CSI 5325 Assignment 4

Sadia Nasrin Tisha

21st April 2022

1 Question 01:

1.1 1

Given that,
Lagrangian formula for ridge regression,

$$Ridge : min[\frac{1}{n}\sum_{i=1}^{n}(h(x_i) - y_i)^2 + \frac{\lambda}{n}\sum_{i=1}^{d} w_i^2]$$

Here,
$h(x) = w^T x$
So,
$\frac{1}{n}\sum_{i=1}^{n}(h(x_i) - y_i)^2 + \frac{\lambda}{n}\sum_{i=1}^{d} w_i^2$
$=\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)^2 + \frac{\lambda}{n}\sum_{i=1}^{d} w_i^2$
$= \frac{1}{n}||wX - y||^2 + \frac{\lambda}{n}\sum_{i=1}^{d} w_i^2$
$=\frac{1}{n}(w^T X^T Xw - 2w^T X^T y + y^T y) + \frac{\lambda}{n} w^T w$
$=\frac{1}{n}(w^T X^T Xw - 2w^T X^T y - y^T y + \lambda w^T w)$

Let, $Q= \frac{1}{n}(w^T X^T Xw - 2w^T X^T y - y^T y + \lambda w^T w)$
$=\frac{d}{dw}Q=\frac{d}{dw}[\frac{1}{n}(w^T X^T Xw - 2w^T X^T y - y^T y + \lambda w^T w)]$
$=\frac{d}{dw}Q=\frac{1}{n}(2X^T Xw - 2X^T y + 2\lambda w)$
$=\frac{d}{dw}Q=\frac{2}{n}(X^T Xw - X^T y + \lambda w)$

To minimize the Lagrangian formula for ridge regression,
Let $\frac{2}{n}(X^T Xw - X^T y + \lambda w) = 0$
$\Rightarrow X^T Xw - X^T y + \lambda w = 0$
$\Rightarrow w(X^T X + \lambda I) - X^T y = 0$
$\Rightarrow w= (X^T X + \lambda I)^{-1} X^T y$

$$Hence, w = (X^T X + \lambda I)^{-1} X^T y$$

1.2 2

Generate Data: According to following function, I have randomly generated 100 data and split the data with 70% training sample and 30% testing samples. Figure 1 shows the scatter plot of x and y of 100 datapoints.

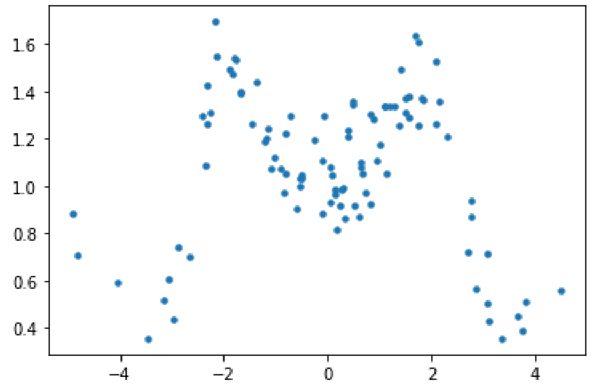

Figure 1: Scatter plot of x and y of 100 datapoints

Training: Here, I have applied the Ridge regression to fit polynomial of order 10. I have applied psudoinverse to train Ridge regression weighs. Figure2 shows the trained plot of Ridge regression where the value of $\lambda = 0.05$. Here by using regularization, it shows that, data are fit well and the data points are close to training sample and also ignored fitting the noise data.

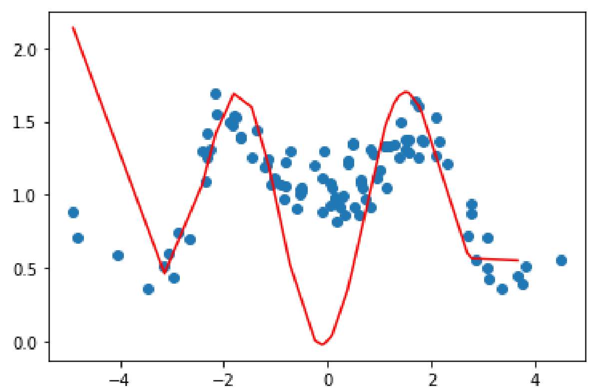

Figure 2: Training data

1.3 3

Here I have applied different value of λ to see the error. Figure3 shows the variation of in sample error and out sample error for different λ. The plot shows that, for $\lambda = 1.0$ it gives the lower E_{in} and E_{out}. It means that for $\lambda = 1.0$, the model fit well with minimum error. Also Figure 4 shows that, data are fit well and most of the data points are close to training sample for $\lambda = 1.0$.
On the other hand, by observing the errors, both E_{in} and E_{out} value is getting higher when we increase the

value of λ. It means, the model will get underfit if we use higher value of λ. It also shows that, for lower value of λ, the error was higher than $\lambda = 1.0$. It denotes that, the model will overfit if the value of λ is lower.

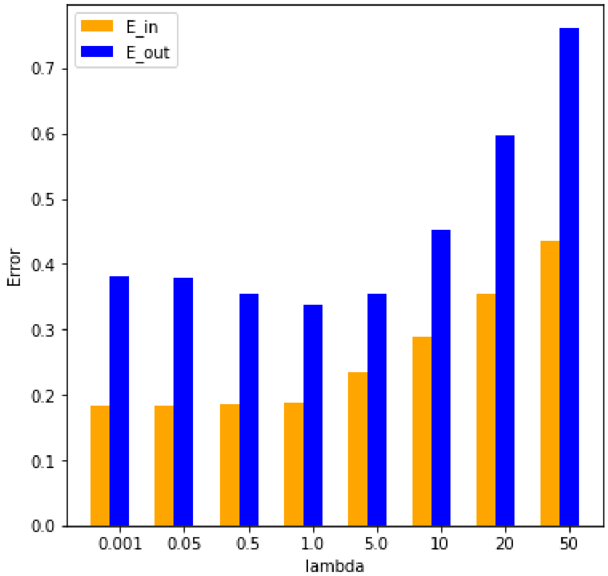

Figure 3: Plot of the in-sample and out-of-sample error for different functions of λ

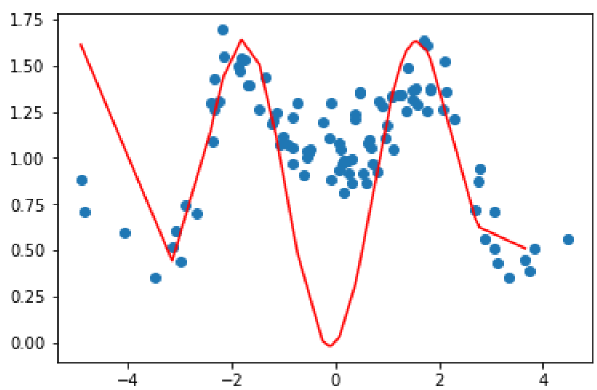

Figure 4: Scatter plot of Ridge regression with $lambda = 1.0$

In figure 5, it shows that, for some value of λ weight was more than zero or higher. So regularization fit that higher weight well. Also, for some λ the weight value became close to zero and that is why the error increased in higher lambda value. For example, for higher $\lambda = 10$ the weight is close to zero and also the error is higher .

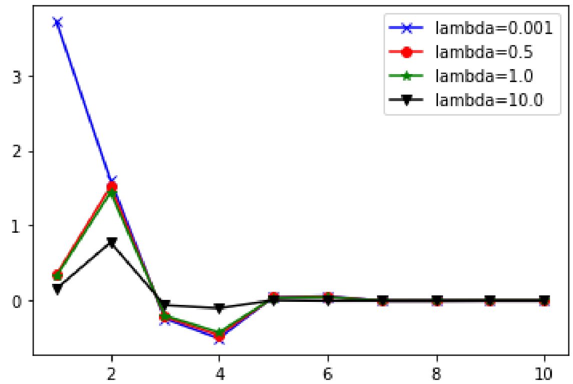

Figure 5: Plot of different weight value for lambda

2 Question 2

Non Parametric RBF: From question 1, I have taken the same 100 datapoint dataset and split the dataset like before and implemented non parametric RBF according to equation 6.1. I have implemented different value of r and measure the error and also observed the shape of the fit. From figure 6 it shows that when I increase the value of r, the error is getting higher. And the lowest error I got when r=0.2 which is 0.013. So from the plot, when r is < 0.1 the model overfit and when $r > 5$, the model underfit.
On the other hand, for observing the shape of fit, figure7 shows that when r=0.2 the red point fit the data well and when the value of r is more than 5 the fit of data is not well and that is why the error increased.

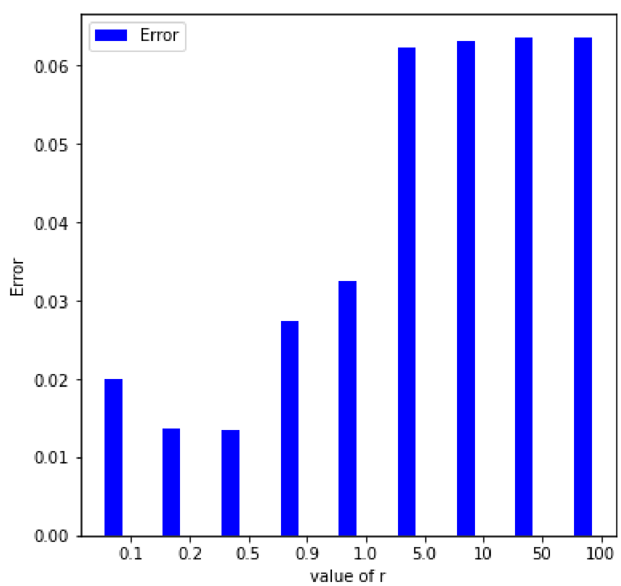

Figure 6: Plot of error for different r(Non Parametric RBF)

Parametric RBF: From question 1, I have taken the same 100 datapoint dataset and split the dataset like before and implemented parametric RBF according to equation 6.4. I have implemented different value of r and k and measure the error and also observe the shape of the fit. From Figure8 is shows that, when the k

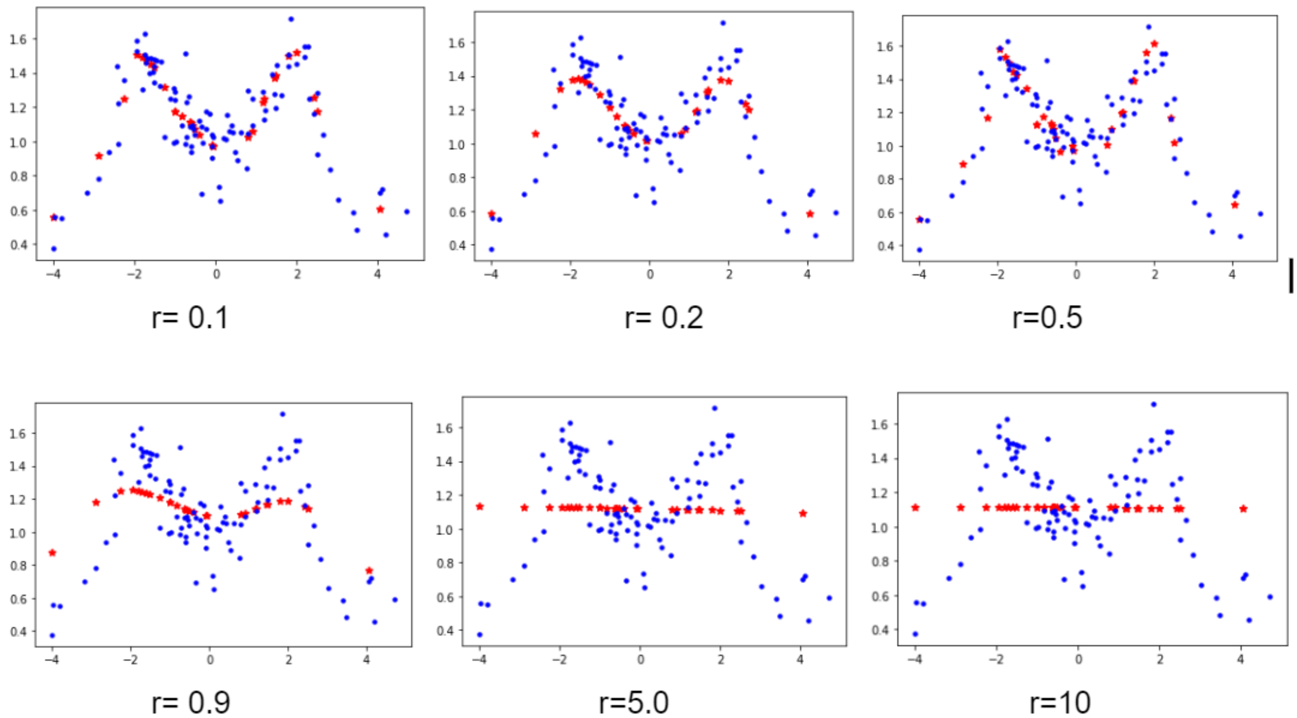

Figure 7: Plot of shape of fit for different r value(Non Parametric RBF)

value is 10, the error is minimum for 100 data points. According to rule of thumb for 100 data, $\sqrt{100} = 10$. In my data for k=10 I got the lowest error.
For implement different value of r, I applied different value of r with k=10. Figure9 shows the the error plot of different value of r. It shows that when the r=0.1 and 0.2, the error is higher that means for lower r overfitting occur and also for r=10,50,100, the error is higher where undefitting occurs. But in case of r= 0.5,0.9,1.0 the error is lower. The lowest error for r is 0.9. Here the rule of thumb for r is $(max(X_{train}) - min(X_{train})/k = [3.9433 - (-4.2525)]/10 = 0.85$, which is almost 0.9 and for r=0.9 I got the lowest error.
On the other hand, for observing the shape of fit, figure10 shows that when r=0.9 the red point fit the data well and when the value of r is more than 5 and less than 0.5, the fit of data is not well and that is why the error increased.

3 Question 3

(a) For N=500, data is randomly selected where in training set there is 500 data and for testing set there is 6791 data.

(b) By using the training and testing set, I have applied the KNN classifier, where, k=3. And calculate E_{in} and E_{out} after fitting the data. The Figure11 shows the E_{in}, E_{out} for KNN, where $E_{in} = 0.1$ and $E_{out} = 0.21$. Figure12 shows that, for 20 random train-test split, the variation of E_{in} and E_{out}.

(c) By using the training and testing set, I have applied the CNN algorithm to condense the data and

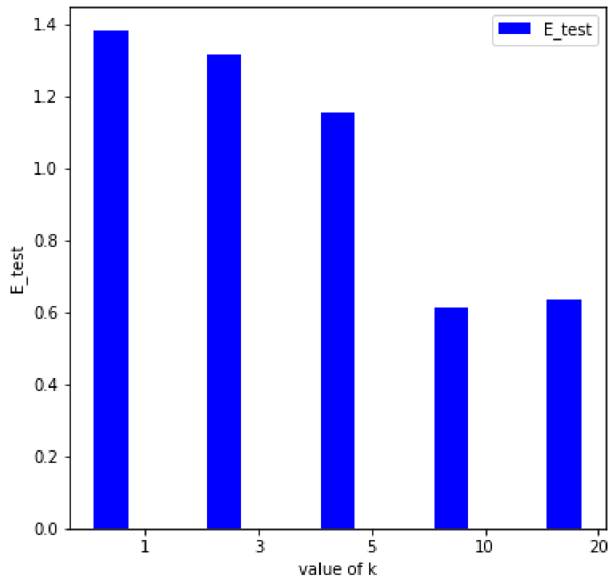

Figure 8: Plot of error for different k value(Parametric RBF)

calculate E_{in} and E_{out} after fitting the data. I have applied 20 random train-test split. The Figure13 shows the E_{in}, E_{out} for CNN, where $E_{in} = 0.07$ and $E_{out} = 0.22$, which is better than KNN as CNN is a condense data . Figure14 shows that, for 20 random train-test split, the variation of E_{in} and E_{out}.

(d) Now I have done 1000 random training-test splits for KNN and CNN algorithm. The observation is given below:-

KNN:
Minimum E_{in}= 0.0, Maximum E_{in}= 0.25, AverageE_{in}= 0.159
Minimum E_{out}= 0.2, Maximum E_{in}= 0.28, AverageE_{in}=0.230

CNN:
Minimum E_{in}= 0.0, Maximum E_{in}= 0.27, AverageE_{in}= 0.169
Minimum E_{out}= 0.2, Maximum E_{in}= 0.233, AverageE_{in}=0.233

Here from the above data, it shows that both full and condense data have almost same amount of error in both E_{in} and E_{out}.

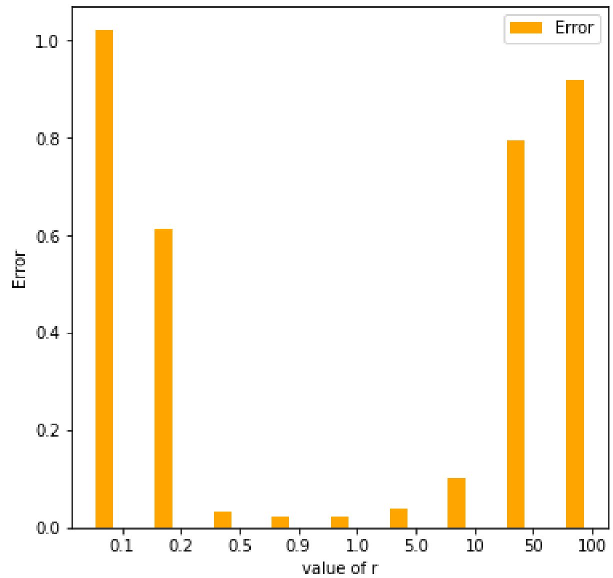

Figure 9: Plot of error for different r value(Parametric RBF)

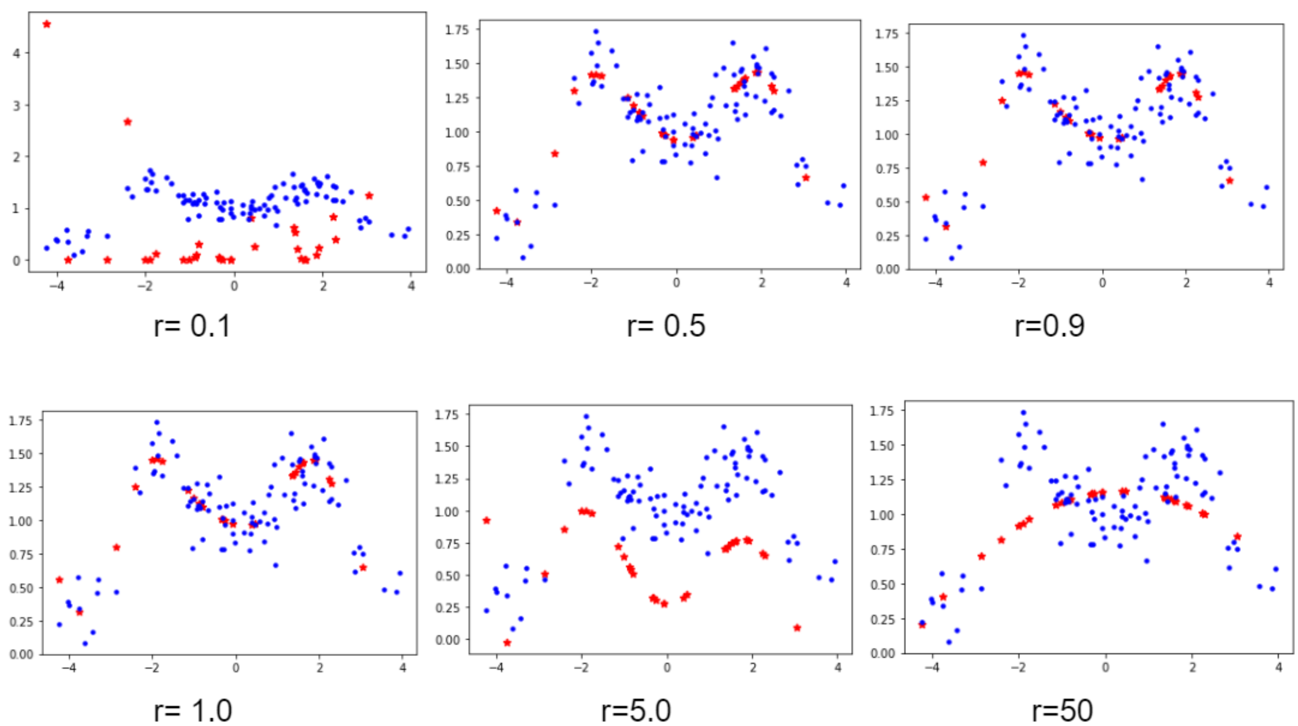

Figure 10: Plot of shape of fit for different r value(Parametric RBF)

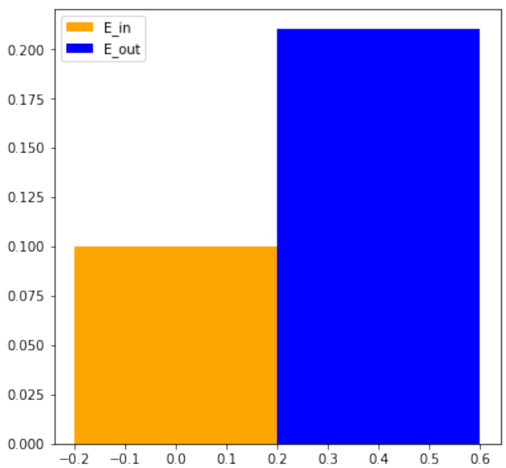

Figure 11: In sample error and out sample error of KNN

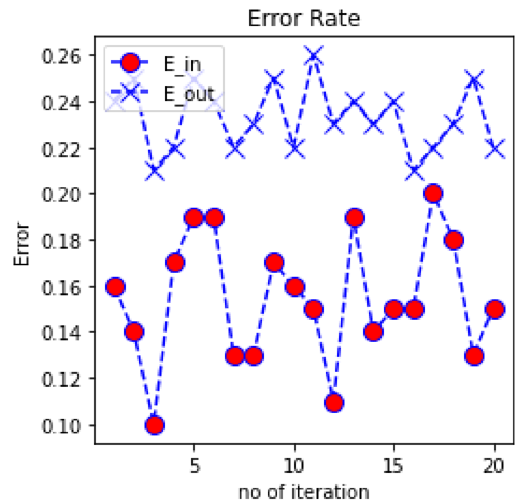

Figure 12: In sample error and out sample error of KNN for 20 random train-test split

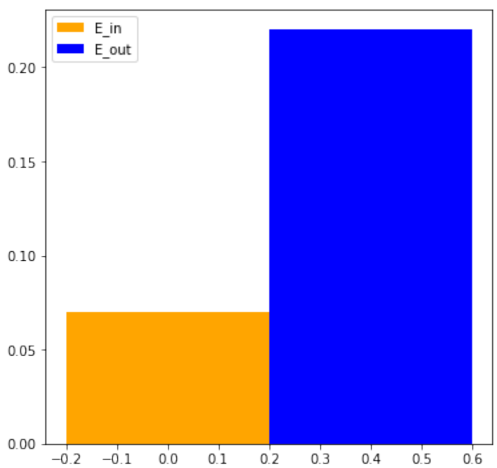

Figure 13: In sample error and out sample error of CNN

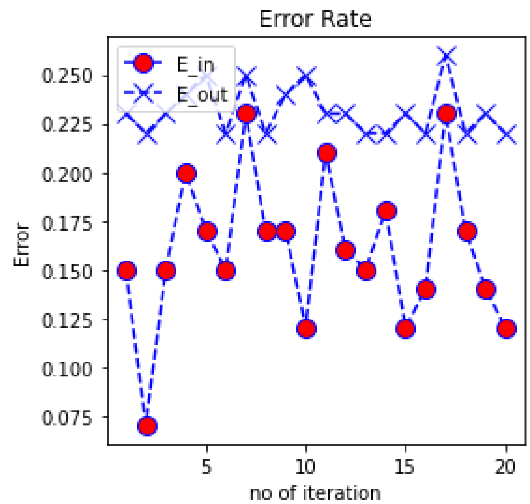

Figure 14: In sample error and out sample error of CNN for 20 random train-test split