10-601 Machine Learning, Fall 2011: Homework 2

Machine Learning Department
Carnegie Mellon University

Due: October 6th, 2011, 5pm

Instructions There are 2 questions on this assignment. The second question involves coding, so start early. Please submit your completed homework to Sharon Cavlovich (GHC 8215) by 5 PM on Thursday, October 6th, 2011. Submit your homework as 2 **separate** sets of pages, one for each question. Please staple or paperclip all pages from a single question together, but **DO NOT** staple pages from different questions together. This will make it much easier for the TA's to split up your homework for grading. Include your name and email address on each set.

1 MLE and MAP [William Bishop, 20 points]

In this problem we will find the maximum likelihood estimator (MLE) and maximum a posteriori (MAP) estimator for the mean of a univariate normal distribution. Specifically, we assume we have N samples, $x_1, \ldots, x_N$ independently drawn from a normal distribution with *known* variance σ^2 and *unknown* mean μ.

1. [5 Points] Please derive the MLE estimator for the mean μ. Make sure to show all of your work.
2. [12 Points] Now derive the MAP estimator for the mean μ. Assume that the prior distribution for the mean is itself a normal distribution with mean ν and variance β^2. Please show all of your work. HINT: You may want to make use of the fact that:

$$\beta^2\left(\sum_{i=1}^{N}(x_i-\mu)^2\right)+\sigma^2(\mu-\nu)^2 = \left[\mu\sqrt{N\beta^2+\sigma^2}-\frac{\sigma^2\nu+\beta^2\sum_{i=1}^{N}x_i}{\sqrt{N\beta^2+\sigma^2}}\right]^2-\frac{[\sigma^2\nu+\beta^2\sum_{i=1}^{N}x_i]^2}{N\beta^2+\sigma^2}+\beta^2\left(\sum_{i=1}^{N}x_i^2\right)+\sigma^2\nu^2$$

3. [3 Points] Please comment on what happens to the MLE and MAP estimators as the number of samples N goes to infinity.

2 Bag of Words [Shing-hon Lau, 40 points]

In this problem, you will code up a Naive Bayes classifier to perform "Bag of Words"-style classification of text documents. Our goal is to train a classifier that can accurately predict whether newsgroup posts come from a newsgroup about baseball or a newsgroup about (ice) hockey.[1] Recall that in "Bag of Words"-style classification, we make the assumption that the probability a word appears in document is conditionally independent of the word position given the class of the document.

If we have a particular document D_i with w_i words in it, we can compute the probability that D_i comes from a class y as:

[1] For those of your unfamiliar with these sports, you can read about them at http://en.wikipedia.org/wiki/Baseball and http://en.wikipedia.org/wiki/Ice_hockey.

$$\mathbb{P}(D_i|Y=y) = \prod_{j=1}^{w_i} \mathbb{P}(X_j = x_j|Y=y).$$

Here, X_j represents the j^{th} position in document D_i and x_j represents the actual word that appears in the j^{th} position in document D_i. As a concrete example, we might have the first document (D_1) which contains 300 words ($w_1 = 300$). The document might belong to the hockey newsgroup ($y =$ 'hockey') and the 15^{th} word in the document might be 'player' ($x_{15} =$ 'player').

Of course, for the purposes of classification, we are more interested in the probability distribution over the document classes given a particular document. We can use Bayes Rule to write:

$$\mathbb{P}(Y=y|D_i) = \frac{\mathbb{P}(Y=y)\prod_{j=1}^{w_i}\mathbb{P}(X_j=x_j|Y=y)}{\mathbb{P}(X_1=x_1,\ldots,X_{w_i}=x_{w_i})}.$$

Note that, for the purposes of classification, we can actually ignore the denominator here and write:

$$\mathbb{P}(Y=y|D_i) \propto \mathbb{P}(Y=y)\prod_{j=1}^{w_i}\mathbb{P}(X_j=x_j|Y=y).$$

1. [2 Points] Why is it that we can ignore the denominator?

Our prediction, $\hat{y}_i$, for the class is then,

$$\hat{y}_i = \arg\max_y \mathbb{P}(Y=y|D_i) = \arg\max_y \mathbb{P}(Y=y)\prod_{j=1}^{w_i}\mathbb{P}(X_j=x_j|Y=y).$$

When working with products of many probabilities, it is often advisable to take a logarithm. This provides us with two advantages. It turns products into sums, which is often more convenient. It also avoids underflow and numerical accuracy issues. Many computing systems will round 10^{-1000} to 0, which is undesirable when we may be working with thousand-word documents. Taking the logarithm gives us:

$$\hat{y}_i = \arg\max_y log(\mathbb{P}(Y=y)) + \sum_{j=1}^{w_i} log(\mathbb{P}(X_j=x_j|Y=y)).$$

2. [2 Points] Why does $\arg\max_y \mathbb{P}(Y=y|D_i) = \arg\max_y log(\mathbb{P}(Y=y|D_i))$?

The ZIP file associated with this assignment contains a total of five items:

- vocabulary.txt
- baseball_train_set.csv
- baseball_test_set.csv
- hockey_train_set.csv
- hockey_test_set.csv

The last four files are CSV (comma-separated value) files. Each of these files consists of 50 lines, with each line containing 5822 different values. Each line corresponds to a single post on a newsgroup, and each of the values indicates the number of times a particular word appears in that post. The words are contained in `vocabulary.txt`; there are 5822 lines in total in the file, corresponding to the 5822 distinct words that appeared in either the training or test set.[2] For example, the 5236^{th} line of `vocabulary.txt` is 'the'. Therefore, if we see a '3' in the 5236^{th} position of a line in the CSV file, we know that the word 'the' appeared three times in that message. You need not use `vocabulary.txt` to code up a successful solution, but it is provided if you wish to sanity check your knowledge of the data format.

3. [20 Points] Code up a "Bag of Words"-style Naive Bayes classifier to handle the provided data. Your classifier should take as input a training set consisting of baseball and hockey posts and a test set also consisting of baseball and hockey posts. Your classifier should output predictions for the classes of the test set (either 'baseball' or 'hockey'). Use the MLE estimate for all parameters in your classifier. If it arises in your code, define $0 \times log(0) = 0$. You may assume that the two classes occur with equal prior probability, (i.e., $\mathbb{P}(Y =$ 'baseball'$) = \mathbb{P}(Y =$ 'hockey'$)$). In case of ties, you should predict 'baseball'.

   You may use the programming language of your choice. If you decide to use Matlab, you can import the CSV files by using File → Import Data. Please turn in a printed version of all your code (including the extension below) and also submit your code electronically via Blackboard. (Hint: If you have more than 100 lines of code, you are almost certainly doing something wrong.)

[2] Typically, one would not know the words that might appear in the test set and you would need to do slightly more bookkeeping to keep track of what words you saw in the training set. Then, you would check if a word in the test set appeared in the training set. However, we're cheating here a little bit to make the assignment easier to code up.

4. [3 Points] Train your classifier using all of the training set (50 posts each from baseball and hockey) and have it classify all of the test set (50 posts each from baseball and hockey). Report, in writing, the answers to the following questions: What is your test set accuracy? What did your classifier end up predicting? Why is using the MLE estimate is a bad idea in this situation?

5. [5 Points] Extend your classifier so that it can compute a MAP estimate using a fair Dirichlet prior. The prior is fair in the sense that it "hallucinates" that each word appears β times in the train set. Formally, all of the parameters of the Dirichlet prior are equal to β. The easiest way to do this is to modify your classifier to take β as an additional parameter. (Hint: This extension should take no more than 3 lines of code.)

6. [5 Points] Train your classifier using all of the training set and have it classify all of the test set, using the following values of β: 0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 2, 5, and 10. Submit a plot with β on the x-axis and classification accuracy on the y-axis. Report, in writing, the answers to the following questions: How does classification accuracy change with respect to β? Is using a large value of β good or bad? Why?

7. [3 Points] Fix $\beta = 0.1$. Train your classifier using the first n posts from the baseball training set and the first n posts from the hockey training set (for a total of $2n$ training examples) for the following values of n: 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50. Test your classifier on the full test set (50 baseball and 50 hockey posts). Submit a plot with n on the x-axis and classification accuracy on the y-axis.