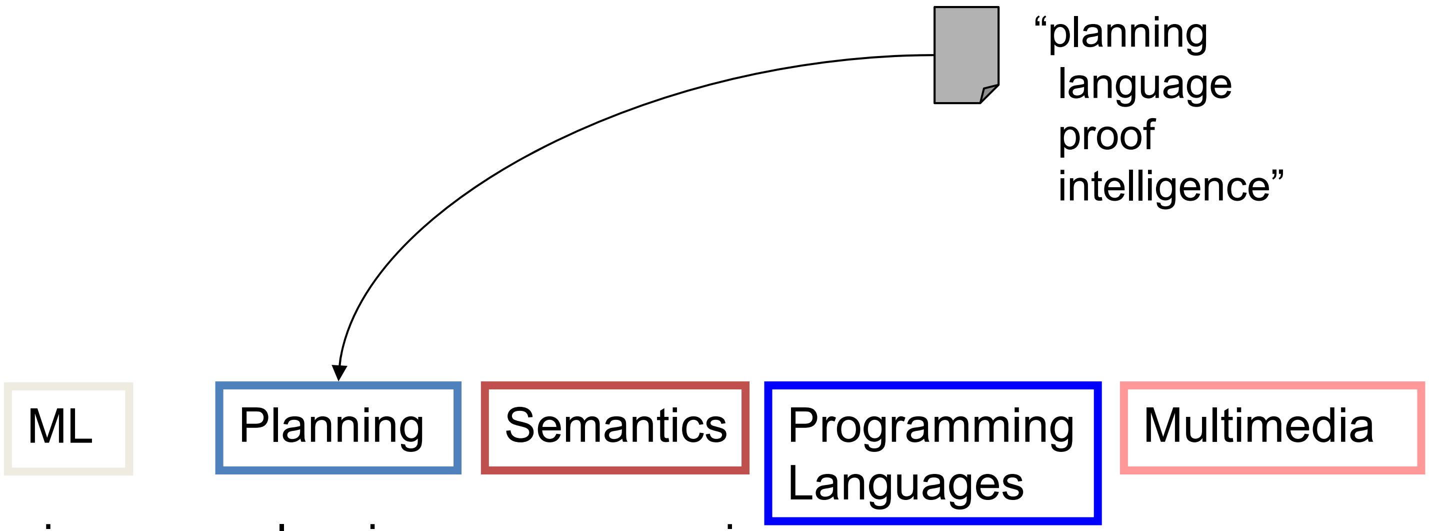
Text Classification

Naïve Bayes Algorithm

Reference: Introduction to Information Retrieval
by C. Manning, P. Raghavan, H. Schutze

Document Classification



Test Data:

“planning language proof intelligence”

Classes:

ML	Planning	Semantics	Programming Languages	Multimedia
learning intelligence algorithm reinforcement network...	planning temporal reasoning plan language...	programming semantics language proof...	garbage collection memory optimization region...	...

Training Data: (the word lists under each class above)

Categorization/Classification

- Given:
  - A description of an instance, $d \in X$
    - X is the *instance language* or *instance space*.
      - Issue: how to represent text documents.
      - Usually some type of high-dimensional space
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
- Determine:
  - The category of d: $\gamma(d) \in C$, where $\gamma(d)$ is a *classification function* whose domain is X and whose range is C.
    - We want to know how to build classification functions ("classifiers").

Supervised Classification

- Given:
  - A description of an instance, $d \in X$
    - X is the *instance language* or *instance space*.
  - A fixed set of classes:

    $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set D of labeled documents with each labeled document $\langle d,c\rangle \in X \times C$
- Determine:
  - A learning method or algorithm which will enable us to learn a classifier $\gamma: X \to C$
  - For a test document *d,* we assign it the class $\gamma(d) \in C$

More Text Classification Examples

Many search engine functionalities use classification

Assigning labels to documents or web-pages:

- Labels are most often topics
  - *"finance," "sports," "news"*
- Labels may be genres
  - *"editorials" "movie-reviews" "news”*
- Labels may be opinion on a person/product
  - *“like”, “hate”, “neutral”*
- Labels may be domain-specific
  - *"interesting-to-me" : "not-interesting-to-me”*
  - *“contains adult language” : “doesn’t”*
  - *language identification: English, French, Chinese, …*
  - *search vertical: about Linux versus not*
  - *“link spam” : “not link spam”*

Classification Methods

- Supervised learning of a document-label assignment function
  - Bayesian approach
  - Support-vector machines (SVM)
  - … plus many other methods
  - No free lunch: requires hand-classified training data
- Many commercial systems use a mixture of methods
- Bayesian text classification is widely employed for spam filtering
  - Solid theoretical foundation
  - Easy and efficient to learn
  - A principled way of combining prior information with data
  - Still explored in some recent works, e.g. "A correlation-Based Feature Weighting Filter for Naïve Bayes", IEEE Trans on Knowledge and Data Engineering (TKDE), 2019.

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:

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

Probabilistic Methods

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

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

$$P(c,d) = P(c \mid d)P(d) = P(d \mid c)P(c)$$

$$\boxed{P(c \mid d) = \frac{P(d \mid c)P(c)}{P(d)}}$$

Naive Bayes Classifiers

Task: Classify a new instance *d* based on a tuple of attribute values into one of the classes $c_j \in C$

$$d = \langle x_1, x_2, \ldots, x_n \rangle$$

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

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

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

MAP is “maximum a posteriori” = most likely class

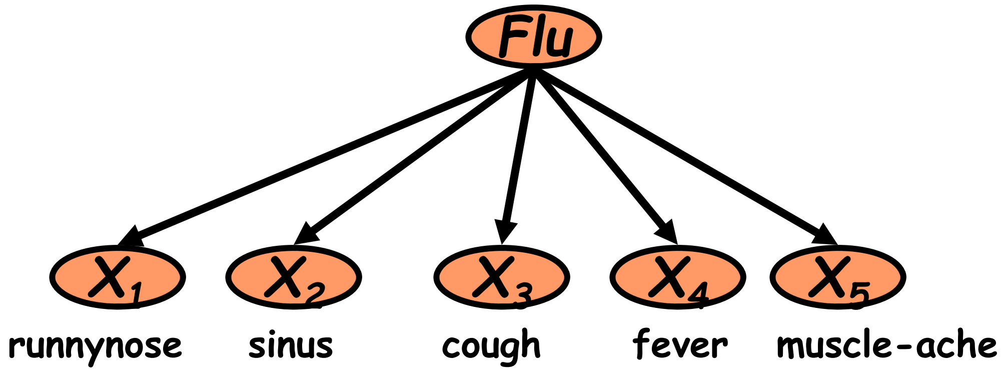
Naive Bayes Classifier: Naive Bayes Assumption

- $P(c_j)$
  - 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}•*|C|*) parameters
  - Could only be estimated if a very, very large number of training examples was available.

Naive 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 Naive Bayes Classifier



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

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

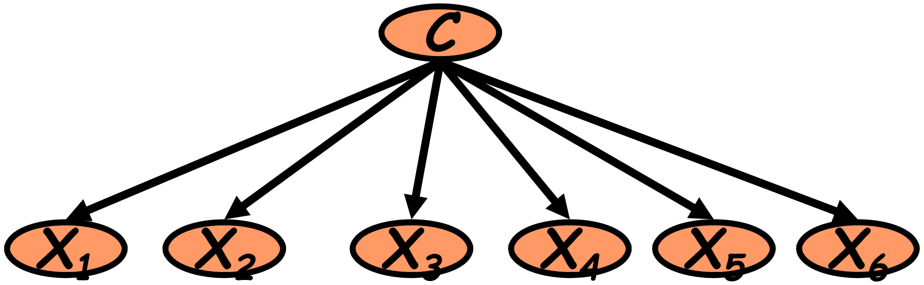
First Naive Bayes Model

- Model 1: Multivariate Bernoulli
  - One feature X_w for each word in dictionary
  - X_w = true if *w* appears in *d;* otherwise X_w = false
  - Naive Bayes assumption:
    - Given the document's class, appearance of one word in the document tells us nothing about chances that another word appears
- Model Learning

$$\hat{P}(X_w = true|c_j) = \begin{array}{c}\text{fraction of documents of class } c_j \\ \text{in which word } w \text{ appears}\end{array}$$

Multivariate Bernoulli Model

Learning the Model

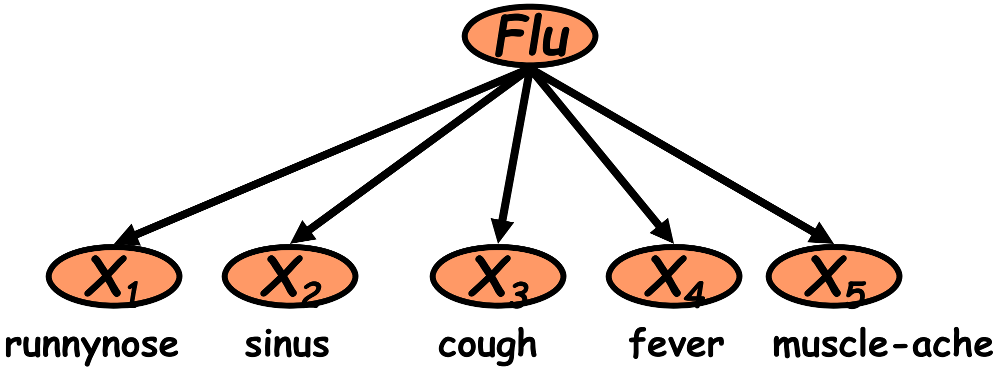


- First attempt: maximum likelihood estimates
  - simply use the frequencies in the data

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

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

Problem with Maximum Likelihood



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

- What if we have seen no training documents with the word ***muscle-ache*** and classified in the topic ***Flu***?

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

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

$$\ell = \arg\max_c \hat{P}(c)\prod_i \hat{P}(X_i = t \mid c)$$

Smoothing to Avoid Overfitting

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

# of values of X_i

$k = 2$ in this case

Bernoulli Naive Bayes Algorithm Learning (Training)

```
TRAINBERNOULLINB(C,D)
$V$ ← EXTRACTVOCABULARY(D)
$N$ ← COUNTDOCS(D)
for each $c \in$ C
do $N_C$ ← COUNTDOCSINCLASS(D, $c$)
    $prior[c] \leftarrow N_C / N$
    for each $t \in V$
    do $N_{ct}$ ← COUNTDOCSINCLASSCONTAININGTERM(D, $c$, $t$)
        $condprob[t][c] \leftarrow (N_{ct} + 1)/(N_c + 2)$
return $V$, $prior$, $condprob$
```

Bernoulli Naive Bayes Algorithm Classifying (Testing)

```
APPLYBERNOULLINB(C, $V$, prior, condprob, $d$)
$V_d \leftarrow$ EXTRACTTERMSFROMDOC($V$, $d$)
for each $c \in \mathbf{C}$
do $score[c] \leftarrow \log prior[c]$
    for each $t \in V$
    do if $t \in V_d$
        then $score[c] \mathrel{+}= \log condprob[t][c]$
        else $score[c] \mathrel{+}= \log(1 - condprob[t][c])$
return $\mathrm{argmax}_{c \in \mathbf{C}}\, score[c]$
```

Second Model

- Model 2: Multinomial = Class conditional unigram
  - One feature X_i for each word position in document
    - feature's values are all words in dictionary
  - Value of X_i is the word in position i
  - Naive Bayes assumption:
    - Given the document's class, word in one position in the document tells us nothing about words in other positions

Multinomial Naïve Bayes Model

- Can create a mega-document for class c_j by concatenating all documents in this class
- Use the frequency of *w* in mega-document

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

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Attributes are text positions, values are words.

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

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

- Still too many possibilities

Using Multinomial Naive Bayes Classifiers to Classify Text: Basic method

- Assume that classification is *independent* of the positions of the words
  - Use same parameters for each position
  - Result is bag-of-words model

- Word appearance does not depend on positions

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

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

- Just have one multinomial feature predicting all words

Multinomial Naive Bayes Learning Approach

- 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 $docs_j$
    ▪ $n \leftarrow$ total number of words in $Text_j$
    ▪ For each word $x_k$ in Vocabulary
        ▪ $n_k \leftarrow$ number of occurrences of $x_k$ in $Text_j$
        ▪ $P(x_k \mid c_j) \leftarrow \frac{n_k + 1}{n + |Vocabulary|}$
```

Multinomial Naive Bayes Classifying (Testing) Approach

- *positions* ← all word positions in current document which contain tokens found in *Vocabulary*

- Return c_{NB}, where

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

Multinomial Naive Bayes: Example

	docID	words in document	in c = China?
Training set	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
Test set	5	Chinese Chinese Chinese Tokyo Japan	?

$$P(c) = \frac{3}{4} \qquad P(\bar{c}) = \frac{1}{4}$$

$$P(\text{Chinese}|c) = \frac{(5+1)}{(8+6)} = \frac{6}{14} = \frac{3}{7} \qquad P(\text{Toyk}o|c) = P(\text{Japan}|c) = \frac{(0+1)}{(8+6)} = \frac{1}{14}$$

$$P(\text{Chinese}|\bar{c}) = \frac{(1+1)}{(3+6)} = \frac{2}{9} \qquad P(\text{Toyk}o|\bar{c}) = P(\text{Japan}|\bar{c}) = \frac{(1+1)}{(3+6)} = \frac{2}{9}$$

Multinomial Naive Bayes: Example

$$P(c) = \frac{3}{4} \qquad P(\bar{c}) = \frac{1}{4}$$

$$P(\text{Chinese}|c) = \frac{(5+1)}{(8+6)} = \frac{6}{14} = \frac{3}{7} \qquad P(\text{Toyk}o|c) = P(\text{Japan}|c) = \frac{(0+1)}{(8+6)} = \frac{1}{14}$$

$$P(\text{Chinese}|\bar{c}) = \frac{(1+1)}{(3+6)} = \frac{2}{9} \qquad P(\text{Toyk}o|\bar{c}) = P(\text{Japan}|\bar{c}) = \frac{(1+1)}{(3+6)} = \frac{2}{9}$$

$$P(c|d_5) \propto \frac{3}{4} \cdot \left(\frac{3}{7}\right)^3 \cdot \frac{1}{14} \cdot \frac{1}{14} \approx 0.0003$$

$$P(\bar{c}|d_5) \propto \frac{1}{4} \cdot \left(\frac{2}{9}\right)^3 \cdot \frac{2}{9} \cdot \frac{2}{9} \approx 0.0001$$

The classifier assigns the test document to $c = \text{China}$

Multinomial Naive Bayes Algorithm Learning (Training)

```
TrainMultinomialNB(C,D)
V ← ExtractVocabulary(D)
N ← CountDocs(D)
for each $c \in$ C
do $N_c$ ← CountDocsInClass(D, c)
    prior[c] ← $N_c/N$
    $text_c$ ← ConcatenateTextOfAllDocsInClass(D, c)
    for each $t \in V$
    do $T_{ct}$ ← CountTokensOfTerm($text_c$, t)
    for each $t \in V$
    do condprob[t][c] ← $\frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
return V, prior, condprob
```

Multinomial Naive Bayes Algorithm Classifying (Testing)

```
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 $\mathrm{argmax}_{c \in \mathbf{C}} score[c]$
```

Underflow Prevention: using logs

- 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.

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

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

Naive Bayes Classifier

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

- Simple interpretation: Each conditional parameter log *P*(x_i|c_j) is a weight that indicates how good an indicator x_i is for c_j.
- The prior log *P*(c_j) is a weight that indicates the relative frequency of c_j.
- The sum is then a measure of how much evidence there is for the document being in the class.
- We select the class with the most evidence for it

Feature Selection: Why?

- Text collections have a large number of features
  - 10,000 – 1,000,000 unique words … and more
- May allow 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 (χ^2)
  - Information theory:
    - How much information does the value of one categorical variable give you about the value of another
    - Mutual information
- They're similar, but χ^2 measures confidence in association, (based on available statistics), while MI measures extent of association (assuming perfect knowledge of probabilities)

Feature Selection

- 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

χ^2 Statistic (CHI)

- $\chi2$ 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?

$$\chi^2(Feature) = \sum (O-E)^2/E = (2-.25)^2/.25 + (3-4.75)^2/4.75$$

$$+ (500-502)^2/502 + (9500-9498)^2/9498 = 12.9 \ (p < .001)$$

- The null hypothesis is rejected with confidence .999, since 12.9 > 10.83 (the value for .999 confidence).
- Higher $\chi2$ values imply higher dependency among the word *w* and the class

	Word w appeared	*Word w not appeared*	
Class = auto	2 *(0.25)*	500 *(502)*	502
Class ≠ auto	3 *(4.75)*	9500 *(9498)*	9503
	5	10000	

expected: f_e

(5/10005)*(502/10005)*10005 = 0.2509

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 w and a class c is:

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

where e_w =1 when the document contains the word w (0 otherwise); e_c = 1 when the document is in class c (0 otherwise)

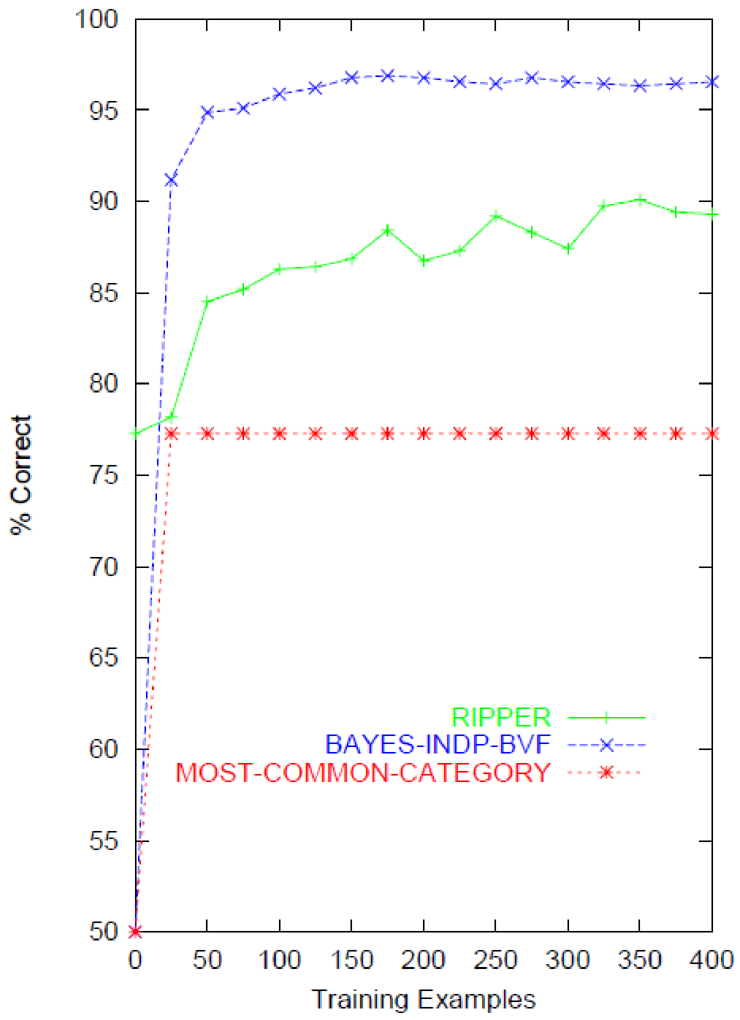
Feature Selection

- 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 uninformative terms
- 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 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

Naive Bayes on spam email



SpamAssassin

- Naive 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

Evaluating Categorization

- Evaluation must be done on test data that are independent of the training data (usually a disjoint set of instances).
  - It’s easy to get good performance on a test set that was available to the learner during training (e.g., just memorize the test set).
  - The holdout method reserves a certain amount for testing and uses the remainder for training
  - Usually: one third for testing, the rest for training

Evaluation Metric

- Metrics (Measures): classification accuracy, precision, recall, F1
- *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.
  - Assuming one class per document

Per class evaluation measures

- Given a class *i*, treat it as a binary classification problem.
- Recall: Fraction of docs in class *i* classified correctly
- Precision: Fraction of docs assigned class *i* that are actually about class *i*
- Accuracy: (1 - error rate) Fraction of all docs classified correctly with respect to class *i*

A combined measure: *F*

- Combined measure that assesses precision/recall tradeoff is **F measure** (weighted harmonic mean):

$$F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

- People usually use balanced F_1 measure
  - i.e., with $\beta = 1$ or $\alpha = ½$

Sec. 15.2.4

Micro- vs. Macro-Averaging

- Handling the evaluation of more than one class
- Macroaveraging: Compute performance for each class, then average.
- Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.

Micro- vs. Macro-Averaging: Example

Class 1

	Truth: yes	Truth: no
Classifier: yes	10	10
Classifier: no	10	970

Class 2

	Truth: yes	Truth: no
Classifier: yes	90	10
Classifier: no	10	890

Micro Ave. Table

	Truth: yes	Truth: no
Classifier: yes	100	20
Classifier: no	20	1860

- Macroaveraged precision: (0.5 + 0.9)/2 = 0.7
- Microaveraged precision: 100/120 = .83
- Microaveraged score is dominated by score on common classes

Cross-validation

- Cross-validation - averaging results over multiple training and test splits of the overall data
- Cross-validation avoids overlapping test sets
  - First step: data is split into k subsets of equal size
  - Second step: each subset in turn is used for testing and the remainder for training
- This is called *k-fold cross-validation*
- The error estimates are averaged to yield an overall error estimate

Cross-validation

- Split the available data set into k equal partitions, namely, P_1, … P_k

Training set	Testing set	Accuracy
$P_2, \dots , P_k$	P_1	A_1
$P_1, P_3, \dots , P_k$	P_2	A_2
:	:	
$P_1, P_2 , \dots , P_{k-1}$	P_k	A_k
	Average Accuracy	A

Violation of NB Assumptions

- The independence assumptions do not really hold of documents written in natural language.
  - Conditional independence
  - Positional independence

Naive Bayes Posterior Probabilities

- Classification results of naive 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

- More robust to irrelevant features than many learning methods

  Irrelevant Features cancel each other without affecting results

  Decision Trees can suffer heavily from this.

- More robust to concept drift (changing class definition over time)
- Very good in domains with many equally important features

  Decision Trees suffer from *fragmentation* in such cases – especially if little data

- Optimal if the Independence Assumptions hold: Bayes Optimal Classifier

  Never true for text, but possible in some domains

- Very Fast Learning and Testing (basically just count the data)
- Low Storage requirements