COMP3105
Assignment 1
Alexander Breeze, 101 143 291
Victor Litzanov, 101 143 028

Part 1 e:

In the L2 loss and Linf loss columns, the L2 and Linf models respectively have the lowest losses, which is likely due to the fact that they were trained to minimize those losses while the other models were trained on different losses and thus had different priorities. The L1 loss does not follow this pattern, as the L2 model has a ridiculously good performance of approximately 10 billion times better than the L1 model, which is the next best.

We also see that the Linf model performs worst in both the L1 and L2 losses. This is likely due to L1 and L2 being relatively similar since they both try to find a line of best fit close to the majority of elements. Meanwhile, Linf has a great tendency to chase outliers.

When comparing the training losses to the test losses, we can see that the test losses are universally higher than their training counterparts. This is caused by the models overfitting the training data. The pattern of a model trained for a given loss model performing best when evaluated by that loss model holds for L1 and L2 but not Linf. This is due to the Linf overfitting for outliers in the training data which are no longer present in the test data, while L1 and L2 are more generalized.

We cannot figure out why the L2 model running on the training data with L1 loss function performs so much better. This result occurs reliably in our data. This does not apply to the test data, and an analysis of our code provided no evidence of any code discrepancy between the two, which forces us to conclude that somehow a model overfitted on L2 loss can perform nearly perfectly on the L1 loss function for the same data, better even than a model overfitted on the L1 loss function for that data.

Part 2 e:

As m increases the loss also increases, indicating that larger sample sizes cause the model to perform worse. This is likely due to more data points making a more complex model which, when trained for a static number of iterations, cannot properly model that additional complexity.

Inversely, increasing d decreases the loss, which is probably because the more information on each data point, the more that the model can pick and choose which weights better synchronize with the given data. This allows it to focus its weights on data attributes which more closely model the given data points.

The eta value increases with the loss, as it represents step size and with large steps the model is incapable of achieving the precision required to reach close to the ideal weights. We can even see the loss being approximately linearly proportional to the step size, indicating that the model can never get closer than approximately one step of the ideal weights.

Comparing the test and training results, they are very similar, essentially indistinguishable from each other. This indicates that the model is not overfitting, however it does not indicate whether or not the model is underfitting. Our best loss value was ~0.2, which

is not optimal. This indicates that our model is most likely underfitting due to insufficient number of training iterations in the gradient descent.

Part 3 c:

For linear regression on auto-mpg data our results are very similar to the results in 1d/e. The L2 model does best with L2 loss due to it being optimized for that loss function, and somehow is approximately 10 billion times better than the L1 model with L1 loss. Linf does best with Linf loss, once again due to it being trained for that loss function, but worst in both other types of loss due to it chasing outliers. In the test results all losses are greater, the L2 model with L1 loss no longer performs ridiculously better than other models, and the Linf loss performs somewhat poorly due to it chasing outliers in the training set, leading to it overfitting itself more than the other models which are more generalized.

For linear classification on parkinsons data our best result has a loss of 0.69. This correlates with our previous findings that our code, as per the assignment guidelines, was underfitting. We can also see that increasing step size increases our loss, likely due to our input being a small number of floats between 0 and 1. To properly model them, we require small and precise weights which cannot be learned with large step sizes.

For linear classification on sonar data our best result has a loss of 0.0025. It also does not follow a pattern, as it is equally good for each step size. This marked improvement over the parkinsons data, and the lack of influence from step size, are both likely due to the sonar data having many more data values per data point. This allows the steps to better cancel each other out regardless of their size.

For both linear classification problems, we can also see that the test score is not only slightly worse than the training score, but also becomes worse as step size increases. This is probably caused by the larger step sizes trying to cancel each other out by overfitting. Meanwhile, the smaller step sizes don't need to cancel each other out as much in the first place.

Part 3 d:

For the auto mpg data file, some data points have a '?' in column 4 (horsepower). The current solution is to simply remove data points containing a '?', but a better option would be to train a model to fill them in based on the other values. This would avoid wasting valuable data points. Also the model is designed to detect miles per gallon, with one of the data values being model year. Although different models of a car can have different stats, the year of a car only matter for that specific model of car and has no bearing on other ones. Although a more intelligent model, such as a neural net, could make this differentiation, our model is linear and thus cannot properly model for such nuanced data.

For the parkinsons data file, the first column (name) is a string. They are all identical, except for the numbers at the end of their name. A naive improvement would be to replace the string with just its number as an integer, and possibly reduce it to the range [0,1], but a more intelligent observation is that the name of the survey has no bearing on the data and thus the entire column can be deleted. This would prevent the model from putting any weight into that data value, as any weight on that essentially random value would be overfitting. That said, there is the possibility of biases between studies and so knowing which data comes from which study

could allow us to find outliers in the provided studies. This could be omitted for a better model or even investigated to determine the underlying causes of those anomalies.

The sonar data file is already fairly simplified, as it is composed almost entirely of floats. The one column of non-floats is already converted into binary labels. There is the possibility that when separating the data into train and test sets, effort could be made to select data points from sonar detection of an even distribution of angles from both rocks and metal tubes, as suggested by the description file, which would ensure that the model would be trained on every possible eventuality in the test file. However, we only have access to a 2D array of floats, so any such intelligent selection is impossible without further information about each data point in the file.