Problem assignment 4
Due: Thursday, February 9, 2017

Linear Regression

In this problem set we use the Boston Housing dataset from the CMU StatLib Library that concerns prices of housing in Boston suburbs. A data sample consists of 13 attribute values (indicating parameters like crime rate, accessibility to major highways etc.) and the median value of housing in thousands we would like to predict. The data are in the file housing.txt, the description of the data is in the file housing_desc.txt on the course web page.

Part 1. Exploratory data analysis.

Examine the dataset housing.txt using Matlab. Answer the following questions.

- (a) How many binary attributes are in the data set? List the attributes.
- (b) Calculate and report correlations in between the first 13 attributes (columns) and the target attribute (column 14). What are the attribute names with the highest positive and negative correlations to the target attribute?
- (c) Note that the correlation is a linear measure of similarity. Examine scatter plots for attributes and the target attribute using the function you wrote in problem set 1. Which scatter plot looks the most linear, and which looks the most nonlinear? Plot these scatter plots and briefly (in 1-2 sentences) explain your choice.
- (d) Calculate all correlations between the 14 columns (using the corrcoef function). Which two attributes have the largest mutual correlation in the dataset?

Part 2. Linear regression.

Our goal is to predict the median value of housing based on the values of 13 attributes. For your convenience the data has been divided into two datasets: (1) a training dataset housing_train.txt you should use in the learning phase, and (2) a testing dataset housing_test.txt to be used for testing.
Assume that we choose a linear regression model to predict the target attribute. Using Matlab:

- (a) Write a function `LR_solve` that takes \( X \) and \( y \) components of the data (\( X \) is a matrix of inputs where rows correspond to examples) and returns a vector of coefficients \( w \) with the minimal mean square fit. (Hint: you can use backslash operator `/` to do the least squares regression directly; check Matlab’s help).

- (b) Write a function `LR_predict` that takes input components of the test data (\( X \)) and a fixed set of weights (\( w \)), and computes vector of linear predictions \( y \).

- (c) Write and submit the program `main4_2.m` that loads the train and test set, learns the weights for the training set, and computes the mean squared error of your predictor on both the training and testing data set. See rules for submission of programs on the course webpage.

- (d) in your report please list the resulting weights, and both mean square errors. Compare the errors for the training and testing set. Which one is better?

**Part 3. Online gradient descent**

The linear regression model can be also learned using the gradient descent method.

(a) Implement an online gradient descent procedure for finding the regression coefficients \( w \). Your program should:

- start with zero weights (all weights set to 0 at the beginning);

- update weights using the annealed learning rate \( 2/t \), where \( t \) denotes the \( t \)-th update step. Thus, for the first data point the learning rate is 2, for the second it is \( 2/2 = 1 \), for the 3-rd is \( 2/3 \) and so on;

- repeat the update procedure for 1000 steps reusing the examples in the training data if necessary (hint: the index of the i-th example in the training set of size \( n \) can be obtained by \( (i \mod n) \) operation);

- return the final set of weights.

(b) Write a program `main4_3.m` that runs the gradient procedure on the data and at the end prints the mean test and train errors. **Your program should normalize the data before running the method by finding the mean and std of the input attributes on the train data (see the lecture notes) and correcting/normalizing both the train and test inputs.** Run it and report the results. Give the mean errors for both the
training and test set. Is the result better or worse than the one obtained by solving the regression problem exactly.

(c) Run the gradient descent on the un-normalized dataset. What happened?

(d) Modify main4.3.m from part b. such that it lets you to progressively observe changes in the mean train and test errors. Use functions init_progress_graph and add_to_progress_graph on the course web page. The init_progress_graph initializes the graph structure and add_to_progress_graph lets you add new data entries on-fly to the graph. Using the two functions plot the mean squared errors for the training and test test for every 50 iteration steps. Submit the program and include the graph in the report.

(d) Experiment with the gradient descent procedure. Try to use: fixed learning rate (say 0.05, 0.01), or different number of update steps (say 500 and 3000). You may want to change the learning rate schedule as well. Try for example $2/\sqrt{n}$. Report your results and any interesting behaviors you observe.

Part 4. Regression with polynomials.

Assume we are not happy with the predictive accuracy of the linear model and we decided to explore a more complex model for predicting housing values. Assume we decide to use a quadratic polynomial to model the relation between $y$ and $x$:

$$f(x, w) = w_0 + \sum_{i=1}^{13} w_i x_i + \sum_{i=1}^{13} \sum_{j=i}^{13} w_{ij} x_i x_j.$$  

- (a) Write a function extendx that takes an input $x$ and returns an expanded $x$ that includes all linear and degree two polynomials.

- (b) What happened to the binary attribute after the transformation?

- (c) Write and submit a Matlab program main4.4.m that computes the regression coefficients for the extended input and both train and test errors for the result.

- (d) Report both errors in your report and compare them with the results in part 2. What do you see? Which method would you use for the prediction? Why? Please do not turn in the weights for this part in your report.