ELL 409

Report

2019CS10404

SOURAV

PART 1.1

A.) Pseudoinverse(PINV)

Using pseudo inverse function we generally use the formulae for calculating PINV is Y = XB + E
Where Y is column matrix of predicted value by the model and the B is the column matrix that contains the weight of our polynomial and X is the matrix of order n*m where n is the number of training values and m is the orde of polynomial and in this matrix contain the array of each degree of train data xi from 0 to m. where E is the Error column matrix that contains the error between our actual and the predicted model.
Now formulae used for finding thes is

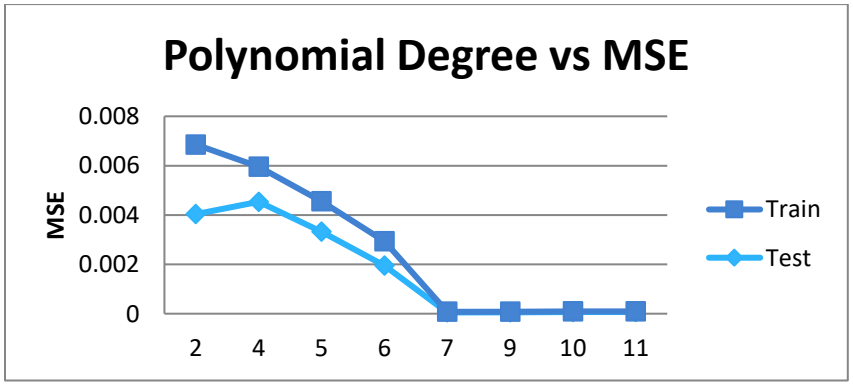
$Y\_predicted = XB$

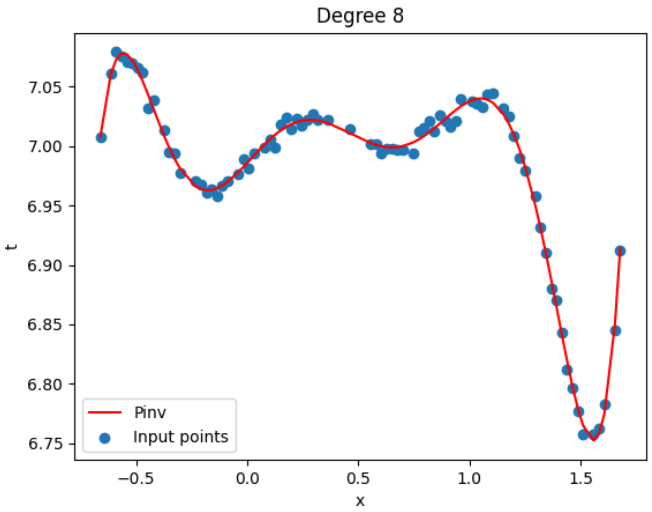
$E = Y\_prdicted - Y$

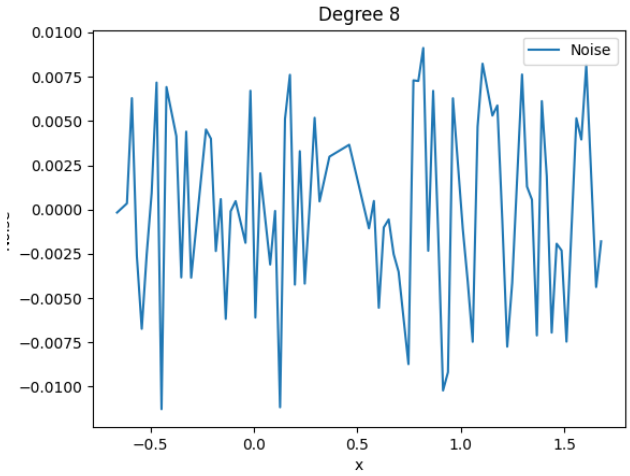
$B = (X^TX)^{-1}X^TY$

Using 20 test points

In this part we are taking the 80% of the training data to train our model and remaining 20% data is for testingthe model and here are the curves shows are Right curve shows the Predicted pinv in straight line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model. And also shows the mse between testing data and training data.

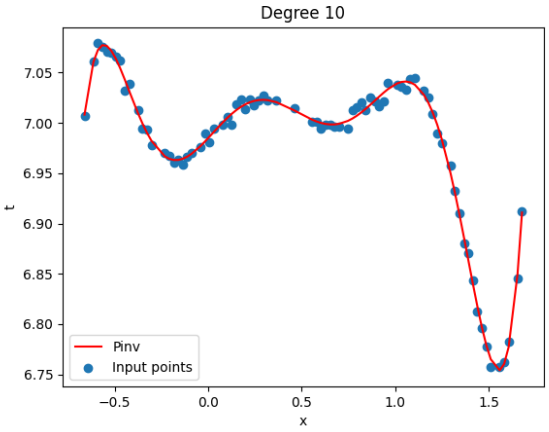


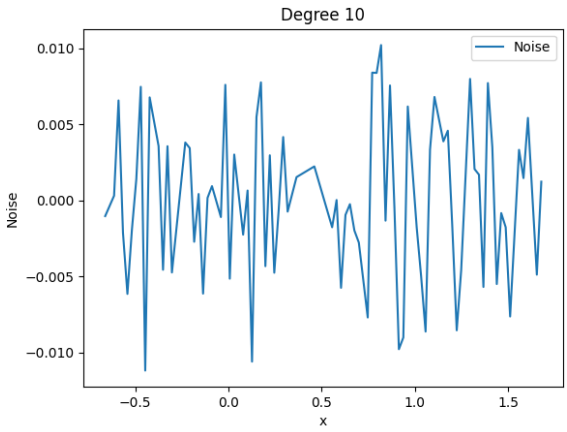




weights = [6.98582293 0.20258072 0.0798666 -1.70960176 1.29353628
2.40357832 -3.17961907 0.95482011 0.00679319]

MSE_Test = 0.00004975 MSE_Train = 0.00002715





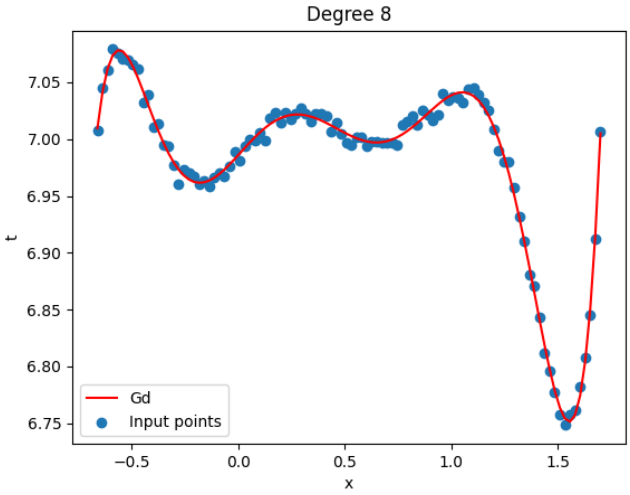
weights = [6.98487708 0.20036875 0.12081482 -1.67835242 1.02315114
2.3877627 -2.56085096 0.66969126 -0.41134698 0.39499986 -0.09277138]

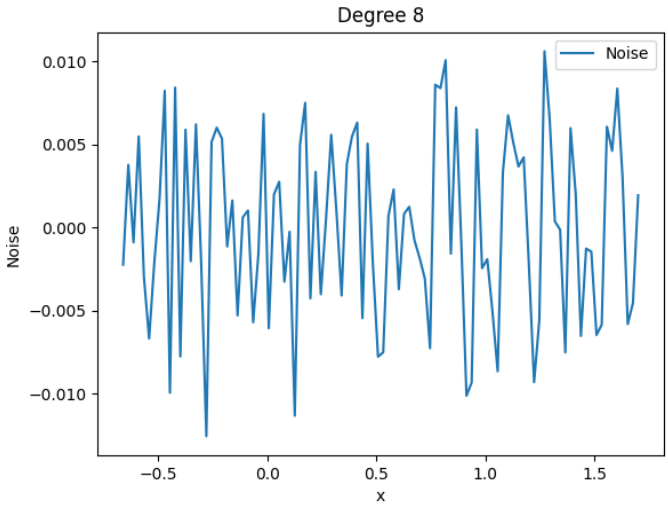
MSE_Test = 0.00006299 MSE_Train = 0.00002617

From the above graphs we easily observe that the degree of the curve is lie in the interval 8>= and <=10 and also able to see the weighgt array and the MSE for the test and train data is also low So of further increase or decrease the degree of the polynomial then it give more error in the test data cse.

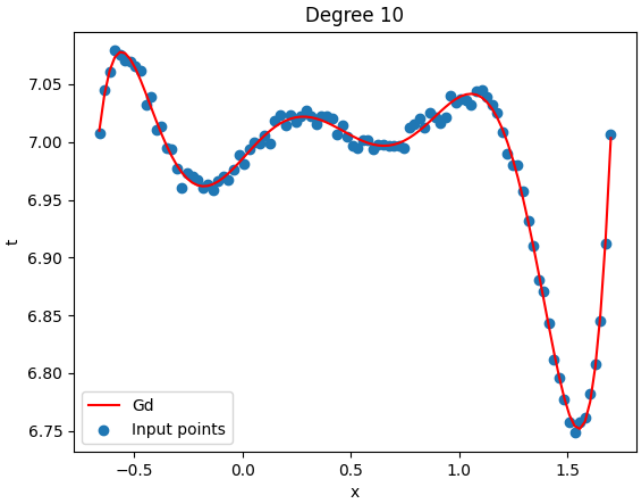
Using 100 points

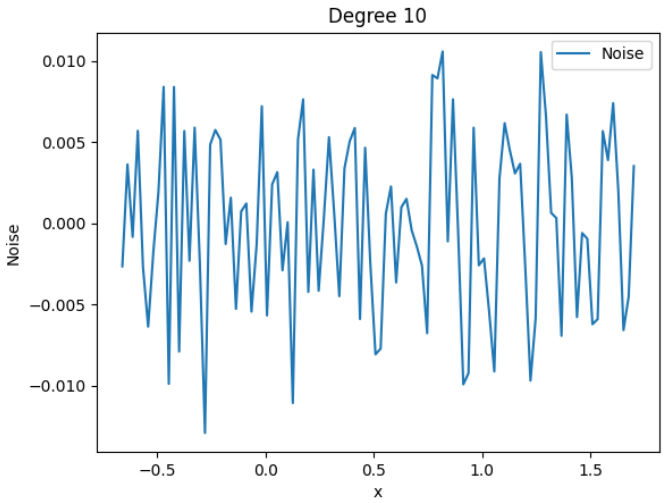
In this part we are taking the 100% of the training data to train our model here are the curves shows are Right curve shows the Predicted pinv in straight line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model.





weights = [6.98576242 0.20682786 0.06279458 -1.7327308 1.35916163 2.40685459 -3.23668333 0.98250867 0.00402392]





weights = [6.985381 0.20585898 0.07809981 -1.72207375 1.25756498 2.40925374 -3.01302662 0.86724908 -0.13292933 0.13533212 -0.03197342]

From the above graphs we easily observe that the degree of the curve is lie in the interval 8>= and <=10 and also able to see the weighgt array and the MSE for the test

and train data is also low So of further increase or decrease the degree of the polynomial then it give more error in the test data cse.

B.) Gradient Descent(GD)

Using Gradient Descet function we implement our cdoe for polynomial regression for training our model so that ir can give best predicted value if we test on any non trained data. For calculating this we use the formulae i.e bi = bi – n d/dw(J) where is the mean squared error i.e J = (∑ (y_prdicted – y)^2)/n

For this we calculate the Y_predicted by Y = XB + E similar as the PInv but the different between pinv and the gradient decsne is the process of finding the weight array i.e B

So here we used formulaes are the :

Y_predicted = XB

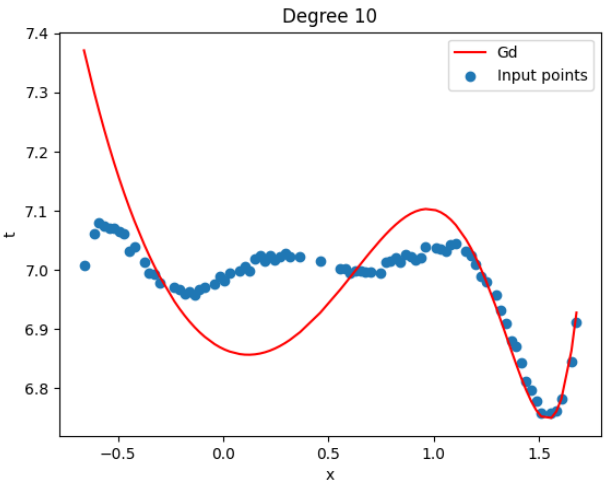
B = B – 2/batch_size*learning rate*(X^T.(Y_predicted – Y))

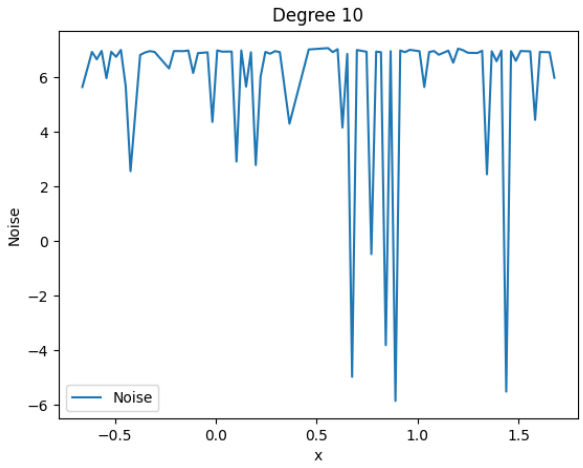
E = Y_predicted - Y

Using 20 test points

In this part we are taking the 80% of the training data to train our model and remaining 20% data is for testingthe model and here are the curves shows are Right curve shows the Predicted gd in line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model. And also shows the mse between testing data and training data. Here we work on different parameter i.e changing the batch_size and changing the learning rate etc.

Batch_size = 75 (Right = x-tcurve,Left = Noise Curve)

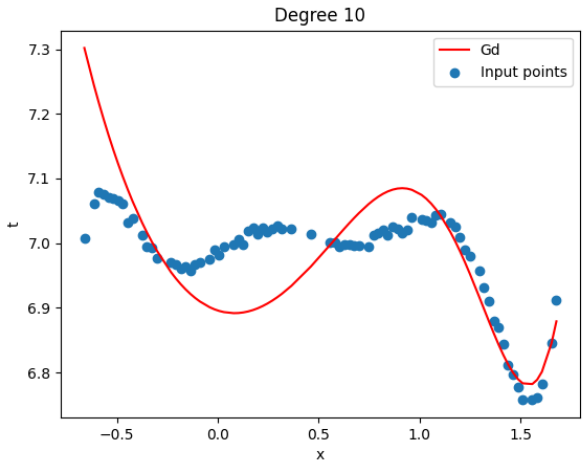


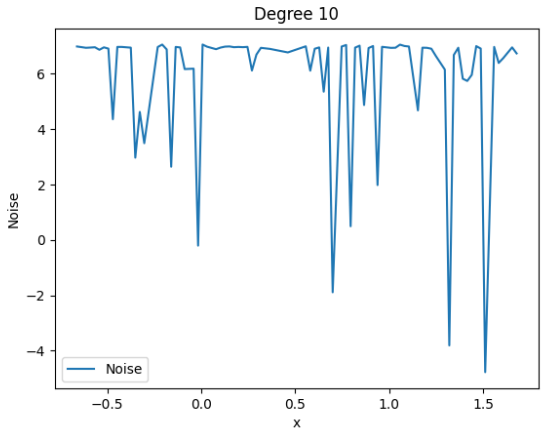


weights = [6.86633315 -0.16831201 0.73371114 -0.15828912 0.0553307 -0.15461257 -0.06099323 -0.07000812 0.02032835 0.04810589 -0.00971588]

MSE_Test = 0.00910053

Batch_size = 50 (Right = x-tcurve,Left = Noise Curve)

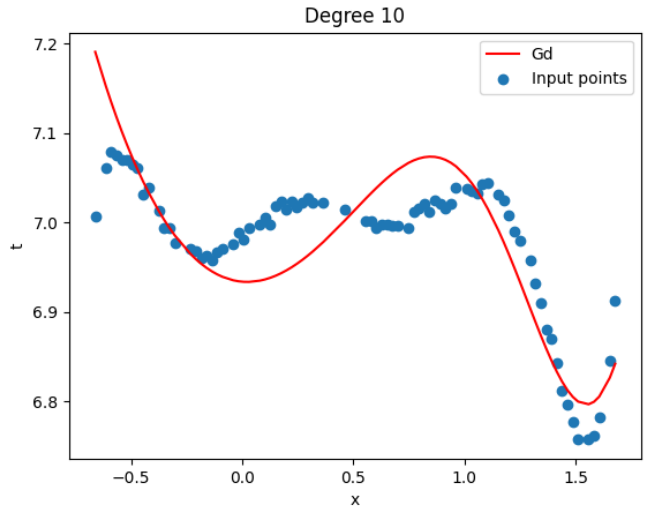


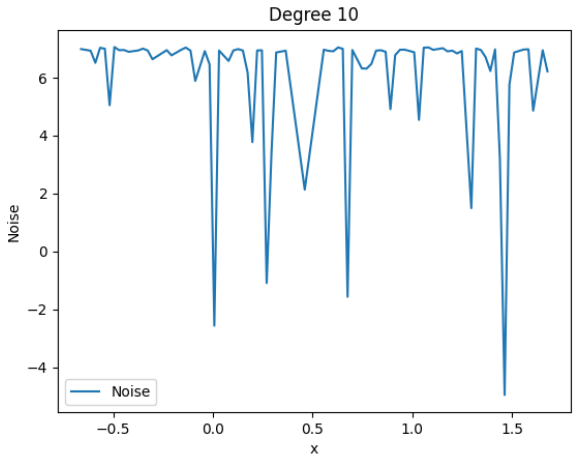


weights = [6.89596414 -0.10080753 0.6296002 -0.16212987 0.00963096 -0.15296251 -0.06582801 -0.05145504 0.03666212 0.06309083 -0.02517511]

MSE_Test = 0.00542244

Batch_size = 25 (Right = x-tcurve,Left = Noise Curve)





weights = [6.93404203 -0.02039674 0.45352654 -0.1439874 -0.03567764 -0.11599608 -0.05576339 -0.02342914 0.03371378 0.05241219 -0.02458964]

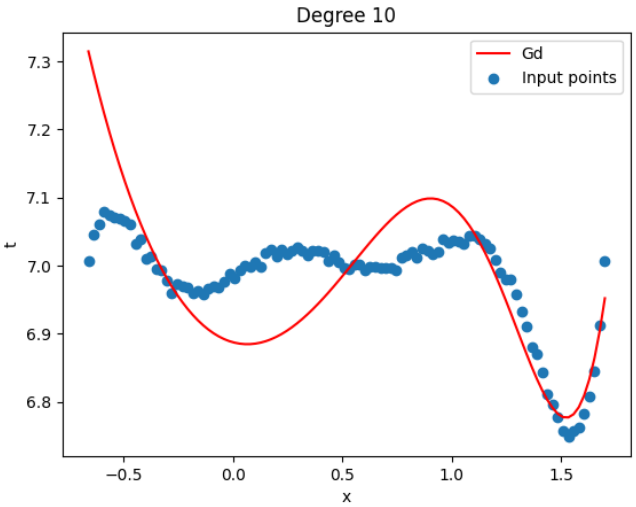
MSE_Test = 0.00299647

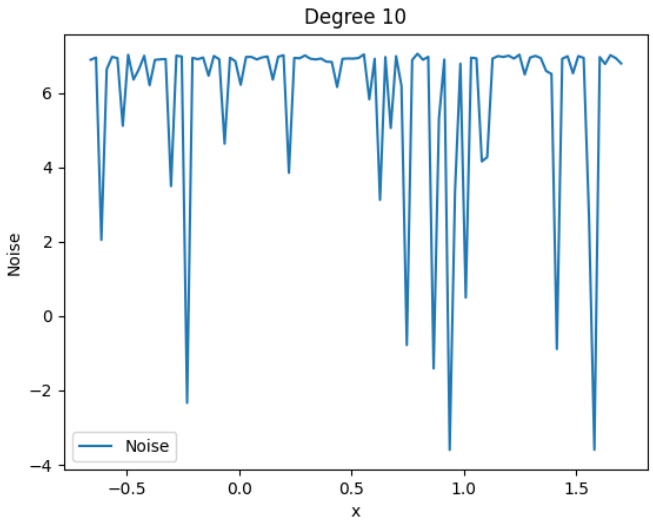
Here we see the difference between the curves after changing the degree and the batch_size if the curves and the MSE is less for test data form poly of degree at 10 if we further increase id decrease the degree we many be found more error then we can easily say that this the best pidicted model.

Using 100 points

In this part we are taking the 100% of the training data to train our model here are the curves shows are Right curve shows the Predicted GD in line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Gd model.

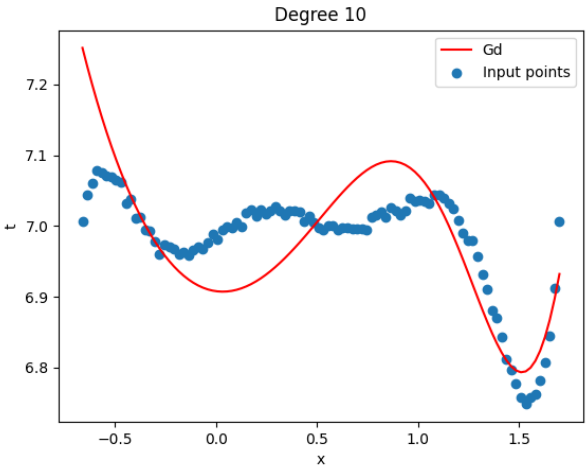
Batch_size = 75 (Right = x-tcurve,Left = Noise Curve)

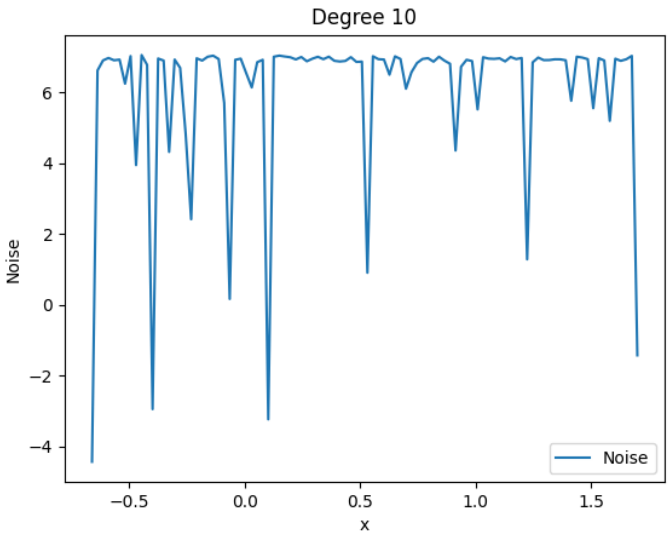




weights = [6.88768137 -0.08747246 0.67934874 -0.18363494 0.00699493 -0.1742578 -0.07038886 -0.05693008 0.04456017 0.07096708 -0.02937844]

Batch_size = 50 (Right = x-tcurve,Left = Noise Curve)

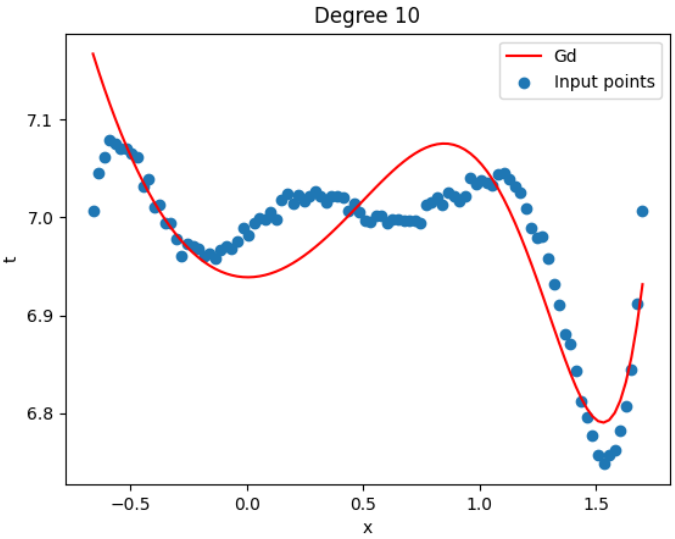


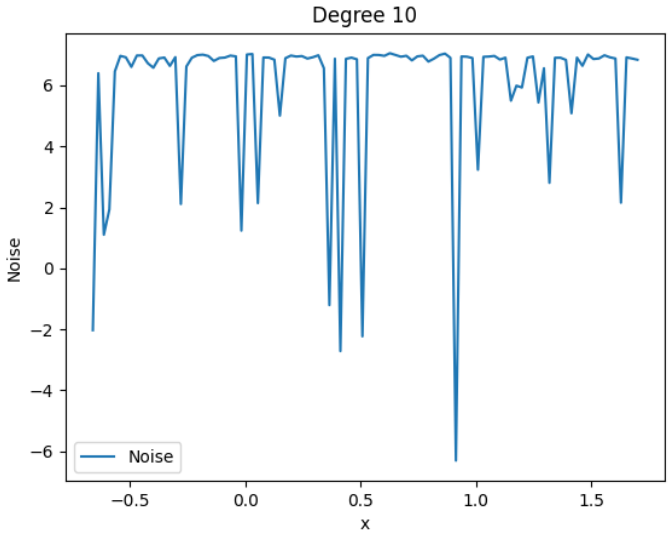


weights = [6.90807953 -0.03905583 0.58540191 -0.17595649 -0.02226575 -0.15842177

-0.06766529 -0.0412603   0.04631715 0.06917874 -0.03178193]

Batch_size = 25 (Right = x-tcurve,Left = Noise Curve)





weights = [6.93893088 -0.00428067 0.41794459 -0.14357648 -0.04175044 -0.09603395

-0.04145398 -0.01371306 0.02169283 0.03008812 -0.0115415]

Here we see the difference between the curves after changing the degree and the batch_size if the curves and the MSE is less for test data form poly of degree at 10 if we further increase id decrease the degree we many be found more error then we can easily say that this the best pidicted model

PART 1.2

A.)Pseudoinverse(PINV)

Using pseudo inverse function we generally use the formulae for calculating PINV is Y = XB + E
Where Y is column matrix of predicted value by the model and the B is the column matrix that contains the weight of our polynomial and X is the matrix of order n*m where n is the number of training values and m is the orde of polynomial and in this matrix contain the array of each degree of train data xi from 0 to m. where E is the Error column matrix that contains the error between our actual and the predicted model.
Now formulae used for finding thes is

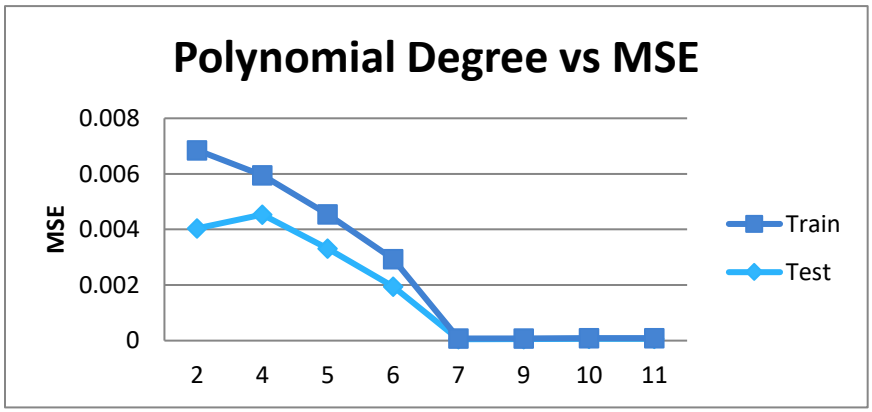
Y_predicted = XB

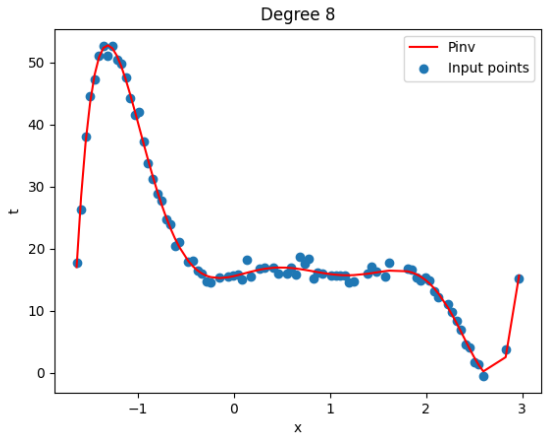
E = Y_prdicted – Y

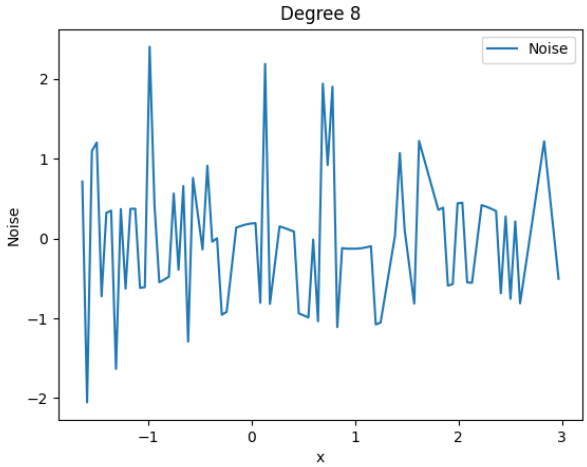
$B = (X^TX)^{-1}X^TY$

Using 20 test points

In this part we are taking the 80% of the training data to train our model and remaining 20% data is for testingthe model and here are the curves shows are Right curve shows the Predicted pinv in straight line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model. And also shows the mse between testing data and training data.



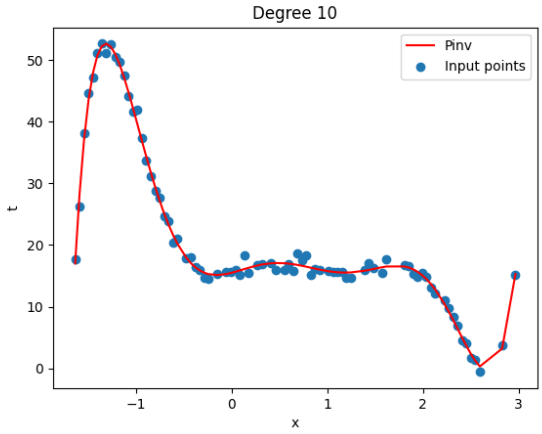


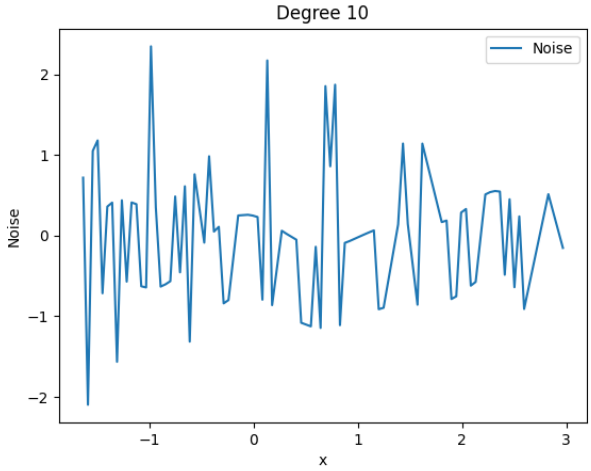


weights = [15.54860991 3.53084501 6.26572208 -22.00062178 11.51809226

5.2351054 -5.14751923 0.92980674 0.00704854]

MSE_Test = 1.02911497 MSE_Train = 0.69202110





weights = [15.4903609 4.02770374 6.9552562 -23.64183768 10.44088557

6.87230572 -4.71682448 0.26217551 0.01117334 0.09742809

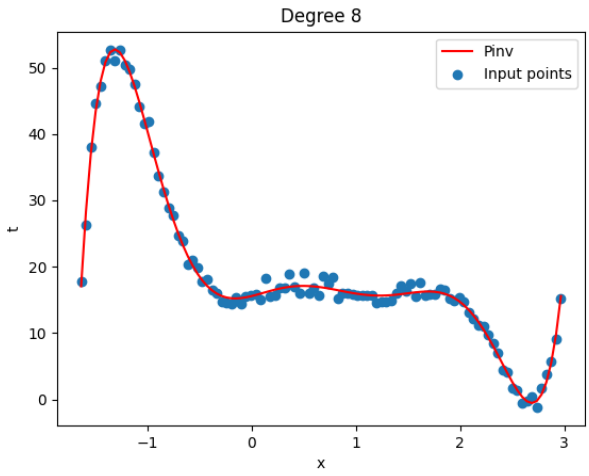
-0.01818091]

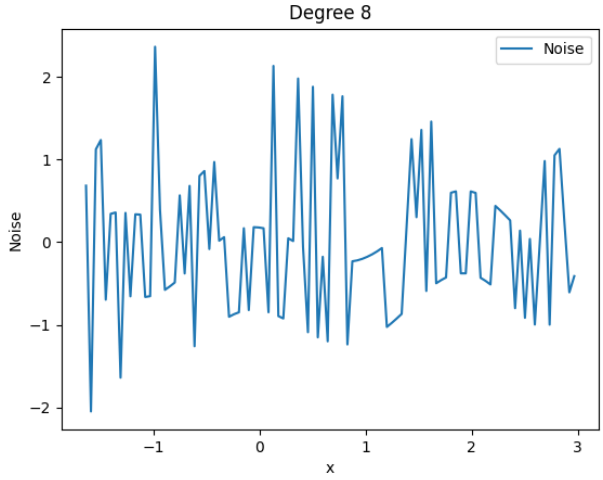
MSE_Test = 0.95906158 MSE_Train = 0.67404456

From the above graphs we easily observe that the degree of the curve is lie in the interval 8>= and <=10 and also able to see the weighgt array and the MSE for the test and train data is also low So of further increase or decrease the degree of the polynomial then it give more error in the test data cse.

Using 100 points

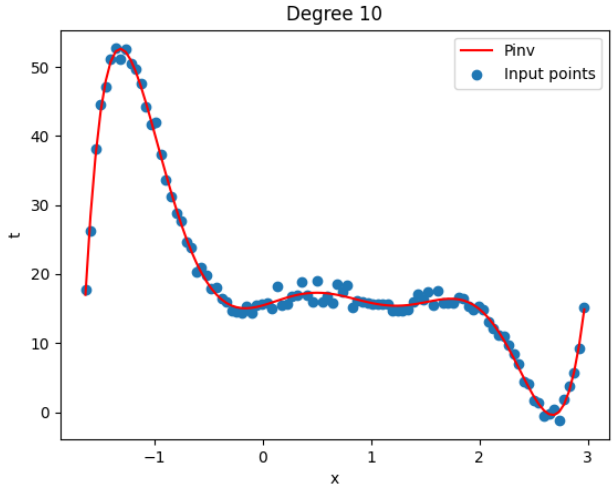
In this part we are taking the 100% of the training data to train our model here are the curves shows are Right curve shows the Predicted pinv in straight line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model.

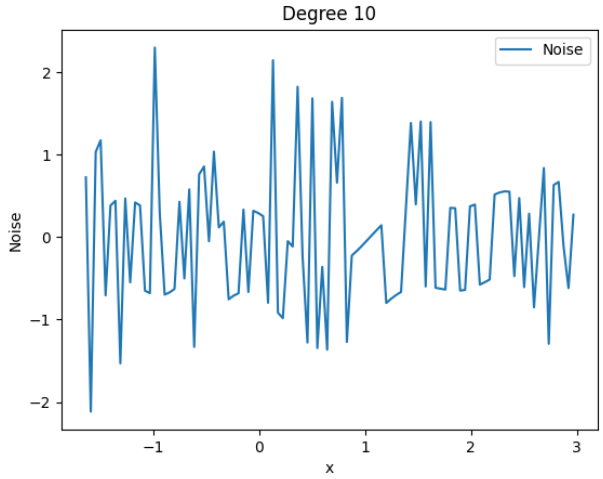




weights = [15.56102132 3.85885446 6.48177593 -22.54466242 11.31185183

5.5056298 -5.12665684 0.88325375 0.01454267]





weights = [15.4509531 4.45586761 7.72454115 -24.58713399 9.37252712

7.6514896 -4.29367362 -0.04580927 -0.02098151 0.14324459

-0.02515405]

From the above graphs we easily observe that the degree of the curve is lie in the interval 8>= and <=10 and also able to see the weighgt array and the MSE for the test

and train data is also low So of further increase or decrease the degree of the polynomial then it give more error in the test data cse.

B.)Gradient Descent(GD)

Using Gradient Descet function we implement our cdoe for polynomial regression for training our model so that ir can give best predicted value if we test on any non trained data. For calculating this we use the formulae i.e bi = bi – n d/dw(J) where is the mean squared error i.e J = $(\sum (y\_prdicted - y)^2)/n$

For this we calculate the Y_predicted by Y = XB + E similar as the PInv but the different between pinv and the gradient decsne is the process of finding the weight array i.e B

So here we used formulaes are the :

Y_predicted = XB

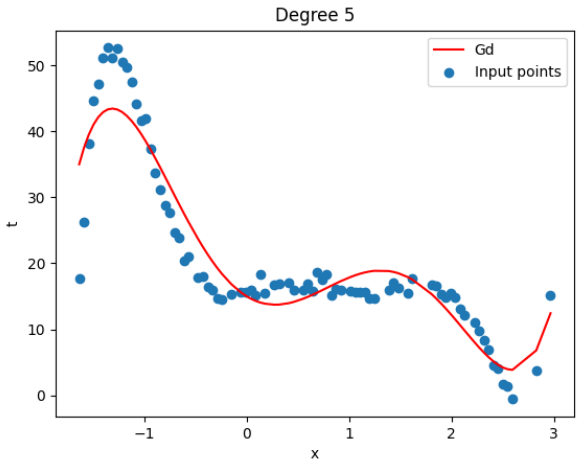
B = B – 2/batch_size*learning rate*(X^T.(Y_predicted – Y))

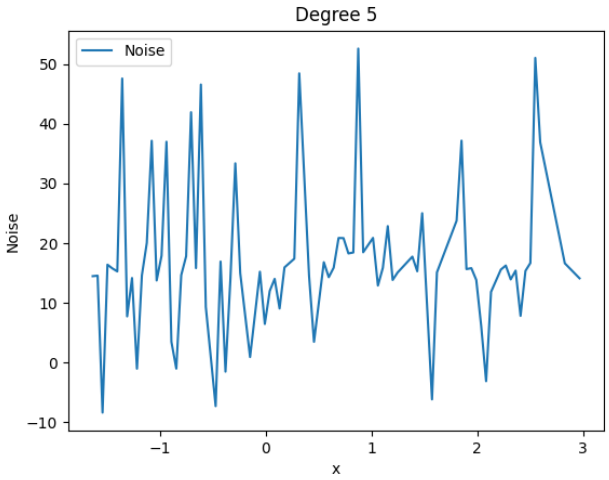
E = Y_predicted - Y

Using 20 test points

In this part we are taking the 80% of the training data to train our model and remaining 20% data is for testingthe model and here are the curves shows are Right curve shows the Predicted gd in line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Pinv model. And also shows the mse between testing data and training data. Here we work on different parameter i.e changing the batch_size and changing the learning rate etc.

Batch_size = 25 (Right = x-tcurve,Left = Noise Curve)

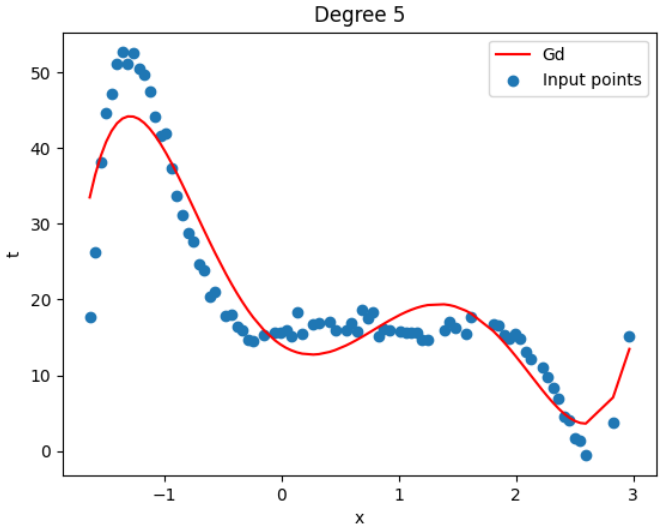


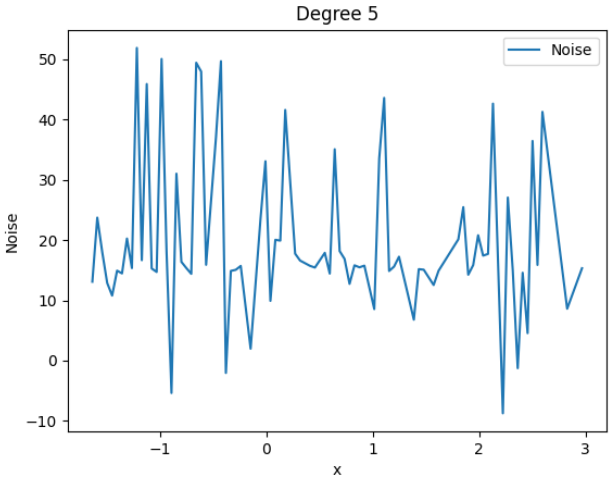


weights = [15.00018347 -9.53258839 18.84887586 -2.52644427 -5.48607588 1.52602371]

MSE_Test = 11.34692010

Batch_size = 50 (Right = x-tcurve,Left = Noise Curve)

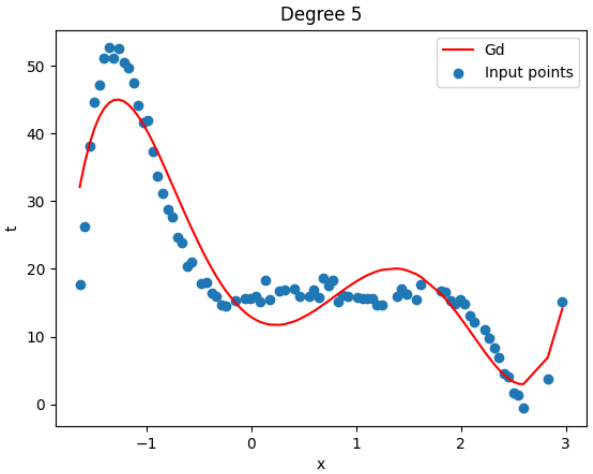


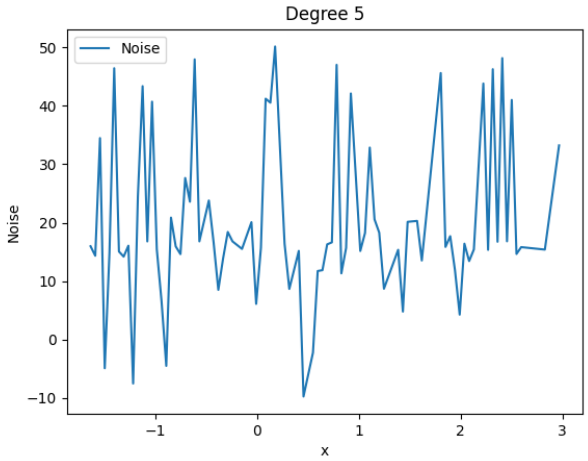


weights = [13.98972236 -10.06502109 21.11140069 -2.61433875 -6.25605218 1.72466963]

MSE_Test = 12.84296067

Batch_size = 25 (Right = x-tcurve,Left = Noise Curve)





weights = [12.88178105 -10.31444718 23.5429843   -2.86617048 -7.0605204
  1.94223848]

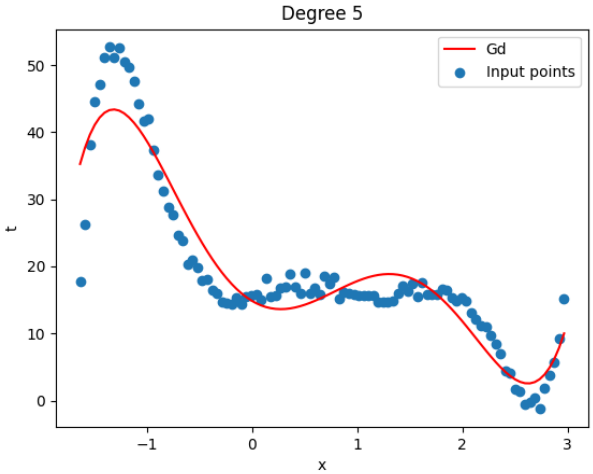
MSE_Test = 14.19817399

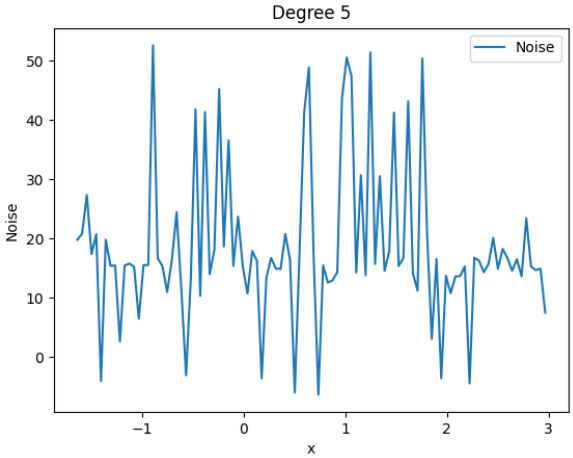
Here we see the difference between the curves after changing the degree and the batch_size if the curves and the MSE is less for test data form poly of degree at 10 if we further increase id decrease the degree we many be found more error then we can easily say that this the best pidicted model

Using 100 points

In this part we are taking the 100% of the training data to train our model here are the curves shows are Right curve shows the Predicted GD in line and scatter points shows the training data and the curve given in the left shown the Graph between noise that prodecued during training the data and the x.The above label of each groh shown is the degree of polynomial that our model work on and after the weight array i.e B is sown below for Gd model.

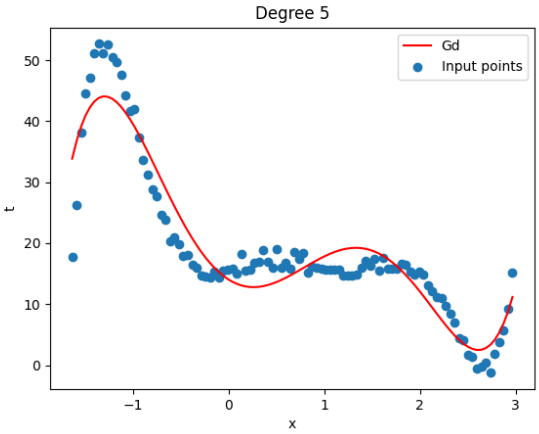
Batch_size = 75 (Right = x-tcurve,Left = Noise Curve)

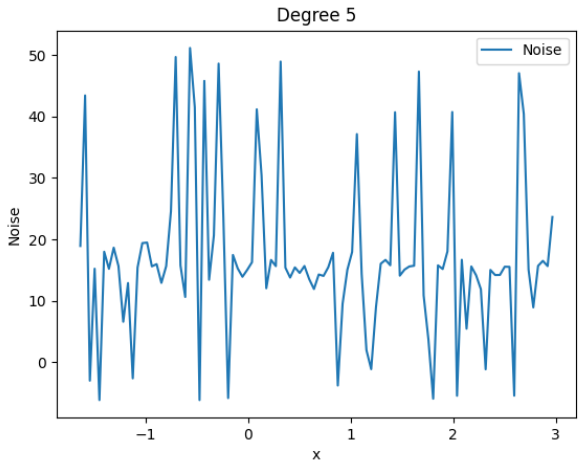




weights = [14.85515591 -9.36158312 18.8457391 -2.5976214 -5.45720458
1.51239398]

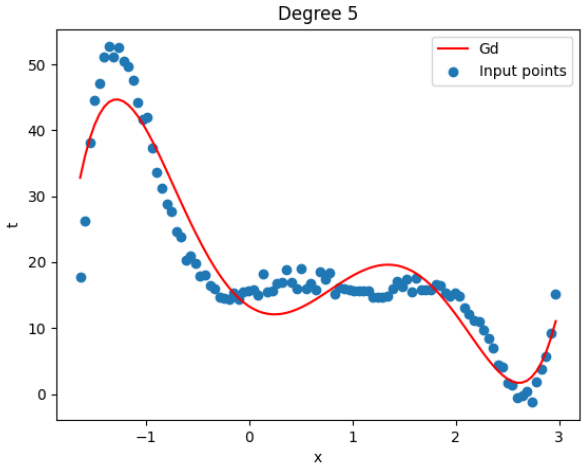
Batch_size = 50 (Right = x-tcurve,Left = Noise Curve)

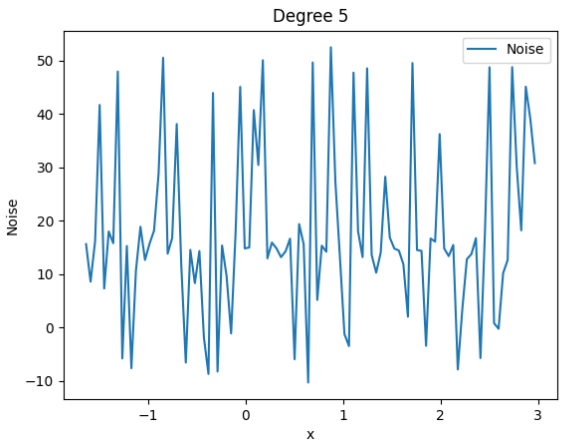




weights = [14.03560995 -9.92177074 20.79262217 -2.63825474 -6.13630812
1.68722588]

Batch_size = 25 (Right = x-tcurve,Left = Noise Curve)





weights = [13.299174   -10.23304328 22.49710843 -2.76355721 -6.71855062
1.83909216]

Here we see the difference between the curves after changing the degree and the batch_size if the curves and the MSE is less for test data form poly of degree at 10 if we further increase id decrease the degree we many be found more error then we can easily say that this the best pidicted model.

PART 2

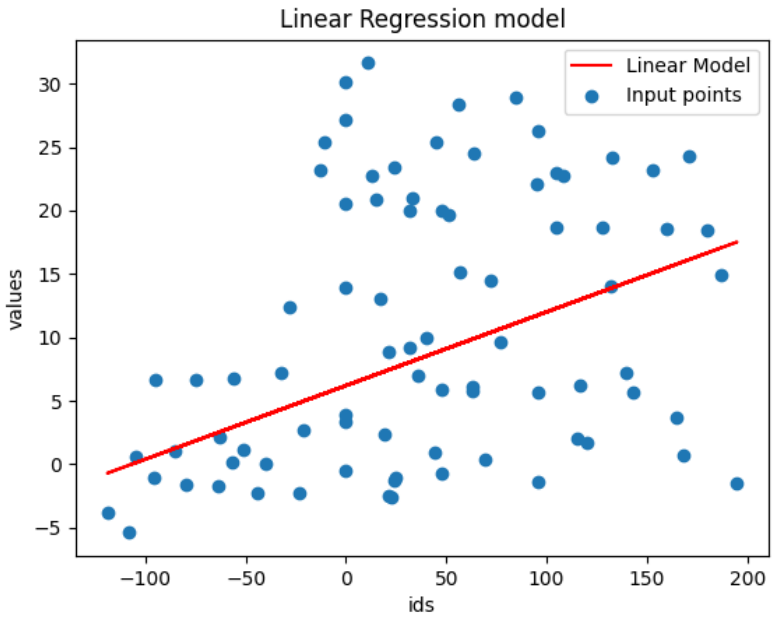
In this part we design a simiple linear regression model and read a csv file from then train our model by using that data and after that we need to test our data here the challenege is that the data we compare is given in US date format and values correspoing to them is the float type so for train our model we need to decode our date string into the some integer so that we predict our model and for that I used that data of month and year and train my model but I get some MSE for test data is 90.The equation used in this model is

$Y = ax + b$

Where $x = -month^2 + year^2$ and we need to find a and b

For that we use the formulae i.e $a = ((\sum xy)*(n) - (\sum x)(\sum y)) / ((n* \sum x^2) - ((\sum x)^2))$

$b = ((\sum y)*(\sum x^2) - (\sum x)(\sum xy)) / ((n* \sum x^2) - ((\sum x)^2))$



weight = [6.19737554277229, 0.05805022302820729]

MSE_Test = 85.72281565097448

Here we see that the MSE is high for this linear regression model this is because of date to int conversion is not efficient enough. But I compare to the testing the final test data is nice engh i.e the MSE for that is 90(approx.) on kaggle so if we design our model so

that the MSE_test for training data is less then obiousily the MSE for the final predicted data is nice enough.