CS276
Information Retrieval and Web Search

Lecture 12: Naïve BayesText Classification

Probabilistic relevance feedback

Recall this idea:

- Rather than reweighting in a vector space…
- If user has told us some relevant and some irrelevant documents, then we can proceed to build a probabilistic classifier, such as the Naive Bayes model we will look at today:
  - $P(t_k|R) = |\mathbf{D}_{rk}| / |\mathbf{D}_r|$
  - $P(t_k|NR) = |\mathbf{D}_{nrk}| / |\mathbf{D}_{nr}|$
    - t_k is a term; $\mathbf{D}_r$ is the set of known relevant documents; $\mathbf{D}_{rk}$ is the subset that contain t_k; $\mathbf{D}_{nr}$ is the set of known irrelevant documents; $\mathbf{D}_{nrk}$ is the subset that contain t_k.

Recall a few probability basics

- For events *a* and *b:*
- Bayes' Rule

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

$$p(\bar{a} \mid b)\,p(b) = p(b \mid \bar{a})\,p(\bar{a})$$

Prior

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

Posterior

- Odds:

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

Text classification:
Naïve Bayes Text Classification

- Today:
  - Introduction to Text Classification
  - Probabilistic Language Models
  - Naïve Bayes text categorization

Is this spam?

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.

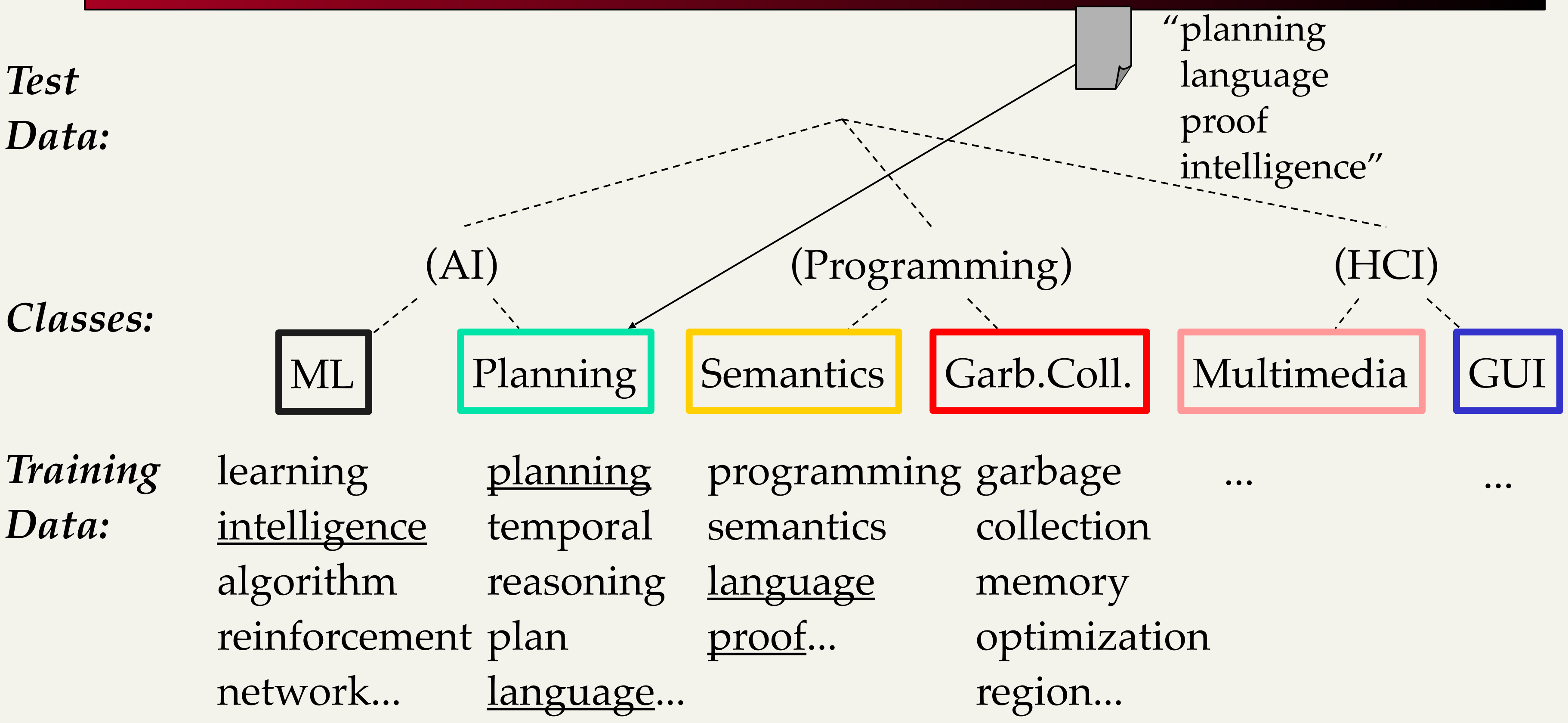
Change your life NOW !

===
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
===

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: $c(x) \in C$, where $c(x)$ is a *categorization function* whose domain is X and whose range is C.
    - We want to know how to build categorization functions ("classifiers").

Document Classification


Test Data:
"planning language proof intelligence"
Classes:
(AI)
(Programming)
(HCI)
ML
Planning
Semantics
Garb.Coll.
Multimedia
GUI
Training Data:
learning intelligence algorithm reinforcement network...
planning temporal reasoning plan language...
programming semantics language proof...
garbage collection memory optimization region...
...
...

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

Text Categorization Examples

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

  e.g., “like”, “hate”, “neutral”
- Labels may be domain-specific binary

  e.g., "interesting-to-me" : "not-interesting-to-me”

  e.g., “spam” : “not-spam”

  e.g., “contains adult language” :“doesn’t”

Classification Methods (1)

- Manual classification
  - Used by Yahoo!, Looksmart, about.com, ODP, Medline
  - Very accurate when job is done by experts
  - Consistent when the problem size and team is small
  - Difficult and expensive to scale

Classification Methods (2)

- Automatic document classification
  - Hand-coded rule-based systems
    - One technique used by CS dept’s spam filter, Reuters, CIA, Verity, …
    - 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

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

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.
- Build a *generative model* that approximates how data is produced
- 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

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

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

Maximum a posteriori Hypothesis

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

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

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

As *P(D)* is constant

Maximum likelihood Hypothesis

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

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

Naive Bayes Classifiers

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

$$c_{MAP} = \underset{c_j \in C}{\operatorname{argmax}}\, P(c_j \mid x_1, x_2, \boxed{?}, x_n)$$

$$= \underset{c_j \in C}{\operatorname{argmax}} \frac{P(x_1, x_2, \boxed{?}, x_n \mid c_j) P(c_j)}{P(x_1, x_2, \boxed{?}, x_n)}$$

$$= \underset{c_j \in C}{\operatorname{argmax}}\, P(x_1, x_2, \boxed{?}, x_n \mid c_j) P(c_j)$$

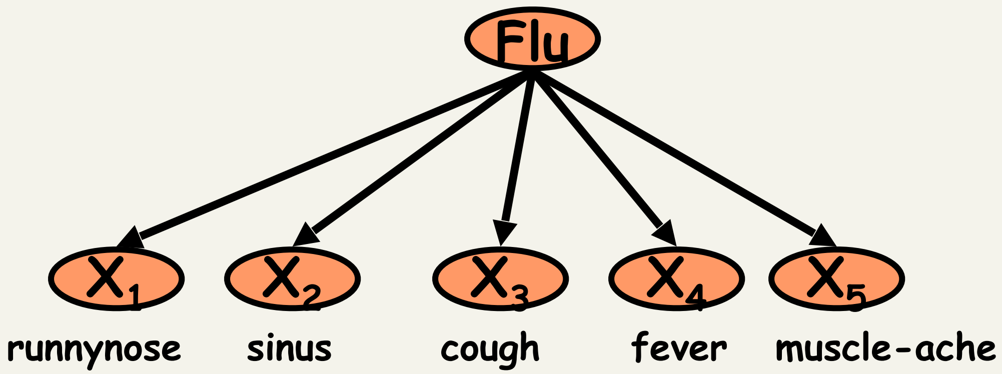
Naïve Bayes Classifier: Naïve Bayes Assumption

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

The Naïve Bayes Classifier

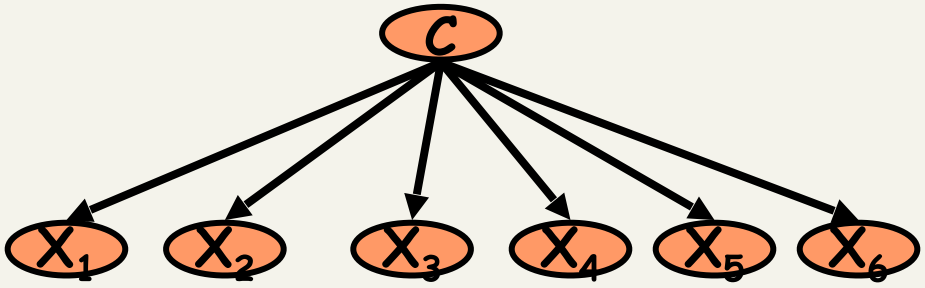


- **Conditional Independence Assumption:** features detect term presence and are independent of each other given the class:

$$P(X_1, \boxed{?}, X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \boxed{?} \bullet P(X_5 \mid C)$$

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

Learning the Model

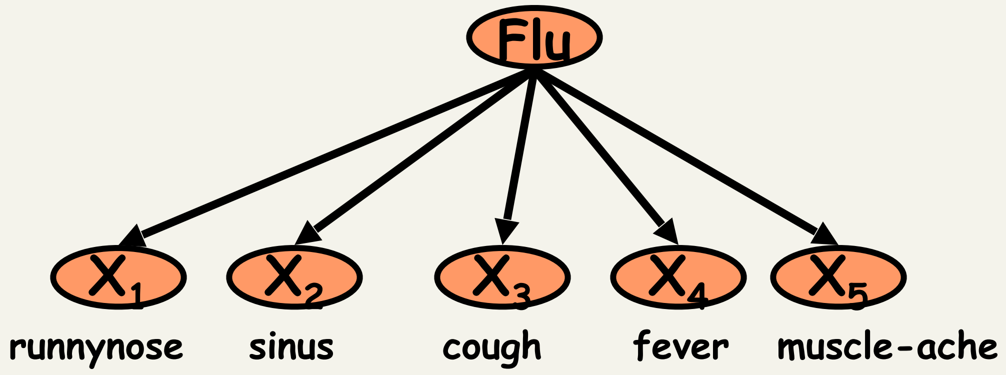

$$\begin{tikzcd} & & & C \arrow[dlll] \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] \arrow[drr] & & \\ X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{tikzcd}$$

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

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

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

Problem with Max Likelihood



$$P(X_1, \boxed{?}, X_5 \mid C) = P(X_1 \mid C) \bullet P(X_2 \mid C) \bullet \boxed{?} \bullet P(X_5 \mid C)$$

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

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

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

$$\boxed{?} = \arg\max_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

Smoothing to Avoid Overfitting

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

# of values of X_i

- Somewhat more subtle version

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

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

extent of "smoothing"

Stochastic Language Models

- Models *probability* of generating strings (each word in turn) in the language (commonly all strings over ∑). 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

$$\underbrace{0.2 \times 0.01 \times 0.02 \times 0.2 \times 0.01}_{\boxed{\text{multiply}}}$$

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

P (🔴 🟡 🔴 🔵)

- **= P (🔴) P (🟡 | 🔴) P (🔴 | 🔴 🟡) P (🔵 | 🔴 🟡 🔴)**

- Unigram Language Models

  P (🔴) P (🟡) P (🔴) P (🔵)

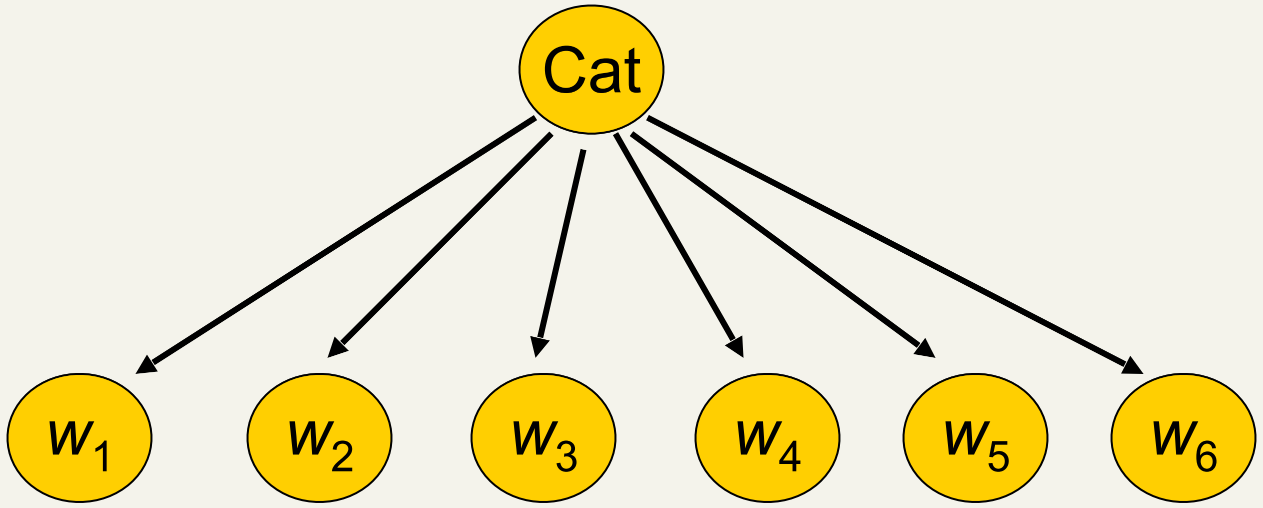
Easy.
Effective!

- Bigram (generally, *n*-gram) Language Models

  P (🔴) P (🟡 | 🔴) P (🔴 | 🟡) P (🔵 | 🔴)

- Other Language Models
  - Grammar-based models (PCFGs), 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.

$$c_{NB} = \operatorname*{argmax}_{c_j \in C} P(c_j) \prod_i P(x_i \mid c_j)$$

$$= \operatorname*{argmax}_{c_j \in C} P(c_j) P(x_1 = \text{"our"} \mid c_j) \boxed{?} P(x_n = \text{"text"} \mid c_j)$$

- 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)

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 $Text_j$
      - $$P(x_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha\,|Vocabulary|}$$

Naïve Bayes: Classifying

- positions ← 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 $|C||V| < |D|L_d$

Why?

- 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.

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.

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

Note: Two Models

- Model 1: Multivariate binomial
  - One feature X_w for each word in dictionary
  - X_w = 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)

Two Models

- 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

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

    - Just have one multinomial feature predicting all words

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

Parameter estimation

- Binomial 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 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

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 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)

χ^2 statistic (CHI)

- χ2 is interested in (fo – fe)²/fe summed over all table entries: is the observed number what you'd expect given the marginals?

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

- The null hypothesis is rejected with confidence .999,
- since 12.9 > 10.83 (the value for .999 confidence).

	Term = jaguar	*Term ≠ jaguar*
Class = auto	2 *(0.25)*	500 *(502)*
Class ≠ auto	3 *(4.75)*	9500 *(9498)*

expected: f_e

observed: f_o

χ^2 statistic (CHI)

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

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

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

$N = A + B + C + D$

Value for complete independence of term and category?

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 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

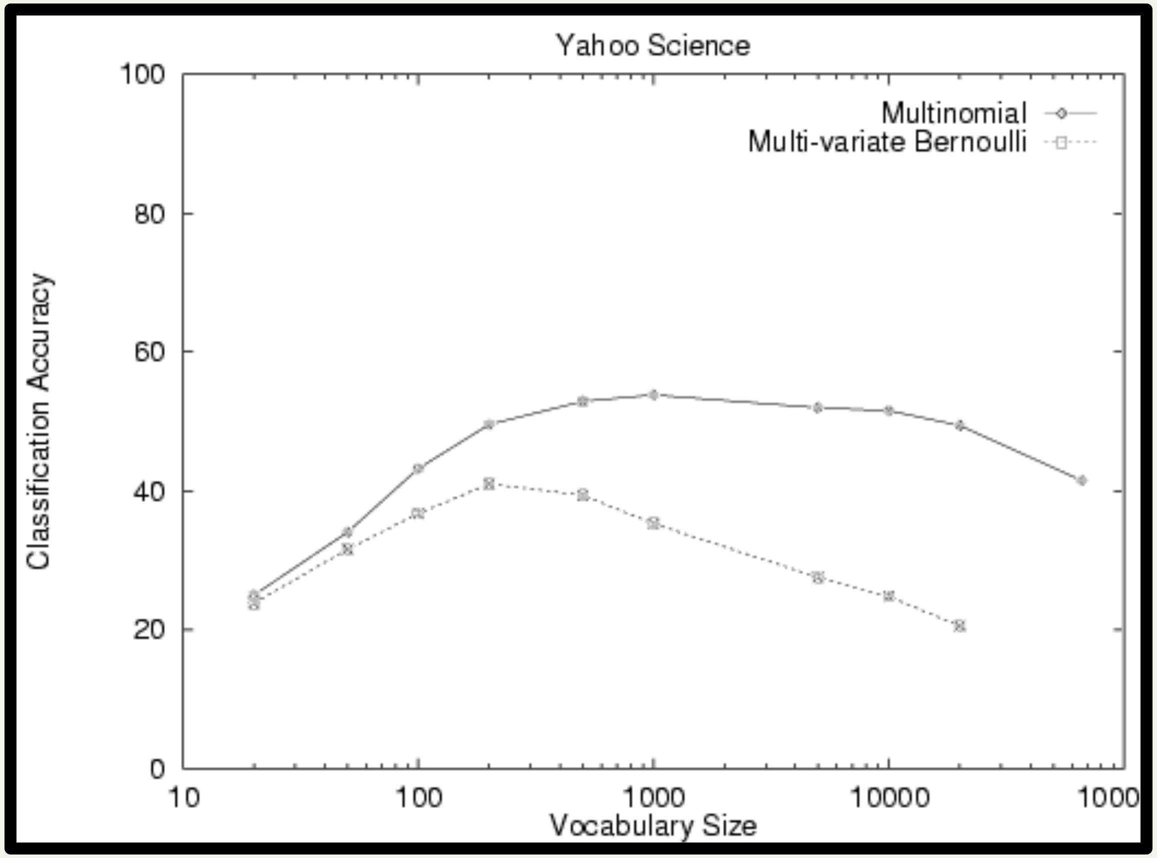
- "Feature selection" really means something different for multinomial NB. It means dictionary truncation
  - The multinomial NB model only has 1 feature
- This "feature selection" normally isn't needed for multinomial NB, but may help a fraction with quantities that are badly estimated

Evaluating Categorization

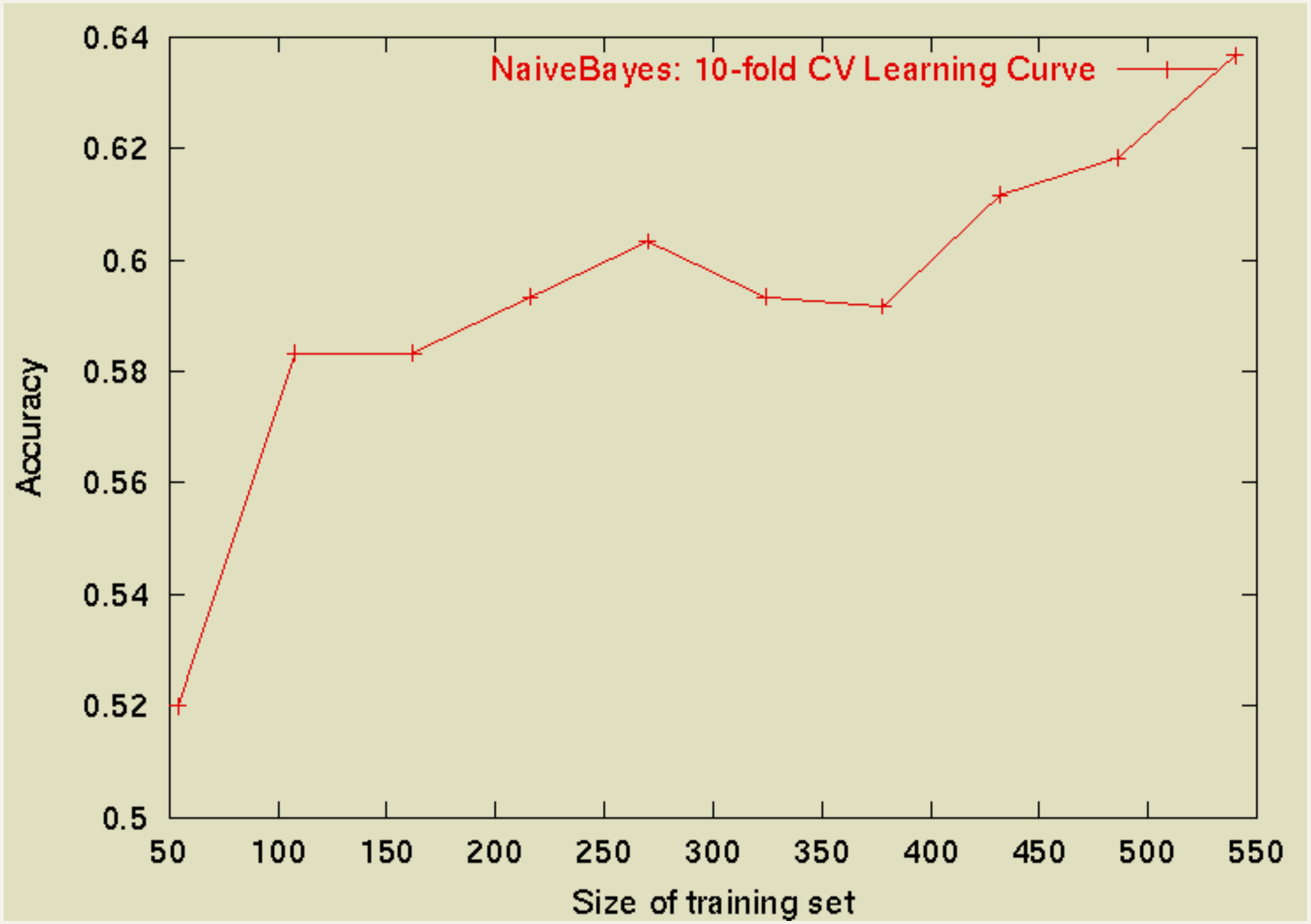
- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
- *Classification accuracy*: *c*/*n* where *n* is the total number of test instances and *c* is the number of test instances correctly classified by the system.
- Results can vary based 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)

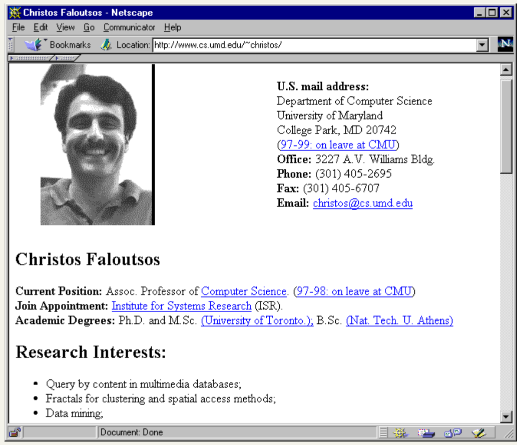


Sample Learning Curve (Yahoo Science Data): need more!



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)


Christos Faloutsos - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://www.cs.umd.edu/~christos/
Document: Done

U.S. mail address:
Department of Computer Science
University of Maryland
College Park, MD 20742
(97-99: on leave at CMU)
Office: 3227 A.V. Williams Bldg.
Phone: (301) 405-2695
Fax: (301) 405-6707
Email: christos@cs.umd.edu

Christos Faloutsos

Current Position: Assoc. Professor of Computer Science. (97-98: on leave at CMU)
Join Appointment: Institute for Systems Research (ISR).
Academic Degrees: Ph.D. and M.Sc. (University of Toronto.); B.Sc. (Nat. Tech. U. Athens)

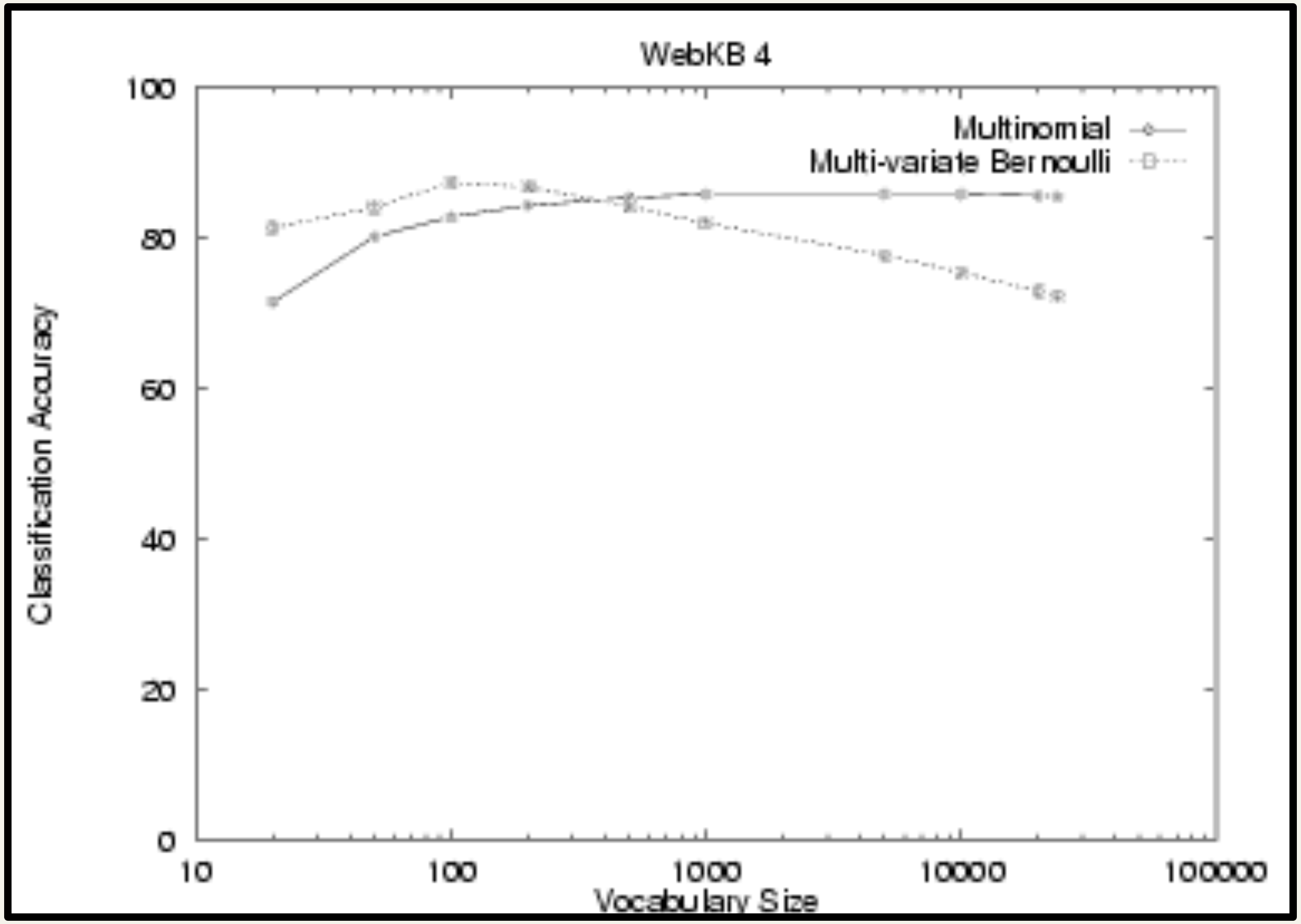
Research Interests:

- Query by content in multimedia databases;
- Fractals for clustering and spatial access methods;
- Data mining;

- 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

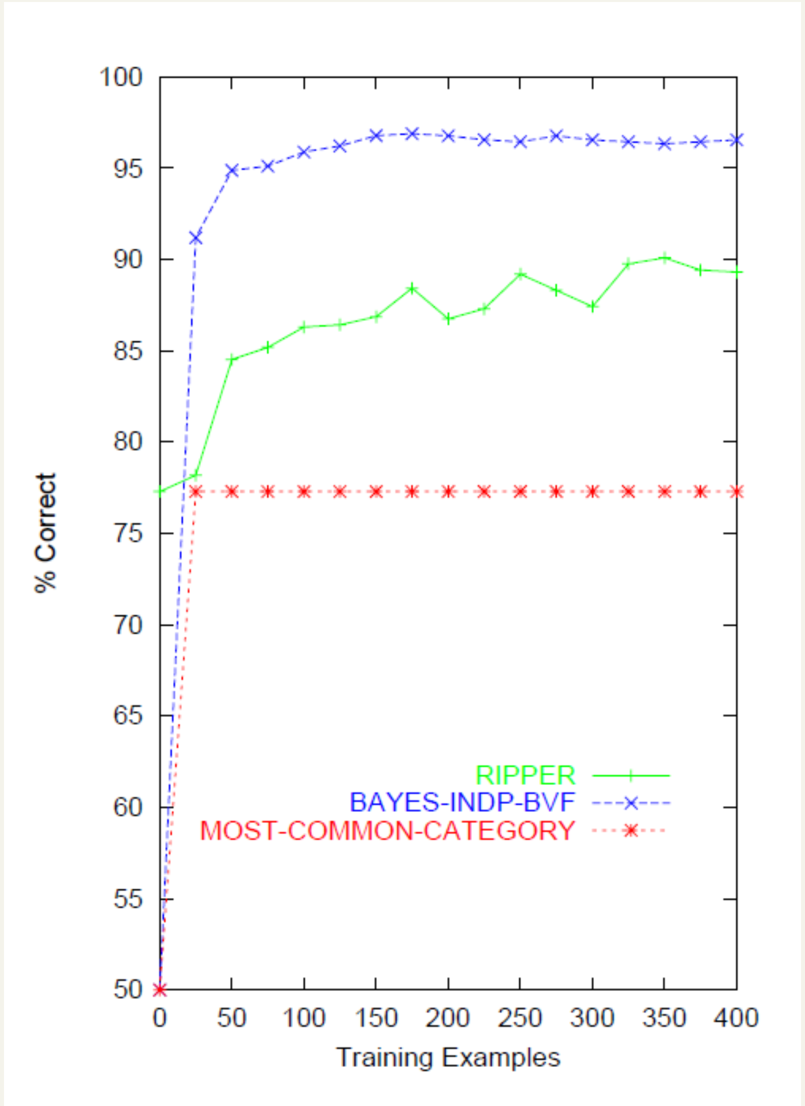
Research Projects

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

Naïve Bayes on spam email



SpamAssassin

- Naïve Bayes has found a home for spam filtering
  - Graham’s *A Plan for Spam*
    - And its 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
- Many email filters use NB classifiers
  - But also many other things: black hole lists, etc.

Violation of NB Assumptions

- Conditional independence
- “Positional independence”
- Examples?

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 generally very close to 0 or 1.

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 ⇒ accurate prediction

but NOT

~~accurate prediction ⇒ 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

- IIR 13
- Fabrizio Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34(1):1-47, 2002.
- 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.