MACHINE LEARNING COMS 4771, HOMEWORK 1
Assigned September 12, 2013. Due September 26, 2013 before 2:40pm.

Please submit separate files for a) write-up, b) Matlab source files and c) figures (if you choose to include them separately from the writeup). Do not include any other files. Your write-up should be in ASCII plain text format (.txt) or Postscript (.ps) or the Adobe Portable Document Format (.pdf). Please do not submit Microsoft Office documents, LaTeX source code, or something more exotic since we will not be able to read it. LaTeX is preferred and highly recommended, but it is not mandatory. You can use any document editing software you wish, as long as the final product is in .ps or .pdf. Even if you do not use LaTeX to prepare your document, you can use LaTeX notation to mark up complicated mathematical expressions, for example, in comments in your Matlab code. See the Tutorials page for more information on LaTeX.

All your code should be written in Matlab. Please submit all your source files, each function in a separate file. Clearly denote what each function does, which problem it belongs to, and what the inputs and the outputs are. Do not resubmit code or data provided to you. Do not submit code written by others. Identical submissions will be detected and both parties will get zero credit. In general, shorter code is better. Sample code is available on the Tutorials web page. Datasets are available from the Handouts web page.

You may include figures directly in your write-up or include them separately as .jpg, .gif, .ps or .eps files, and refer to them by filename.

When you are ready to submit your work, submit it via courseworks.columbia.edu. If something goes wrong, ask the TAs for help. If nothing else works, send your homework to the TAs. Thanks.

1 Problem 1 (10 points)

Cross-validation for Polynomial Fitting: Consider the problem of fitting a polynomial function. Assume we wish to find a one dimensional function that takes a scalar input and outputs a scalar $f : \mathbb{R} \to \mathbb{R}$. The function has the form

$$f(x;\boldsymbol{\theta}) \quad = \quad \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_d x^d$$

where d is the degree of the polynomial. Develop code that finds the θ which minimizes the risk

$$R_{emp}(\boldsymbol{\theta}) \quad = \quad \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}(y_i - f(x;\boldsymbol{\theta}))^2$$

on a data-set. To help you get started, download the Matlab code in "polyreg.m" (on the tutorial web page) to do polynomial curve fitting. Use your code on the dataset "dataset1a.txt". In that data-set the first column corresponds to the scalar features $\{x_1, \ldots, x_N\}$ and the second column corresponds to the scalar labels $\{y_1, \ldots, y_N\}$. Fit a polynomial model to this data for various choices for d, the degree of the polynomial. Which value(s) of d seems somewhat more reasonable? Show a few plots documenting your work, for each show as well the values of weights $\boldsymbol{\theta}$. Use cross-validation by randomly splitting the data-set into two halves to select the complexity of the model (i.e., the degree of the polynomial). Include a few plots showing the training and testing risk across various choices of d and plot the your $f(x;\theta$ (overlayed on the data) for the best choice of d according to cross-validation.

2 Problem 2 (10 points)

Sinusoidal Regression: Modify the Matlab code for "polyreg.m" such that it learns a one-dimensional regression function $f : \mathbb{R} \to \mathbb{R}$ where the function is formed using a sinusoidal basis rather than a polynomial basis. Use basis functions of the form

$$f(x;\boldsymbol{\theta}) \quad = \quad \theta_0 + \sum_{i=1}^{k} \theta_i \sin(i \times x) + \sum_{i=1}^{k} \theta_{i+k} \cos(i \times x)$$

for different choices of k. Fit the data in "dataset1b.txt" with the first 100 points as training and the second 100 points as testing. Try various values of k with cross-validation to find a good setting. Compute and show the training and testing error for the best value of k model and show the fit graphically by saving the Matlab plot of $f(x)$ overlayed on the data.

3 Problem 3 (20 points)

Logistic Regression: Implement a linear logistic regression algorithm for binary classification in Matlab using gradient descent. Your code should accept a dataset $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$ and find a parameter vector $\boldsymbol{\theta} \in \mathbb{R}^d$ for the classification function

$$f(\mathbf{x};\boldsymbol{\theta}) \quad = \quad \left(1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x})\right)^{-1}$$

which minimizes the empirical risk with logistic loss

$$R_{emp}(\boldsymbol{\theta}) \quad = \quad \frac{1}{N} \sum_{i=1}^{N} (y_i - 1)\log(1 - f(\mathbf{x}_i;\boldsymbol{\theta})) - y_i \log(f(\mathbf{x}_i;\boldsymbol{\theta})).$$

Since you are using gradient descent, you will have to specify the step size η and the tolerance ϵ. Pick reasonable values for η and ϵ to then use your code to learn a classification function for the dataset in "dataset2.mat". Type "load dataset2" and you will have the variables X (input vectors) and Y (binary labels) in your Matlab environment which contain the dataset. Use the whole data set as training. Show with figures the resulting linear decision boundary on the 2D X data. Show the binary classification error and the empirical risk you obtained throughout the run from random initialization until convergence. Note the number of iterations needed for your choice of η and ϵ.

4 Problem 4 (10 points)

Logistic Squashing Function. The logistic squashing function is given by $g(z) = 1/(1 + \exp(-z))$. Show that it satisfies the property $g(-z) = 1 - g(z)$. Also show that its inverse is given by $g^{-1}(y) = \ln(y/(1 - y))$.