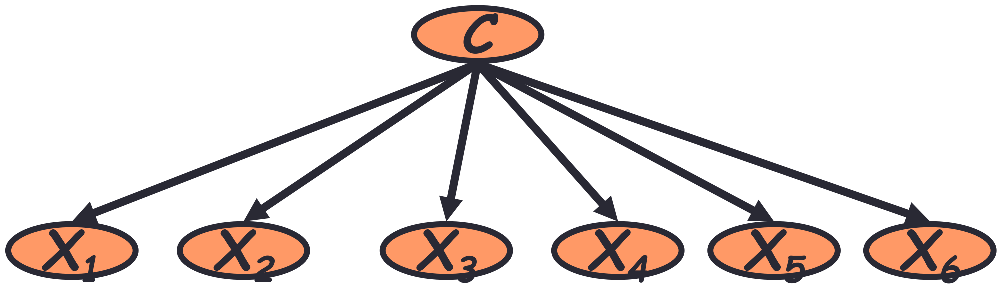


- **Conditional Independence Assumption:** features detect term **presence** and are independent of each other given the class:

$$P(X_1, \ldots, X_5 \mid C) = P(X_1 \mid C) \cdot P(X_2 \mid C) \cdot \ldots \cdot P(X_5 \mid C)$$

- This model is appropriate for binary variables
  - **Multivariate binomial model**

Learning the Model



- First attempt: maximum likelihood estimates
  - Simply use the occurrences (i.e. frequencies) in the data
  - Notation:
    - $N(C = c_j)$ is the set of documents of class c_j in the training set
    - $N(X_i = x_i, C = c_j)$ is the set of documents of class c_j where word X_i appears (i.e. $x_i = 1$) or does not appear (i.e. $x_i = 0$)

$$\hat{P}(c_j) = \frac{N(C = c_j)}{N}$$

$$\hat{P}(x_i|c_j) = \frac{N(X_i = x_i, C = c_j)}{N(C = c_j)}$$

NB Bernoulli: the Learning stage

```
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
```

Problems to be solved to apply Bayes

- Determine the notion of document as the joint event

  D=(x1, x2, …, xn)=(xD1, xD2, …, xDn)

- Determine how xi is related to the document content
- Determine how to estimate
  - $P(C_j)$ for the different classes $j=1, \ldots, k$
  - $P(x^D_i)$ for the different properties/features $i=1, \ldots, n$
  - $P(x^D_1, x^D_2, \ldots, x^D_n \mid C_j)$ for the different tuples and classes
- Define the law that select among the different

  $P(C_j \mid x^D_1, x^D_2, \ldots, x^D_n)$ j=1, …k

  - Argmax? Best *m* scores? Thresholds?

Problems to be solved to apply Bayes

- Define the law that selects among the different

$$P(C_j \mid x^D{}_1, x^D{}_2, \ldots, x^D{}_n)\ j=1, \ldots k$$

  - (A) Argmax? (B) Best *m* scores? (C) Thresholds?

A. ARGMAX is applicable for every task in which *multiclassification is not applicable*:
   - Spam/not spam
   - FAKE news detection

B. When a fixed number (n>1) of categories is requested seemingly the model output the ***n* most likely classes**

C. Thresholds can be imposed for more flexible behaviour, and they can be usually *estimated from the training data*

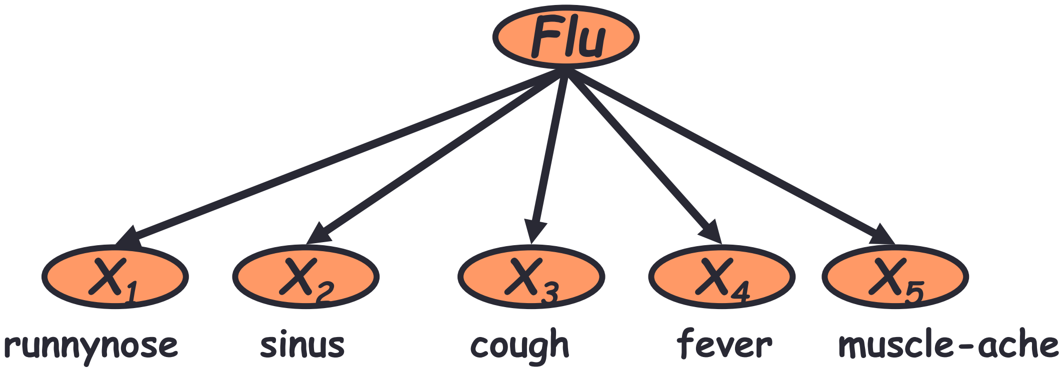
NB Bernoulli Model: Classification

- When multiclassification is not necessary:

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

$V_d \leftarrow$ ExtractTermsFromDoc(V, d)
for each $c \in \mathbb{C}$
do $score[c] \leftarrow \log prior[c]$
    for each $t \in V$
    do if $t \in V_d$
        then $score[c] \mathrel{+}= \log condprob[t][c]$
        else $score[c] \mathrel{+}= \log(1 - condprob[t][c])$
return $\arg\max_{c \in \mathbb{C}} score[c]$

Problem with Max Likelihood



$$P(X_1, \ldots, X_5 \mid C) = P(X_1 \mid C) \cdot P(X_2 \mid C) \cdot \ldots \cdot P(X_5 \mid C)$$

- What if we have seen no training cases where patient had *no flu* and *muscle aches*?

$$\hat{P}(X_5 = true \mid C = no\_flu) = \frac{N(X_5 = true, C = no\_flue)}{N(C = nf)} = 0$$

- Zero probabilities cannot be conditioned away, no matter the other evidence!

$$\widehat{\mathrm{argmax}}_c\, P\,(c) \prod_i \hat{P}(x_i|c)$$

A digression: Estimation via smoothing

- *Laplace smoothing*
  - every feature has an a priori probability *p*,
  - It is assumed that it has been observed in a number of *m* virtual examples.

$$P(x_j \mid c_i) = \frac{n_{ij} + mp}{n_i + m}$$

- Usually:
  - A uniform distrbution on all words is assumed so that $p = 1/|V|$ and $m = |V|$
  - It is equivalent to observing every word in the dictionary once for each category.

Bayesian Classification: an alternative view

- Is there any alternative way of looking to the *joint* event $C \wedge D$?

- In the **Bernoulli model,** we determine the occurrence the event D as a instantaneous selection of individual words w_j from the Vocabulary V
  - Every D is a subset of V, thus characterized by a binary string across the entire V
  - There are as many binary strings as $2^{|V|}$

- An alternative consists in modelling the event D as the occurrence of some words w_j in m distinct positions, where m is $|D|$, i.e. the size of the document
- This brings to map a document D into a sequence of words $(w_1, \ldots., w_m)$ from V, i.e. *strings of words*
- The resulting model is called **Multinomial model** as every positions corresponds to a different stochastic variable

A digression: Stochastic Language Models

- Models *probability* of generating strings in the language (commonly all strings over an alphabet ∑), e.g., unigram model

Model M

0.2	the
0.1	a
0.01	man
0.01	woman
0.03	said
0.02	likes
…	

the	man	likes	the	woman
0.2	0.01	0.02	0.2	0.01

multiply

P(s | M) = 0.00000008

Stochastic Language Models

- Model *probability* of generating any string

Model M1

0.2	the
0.01	class
0.0001	sayst
0.0001	pleaseth
0.0001	yon
0.0005	maiden
0.01	woman

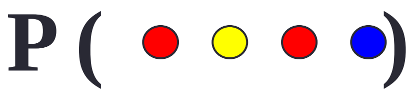
Model M2

0.2	the
0.0001	class
0.03	sayst
0.02	pleaseth
0.1	yon
0.01	maiden
0.0001	woman

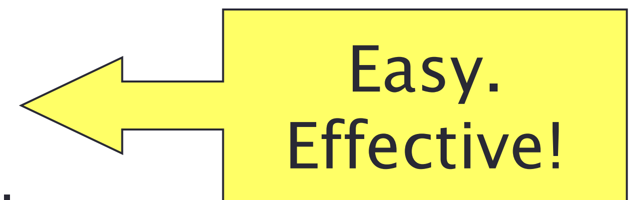
the	class	pleaseth	yon	maiden
0.2	0.01	0.0001	0.0001	0.0005
0.2	0.0001	0.02	0.1	0.01

$$P(s|M2) > P(s|M1)$$

Unigram and higher-order models



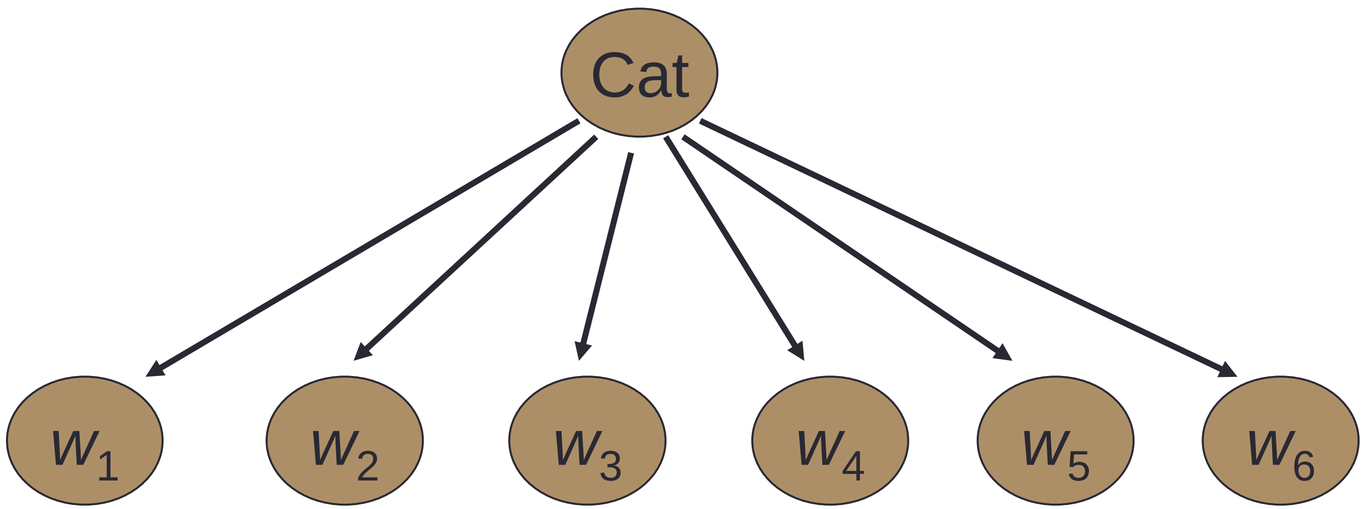
- 
- Unigram Language Models



- Bigram (generally, n-gram) Language Models

- Other Language Models
  - Grammar-based models (such as Probabilistic Context Free Grammars, PCFG), etc.
    - Probably not the first thing to try in IR

Naïve Bayes via a class conditional language model = multinomial NB



- Effectively, the probability of each class is done as a class-specific unigram language model

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_i P(x_i \mid c_j)$$
$$= \underset{c_j \in C}{\operatorname{argmax}} P(c_j) P(x_1 = \text{"our"} \mid c_j) \ldots P(x_n = \text{"text"} \mid c_j)$$

- Still too many possibilities
- Assume that classification is *independent* of the positions of the words
  - Use same parameters for each position
  - Result is bag of words model (over tokens not types)

Multinomial Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(x_k \mid c_j)$ terms
  - For each c_j in C do
    - $docs_j \leftarrow$ subset of documents for which the target class is c_j
    - $$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

  - $Text_j \leftarrow$ single document containing all $docs_j$
  - for each word x_k in *Vocabulary*
    - $n_k \leftarrow$ number of occurrences of x_k in $Text_j$
    - $$P(x_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Multinomial Naïve Bayes: Classifying

- positions $\leftarrow$ all word positions in current document which contain tokens found in *Vocabulary*
- Return c_{NB}, where

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} P(c_j) \prod_{i \in positions} P(x_i \mid c_j)$$

Naive Bayes: Time Complexity

- Training Time: $O(|D|L_d + |C||V|))$
  where L_d is the average length of a document in D.
  - Assumes V and all D_i , n_i, and n_{ij} pre-computed in $O(|D|L_d)$ time during one pass through all of the data.
  - Generally just $O(|D|L_d)$ since usually $|C||V| < |D|L_d$
- Test Time: $O(|C| L_t)$
  where L_t is the average length of a test document.
  - Very efficient overall, linearly proportional to the time needed to just read in all the data.

Multinomial NB: Learning Algorithm

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 $V, prior, condprob$

Multinomial NB: Classification Algorithm

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

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

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

do $score[c] \leftarrow \log prior[c]$

    for each $t \in W$

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

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

Underflow Prevention

- Multiplying lots of probabilities, which are between 0 and 1 by definition, can result in floating-point underflow.
- Since log(xy) = log(x) + log(y), it is better to perform all computations by summing logs of probabilities rather than multiplying probabilities.
- Class with highest final un-normalized log probability score is still the most probable.

$$c_{NB} = \operatorname*{argmax}_{c_j \in C} \log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j)$$

Note on the two models

- Model 1: Multivariate binomial
  - One feature X_w for each word in dictionary
  - $X_w = \mathit{true}$ in document *d* if *w* appears in *d*
  - Naive Bayes assumption:
    - Given the document's topic, appearance of one word in the document tells us nothing about chances that another word appears
- This is the model used in the binary independence model in classic probabilistic relevance feedback in hand-classified data (Maron in IR was a very early user of NB)

Note: the two models (2)

- Model 2: Multinomial = Class conditional unigram
  - One feature X_i for each word pos in document
    - feature's values are all words in dictionary
  - Value of X_i is the word in position *i*
  - Naïve Bayes assumption:
    - Given the document's topic, word in one position in the document tells us nothing about words in other positions
  - Second assumption:
    - Word appearance does not depend on position

$$P(X_i = w \mid c) = P(X_j = w \mid c)$$

for all positions *i,j*, word *w*, and class *c*

- Just have one multinomial feature predicting all words

Parameter estimation

- Binomial model:

$$\hat{P}(X_w = true \mid c_j) =$$

fraction of documents of topic c_j in which word *w* appears

- Multinomial model:

$$\hat{P}(X_i = w \mid c_j) =$$

fraction of times in which word *w* appears across all documents of topic c_j

  - Can create a mega-document for topic *j* by concatenating all documents in this topic
  - Use frequency of *w* in mega-document

Classification

- Multinomial vs Multivariate binomial?
  - Multinomial is in general better
    - See results figures later

NB example

- Given: 4 documents
  - D1 (SPORTS): China soccer
  - D2 (SPORTS): Japan baseball
  - D3 (POLITICS): China trade
  - D4 (POLITICS): Japan Japan exports
- Classify:
  - D5: soccer
  - D6: Japan
- Use
  - Add-one smoothing
  - Multinomial model
  - Multivariate binomial model

NB example

- p(SPORTS)=0.5
- p(POLITICS)=0.5
- V = {China, soccer, baseball, Japan, trade, exports}

Multivariate Binomial

p(China|SPORTS)=1/2 (or better(1+1)/(2+2))

p(soccer|SPORTS)=(1+1)/(2+2)

...

p(exports|SPORTS)=(0+1)/(2+2)

p(China|POLITICS)=(1+1)/(2+2)

p(soccer|POLITICS)=(0+1)/(2+2)

...

p(exports|POLITICS)=(1+1)/(2+2)

p(SPORTS|D5) ca =
p(D5|SPORTS)p(SPORTS) =
(1-p(China|SPORTS))p(soccer|SPORTS) .... (1-p(exports|SPORTS)).p(SPORTS)=
1/2*1/2* ... *(1-1/4)*(0.5)

p(POLITICS|D5) ca =
p(D5| POLITICS)p(POLITICS) =
(1-p(China|POLITICS))p(soccer|POLITICS) .... (1-p(exports|POLITICS))=
1/2*1/4* ... *(1-1/2)*(0.5)

da cui p(POLITICS|D5) < p(SPORTS|D5), e quindi:
D5 $\in$ SPORTS AND D5 $\notin$ POLITICS

Multinomial NB

Again:
V = {China, soccer, baseball, Japan, trade, exports}

p(SPORTS)=0.5
p(POLITICS)=0.5

p(China|SPORTS)=(1+1)/(4+2)
p(soccer|SPORTS)=(1+1)/(4+2)
...
p(exports|SPORTS)=(0+1)/(4+2)
p(China|POLITICS)=(1+1)/(5+2)
p(soccer|POLITICS)=(0+1)/(5+2)
...
p(exports|POLITICS)=(1+1)/(5+2)

p(SPORTS|D5)= ca
= p(D5|SPORTS)p(SPORTS)=p(soccer|SPORTS)p(SPORTS)=1/6

p(POLITICS|D5)= ca
p(D5|POLITICS)p(POLITICS)=p(soccer|POLITICS)p(POLITICS)=
= (1/7)*(1/2) = 1/14

da cui p(POLITICS|D5) < p(SPORTS|D5), e quindi:
D5 $\in$ SPORTS AND D5 $\notin$ POLITICS

Feature Selection: Why?

- Text collections have a large number of features
  - 10,000 – 1,000,000 unique words … and more
- Feature Selection:
  - is **the process by which a large set of available features are neglected during the classification**
  - Not reliable, not well estimated, not useful
- May make using a particular classifier feasible, e.g. reduce the training time
  - Some classifiers can't deal with 100,000 of features
  - Training time for some methods is quadratic or worse in the number of features
- Can improve generalization (performance)
  - Eliminates noise features+ Avoids overfitting

Feature selection: how?

- Two idea:
  - Hypothesis testing statistics:
    - Are we confident that the value of one categorical variable is associated with the value of another?
    - *Chi-square test*
  - Information theory:
    - How much information does the value of one categorical variable give you about the value of another?
    - Mutual information

- They're similar, but χ^2 measures confidence in association, (based on available statistics), while MI measures extent of association (assuming perfect knowledge of probabilities)

Feature selection via Mutual Information

- In training set, choose *k* words which best discriminate (give most info on) the categories.
- The Mutual Information between a word *w* and a class *c* is:

$$I(w,c) = \sum_{e_w \in \{0,1\}} \sum_{e_c \in \{0,1\}} p(e_w, e_c) \log \frac{p(e_w, e_c)}{p(e_w)\,p(e_c)}$$

  - For each word *w* and each category *c*

Feature selection via Mutual Information

- In training set, choose *k* words which best discriminate (give most info on) the categories.
- The Mutual Information between a word *w* and a class *c* is:

$$I(W=w, C=c) = \sum_{\substack{W=w \\ W\neq w}} \sum_{\substack{C=c \\ C\neq c}} p(W,C) \log \frac{p(W,C)}{p(W)\,p(C)}$$

  - For each word *w* and each category *c*

Pointwise Mutual Information

- Instead of looking to the entire distribution of *W* and *C* we can estimate MI on specific word-category pairs (w_i, c_j), so that the only quantity tha can be used to rank features/words w_i by importance in modeling one (or more categories) is:

$$pmi(w_i, c_j) = log_2 \frac{p(w_i, c_j)}{p(w_i)p(c_j)}$$

- Given the set of the terms w that appear in a document in the training set and that are mostly associated to a given category c_j, i.e.

$$W_j = \{ w \mid pmi(w, c_j) > \tau_j \}$$

then the final Vocabulary V for the training data set is:

$$V = \bigcup_j W_j$$

Feature selection via MI (contd.)

- For each category we build a list of *k* most discriminating terms.
- For example (on 20 Newsgroups):
  - ***sci.electronics:*** circuit, voltage, amp, ground, copy, battery, electronics, cooling, …
  - ***rec.autos:*** car, cars, engine, ford, dealer, mustang, oil, collision, autos, tires, toyota, …
- Greedy: does not account for correlations between terms
- Why?

Feature Selection

- Mutual Information
  - Clear information-theoretic interpretation
  - May select rare uninformative terms
- Chi-square
  - Statistical foundation
  - May select very slightly informative frequent terms that are not very useful for classification

- Just use the commonest terms?
  - No particular foundation
  - In practice, this is often 90% as good

Feature selection for NB

- In general feature selection is *necessary* for binomial NB.
- Otherwise you suffer from noise, multi-counting

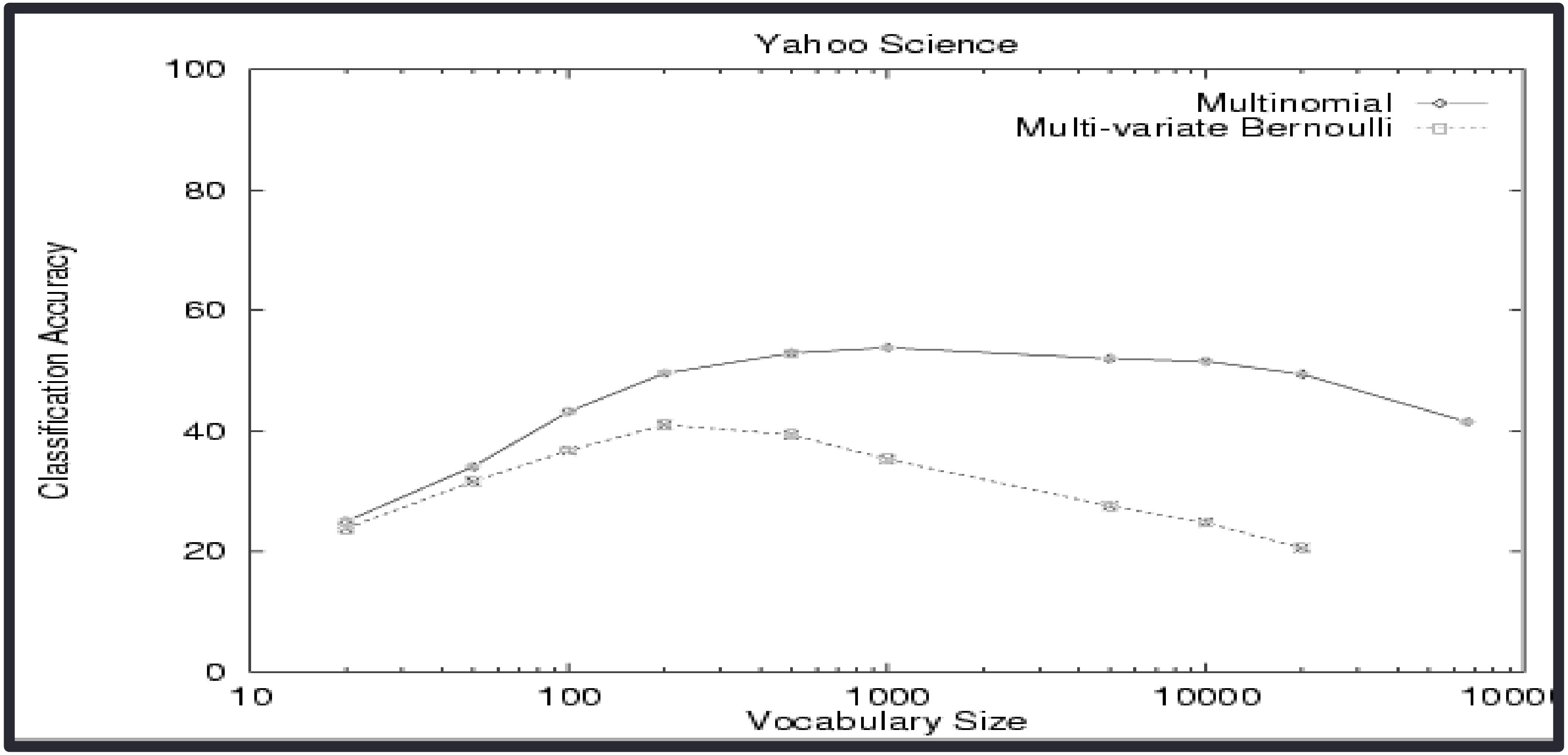
- “Feature selection” really means something different for multinomial NB. It means dictionary truncation
  - The multinomial NB model only has 1 feature
- This “feature selection” normally isn’t needed for multinomial NB, but may help a fraction with quantities that are badly estimated

Evaluating Categorization

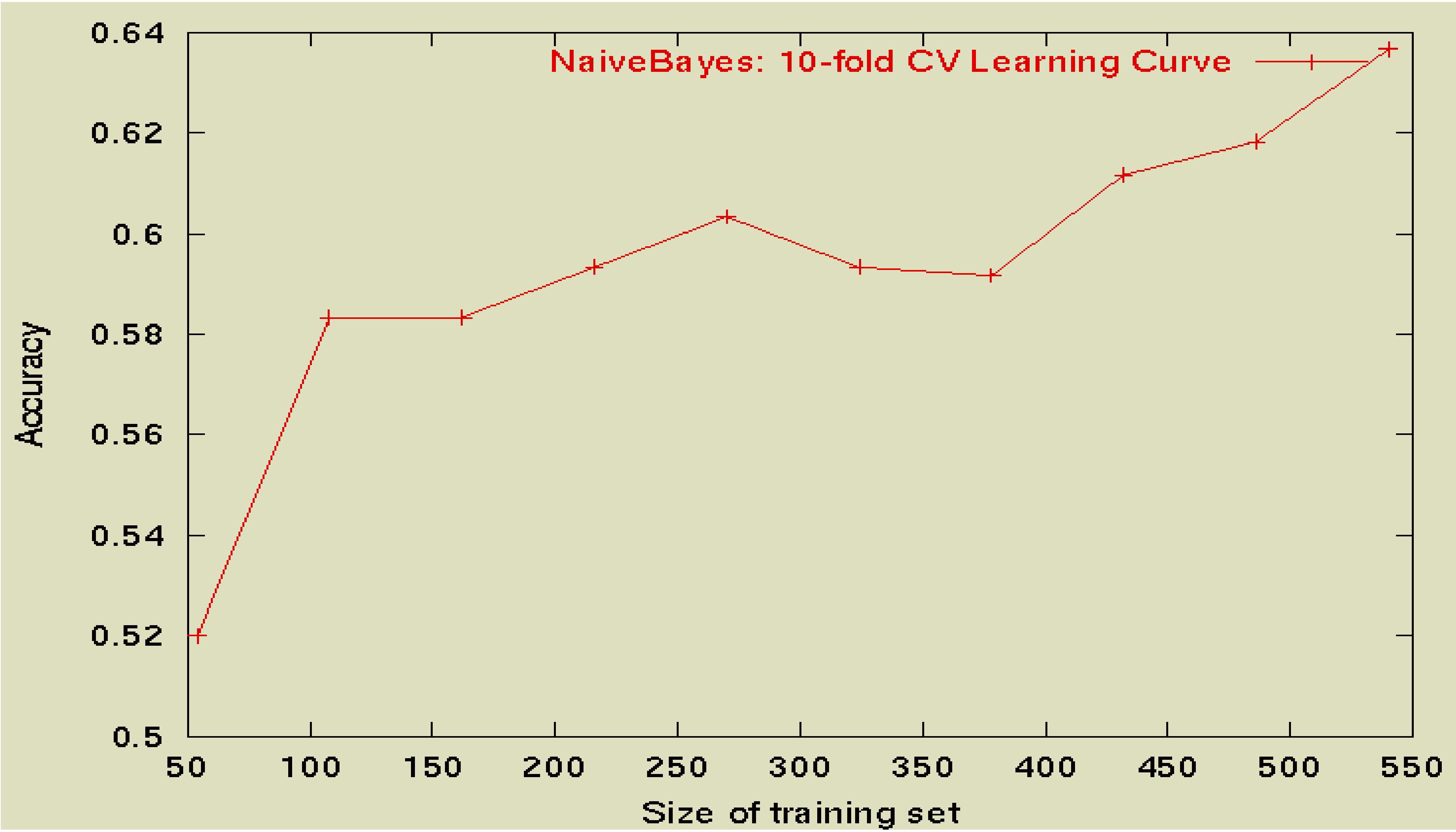
- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- *Classification accuracy*: c/n where n is the total number of test instances and c is the number of test instances correctly classified by the system.
- Results can vary based on sampling error due to different training and test sets.
- Average results over multiple training and test sets (splits of the overall data) for the best results.

Example: AutoYahoo!

- Classify 13,589 Yahoo! webpages in “Science” subtree into 95 different topics (hierarchy depth 2)



Sample Learning Curve
(Yahoo Science Data): need more!



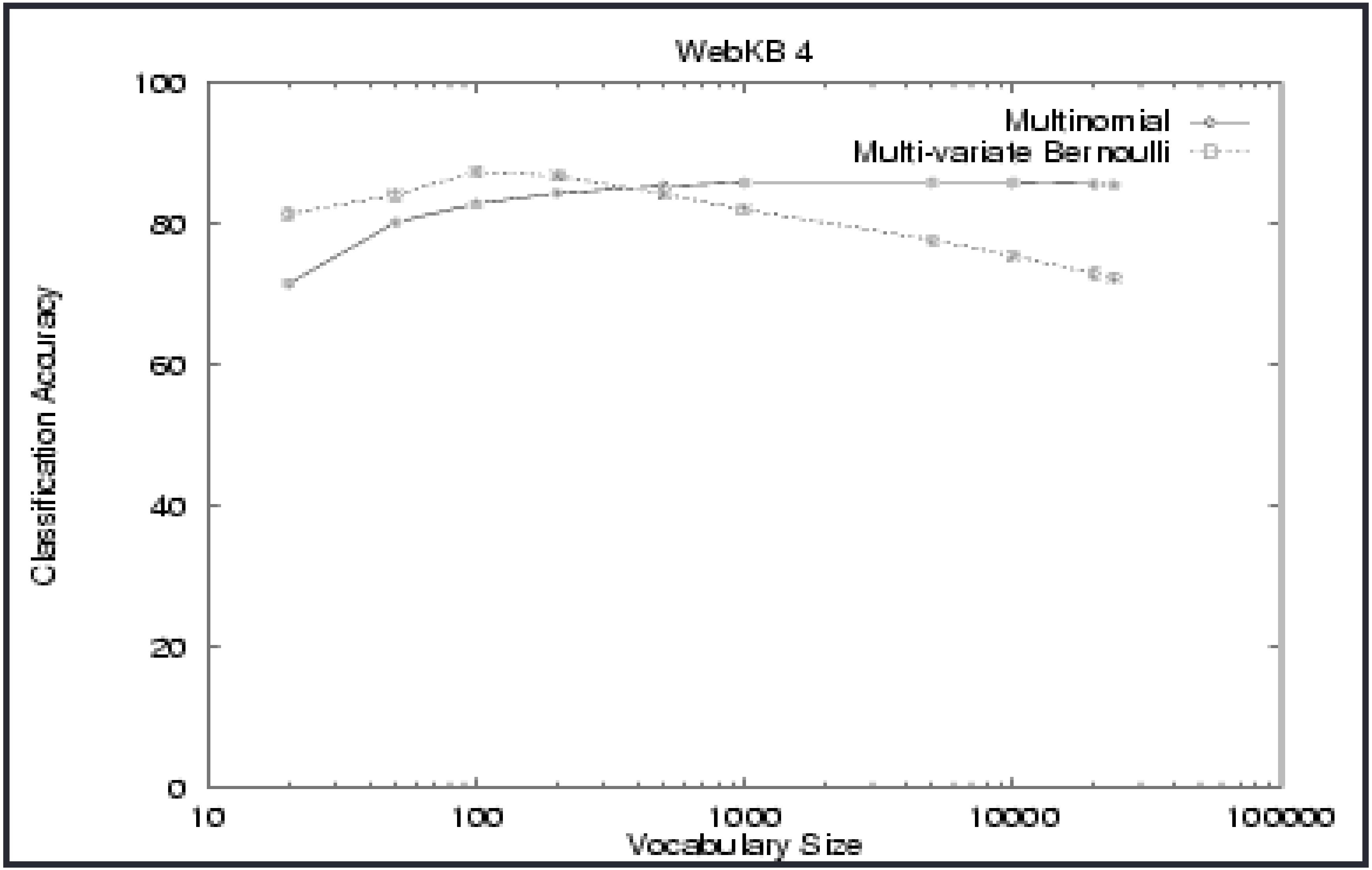
WebKB Experiment

- Classify webpages from CS departments into:
  - student, faculty, course,project
- Train on ~5,000 hand-labeled web pages
  - Cornell, Washington, U.Texas, Wisconsin
- Crawl and classify a new site (CMU)

- Results:

	Student	Faculty	Person	Project	Course	Departmt
Extracted	180	66	246	99	28	1
Correct	130	28	194	72	25	1
Accuracy:	72%	42%	79%	73%	89%	100%

NB Model Comparison



Faculty

associate	0.00417
chair	0.00303
member	0.00288
ph	0.00287
director	0.00282
fax	0.00279
journal	0.00271
recent	0.00260
received	0.00258
award	0.00250

Students

resume	0.00516
advisor	0.00456
student	0.00387
working	0.00361
stuff	0.00359
links	0.00355
homepage	0.00345
interests	0.00332
personal	0.00332
favorite	0.00310

Courses

homework	0.00413
syllabus	0.00399
assignments	0.00388
exam	0.00385
grading	0.00381
midterm	0.00374
pm	0.00371
instructor	0.00370
due	0.00364
final	0.00355

Departments

departmental	0.01246
colloquia	0.01076
epartment	0.01045
seminars	0.00997
schedules	0.00879
webmaster	0.00879
events	0.00826
facilities	0.00807
eople	0.00772
postgraduate	0.00764

Research Projects

investigators	0.00256
group	0.00250
members	0.00242
researchers	0.00241
laboratory	0.00238
develop	0.00201
related	0.00200
arpa	0.00187
affiliated	0.00184
project	0.00183

Others

type	0.00164
jan	0.00148
enter	0.00145
random	0.00142
program	0.00136
net	0.00128
time	0.00128
format	0.00124
access	0.00117
begin	0.00116

Faculty

associate	0.00417
chair	0.00303
member	0.00288
ph	0.00287
director	0.00282
fax	0.00279
journal	0.00271
recent	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
webmaster	[illegible]
events	[illegible]
facilities	0.00807
eople	0.00772
postgraduate	0.00764

Students

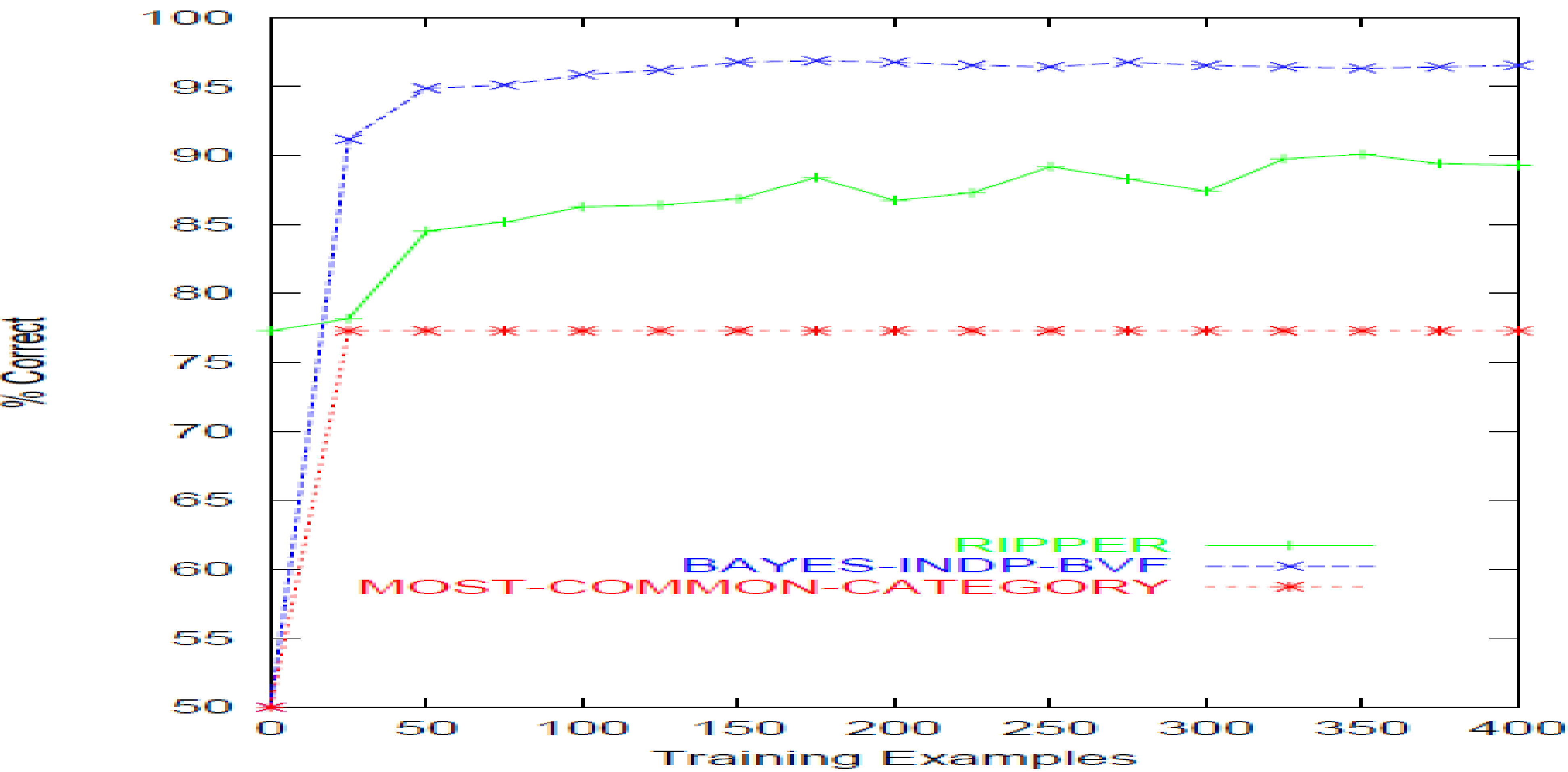
resume	0.00516
advisor	0.00456
student	0.00387
working	0.00361
stuff	0.00359
links	0.00355
[illegible]	[illegible]
arpa	0.00187
affiliated	0.00184
project	0.00183

Courses

homework	0.00413
syllabus	0.00399
assignments	0.00388
exam	0.00385
grading	0.00381
midterm	0.00374
[illegible]	0.00371
[illegible]	0.00370
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	0.00128
[illegible]	0.00128
format	0.00124
access	0.00117
begin	0.00116

These feature sets correspond to domain dictionaries. Specific terms such as *chair* or *director* are selected for the individual classes such as `Faculty` as well as *advisor* for `Student`: it is a form of “knowledge” emerging automatically from annotated data

Naïve Bayes on spam email



Violation of NB Assumptions

- Conditional independence
- “Positional independence”

- Examples?
  - *Computer* vs. *science* in the **`Technology`** category
  - *par* vs. *condition* in the **`Law, Politics`** category

  - *Box office* vs. *Office Box*
  - *Taxonomy tree* vs. *Tree taxonomy*
  - *(Dog eats* vs. *eating dogs)* vs. *(Eating vegetables* vs. *vegetables eat)*

When does Naive Bayes work?

- Sometimes NB performs well even if the Conditional Independence assumptions are badly violated.
- Classification is about predicting the correct class label and NOT about accurately estimating probabilities.

Assume two classes c_1 and c_2.
A new case A arrives.

NB will classify A to c_1 if:

$$P(A, c_1) > P(A, c_2)$$

	$P(A,c_1)$	$P(A,c_2)$	Class of A
Actual Probability	0.1	0.01	c_1
Estimated Probability by NB	0.08	0.07	c_1

Besides the big error in estimating the probabilities the classification is still correct.

Correct estimation $\Rightarrow$ accurate prediction

but NOT

accurate prediction $\not\Rightarrow$ Correct estimation

Naive Bayes is *not-so*-Naive

- Naïve Bayes: First and Second place in KDD-CUP 97 competition, among 16 (then) state of the art algorithms
  - Goal: Financial services industry direct mail response prediction model: Predict if the recipient of mail will actually respond to the advertisement – 750,000 records.
- Robust to Irrelevant Features
  - Irrelevant Features cancel each other without affecting results
  - Instead Decision Trees can heavily suffer from this.
- Very good in domains with many <u>equally important</u> features
  - Decision Trees suffer from *fragmentation* in such cases – especially if little data
- A good dependable baseline for text classification (but not the best)!
- Optimal if the Independence Assumptions hold: If assumed independence is correct, then it is the Bayes Optimal Classifier for problem
- Very Fast: Learning with one pass over the data; testing linear in the number of attributes, and document collection size
- Low Storage requirements

Resources

- Fabrizio Sebastiani. *Machine Learning in Automated Text Categorization*. ACM Computing Surveys, 34(1):1-47, 2002. (http://faure.iei.pi.cnr.it/~fabrizio/Publications/ACMCS01/ACMCS01.pdf)
- Andrew McCallum and Kamal Nigam. *A Comparison of Event Models for Naive Bayes Text Classification*. In AAAI/ICML-98 Workshop on Learning for Text Categorization, pp. 41-48.
- Tom Mitchell, *Machine Learning*. McGraw-Hill, 1997.
  - Clear simple explanation
- Yiming Yang & Xin Liu, *A re-examination of text categorization methods*. Proceedings of SIGIR, 1999.

Summary

- A general type of learning is the probabilistic one. Learning here means
  - Describe the problem through a generative model that makes the relations between input (e.g. symptoms) and output variables (e.g. diagnoses) explicit
  - Find the best parameters for the model (i.e. analytical probability distributions or estimation of discrete probabilities) able to decide about the problem in an accurate way
- An example: NB document classifiction (discrete case)
- Most applied models:
  - Multivariate Binomial (o Bernoulli) NB
  - Multinomial NB

Summary (2)

- In estimating the parameters of a NB classifiers a cengtral role isplayed by the so-called *smoothing* techniques: inaccurate estimation processes may result in a poor, i.e. inaccurate, results
  - Smoothing allows to improve the estimate of some parameters that are particularly problematic
    - Some target phenomena (e.g. very rare words)
    - Structural lacks on the adopted annotated sample

- NB classification is to be preferred for its robustness and efficiency
- It is widely adopted as a baseline in several researches and applications