MATH96012 Project 2

Alexander John Pinches CID:01201653

November 15, 2019

Part 1.3

I chose the second version of the cost function as this allowed for vectorisation making it faster the principle disadvantage being this is only applicable in the CLR case.

Part 2.3

In python the code would be much more readable and easier to debug without the difficulties inherent in compiled languages and in python sticking to numpy esentially means running blocks of compiled C so is very fast. However Fortran will be faster because of optimisations made by the compiler across the whole algorithm and as we are evaluating this function many times small speed ups in individual runs can make a large difference to the overall time.

Part 2.4

In figure 1 we see that MLR achieves close to if we classified at random error rates for all m where as neural networks achieve close to 0 error rate. The only disadvantage of the neural networks is the increased training time for small m. This shows for this problem that neural betworks are far superior. The two series are seperated out with LR in figure 2 and the NN in figure 3. We see a clear increase in error rate as the number of classes for mlr however for nn we see the error rate remaining seemingly constant at 0. Highlighting how much better the predictive ability of the NN is.

Figures

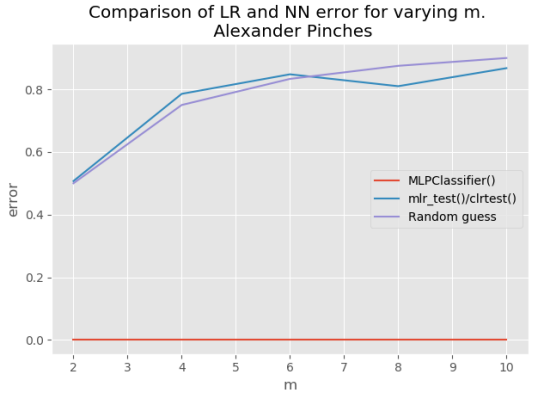

Figure 1: Figure for question 2.4 showing the error over m for mlr and neural networks

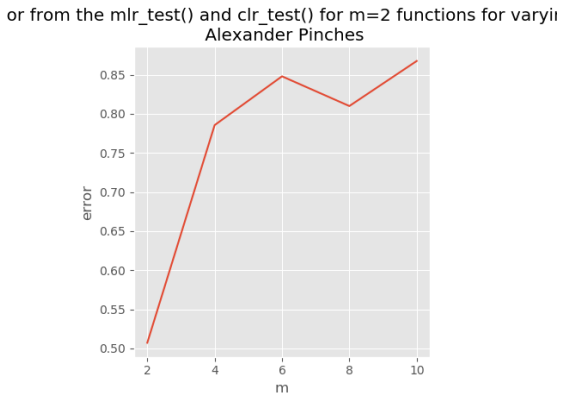

Figure 2: Figure for question 2.4 showing the error over m for mlr

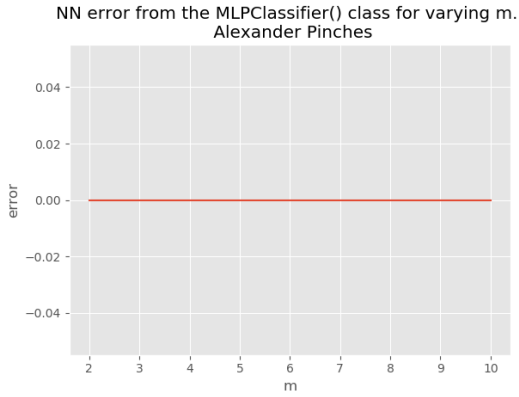

Figure 3: Figure for question 2.4 showing the error over m for neural networks