

CSI 5325 Assignment 4

Greg Hamerly

assigned March 19, 2009; due April 2, 2009

1 Expectations

For all assignments in this class you should work individually, but feel free to discuss your work with me or your classmates.

The writeups you do for this course should be of high quality; complete but concise. Please structure your writeups well, introducing ideas logically (rather than simply chronologically). Your writeup should be formatted as if you would submit it to a conference or journal. It should include the following sections in the given order:

- Introduction – Describe the problem(s) you are working on at a high level, and give the reader a summary of the results you found.
- Methodology – Explain your data sets, data preprocessing, the algorithms and techniques you apply, the hypotheses you are testing, and how you will evaluate the success of your experiments. Use tables and figures as are helpful.
- Results – Explain the results of the experiments, using graphs and tables and text. Then, analyze your results.
- Conclusion – Summarize the work again, and if applicable, describe further work that you think would build on what you've done.

For this particular assignment, each section (introduction, methodology, etc.) should address each of the parts of this assignment below. You could, for example, have a subsection which discusses each part within each section.

1.1 Submitting your work

You should turn in all your work in printed format in class, as well as by email to `hamerly@cs.baylor.edu`. Your email should have attached a zip file which has a single folder that contains all your materials. The zip file name should follow this pattern: `'lastname_nn.zip'`, where lastname is your last name, and nn is the assignment number (like '01'). Please DO include all source code with your emails, but DO NOT print out your source code to hand in (just print and turn in writeup).

1.2 Tools for the course

As suggested in assignment 1, your work should be composed in \LaTeX , and I suggest looking at MATLAB for programming.

2 Implement a naïve-Bayes classifier

Recall the naïve Bayes classifier we learned about in chapter 6 of Mitchell. Implement a classifier which takes as input a text document and predicts a class.

Of course, there are two modes for this software, like usual:

- Training mode – given a set of text documents, each labeled with some class, train a classifier which learns these classes. This will involve parsing each text file, constructing a vocabulary that spans all documents, and counting the frequency of each word for each class.
- Classification mode – given a new text document and a learned classifier, predict the class for the document.

For this project, we will work on the ‘20 newsgroups’ dataset, which can be found here:

- <http://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups>

There are 20 classes in this dataset, so you will train a classifier that distinguishes between the 20 classes. It will probably be easy to use Matlab for this problem *after* you use a parser (e.g. perl is easy) to convert from text to a bag-of-words vector format.

Things to think about and try:

- Report your findings using appropriate graphs and analysis.
- Use holdout sets or cross-validation as appropriate to determine error.
- Try estimating the confidence bounds on the true error, as described in chapter 5 of Mitchell.
- Determine which words are most probable indicators for different classes.
- How does classification accuracy vary as you vary the amount of training?

One method for presenting classification results that is particularly useful is a ‘confusion matrix’, which in this case would be a 20×20 matrix that shows how often the classifier makes different mistakes. Here’s a fake confusion matrix I just made up for a 5-class problem:

	class 1	class 2	class 3	class 4	class 5
class 1	23	4	2	7	4
class 2	3	22	1	9	3
class 3	6	9	14	7	1
class 4	0	4	4	21	8
class 5	5	5	9	3	20

In this confusion matrix, the true class is given in the left column, and the predicted class is given in the top row. From this we can calculate many things. Suppose that in Matlab I have the above matrix stored as the variable `cm`, then:

- The total number of examples in the matrix is:
`n = sum(sum(cm)) = 194`
- The accuracy of the test was:
`accuracy = sum(diag(cm)) / n = 0.51546`
- The accuracy of classifying class 1 was:
`accuracy1 = cm(1,1) / sum(cm(1,:)) = 0.57500`
- etc.

The value of a confusion matrix is that it contains all the information a reader needs to re-calculate the statistics in which he/she is interested.

3 Going further

Try out extensions like:

- stemming (search google for stemming software)
- stop word removal (search google for lists of stop words)
- smoothing for zero counts
- output probabilities instead of classes (e.g. $Pr(class|document)$ for all 20 classes), so that the user can see the confidence of the classifier
- your own ideas.

4 Evaluation criteria

The following sections will be used for grading your assignment (these adapted from Charles Elkan’s grading criteria):

Category	Points
Introduction insightful on background and motivation	3
Precise and reproducible description of technical work	3
Sensible implementation decisions (code reuse, choice of PL, etc.)	2
Sufficiently conclusive experiments (large datasets, lesion studies, etc.)	3
Well-designed, well-described, reproducible experiments	3
Understandable presentation of results, preferably graphical	3
Clear, correct analysis of results (including statistical significance)	3
Insightful discussion of all major experimental results	3
Exciting and useful results with general applicability	3
Appropriate organization (logical, not chronological) and easy-to-read plain writing style	3
Correct spelling, grammar, and choice of words	3
Total	32

A lesion study means taking parts out of the system you are using to identify what will cause a system’s performance to degrade. For this assignment, you need not perform statistical significance tests, but careful analysis is still important.