CSCI 5090/7090 Machine Learning, Spring 2018: Homework 2

Due: Wednesday, February 28^{th}, 9:00 PM

1 Naive Bayes Document Classifier [100 points]

In this question, you will implement the Naive Bayes document classifier and apply it to the classic 20 newsgroups dataset[1]. In this dataset, each document is a posting that was made to one of 20 different usenet newsgroups. Our goal is to write a program which can predict which newsgroup a given document was posted to.

For this question, you may write your code and solution in teams of at most 2. If you decide to do this, you should submit one copy of your solutions (both code and answers to questions) per team. This copy should be clearly marked with the names of both team members.

1.1 Model

Say we have a document D containing n words; call the words $\{X_1, \cdots, X_n\}$. The value of random value X_i is the word found in position i in the document. We wish to predict the label Y of the document, which can be one of m categories. We could use the model:

$$P(Y|X_1, \cdots, X_n) \propto P(X_1, \cdots, X_n|Y)P(Y) = P(Y)\prod_i^n P(X_i|Y) \tag{1}$$

As usual with discrete data, we assume that $P(X_i|Y)$ is a multinomial distribution over some vocabulary V; that is, each X_i can take one of $|V|$ possible values corresponding to the words in the vocabulary. Each word in a document is assumed to be an *iid* draw from this distribution.

1.2 Data

The data file (available on the website) contains six files:

1. **vocabulary.txt** is a list of the words that may appear in documents. The line number is words id in other files. That is, the first word ('archive') has wordId 1, the second ('name') has wordId 2, etc
2. **newsgrouplabels.txt** is a list of newsgroups from which a docment may have come. Again, the line number corresponds to the label's id, which is used in the .label files. The first line ('alt.atheism') has id 1, etc.
3. **train.label** Each line corresponds to the label for one document from the training set. Again, the documents id (docId) is the line number.
4. **test.label** The same as train.label, except that the labels are for the test documents.

[1] `http://people.csail.mit.edu/jrennie/20Newsgroups/`

5. **train.data** Specifies the counts for each of the words used in each of the documents. Each line is of the form "`docId wordId count`", where `count` specifies the number of times the word with id `wordId` in the training document with id `docId`. All word/document pairs that do not appear in the file have count 0.

6. **test.data** Same as train.data, except that it specified counts for test documents.

1.3 Implementation

Your task is to implement the Naive Bayes classifier specified above. You should estimate $P(Y)$ using the MLE, and estimate $P(X|Y)$ using additive smoothing[2] where $\alpha = 1/|V|$.

Question 1. In your answer sheet, report your overall testing accuracy (Nubmer of correctly classified documents in the test set over the total number of test documents), and print out the confusion matrix (the matrix C, where cij is the number of times a document with ground truth category j was classified as category i). **[10 points]**

Question 2. Are there any newsgroups that the algorithm confuses more often than others? Why do you think this is? **[5 points]**

In your initial implementation, you used a prior (i.e. by using additive smoothing) to estimate $P(X|Y)$ and I told you set $\alpha = 1/|V|$. Hopefully you wondered where this value came from. In practice, the choice of prior is a difficult question in Bayesian learning: either we must use domain knowledge, or we must look at the performance of different values on some validation set. Here we will use the performance on the testing set to gauge the effect of α.

Question 3. Re-train your Naive Bayes classifier for values of α between .00001 and 1 and report the accuracy over the test set for each value of α. Create a plot with values of α on the x-axis and accuracy on the y-axis. Use a logarithmic scale for the x-axis. Explain in a few sentences why do you think accuracy drops for both small and large values of α. **[5 points]**

1.4 Logspace Arithmetic

When working with very large or very small numbers (such as probabilities), it is useful to work in *logspace* to avoid numerical precision issues. In logspace, we keep track of the logs of numbers, instead of the numbers themselves. For example, if $p(x)$ and $p(y)$ are probability values, instead of storing $p(x)$ and $p(y)$ and computing $p(x) * p(y)$, we work in log space by storing $\log(p(x)), \log(p(y))$, and we can compute the log of the product $\log(p(x) * p(y))$ by taking the sum: $\log(p(x) * p(y)) = \log(p(x)) + \log(p(y))$.

1.5 [Optional]

One good practice to make sure your implementation is correct, is to compare your code with Naive Bayes implementations avaiable in Python libraries such Scikit. Interestingly, Scikit already includes the 20 newsgroups text dataset, thus you can easily run Naive Bayes classifier on this dataset and check the results with your own implementation. See `http://scikit-learn.org/stable/datasets/twenty_newsgroups.html` for more details.

[2] `https://en.wikipedia.org/wiki/Additive_smoothing`