Emre Zeytinoğlu 28190

CS412 HW2

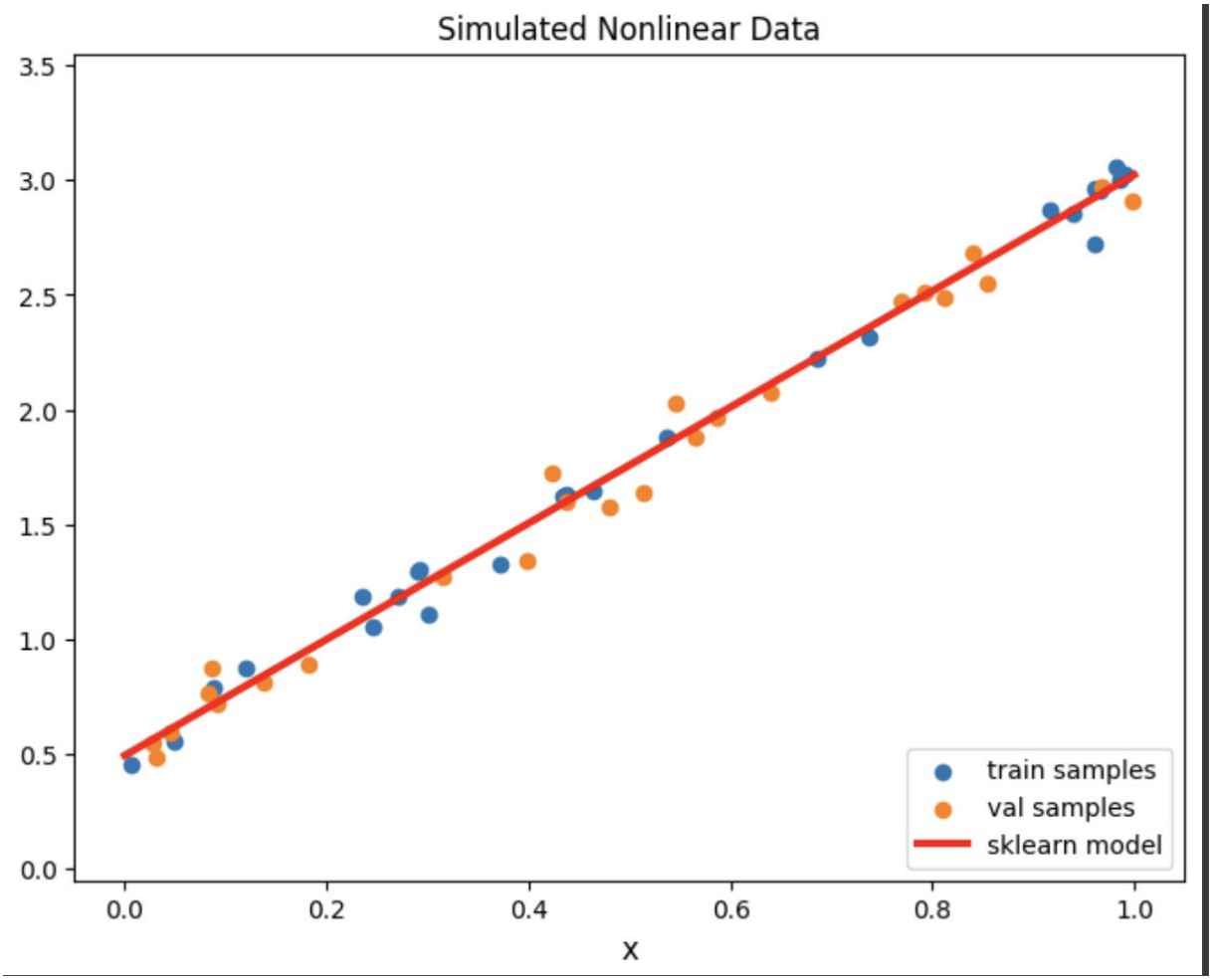
Collab Link:
https://colab.research.google.com/drive/1mTzKGvnS65tWGRZzX29iViBUvxntqCYs?usp=sharing

PART1

1a)

The reported mean squared error for the Sklearn model is 0.00795462682779033, informing us that the average squared difference between the target variable values on the validation set that were predicted and those that were actually observed was quite minimal.

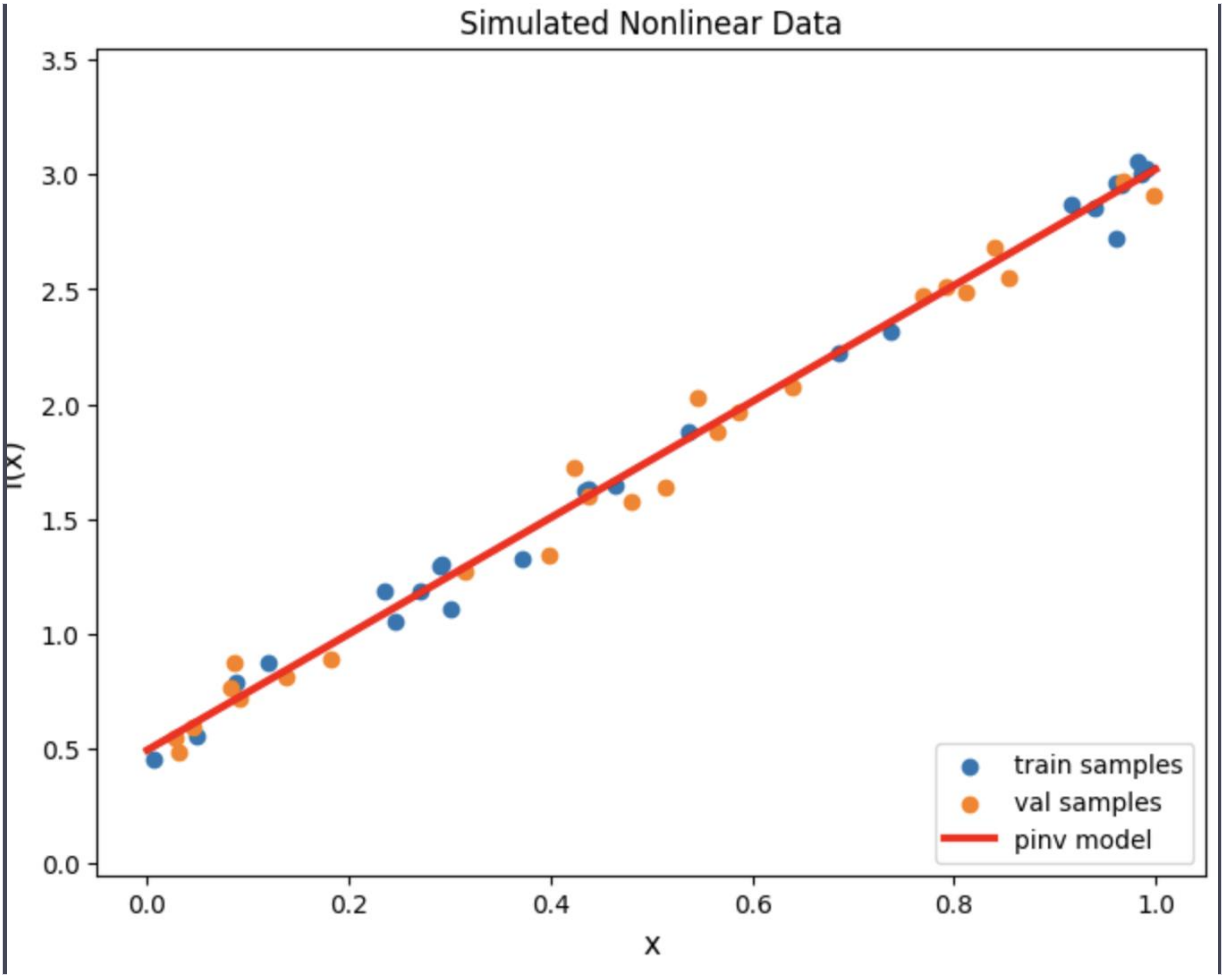
As a result, it appears that the model is successfully identifying the fundamental patterns in the data. Using a scatter plot and the regression line, Dataset 1 successfully fits a linear regression model to the data using sklearn.



1b)

The mean squared error (MSE) on the validation set as a result of using the pseudo-inverse technique on Dataset 1 was 0.0079.

The regression line discovered by our implementation was then plotted onto the scatter plot of Dataset 1 to create a visual representation of the findings. The regression line created using the pseudo-inverse approach is represented by the red line, and it adequately matches the data.



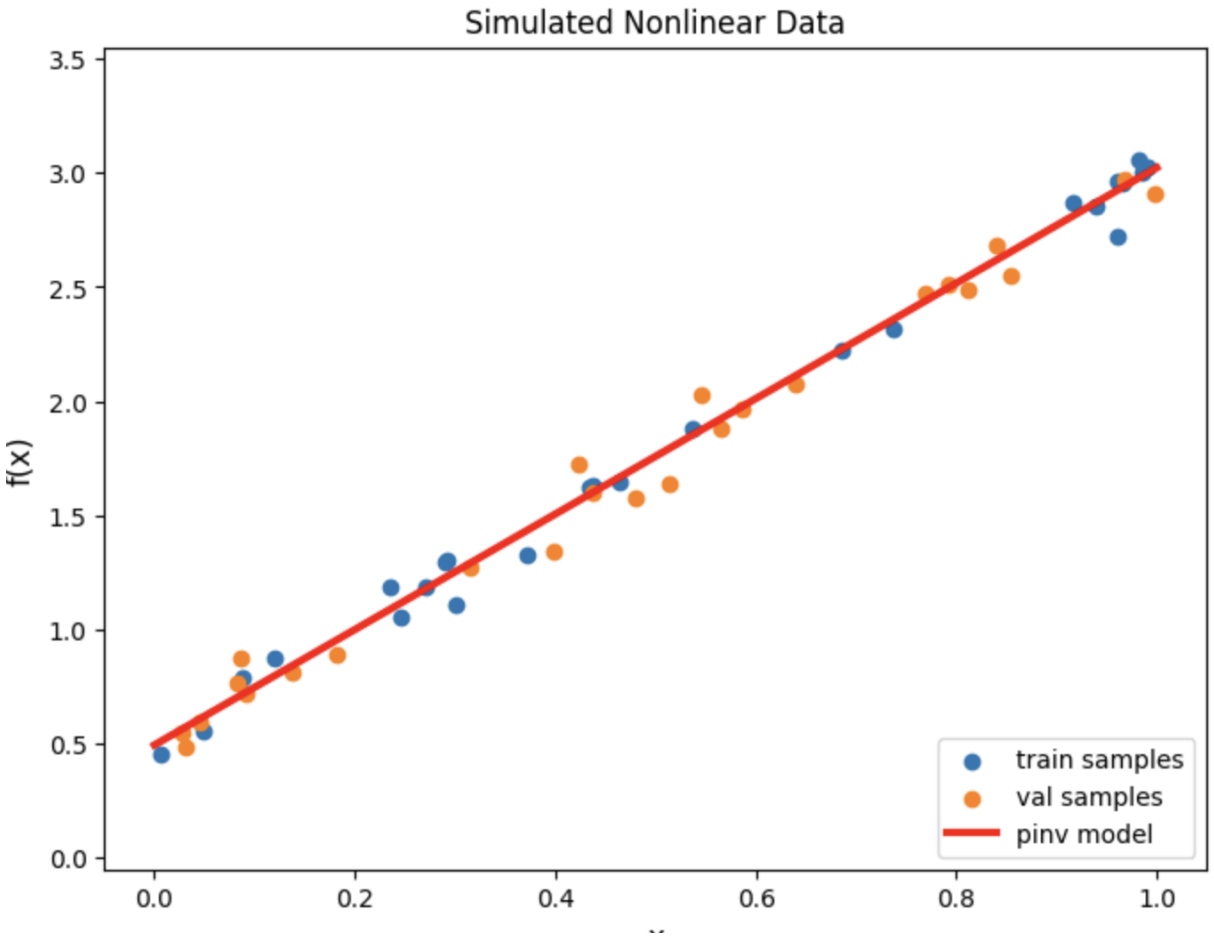
1c)

In this part, we implemented the gradient descent optimization algorithm to find regression coefficients in an iterative manner. We printed the MSE error at certain steps to monitor the convergence of the algorithm.

MSE error at step 1: 4.1147
MSE error at step 100: 0.0625
MSE error at step 200: 0.0150
MSE error at step 300: 0.0069
MSE error at step 400: 0.0056
MSE error at step 500: 0.0053
MSE error at step 600: 0.0053
MSE error at step 700: 0.0053
MSE error at step 800: 0.0053
MSE error at step 900: 0.0053
MSE error at step 1000: 0.0053

After finding the regression coefficients, we drew the regression line onto the scatter plot of Dataset 1 by making a scatter plot of train and

validation samples. For drawing the regression line, we first constructed the extended version of x_grid, just like we did to train and validation data matrices, and then used the regression coefficients to find the model's predictions.



QUESTION

In Part 1, you should comment on whether the gradient descent solution is the same (or very close) to solutions obtained for Part1.a and b. If not, add a line of explanation as to why you think it is not.

The gradient solution is the same to solutions obtained for part a and b. Because plots and lines are same.

PART 2

2a)

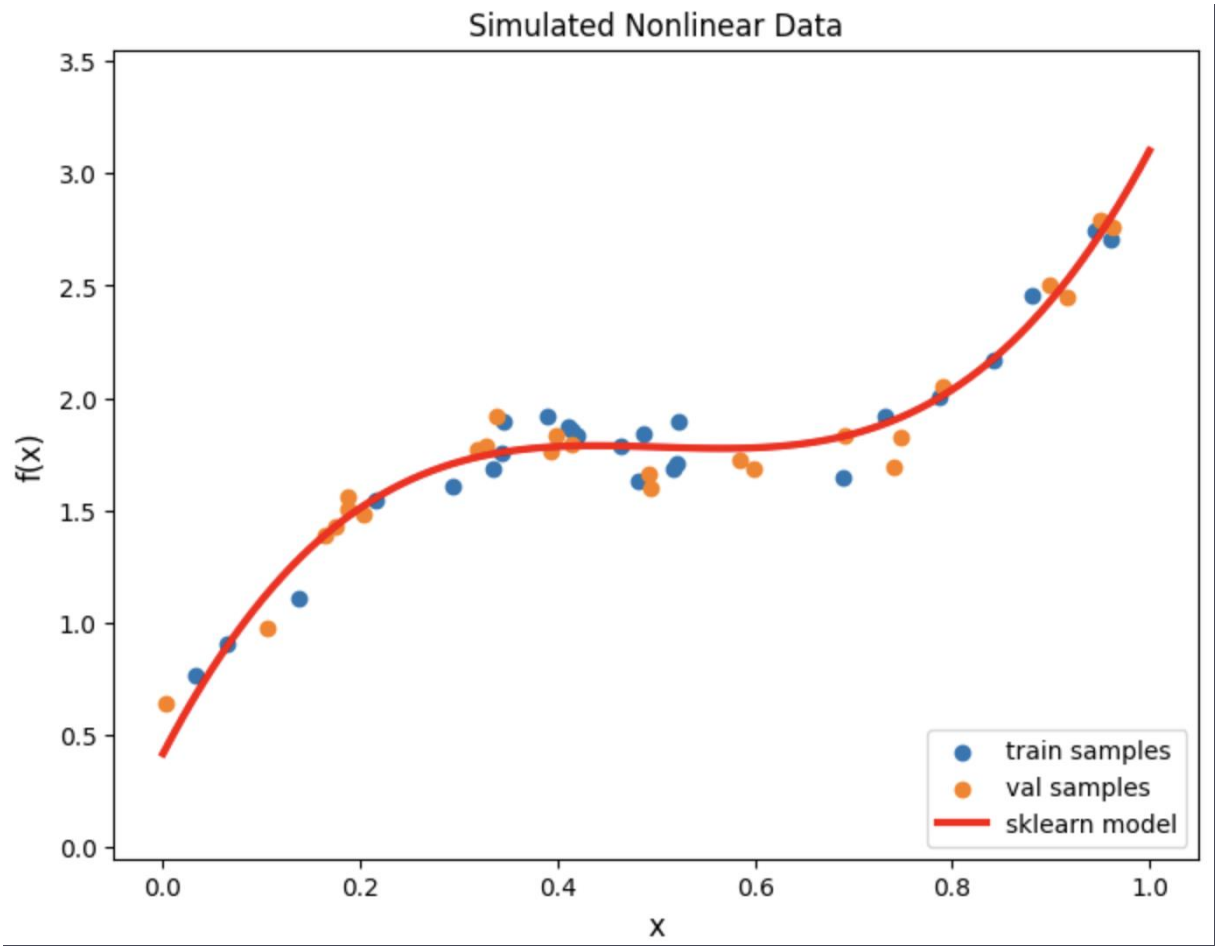
1.

In this part, we performed polynomial regression using the sklearn library on the input data matrix X with polynomial features of degrees 1, 3, 5, and 7. We made predictions on the validation set and evaluated the model's

performance on the validation set using the mean squared error (MSE) metric. The best model was the one with the lowest MSE, which was achieved with a polynomial degree of 3.

1 ——->MSE of sklearn model: 0.05830097537046252
3 ——->MSE of sklearn model: 0.009619587973839388
5 ——->MSE of sklearn model: 0.010278583809366295
7 ——->MSE of sklearn model: 0.010116792111851689

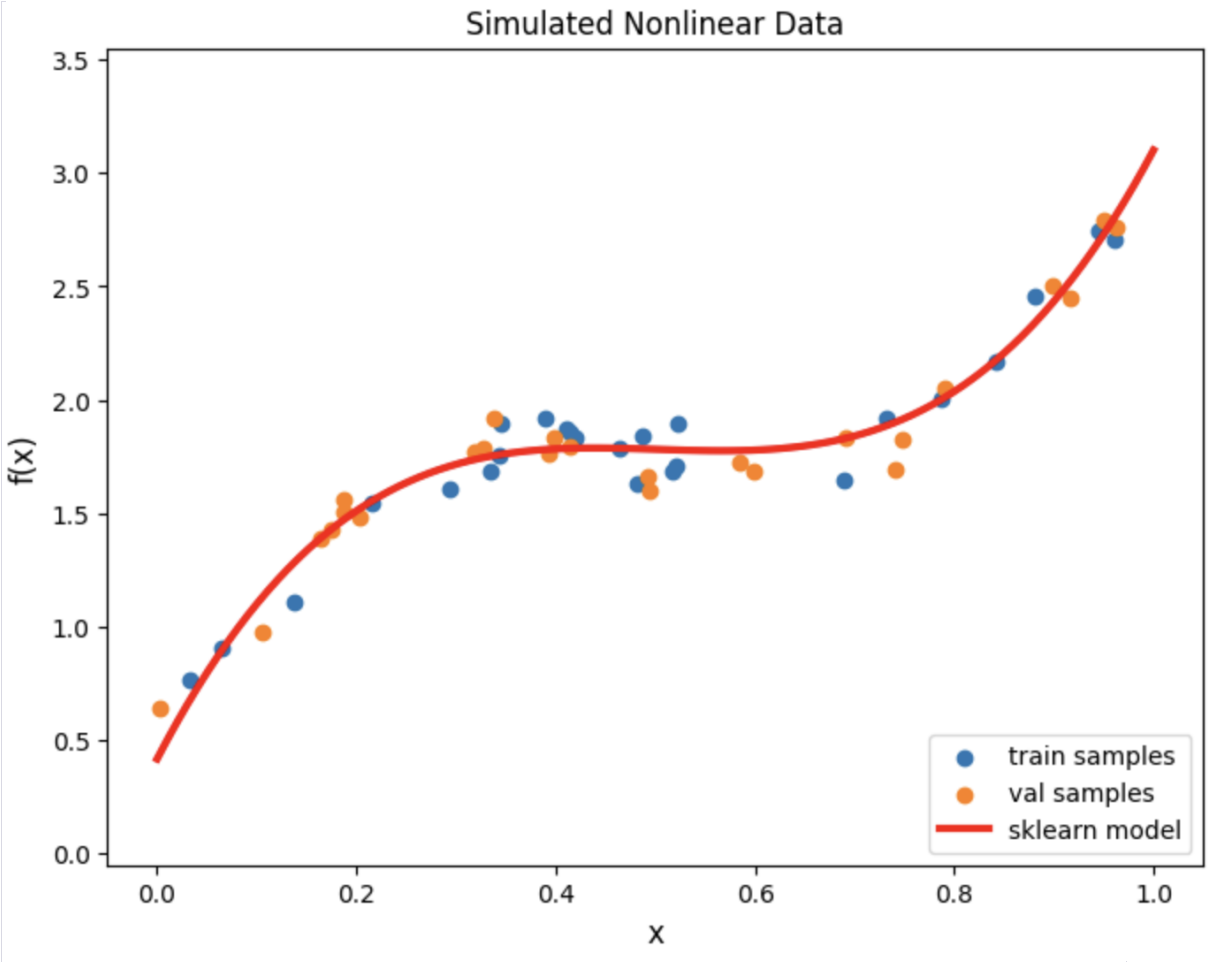
After obtaining the best model, we made a scatter plot of the data using both train and validation sets and plotted the predicted y values of the model using the predict() function of the LinearRegression object. Red line allowed us to visually inspect how well the polynomial regression model fits the data.



2b)

In this section, we utilized the np.concatenate() method and added columns of powers of the original data vector to generate the data matrices containing polynomial features for training and validation.

Then, using the train and val sets of data and the plot_samples() function, we created a scatter plot of the data and projected the model's predictions on it using the regression coefficients we discovered. The validation set's MSE for the model was 0.0096, a very low value that shows the model fits the data well.



QUESTION

In Part 2, comment on the effect of the degree parameter. What happens when it is chosen too small or too big? What do you think is the optimal degree value, and why? Discuss from the perspective of underfitting/overfitting.

We observed that the training error (MSE) lowers and the model's capacity to fit the training data increases when the degree of the polynomial is increased.

Though the validation error (MSE) initially declines as the degree rises, it soon starts to rise again, as seen by the model's evaluation on the validation set. Because the model grows too sophisticated and begins to fit the noise in the training data when the degree is increased excessively, this might lead to poor generalization to the

unseen validation data. Overfitting is the process of a model failing to generalize well to new data because it is too complex for the training set of data.

However, if we pick a degree that is too little, the model might not be complex enough to capture the underlying patterns in the data, which could lead to significant training and validation mistakes. Underfitting occurs when a model is too simplistic to adequately represent the complexity of the underlying function generating the data.