Solutions to Homework 4 (covering Statistics Lectures 5 and 6)

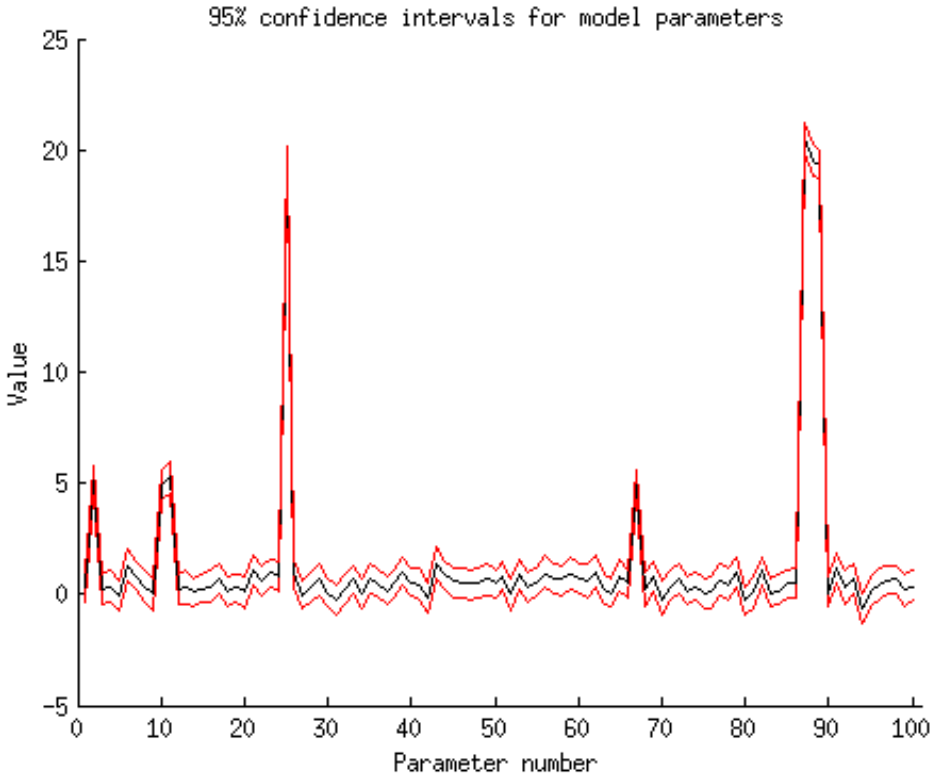
Contents

Problem 0

```
load('Homework4.mat');
```

Problem 1

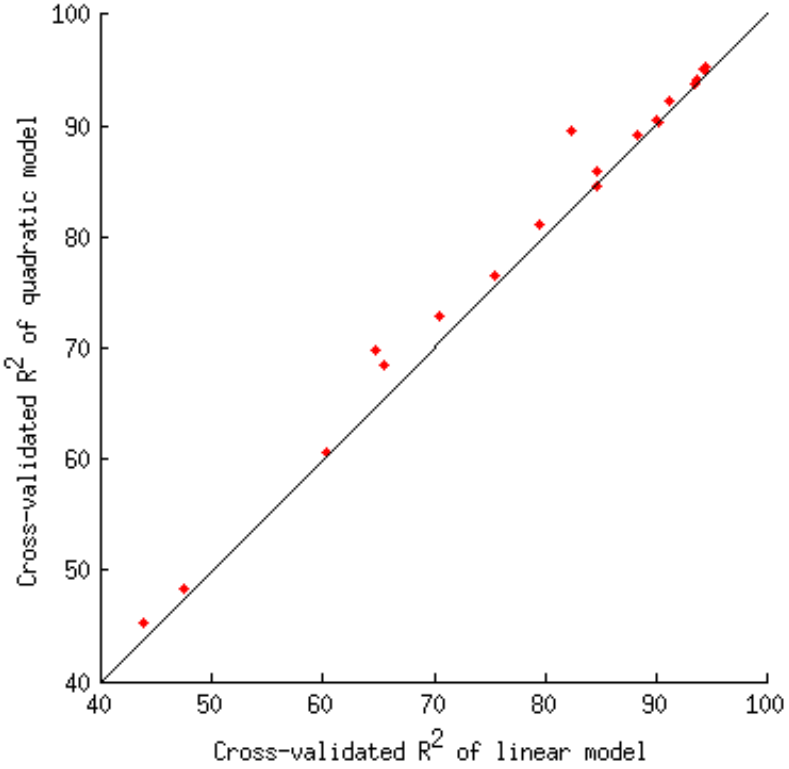
```
% define some constants
numboots = 500;

% calculate some quantities (to make the code general)
n = length(data1);                  % number of data points
numparams = size(regressors1,2);    % number of regressors

% initialize
params = zeros(numboots,numparams);  % this will hold the parameter estimates

% perform bootstraps
for boot=1:numboots
  ix = ceil(n*rand(1,n));       % construct vector of indices
  X = regressors1(ix,:);        % construct regressor matrix
  h = inv(X'*X)*X'*data1(ix);   % estimate parameters
  params(boot,:) = h;           % record parameters
end

% use percentiles to summarize bootstrap results
paramsP = prctile(params,[2.5 50 97.5],1);

% visualize
figure;
hold on;
plot(paramsP(1,:),'r-');
plot(paramsP(2,:),'k-');
plot(paramsP(3,:),'r-');
ax = axis;
axis([0 numparams+1 ax(3:4)]);
xlabel('Parameter number');
ylabel('Value');
title('95% confidence intervals for model parameters');
```



Problem 2

```
% define some constants
numfolds = 10;  % number of folds to use in cross-validation

% calculate some quantities (to make the code general)
numsubjects = size(xdata,2);
numdatapoints = size(xdata,1);

% generate a random split of the data points into parts.
% allix is a 2D matrix of indices, with dimensions [parts] x [indices].
allix = randperm(numdatapoints);           % all data indices, randomly ordered
numineach = ceil(numdatapoints/numfolds);  % at least one part must have this many data points
allix = reshape([allix NaN(1,numfolds*numineach-numdatapoints)],numfolds,numineach);

% define a R^2 function
computeR2 = @(model,data) 100 * (1 - sum((data-model).^2) / sum((data-mean(data)).^2));

% initialize
R2_linear = zeros(1,numsubjects);
R2_quadratic = zeros(1,numsubjects);

% analyze each subject
for subject=1:numsubjects

  % perform k-fold cross-validation
  prediction_linear = zeros(numdatapoints,1);
  prediction_quadratic = zeros(numdatapoints,1);
  for fold=1:numfolds

    % figure out data indices
```

```
    testix = allix(fold,:);            % indices to use for testing
    testix(isnan(testix)) = [];        % remove NaNs if necessary
    trainix = setdiff(1:numdatapoints,testix);  % indices to use for training

    % prepare regressor matrices and data
    Xtrain = [xdata(trainix,subject) ones(length(trainix),1)];
    Xtest =  [xdata(testix,subject) ones(length(testix),1)];
    ytrain = ydata(trainix,subject);
    ytest =  ydata(testix,subject);

    % train and test linear model
    h = inv(Xtrain'*Xtrain)*Xtrain'*ytrain;
    prediction_linear(testix) = Xtest*h;

    % prepare regressor matrices and data
    Xtrain = [xdata(trainix,subject).^2 xdata(trainix,subject) ones(length(trainix),1)];
    Xtest =  [xdata(testix,subject).^2 xdata(testix,subject) ones(length(testix),1)];
    ytrain = ydata(trainix,subject);
    ytest =  ydata(testix,subject);

    % train and test quadratic model
    h = inv(Xtrain'*Xtrain)*Xtrain'*ytrain;
    prediction_quadratic(testix) = Xtest*h;

  end

  % quantify accuracy
  R2_linear(subject)    = computeR2(prediction_linear,   ydata(:,subject));
  R2_quadratic(subject) = computeR2(prediction_quadratic,ydata(:,subject));

end

% visualize
figure;
hold on;
scatter(R2_linear,R2_quadratic,'r.');
axis equal square;
ax = axis;
plot(ax(1:2),ax(1:2),'k-');
xlabel('Cross-validated R^2 of linear model');
ylabel('Cross-validated R^2 of quadratic model');
```



Published with MATLAB® R2012b