ENSC 413
Assignment #2
Model Order Selection
Cyrus WaChong
301306459
Spring 2020

## Introduction:

This report will explain the methodology used to find the order of the function that best fits the noisy data given. The code was run multiple times to ensure that the random choice of weights does not affect the final value of k chosen, but only one run will be shown. This report will also include all the figures and methods used to reach our final choice of k.

## Results:

The first step was to plot all the resulting functions (k ranging from 0 to 10) to see which functions would be further examined, making the testing more efficient. Immediately a few functions were excluded, such as order 0 and 1, as they were incredibly inaccurate, and the data points supplied were obviously not of these orders. The plot of the noisy data supplied can be seen in figure 1.

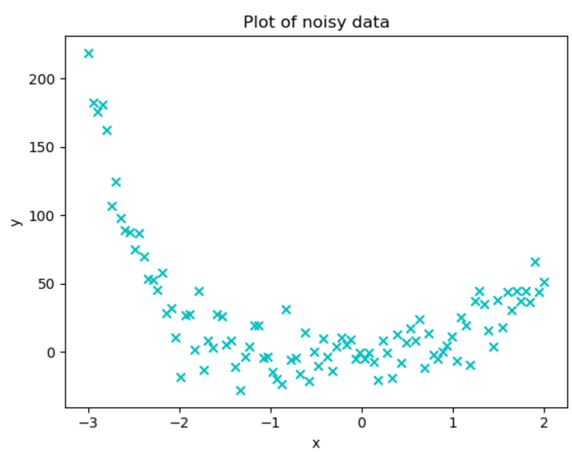



Figure 1: Noisy data plot

The points seem to approach from the left with a negative slope to some minimum value, once this minimum has been reached the data points seem to have a positive slope, to which our dataset terminates. This is a common characteristic of even order polynomials, but since the right-hand side is not obvious, we will still test all values. We see that it is also flatter than you would typically see with a high valued polynomial, so the assumption is that it is of a higher value.

To test every value of k, the stochastic gradient descent method is chosen. The method is very similar to the polynomial example demonstrated in class, but we must do it for 11 different values of k. The data set was split into 60% training and 40% validation. The batch size and patience parameters were chosen to be the same as the example shown in class, which are 10 and 15 respectively.

To decide which of the values of k is the correct value, once the functions are plotted on the noise data plot, we will then plot the error functions of each of the functions to ultimately choose which value of k is correct. The lower the error value, the better the curve fit to the dataset. This method will be implemented 4 times to verify the result, and make sure it is consistent for many runs. The best fit polynomials can be seen below in Figure 2.

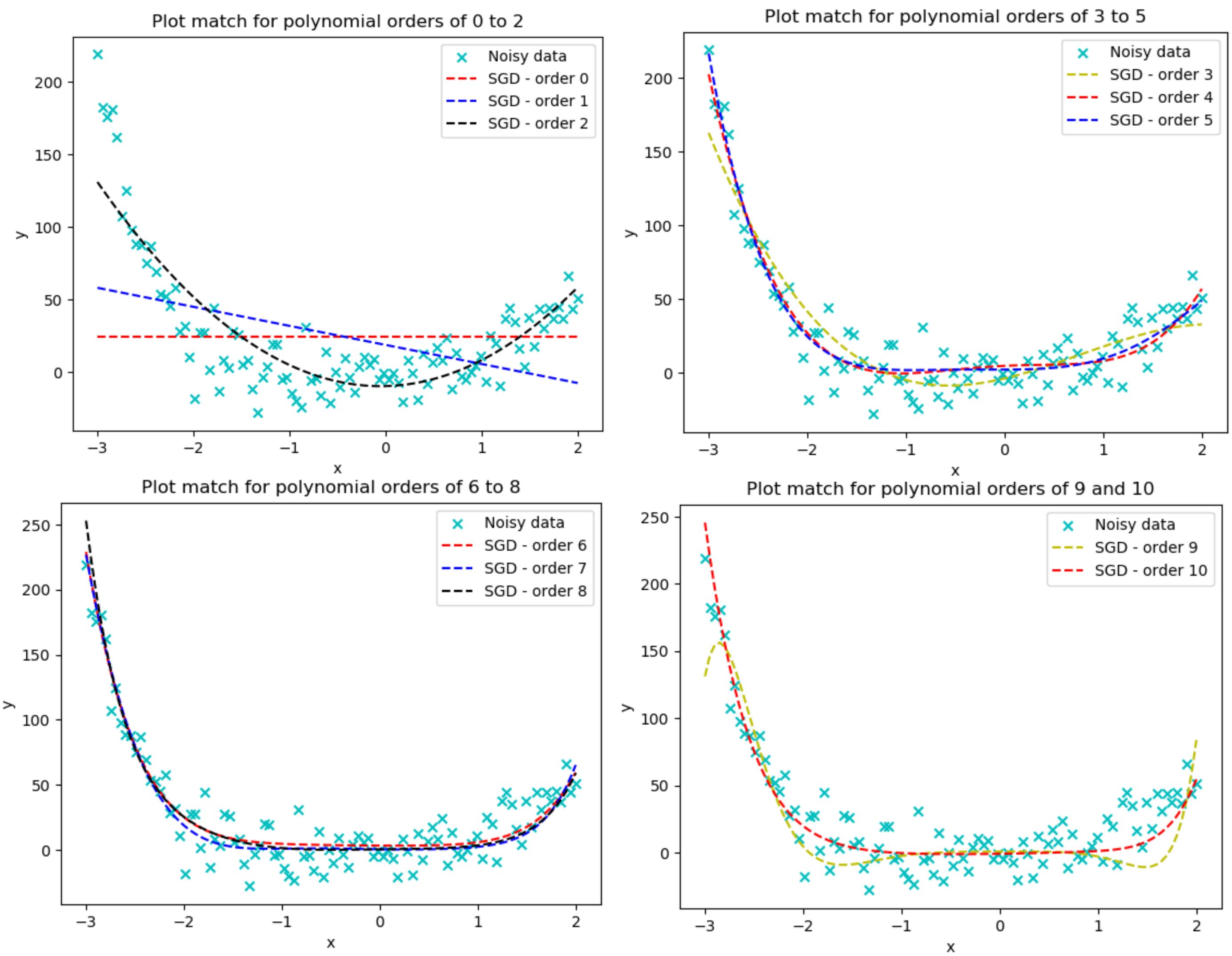
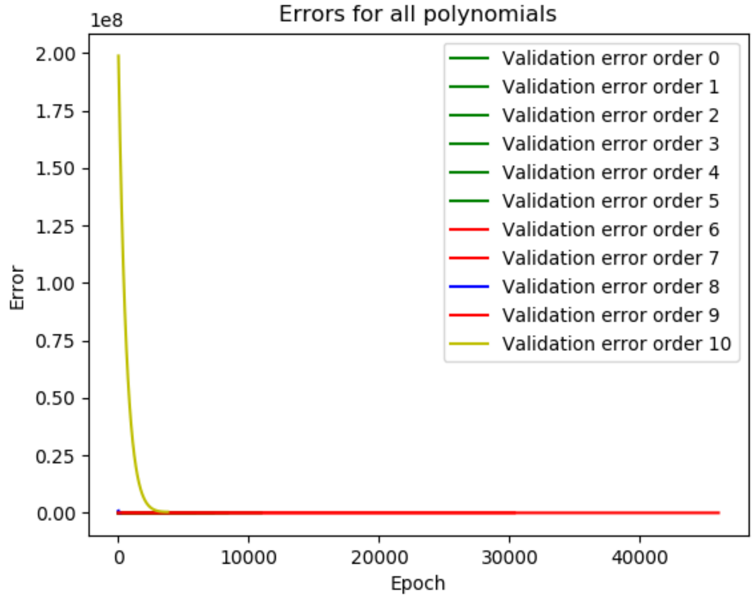



Figure 2: One group of results for each polynomial order using stochastic gradient descent

The stochastic gradient descent is used on polynomials ranging from 0 to 10. The training was limited to 50000 epochs, with a patience value of 15 and batch size of 10. As expected, the order 0 and 1 polynomials do not fit the curve well at all. Visually, it is hard to distinguish the perfomance of any of the other polynomials, so this is where the error function must be analyzed.

In Figure 3 all the error plots were compared, which was quite difficult because the ranges of ech one were very different.



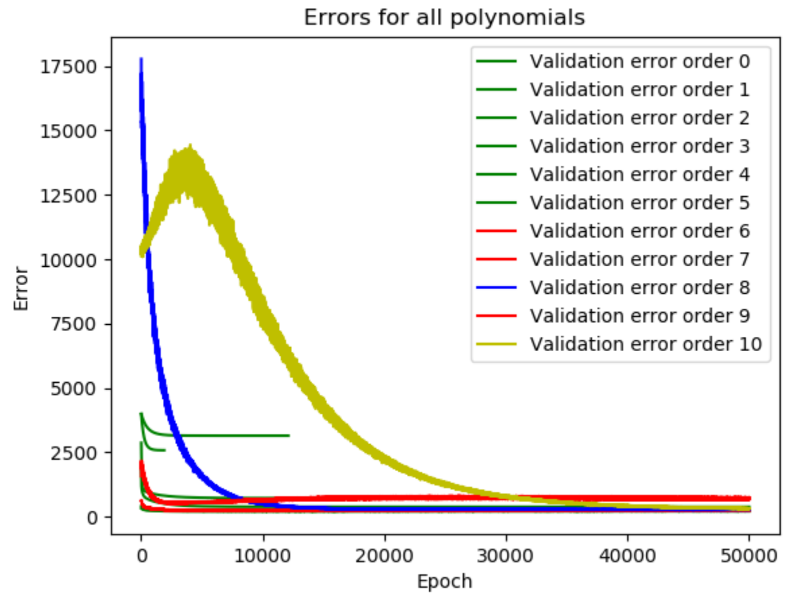




Figure 3: Loss function of all the polynomials with different learning rates

The two plots in Figure 3 are plotted with different learning rates. The range of epoch shown was continuously limited until we could reach the 3 most similar loss functions which pertained to polynomials of orders 4, 5 and 6. The results can be seen in Figure 4, with 4 tests shown to prove the polynomial of order 5 best fit the data set. The learning rates used can also be seen in Table 1.

| Polynomial order | Learning rate |
|---|---|
| 0 | 1e-5 |
| 1 | 1e-5 |
| 2 | 1e-5 |
| 3 | 1e-5 |
| 4 | 5e-6 |
| 5 | 1e-6 |
| 6 | 3e-7 |
| 7 | 1e-8 |
| 8 | 5e-9 |
| 9 | 1e-9 |
| 10 | 1e-10 |

Table 1: Learning rates used find value of k

As the order of the polynomial increased, the learning rate had to be decrease or it the error would diverge, and produce incomprehensible results. This was found by trial and error, but is understandable as the higher the order of the polynomial, the more of an affect the constant attached to the variable has.

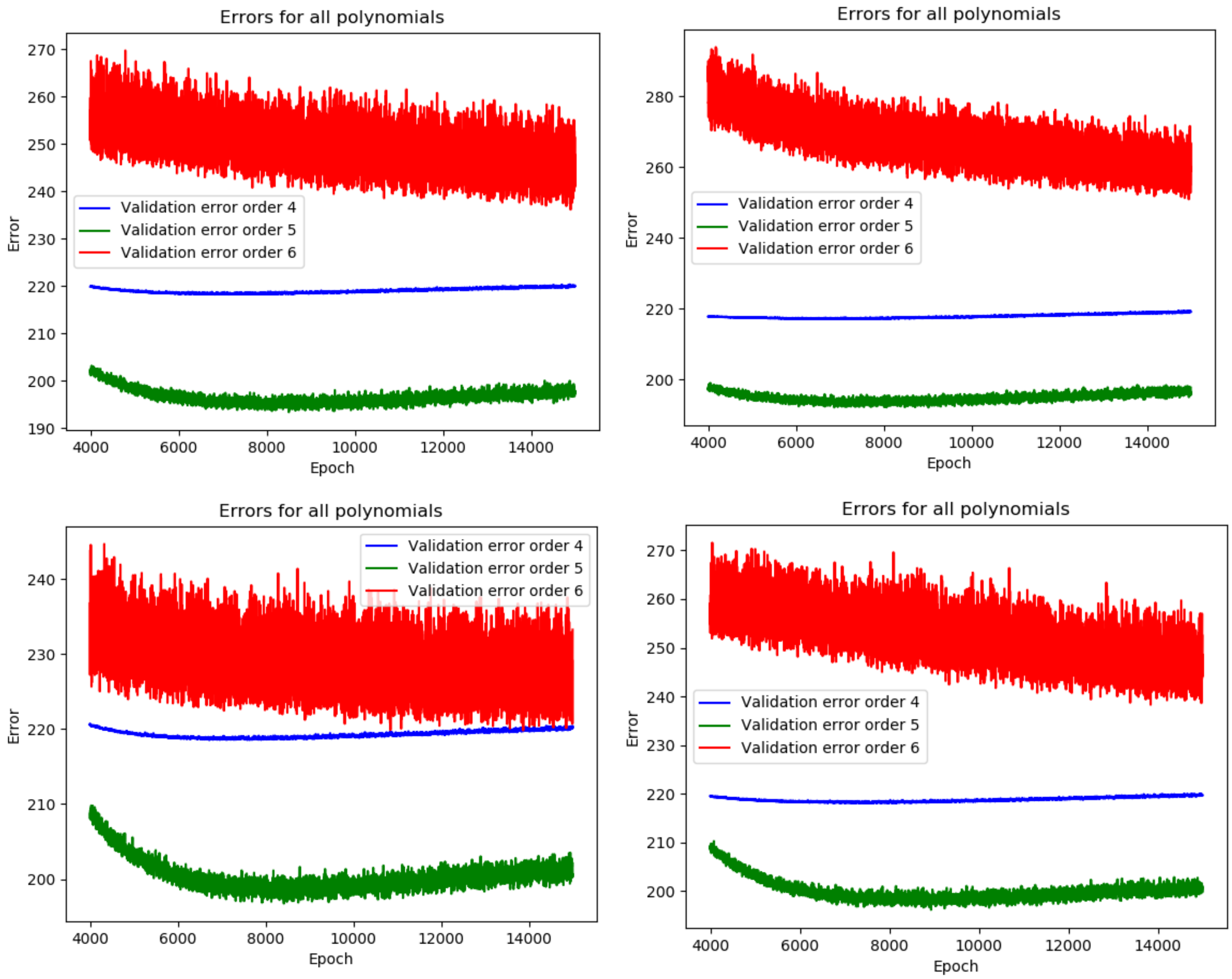



Figure 4: Error functions of polynomials of order 4, 5 and 6

## Conclusion:

As was explained above, the 3 orders that had the lowest error values when training finished were orders 4, 5 and 6. Once these were plotted versus eachother, it is very evident that the polynomial of order 5 best fits the noise data set, as it has a drastically lower error throughout the entire time of training. **This leads to the conclusion that the value of k must be 5,** since the plot on the noise data set fits well, while the loss/error function is the lowest of all 11 possible orders.