Document Classification Using Bag-of-Words Naive-Bayes Classifier

Teal Hobson-Lowther
Colorado School of Mines
December 10, 2015

I. Introduction

This paper will describe the methods and results of applying the Naive-Bayes Classifier to several different categories of text documents. In the second section, the data will be described. In the third section, the methods used to develop an algorithm that performs classification will be discussed. This includes both data pre-processing and a description of the Naive-Bayes Classifier in general. In the fourth section, the experiments performed, and their results, will be investigated. In the final section the reader will find a discussion of the results found.

II. Data

The dataset used for this paper consists of 20 text document classes, and 1000 sample excerpts from each class, for a total of 20,000 total documents. The classes are as follows: Atheism, Graphics, MS-Windows, PC Hardware, Mac Hardware, Windows 10, For Sale, Autos, Motorcycles, Baseball, Hockey, Cryptology, Electronics, Medical, Space , Christianity, Guns, Middle East, Politics, and Religion. These articles are pulled from different news sources, and as such serve as a diverse sampling.

III. Methods

This section will describe the general methodology of the Naive-Bayes Classifier, how it is applied to our data-set in general, and the data pre-processing that must occur before we perform any analysis.

A. Naive-Bayes Classification

In general, Naive-Bayes Classification (NB) is an extension of the Maximum Likelihood Estimation (MLE) that allows for the classification of data.

It is called "Naive" because it assumes that all elements of the input data are independent, and the "Bayes" term is included because the NB classifier exploits Bayes' Rule to develop a probability measure based on an input and past data.

Bayes' Rule is the most basic equation of conditional probability. It allows the user to extract the probability of an event A occurring given an event B also occurred ($p(A|B)$), as long as the user knows the probability of the event B occurring given A has occurred ($p(B|A)$), the probability of A occurring ($p(A)$), and the probability of event B occurring ($p(B)$). In equation form, it reads:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)} \tag{1}$$

In general, we don't focus on the denominator and just make a simpler statement, that $p(A|B) \propto p(B|A)p(A)$

However, the equation 1 can be altered for n independent events B_i in the following way:

$$p(A|B_1, B_2, ..., B_n) \propto p(B_1, B_2, ..., B_n|A)p(A) \tag{2}$$

The Naive assumption that all events A, B_i are independent makes the math much simpler. In particular, independence of events B_i implies that $p(B_1, B_2, ..., B_n|A) = \prod_{i=1}^{n} p(B_i|A)$, and 2 becomes:

$$p(A|B_1, B_2, ..., B_n) \propto p(A) \prod_{i=1}^{n} p(B_i|A) \tag{3}$$

Using this information for classification is not a complicated matter. Let C_j denote the jth class, and the elements of each input $\mathbf{x}$ be x_i. Then the classification is as follows:

$$\begin{aligned} y(\mathbf{x}) &= \underset{y}{\operatorname{argmax}}\, p(y = C_j)p(\mathbf{x}|y = C_j) \\ &= \underset{y}{\operatorname{argmax}}\, p(y = C_j) \prod_{i=1}^{n} p(x_i|y = C_j) \end{aligned} \tag{4}$$

B. Application

To perform Naive Bayes Classification, we split the all of the documents into two groups: 80% of the data (800 documents per class) is used for training the classifier. The remaining 20% (200 documents per class) is reserved as testing data, to determine the quality of our classifier.

Say we have P documents in a single class (with J classes), and let $\mathbf{x}_p$ denote the pth document from the training set. The elements of $\mathbf{x}_p$ are words, denoted x_{pi}. We first collect all of the training documents of a single class and put them into a "bag of words", so that we have a vector containing all words of the training set for that class. Then, we count the frequency of each word and use it to construct the probability of each word in the class. This gives us $p(x_i|y = C_j)$ for each word x_i and each class C_j.

Now, we can determine whether a document belongs to class a or b by taking all the elements of the new document, $xtest_i$, and calculating $\prod_{i=1}^{n} p(xtest_i|y = C_a)$ and $\prod_{i=1}^{n} p(xtest_i|y = C_b)$. If $\prod_{i=1}^{n} p(xtest_i|y = C_a) >$

$\prod_{i=1}^{n} p(xtest_i|y = C_b)$, we say the new document belongs to class a. If the converse is true, we say the new document belongs to class b. The classification is readily extended to multiple class comparison, by simply taking the maximum $\text{argmax}_j \prod_{i=1}^{n} p(xtest_i|y = C_j)$ over all classes being compared C_j.

C. Data Pre-Processing

Each document in the dataset contains a header that indicates the context of the document, among other things. The first step of data pre-processing is to remove this header. The second step is to split the training documents into words, and eradicate all special characters that would hinder the classification process. These words were counted and a matrix containing the probability of each word given a particular class, for all classes, was constructed.

IV. Experiments

A. Binary Classification

Binary classification consists of allowing our algorithm to determine whether a new text document belongs to class A or B, using the methodology described in III-A. Six different pairs of classes were compared. Some pairs were chosen under the assumption that their content was much different, so the classifier would have an easy time classifying them. Others were chosen because their content was similar, and we expected the machine to have a hard time distinguishing between the two. The confusion matrices are shown in Table I. The diagonal elements of each 2x2 matrix are correct classifications, and the off-diagonal elements are misclassifications.

True	Predicted		Accuracy
Atheism	199	1	.995
Cryptology	14	186	.93
Politics	165	35	.825
Guns	33	167	.835
Christianity	200	0	1
Religion	33	167	.835
Hockey	198	2	.99
Cryptology	7	193	.965
Electronics	199	1	995
Mid. East	38	162	.81
Mid. East	172	28	.86
Politics	13	187	.935

TABLE I: Summary of pair-wise binary classifications. Diagonal elements are correct classifications, and off-diagonal elements are misclassifications.

B. Trinary Classification

Trinary classification consists of allowing our algorithm to determine whether a new text document belongs to class A, B, or C, using the methodology described in III-A. Six different three-class groups were compared. Some groups were chosen under the assumption that their content was much different, so the classifier would have an easy time classifying them. Others were chosen because their content was similar, and we expected the machine to have a hard time distinguishing between the three. The confusion matrices are shown in Table II. The diagonal elements of each 3x3 matrix are correct classifications, and the off-diagonal elements are misclassifications.

True	Predicted			Accuracy
Atheism	199	0	1	.995
Cryptology	13	183	4	.915
Guns	15	7	178	.89
Hockey	158	32	10	.79
Baseball	0	195	5	.975
Autos	0	1	199	.995
Religion	198	0	2	.99
Medical	43	147	10	.735
Graphics	24	3	173	.865
PC Hardware	199	0	1	.995
Windows 10	154	44	2	.22
MS-Windows	195	0	5	.025
Atheism	181	6	13	.905
Christianity	1	199	0	.995
Religion	77	26	97	.485
Space	120	48	32	.6
Electronics	0	198	2	.99
Guns	0	7	193	.965

TABLE II: Summary of trinary classifications. Diagonal elements are correct classifications, and off-diagonal elements are misclassifications.

V. Discussion

A. Binary Classification

Our Binary Naive-Bayes Classifier performed rather well. As can be seen in I, the upper limit of total classification accuracy was 97.8%, and was achieved by the pairing of Hockey vs. Cryptology. This is expected, because the subject matter is so disparate. Our lower limit of total classification accuracy is 83%, when classifying Politics vs. Guns. This could be because gun control issues are frequently debated in general political discussion.

B. Trinary Classification

Our Trinary Naive-Bayes Classifier performed with mixed quality. The introduction of a third class resulted in generally lower group-wide classification accuracy. A rather disparate grouping (Atheism, Cryptology, and Guns) produced the best results, as expected, with classification accuracies of 99.5%, 91.5%, and 89%, resp. The grouping most similar in topic, PC Hardware, Windows 10, and MS-Windows, got the worst results: 99.5%, 22%, and 2.5% classification accuracy, resp. It seems that the classifier incorrectly puts almost all Windows 10 and MS-Windows documents into the PC Hardware category. Also notable: 38.5% of Religious documents were misclassified as articles about Atheism.