Atta Ebrahimi

CS 1571

Rebecca Hwa

11/29/16

Linear Regression – Gradient Descent

Part One:

In order to choose the alpha values for this part I literally iterated from 0 to 1 on by a thousandth until I found a decent value. The value .232 worked decently well in the way I implemented alpha. Originally I had alpha just be used as is which was causing my program not to converge. In order to solve this, I divided alpha by the iteration number so that it was smaller as learning progressed. This allowed the program to converge and allowed me to have a decent error of 2.82 by the end of training.

The criteria I chose for convergence was just a small difference in change from the old weights to the new weights. The number I ended up using was .001, any change under that was deemed as convergence. This seemed to work pretty well, it could have been smaller but I thought this was decent as I didn’t want the change to be so low that it would run post-convergence.

Part Two:

The learning rate I chose was .0005, which is seemingly small but worked decently well. The reasoning for this choice was done in a similar way to the first part, but as I didn’t have enough time to do as robust a search I decided to just leave it at a low number but not ridiculously low. I left the convergence criterion the same as in part one.

The main change and most important reduction in error that I implemented was normalizing the features. This dropped the error from approximately 3000 to 28 which is huge. I ended up using most of the features. When deciding on which ones I wanted to eliminate I initially looked through and popped the features that I decided were unimportant based on just how much I personally thought they would apply. After that didn’t yield much change, I decided to look at the top scorers and see what variables tended to be important to them. From this I saw that health surprisingly didn’t matter, as well as family relationships, and in many cases study time didn’t matter either. Removing these features and normalizing did not yield a lot of change, but it did change it at least a little. I’ve noted my test cases in the output.txt

The trained model performed pretty well with the normalized data set with an error at roughly 28. I figured a low MSE is good so I also assumed less than 100 is good.