Machine Learning
Brendan Shillingford
July 6, 2016

Linear models

In this practical, we continue learning Torch and learn to implement linear models.

See the `README.md` file from Practical 1 for setup instructions for the lab machine; the first practical's files can be found here: https://github.com/oxford-cs-ml-2015/practical1

Linear neurons with SGD and least squares

We consider two possible implementations of linear models. The first implementation of linear models for this tutorial should be downloaded from:

https://github.com/torch/demos/tree/master/linear-regression

This demo implements online stochastic gradient descent for estimating the three parameters of a linear model.

1. Familiarize yourself with every step of the code. For information on stochastic gradient descent, I recommend the following Wikipedia pages:

   http://en.wikipedia.org/wiki/Stochastic_gradient_descent

   http://en.wikipedia.org/wiki/Gradient_descent

   http://en.wikipedia.org/wiki/Gradient

   Reading the above pages will maximize the opportunity for you to learn about optimization on Thursday's lecture.

2. Modify section 5 of the code (Test the trained model) to compute the predictions for the following test dataset of three observations and two input features (fertilizer and insecticide):

   ```
    dataTest = torch.Tensor{
      {6,  4},
      {10,  5},
      {14,  8}
   }
   ```

   What are the values of the three parameters? What happens to the parameters and predictions when the number of *epochs* is either `1e3` (1×10^3) or `1e5` (1×10^5)? **Hand in the answers to these questions.**

3. Implement the least squares solution

   $$\theta = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

   using the same dataset. What are the predictions for the above test set? How do they compare to the predictions of the linear neuron trained with SGD? How do the parameters compare? **Hand in your answers.**

Handin

See directions above. Hand in answers to 2 and 3.

Advanced: For enthusiastic students

Implement the nonlinear regression demos with polynomials described in Lecture 3. That is, generate data with a second order polynomial and use polynomials of different orders to estimate the nonlinear regression function. Confirm what happens as you vary the number of data, and the regularization coefficient.