**Bilkent University - GE 461: Introduction to Data Science – Project 3 Report**

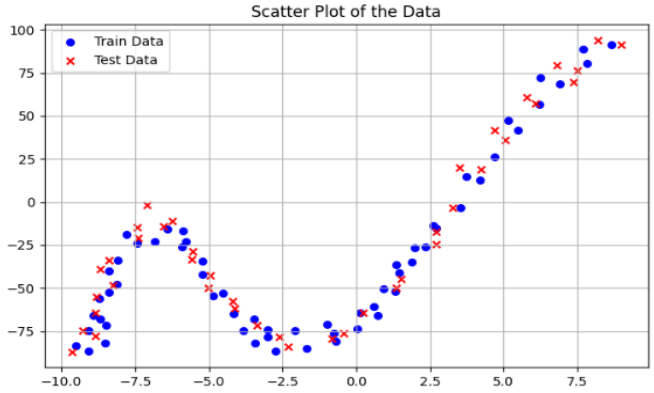


*Fig. 1. Scatter Plot of the Data*

**Linear Regressor or Single Hidden Layer?:** As we can see from the plot, It is evident from visualizing the data that this dataset requires more than a linear regressor. Third-degree polynomials appear to be the basis for the relationship between the input and the output. Additionally, without a hidden layer (linear regressor) has significant losses. Here are some examples of its results and visuals:

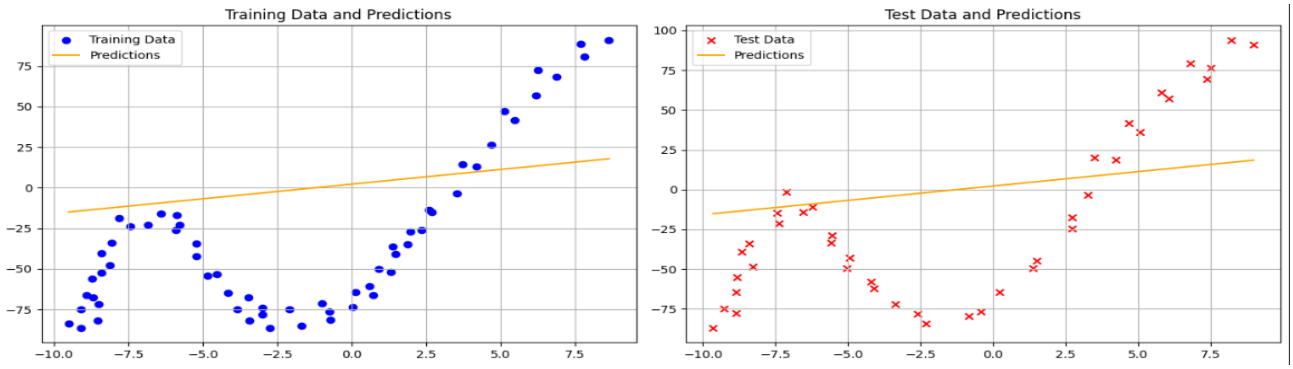


*Fig. 2. Plots of the Linear Regressor Model's Train and Test Dataset Predictions*

| Dataset | SSE | MSE | $R^2$ |
|---|---|---|---|
| Train | 166646.81 | 2777.45 | 0.49 |
| Test | 99178.56 | 2418.99 | 0.57 |

*Table. 1. Results of the Linear Regressor Case*

It is clear that simple model is unable to adequately represent the underlying data distribution, as seen by the high MSE scores and low $R^2$ measure.

Now let's try a model with a single hidden layer model to see whether it is enough to capture the relationship. Here are the results and visualizations:

| Dataset | SSE | MSE | $R^2$ |
|---|---|---|---|
| Train | 36479.32 | 607.76 | 0.74 |
| Test | 29124.82 | 710.36 | 0.76 |

*Table. 2. Results of the Single Hidden Layer Case*

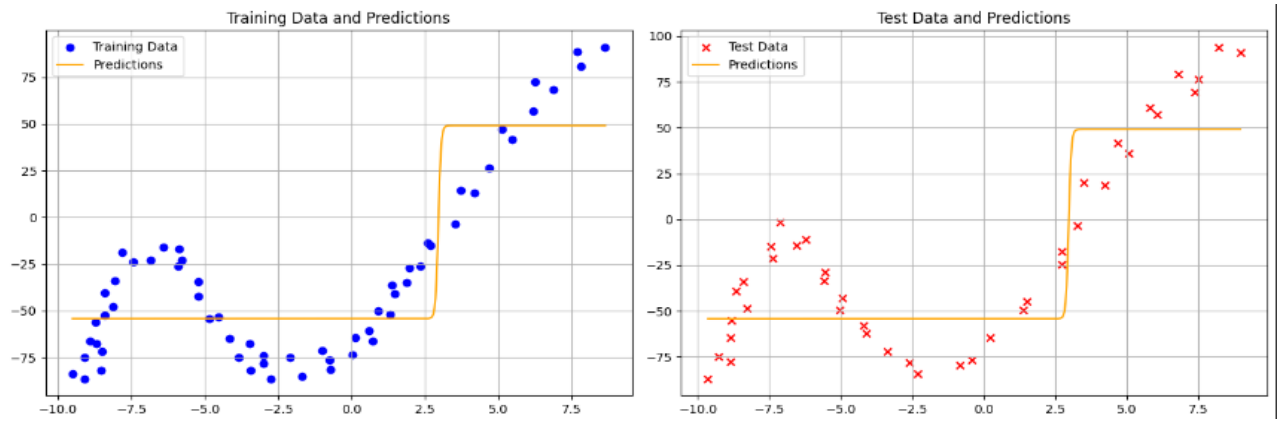


*Fig. 3. Plots of the Single Hidden Layer Model's Train and Test Dataset Predictions*

We need more hidden units since it is evident that a model with only one hidden unit is unable to completely understand the underlying distribution of the data.
**Number of Hidden Units Required:** We may examine the plots and loss values for each model with various hidden units to get an estimate of the least number of hidden units. I have experimented with each of numbers 2,4,8,16, and 32 individually on the test dataset since they are specified in section C. Only loss values are shown since we have a page constraint:

| # of Hidden Units | SSE | MSE | $R^2$ |
|---|---|---|---|
| 2 | 20972.85 | 511.53 | 0.82 |
| 4 | 19271.99 | 470.05 | 0.84 |
| 8 | 13441.51 | 327.84 | 0.89 |
| 16 | 12492.83 | 304.70 | 0.90 |
| 32 | 12059.66 | 294.14 | 0.90 |

*Table. 3. Results of Different Number of Hidden Units and Loss Values*

Outcomes are better with larger hidden unit sizes. As a result, 8 would be the absolute least, but 32 units is what I would advise because of its better graph.

**Good Learning Rate:** We discovered that 32 is the ideal hidden layer size, therefore I'll adjust number of layers to 32 hidden layers. I'll look for values of $10^{-5}$, $10^{-4}$, $10^{-3}$, $10^{-2}$, and $10^{-1}$ and examine the test set's plots and loss value for each learning rate. The following are the results:

| Learning Rate | SSE | MSE | $R^2$ |
|---|---|---|---|
| $10^{-5}$ | 24828.23 | 605.57 | 0.82 |
| $10^{-4}$ | 18406.20 | 448.93 | 0.85 |
| $10^{-3}$ | 12020.89 | 293.19 | 0.90 |
| $10^{-2}$ | 55891.90 | 1363.22 | 0.59 |
| $10^{-1}$ | 71806.51 | 1751.38 | 0.68 |

*Table. 4. Results of Different Learning Rates and Loss Values*

As we can see, a learning rate of $10^{-3}$ produced the lowest loss value. Consequently, we may say that a learning rate of $10^{-3}$ is ideal.
**Number of Epochs:** I experimented with epochs of 1000, 5000, 10000, 20000, 50000, 100000, and 200000 and set hidden units as 32 and learning rate as $10^{-3}$. The following are the outcomes:

| # of Epochs | SSE | MSE | $R^2$ |
|---|---|---|---|
| 1000 | 28611.73 | 691.85 | 0.80 |
| 5000 | 20787.65 | 507.02 | 0.85 |
| 10000 | 18728.06 | 456.78 | 0.84 |
| 20000 | 19332.77 | 471.53 | 0.85 |
| 50000 | 14044.88 | 342.56 | 0.90 |
| 100000 | 13249.50 | 323.16 | 0.90 |
| 200000 | 11461.03 | 279.54 | 0.90 |

*Table. 5. Results of Different Number of Epochs and Loss Values*

By looking the results we see that 200000 epochs yield the best results. Note that we tested optimal numbers of hidden layers, learning rate, and numbers of epochs iteratively. This can lead us to

stuck in local optima. Hence, I've also tried each combination of these values and the results are same.

**Initialization of Weights:** I've used Xavier Initialization since when learning for sigmoid activation functions, random weight initialization between 0 and 1 might kill the learning. After multiplication and the sigmoid function, small random weights frequently result in activations that are near to zero since the sigmoid squashes values towards 0. As a result, the network finds it challenging to learn intricate patterns. This is addressed by Xavier initialization by ensuring that the input variance and the output variance of each layer are equal. In training, this helps keep the gradients from disappearing or blowing up.

**Effect of Normalizing:** I initially computed the mean and standard deviation of features in training data, then subtracted the mean and divided by the standard deviation. This adjusts the features to have a zero mean and unit variance, which may result in faster convergence, and possibly enhanced generalization. Plot is shown:

**Question 1B)**

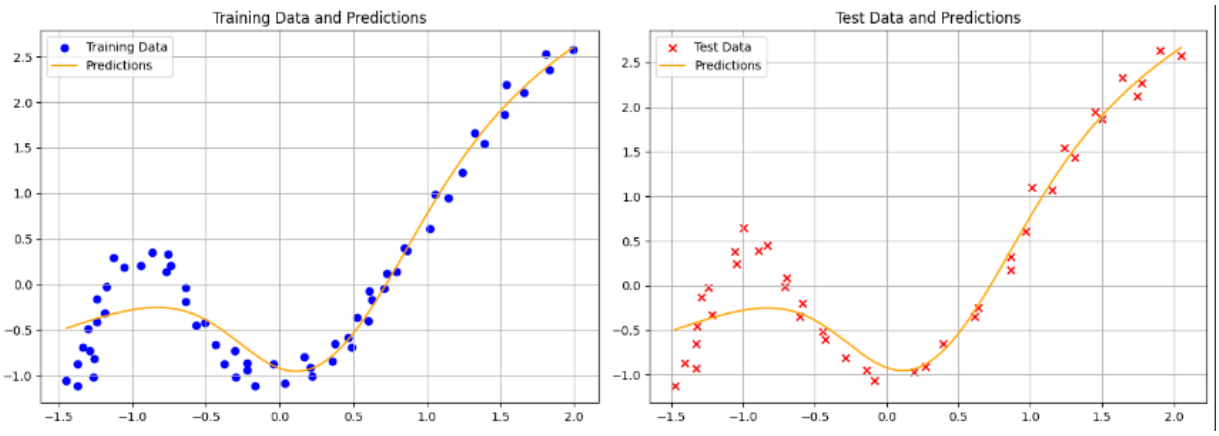


*Fig. 4. Plot with Normalized Model's Train and Test Dataset Predictions*

After the normalization, $R^2$ value increased 0.90 to 0.92 and decreased loss values. So, I decided to use normalization of the data.

ANN used (specify the number of hidden units): **32**
Learning rate: $\mathbf{10^{-3}}$
Range of initial weights: $\left(-\mathbf{3}\sqrt{\frac{\mathbf{2}}{\boldsymbol{n_{in}+n_{out}}}}, \mathbf{3}\sqrt{\frac{\mathbf{2}}{\boldsymbol{n_{in}+n_{out}}}}\right)$ where $n_{in}$ and $n_{out}$ is number of input and output neurons respectively
Number of epochs: **200.000**
When to stop: After epochs are finished
Is normalization used: **Yes**
Training loss (averaged over training instances): **194.8472**
Test loss (averaged over test instances): **252.4897**

**Question 1C)**

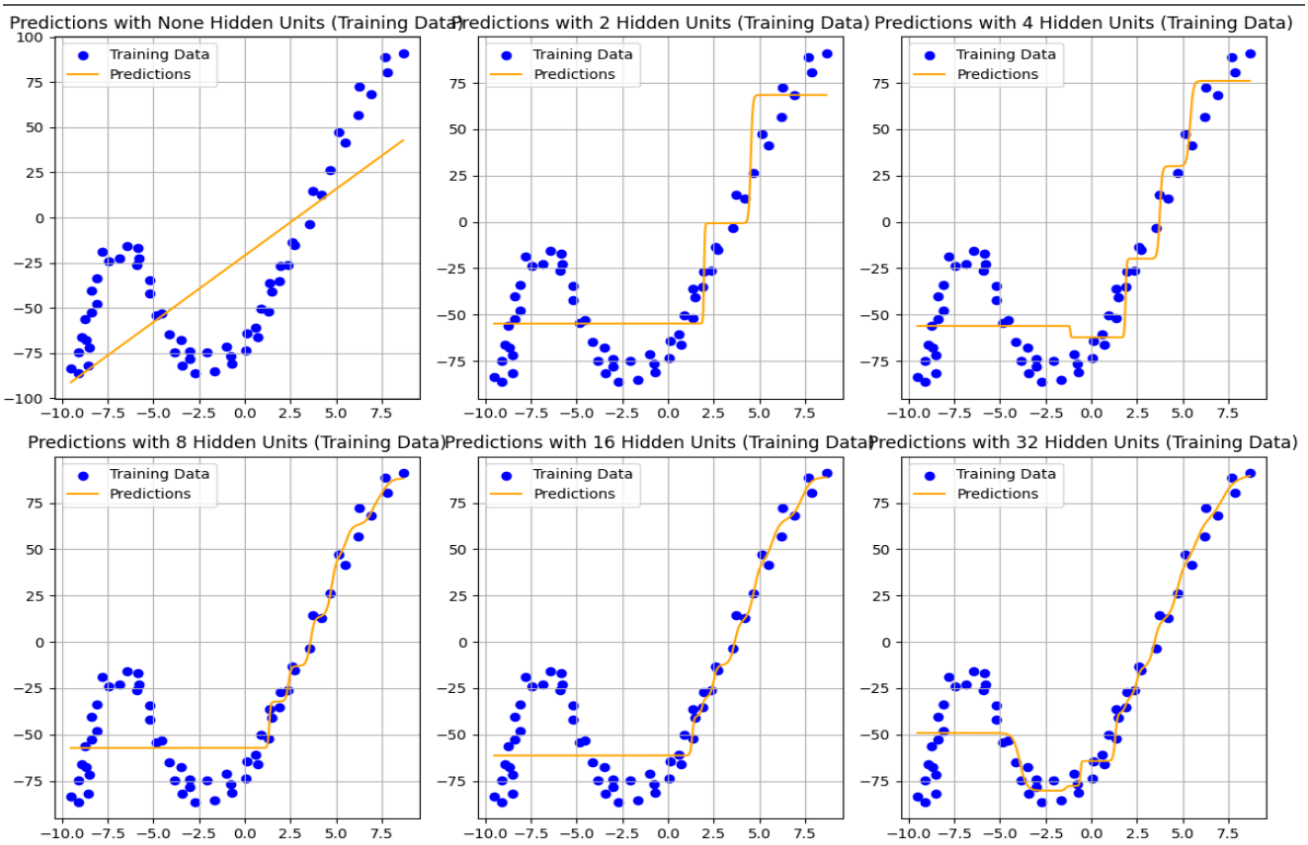


*Fig. 5. Plots of Model's with different hidden units Train Dataset Predictions*

| Hidden Unit Size | Average Train Loss | Average Test Loss | Std. Dev. Train Loss | Std. Dev. Test Loss |
|---|---|---|---|---|
| None | 1330.81 | 1503.04 | 1082.76 | 1415.12 |
| 2 | 576.07 | 718.95 | 580.46 | 757.23 |
| 4 | 465.08 | 582.18 | 559.39 | 645.41 |
| 8 | 223.90 | 333.56 | 355.67 | 494.55 |
| 16 | 227.21 | 338.74 | 363.32 | 550.65 |
| 32 | 214.85 | 282.08 | 385.63 | 470.91 |

*Table. 6. Results of of Model's with different hidden units loss values and std. devs*

The plots and table show that model complexity impacts prediction quality. Increase in hidden unit number resulted in considerable improvements in training set loss. However, after 8 to 16 number of units, the test set loss stopped improving and worsened, indicating overfitting. These findings emphasize the necessity of determining the ideal network complexity in order to avoid overfitting and achieve the highest rate of generalization.