

Programming Workshop Five Naïve Bayesian Classifier

Student Name: _____

Student Name: _____

In this programming exercise, you will implement a simple classifier of documents by their content, for example into spam and non-spam E-mails.

Let D denote a document undergoing classification. Let $C = \{\text{spam}, \text{nonspam}\}$ be a set of two possible classes D can belong to. Let $\{w_1, w_2, \dots, w_n\}$ be a vector of words in a dictionary. We define the Bayesian posterior probability as follows:

$$p(C|D) = \max \begin{cases} p(C = \text{spam}) \prod_i p(w_i | C = \text{spam}) \\ p(C = \text{nonspam}) \prod_i p(w_i | C = \text{nonspam}) \end{cases}$$

Terms $p(C=\text{spam})$ and $p(C=\text{nonspam})$ are called prior probabilities.

In other words, we can compute two posterior probabilities and select the class that gives us the maximum posterior probability. This step is called classification. During this step, an unseen document is scanned word by word and for each word, $p(w_i|C=\text{spam})$ is looked up and multiplied by the intermediate result. Once the end of the document is reached, the result is multiplied by the prior probability. Next, the process is repeated for nonspam posterior computation. The maximum of the two posteriors is chosen and a class is assigned to the document.

The training phase involves finding priors and $p(w|C)$ from some training sets.

Recall two basic properties in probability:

1. Probability of an event A is $0 \leq p(A) \leq 1$.
2. The sum of probabilities of all possible outcomes must be 1.

Try the following example. Assume you are given two e-mail messages:

spam e-mail message:

Now you can get out of DEBT fast, and start to enjoy life again. Simply complete our fast two minute online application, and help will be on its way too you within 48 hours, but usually the next business day!

nonspam e-mail message:

Write a program that allows the user to submit a question. When a question is submitted, the server should choose a random response from a list of vague answers and return a new page displaying the answer. Implement the program as a Perl program using CGI.pm.

1. Explain how you will create a dictionary of words for your classifier.
2. Compute $p(w_i='program'|C=spam)$ and $p(w_i='program'|C=nonspam)$. Show all your work.
3. Compute $p(w_i='DEBT'|C=spam)$ and $p(w_i='DEBT'|C=nonspam)$. Show all your work.
4. Name and describe at least three practical problems in the design of the classifier described above.