Evaluation for Text Categorization

- Classification accuracy:
  - usual in ML,
  - the proportion of correct decisions,
  - Not appropriate if the population rate of the class is low
- Precision, Recall and F_1
  - Better measures

Evaluation for sets of classes

- How can we combine evaluation w.r.t. single classes into an evaluation for prediction over multiple classes?
- Two aggregate measures
  - Macro-Averaging, computes a simple average over the classes of the precision, recall, and F_1 measures
  - Micro-Averaging, pools per-doc decisions across classes and then compute precision, recall, and F_1 on the pooled contingency table

Macro and Micro Averaging

- Macro-averaging gives the same weight to each class
- Micro-averaging gives the same weight to each per-doc decision

Example

Class 1	Truth: "yes"	Truth: "no"
Pred: "yes"	10	10
Pred: "no"	10	970

Class 2	Truth: "yes"	Truth: "no"
Pred: "yes"	90	10
Pred: "no"	10	890

POOLED	Truth: "yes"	Truth: "no"
Pred: "yes"	100	20
Pred: "no"	20	1860

Macro-Averaged Precision: (.5+.9)/2 = .7

Micro-averaged Precision: 100/120 = .833…

Benchmark Collections (used in Text Categorization)

Reuters-21578

- The most widely used in text categorization. It consists of newswire articles which are labeled with some number of topical classifications (zero or more out of 115 classes). 9603 train + 3299 test documents

Reuters RCV1

- Newstories, larger than the previous (about 810K documents) and a hierarchically structured set of (103) leaf classes

Oshumed

- a ML set of 348K docs classified under a hierarchically structured set of 14K classes (MESH thesaurus). Title+abstracts of scientific medical papers.

20 Newsgroups

- 18491 articles from the 20 Usenet newsgroups

The inductive construction of classifiers

Two different phases to build a classifier h_i for category $c_i \in C$

1. Defintion of a function $CSV_i : D \rightarrow \mathcal{R}$, a categorization status value, representing the strength of the evidence that a given document d_j belongs to c_i
2. Definition of a threshold τ_i such that
   - $CSV_i(d_j) \geq \tau_i$ interpreted as a decision to classify d_j under c_i
   - $CSV_i(d_j) \leq \tau_i$ interpreted as a decision not to classify d_j under c_i

CSV and Proportional thresholding

- Two different ways to determine the thresholds τ_i once given CSV_i are [Yang01]

  1. CSV thresholding: τ_i is a value returned by the CSV_i function. May or may not be equal for all the categories. Obtained on a validation set
  2. Proportional thresholding: τ_i are the values such that the validation set frequencies for each class is as close as possible to the same frequencies in the training set

- CSV thresholding is theoretically better motivated, and generally produce superior effectiveness, but computationally more expansive
- Thresholding is needed only for ‘hard’ classification. In ‘soft’ classification the decision is taken by the expert, and the CSV_i scores can be used for ranking purposes

Probabilistic Classifiers

- Probabilistic classifiers view $CSV_j(d_i)$ in terms of $P(c_j|d_i)$, and compute it by means of the Bayes' theorem
  - $P(c_j|d_i) = P(d_i|c_j)P(c_j)/P(d_i)$
  - Maximum a posteriori Hypothesys (MAP) argmax $P(c_j|d_i)$
- Classes are viewed as generators of documents
- The prior probability $P(c_j)$ is the probability that a document d is in c_j

Naive Bayes 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 $c_j \in C$

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

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

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

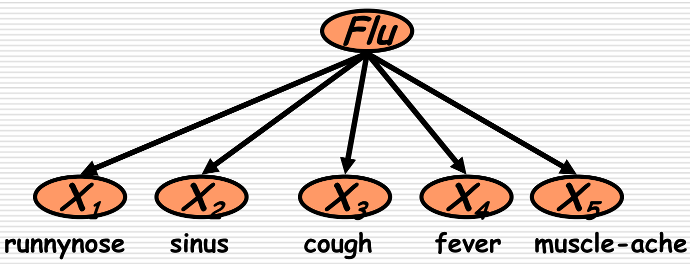
Naïve Bayes Classifier: Assumption

- $P(c_j)$
  - 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

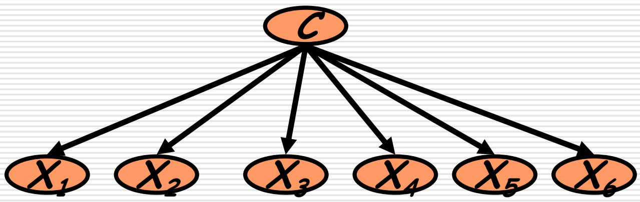


- **Conditional Independence Assumption:** features are independent of each other given the class:

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

- This model is appropriate for binary variables
- Only n|C| parameters (+|C|) to estimate

Learning the Model

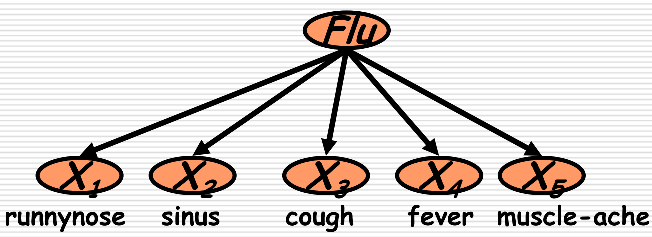


maximum likelihood estimates: most likely value of each parameter given the training data

- i.e. simply use the frequencies in the data

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

Problem with Max Likelihood



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

- What if we have seen no training 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!

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

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 (k)

Stochastic Language Models

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

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

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)

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
    - $P(c_j) \leftarrow \dfrac{|docs_j|}{|\text{total \# documents}|}$
    - $Text_j \leftarrow$ single document containing all $docs_j$
    - for each word x_k in *Vocabulary*
      - $n_k \leftarrow$ number of occurrences of x_k in $Text_j$
      - $P(x_k \mid c_j) \leftarrow \dfrac{n_k + \alpha}{n + \alpha |Vocabulary|}$

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 $|C||V| < |D|L_d$
- 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.

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

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 you get from binary independence model in probabilistic relevance feedback in hand-classified data

Two Models

- Model 2: Multinomial
  - 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

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

Parameter estimation

- Binomial model:

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

- Multinomial model:

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