CMPUT 340—Introduction to Numerical Methods

Assignment 3

Winter 2007
Department of Computing Science
University of Alberta

Due: *Wednesday, March 28 at 23:59:59 local time*
Worth: 15% of final grade

Instructor: Dale Schuurmans, Ath409, x2-4806, dale@cs.ualberta.ca

---

Note: For this assignment you will need to go to the course webpage and download the file a3files.zip. Unzipping this file will create 4 files, data1.mat, quadfeatures.m, quadkernel.m, and gausskernel.m, which will be needed. You need to submit the answers to all written questions in hard copy to the course drop box. Matlab functions need to be submitted in .m files. You can submit your files by typing "`astep -c c340 -p a3` *file1 file2 ...*". See http://ugweb.cs.ualberta.ca/~astep

Note: For this assignment you *are* allowed to use the functions in `help matfun`, but not those in `help stats`.

Part 1 (Linear least squares—2%)

(a) (1%) Write a Matlab function `[w1] = fitlin(X,y)` which returns a vector of weights `w1` corresponding to the minimum sum of squared errors linear function for predicting `y` given `X`.

(b) (1%) Write a Matlab function `[w2] = fitlinreg(X,y,lambda)` which returns a vector of weights `w2` corresponding to the minimum *regularized* sum of squared errors linear function. That is, return the weight vector `w2` that minimizes

$$\sum_{i=1}^{t}(\mathbf{w}\cdot\mathbf{x}_i - y_i)^2 + \lambda\sum_{j=1}^{n} w_j^2$$

Part 2 (Generalized linear least squares—3%)

(a) (2%) Write a Matlab function `[w3] = fitgenlinreg(X,y,lambda,featurefun)` which returns a vector of weights `w3` corresponding to coefficients for the features determined in `featurefun` that minimizes the regularized sum of squared errors of predicting `y` given `X`.

Note that your routine should take the name of a function `featurefun` as the last argument, and call this function using `feval` on the row vectors in `X`. A suitable feature function you can use, `'quadfeatures'`, is provided in a3files.zip.

(b) (1%) Write a Matlab function `[yhat] = predictgenlin(Xtest,featurefun,w3)` which returns a vector of predictions `yhat` for the row vectors in `Xtest` determined by the features in `featurefun` combined with weights `w3`.

Part 3 (Comparison—2%)

In this question you will compare the three different function estimation algorithms implemented above. They will be compared on synthetic data that has the form

$$\mathbf{X} = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,n-1} \\ \vdots & & & \vdots \\ 1 & x_{t,1} & \cdots & x_{t,n-1} \end{bmatrix} \quad \mathtt{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_t \end{bmatrix}$$

The data will be generated as follows

```
n = 2                                % dimension
t = 10                               % training size
u = [0; ones(n-1,1)]                 % target weights
sigma = 0.1                          % noise level
X = [ones(t,1) rand(t, n-1)]         % training patterns
y = (X*u).^2 + randn(t,1)*sigma % target values
lambda = 0.005                       % regularization parameter
```

Repeat the following steps 100 times and accumulate the sum of squared error for each kind of function in two tables: one for the training errors and one for the testing errors. Report the *averages* and the *variances* for the squared error of each kind of function in two tables (one training error and the other testing error).

A: Generate a random training set `X`, `y` using the model, and solve for each kind of function: `w1 = fitlin(X,y)`, `w2 = fitlinreg(X,y,lambda)`, and `w3 = fitgenlinreg(X,y, lambda,'quadfeatures')`. The feature function `'quadfeatures'` is in a3files.zip obtained from the course webpage.

B: Record the squared error that each of the three hypotheses obtained on the training data. To do this for each of the functions, you need to calculate the predictions `yhat` on the training data. For the first two functions, this can be easily done by calculating `yhat1 = X*w1` and `yhat2 = X*w2` respectively. For the third function, you need to use `yhat3 = predictgenlin(X,'quadfeatures',w3)`.

C: Generate `te = 1000` test examples using the same data generation process as above. Record the squared error that each of the three hypotheses generated in Step A obtained on the test data. This will require you to compute the `yhat` predictions on the test data, which can be done the same way as in Step B.

Remember to submit all written answers in the course drop box.

Part 4 (Linear least squares: Dual form—3%)

(a) (2%) Write a Matlab function `[a1] = fitduallinreg(X,y,lambda)` which returns a vector of training example weights `a1` such that $\mathbf{X}^\top \mathbf{a}_1 = \mathbf{w}_2$. That is, `a1` corresponds to the dual linear solution of `w2` for minimizing the regularized sum of squared errors for predicting `y` given `X`.

(b) (1%) Write a Matlab function `[yhat] = predictduallin(Xtest,X,a1)` which returns a vector of predictions `yhat` for the row vectors in `Xtest` determined by the dual weight vector `a1` combined with training data `X`.

Make sure to check that 4(a) combined with 4(b) yield the same final predictions as the weights `w2` determined in 1(b).

Part 5 (Kernel least squares—3%)

(a) (2%) Write a Matlab function `[a2] = fitdualgenlinreg(X,y,lambda,kernel,par)` which returns a vector of training example weights `a2` corresponding to the dual solution (for the feature representation implicit in the `kernel` function) that minimizes the regularized sum of squared errors of predicting `y` given `X`.

Note that your routine should take the name of a function `kernel` as the second last argument. The last argument `par` serves as a parameter that is passed to the kernel. So in particular, the kernel function must be called using `feval(kernel,X1,X2,par)`. A suitable kernel function you can use, `'quadkernel'`, is provided in a3files.zip. (Note that `'quadkernel'` corresponds to the same feature representation as `'quadfeatures'` above, but it does so in a much more efficient way.)

(b) (1%) Write a Matlab function `[yhat] = predictdualgenlin(Xtest,X,kernel,par,a2)` which returns a vector of predictions `yhat` for the row vectors in `Xtest` determined by the dual weight vector `a2` combined with the training data `X` using the features implicitly defined in the `kernel` with parameter `par`.

Make sure to check that 5(a) combined with 5(b), using the kernel `'quadkernel'`, yields the same final predictions as 2(a) combined with 2(b) using the feature function `'quadfeatures'`.

Note that `'quadkernel'` ignores the parameter `par`, but other kernels, such as `'gausskernel'`, do not.

Part 6 (Real data comparison—2%)

In this exercise you will tackle a real world prediction problem. The problem is to estimate how to predict the log volume of a prostate cancer tumor (*lcavol*) from 7 clinical measurements: total log weight of the prostate (*lweight*), age of the patient (*age*), log of benign prostatic hyperplasia amount (*lbph*), seminal vesicle invasion (*svi*), log of capsular penetration (*lcp*), Gleason score (*gleason*), percent Gleason scores 4 or 5 (*pgg45*).

The datafile data1.mat is provided in a3files.zip. If you type "`load data1.mat`" in Matlab, you will load the training data into a matrix `X` and a vector `y` and the test data into a matrix `Xtest` and a vector `ytest`. Each row of the matrices corresponds to a 7 dimensional vector containing the clinical measurements for a patient, and the corresponding entry in the associated y-vector gives the discovered log volume of the patient's prostate cancer tumor. The goal is to estimate a function that can predict the log volume of the prostate cancer tumor from the vector of 7 clinical measurements. Here, functions will be estimated on the training data and then tested on the separate test data.

Use the following parameters to estimate different prediction functions:

```
load data1.mat;
ww1 = fitlin(X,y);
ww2 = fitlinreg(X,y,25);
ww3 = fitgenlinreg(X,y,1e5,'quadfeatures');
aa1 = fitdualgenlinreg(X,y,1e5,'quadkernel',0);
aa2 = fitdualgenlinreg(X,y,0.005,'gausskernel',50);
```

Report the mean sum of squares error that each of these four functions make on both the training and the test data. Which of these techniques do you think performs the best on this data? Why?
Remember to submit all written answers in the course drop box.