Assignment 2: Spam Filtering using Naive Bayes

Mario Döbler

Institute for Signal Processing and System Theory
University of Stuttgart

1 Introduction

Naive Bayes classifiers are famously known to work well for document classification and spam filtering, despite their over-simplified assumption of conditional independence. In this assignment we will take a closer look at Spam Filtering using Naive Bayes.

For a small recap of Bayes theorem, let us consider the following fictional example: *80% of all emails are spam. 10% of spam emails contain the word "prize", while only 1% of non-spam emails contain the word "prize". What is the probability that an email is spam, given that it contains the word "prize"?*

We have the hypothesis $H = \{H(\text{am}), S(\text{pam})\}$ that an email is either ham (non-spam) or spam. Using Naive Bayes we can compute the posterior probability of the email being spam as

$$P(S|\text{"prize"}) = \frac{P(\text{"prize"}|S)P(S)}{P(\text{"prize"})}.$$

Using the law of total probability we can rewrite the equation as follows:

$$\begin{aligned} P(S|\text{"prize"}) &= \frac{P(\text{"prize"}|S)P(S)}{P(\text{"prize"}|S)P(S) + P(\text{"prize"}|H)P(H)} \\ &= \frac{0.1 \times 0.8}{0.1 \times 0.8 + 0.01 \times 0.2} \\ &\approx 0.976. \end{aligned}$$

Up to now, we only considered a single word. In the following we will extend this concept to classify messages as spam or ham, by examining every word in the message.

2 Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms applying Bayes theorem with the naive assumption of conditional independence. In other words

$$P(x_1, \dots, x_n|H) = \prod_{i=1}^{n} P(x_i|H)$$

and therefore

$$P(H|x_1,\ldots,x_n) = \frac{P(H)\prod_{i=1}^{n} P(x_i|H)}{P(x_1,\ldots,x_n)}.$$

Since $P(x_1,\ldots,x_n)$ is constant given the input

$$P(H|x_1,\ldots,x_n) \propto P(H)\prod_{i=1}^{n} P(x_i|H),$$

we can use the decision rule

$$\hat{H} = \arg\max_{H} P(H)\prod_{i=1}^{n} P(x_i|H).$$

Needless to say, this corresponds to MAP-decision. Another possibility would be to use ML-decision (assuming equal priors).

We can make various assumptions regarding the distribution of $P(x_i|H)$ resulting in different Naive Bayes classifiers.

2.1 Multinomial Naive Bayes

In this assignment we assume multinomially distributed data, which is one of the two classic naive Bayes variants used in text classification.

Multinomial Naive Bayes treats each message m as a bag of tokens t_i. The message m can be represented by a vector $\underline{x} = [x_1,\ldots,x_n]$, where each x_i is the number of occurences of t_i in message m.

The decision rule can be expressed as:

$$\hat{H} = \arg\max_{H} P(H)\prod_{i=1}^{n} P(t_i|H)^{x_i}$$

where each $P(t|H)$ is estimated with maximum likelihood as:

$$P(t|H) = \frac{N_{t,c}}{N_c}$$

where $N_{t,c}$ is the number of occurrences of token t in the training messages corresponding to hypothesis c and $N_c = \sum_t N_{t,c}$ is the number of occurrences of all tokens corresponding to hypothesis c.

3 Practical aspects

3.1 Smoothing

In practice a smoothed version of the maximum likelihood estimate is used:

$$P(t|H) = \frac{N_{t,c} + \alpha}{N_c + \alpha n}$$

Setting $\alpha = 1$ is called Laplace smoothing, while $\alpha < 1$ is called Lidstone smoothing.

3.2 Floating point underflow

Computing the product of many small number (close to zero) results in floating point underflow, where the product is too small to be represented and will be replaced by 0. Instead of computing

$$P(S)\prod_{i=1}^{n}P(x_i|S)$$

we can equivalently compute

$$\log\left(P(S)\prod_{i=1}^{n}P(x_i|S)\right)=\log(P(s))+\sum_{i=1}^{n}\log(P(x_i|S))$$

to avoid floating point underflow.
Since $\log(x)>\log(y)$ iff $x>y$, our new decision rule is

$$\log(P(S))+\sum_{i=1}^{n}\log(P(x_i|S))\underset{H}{\overset{S}{\gtrless}}\log(P(H))+\sum_{i=1}^{n}\log(P(x_i|H)).$$

4 Tasks

1. How many likelihood parameters have to be estimated in general if we don't make the naive assumption? And how many if we make the naive assumption?
2. Why do we have to make the naive assumption in practice? What problem is addressed by that? Is it a realistic assumption?
3. Why is it necessary to apply smoothing in practice?
4. Implement Naive Bayes for Spam Filtering (without using MultinomialNB from scikit). Report your train and test accuracy.

Optional tasks:

- Use a k-fold cross validation for a better estimate of the accuracy.
- Use another assumption regarding the distribution of $P(x_i|H)$. How is the performance?