Turgut Alp Edis

21702587

GE461 Project 3 – Supervised Learning

a) Finding Configurations

The dataset given for this project is not sufficient for linear regressor because as it can be seen in figure 1, the sum of square loss is too large. Also, the values of given input and output are not linear (as it can be seen as points in figure 1). Therefore, ANN with a single layer is necessary to minimize the error and predict the values correctly.

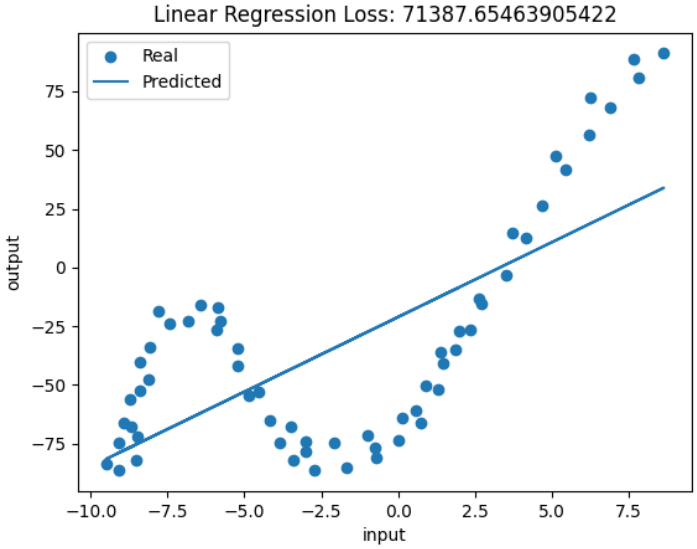

Figure 1. Linear Regression Line for Train Input and Output

For ANN, since it is mentioned in the part c of the project, I begin with using 2 hidden units, 10000 iterations (epochs) and 0.1% (0.001) learning rate. However, with these statistics, the error is approximately 165641. Since it is bigger than the error in linear regression, firstly, I adjust the number of hidden units. So, when I increase the hidden unit number to 4, the error suddenly drops to approximately 25349. So, I think that at least 4 hidden units will be enough to predict the values. When I increase the count of hidden units, the error decreases as it can be seen in table 1.

Number of Hidden Units	Error
2	166889.15
4	26518.58
8	22892.69
16	22043.44
32	15371.91

Table 1. Number of Errors for Hidden Units 2, 4, 8, 16 and 32

As a result, minimum number of hidden units will be 4, but for getting less error, I use 32 hidden units.

With 32 hidden units, I test the learning rate to find the ideal points of prediction. I begin with 32 hidden units, 10000 iterations (epochs) and 1% (0.01) learning rate. Then, I decrease the learning rate to 0.01% (0.001). After, I decrease the learning rate to 0.001% (0.0001). The error values can be seen in Table 2.

Learning Rate	Error
1% (0.01)	85257.05
0.1% (0.001)	15227.82
0.01% (0.0001)	31480.35

Table 2. Number of Errors for Learning Rates 0.01, 0.001 and 0.0001

As it can be concluded in Table 2, 0.1% is the best learning rate for this project because it has the minimum error since below and upper of that rate gives more error.

For weights, I used random values initially, then I rearrange the values according to the error of predicted output value. I used every index in sequential manner to make sure that the weight of layers is adjusted according to the input at least 1 time.

To test sufficient epoch number, I test the network with 100, 1000, 10000, 100000 epochs respectively.

Epoch Number	Error
100	126460.08
1000	34069.31
10000	15781.12
100000	9342.84

Table 3. Number of Errors for Epochs 100, 1000, 10000 and 100000

From Table 3, 100000 epochs give the best result, so I choose 100000 for this project to predict the values more accurately.

The effect of normalization is not significant since the values are already

b) Plots and Answers to the Questions

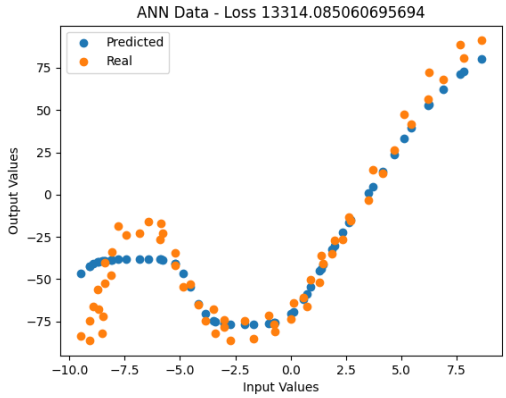

Figure 2. The Plot of Predicted Output from Train Data

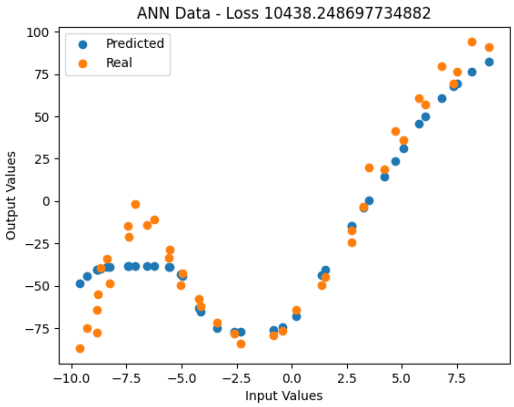

Figure 3. The Plot of Predicted Output from Test Data

ANN used (specify the number of hidden units): 32

Learning rate: 0.001

Range of initial weights: 0,1

Number of epochs: 100000

When to stop: 100000

Is normalization used: No

Training loss (averaged over training instances): 13314.08

Test loss (averaged over test instances): 10438.24

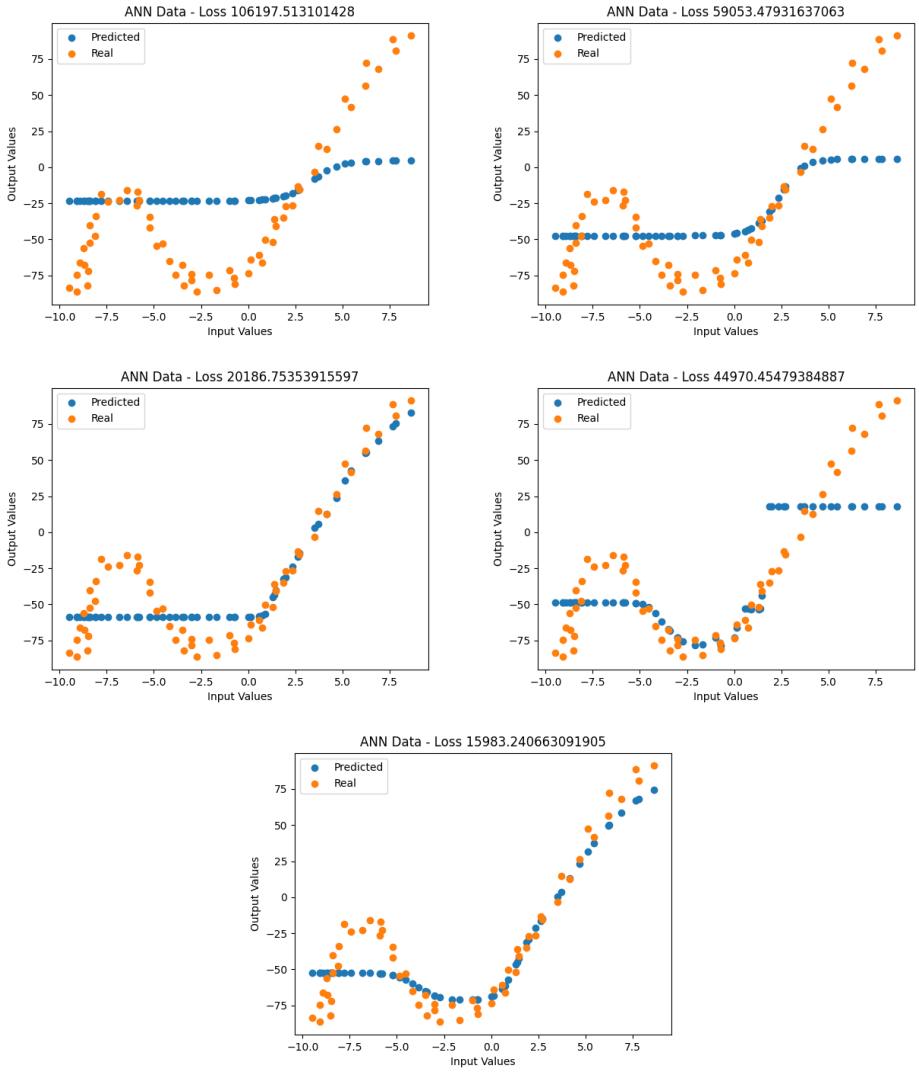

Figure 4, 5, 6, 7 and 8 Graphs of ANN with 2, 4, 8, 16 and 32 hidden unit respectively

Hidden Unit	Training Loss	Test Loss	Avg Train Loss	Std Dev Train	Avg Test Loss	Std Dev Test
2	197279.32	122780.31	3287.98	3317.61	2994.64	2497.69
4	61542.67	53297.70	1025.71	1036.25	1299.94	1084.97
8	20331.79	19137.48	338.86	343.63	466.76	390.44
16	53561.84	56257.58	892.69	902.11	1372.13	1145.15
32	21508.06	20560.39	358.46	363.40	501.47	419.37

Table 4. Error Statistics of ANN with different hidden unit count

Number of hidden units decreases the error when it increases. When the configurations maintain at same level for different hidden units as it can be seen in the Table 1, the error decreases. However, when the configurations are changed for each ANN, even if the number of units are many, the error can be bigger than the less hidden unit ANN. The reason is that the complexity is not only critical element for the neural network, but different configurations also affect the network as it can be seen in figure 4, 5, 6, 7 and 8 and table 4.