BUILDING A SPAM FILTER USING NAÏVE BAYES
Review: Bayes’ Rule & Diagnosis

\[ P(a|b) = \frac{P(b|a) \times P(a)}{P(b)} \]

- Useful for assessing diagnostic probability from causal probability:

\[ P(\text{Cause}|\text{Effect}) = \frac{P(\text{Effect}|\text{Cause}) \times P(\text{Cause})}{P(\text{Effect})} \]
Review: Bayes’ Rule For Diagnosis II

\[ P(\text{Disease} \mid \text{Symptom}) = \frac{P(\text{Symptom} \mid \text{Disease}) \ast P(\text{Disease})}{P(\text{Symptom})} \]

Imagine:

- disease = TB, symptom = coughing
- \( P(\text{disease} \mid \text{symptom}) \) is different in TB-indicated country vs. USA
- \( P(\text{symptom} \mid \text{disease}) \) should be the same
  - It is more widely useful to learn \( P(\text{symptom} \mid \text{disease}) \)

- What about \( P(\text{symptom}) \)?
  - Last time: Use conditioning
  - For determining, e.g., the most likely disease given the symptom, we can just ignore \( P(\text{symptom}) \)!!! (Coming up: Slide 11)
Review: Naïve Bayes I

By Bayes Rule

\[ P(C|T, X) = \frac{P(T, X|C)P(C)}{P(T, X)} \]

If \( T \) and \( X \) are conditionally independent given \( C \):

\[ P(C|T, X) = \frac{P(T|C)P(X|C)P(C)}{P(T, X)} \]

This is a Naïve Bayes Model:

All effects assumed conditionally independent given Cause
Review: Bayes' Rule II

- More generally, if Effect\(_i\) are conditionally independent given Cause:

\[
P(\text{Cause}, \text{Effect}_1, \ldots, \text{Effect}_n) = P(\text{Cause}) \prod_{i} P(\text{Effect}_i | \text{Cause})
\]

- And total number of parameters is \textit{linear} in \(n\)

![Diagram showing a cause and effect relationship with conditions: Flu \(\rightarrow\) X\(_1\) runnynose, X\(_2\) sinus, X\(_3\) cough, X\(_4\) fever, X\(_5\) muscle-ache.]}
Spam or not Spam: that is the question.

From: """ <takworlId@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*.
    —*(Important Issue: how do we represent text documents?)*
  • A *fixed set of categories*:
    \[ C = \{ c_1, c_2, ..., c_n \} \]

• To 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
A Graphical View of Text Classification

Text feature 2

Text feature 1

Graphics

NLP

AI

Theory

Arch.
Bayesian Methods for Classification

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

- First step:
  \[
  P(C \mid X) = \frac{P(X \mid C)P(C)}{P(X)}
  \]
  
  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!!

\[
c_{MAP} \equiv \text{argmax}_{c \in C} P(c \mid X)
\]

\[
= \text{argmax}_{c \in C} \frac{P(X \mid c)P(c)}{P(X)}
\]

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

As $P(X)$ is constant

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

$$c_{ML} \equiv \arg\max_{c \in C} P(X | c)$$

Maximum Likelihood Estimate ("MLE")
Naïve Bayes Classifiers: Step 1

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

$$c_{MAP} = \arg\max_{c \in C} P(c \mid x_1, x_2, \ldots, x_n)$$

$$= \arg\max_{c \in C} \frac{P(x_1, x_2, \ldots, x_n \mid c) P(c)}{P(x_1, x_2, \ldots, x_n)}$$

$$= \arg\max_{c \in C} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$
Naïve Bayes Classifier: Step 2

To estimate: $c_{MAP} = \arg\max_{c_j\in C} P(x_1, x_2, \ldots, x_n \mid c_j)P(c_j)$

- $P(c_j)$: Can be estimated from the frequency of classes in the training examples.

- $P(x_1, x_2, \ldots, x_n \mid c_j)$: Problem!!
  - $O(|X|^n \cdot |C|)$ parameters required to estimate full joint prob. distribution

Solution:

**Naïve Bayes Conditional Independence Assumption:**

$$P(x_1, x_2, \ldots, x_n \mid c_j) = \prod_{i} P(x_i \mid c_j)$$
Naïve Bayes Classifier for *Boolean* variables

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

\[
P(X_1, \ldots, X_5 \mid C) = P(X_1 \mid C) \cdot P(X_2 \mid C) \cdot \cdots \cdot P(X_5 \mid C)
\]
Learning the Model

- First attempt: *maximum likelihood* estimates

  - Given training data for $N$ individuals, where $\text{count}(X=x)$ 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 $X$

\[
\hat{P}(c) = \frac{\text{count}(C = c)}{|N|}
\]

\[
\hat{P}(x | c) = \frac{\text{count}(X = x, C = c)}{\text{count} (C = c)}
\]
Problem with Max Likelihood for Naïve Bayes

What if no training cases where patient with flu had a cough?

\[
P(X_1, \ldots, X_5 \mid Flu) = P(X_1 \mid Flu) \cdot P(X_2 \mid Flu) \cdot \cdots \cdot P(X_5 \mid Flu)
\]

- Zero probabilities overwhelm any other evidence!

\[
\hat{P}(X_3 = t \mid flu) = \frac{\text{count}(X_3 = t, flu)}{\text{count}(flu)} = 0
\]

So if \(X_3 = t\), \(P(X_1, \ldots, X_3 \mid flu) = 0\)
“Add-1” Laplace Smoothing to Avoid Overfitting

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

- Slightly better version

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

- Extent of “smoothing”

- # of values of \( X_i \), here 2
Using Naive Bayes Classifiers to Classify Text: Basic method for *Multinomial Variables*

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

$$P(c, D = < w_1, w_2, ..., w_n >) = P(c) \prod_{i=1}^{N} P(w_i | c)$$

- This is a Naïve Bayes classifier for *multinomial* variables.
- **Note that word order is assumed irrelevant 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 $P(c_j)$ and $P(w_i \mid c_j)$ terms,
  - For each $c_j$ in $C$ do

\[
P(c_j) \leftarrow \frac{\text{count}_{docs}(C = c_j)}{|docs|}
\]

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

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

\[
P(w_i \mid c) \leftarrow \frac{\text{count}_{doctokens}(W = w_i, C = c)}{\text{count}_{doctokens}(C = c)}
\]
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.

- with $0 \leq \alpha \leq 1$

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