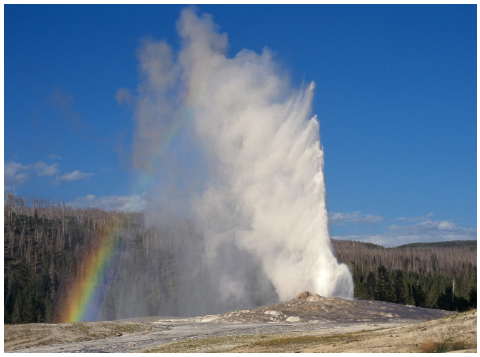
CMSC 352: Machine Learning
Linear Regression Exercise



The purpose of this exercise is to give you experience using the numpy and matplotlib modules of Python. You will also explore some of the practical problems of working with gradient descent algorithms.

The regressProblem.py program implements univariate linear regression via gradient descent. You should read through all functions into you understand what they do.

Before you can finish this exercise, you need to figure out how gradient learning applies to the minimization of the loss function J when there are two parameters, θ_0 and θ_1. This is usually handled by letting $x_0=1$ in the formula $h(\theta_0,\theta_1)=\theta_0 x_0+\theta_1 x_1$. Within the Python code, you will note that I prepend a 1.0 onto the data to set $x_0=1$ in this way. With regard to learning, if you carry out the gradient learning calculation of $\frac{\partial J}{\partial \theta_i}$ I did for i=1 in class, you will see that for $h(\theta_0,\theta_1)=\theta_0 x_0+\theta_1 x_1$ this works the same, since when differentiating with respect to θ_0, θ_1 is a constant.

1. The gradStep(dat,p) function is used to take one gradient step. It forms the sum of the $\frac{\partial J}{\partial \theta_i}$ over the different training inputs as we derived in class. Before going on you need to complete this function. You should try to predict the third variable (the one with index 2) from the bias (variable 0) and duration (variable 1). You can do this with loops or use numpy arrays and a bit of vector algebra.

2. Now try to find good parameters for the oldFaithfulLtd.m data. Old Faithful is a famous geyser in Yellowstone Park. This data file is has duration of eruption (seconds) and interval between eruption (minutes), in the first and second columns, respectively. You should read a bit more about this at http://www.geyserstudy.org/geyser.aspx?pGeyserNo=OLDFAITHFUL. Find a stable set of linear parameters and write them down as a linear equation. Based on your equation, what interval do you expect for a geyser with an eruption duration of 5 minutes?

3. Write a new version of gradStep called gradStepStochastic. This version of gradStep should implement stochastic gradient descent. In this case you will update the weight parameters each time a single training sample has occurred. This should be faster than regular gradient descent, but may require you to change the learning rate. Counting presentations of an entire data set, how much faster is this version of gradient descent on the geyser dataset?