
BUILDING A SPAM FILTER USING NAÏVE BAYES

Spam or not Spam: that is the question.

From: "" <takworld@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 Problems

- **Given:**

- A description of an instance, $x \in X$, where X is the *instance language* or *instance space*.

—(*Issue: how do we represent text documents?*)

- 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 .

—*We want to automatically build categorization functions (“classifiers”).*

EXAMPLES OF TEXT CATEGORIZATION

- **Categories = SPAM?**
 - “spam” / “not spam”
- **Categories = TOPICS**
 - “finance” / “sports” / “asia”
- **Categories = OPINION**
 - “like” / “hate” / “neutral”
- **Categories = AUTHOR**
 - “Shakespeare” / “Marlowe” / “Ben Jonson”
 - The Federalist papers

Bayesian Methods for Classification

- Uses *Bayes theorem* to build a *generative model* that approximates how data is produced.

- First step:

$$P(C | X) = \frac{P(X | C)P(C)}{P(X)}$$

Where C: Categories, X: Instance to be classified


- 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 each instance.

Maximum a posteriori (MAP) Hypothesis

- Let c_{MAP} be the most probable category.
Then goodbye to that nasty normalization!!

$$c_{MAP} \equiv \operatorname{argmax}_{c \in C} P(c \mid X)$$

$$= \operatorname{argmax}_{c \in C} \frac{P(D \mid c)P(c)}{P(X)}$$



No need to
compute
 $P(X)!!!!$

$$= \operatorname{argmax}_{c \in C} P(X \mid c)P(c)$$



As $P(X)$ is
constant

Maximum likelihood Hypothesis

If all hypotheses are *a priori* equally likely,
to find the maximally likely category c_{ML} ,
we only need to consider the $P(X/c)$ term:

$$c_{ML} \equiv \operatorname{argmax}_{c \in C} P(X | c)$$

Maximum
Likelihood
Estimate
("MLE")

Naïve Bayes Classifiers: Step 1

Assume that instance X described by n-dimensional vector of attributes $X = \langle x_1, x_2, \dots, x_n \rangle$

then

$$\begin{aligned} c_{MAP} &= \operatorname{argmax}_{c \in C} P(c \mid x_1, x_2, \dots, x_n) \\ &= \operatorname{argmax}_{c \in C} \frac{P(x_1, x_2, \dots, x_n \mid c)P(c)}{P(x_1, x_2, \dots, x_n)} \\ &= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n \mid c)P(c) \end{aligned}$$

Naïve Bayes Classifier: Step 2

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

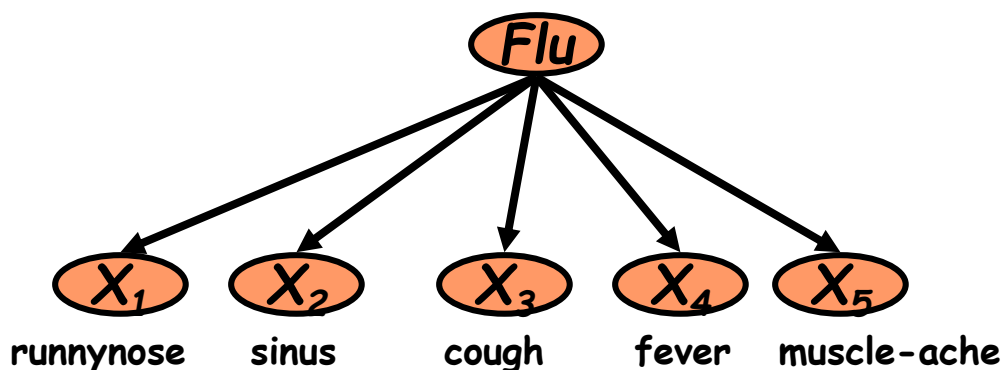
- $P(c_j)$: Can be estimated from the frequency of classes in the training examples.
- $P(x_1, x_2, \dots, x_n | c_j)$: *Problem!!*
 - $O(|X|^n \cdot |C|)$ parameters required to estimate full joint prob. distribution

Solution:

Naïve Bayes Conditional Independence Assumption:

$$P(x_1, x_2, \dots, x_n | c_j) = \prod_i P(x_i | c_j)$$

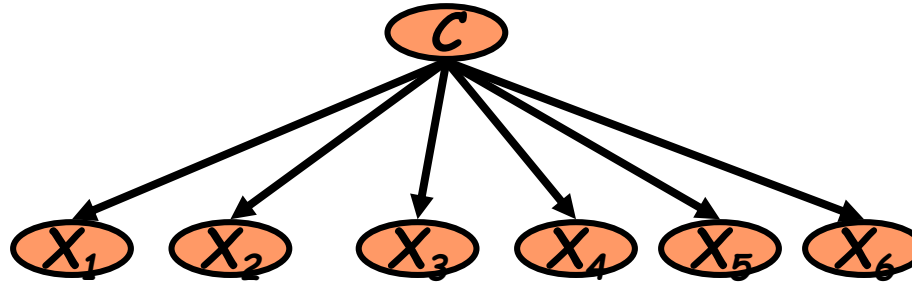
Naïve Bayes Classifier for *Binary* variables



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

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

Learning the Model

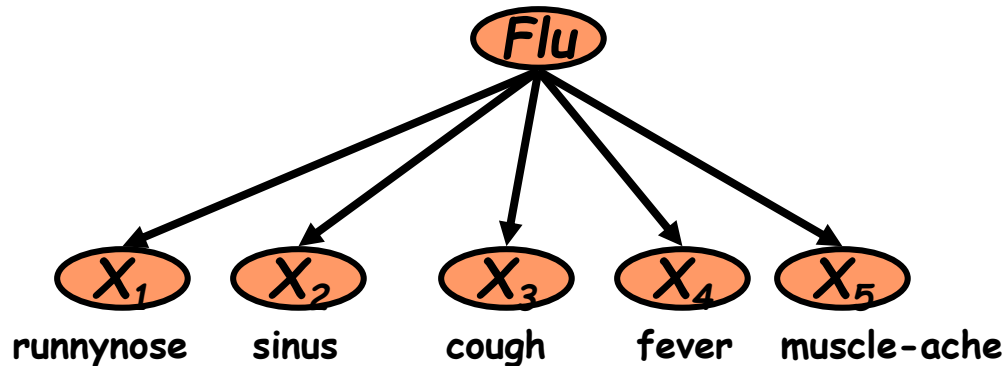


- **First attempt: *maximum likelihood* estimates**
 - Given training data for N individuals, where $count(X=x)$ is the number of those individuals for which $X=x$, e.g $Flu=true$
 - For each category c and each value x for a variable X

$$\hat{P}(c) = \frac{count(C = c)}{|N|}$$

$$\hat{P}(x | c) = \frac{count(X = x, C = c)}{count(C = c)}$$

Problem with Max Likelihood for Naïve Bayes



$$P(X_1, \dots, X_5 \mid Flu) = P(X_1 \mid Flu) \bullet P(X_2 \mid Flu) \bullet \dots \bullet P(X_5 \mid Flu)$$

- What if no training cases where patient had muscle aches but no flu?

$$\hat{P}(X_5 = t \mid \neg flu) = \frac{\text{count}(X_5 = t, \neg flu)}{\text{count}(\neg flu)} = \mathbf{0}$$

So if $X_5 = t$, $P(X_1, \dots, X_5 \mid \neg flu) = \mathbf{0}$

Zero probabilities overwhelm any other evidence!

“Add-1” Laplace Smoothing to Avoid Overfitting

$$\hat{P}(x | c) = \frac{\text{count}(X = x, C = c) + 1}{\text{count}(C = c) + |X|}$$

of values of X_i , here 2

- Slightly better version

$$\hat{P}(x | c) = \frac{\text{count}(X = x, C = c) + \alpha}{\text{count}(C = c) + \alpha |X|}$$

extent of
“smoothing”

Using Naive Bayes Classifiers to Classify Text:

Basic method

- **As a generative model:**
 1. Randomly pick a category c according to $P(c)$
 2. For a document of length N , for *each word*:
 1. Generate $word_i$ according to $P(w/c)$

$$P(c, D = \langle w_1, w_2, \dots, w_n \rangle) = P(c) \prod_{i=1}^N P(w_i | c)$$

- **This is a Naïve Bayes classifier for *multinomial* variables.**
- ***Note that word order really doesn't matter here***
 - Uses same parameters for each position
 - Result is *bag of words* model
 - Views document not as an ordered list of words, but as a *multiset*

Naïve Bayes: Learning (First attempt)

- From training corpus, extract *Vocabulary*
- Calculate required estimates of $P(c)$ and $P(w/c)$ terms,
 - For each c_j in C do

$$P(c) \leftarrow \frac{\text{count}_{docs}(C = c)}{|docs|}$$

where $\text{count}_{docs}(x)$ is the number of documents for which x is true.

- For each word $w_i \in \text{Vocabulary}$ and $c \in C$, where $\text{count}_{doctokens}(x)$ is the number of tokens over *all* documents for which x is true of that document and that token...

$$P(w_i | c) \leftarrow \frac{\text{count}_{doctokens}(W = w_i, C = c)}{\text{count}_{doctokens}(C = c)}$$

Naïve Bayes: Learning (Second attempt)

- Laplace smoothing must be done over the vocabulary items.
 - We can assume we have at least one instance of each *category*, so we don't need to smooth these.
- Assume a single new word UNK, that occurs nowhere within the training document set.
- Map all unknown words in documents to be classified (*test documents*) to UNK.
- For $0 \leq \alpha \leq 1$,

$$P(w_i | c) \leftarrow \frac{\text{count}_{\text{doctokens}}(W = w_i, C = c) + \alpha}{\text{count}_{\text{doctokens}}(C = c) + \alpha(|V| + 1)}$$

Naïve Bayes: Classifying

- Compute c_{NB} using *either*

$$c_{NB} = \arg \max_c P(c) \prod_{i=1}^N P(w_i | c)$$

$$c_{NB} = \arg \max_c P(c) \prod_{w \in V} P(w | c)^{\text{count}(w)}$$

where $\text{count}(w)$: the number of times word w occurs in doc

(The two are equivalent..)

PANTEL AND LIN: SPAMCOP

- **Uses a Naïve Bayes classifier**
- **M is spam if $P(\text{Spam}|\mathbf{M}) > P(\text{NonSpam}|\mathbf{M})$**
- **Method**
 - Tokenize message using Porter Stemmer
 - Estimate $P(x_k|C)$ using m-estimate (a form of smoothing)
 - Remove words that do not satisfy certain conditions
 - **Train: 160 spams, 466 non-spams**
 - **Test: 277 spams, 346 non-spams**
- **Results: ERROR RATE of 4.33%**
 - Worse results using trigrams

Naive Bayes is (was) 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.

- **A good dependable baseline for text classification**
 - But not the best *by itself*!
- **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**

Engineering: 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.

$$c_{NB} = \operatorname{argmax}_{c_j \in C} \log P(c_j) + \sum_{i \in \text{positions}} \log P(w_i | c_j)$$

REFERENCES

- Mosteller, F., & Wallace, D. L. (1984). *Applied Bayesian and Classical Inference: the Case of the Federalist Papers* (2nd ed.). New York: Springer-Verlag.
- P. Pantel and D. Lin, 1998. “SPAMCOP: A Spam classification and organization program”, In Proc. Of the 1998 workshop on learning for text categorization, AAAI
- Sebastiani, F., 2002, “Machine Learning in Automated Text Categorization”, ACM Computing Surveys, 34(1), 1-47