

---

# Text Categorization using Naïve Bayes

## Mausam

(based on slides of Dan Weld, Dan Jurafsky, Prabhakar Raghavan, Hinrich Schutze, Guillaume Obozinski, David D. Lewis, Fei Xia)

# Categorization

---

- **Given:**
  - A **description of an instance**,  $x \in X$ , where  $X$  is the *instance language* or *instance space*.
  - A **fixed set of categories**:  
$$C = \{c_1, c_2, \dots, c_n\}$$
- **Determine:**
  - The **category of  $x$** :  $c(x) \in C$ , where  $c(x)$  is a categorization function whose domain is  $X$  and whose range is  $C$ .

# County vs. Country?

[article](#) [discussion](#) [edit this page](#) [history](#)

*• Ten things you didn't know about images on Wikipedia •*

## King County, Washington

From Wikipedia, the free encyclopedia

Coordinates: 47°47′, -121°8′

*"King County" redirects here. For other uses, see [King County \(disambiguation\)](#).*

**King County** is located in the [U.S. state](#) of [Washington](#). The population in the 2000 census was 1,737,034 and in 2006 was an estimated 1,835,300. By population, King is the largest county in Washington, and the 12th largest in the United States. As of 2006, the county had a population comparable to that of the state of [Nebraska](#).

The county seat is [Seattle](#), which is the state's largest city. About two-thirds of the county's population lives in the city's [suburbs](#). King County ranks among the 100 [highest-income counties](#) in the United States.

**Contents** [\[show\]](#)

## History [\[edit\]](#)

The county was formed out of territory within [Thurston County](#) on [December 22, 1852](#), by the [Oregon Territory](#) legislature, and was named after [Alabama](#) resident [William Rufus King](#), vice president under president [Franklin Pierce](#). Seattle was made the county seat on [January 11, 1853](#).<sup>[[1](#)]</sup><sup>[[2](#)]</sup><sup>[[2](#)]</sup>

King County originally extended to the [Olympic Peninsula](#). According to historian [Bill Speidel](#),

### King County, Washington



### King County

#### Map



Location in the state of [Washington](#)



Washington's location in the USA

#### Statistics

**Founded** December 22, 1852

**Seat** Seattle

[article](#) [discussion](#) [edit this page](#) [history](#)

*• Ten things you didn't know about images on Wikipedia •*

## Kenya

From Wikipedia, the free encyclopedia

**This article needs additional [references or sources](#) for [verification](#). Please help [improve this article](#) by adding [reliable references](#). Unverifiable m**

**be [challenged](#) and removed.**

The **Republic of Kenya** is a country in [Eastern Africa](#). It is bordered by [Ethiopia](#) to the north, [Somalia](#) to the northeast, [Tanzania](#) to the south, [Uganda](#) to the west, and [Sudan](#) to the northwest, with the [Indian Ocean](#) running along the southeast [border](#).

**Contents** [\[show\]](#)

## History [\[edit\]](#)

*Main article: [History of Kenya](#)*

[Paleontologists](#) have discovered many fossils of [prehistoric](#) animals in Kenya. At one of the rare [dinosaur fossil](#) sites in Africa, two hundred [Cretaceous theropod](#) and [giant crocodile](#) fossils have been discovered in Kenya, dating from the [Mesozoic Era](#), over 200 million years ago. The fossils were found in an excavation conducted by a team from the [University of Utah](#) and the [National Museums of Kenya](#) in July-August 2004 at

### Jamhuri ya Republic of



Flag

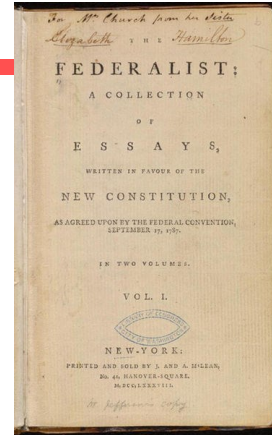
**Motto**  
"Harambee"  
"Let us all pull"

**Anthem**  
*Ee Mungu Ngu*  
"Oh God of All"

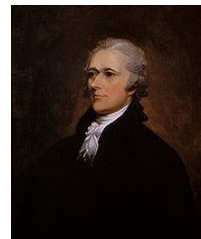


# Who wrote which Federalist papers?

- 1787-8: anonymous essays try to convince New York to ratify U.S Constitution: Jay, Madison, Hamilton.
- Authorship of 12 of the letters in dispute
- 1963: solved by Mosteller and Wallace using Bayesian methods



James Madison



Alexander Hamilton

# Male or female author?

- The main aim of this article is to propose an exercise in stylistic analysis which can be employed in the teaching of English language. It details the design and results of a workshop activity on narrative carried out with undergraduates in a university department of English. The methods proposed are intended to enable students to obtain insights into aspects of cohesion and narrative structure: insights, it is suggested, which are not as readily obtainable through more traditional methods.
- My aim in this article is to propose a new approach to understanding of what some of these so-called apposition markers indicate. It will be argued that the decision to put something in other words is essentially a decision about style, a point which is, perhaps, anticipated by Burton-Roberts when he describes loose apposition as a rhetorical device. However, he does not justify this suggestion by giving the criteria for classifying a mode of expression as a rhetorical device.

Female writers use

more first person/second person pronouns





more gender laden third person pronouns

(overall more personalization)

approach to understanding of

# Positive or negative movie review?

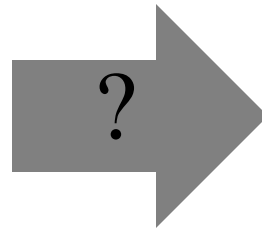
---

-  • unbelievably disappointing
-  • Full of zany characters and richly applied satire, and some great plot twists
-  • this is the greatest screwball comedy ever filmed
-  • It was pathetic. The worst part about it was the boxing scenes.

# What is the subject of this article?

## MeSH Subject Category Hierarchy

### MEDLINE Article



- Antagonists and Inhibitors
- Blood Supply
- Chemistry
- Drug Therapy
- Embryology
- Epidemiology
- ...

# Text Classification

---

- Assigning documents to a fixed set of categories, *e.g.*
- Web pages
  - Yahoo-like classification
  - Assigning subject categories, topics, or genres
- Email messages
  - Spam filtering
  - Prioritizing
  - Folderizing
- Blogs/Letters/Books
  - Authorship identification
  - Age/gender identification
- Reviews/Social media
  - Language Identification
  - Sentiment analysis
  - ...



# Classification Methods:

## Hand-coded rules

---

- Rules based on combinations of words or other features
  - spam: black-list-address OR (“dollars” AND “have been selected”)
- Accuracy can be high
  - If rules carefully refined by expert
- But building and maintaining these rules is expensive

# Learning for Text Categorization

---

- Hard to construct text categorization functions.
- Learning Algorithms:
  - **Bayesian (naïve)**
  - Neural network
  - Logistic Regression
  - Nearest Neighbor (case based)
  - Support Vector Machines (SVM)

# Bayesian Methods

---

- 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 Applied to Documents and Classes

---

- For a document  $d$  and a class  $c$

$$P(c | d) = \frac{P(d | c)P(c)}{P(d)}$$

# Naïve Bayes Classifier (I)

---

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(c | d)$$

MAP is “maximum a posteriori” = most likely class

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)}$$

Bayes Rule

$$= \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

Dropping the denominator

# Naïve Bayes Classifier (II)

---

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

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

Document  $d$   
represented as  
features  $x_1 \dots x_n$

# Naïve Bayes Classifier (IV)

---

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

$O(|X|^n \cdot |C|)$  parameters

Could only be estimated if a very, very large number of training examples was available.

How often does this class occur?

We can just count the relative frequencies in a corpus

# Multinomial Naïve Bayes Independence Assumptions

---

$$P(x_1, x_2, \dots, x_n | c)$$

- **Bag of Words assumption:** Assume position doesn't matter



# The bag of words representation

---

Y (

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.

) = C



# The bag of words representation

---

Y(

I **love** this movie! It's **sweet**, but with **satirical** humor. The dialogue is **great** and the adventure scenes are **fun**... It manages to be **whimsical** and **romantic** while **laughing** at the conventions of the fairy tale genre. I would **recommend** it to just about anyone. I've seen it **several** times, and I'm always **happy** to see it **again** whenever I have a friend who hasn't seen it yet.

)=C



# The bag of words representation: using a subset of words

---

Y (

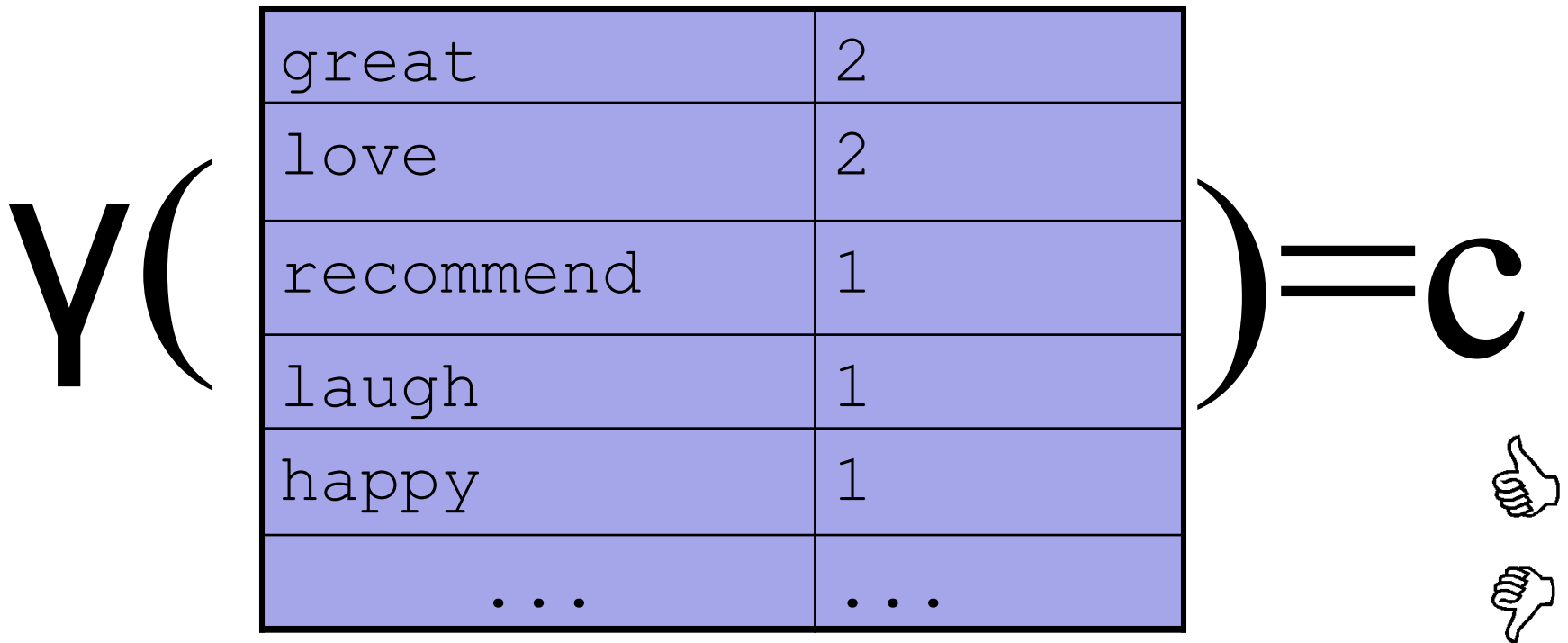
```
x love xxxxxxxxxxxxxxxxxxxx sweet
xxxxxxxx satirical xxxxxxxxxxxx
xxxxxxxxxxxx great xxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxx fun xxxx
xxxxxxxxxxxxxxxx whimsical xxxx
romantic xxxx laughing
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxx recommend xxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx several xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxx happy xxxxxxxxxxxx again
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

) = C



# The bag of words representation

---



# Bag of words for document classification

---

?

Test document

parser  
language  
label  
translation  
...

Machine Learning

learning  
training  
algorithm  
shrinkage  
network...

NLP

parser  
tag  
training  
translation  
language...

Garbage Collection

garbage  
collection  
memory  
optimization  
region...

Planning

planning  
temporal  
reasoning  
plan  
language...

GUI

...

# Multinomial Naïve Bayes Classifier

---

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

$$c_{NB} = \operatorname{argmax}_{c \in C} P(c) \prod_{x \in X} P(x | c)$$

# Multinomial Naïve Bayes Independence Assumptions

---

$$P(x_1, x_2, \dots, x_n | c)$$

- **Bag of Words assumption:** Assume position doesn't matter
- **Conditional Independence:** Assume the feature probabilities  $P(x_i | c_j)$  are independent given the class  $c$ .

$$P(x_1, \dots, x_n | c) = P(x_1 | c) \cdot P(x_2 | c) \cdot P(x_3 | c) \cdot \dots \cdot P(x_n | c)$$

# Applying Multinomial Naive Bayes Classifiers to Text Classification

---

positions  $\leftarrow$  all word positions in test document

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



# Learning the Multinomial Naïve Bayes Model

---

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

$$\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

# Parameter estimation

---

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

fraction of times word  $w_i$  appears  
among all words in documents of topic  $c_j$

- Create mega-document for topic  $j$  by concatenating all docs in this topic
  - Use frequency of  $w$  in mega-document

# Problem with Maximum Likelihood

- What if we have seen no training documents with the word *fantastic* and classified in the topic **positive** (*thumbs-up*)?

$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{\text{count}(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0$$

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

$$c_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

# Laplace (add-1) smoothing for Naïve Bayes

---

$$\begin{aligned}
 \hat{P}(w_i | c) &= \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)} \\
 &= \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} \text{count}(w, c) + |V|}
 \end{aligned}$$

# Multinomial Naïve Bayes: Learning

---

- From training corpus, extract *Vocabulary*
- Calculate  $P(c_j)$  terms
  - For each  $c_j$  in  $C$  do
    - $docs_j \leftarrow$  all docs with class =  $c_j$
    - $$P(c_j) \square \frac{|docs_j|}{|\text{total \# documents}|}$$
- Calculate  $P(w_k | c_j)$  terms
  - $Text_j \leftarrow$  single doc containing all  $docs_j$
  - For each word  $w_k$  in *Vocabulary*
    - $n_k \leftarrow$  # of occurrences of  $w_k$  in  $Text_j$
    - $$P(w_k | c_j) \square \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

# Naïve 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.

# Easy to Implement

---

- But...
- If you do... it probably won't work...

# Probabilities: Important Detail!

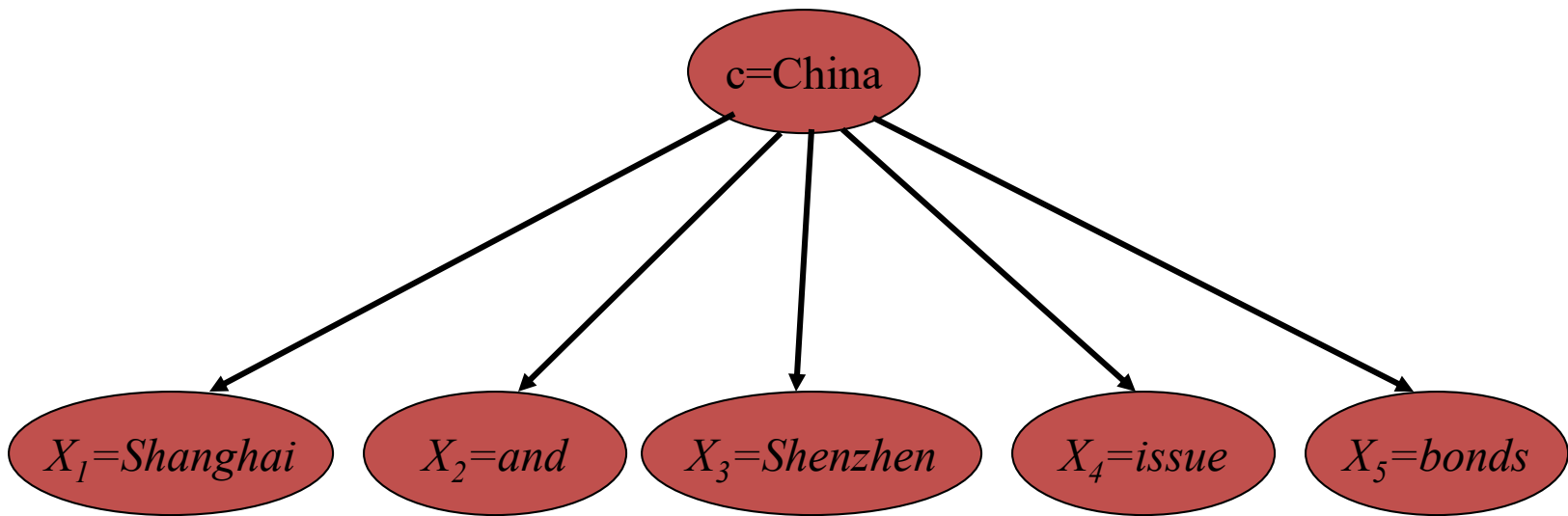
---

- We are multiplying lots of small numbers  
Danger of underflow!
  - $0.5^{57} = 7 \text{ E } -18$
- Solution? Use logs and add!
  - $p_1 * p_2 = e^{\log(p1)+\log(p2)}$
  - Always keep in log form



# Generative Model for Multinomial Naïve Bayes

---



# Each class = a unigram language model

- Assigning each word:  $P(\text{word} \mid c)$
- Assigning each sentence:  $P(s \mid c) = \prod P(\text{word} \mid c)$

Class <i>pos</i>		<u>l</u>	<u>love</u>	<u>this</u>	<u>fun</u>	<u>film</u>
0.1	l					
0.1	love	0.1	0.1	.05	0.01	0.1
0.01	this					
0.05	fun					
0.1	film					
...						

$P(s \mid \text{pos}) = 0.0000005$

# Naïve Bayes as a Language Model

Sec 13.2.1

- Which class assigns the higher probability to s?

Model pos	
0.1	I
0.1	love
0.01	this
0.05	fun
0.1	film

Model neg	
0.2	I
0.001	love
0.01	this
0.005	fun
0.1	film

<u>I</u>	<u>love</u>	<u>this</u>	<u>fun</u>	<u>film</u>
0.1	0.1	0.01	0.05	0.1
0.2	0.001	0.01	0.005	0.1

$$P(s|\text{pos}) > P(s|\text{neg})$$

# Example: Naïve Bayes in Spam Filtering

---

- SpamAssassin Features:

- Mentions Generic Viagra
- Online Pharmacy
- Mentions millions of (dollar) ((dollar) NN,NNN,NNN.NN)
- Phrase: impress ... girl
- From: starts with many numbers
- Subject is all capitals
- HTML has a low ratio of text to image area
- One hundred percent guaranteed
- Claims you can be removed from the list
- 'Prestigious Non-Accredited Universities'
- [http://spamassassin.apache.org/tests\\_3\\_3\\_x.html](http://spamassassin.apache.org/tests_3_3_x.html)

# Advantages

---

- Simple to implement
  - No numerical optimization, matrix algebra, etc
- Efficient to train and use
  - Easy to update with new data
  - Fast to apply
- Binary/multi-class
- Good in domains with many equally important features
  - Decision Trees suffer from fragmentation in such cases – especially if little data
- Comparatively good effectiveness with small training sets
- A good dependable baseline for text classification
  - But we will see other classifiers that give better accuracy

# Disadvantages

---

- Independence assumption wrong
  - Absurd estimates of class probabilities
    - Output probabilities close to 0 or 1
  - Thresholds must be tuned; not set analytically
- Generative model
  - Generally lower effectiveness than discriminative techniques

# Experimental Evaluation

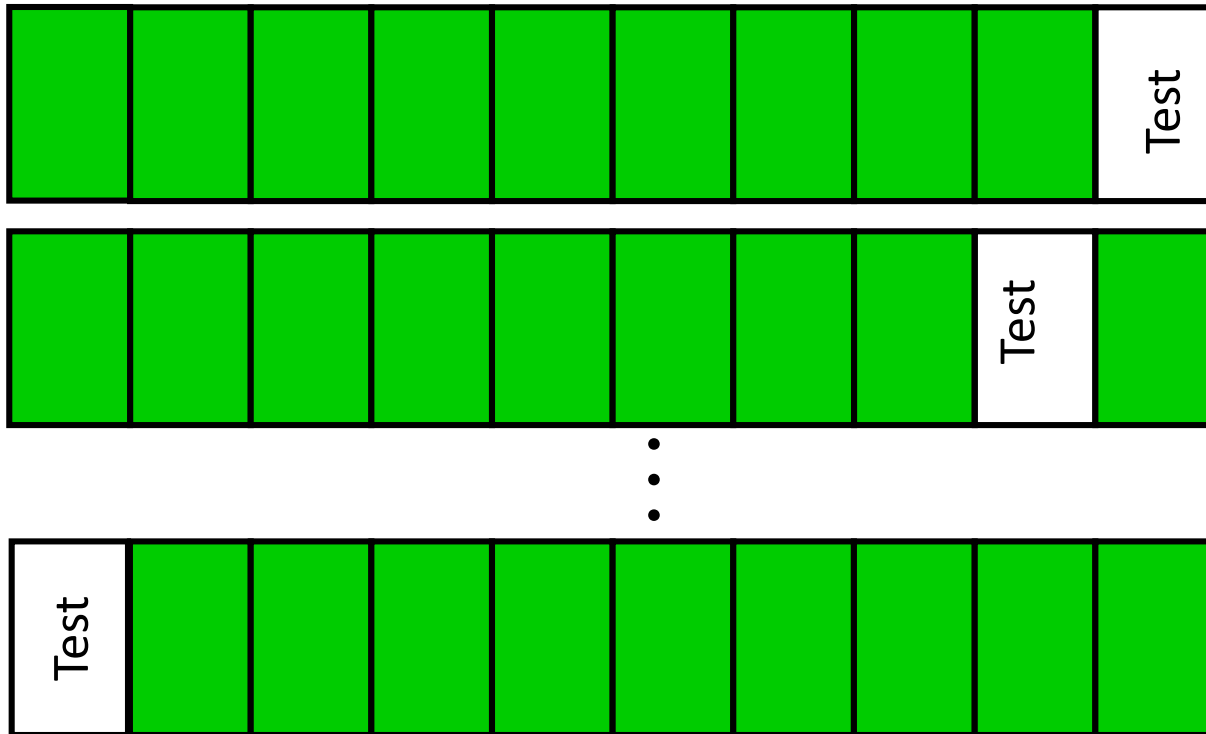
---

Question: How do we estimate the performance of classifier on unseen data?

- Can't just at accuracy on training data – this will yield an over optimistic estimate of performance
- Solution: **Cross-validation**
- Note: this is sometimes called estimating how well the classifier will generalize

# Evaluation: Cross Validation

- Partition examples into  $k$  disjoint sets
- Now create  $k$  training sets
  - Each set is union of all equiv classes *except one*
  - So each set has  $(k-1)/k$  of the original training data





# Cross-Validation (2)

---

- Leave-one-out
  - Use if  $< 100$  examples (rough estimate)
  - Hold out one example, train on remaining examples
- 10-fold
  - If have 100-1000's of examples
- M of N fold
  - Repeat M times
  - Divide data into N folds, do N fold cross-validation

# Evaluation Metrics

---

- **Accuracy**: no. of questions correctly answered
- **Precision** (for one label): accuracy when classification = label
- **Recall** (for one label): measures how many instances of a label were missed.
- **F-measure** (for one label): harmonic mean of precision & recall.
- **Area under Precision-recall curve** (for one label): vary parameter to show different points on p-r curve; take the area

# Precision & Recall

---

Two class situation

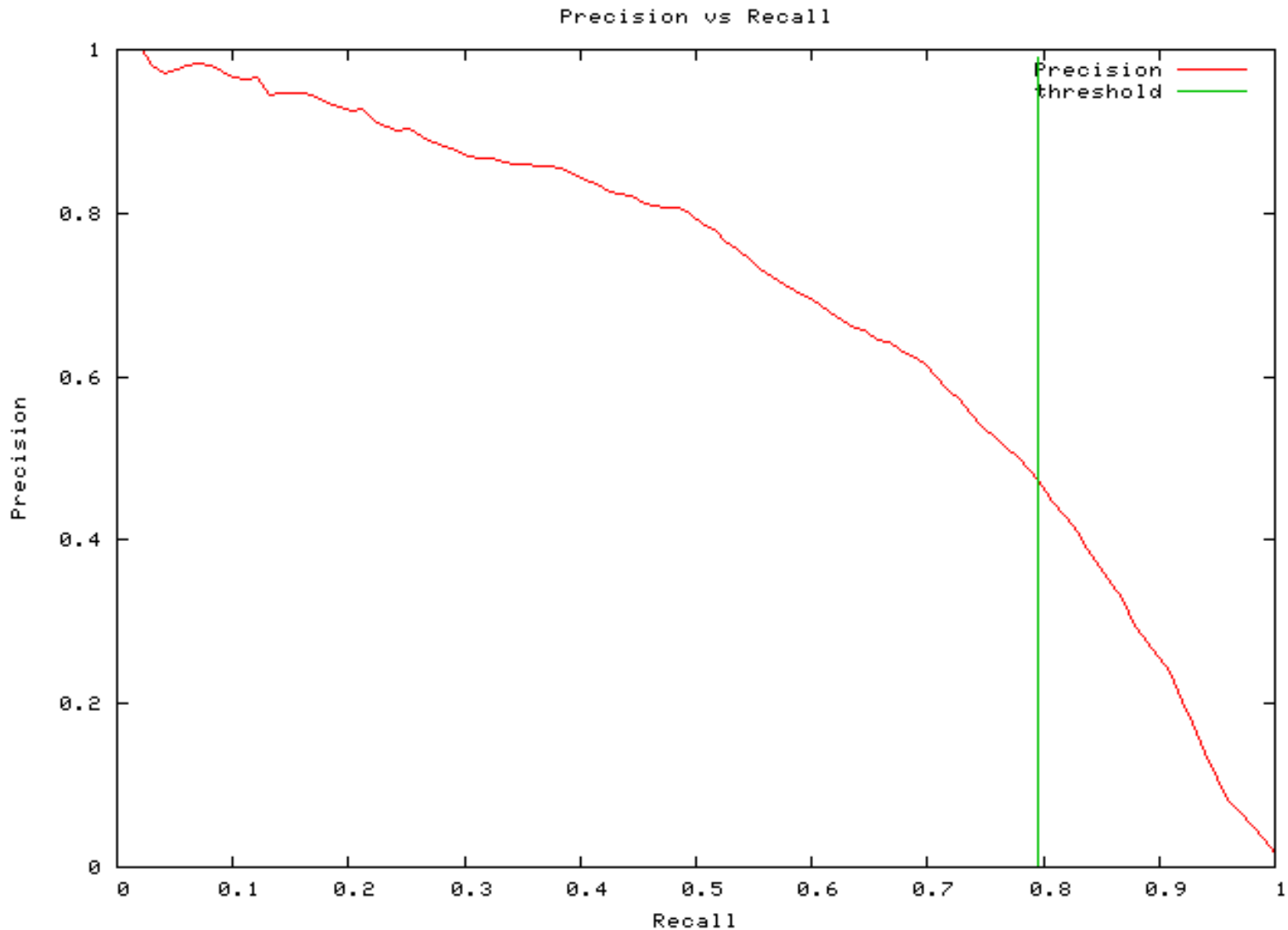
	Predicted	
	“P”	“N”
Actual	P	TP FN
	N	FP TN

$$\text{Precision} = \text{TP}/(\text{TP}+\text{FP})$$

$$\text{Recall} = \text{TP}/(\text{TP}+\text{FN})$$

$$\text{F-measure} = 2pr/(p+r)$$

# A typical precision-recall curve



# Construct Better Features

---

- Key to machine learning is having good features
- In gen 2 ML, large effort devoted to constructing appropriate features
- Ideas??

# Issues in document representation

---

Cooper's concordance of Wordsworth was published in 1911. The applications of full-text retrieval are legion: they include résumé scanning, litigation support and searching published journals on-line.

- *Cooper's vs. Cooper vs. Coopers.*
- *Full-text vs. full text vs. {full, text} vs. fulltext.*
- *résumé vs. resume.*

# Punctuation

---

- *Ne'er*: use language-specific, handcrafted “locale” to normalize.
- *State-of-the-art*: break up hyphenated sequence.
- *U.S.A.* vs. *USA*
- *a.out*

# Numbers

---

- 3/12/91
- Mar. 12, 1991
- 55 B.C.
- B-52
- 100.2.86.144
  - Generally, don't represent as text
  - Creation dates for docs



# Possible Feature Ideas

---

- Look at capitalization (may indicated a proper noun)
- Look for commonly occurring sequences
  - E.g. New York, New York City
  - Limit to 2-3 consecutive words
  - Keep all that meet minimum threshold (e.g. occur at least 5 or 10 times in corpus)

# Case folding

---

- Reduce all letters to lower case
- Exception: upper case in mid-sentence
  - *e.g., General Motors*
  - *Fed vs. fed*
  - *SAIL vs. sail*

# Thesauri and Soundex

---

- Handle synonyms and spelling variations
  - Hand-constructed equivalence classes
    - e.g., *car* = *automobile*

# Spell Correction

---

- Look for all words within (say) edit distance 3 (Insert/Delete/Replace) at query time
  - *e.g., artificial intelligence*
- Spell correction is expensive and slows the processing significantly
  - Invoke only when index returns zero matches?

# Lemmatization

---

- Reduce inflectional/variant forms to base form
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*

*the boy's cars are different colors*

→

*the boy car be different color*

# Stemming

---

- Are there different index terms?
  - retrieve, retrieving, retrieval, retrieved, retrieves...
- Stemming algorithm:
  - (retrieve, retrieving, retrieval, retrieved, retrieves) ⇒ **retriev**
  - Strips prefixes of suffixes (-s, -ed, -ly, -ness)
  - Morphological stemming
- Problems: sand / sander & wand / wander

# Stemming Continued

---

- Can reduce vocabulary by  $\sim 1/3$
- C, Java, Perl versions, python, c#  
[www.tartarus.org/~martin/PorterStemmer](http://www.tartarus.org/~martin/PorterStemmer)
- Criterion for removing a suffix
  - Does "a document is about  $w_1$ " mean the same as
  - a "a document about  $w_2$ "
- Problems: sand / sander & wand / wander
- Commercial SEs use giant in-memory tables

# Stemming vs Lemmatization

---

Stemming reduces word-forms to (pseudo)stems, whereas lemmatization reduces the word-forms to linguistically valid lemmas. This difference is apparent in languages with more complex morphology, but may be irrelevant for many IR applications;

Lemmatization deals only with inflectional variance, whereas stemming may also deal with derivational variance;

In terms of implementation, lemmatization is usually more sophisticated (especially for morphologically complex languages) and usually requires some sort of lexica. Satisfactory stemming, on the other hand, can be achieved with rather simple rule-based approaches.



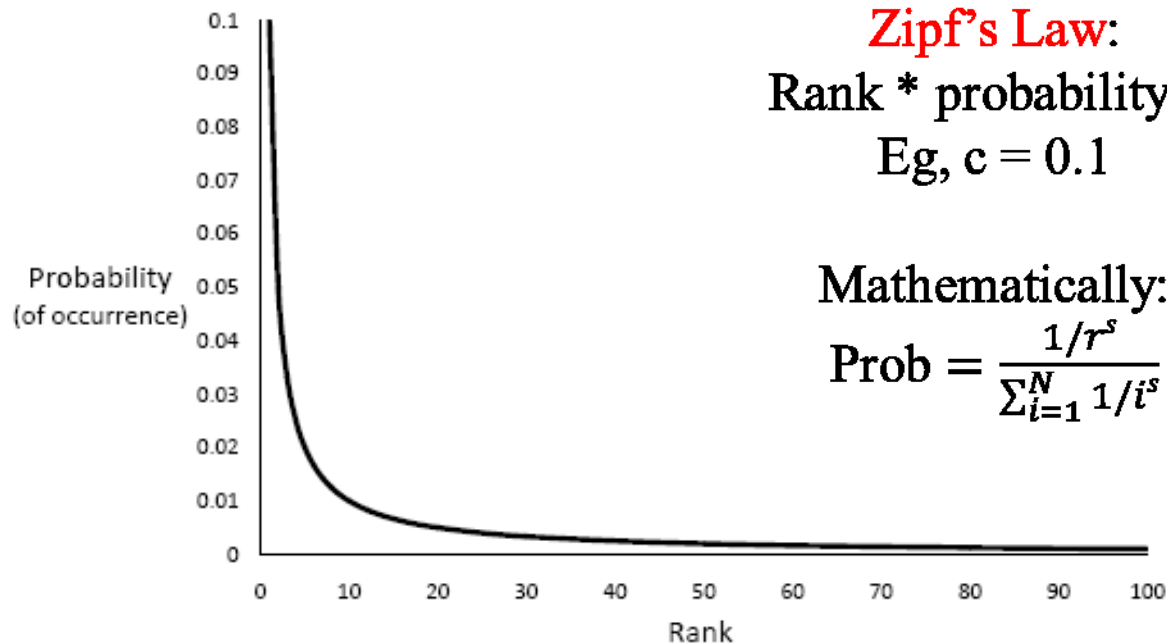
# Features

Sec 15.3.2

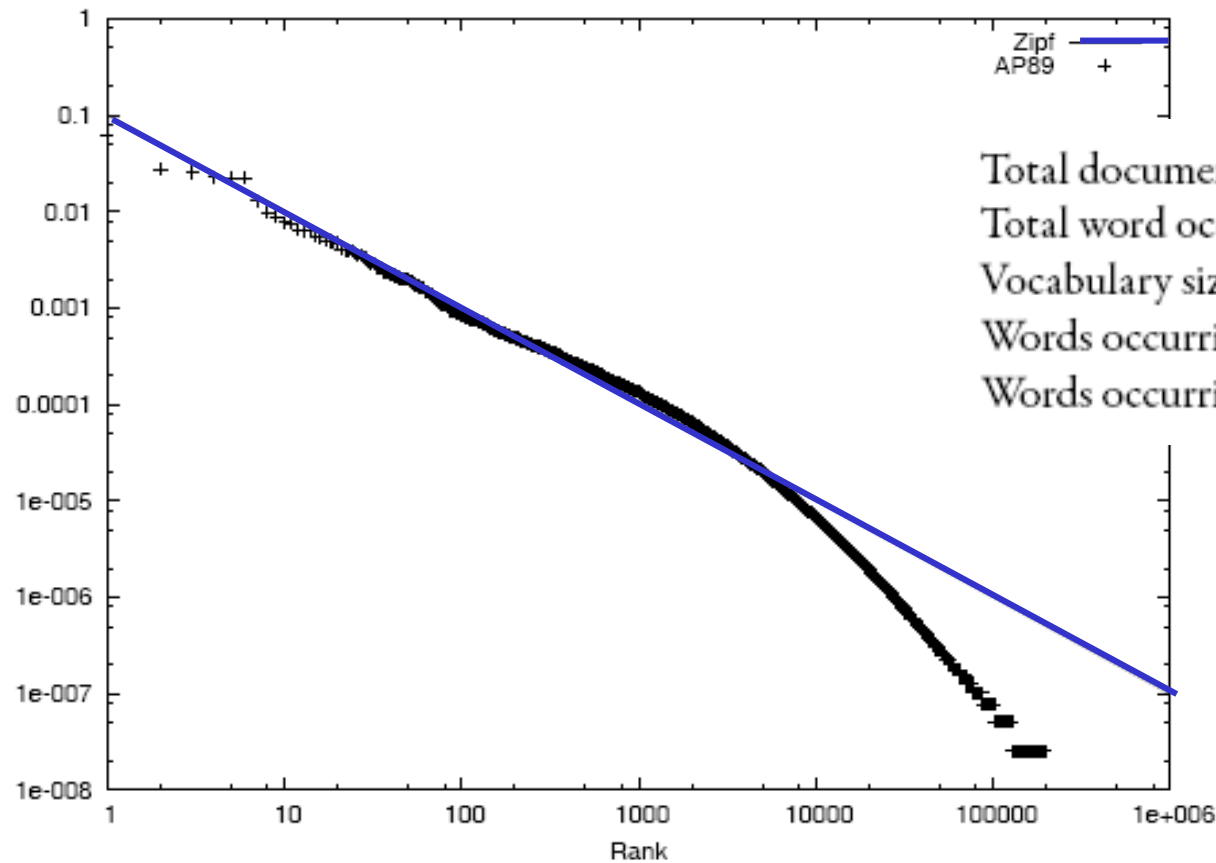
- 
- Domain-specific features and weights: *very* important in real performance
  - Upweighting: Counting a word as if it occurred twice:
    - title words (Cohen & Singer 1996)
    - first sentence of each paragraph (Murata, 1999)
    - In sentences that contain title words (Ko *et al*, 2002)

# Properties of Text

- Word frequencies - skewed distribution
- 'The' and 'of' account for 10% of all words
- Six most common words account for 40%



# Associate Press Corpus 'AP89'



Total documents	84,678
Total word occurrences	39,749,179
Vocabulary size	198,763
Words occurring > 1000 times	4,169
Words occurring once	70,064

# Middle Ground

---

- Very common words → bad features

- Language-based stop list:

words that bear little meaning

20-500 words

[http://www.dcs.gla.ac.uk/idom/ir\\_resources/linguistic\\_utils/stop\\_words](http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words)

- Subject-dependent stop lists

- Very rare words *also* bad features

Drop words appearing less than k times / corpus

# Word Frequency

---

- Which word is more indicative of document similarity?
  - ‘book,’ or ‘Rumpelstiltskin’?
  - Need to consider “**document frequency**”--- how frequently the word appears in doc collection.
- Which doc is a better match for the query “Kangaroo”?
  - One with a single mention of Kangaroos... or a doc that mentions it 10 times?
  - Need to consider “**term frequency**”--- how many times the word appears in the current document.

# TF x IDF

---

$$w_{ik} = tf_{ik} * \log(N / n_k)$$

$T_k$  = term  $k$  in document  $D_i$

$tf_{ik}$  = frequency of term  $T_k$  in document  $D_i$

$idf_k$  = inverse document frequency of term  $T_k$  in  $C$

$$idf_k = \log\left(\frac{N}{n_k}\right)$$

$N$  = total number of documents in the collection  $C$

$n_k$  = the number of documents in  $C$  that contain  $T_k$

# Inverse Document Frequency

---

- IDF provides high values for rare words and low values for common words

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{5000}\right) = 0.301$$

$$\log\left(\frac{10000}{20}\right) = 2.698$$

$$\log\left(\frac{10000}{1}\right) = 4$$

- Add 1 to avoid 0.

# TF-IDF normalization

---

- Normalize the term weights
  - so longer docs not given more weight (fairness)
  - force all values to fall within a certain range: [0, 1]

$$w_{ik} = \frac{tf_{ik} (1 + \log(N / n_k))}{\sqrt{\sum_{k=1}^t (tf_{ik})^2 [1 + \log(N / n_k)]^2}}$$



# Term-Weighted Complement Naïve Bayes

- Let  $\vec{d} = (d_1, \dots, d_n)$  be a set of documents;  $d_{ij}$  is the count of word  $i$  in document  $j$ .
- Let  $\vec{y} = (y_1, \dots, y_n)$  be the labels.
- $\text{TWCNB}(\vec{d}, \vec{y})$

$$d_{ij} = \log(d_{ij} + 1) \text{ (TF transform § 4.1)}$$

$$d_{ij} = d_{ij} \log \frac{\sum_k 1}{\sum_k \delta_{ik}} \text{ (IDF transform § 4.2)}$$

$$d_{ij} = \frac{d_{ij}}{\sqrt{\sum_k (d_{kj})^2}} \text{ (length norm. § 4.3)}$$

$$\hat{\theta}_{ci} = \frac{\sum_{j: y_j \neq c} d_{ij} + \alpha_i}{\sum_{j: y_j \neq c} \sum_k d_{kj} + \alpha} \text{ (complement § 3.1)}$$

$$w_{ci} = \log \hat{\theta}_{ci}$$

Let  $t = (t_1, \dots, t_n)$  be a test document; let  $t_i$  be the count of word  $i$ .

Label the document according to

$$l(t) = \arg \min_c \sum_i t_i w_{ci}$$

Tackling the Poor Assumptions of Naive Bayes Text Classifiers

Jason D. M. Rennie  
Lawrence Shih  
Jaime Teevan  
David R. Karger

Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139

JRENNIE@MIT.EDU  
KAI@MIT.EDU  
TEEVAN@MIT.EDU  
KARGER@MIT.EDU

---

# Multi-class Problems

# Evaluation:

## Classic Reuters-21578 Data Set

Sec 15.2.4

- Most (over)used data set, 21,578 docs (each 90 types, 200 tokens)
- 9603 training, 3299 test articles (ModApte/Lewis split)
- 118 categories
  - An article can be in more than one category
  - Learn 118 binary category distinctions
- Average document (with at least one category) has 1.24 classes
- Only about 10 out of 118 categories are large

Common categories  
(#train, #test)

- |                            |                       |
|----------------------------|-----------------------|
| • Earn (2877, 1087)        | • Trade (369,119)     |
| • Acquisitions (1650, 179) | • Interest (347, 131) |
| • Money-fx (538, 179)      | • Ship (197, 89)      |
| • Grain (433, 149)         | • Wheat (212, 71)     |
| • Crude (389, 189)         | • Corn (182, 56)      |

# Reuters Text Categorization data set (Reuters-21578) document

Sec 1524

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="12981"  
NEWID="798">

<DATE> 2-MAR-1987 16:51:43.42</DATE>

<TOPICS><D>livestock</D><D>hog</D></TOPICS>

<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>

<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Pork Congress kicks off tomorrow, March 3, in Indianapolis with 160 of the nations pork producers from 44 member states determining industry positions on a number of issues, according to the National Pork Producers Council, NPPC.

Delegates to the three day Congress will be considering 26 resolutions concerning various issues, including the future direction of farm policy and the tax law as it applies to the agriculture sector. The delegates will also debate whether to endorse concepts of a national PRV (pseudorabies virus) control and eradication program, the NPPC said.

A large trade show, in conjunction with the congress, will feature the latest in technology in all areas of the industry, the NPPC added. Reuter

&#3;</BODY></TEXT></REUTERS>

# Precision & Recall

## Two class situation

		Predicted	
		“P”	“N”
Actual	P	TP	FN
	N	FP	TN

## Multi-class situation:

Classname	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
alt.atheism	1	6																		
soc.religion.christian	2	9	0																	
sci.space	3		1	1																
talk.politics.misc	4			3	24															
talk.religion.misc	5				23	0														
rec.autos	6					1	0													
comp.windows.x	7						1	0												
talk.politics.mideast	8							0	0											
sci.crypt	9								0											
rec.motorcycles	10									0										
comp.graphics	11										0									
comp.sys.ibm.pc.hardware	12											0	23							
comp.sys.mac.hardware	13												10	8						
sci.electronics	14													13	6					
misc.forsale	15														8	1				
sci.med	16															2	0			
comp.os.mswindows.misc	17																0	1		
rec.sport.baseball	18																	0	1	
talk.politics.guns	19																		1	0
rec.sport.hockey	20																			0

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{F-measure} = 2pr / (p+r)$$

# Micro-- vs. Macro--Averaging

---

- If we have more than one class, how do we combine multiple performance measures into one quantity?
- Macroaveraging
  - Compute performance for each class, then average.
- Microaveraging
  - Collect decisions for all classes, compute contingency table, evaluate

# Precision & Recall

Multi-class Multi-label situation:

Classname	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
alt.atheism	1	6	1	3	32	1	1	2	1	2	0	0	0	0	0	0	0	0	0	0
soc.religion.christian	2	9	0	1	6	0	0	1	0	0	0	0	0	1	2	2	0	0	0	1
sci.space	3	3	1	1	0	1	2	0	1	1	9	0	0	1	2	3	0	0	1	1
talk.politics.misc	4	2	0	3	24	3	0	17	3	0	0	0	0	0	0	1	0	1	33	0
talk.religion.misc	5	88	36	2	23	0	1	0	0	0	0	0	0	0	0	2	0	1	15	0
rec.autos	6	0	0	0	3	1	0	0	0	7	1	2	1	6	4	1	0	0	2	0
comp.windows.x	7	1	1	2	1	0	1	0	2	2	30	5	3	1	1	2	1	1	0	0
talk.politics.mideast	8	0	3	1	18	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0
sci.crypt	9	1	0	1	2	1	0	3	0	0	3	0	1	0	0	1	0	0	3	0
rec.motorcycles	10	0	0	0	1	0	4	1	0	0	1	2	0	1	2	1	0	0	1	0
comp.graphics	11	0	1	2	1	1	0	10	1	2	0	23	7	3	3	3	0	0	0	0
comp.sys.ibm.pc.hardware	12	0	0	0	0	0	2	7	0	1	0	5	23	12	3	1	3	0	0	0
comp.sys.mac.hardware	13	0	0	1	1	0	2	1	0	0	0	7	10	8	9	1	0	0	0	0
sci.electronics	14	1	0	1	0	1	5	2	0	2	0	7	13	13	6	3	0	1	0	0
misc.forsale	15	0	1	4	2	0	12	1	0	0	4	1	19	10	8	1	0	1	1	2
sci.med	16	0	1	5	0	1	1	0	0	0	1	2	0	2	7	2	0	1	1	1
comp.os.mswindows.misc	17	1	0	2	0	1	1	58	1	3	0	38	71	17	3	6	0	0	1	0
rec.sport.baseball	18	2	1	1	0	0	0	0	0	0	0	4	0	0	0	1	1	0	1	7
talk.politics.guns	19	0	0	0	9	5	1	0	0	1	0	0	0	0	1	0	0	1	1	0
rec.sport.hockey	20	0	1	0	0	0	1	0	0	0	2	0	0	1	1	0	0	0	3	0

**Aggregate**

$$\text{Average Macro Precision} = \Sigma p_i / N$$

$$\text{Average Macro Recall} = \Sigma r_i / N$$

$$\text{Average Macro F-measure} = 2p_M r_M / (p_M + r_M)$$

$$\text{Average Micro Precision} = \Sigma TP_i / \Sigma_i Col_i$$

$$\text{Average Micro Recall} = \Sigma TP_i / \Sigma_i Row_i$$

$$\text{Average Micro F-measure} = 2p_\mu r_\mu / (p_\mu + r_\mu)$$

$$\text{Precision(class } i) = TP_i / (TP_i + FP_i)$$

$$\text{Recall(class } i) = TP_i / (TP_i + FN_i)$$

$$\text{F-measure(class } i) = 2p_i r_i / (p_i + r_i)$$

$$\text{Precision(class 1)} = 251 / (\text{Column}_1)$$

$$\text{Recall(class 1)} = 251 / (\text{Row}_1)$$

$$\text{F-measure(class 1)} = 2p_1 r_1 / (p_1 + r_1)$$

# Precision & Recall

Multi-class situation:

Missed predictions

Aggregate

$$\text{Average Macro Precision} = \Sigma p_i / N$$

$$\text{Average Macro Recall} = \Sigma r_i / N$$

$$\text{Average Macro F-measure} = 2p_M r_M / (p_M + r_M)$$

$$\text{Average Micro Precision} = \Sigma TP_i / \Sigma_i Col_i$$

$$\text{Average Micro Recall} = \Sigma TP_i / \Sigma_i Row_i$$

$$\text{Average Micro F-measure} = 2p_\mu r_\mu / (p_\mu + r_\mu)$$

*Aren't  $\mu$  prec and  $\mu$  recall the same?*

Classifier hallucinations

Classname	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
alt.atheism	1	6	1	3	32	1	1	2	1	2	0	0	0	0	0	0	0	0	0	0
soc.religion.christian	2	9	0	1	6	0	0	1	0	0	0	0	0	1	2	2	0	0	0	1
sci.space	3	3	1	1	0	1	2	0	1	1	9	0	0	1	2	3	0	0	1	1
talk.politics.misc	4	2	0	3	24	3	0	17	3	0	0	0	0	0	0	1	0	1	33	0
talk.religion.misc	5	88	36	2	23	0	1	0	0	0	0	0	0	0	0	2	0	1	15	0
rec.autos	6	0	0	0	3	1	0	0	0	7	1	2	1	6	4	1	0	0	2	0
comp.windows.x	7	1	1	2	1	0	1	0	2	2	30	5	3	1	1	2	1	1	0	0
talk.politics.mideast	8	0	3	1	18	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0
sci.crypt	9	1	0	1	2	1	0	3	0	0	3	0	1	0	0	1	0	0	3	0
rec.motorcycles	10	0	0	0	1	0	4	1	0	0	1	2	0	1	2	1	0	0	1	0
comp.graphics	11	0	1	2	1	1	0	10	1	2	0	23	7	3	3	3	0	0	0	0
comp.sys.ibm.pc.hardware	12	0	0	0	0	0	2	7	0	1	0	5	23	12	3	1	3	0	0	0
comp.sys.mac.hardware	13	0	0	1	1	0	2	1	0	0	0	7	10	8	9	1	0	0	0	0
sci.electronics	14	1	0	1	0	1	5	2	0	2	0	7	13	13	6	3	0	1	0	0
misc.forsale	15	0	1	4	2	0	12	1	0	0	4	1	19	10	8	1	0	1	1	2
sci.med	16	0	1	5	0	1	1	0	0	0	1	2	0	2	7	2	0	1	1	1
comp.os.mswindows.misc	17	1	0	2	0	1	1	58	1	3	0	38	71	17	3	6	0	0	0	0
rec.sport.baseball	18	2	1	1	0	0	0	0	0	0	4	0	0	0	1	1	0	0	1	7
talk.politics.guns	19	0	0	0	9	5	1	0	0	1	0	0	0	0	1	0	0	1	1	0
rec.sport.hockey	20	0	1	0	0	0	1	0	0	0	2	0	0	1	1	0	0	0	3	0

$$\text{Precision(class } i) = TP_i / (TP_i + FP_i)$$

$$\text{Recall(class } i) = TP_i / (TP_i + FN_i)$$

$$\text{F-measure(class } i) = 2p_i r_i / (p_i + r_i)$$

$$\text{Precision(class 1) = } 251 / (\text{Column}_1)$$

$$\text{Recall(class 1) = } 251 / (\text{Row}_1)$$

$$\text{F-measure(class 1) = } 2p_i r_i / (p_i + r_i)$$



# Multi-Class Classification

---

- What?
  - Converting a k-class problem to a binary problem.
- Why?
  - For some ML algorithms, a direct extension to the multiclass case may be problematic.
- How?
  - Many methods

# Methods

---

- One-vs-all
- All-pairs
- Error-correcting Output Codes (ECOC)
- ...

# One-vs-all

---

- Idea:
- Create many 1 vs other classifiers
  - Classes = City, County, Country
  - Classifier 1 = {City} {County, Country}
  - Classifier 2 = {County} {City, Country}
  - Classifier 3 = {Country} {City, County}
- Training time:
  - For each class  $c_m$ , train a classifier  $cl_m(x)$ 
    - replace  $(x,y)$  with
      - $(x, 1)$  if  $y = c_m$
      - $(x, -1)$  if  $y \neq c_m$

# An example: training

---

- $x_1 \ c_1 \ \dots$
- $x_2 \ c_2 \ \dots$
- $x_3 \ c_1 \ \dots$
- $x_4 \ c_3 \ \dots$

for  $c_1$ -vs-all:

$x_1 \ 1 \ \dots$   
 $x_2 \ -1 \ \dots$   
 $x_3 \ 1 \ \dots$   
 $x_4 \ -1 \ \dots$

for  $c_2$ -vs-all:

$x_1 \ -1$   
 $x_2 \ 1 \ \dots$   
 $x_3 \ -1 \ \dots$   
 $x_4 \ -1 \ \dots$

for  $c_3$ -vs-all:

$x_1 \ -1 \ \dots$   
 $x_2 \ -1 \ \dots$   
 $x_3 \ -1 \ \dots$   
 $x_4 \ 1 \ \dots$

## One-vs-all (cont)

---

- Testing time: given a new example  $x$ 
  - Run each of the  $k$  classifiers on  $x$
  - Choose the class  $c_m$  with the highest confidence score  $cl_m(x)$ :

$$c^* = \arg \max_m cl_m(x)$$

# An example: testing

---

- $x_1 \ c_1 \ \dots$
- $x_2 \ c_2 \ \dots$
- $x_3 \ c_1 \ \dots$
- $x_4 \ c_3 \ \dots$

→ three classifiers

for c1-vs-all:

$x \ ?? \quad 1 \quad 0.7 \quad -1 \quad 0.3$

for c2-vs-all

$x \ ?? \quad 1 \quad 0.2 \quad -1 \quad 0.8$

for c3-vs-all

$x \ ?? \quad 1 \quad 0.6 \quad -1 \quad 0.4$

Test data:

$x \ ?? \ f_1 \ v_1 \ \dots$

⇒ what's the system prediction for  $x$ ?

# All-pairs (All-vs-All (AVA))

---

- Idea:
  - For each pair of classes build a classifier
  - {City vs. County}, {City vs Country}, {County vs. Country}
  - $C_k^2$  classifiers: one classifier for each class pair.
- Training:
  - For each pair  $(c_m, c_n)$  of classes, train a classifier  $cl_{mn}$ 
    - replace a training instance  $(x,y)$  with
      - $(x, 1)$  if  $y = c_m$
      - $(x, -1)$  if  $y = c_n$
      - otherwise ignore the instance

# An example: training

---

- $x_1 \ c_1 \ \dots$
- $x_2 \ c_2 \ \dots$
- $x_3 \ c_1 \ \dots$
- $x_4 \ c_3 \ \dots$

for  $c_1$ -vs- $c_2$ :

$x_1 \ 1 \ \dots$

$x_2 \ -1 \ \dots$

$x_3 \ 1 \ \dots$

for  $c_2$ -vs- $c_3$ :

$x_2 \ 1 \ \dots$

$x_4 \ -1 \ \dots$

for  $c_1$ -vs- $c_3$ :

$x_1 \ 1 \ \dots$

$x_3 \ 1 \ \dots$

$x_4 \ -1 \ \dots$



# All-pairs (cont)

---

- Testing time: given a new example  $x$ 
  - Run each of the  $C_k^2$  classifiers on  $x$
  - Max-win strategy: Choose the class  $c_m$  that wins the most pairwise comparisons:
  - Other coupling models have been proposed: e.g., (Hastie and Tibshirani, 1998)

# An example: testing

---

- $x_1$   $c_1$  ...
- $x_2$   $c_2$  ...
- $x_3$   $c_1$  ...
- $x_4$   $c_3$  ...

→ three classifiers

for  $c_1$ -vs- $c_2$ :

$x$  ??    1   0.7   -1   0.3

for  $c_2$ -vs- $c_3$

$x$  ??    1   0.2   -1   0.8

for  $c_1$ -vs- $c_3$

$x$  ??    1   0.6   -1   0.4

Test data:

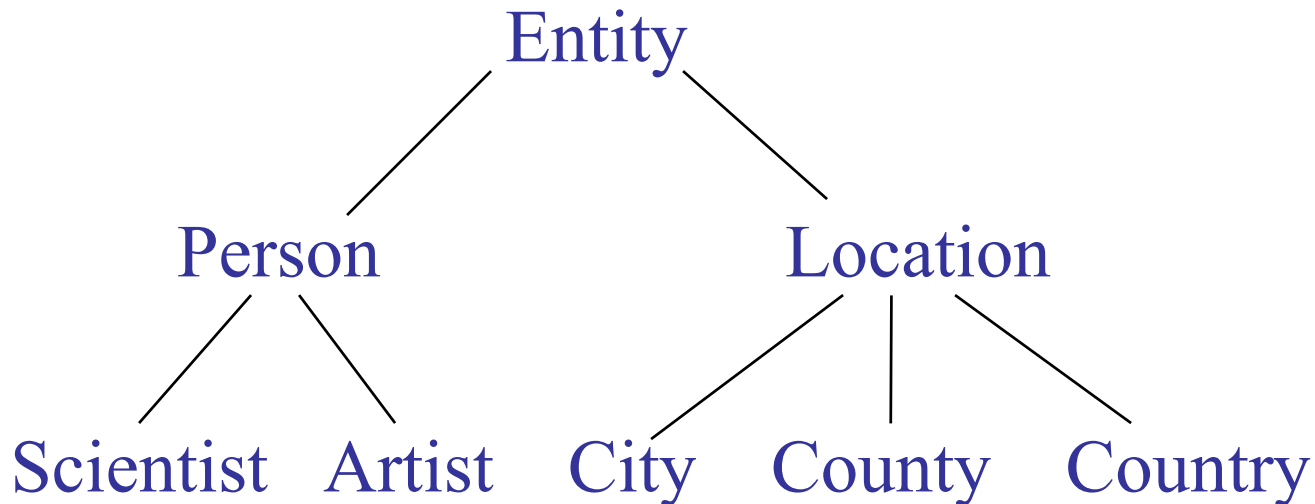
$x$  ??    $f_1$   $v_1$  ...

⇒ what's the system prediction for  $x$ ?

# Hierarchical Categorization

---

- Pick the category with max probability
- Create many OVA/AVA classifiers
- Use a hierarchical approach (wherever hierarchy available)



# Summary

---

- Different methods:
  - Direct multiclass, if possible
  - One-vs-all (a.k.a. one-per-class):  $k$ -classifiers
  - All-pairs:  $C_k^2$  classifiers
  - Hierarchical classification ( $\log C$  classifiers)
- Some studies report that All-pairs works better than one-vs-all.