Homework 2 - Math Review

Machine Learning - CSC 84020 - Spring 2010 - Prof. Rosenberg

Due March 12 at 2:00pm

Problem 1) (5 pts.) Prove that if A is symmetric, then $x^T Ay = y^T Ax$.

Problem 2) Consider the likelihood function used in logistic regression.

$$p(\mathbf{t}, \mathbf{X}|\pi, \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = \prod_{n=0}^{N-1} (\pi N(\mathbf{x}_n|\boldsymbol{\mu}_0, \boldsymbol{\Sigma}))^{t_n} ((1-\pi)N(\mathbf{x}_n|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}))^{1-t_n}$$

2a. (10 pts)
Identify the maximum likelihood estimate of $\boldsymbol{\mu_0}$.

2b. (10 pts)
Identify the maximum likelihood estimate of $\boldsymbol{\Sigma}$.

Problem 3) (75 pts)

Implement Polynomial Fitting with Linear Regression with L2-Regularization.

You may use python, java or c++ to perform this.
You may (and should) use matrix libraries for this assignment. For python, NumPy is a good (though not the only option. For Java, JAMA is good. For c++, gsl is a useful tool to know.
Your program should take 4 parameters.

- A training file – comma separated values posted to the course webpage
- A testing file – comma separated values posted to the course webpage
- The number of parameters – the degree of the polynomial to fit
- A λ value – the regularization penalty.

Your code should run using the following command line format

```
./linear_regression training.csv testing.csv <d> <lambda>
./linear_regression.py  training.csv testing.csv <d> <lambda>
java -cp=<CLASSPATH> LinearRegression training.csv testing.csv <d> <lambda>
```

Your code should 1) train a regularized linear regression model using the training data, 2) evaluate the model on the training data, and 3) evaluate the model on the testing data. The training and testing data files are comma separated valued plain text files containing one data point per row. The format is

```
x,t
x,t
x,t
etc.
```

Your code should output the training (mean squared) error and the testing (mean squared) error, **in that order** separated by a comma and ended by a new line. Your code should not produce any other information if it runs correctly. If there is an error, your code may report error information like "-4 is not a valid value for lambda".
For example:

```
$./linear_regression training.csv testing.csv <d> <lambda>
.011, .101
$
```

Deliverables

- All source code and libraries required for your project.
- A README/Report file whose contents are described below.

In addition to the implementation, this assignment asks you to experiment with your code on training and testing data. The experiments are described below.

Linear Regression Experiments

- Evaluate the training and testing error with no regularization.
- Report the training and testing data for at least 50 parameter values.
- Identify when overfitting occurs
- Discuss the results of this experiment. What does it tell you about the function generating the data?
- Introduce a regularization term while using a number of parameters that would otherwise lead to overfitting – experiment with at least 10 values for lambda.

- Identify the best value for lambda – the best testing data performance.
- Discuss the results of this experiment – what does the best value of λ tell you about the function generating the data? How would you determine the best value of λ using only the training data?

The README can be in any electronic format you like and should minimally include the following

- A list and description of every file included in the submission – including which matrix library (if any) you use.
- A description of how your code is compiled.
- A high level description of the task – here: Linear Regression with L2 Regularization. This should be about a paragraph long, and doesn't necessarily need a lot of mathematical notation. It should be understandable to a reasonably informed and proficient coder.
- A report of the experiments – as described above. Graphs may be particularly helpful for this assignment
- Any other points of interest – running time, complexity, unique qualities of your implementation, etc.

Grading:

- **10 points - Compilation** Each file must compile without error or warning into an executable as described above. (Note: for python, no points are awarded for compilation, but execution is worth 20 points.)
- **10 points - Execution** Each executable must run without error or warning on valid input using the command line parameters described above. (Note: for python, no points are awarded for compilation, but execution is worth 30 points.)
- **10 points - README** Does your README documentation completely and accurately describe the linear regression algorithm, as well as how to compile and run your project.
- **15 points - Report** Within your README have you reported the training and testing error using at least 40 settings of d, the number of parameters without using regulariztion? Have you identified at what point overfitting occurs? Have you evaluated the training and testing error using at least 10 settings of λ? Have you hypothesized the best parameterization for this dataset? Have you discussed the use/merit/demerits of regularization

- **7 points - Within Code Documentation** Is the code documented for obvious understanding of the use, preconditions, and postconditions of each function?
- **8 points - Style** Are functions and variables given self-explanatory names? Are functions used to aid intelligibility of the code? Are functions used to reduce repeated blocks of code? Is indentation, spacing, use of parentheses, use of braces consistent, and sensible? For example, if you use brackets on the same line at the start of a block, always do so. If you place a brace on the line following the start of the brace, always do so. If you put a space between variables and operators, e.g. `if (i == j)`, always do so. So, `if (i == j) i = j+k;` is bad. It should be `if (i == j) i = j + k;` or if you prefer `if (i==j) i=j+k;`. You will be graded on consistency in these decisions, not on any particular style.
- **15 points - Correctness** Is the algorithm implemented correctly?