

Part 9: Text Classification; The Naïve Bayes algorithm



Francesco Ricci

Most of these slides comes from the
course:

Information Retrieval and Web Search,
Christopher Manning and Prabhakar
Raghavan

Content

- Introduction to Text Classification
- Bayes rule
- Naïve Bayes text classification
- Feature independence assumption
- Multivariate and Multinomial approaches
- Smoothing (avoid overfitting)
- Feature selection
 - Chi square and Mutual Information
- Evaluating NB classification.

Relevance feedback revisited

- In relevance feedback, the user marks a number of documents as relevant/nonrelevant
- We then try to use this information to return better search results
- Suppose we just tried to learn a filter for nonrelevant documents
- This is an instance of a **text classification** problem:
 - Two “classes”: **relevant, nonrelevant**
 - For each document, decide whether it is relevant or nonrelevant
- The notion of **classification** is very general and has many applications within and beyond information retrieval.

Standing queries

- The path from information retrieval to text classification:
 - You have an information need, say:
 - "Unrest in the Niger delta region"
 - You want to rerun an appropriate query periodically to find new news items on this topic
 - You will be sent new documents that are found
 - I.e., it's classification not ranking
- Such queries are called ***standing queries***
 - Long used by "information professionals"
 - A modern mass instantiation is **Google Alerts.**

Google alerts



[FAQ](#) | [Sign in](#)

Welcome to Google Alerts

Google Alerts are email updates of the latest relevant Google results (web, news, etc.) based on your choice of query or topic.

Some handy uses of Google Alerts include:

- monitoring a developing news story
- keeping current on a competitor or industry
- getting the latest on a celebrity or event
- keeping tabs on your favorite sports teams

Create an alert with the form on the right.

You can also [sign in to manage your alerts](#)

Create a Google Alert

Enter the topic you wish to monitor.

Search terms:

Type:

How often:

Email length:

Your email:

Google will not sell or share your email address.

Spam filtering: Another text classification task

From: "" <takworld@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>

=====

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 classes:

$$C = \{c_1, c_2, \dots, c_J\}$$

- Determine:

- The category of x : $c(x) \in C$, where $c(x)$ is a *classification function* whose domain is X and whose range is C

- We want to know how to build classification functions ("classifiers").

Document Classification

Test Data:



“planning
language
proof
intelligence”

Classes:



Training Data:

learning	<u>planning</u>	programming	garbage
<u>intelligence</u>	temporal	semantics	collection		
algorithm	reasoning	<u>language</u>	memory		
reinforcement	plan	<u>proof...</u>	optimization		
network...	<u>language...</u>		region...		

(Note: in real life there is often a hierarchy, not present in the above problem statement; and also, you get papers on "ML approaches to Garb. Coll.")

More Text Classification Examples

- ❑ Many search engine functionalities use classification
- ❑ Assign labels to each document or web-page:
- ❑ 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 on a person/product
e.g., "like", "hate", "neutral"
- ❑ Labels may be domain-specific
e.g., "interesting-to-me" : "not-interesting-to-me"
e.g., "contains adult language" : "doesn't"
e.g., language identification: English, French, Chinese, ...
e.g., search vertical: about Linux versus not
e.g., "link spam" : "not link spam"

Classification Methods (1)

□ Manual classification

- Used by Yahoo! (originally; now present but downplayed), Looksmart, about.com, ODP, PubMed
- Very accurate when job is done by experts
- Consistent when the problem size and team is small
- Difficult and expensive to scale
 - Means we need automatic classification methods for big problems.

Classification Methods (2)

□ Hand-coded rule-based systems

- One technique used by CS dept's spam filter, Reuters, CIA, etc.
- Companies (Verity) provide "IDE" for writing such rules
- E.g., assign category if document contains a given boolean combination of words
- Standing queries: Commercial systems have complex query languages (everything in IR query languages + accumulators)
- Accuracy is often very high if a rule has been carefully refined over time by a subject expert
- Building and maintaining these rules is 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"
subtopic topic    * 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
```

□ Note:

- maintenance issues (author, etc.)
- Hand-weighting of terms

Classification Methods (3)

- **Supervised learning** of a document-label assignment function
 - Many systems partly rely on machine learning (Autonomy, MSN, Verity, Enkata, Yahoo!, ...)
 - k-Nearest Neighbors (simple, powerful)
 - Naive Bayes (simple, common method)
 - Support-vector machines (new, more powerful)
 - ... plus many other methods
 - No free lunch: **requires hand-classified training data**
 - But data can be built up (and refined) by amateurs
- Note that many commercial systems use a mixture of methods.

Recall a few probability basics

□ For events a and b :

□ **Bayes' Rule**

$$p(a, b) = p(a \cap b) = p(a | b)p(b) = p(b | a)p(a)$$

$$p(\bar{a} | b)p(b) = p(b | \bar{a})p(\bar{a})$$

$$p(a | b) = \frac{p(b | a)p(a)}{p(b)} = \frac{p(b | a)p(a)}{\sum_{x=a, \bar{a}} p(b | x)p(x)}$$

Posterior

Prior

□ **Odds:**

$$O(a) = \frac{p(a)}{p(\bar{a})} = \frac{p(a)}{1 - p(a)}$$

Bayesian Methods

- Our focus this lecture
- Learning and classification methods based on probability theory
- Bayes theorem plays a critical role in probabilistic learning and classification
- Uses *prior* probability of each category given no information about an item
- Categorization produces a *posterior* probability distribution over the possible categories given a description of an item.

Bayes' Rule

$$P(C, D) = P(C | D)P(D) = P(D | C)P(C)$$

$$P(C | D) = \frac{P(D | C)P(C)}{P(D)}$$

Naive Bayes Classifiers

- Task: Classify a new instance D based on a tuple of attribute values $D = \langle x_1, x_2, \dots, x_n \rangle$ into one of the classes $c_j \in C$

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

Maximum A
Posteriori
class

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

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

Naïve Bayes Assumption

- $P(c_j)$
 - Can be estimated from the frequency of classes in the training examples
- $P(x_1, x_2, \dots, x_n | c_j)$
 - $O(|X|^n \cdot |C|)$ parameters (*assuming X finite*)
 - 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 $P(x_i | c_j)$.

The Naïve Bayes Classifier



- **Conditional Independence Assumption:**

features detect term presence and are independent of each other given the class:

$$P(X_1, \dots, X_5 | C) = P(X_1 | C) \cdot P(X_2 | C) \cdot \dots \cdot P(X_5 | C)$$

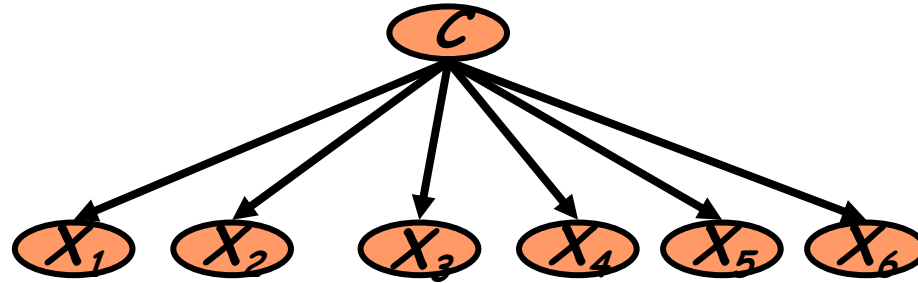
- This model is appropriate for **binary** variables

- Multivariate Bernoulli model

= many variables

= only 2 values

Learning the Model

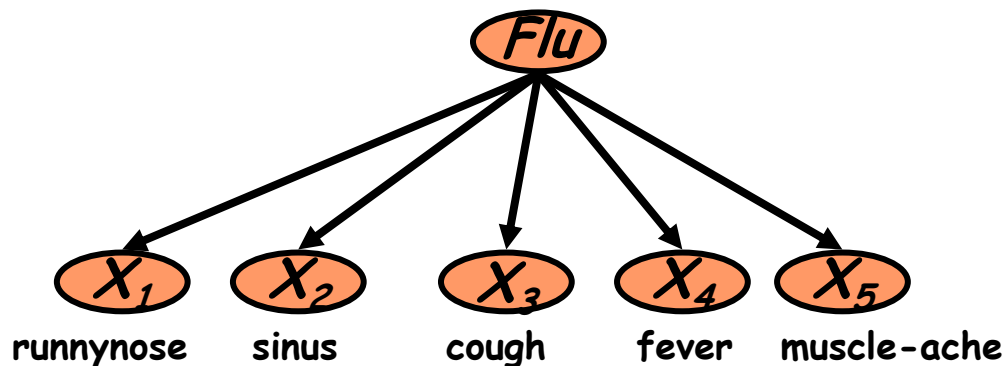


- First attempt: maximum likelihood estimates
 - simply use the frequencies in the data

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

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

Problem with Max Likelihood



$$P(X_1, \dots, X_5 | C) = P(X_1 | C) \cdot P(X_2 | C) \cdot \dots \cdot P(X_5 | C)$$

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

$$\hat{P}(X_5 = t | C = nf) = \frac{N(X_5 = t, C = nf)}{N(C = nf)} = 0$$

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

$$\ell = \arg \max_c \hat{P}(c) \prod_i \hat{P}(x_i | c)$$

Smoothing to Avoid Overfitting

$$\hat{P}(x_i | c_j) = \frac{N(X_i = x_i, C = c_j) + 1}{N(C = c_j) + k}$$

of values of X_i

- Somewhat more subtle version

overall fraction in data where $X_i = x_{i,k}$

$$\hat{P}(x_{i,k} | c_j) = \frac{N(X_i = x_{i,k}, C = c_j) + mp_{i,k}}{N(C = c_j) + m}$$

extent of "smoothing"

Multinomial Naive Bayes Classifiers: Basic method

- Attributes (X_i) are text positions, values (x_i) are words:

$$\begin{aligned}c_{NB} &= \operatorname{argmax}_{c_j \in C} P(c_j) \prod_i P(x_i | c_j) \\ &= \operatorname{argmax}_{c_j \in C} P(c_j) P(X_1 = \text{"our"} | c_j) \cdots P(X_n = \text{"text"} | c_j)\end{aligned}$$

- Still too many possibilities
- Assume that classification is *independent* of the positions of the words

$$P(X_i = w | c) = P(X_j = w | c)$$

Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*
- Calculate required $P(c_j)$ and $P(x_k | c_j)$ terms
 - For each class 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*
 - $n_{jk} \leftarrow$ number of occurrences of x_k in $Text_j$
 - $n_j \leftarrow$ number of words in $Text_j$

$$P(x_k | c_j) \leftarrow \frac{n_{jk} + \alpha}{n_j + \alpha |Vocabulary|}$$

Assume $\alpha = 1$;
this is for
smoothing

Naïve Bayes: Classifying

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

$$c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{i \in \text{positions}} P(x_i | c_j)$$

Naive Bayes: Time Complexity

- **Training Time:** if L_d is the average length of a document in D
 - $O(|D|L_d + |C||V|)$
 - Scan the documents to compute the vocabulary and the frequencies of words
 - Assumes that V and all $docs_j$, n_j , and n_{jk} are 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.

Number of conditional probabilities to estimate

Underflow Prevention: log space

- ❑ 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

$$c_{NB} = \operatorname{argmax}_{c_j \in C} \left(\log P(c_j) + \sum_{i \in \text{positions}} \log P(x_i | c_j) \right)$$

- ❑ Note that model is now just max of sum of weights...

Sounds familiar?

Summary - Two Models: Multivariate Bernoulli

- One feature X_w for each word in dictionary
- $X_w = \text{true}$ 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 (independence)
- This is the model used in the binary independence model in classic probabilistic relevance feedback in hand-classified data.

Summary - Two Models: Multinomial

- One feature X_i for each word positions 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 - for all positions i, j , word w , and class c

$$P(X_i = w | c) = P(X_j = w | c)$$

- Just have one multinomial feature predicting all words.

Parameter estimation

□ Multivariate Bernoulli model:

$$\hat{P}(X_w = t \mid c_j) = \text{fraction of documents of topic } c_j \text{ in which word } w \text{ appears}$$

□ Multinomial model:

$$\hat{P}(X_i = w \mid c_j) = \text{fraction of times in which word } w \text{ appears across all documents of topic } c_j$$

- 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 Bernoulli?
- Multinomial model is almost always more effective in text applications!
 - See results figures later
- See *IIR* sections 13.2 and 13.3 for worked examples with each model

Feature Selection: Why?

- Text collections have a large number of features
 - 10,000 – 1,000,000 unique words ... and more
- May make using a particular classifier feasible
 - Some classifiers can't deal with 100,000 of features
- Reduces training time
 - 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 ideas:

- **Hypothesis testing statistics:**

- Are we confident that the value of one categorical variable is associated with the value of another

- *Chi-square test*

- (<http://faculty.vassar.edu/lowry/webtext.html> chapter 8)

- **Information theory:**

- How much information does the value of one categorical variable give you about the value of another

- *Mutual information*

χ^2 statistic (CHI)

	<i>Term = jaguar</i>	<i>Term \neq jaguar</i>
<i>Class = auto</i>	2	500
<i>Class \neq auto</i>	3	9500

- ❑ If the term "jaguar" is **not relevant** to decide whether a document is about "auto" then the probability to observe a document about "auto" should be equal to the same probability conditioned that term is present or not present
- ❑ $P(C(d)=auto) \stackrel{?}{=} P(C(d)=auto \mid T = j) \stackrel{?}{=} P(C(d)=auto \mid T \neq j)$
- ❑ $502/(502+9503) \stackrel{?}{=} 2/5 \stackrel{?}{=} 500/10000$
- ❑ $0.0501 \stackrel{?}{=} 0.4 \stackrel{?}{=} 0.05$ NOT REALLY!

χ^2 statistic (CHI)

- χ^2 is interested in $(f_o - f_e)^2 / f_e$ summed over all table entries: is the observed number what you'd expect given the marginals?

$$\chi^2(j, a) = \sum (O - E)^2 / E = (2 - .25)^2 / .25 + (3 - 4.75)^2 / 4.75 + (500 - 502)^2 / 502 + (9500 - 9498)^2 / 9498 = 12.9 \quad (p < .001)$$

- The null hypothesis is rejected with confidence .999,
- since $12.9 > 10.83$ (the value for .999 confidence for a 1 degree of freedom χ^2 distribution).

	<i>Term = jaguar</i>	<i>Term ≠ jaguar</i>	
<i>Class = auto</i>	2 (0.25)	500 (502)	expected: f_e
<i>Class ≠ auto</i>	3 (4.75)	9500 (9498)	observed: f_o

χ^2 statistic (CHI)

There is a simpler formula for 2x2 χ^2 :

$$\chi^2(t, c) = \frac{N \times (AD - CB)^2}{(A + C) \times (B + D) \times (A + B) \times (C + D)}$$

$A = \#(t, c)$	$C = \#(\neg t, c)$
$B = \#(t, \neg c)$	$D = \#(\neg t, \neg c)$

$$N = A + B + C + D$$

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, class 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

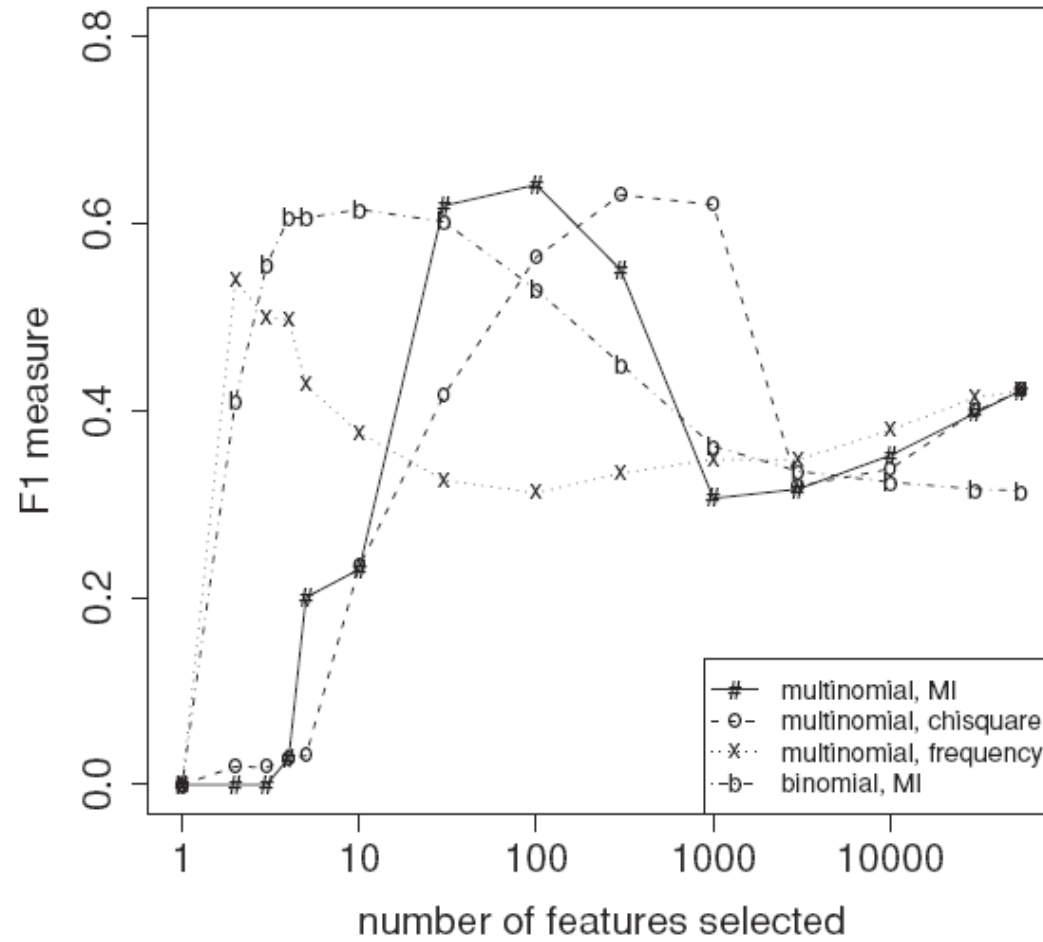
Feature selection via MI (contd.)

- 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 very slightly informative frequent terms that are not very useful for classification
- Chi-square
 - Statistical foundation
 - May select **rare statistically correlated but uninformative** terms
- Just use the commonest terms?
 - No particular foundation
 - In practice, this is often 90% as good.

Example



Feature selection for NB

- In general feature selection is *necessary* for multivariate Bernoulli 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.

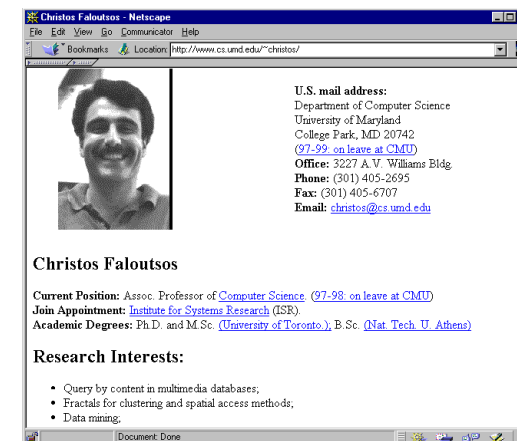
Evaluating Categorization

- ❑ 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
 - Adequate if one class per document
 - Otherwise F measure for each class
- ❑ 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.

WebKB Experiment (1998)

- ❑ 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	Department
Extracted	180	66	246	99	28	1
Correct	130	28	194	72	25	1
Accuracy:	72%	42%	79%	73%	89%	100%

Most relevant features: MI

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

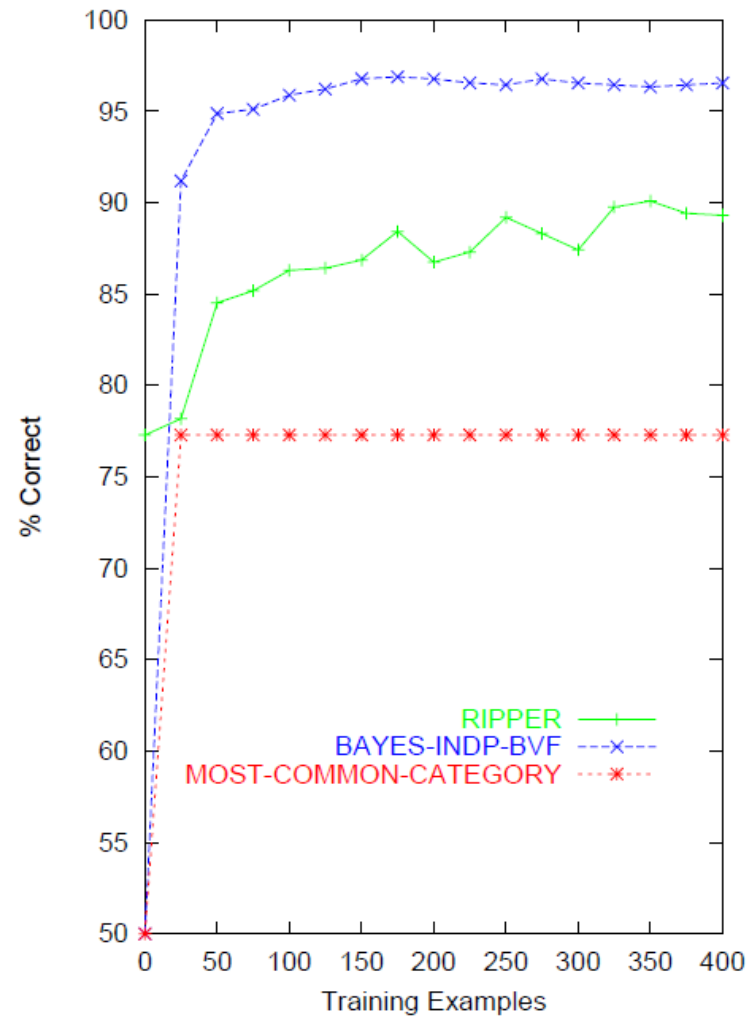
Research Projects

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

Naïve Bayes on spam email



SpamAssassin

- Naïve Bayes has found a home in spam filtering
 - Paul Graham's *A Plan for Spam*
 - A mutant with more mutant offspring...
 - Naive Bayes-like classifier with weird parameter estimation
 - Widely used in spam filters
 - Classic Naive Bayes superior when appropriately used (according to David D. Lewis)
 - But also many other things: black hole lists, etc.
- Many email topic filters also use NB classifiers.

Naïve Bayes Posterior Probabilities

- **Classification results** of naïve Bayes (the class with maximum posterior probability) **are usually fairly accurate**
- However, due to the inadequacy of the conditional independence assumption, the **actual posterior-probability numerical estimates are not**
 - Output probabilities are commonly very close to 0 or 1
- Correct estimation \Rightarrow accurate prediction, but correct probability estimation is **NOT** necessary for accurate prediction (just need right ordering of probabilities).

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 equally important 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 of counting over the data; testing linear in the number of attributes, and document collection size

- Low Storage requirements

Resources

- IIR 13
- Tom Mitchell, Machine Learning. McGraw-Hill, 1997.
 - Clear simple explanation of Naïve Bayes