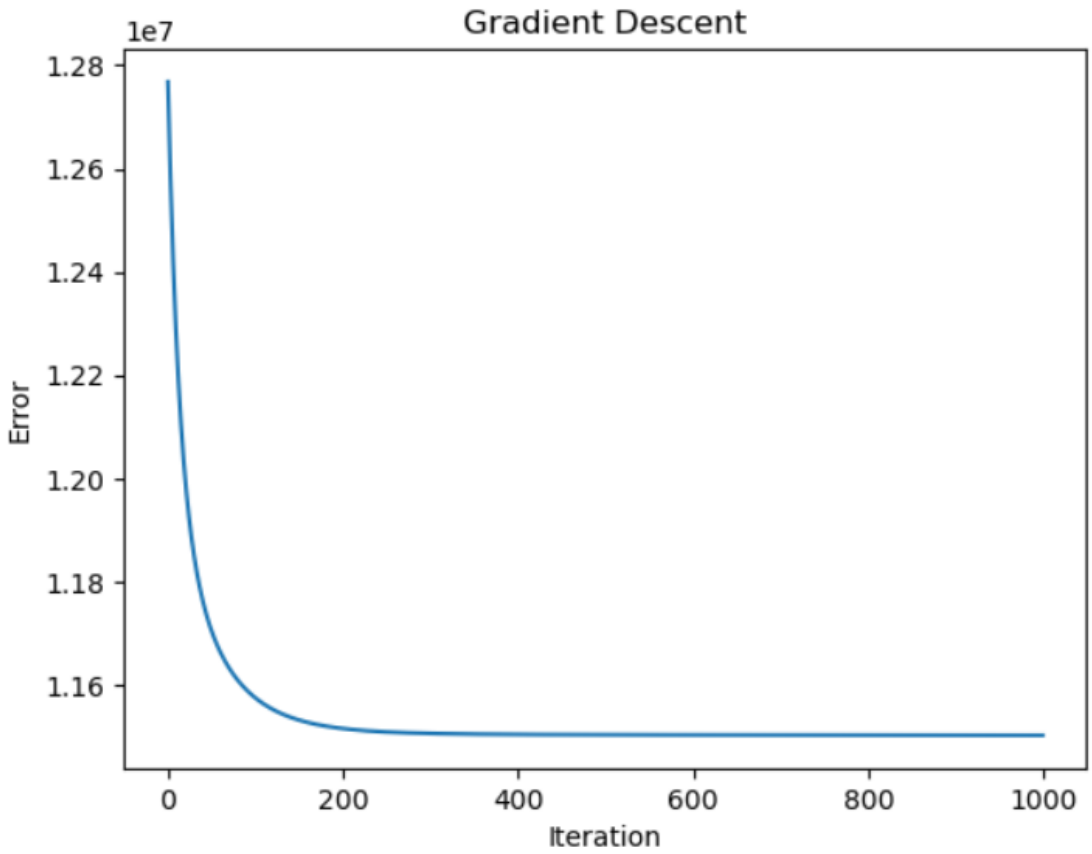
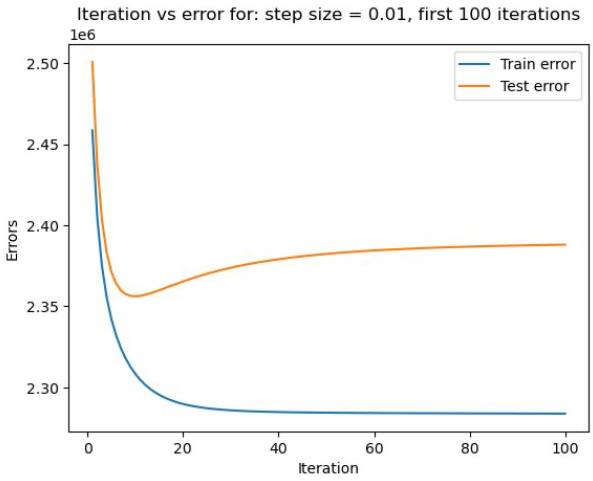
Assignment 2 – applied deep learning

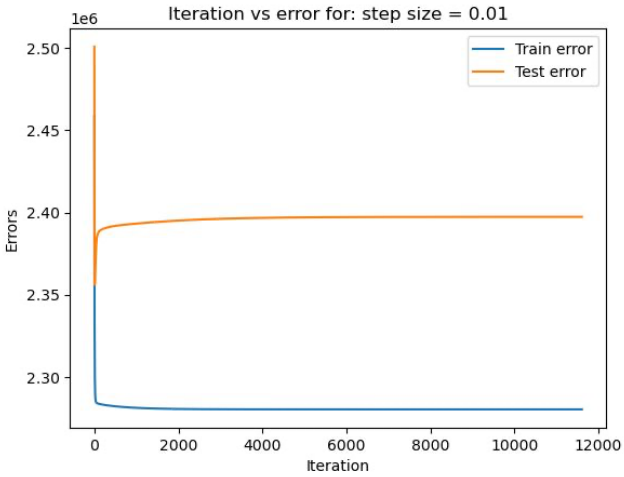
1.



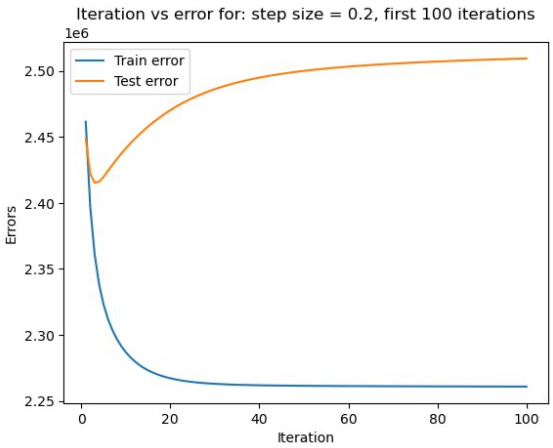
2.

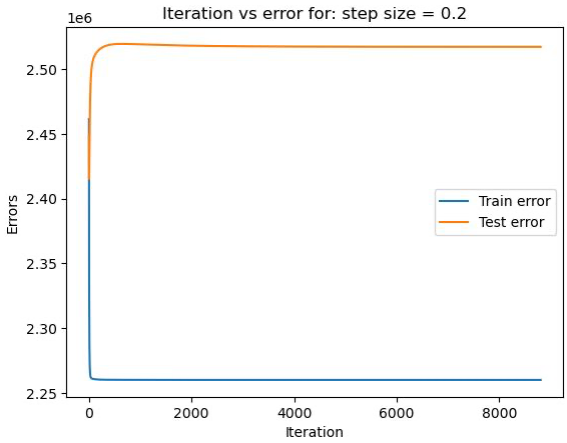
We ran the experiment with a few different hyper-parameters, and for each run we plotted two graphs: one full graph until convergence is reached, and one graph that is zoomed in for the first 100 iterations (easier to identify trends).





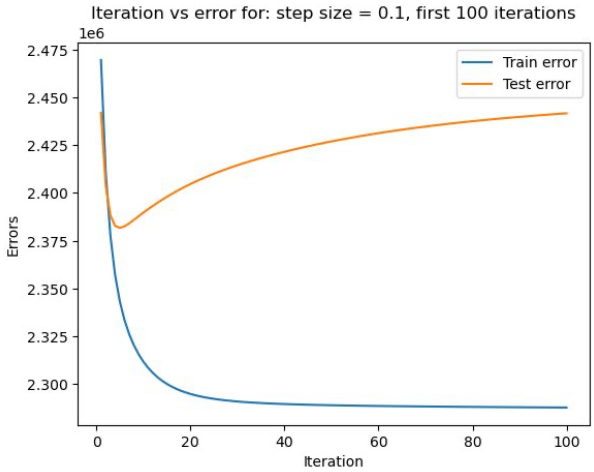
For a step size of 0.01, we can see convergence is reached after about 12000 iterations. We can clearly see the model trends from under-fitting to over-fitting, as at the beginning both train and test errors are high, and as more iterations are performed, we see the training error drops, but the test error increases.

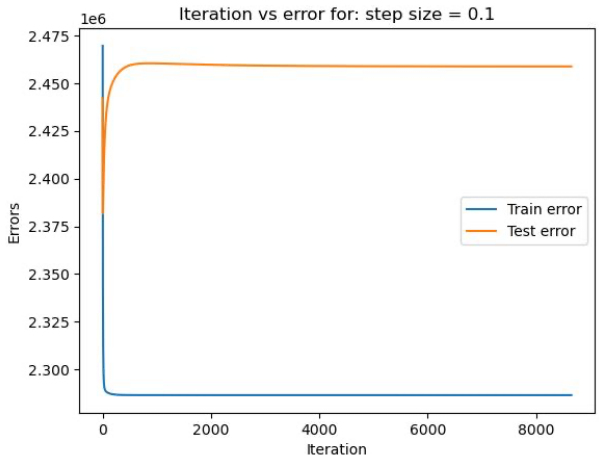




For a step size of 0.2, we see similar trends. Although, as we can expect for a larger step size, the model converges after about 8000 iterations.

Also, since the step size is larger, we can see the model is even more over-fitting than before, as the difference between the test error and train error is almost twice as large as before!



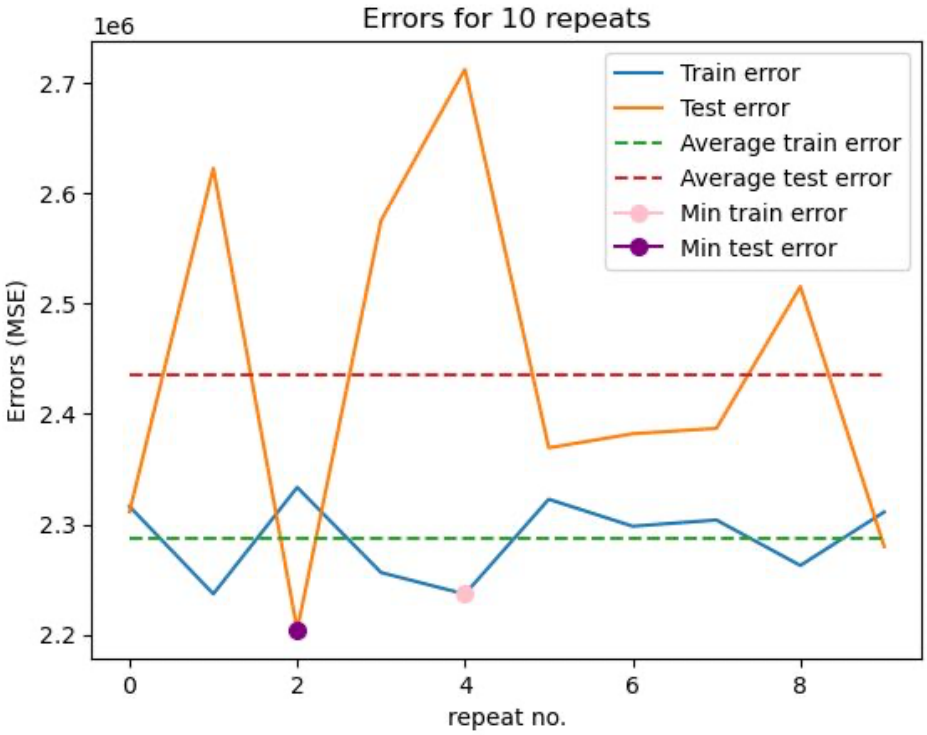


For a step size of 0.1, we see that the number of iterations for convergence stays about the same, since 0.1 and 0.2 are relatively the same size, but we can see that the model is slightly less over-fitted, since the difference between the train and test errors decreases.

We can deduct that by decreasing the step size, we see less over-fitting from the model (smaller difference between train and test errors) but slower convergence.

Note: since we run each experiment with a different data split chosen randomly, not all runs produce the same graphs. We chose the graphs that we think describe best the expected trends.

3.



We can see that on average for 10 runs of the experiment (with step size of 0.1 and stopping criteria of 1e-5) with random data splits, the model is overall over-fitting since the average train error is lower than the test error. But the minimum of these graphs shows us something interesting: the minimum of the test errors is lower the minimum of the train error!

This shows us that the fitting of the model to the data is strongly influenced by how the train and test data are split, the initial guess of the gradient descent algorithm and other parameters that we can't necessarily control.