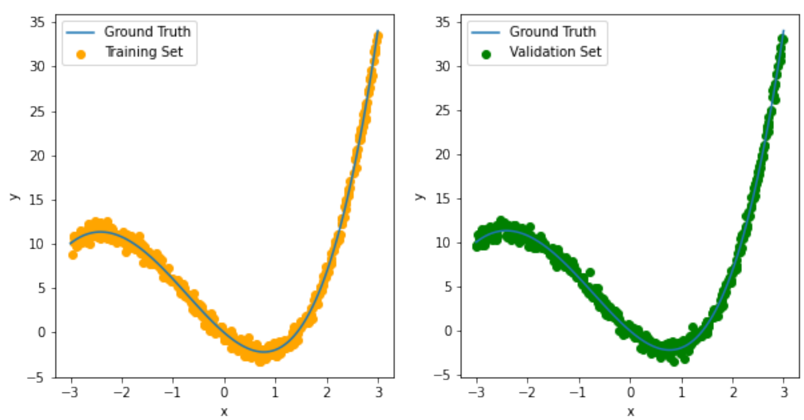
Deep Learning Lab: Linear Regression

Anthony Bugatto

22 October 2022

1 Dataset (1-4)

For points 1-3 we can see the class NoisyDataset has implemented a generalized noisy dataset functionality given a definition of the pointwise feature vector. In this case it has been defined with a generalized polynomial feature in the polynomialFeatures function. The results of this class are given below:



From the figure it can be seen that the data in the training and validation sets is about equally distributed along the polynomial.

2 Regression Model (5-11)

Using pytorch Liner with the SGD optimizer and MSE loss the RegressionModel class has implemented the generalized linear regression task given a dataset object of NoisyDataset.

We can see that the nn.torch.Linear function includes a bias term set to either true or false. Setting this term as true results in the regression equation taking the bias term form:

$$Y = Xw + b$$

instead of the matrix form, in which we set one feature to be 1 and let one of our weights take the form of the bias:

$$y = X_{bias} w$$

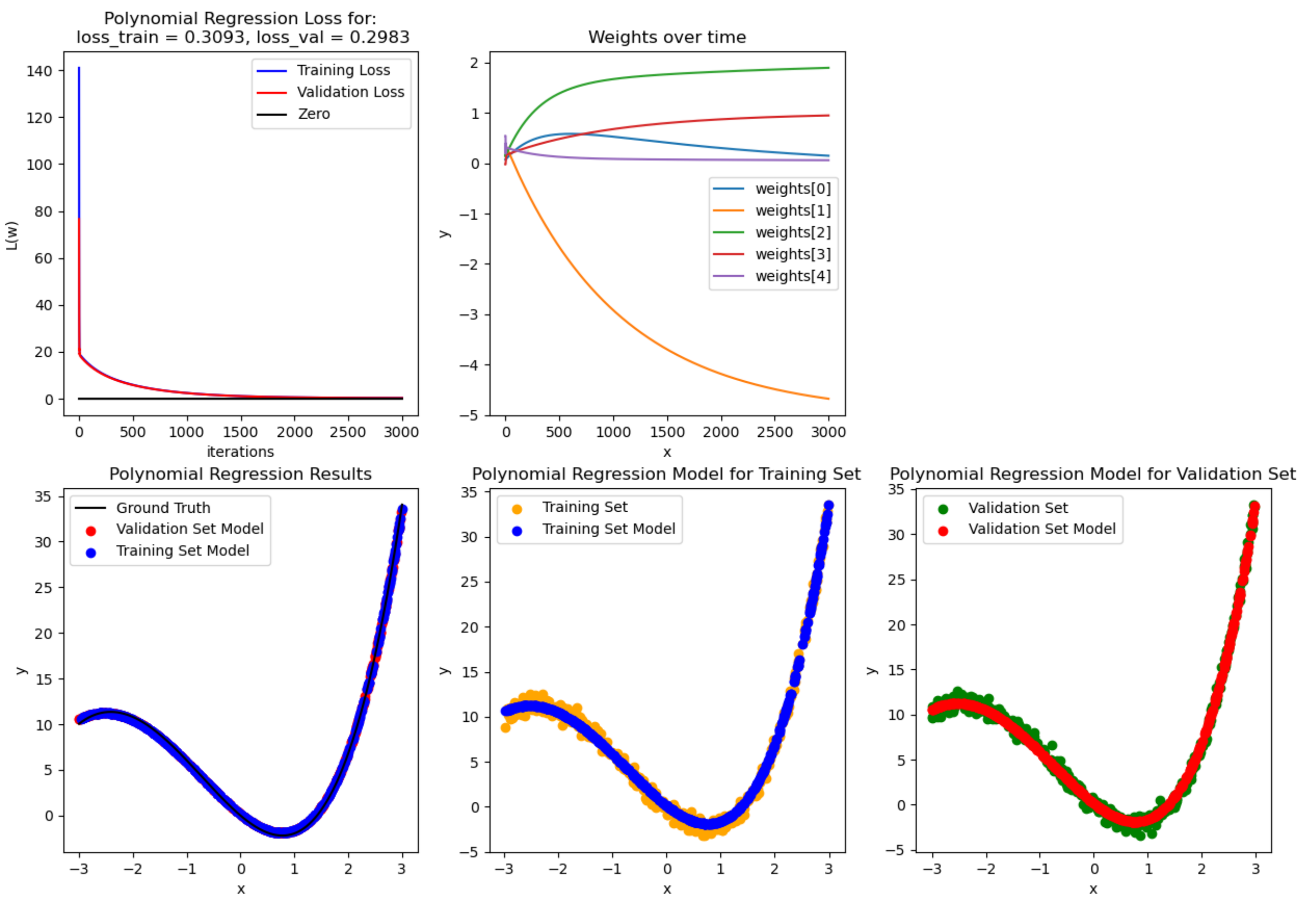
After some trial and error with the learning rate (alpha) and iterations (maxIts) it was discovered that the linear model diverged in training unless the alpha was roughly equal or lower than .001. The training was found to converge to roughly .3 at a little under 3000 iterations.

In the figure on page 3 we can see that there are 5 plots. The first plot shows the training and validation loss as a function of SGD iterations. The second shows the weight coefficients evolving as a function of iterations. The third plot shows the validation set model and the training set model overlayed with the ground truth. we can see that the model does a very good job of learning the polynomial in both training and validation sets. The fourth, and fifth plots show the training and validation sets along with the their corresponding models overlayed.

From experiments when the training set was reduced to 10 the model did a very bad job fitting to the data and didn't even look like the input polynomial.

3 Questions

1. Overfitting is caused by high variance and low bias in the model. This means that the model cannot generalize to new data and is too similar to the training set.
2. Overfitting can be fixed by lowering the number of features in the feature space.


Polynomial Regression Loss for:
loss_train = 0.3093, loss_val = 0.2983
Training Loss
Validation Loss
Zero
L(w)
iterations
Weights over time
weights[0]
weights[1]
weights[2]
weights[3]
weights[4]
y
x
Polynomial Regression Results
Ground Truth
Validation Set Model
Training Set Model
y
x
Polynomial Regression Model for Training Set
Training Set
Training Set Model
y
x
Polynomial Regression Model for Validation Set
Validation Set
Validation Set Model
y
x