Iteration 1 Progress Report

Ray Smith, Josh Duchene, Francesca Tyler, Matthew Duff,
Nathan Lenar, Daniel Owens, Nash Wellington

The main issues we ran into involved how to store our input/output labels as the math for the neural network works easier when iterating over an array or arraylist, but we wanted to store them in a hashmap for faster access. We ended up having to research and write some functions to convert our hashmaps to arrays.

There were various bugs that needed fixing, such as counting occurrences being called multiple times continuing to add occurences of words, so there were duplicate counts, and various issues with the math used in the neural network. We went step by step running the functions to find where these issues are.

There were also issues with the network when setting the starting weights. None of us were experienced with this and we made the mistake of setting all weights to 1 originally, then 0.1, both of which made it impossible to train the network. We eventually did more research and found setting the weights randomly is a good approach, and each weight now starts at either 0.1 or -0.1 at random, which ended up working great.

Another issue that came up during iteration 1 was implementing a maven based project structure. Initially we had build the project as a general project, which made it incredibly difficult to 1) import dependencies from code that we were reliant on and 2) implementing tests. After rebuilding our project with maven we were able to more efficiently compile and test our code, although it may have caused some issues with github integration for a few of the members.

For the most part all features were implemented as planned. We originally wanted to have full functionality to read in the .csv to use as a training set and testing, but due to time limitations didn’t finish making the full connection. We can read in .csv files, and we can train and test the network, these are just separate parts at this stage.

We decided against using TestNG because we realized the JUnit had the functionality that we'd needed without the extra hassle of working with a framework placed on top of it. Eclipse has incredible JUnit integration which made it easy to make basic tests.

We have a very strong focus on unit testing and less on the integration testing in the first iteration of our code as a lot of the pieces don't integrate at all or are inter-related in such a way that one section relies completely on another. We're primarily using JUnit for integration testing. Despite some of its limitations, it succeeds at acting as a driver class to pull multiple modules together.

We also did not end up fully implementing with TDD plan as it can often be difficult to decide the behavior of the code before getting into the gritty details of it. With this part of the code we still attempted to keep testability in mind as it was being developed. Some of these functions tests couldn't be automated, especially relating to the neural network.