Università della Svizzera italiana

Institute of Computing CI

Deep Learning Lab **2021**

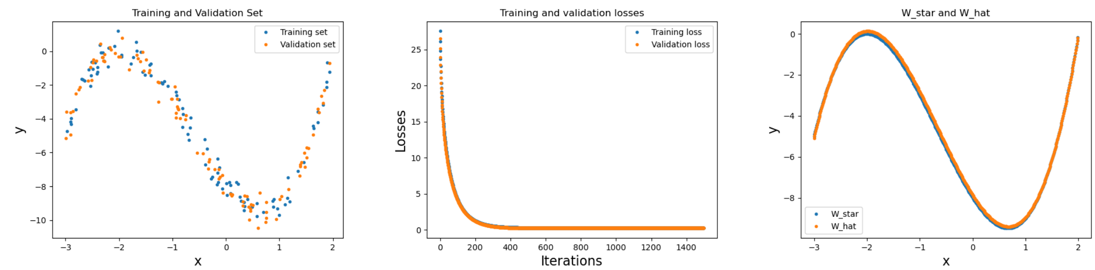
Student: Naga Venkata Sai Jitin Jami

---

1. Polynomial Regression

$$p(x) = x^3 + 2x^2 - 4x - 8 \tag{1}$$

Using *nn.linear*, *optim.SGD* methods from the *pyTorch* library to build a single layer neural network to find the parameters in *w_star* using Stochastic Gradient Descent optimization. The suitable *learning_rate* and *max_iterations* was determined to be 0.01 and 1500 respectively.



(a) Training and Validation set

(b) Losses

(c) W_star vs W_hat

Figure 1: Linear Regression

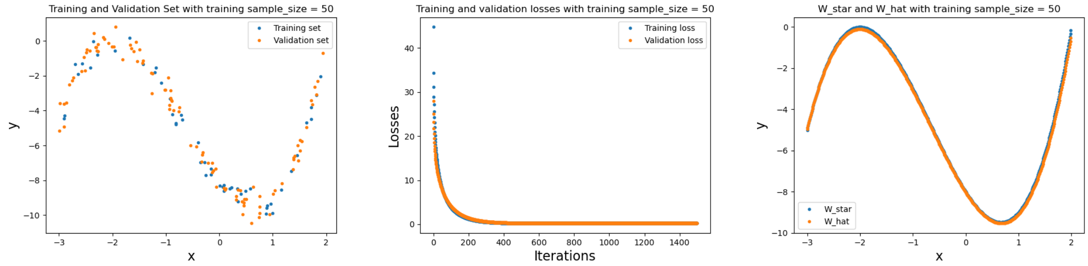
In the plot 1b, we can see that the training and validation loss steadily drops as more iterations go by. This is expected of an SGD optimizer where we are minimising loss by finding the negative gradient of the loss function. It can be noted that the max number of iterations can actually be reduced to the range of 600-800 since the optimization problem seemed to have converged.
In plot 1c, the plot for the polynomial defined by input vector *w_star* and result of linear regression *w_hat* with constant w_0 can be seen. Visually speaking, the results are pretty accurate and mathematically the losses have been minimized to a magnitude of 0.25.

1.1. Reducing training data

Reducing the training data to 50, 10 and 5 leads to interesting results. As the number of training samples keep dropping the training data of course becomes more sparse on the plot with training and validation set.

1.1.1. Training Data = 50 samples

In decreasing the training samples to 50, the loss vs iteration graph in 2b doesn't look very different from the original case. But that is visually of course, mathematically we're seeing a slightly underfit model because there's not enough training data to learn from. The validation losses are also reducing so its safe to say that the general trend has been learnt pretty well, evident from the *W_star* vs *W_hat* where the equation line and prediction line are overlapping.

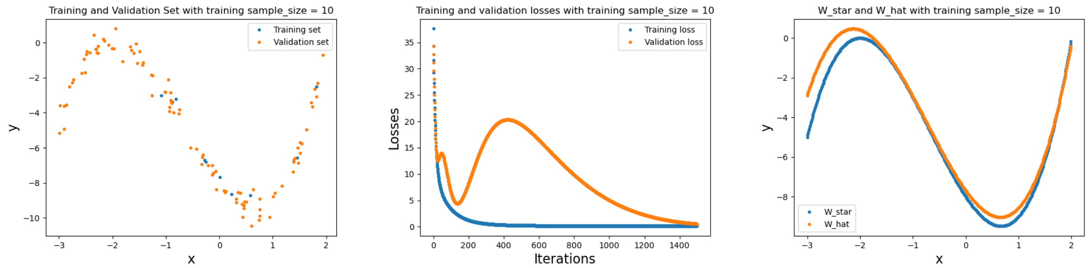


(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 2: Training sample_size = 50

1.1.2. Training Data = 10 samples

In decreasing the training samples to 10, the loss vs iteration graph in 3b looks vastly different. The clear signs of overfit model is visible because the validation losses shoot up and the start reducing. Perhaps upon increasing the number of iterations, we could eventually arrive at a solution where the validation and training losses are as low as with more number of training samples. The overfit behavior is more evident in the *W_star* vs *W_hat* where the prediction line is not in perfect overlap with the equation line.



(a) Training and Validation set (b) Losses (c) W_star vs W_hat

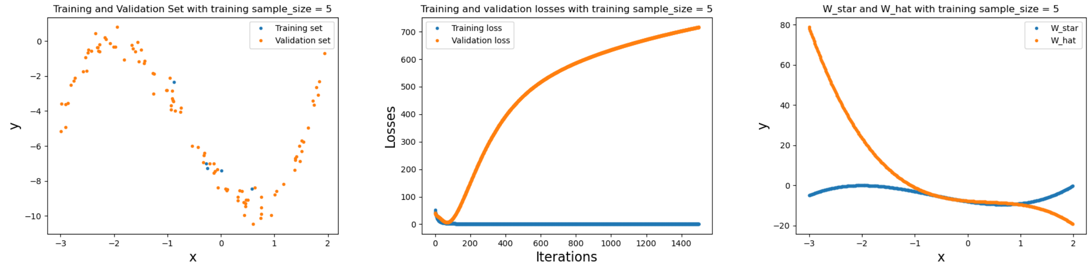
Figure 3: Training sample_size = 10

1.1.3. Training Data = 5 samples

In decreasing the training samples to 5, the loss vs iteration graph in 4b depicts awful performance from the model. The signs of an incorrect model are clearly visible with exponentially increasing validation losses while the training loss is ever reducing. This is evidence that because of low number of training samples, the model is overfit to those 5 data points. Hence the training losses are practically 0 but validation losses are extremely high. The incorrect model is more evident in the *W_star* vs *W_hat* where the prediction line is no where close to what the equation line is. Its evident that the model overfit to a completely different polynomial because the low training data and high iterations reinforced a bad generalisation in the model.

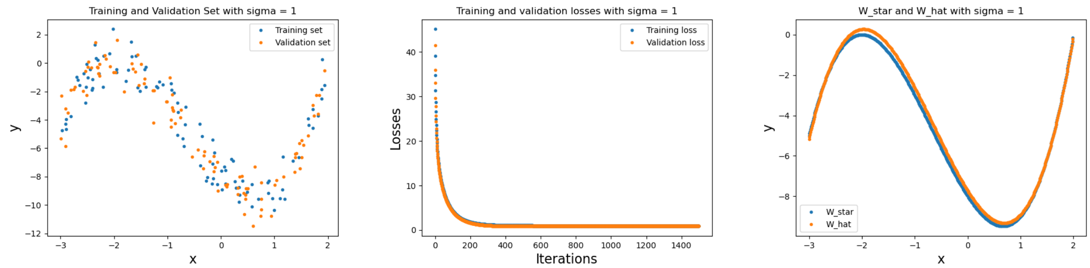
1.2. Increasing $sigma$

Increasing the sigma when generating the training and validation set introduces a lot more noise when generating the datasets. This would lead to an underfit model since the variation in data is too high for the neural network to learn. So the predicted w_hat is vastly different from the actual w_star and the variance keeps increase as we increase sigma. Another peculiar behaviour observed was that the training and validation loss have odd behaviors as we increase sigma where the validation error is lower than the training error. This could be because of the order of noise added is the same as the order of values of y.



(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 4: Training sample_size = 5



(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 5: sigma = 1

1.3. Increasing the order of the polynomial

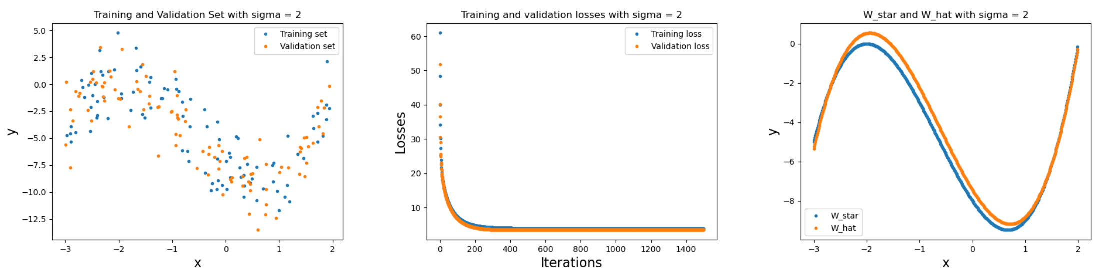
Upon increasing the degree of the polynomial from 3 to 4 with 10 data points for training, the combined effect of higher complexity and lower training data points is evident from the behavior observed in the graphs on figure: 9. The magnitude of the loss metrics for predicting a 4th degree polynomial with 10 training data points is an order higher than the case with 3rd. degree polynomial. Same high order characteristics can be observed in the the difference between graph of *w_star* and *w_hat*.

The original 3rd order polynomial:

$$p(x) = x^3 + 2x^2 - 4x - 8 \tag{2}$$

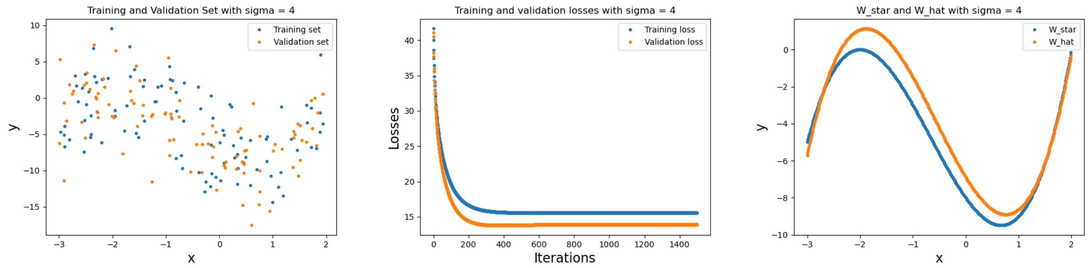
The 4th order polynomial:

$$p(x) = 2x^4 + x^3 + 2x^2 - 4x - 8 \tag{3}$$



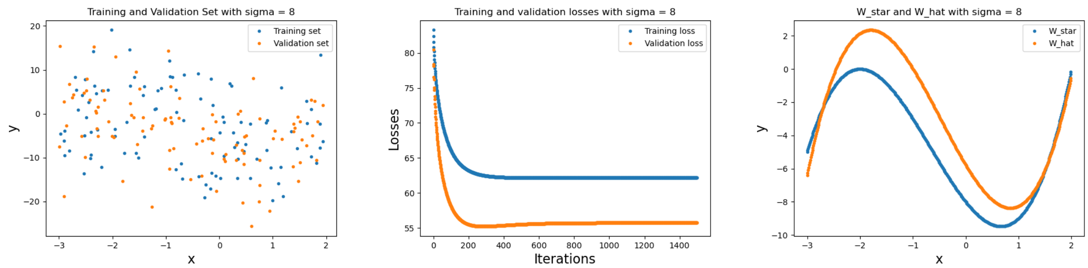
(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 6: sigma = 2



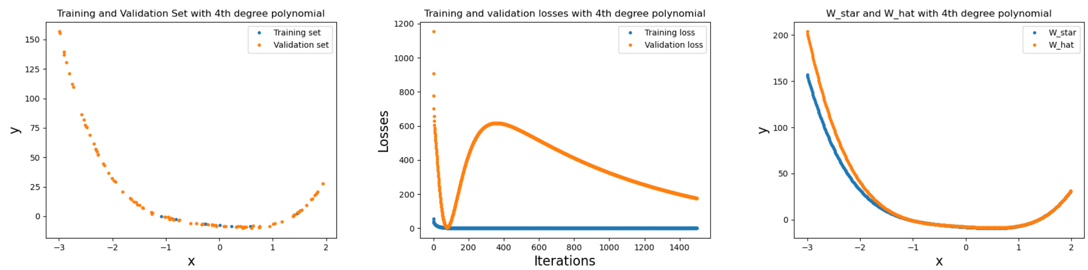
(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 7: sigma = 4



(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 8: sigma = 8



(a) Training and Validation set (b) Losses (c) W_star vs W_hat

Figure 9: 4th Degree polynomial