# Machine Learning - Homework 3

Spring 2019

## Exercise 1 (10 pts)

For this exercise consider the Boston data set again, modeling medv in dependence of lstat. Fix the learning rate $\eta = 0.000009$. Set the initial values of the weights as $(\beta_0, \beta_1) = (30, 0)$ (this makes it converge more quickly than $(0, 0)$).

1. Write a stochastic gradient descent algorithm with 20 epochs for the Boston data set.
2. Write a mini-batch gradient descent algorithm with 20 epochs for the Boston data set.
3. Plot a learning curve for your stochastic gradient descent algorithm where the learning performance is measured by the RSE and the experience is given by the number of epochs, from 1 to 20.
4. Plot a learning curve for your mini-batch gradient descent algorithm where the learning performance is measured by the RSE and the experience is given by the number of epochs, from 1 to 20.
5. Measure and plot the run time for your stochastic gradient descent algorithm for the above number of epochs.
6. Measure and plot the run time for your mini-batch gradient descent algorithm for the above number of epochs.
7. Which combination of batch-size (1 or 32) and number of epochs would you recommend?

**Note:** Instead of 1. and 2. you can also write one general mini-batch gradient descent algorithm with variable batch size (where batch size 1 corresponds to stochastic gradient descent). For more generality you can also use a variable for the number of epochs. If you like you can then play around with different batch sizes and find the best batch size in terms of learning performance.

**Hints:**

- Instead of creating the batches before the loops of your actual algorithm (as I tried to do at the end of class), it is probably much easier to simply use the correct indices of the rows of the data (corresponding to the batches) inside the loops. For stochastic gradient descent this means you need an inner loop indexed in a variable $j$ and use `Data[j]` for each update. Going with $j$ from 1 to 506 is one epoch. For mini-batch (size 32) gradient descent your inner loop goes in $j$ from 1 to `floor(506/32)` and your $j$-th batch is given by `Data[((j-1)*32+1):(j*32)]`.
- For plotting the learning curve you can simply use something like `plot(1:NumberofEpochs,RSEoftheseEpochs)`. Analogously for the run times.
- To measure the run time you can use `sys.time()`, once at the beginning and once at the end of your algorithm, and the difference of the two will give you the run time.