CSCI 5417
Information Retrieval Systems
Jim Martin

Lecture 11
9/29/2011

Today 9/29

- Classification
  - Naïve Bayes classification
    - Unigram LM

Where we are...

- Basics of ad hoc retrieval
  - Indexing
  - Term weighting/scoring
    - Cosine
  - Evaluation
- Document classification
- Clustering
- Information extraction
- Sentiment/Opinion mining

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
===

Text Categorization Examples

Assign labels to each document or web-page:

- Labels are most often topics such as Yahoo-categories
  finance, sports, news>world>asia>business
- Labels may be genres
  editorials, movie-reviews, news
- Labels may be opinion
  like, hate, neutral
- Labels may be domain-specific
  "interesting-to-me" : "not-interesting-to-me"
  "spam" : "not-spam"
  "contains adult content" :"doesn't"
  important to read now: not important

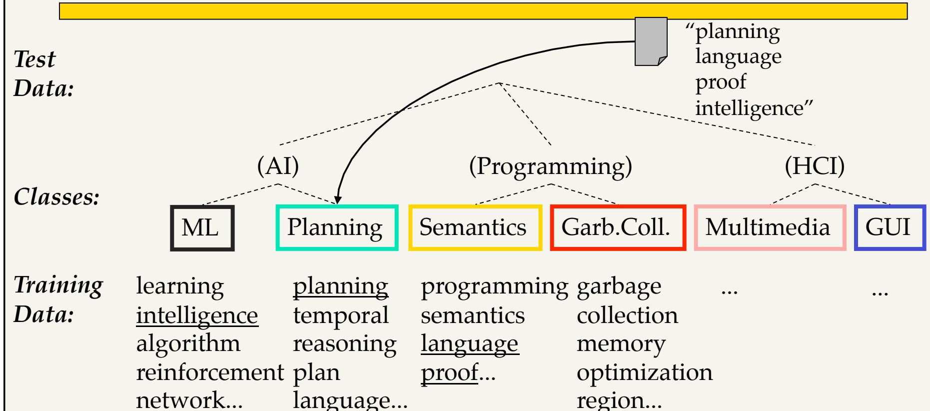
Categorization/Classification

- Given:
  - A description of an instance, $x \in X$, where X is the *instance language* or *instance space*.
    - Issue for us is how to represent text documents
  - And 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 (i.e. "classifiers").

Text Classification Types

- Those examples can be further classified by type
  - Binary
    - Spam/not spam, contains adult content/doesn't
  - Multiway
    - Business vs. sports vs. gossip
  - Hierarchical
    - News> UK > Wales> Weather >
  - Mixture model
    - .8 basketball, .2 business

Document Classification



Bayesian Classifiers

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

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

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

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

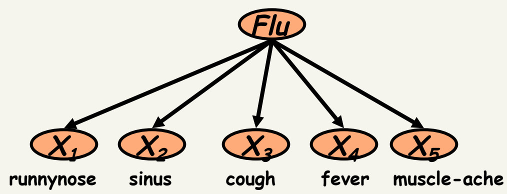
Naïve Bayes Classifiers

- $P(c_j)$
  - 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 $P(x_i|c_j)$.

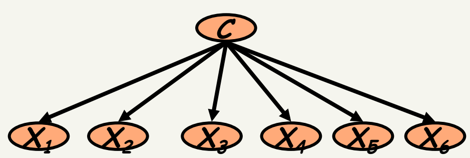
The Naïve Bayes Classifier (Belief Net)



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

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

Learning the Model



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

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

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

Smoothing to Avoid Overfitting

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

Add-One smoothing

# of values of X_i

Stochastic Language Models

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

13.2.1

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

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

13.2.1

Unigram and higher-order models

P (● ○ ● ●)

- = P (●) P (○ | ●) P (● | ● ○) P (● | ● ○ ●)
- Unigram Language Models

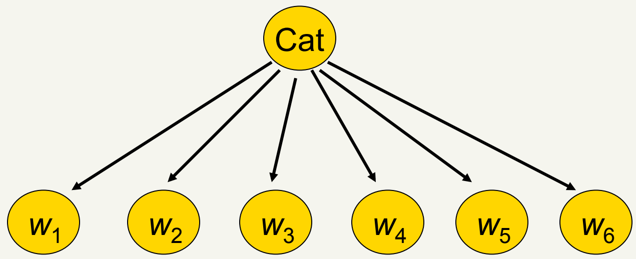
  P (●) P (○) P (●) P (●)

  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

13.2.1

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 to Classify Text

- Attributes are text positions, values are words.

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

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

Multinomial Model

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, $prior$, $condprob$

Naïve Bayes: Classifying

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

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

Apply Multinomial

```
ApplyMultinomialNB(ℂ, V, prior, condprob, d)
W ← ExtractTokensFromDoc(V, d)
for each c ∈ ℂ
do score[c] ← log prior[c]
    for each t ∈ W
    do score[c] += log condprob[t][c]
return arg max_{c∈ℂ} score[c]
```

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.

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.

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

- Note that model is now just max of sum of weights…

Naïve Bayes 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

Naïve Bayes example

- V is {China, soccer, Japan, baseball, trade exports}
- |V| = 6
- Sizes
  - Sports = 2 docs, 4 tokens
  - Politics = 2 docs, 5 tokens

Japan	Raw	Sm
Sports	1/4	2/10
Politics	2/5	3/11

soccer	Raw	Sm
Sports	1/4	2/10
Politics	0/5	1/11

Naïve Bayes example

- Classifying
  - Soccer (as a doc)
    - Soccer | sports = .2
    - Soccer | politics = .09

    Sports > Politics or

    .2/.2+.09 = .69

    .09/.2+.09 = .31

New example

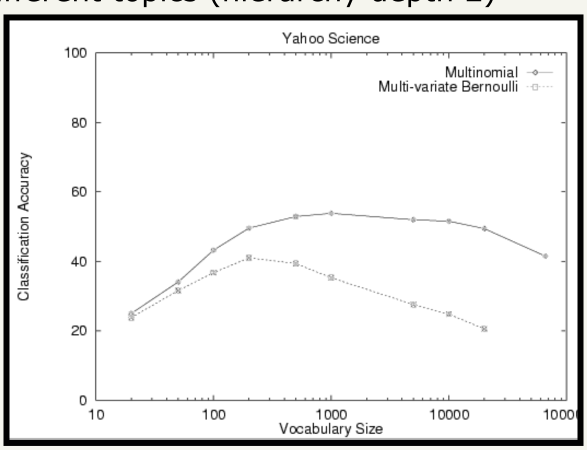
- What about a doc like the following?
  - *Japan soccer*
    - Sports
      - P(japan|sports)P(soccer|sports)P(sports)
      - .2 * .2 * .5 = .02
    - Politics
      - P(japan|politics)P(soccer|politics)P(politics)
      - .27 * .09 *. 5 = .01
    - Or
      - .66 to .33

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.
- 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)

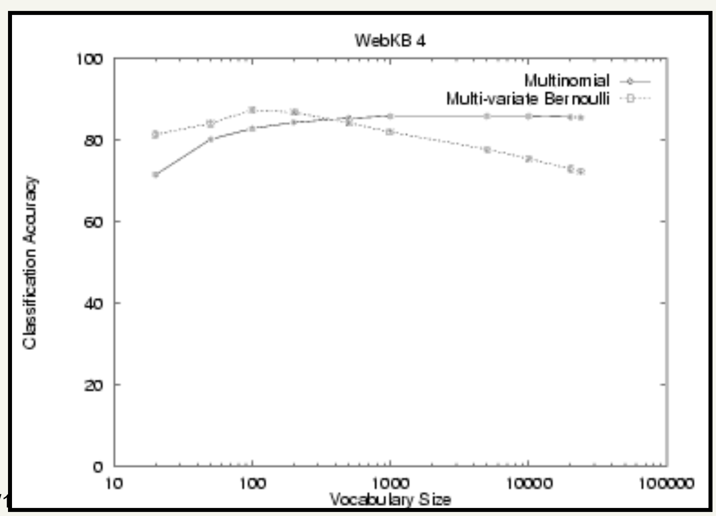


WebKB Experiment

- 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)

	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

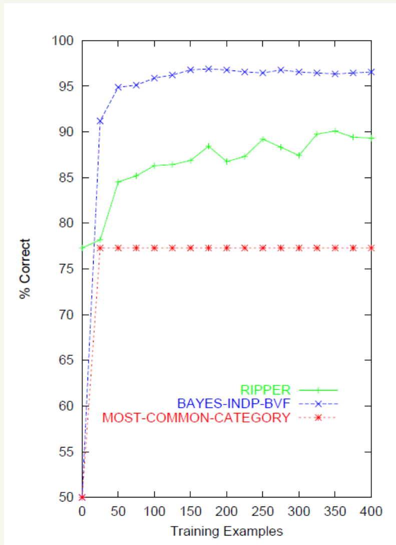
Research Projects	
investigators	0.00256
group	0.00250
members	0.00242
researchers	0.00241
laboratory	0.00238
develop	0.00201
related	0.00200
arpa	0.00187
affiliated	0.00184
project	0.00183

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

SpamAssassin

- Naïve Bayes made a big splash with spam filtering
  - Paul Graham's *A Plan for Spam*
    - And its 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.

Naïve Bayes on spam email



Naive Bayes is Not So Naive

- Does well in many standard evaluation competitions
- 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
- Very Fast: Learning with one pass over the data; testing linear in the number of attributes, and document collection size
- Low Storage requirements

Next couple of classes

- Other classification issues
  - What about vector spaces?
    - Lucene infrastructure
  - Better ML approaches
    - SVMs etc.