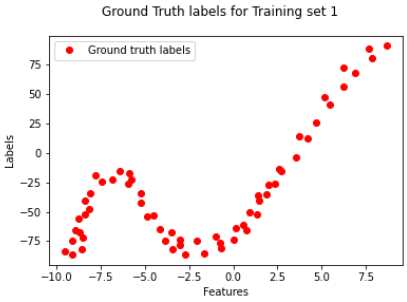
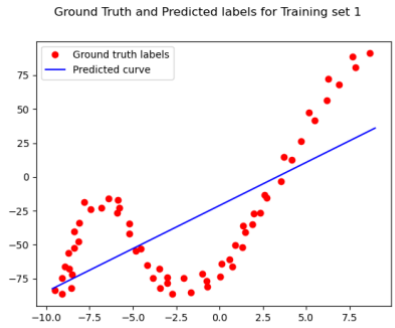
a) For Dataset 1

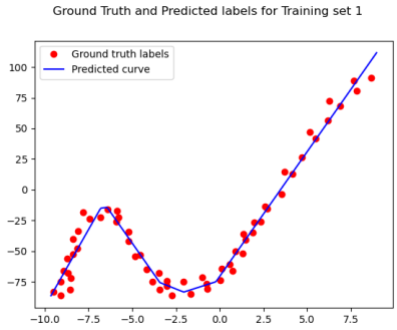
There are 60 training samples and 41 test samples in the first dataset. To see the data, the ground truth labels over features are shown below.

Since the data is not linear, using a linear regressor would not work in this case. Hereupon, below shown the prediction curve of the linear regression model.





On the other hand, the ANN model with 1 hidden layer has been tested with {16, 8, 4, 2, 1} hidden units. The best performing has been with 16 units with the 8-units-model performed very similarly. The activation function has been selected as Leaky ReLU among {Sigmoid, ReLU, Tanh, Leaky ReLU}. The loss function has been selected as Mean Squared Error because of the task being linear regression. The model was tested with different learning rates, the best performing one was selected among {0.001,0.0001,0.00001}. The weight initialization has been done with Xavier/He initialization. Because initializing with zeros or ones do not perform any good in ANN models as the number of units will not make any difference in that case. Not surprisingly, the model worked better with stochastic gradient descent than with full-batch gradient descent. Use of momentum is not necessary, but since it speeds up the learning by avoiding local optima and make gradient descent go towards a global minimum, it is used. The normalization significantly affects the learning in this application, because of vanishing and exploding gradients problem, so the standard scaling is used.

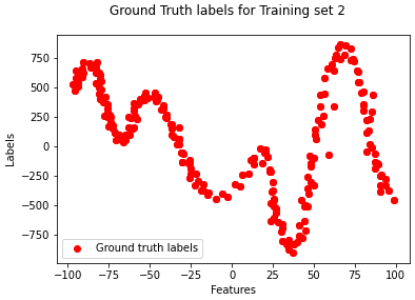


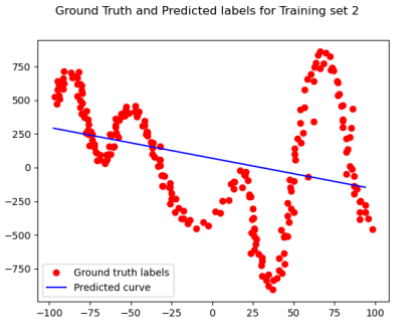
ANN used (specify the number of hidden units): 8
Selected activation function: Leaky ReLU
Selected loss function: MSE
Learning rate: 0.0001
Range of initial weights: [-1, 1] in the first layer, [-0.35, 0.35] in the last layer
Number of epochs: 3000
When to stop: Early stopping on training loss
Is momentum used (if so, value of the momentum factor): 0.95
Is normalization used: Standard scaling
Stochastic or batch learning: Stochastic
Training loss (averaged over training instances): 0.0247
Test loss (averaged over test instances): 0.0382

a) For Dataset 2

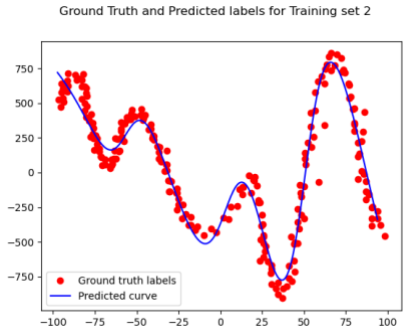
There are 229 training samples and 92 test samples in the first dataset. To see the data, the ground truth labels over features are shown below.

Since the data is not linear, using a linear regressor would not work in this case. Hereupon, below shown the prediction curve of the linear regression model.



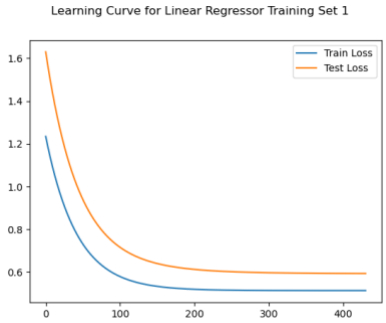


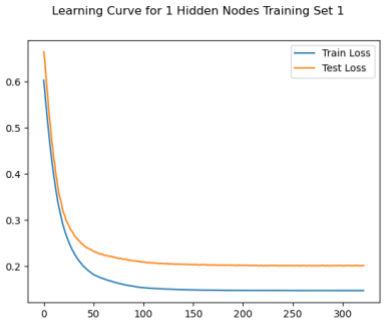
On the other hand, the ANN model with 1 hidden layer has been tested with {16, 8, 4, 2, 1} hidden units. The best performing has been with 16. The activation function has been selected as Tanh among {Sigmoid, ReLU, Tanh, Leaky ReLU}. The loss function has been selected as Mean Squared Error because of the task being linear regression. The model was tested with different learning rates, the best performing one was selected among {0.001,0.0001,0.00001}.

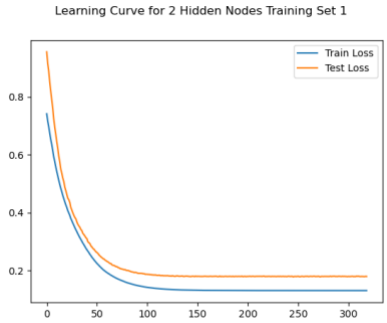


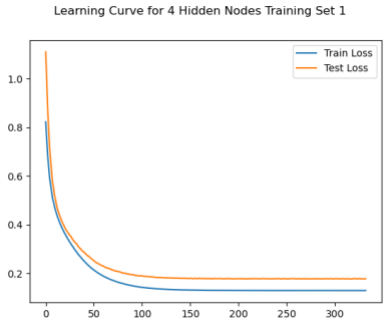
ANN used (specify the number of hidden units): 16
Selected activation function: Tanh
Selected loss function: MSE
Learning rate: 0.0001
Range of initial weights: [-1, 1] in the first layer, [-0.25, 0.25] in the last layer
Number of epochs: 6000
When to stop: Early stopping on training loss
Is momentum used (if so, value of the momentum factor): 0.9
Is normalization used: Standard scaling
Stochastic or batch learning: Stochastic
Training loss (averaged over training instances): 0.0951
Test loss (averaged over test instances): 0.216

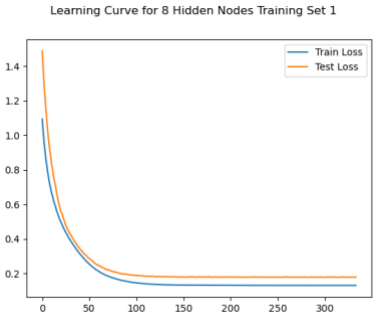
c) For Dataset 1

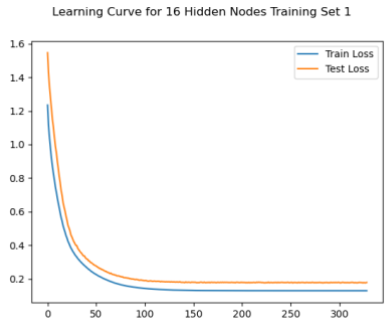




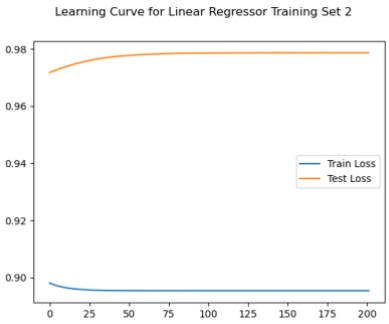


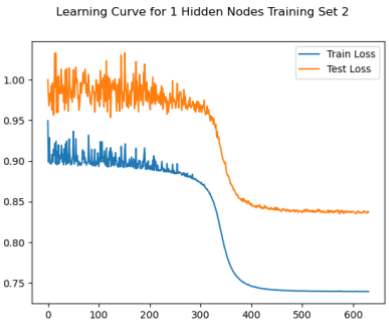


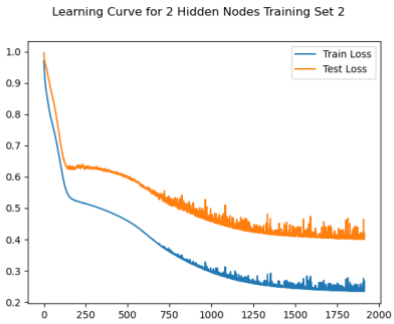


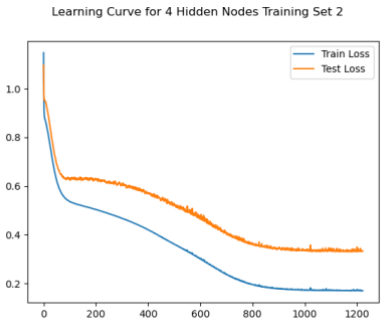


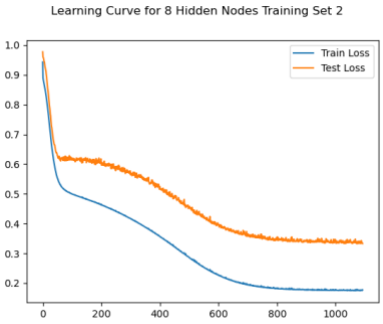
c) For Dataset 2

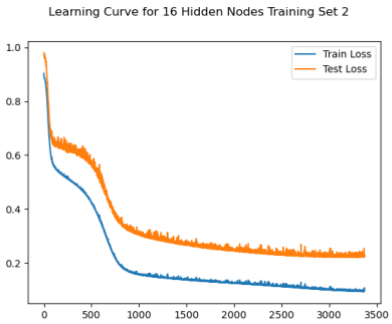






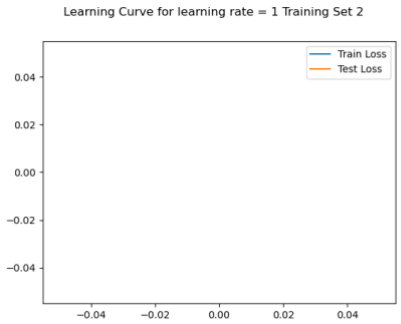


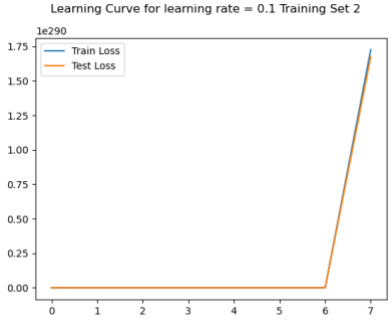


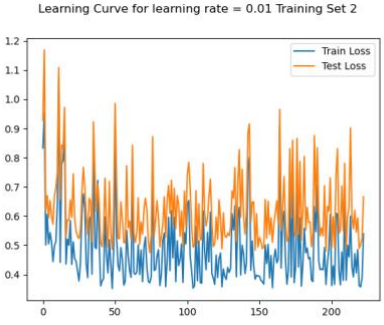


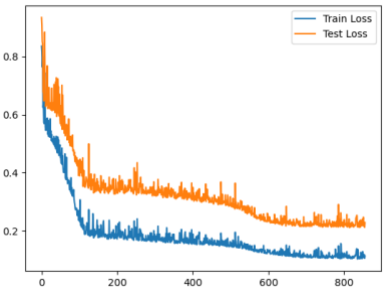
The more hidden units, the more convergence in the learning curve. To say, when the complexity of the model increases, the loss can get even more decreased, however the longer training time (more epochs). The learning curve of the linear regressor for the second dataset has shown that the complexity is not enough to generalize anything. So when the complexity is increased the learning started.
The important thing to consider when increasing the number of hidden units is that there is always a possibility to overfit to the training set, even though there has not been any sign in these models.

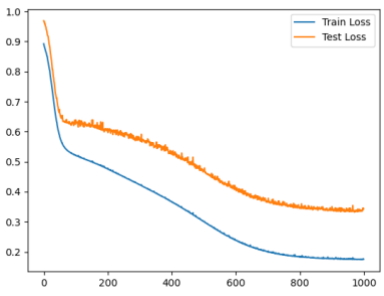
d) For Dataset 2





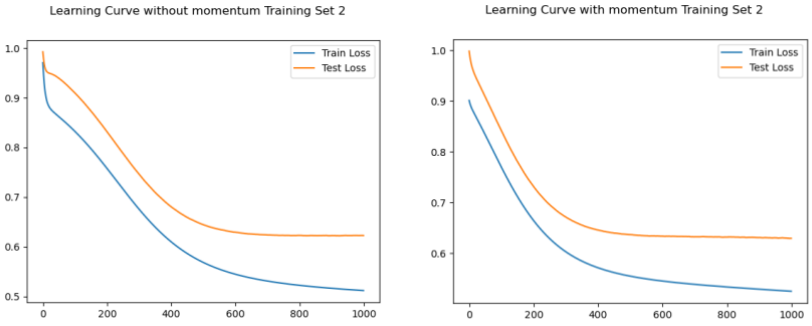






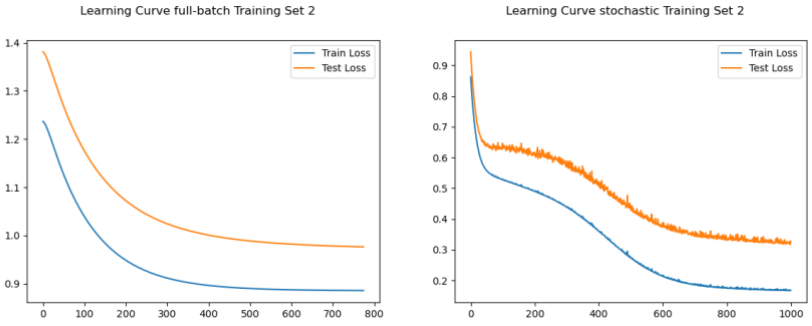
Early stopping on training loss decrease has been utilized in all models. The criterion to stop is when the last 100 epochs' average training loss mean is not higher than previous 100 epochs before that with 0.01, then the training is stopped.
This way, the learning rate can be analyzed by the epoch number that the model halted training. As can be seen from the Figures 1 and 2, where the learning rates are 1 and 0.1 respectively, there has been no learning at all, because the learning rate was too high.
When the learning rate is 0.01, the learning curve has fluctuated, because this rate is still too high, and some batches generalized the model whereas others made it worse. But in the 0.001 model the learning took 800 epochs and despite the fluctuation the model performed well. On the other hand, in the 0.0001 model, the learning took more epochs but performed very similarly to the 0.001 model, so we can say that this learning rate might be too low for our model.

e) For Dataset 2



The momentum by the definition should speed up the learning by driving the gradients towards the optimal minima and prevent them to end up in a local minimum. On left no momentum is used and on right a momentum of 0.9 is used. Both models performed similarly, but the model without momentum had suffered from a possible local minimum in the first epochs. It seems that the momentum did not change anything, however if the number of epochs were higher and the learning rate were lower, then momentum would affect the models training more.

f) For Dataset 2



The full-batch training worked better with higher learning rates, whereas stochastic gradient descent performed well with lower learning rates. The number of epochs were kept the same, however due to the early stopping criterion full-batch model took less epochs than stochastic model. This is because the training loss did not converge more, and the training stopped.
The stochastic gradient descent model performed much better than the full-batch gradient descent model, as can be seen from figures the training and test losses had 4-5-fold difference in two models.

Gün Kaynar
22101351
Bilkent University, CS