ELL409 REPORT

ASSIGNMENT 1

SARANSH AGARWAL 2019MT60763

Part 1 a)

Optimization using Moore-Penrose Pseudoinverse

For 100 Data points

The model is trained using 80 datapoint's and tested remaining 20 datapoint's

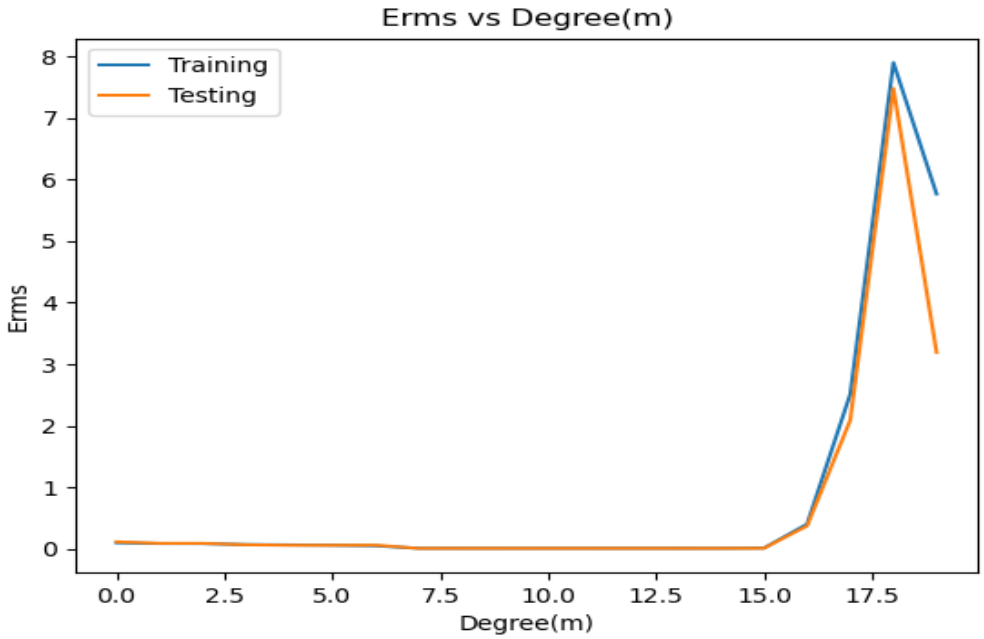

Figure 1: Root mean square error (Erms) vs degree (order) is plotted for testing and training data

```
[0.00000000e+00 9.73077023e-02 1.13904608e-01]
[1.00000000e+00 8.70219929e-02 8.91965481e-02]
[2.00000000e+00 8.57605026e-02 8.87701743e-02]
[3.00000000e+00 6.89432439e-02 6.63685686e-02]
[4.00000000e+00 6.42083806e-02 5.86701996e-02]
[5.00000000e+00 5.90603659e-02 5.76480060e-02]
[6.00000000e+00 4.93652094e-02 5.96645018e-02]
[7.00000000e+00 8.66373735e-03 7.69193950e-03]
[8.00000000e+00 8.62690165e-03 7.82592979e-03]
[9.00000000e+00 8.62661002e-03 7.80595813e-03]
[1.00000000e+01 8.61356666e-03 8.03947865e-03]
[1.10000000e+01 8.57700787e-03 8.05505312e-03]
[1.20000000e+01 8.57683842e-03 8.10108009e-03]
[1.30000000e+01 8.51567007e-03 8.01646255e-03]
[1.40000000e+01 8.51492841e-03 8.24973643e-03]
[1.50000000e+01 1.25209111e-02 9.35803960e-03]
[1.60000000e+01 4.02712376e-01 3.72247736e-01]
[1.70000000e+01 2.51331375e+00 2.09079782e+00]
[1.80000000e+01 7.89993716e+00 7.47925191e+00]
[1.90000000e+01 5.76393032e+00 3.19103578e+00]]
```

Figure 2: values of Root mean square error Erms for testing data(3-col) and training data(2-col)

Conclusion: From the above plot of Root mean square error (Erms) vs order we can conclude that optimal order of polynomial will we 15 as after that curve starts overfitting. From the values of Root mean square error Erms for testing data we can see that at m=15 it attains minima and after that it starts increasing so 15 will be correct choice for m.

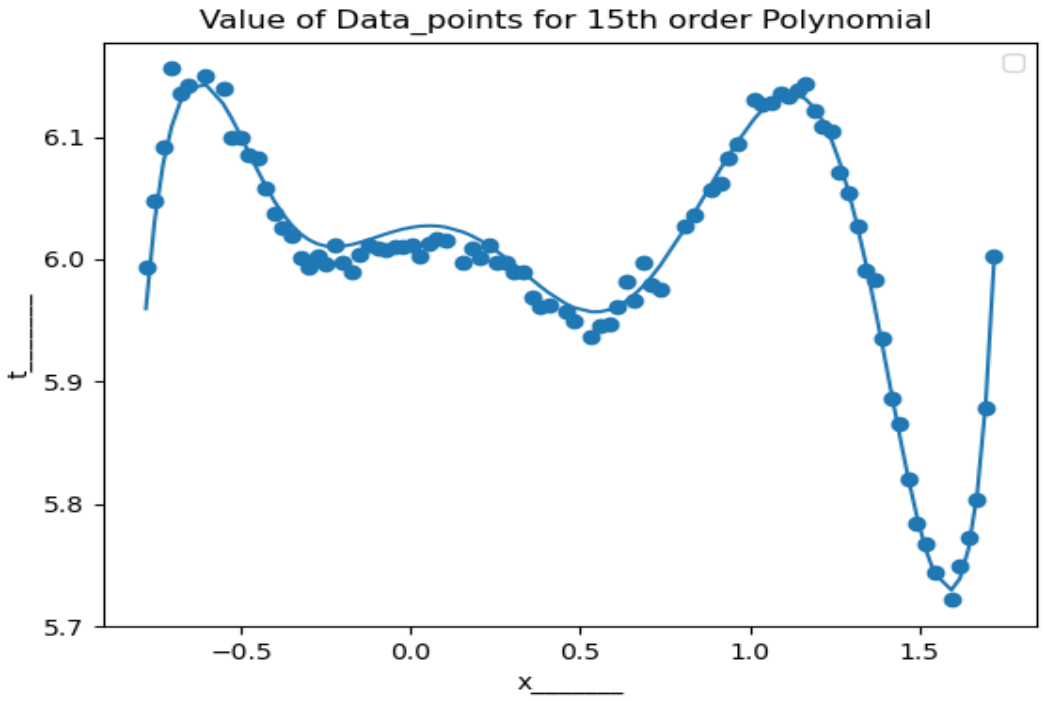

Figure 3: The curve for m =15 for datapoints (x, t)

Conclusions:

1)The 15-order polynomial obtained:

Y (x, w) = 6.02597837 + (0.04958715) x^1 + (-0.38316701) x^2 +(-0.8467778) x^3+ (1.64440306) x^4+(-1.2763578) x^5+(4.0597695) x^6+ (6.71398831) x^7+ (-25.13371282) x^8 +(4.14651425) x^9+ (35.25075248) x^{10}+(-26.45172552) x^{11} +(-10.65425787) x^{12} +(20.76354201) x^{13}+ (-9.18061533) x^{14}+(1.38225038) x^{15}

2)**Training_ error at m =15:** 1.25209e-02

3)**Testing error at m =15**: 9.358039e-03

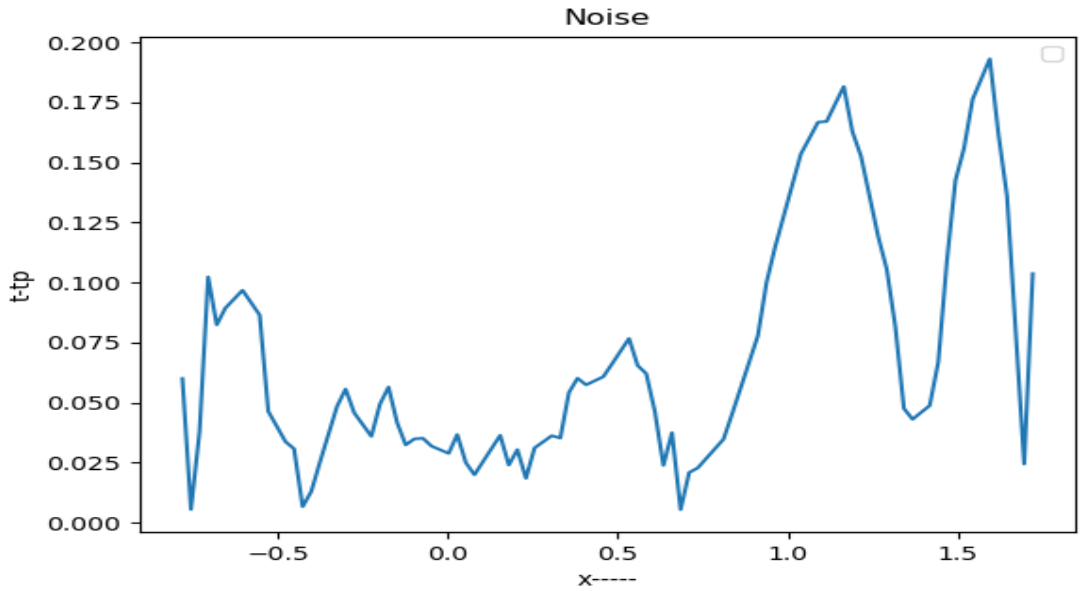

Figure 4: Plot for Noise

Conclusion:

1) **Mean** = 0.00056692 (This is calculated mean of the Gaussian distribution) expected values was 0. Calculated value is close to expected in our case.

2) **Variance** = 0.000107281(This is calculated variance which was unknown)

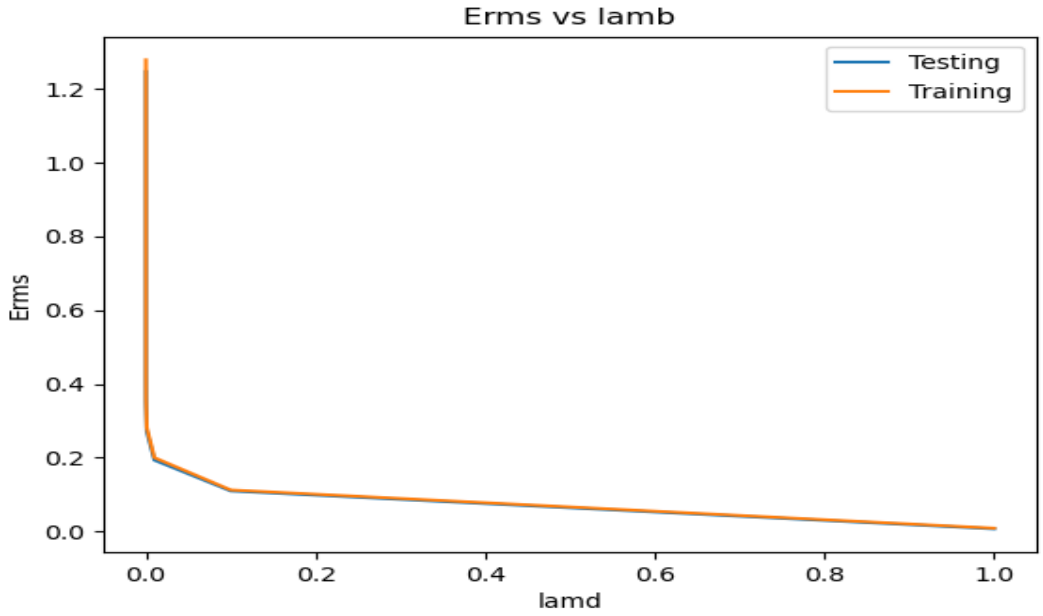

Figure 5: Root mean square error Vs lambda for testing and training data

For 20 Data points

The model is trained using 10 datapoint's and tested remaining 10 datapoint's

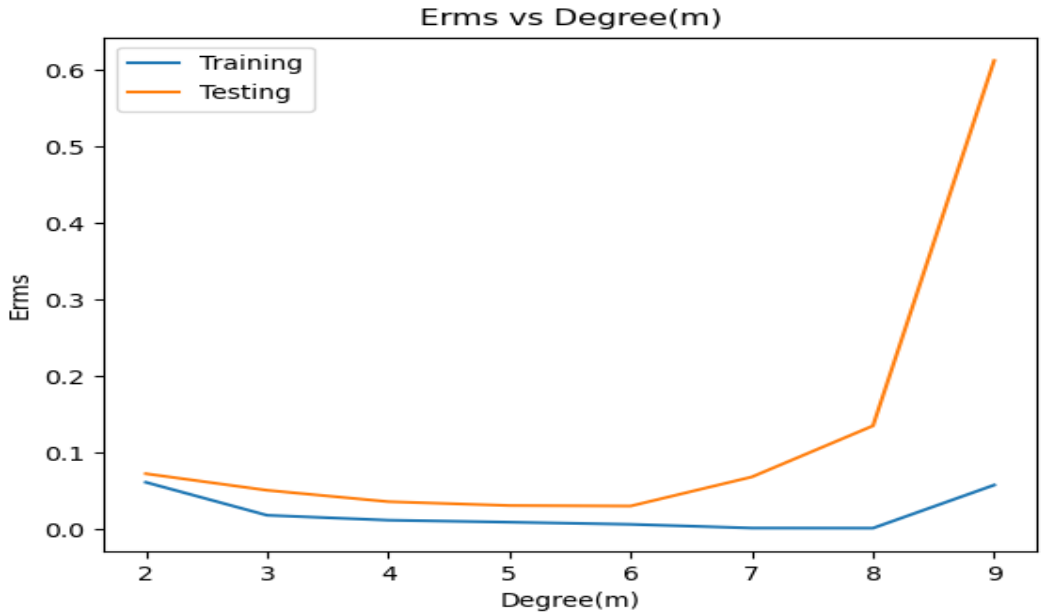

Figure 6: Root mean square error (Erms) vs degree (order) is plotted for testing and training data

```
[2.00000000e+00 6.16808647e-02 7.29020656e-02]
[3.00000000e+00 1.85951046e-02 5.12618738e-02]
[4.00000000e+00 1.21911546e-02 3.63766249e-02]
[5.00000000e+00 9.58174045e-03 3.13236869e-02]
[6.00000000e+00 6.76934542e-03 3.06900951e-02]
[7.00000000e+00 1.92833852e-03 6.86770786e-02]
[8.00000000e+00 1.80577020e-03 1.35481166e-01]
[9.00000000e+00 5.80930333e-02 6.12631869e-01]]
```

Figure 7: values of Root mean square error (Erms) for testing(3-col) and training data(2-col)

Conclusion: From the plot of Root mean square error (Erms) vs order we can conclude that optimal order of polynomial will we 6 as after that curve starts overfitting. From the values of Erms for testing data we can see that at m=6 it attains minima and after that it starts increasing so 6 will be correct choice for m.

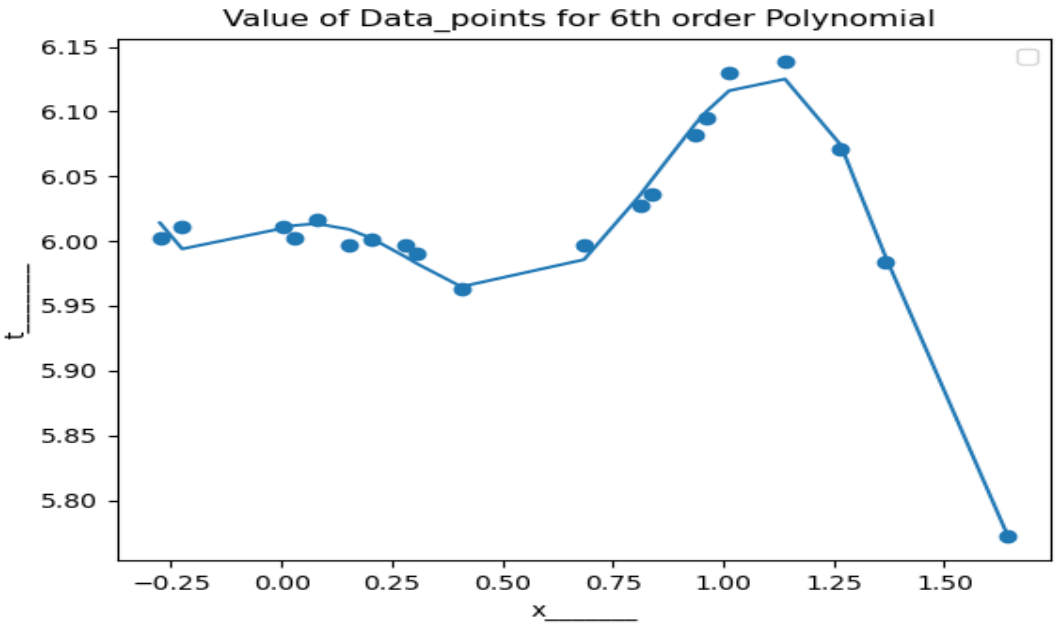

Figure 8: The curve for m=6 for datapoints (x, t)

Conclusions:

1)**The 6-order polynomial obtained:**

Y (x, w) = 5.99188564 + (0.25164596) x^1 + (-0.29827922) x^2 +(-4.7816597) x^3 + (12.14124147) x^4 +(-9.65365015) x^5+(2.46471237) x^6

2)**Training_ error at m = 6**: 6.76934e-03

3)**Testing error at m = 6**: = 3.069009e-02

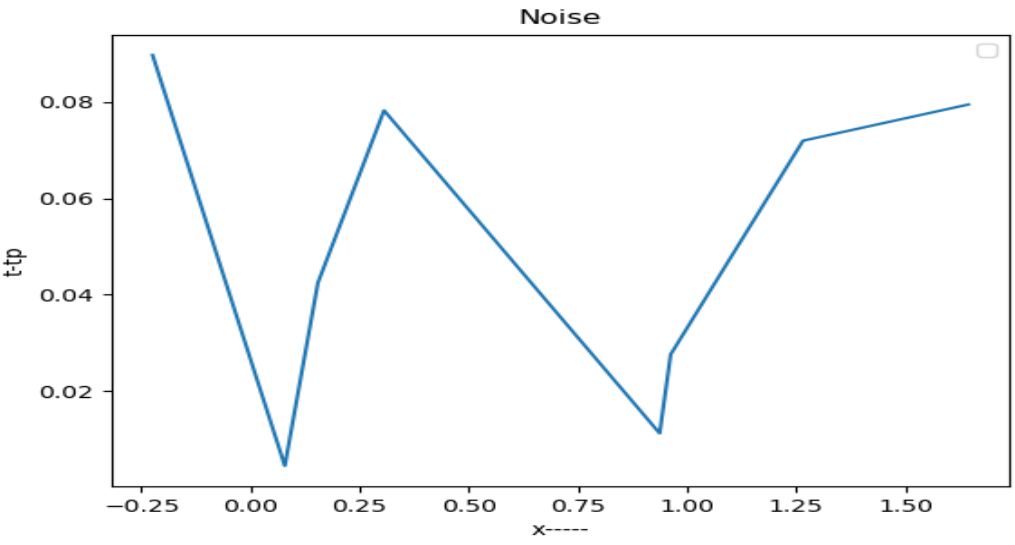

Figure 9: Plot for Noise

Conclusion:

1) **Mean** = 8.07160386e-05(This is calculated mean of the Gaussian distribution) expected values was 0. Calculated value is closer to expected in our case.

2) **Variance** = 8.071603865e-05(This is calculated variance which was unknown)

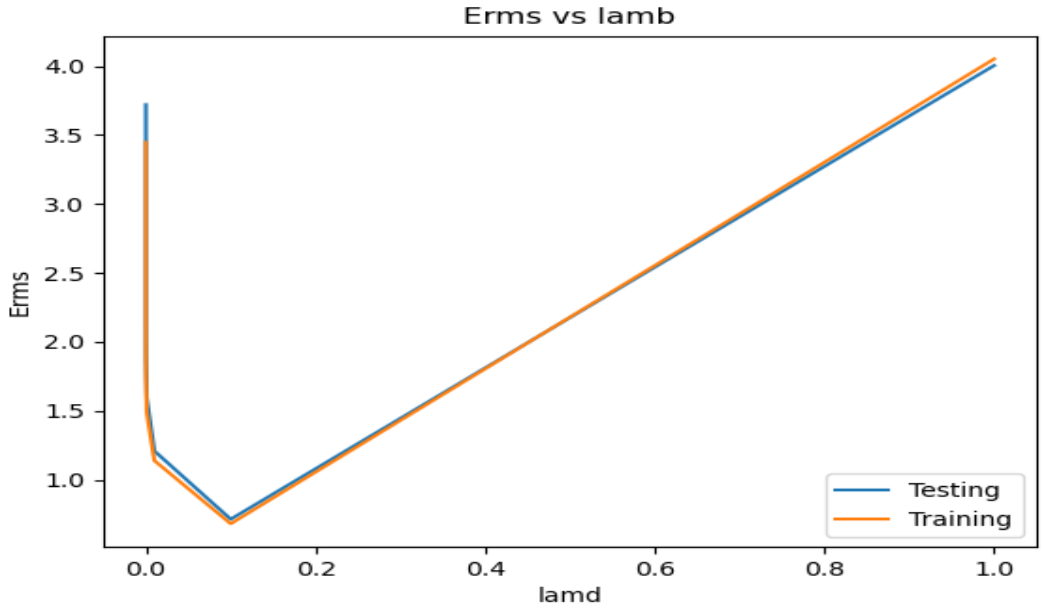

Figure 10: Root mean square error Vs lambda for testing and training data

FINAL CONCLUSION:

For 100 data points, I have considered 80 data-points as training set and remaining 20 data-points as testing set. For the other 20 points given I have considered 10 data-points as training set and remaining 10 data-points as testing set.

When we are using less points, it can be observed from the plot that testing and training error is growing exponentially. The reason is that since, the training set is small we are not able to get a good fit model. Whereas when we consider large data set, we can observe that testing and training error does not grow steadily which indicates that model is close to good fit.

Also, when we are using less points for training set, we can observe that the noise variation is not approaching gaussian because the model is insufficient to get better results whereas when we are using large points for training set the noise is approaching to gaussian.

Finally, we get a model close to good fit which is a 15-order polynomial using 80 training points whereas we get a 6-order polynomial when we use 10 training points.

Optimization using Stochastic Gradient Descent (SGD)

For 100 Data points

The model is trained using 80 datapoint's and tested remaining 20 datapoint's

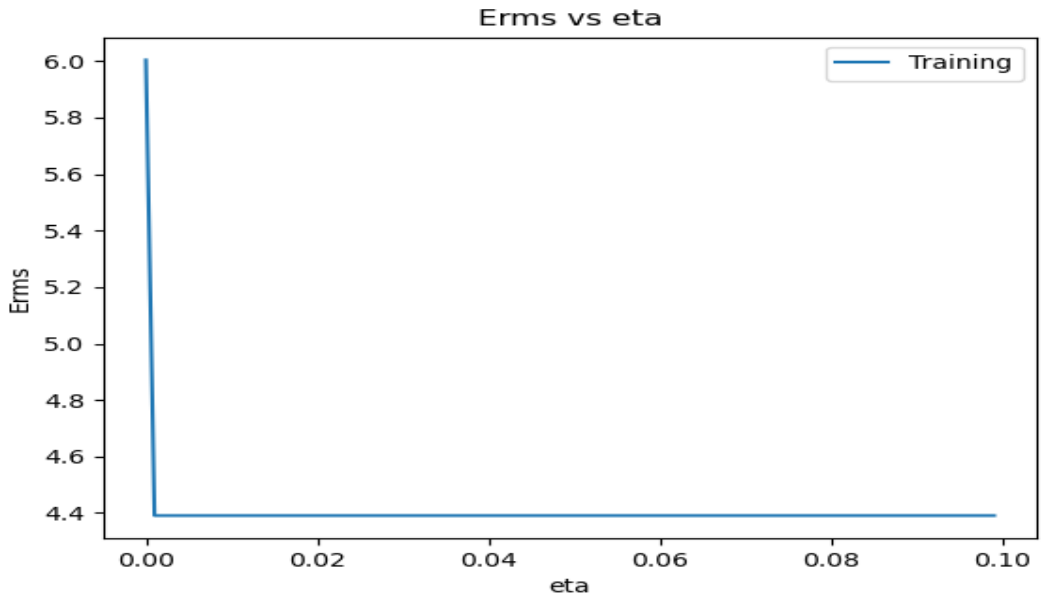

Figure 11: Root mean square error (Erms) vs eta (batch-size=1, lamb = 10**-200, m=3, no of iterations=20000)

Conclusion: We get minimum Root mean square error (Erms) for eta = 0.0001 so eta must be chosen as 0.0001

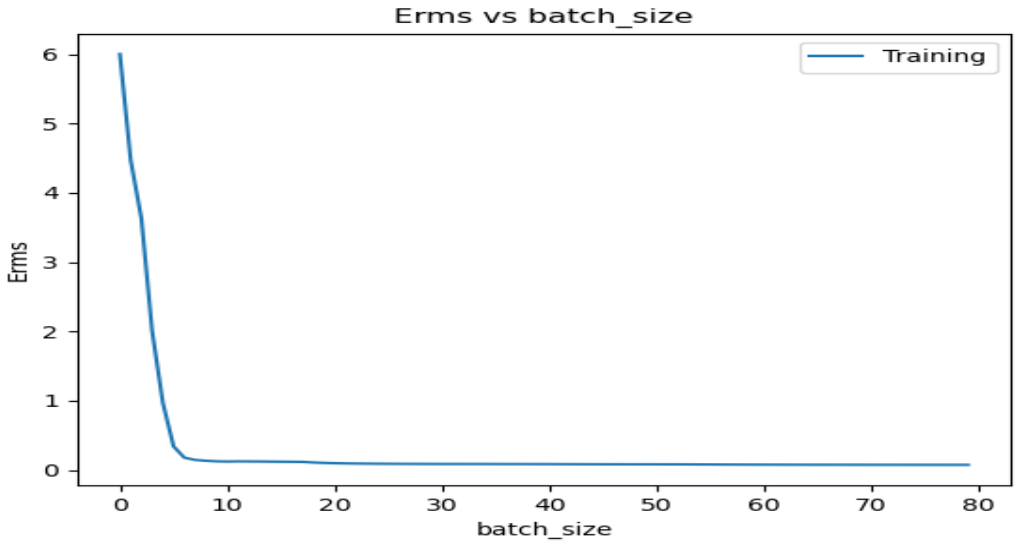

Figure 12: Root mean square error Erms vs batch-size (eta=0.0001, lamb = 10**-20, m=3, no of iterations=5000)

Conclusion: From the above graph of Root mean square error(Erms) vs batch-size we can observe that as batch-size increases Root mean square error (Erms) decreases when batch-size increases from 0 to 5 Root mean square error (Erms) decreases very fast but after 5 Root mean square error (Erms) becomes almost constant and it get minimize for full batch-size. Thus, changing batch-size will not have any extra effect.

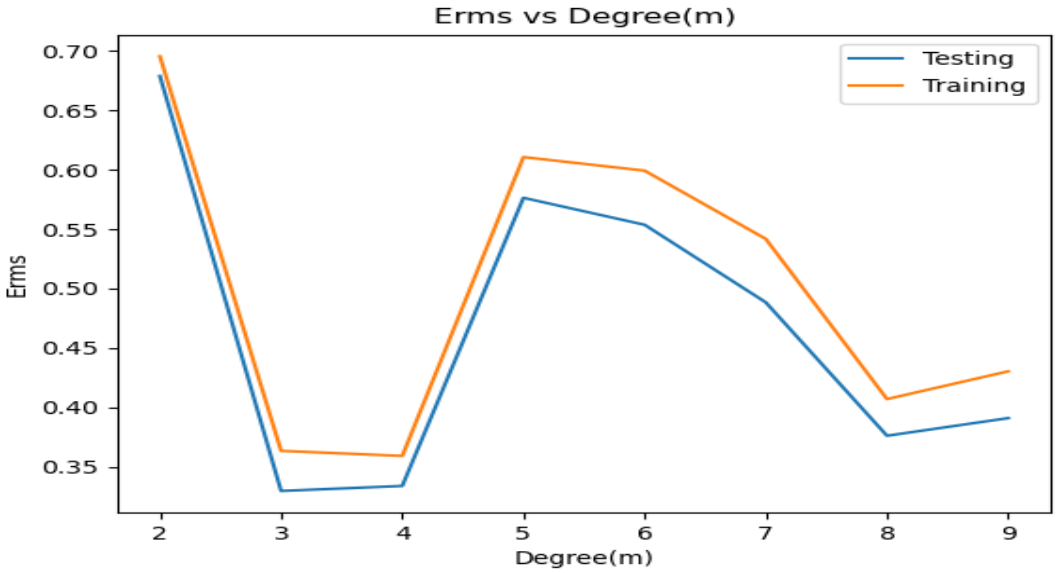

Figure 13: Root mean square error (Erms) vs degree (order) is plotted for testing and training data

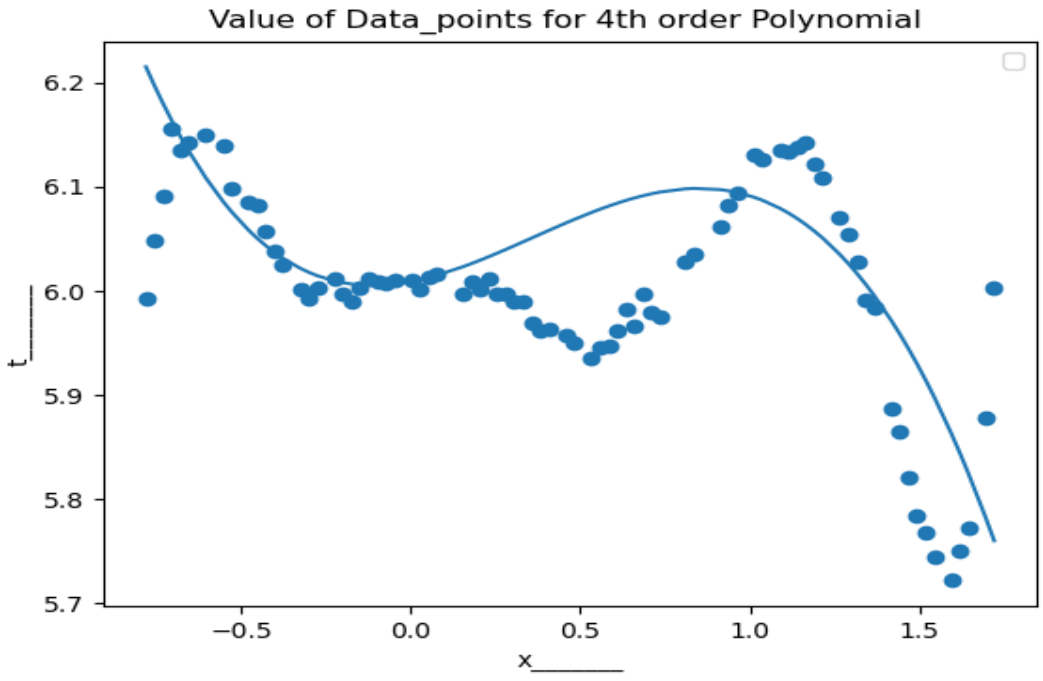

Figure 14: The curve for m=4 for datapoints (x, t)

Conclusion:

1)**The 4-order polynomial obtained:**

Y (x, w) = 5.97889782 + (0.0071313) x^1 + (0.29084741) x^2 +(-0.26554759) x^3+ (0.0135184)x^4

2)**Training_ error at m =4**: 0.361

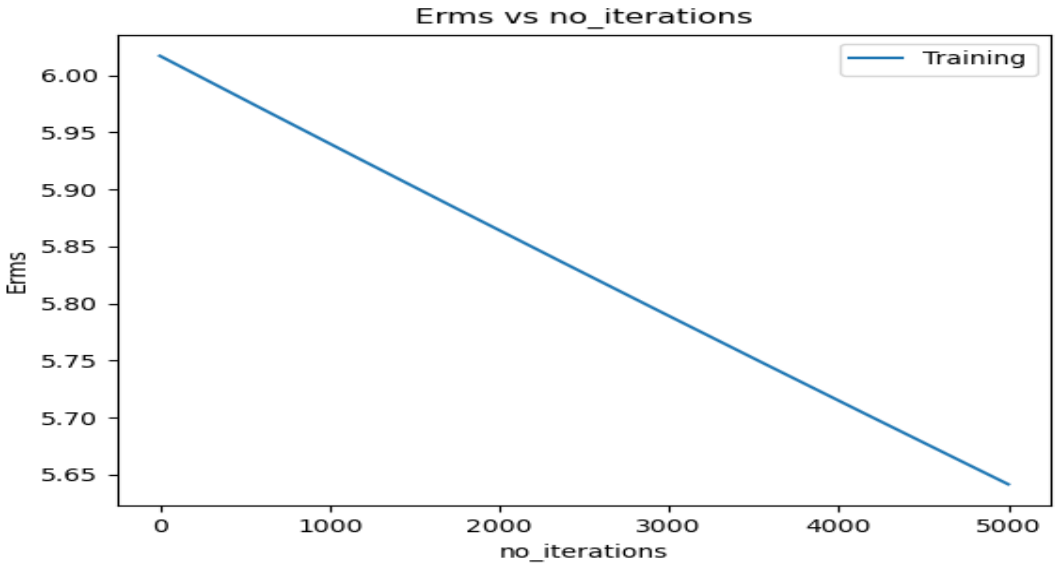

Figure 15: Root mean square error (Erms) vs number of iterations (eta=0.00001, lamb = 10**-20, m=3, batch-size=1)

Conclusion: From the above graph of Root mean square error (Erms) vs number of iterations we can observe that as number of iterations increases Root mean square error (Erms) decreases which shows that stochastic gradient descent (SGD) convergences.

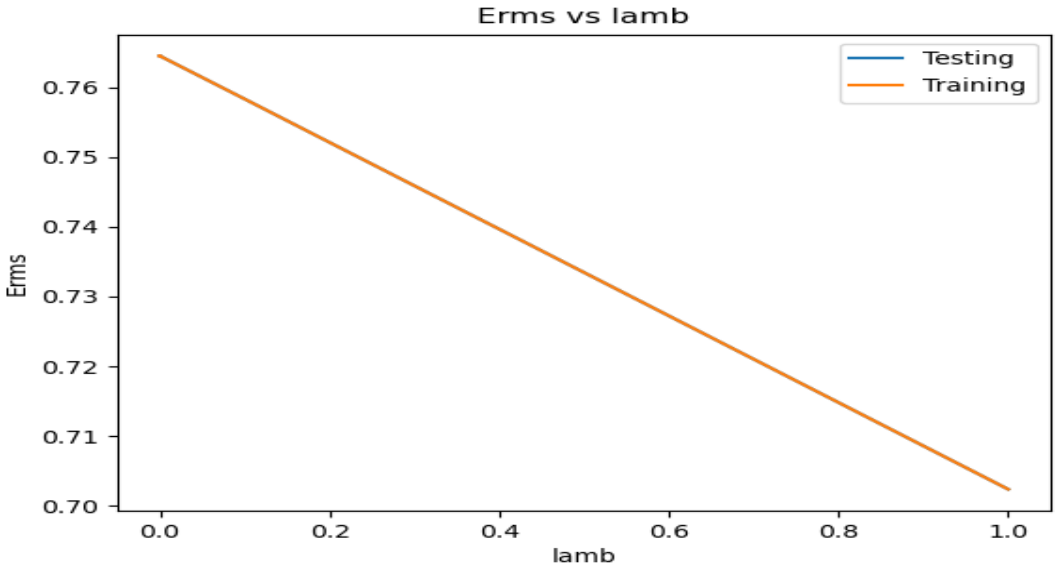

Figure 16: Root mean square error Vs lambda for testing and training data

For 20 Data points

The model is trained using 10 datapoint's and tested remaining 10 datapoint's

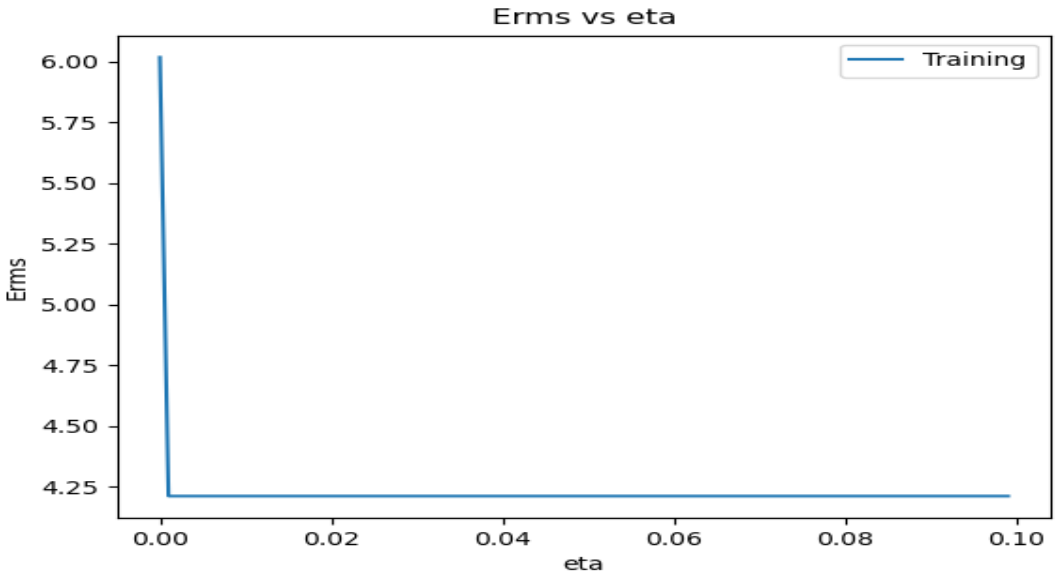

Figure 17: Root mean square error (Erms) vs eta (batch-size=1, lamb = 10**-200, m=3, no of iteration=20000)

Conclusion: We get minimum Root mean square error (Erms) for eta = 0.0001 so eta must be chosen as 0.0001.

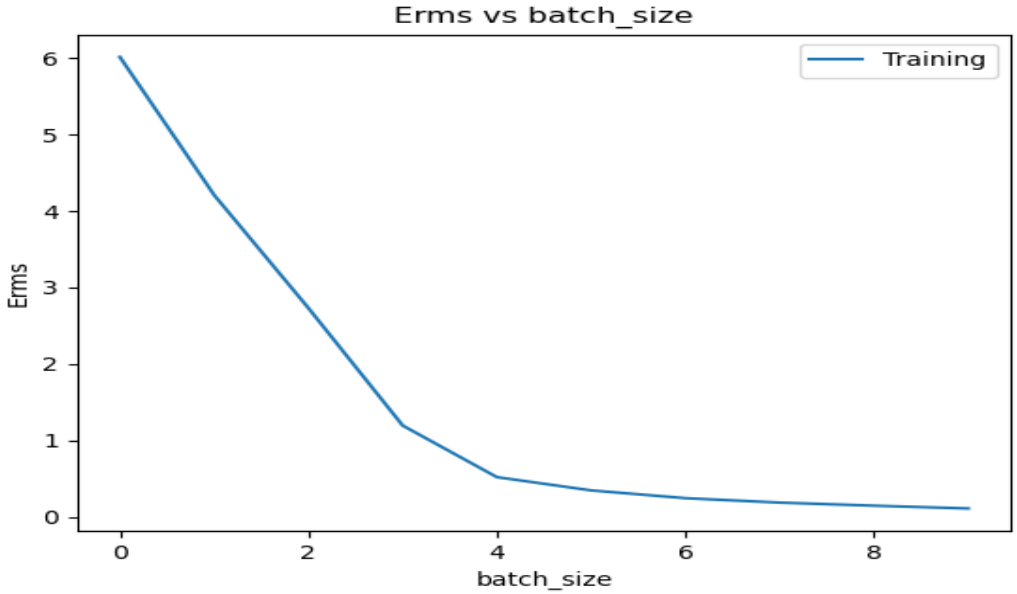

Figure 18: Root mean square error (Erms) vs batch-size (eta=0.0001, lamb = 10**-20, m=3, no of iterations=5000)

Conclusion: From the above graph of Root mean square error (Erms) vs batch-size we can observe that as batch-size increases Root mean square error (Erms) decreases and it get minimize for full batch-size. Thus, changing batch-size will have good effect in this case.

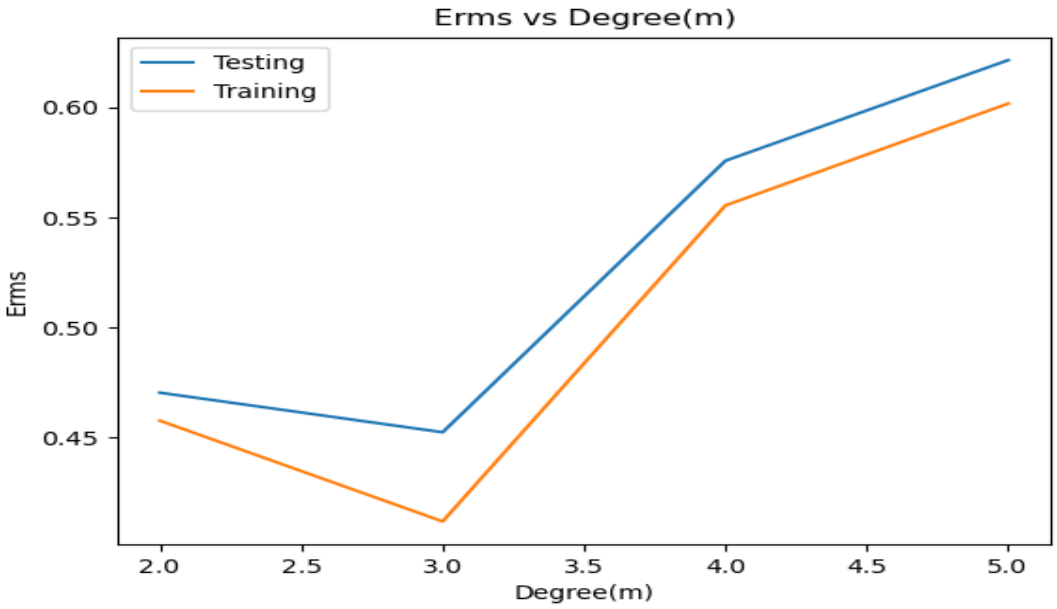

Figure 19: Root mean square error (Erms) vs degree (order) is plotted for testing and training data

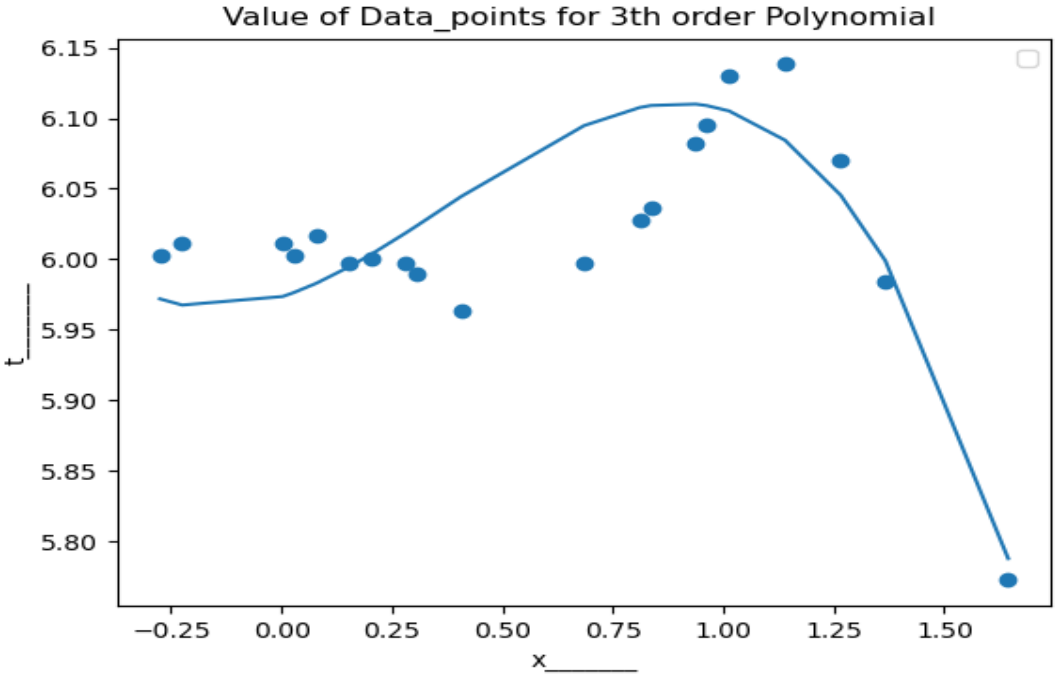

Figure 20: The curve for m=3 for datapoints(x,t)

Conclusion:

1)**The 3-order polynomial obtained**

Y (x, w) = (5.99800959) + (0.18855342) x^1+ (0.05094567) x^2+(-0.17913717) x^3

2)**Training_ error at m = 3:** 0.4472

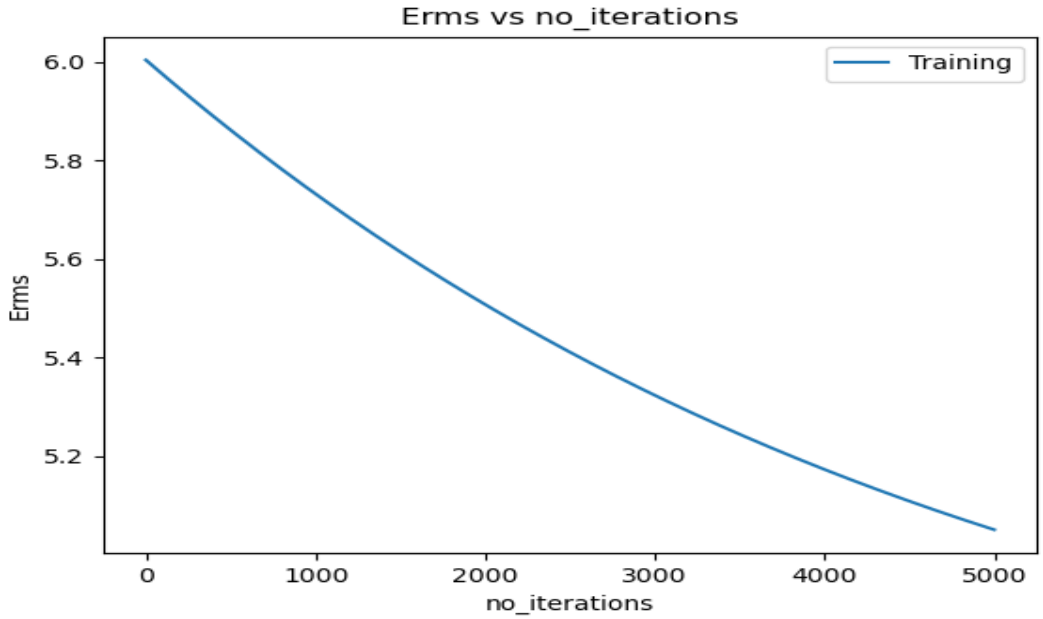

Figure 21: Erms vs number of iterations (eta=0.00001, lamb = 10**-20, m=3, batch-size=1)

Conclusion: From the above graph of Root mean square error (Erms) vs number of iterations we can observe that as number of iterations increases Root mean square error (Erms) decreases which shows that stochastic gradient descent (SGD) convergences.

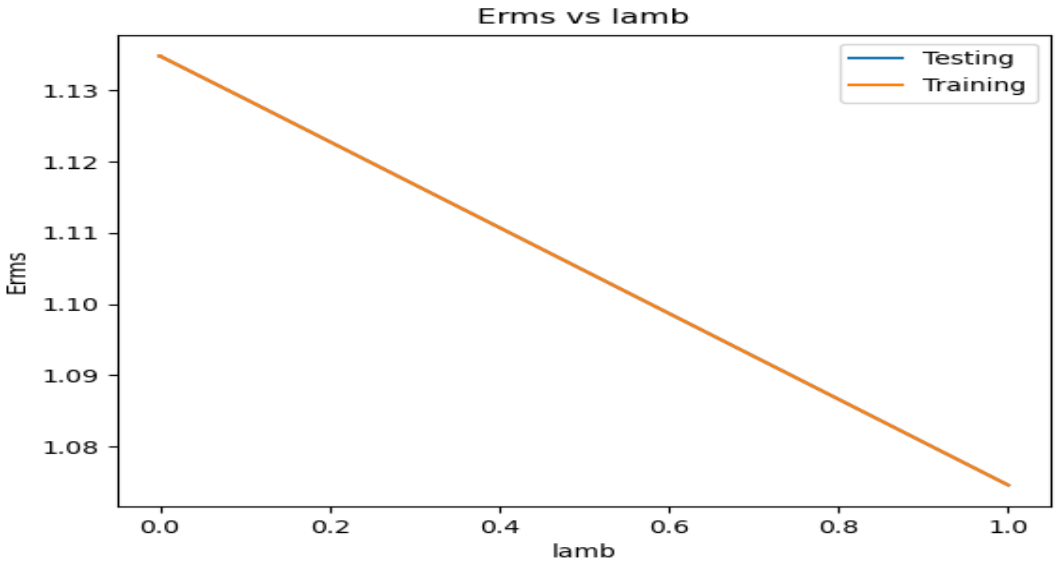

Figure 22: Root mean square error Vs lambda for testing and training data

FINAL CONCLUSION:

For 100 data points, I have considered 80 data-points as training set and remaining 20 data-points as testing set. For the other 20 points given I have considered 10 data-points as training set and remaining 10 data-points as testing set. Using SGD we can observe that the trends in some parameters as described below are clearer when we are using training set of 80 points than 10 points

Case1: When 80 data-points are considered for training:

From figure 11, Erms vs eta (learning rate) plot, keeping other parameters constant we can observe that initially Erms decreases then it becomes almost constant.

From figure 12, Erms vs batch size, keeping other parameters constant we can observe that Erms decreases more steadily then becomes constant. So, the effect of batch size is not considerably seen.

From figure 15, Erms vs number of iterations, keeping other parameters constant we can observe that as number of iterations increases Root mean square error (Erms) decreases which shows that stochastic gradient descent (SGD) convergences.

From figure 14, x vs t we can observe that for 4 order polynomial graph it is passing through more data points so we can say model is close to good fit model.

Case2: When 10 data-points are considered for training:

From figure 17, Erms vs eta plot, keeping other parameters constant we can observe that initially Erms decreases then it becomes almost constant.

From figure 18, Erms vs batch size, keeping other parameters constant we can observe that Erms decreases at a slow rate then becomes constant. Therefore, we can infer that effect of batch size is more prominent.

From figure 21, Erms vs number of iterations, keeping other parameters constant we can observe that as number of iterations increases Root mean square error (Erms) decreases which shows that stochastic gradient descent (SGD) convergences.

From figure 20, x vs t we can observe that for 3 order polynomial graph it is passing through less data points so we can say model is underfitted.

Part 1 b) The generated from a different polynomial, and the noise is now non-Gaussian.

Optimization using Moore-Penrose Pseudoinverse

For 100 Data points

The model is trained using 80 datapoint's and tested remaining 20 datapoint's

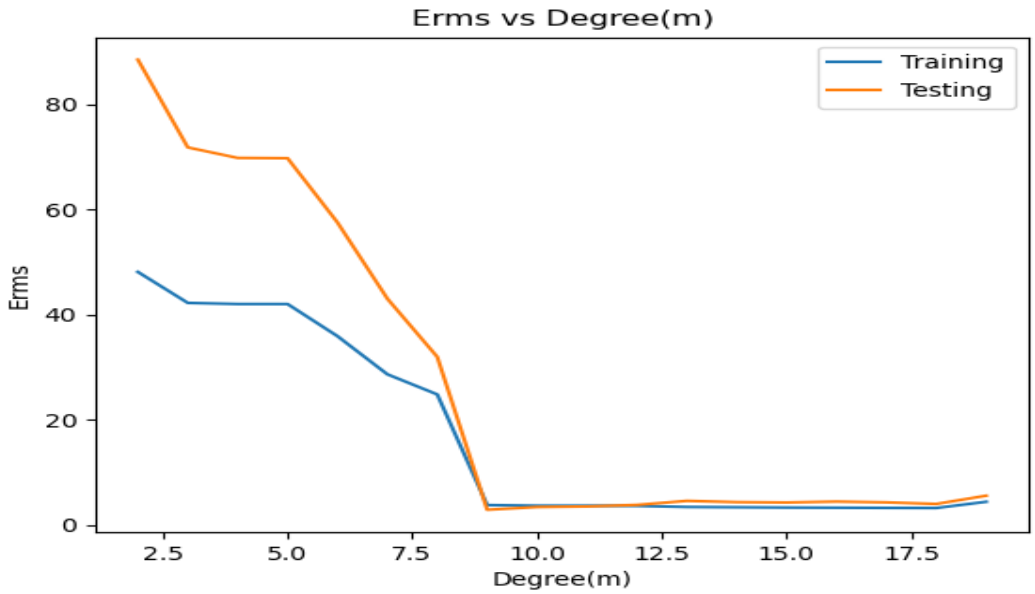

Figure 23: Root mean square error (Erms) vs degree (order) is plotted for testing data

Conclusion: From the plot of Root mean square error (Erms) vs order we can conclude that optimal order will we 9 as after that curve starts overfitting

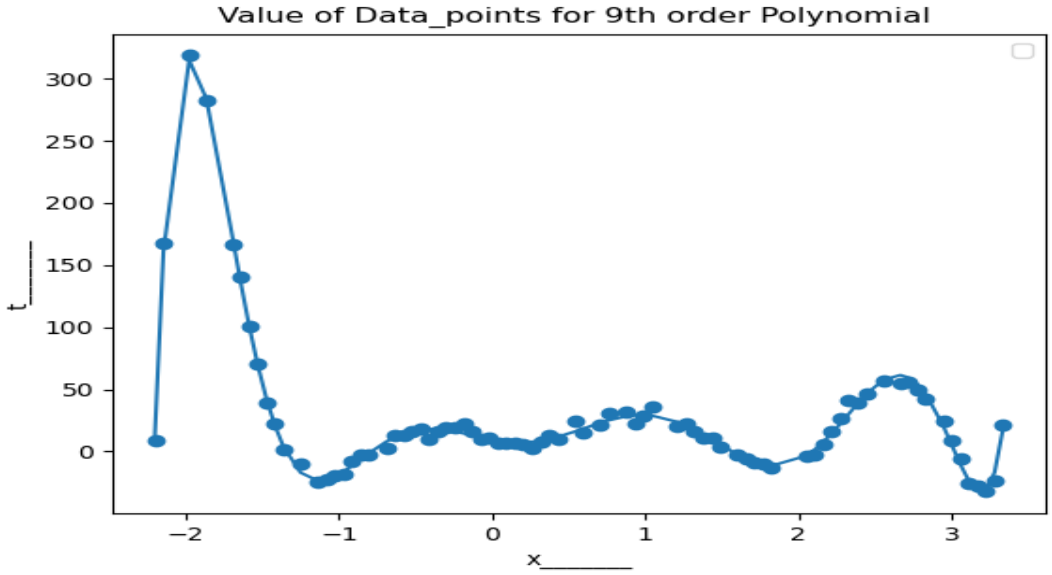

Figure 24: The curve for m =9 for datapoints(x,t)

Conclusions:

1)The 9-order polynomial obtained

Y (x, w) = 9.512623 + (-22.42638853) x^1 + (44.09102529) x^2 +(91.17511202) x^3 + (-88.08799598) x^4 +(-47.00999589) x^5+(46.01203999) x^6 +(1.08394268) x^7 + (-6.12808471) x^8 + (1.01892279) x^9

2) **Variance** = 12.608

Part 2) Trying to model a real-world time-series data set. It contains measurements of a certain quantity at different points in time. In each row, the first value is the date of the measurement (in US format, so month comes before day), and the second value is the actual measurement. We have to train a linear regression model.

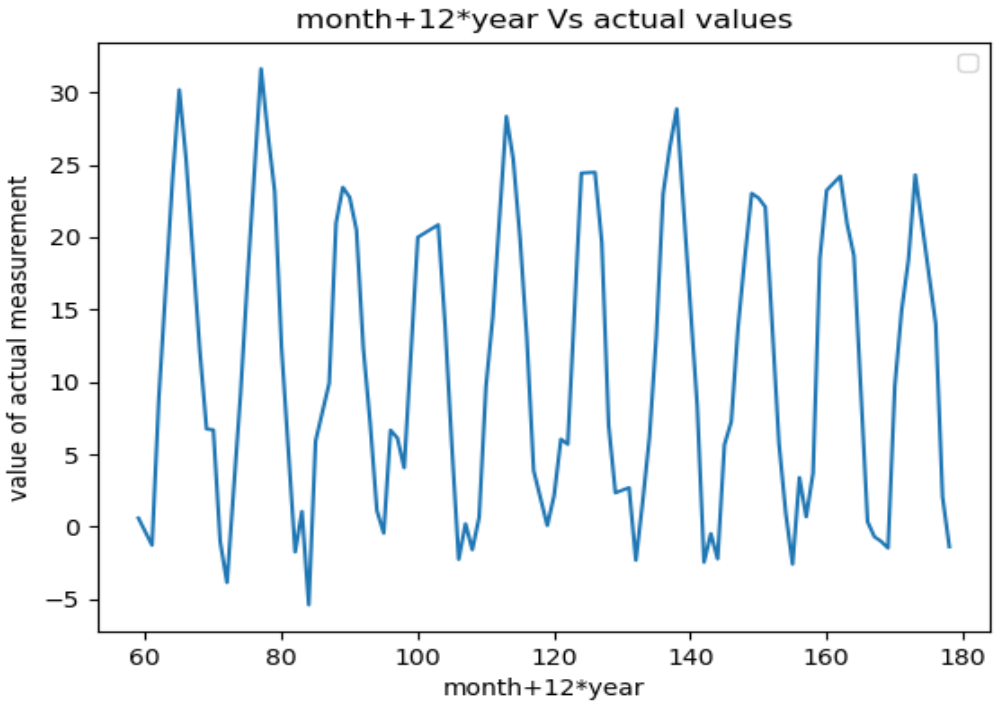

Figure 25: plot of month+12*year Vs actual measurement for each data-point

Explanation: We have extracted month and year for each datapoints and multiplied in a way that gives month+12*year. Then we have plotted month + 12 *year on x axis and value of actual measurement on the y axis and we get above plot. In the plot we can see that values are ranging from negative to positive. The plot is similar to somewhat of sin(x) and cos(x) so we get some-idea that basis function will be some function of sin and cos i.e., basis function will be sinusoidal. So, I have chosen design matrix as.

$$\begin{aligned} \text{Ø}_j(x) &= 1 && \text{if } j=0 \\ &= \sin(((j+1/2)\ *57.1*x_j)/110) && \text{if } j \text{ is odd} \\ &= \cos(((j+1/2)\ *57.1*x_j)/110) && \text{if } j \text{ is even} \end{aligned}$$

Following graphs are plotted using Moore Penrose pseudoinverse optimization techniques with lambda = 10**-18.

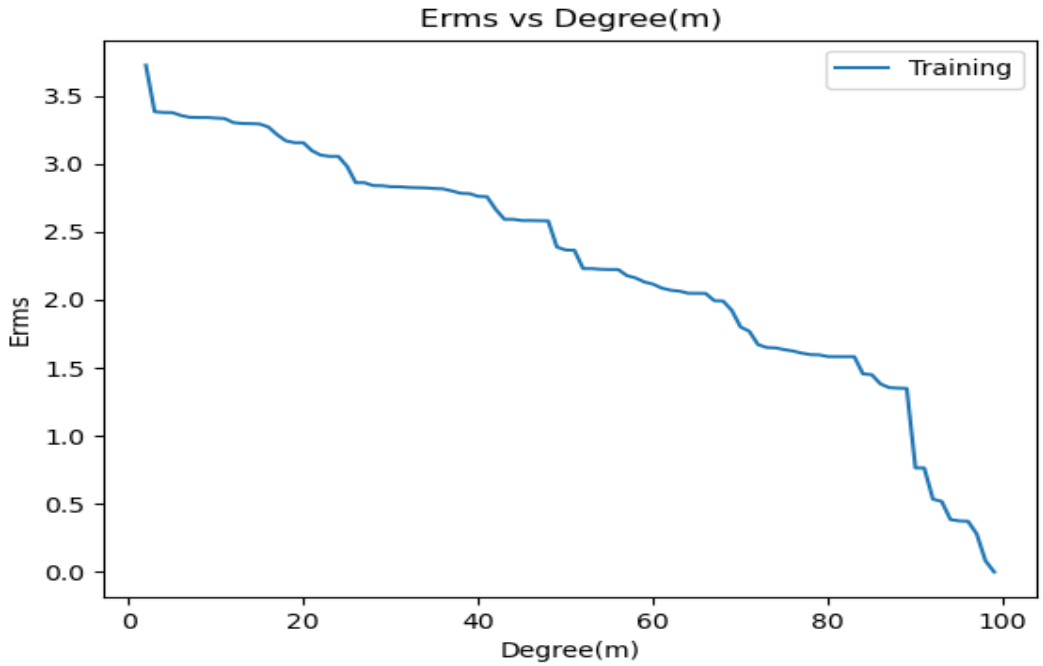

Figure 26: Root mean square error (Erms) vs degree (order) is plotted for training data

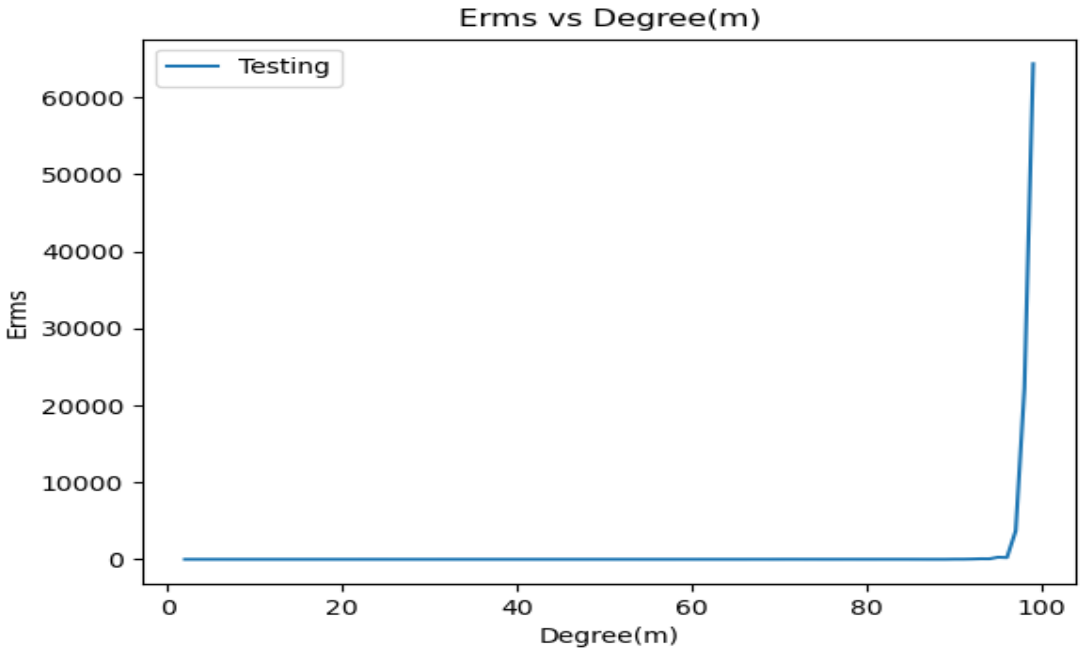

Figure 27: Root mean square error (Erms) vs degree (order) is plotted for testing data

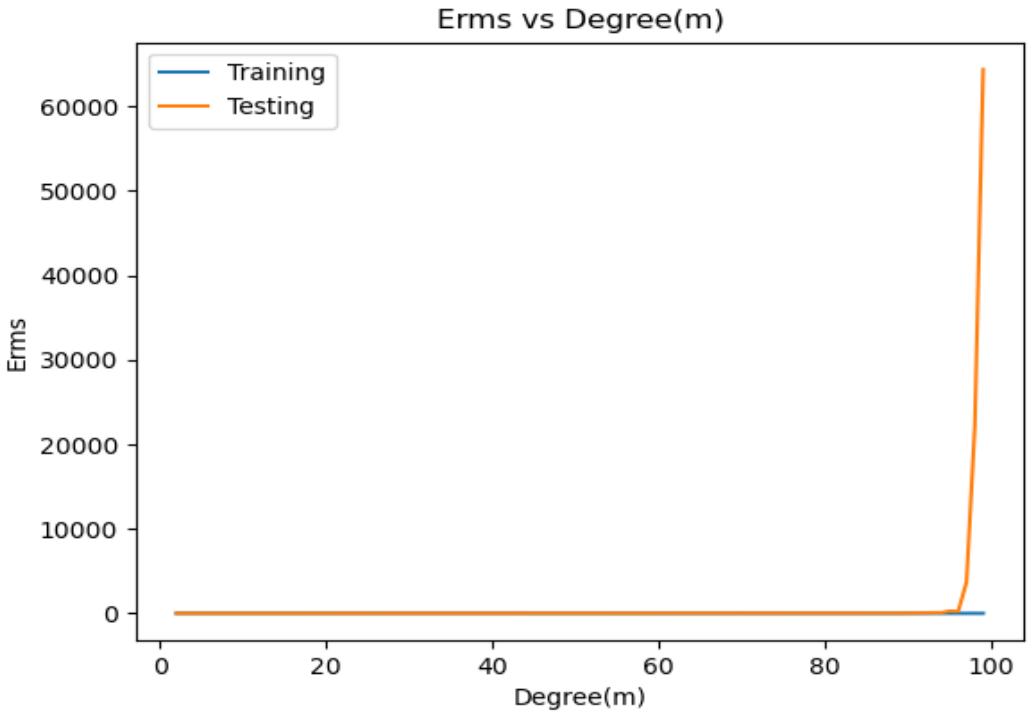

Figure 28: Root mean square error (Erms) vs degree (order) is plotted for testing and training data

	id	value	
	05-01-2010	27.45035	
	04-01-2009	23.212	
	09-01-2013	6.959757	
	01-01-2006	5.367286	
	02-01-2007	9.241895	
	08-01-2012	14.71121	
	06-01-2014	26.35657	
	03-01-2008	14.07444	
	12-01-2004	-0.14312	
	07-01-2011	21.02886	

Figure 29: Values obtained for the testing data

The above data on Kaggle competition obtained a score of 6.74