Machine Learning - hw1

Huiting Hong A061610

October 19, 2017

Abstract

This assignment is going to solve 3 mathematical problems and 2 programming problems related to polynomial fitting and polynomial regression. This report will focus more on discussing the problem while implementation details please go to my github repository[1] to see more detail.

1 Jensens's Inequality

Please see the hw1-1.pdf file for derivation detail.

2 Entropy of the univariate Gaussian

Please see the hw1-2.pdf file for derivation detail.

3 KL divergence between two Gaussians

Please see the hw1-1.pdf file for evaluation detail.

4 Polynomial Curve Fitting

We are going to write a program to implement linear regression, since the data in this problem is one dimension, we write the polynomial function as :

$$y(x,w) = w_0 + w_1 x + w_2 x^2 + ... + w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{1}$$

where the x is the input data $x_1, x_2, ..., x_{20}$. After minimizing the error function base on the batch-learning[2] solution, we can see the result of RMS error versus different M as follows:

which the error function here is non-regularized :

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}(y(x_n,w) - t_n)^2 \tag{2}$$

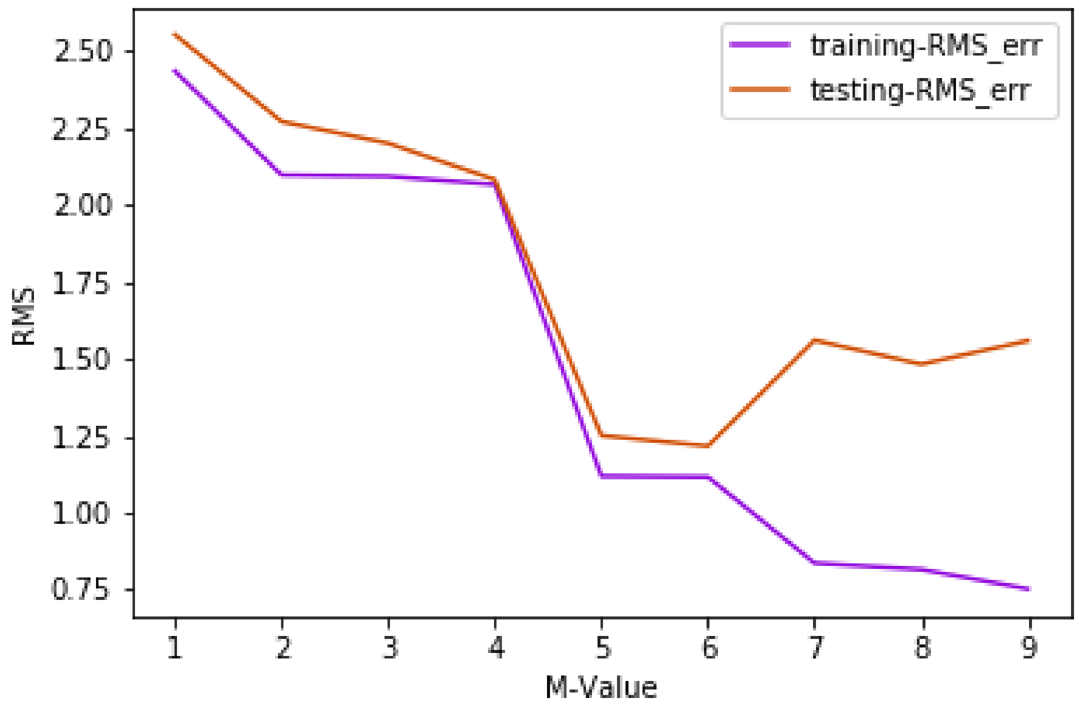

Figure 1: The RMS error of training and testing set depends on different order, M. We can see that the error decreases when the order increases. However when order goes too high, it will cause overfitting problem which RMS error on testing set will suddenly bump up.

4.1 Overfitting Problem Discussion

Here we can discuss on the trade-off between the model-complexity and the prediction error, which is known as Bias-Variance-dilemma[3]. The bias means the error between ground truth and prediction; the variance means the model complexity.

The model here means the y(x,w) function mentioned above, when M increases, the function will look more oscillating, and so it is more possible to fit better on the data. However, when M goes too high, it means our model oscillates too strong, and that is the point we can't fit well on the testing data. (which is called overfitting problem.)

Therefore, we want to find a proper M that makes the model complex enough while keep the error on testing data also small. From the Figure1 we can know that when M goes to 7 or larger, it will cause the overfitting problem.

4.2 Regularized Error function

Now we consider the regularized error function:

$$E(w) = \frac{1}{2}\sum_{n=1}^{N} y(x_n, w) - {t_n}^2 + \frac{\lambda}{2}||w||^2 \tag{3}$$

where

$$||w||^2 = w^T W = w_0^2 + w_1^2 + w_2^2 + ... + w_M^2 \tag{4}$$

Here we also use the batch-learning[2] approach on solving the best W, while the solution turn out to be:

$$W^* = (\phi^T \phi + \lambda I)^{-1} \phi^T t \tag{5}$$

where

$$\phi = \begin{vmatrix} \vec{x_1}^0 & \vec{x_1}^1 & \vec{x_1}^2 ... \vec{x_1}^M \\ \vec{x_2}^0 & \vec{x_2}^1 & \vec{x_2}^2 ... \vec{x_2}^M \\ ... & & \\ \vec{x_N}^0 & \vec{x_N}^1 & \vec{x_N}^2 ... \vec{x_N}^M \end{vmatrix} \tag{6}$$

$\vec{x}$ means that it could be a vector form, but in this problem it is a scalar since it is just one dimension, and t is our target data.

The result of RMS error on training and testing set are as follows:

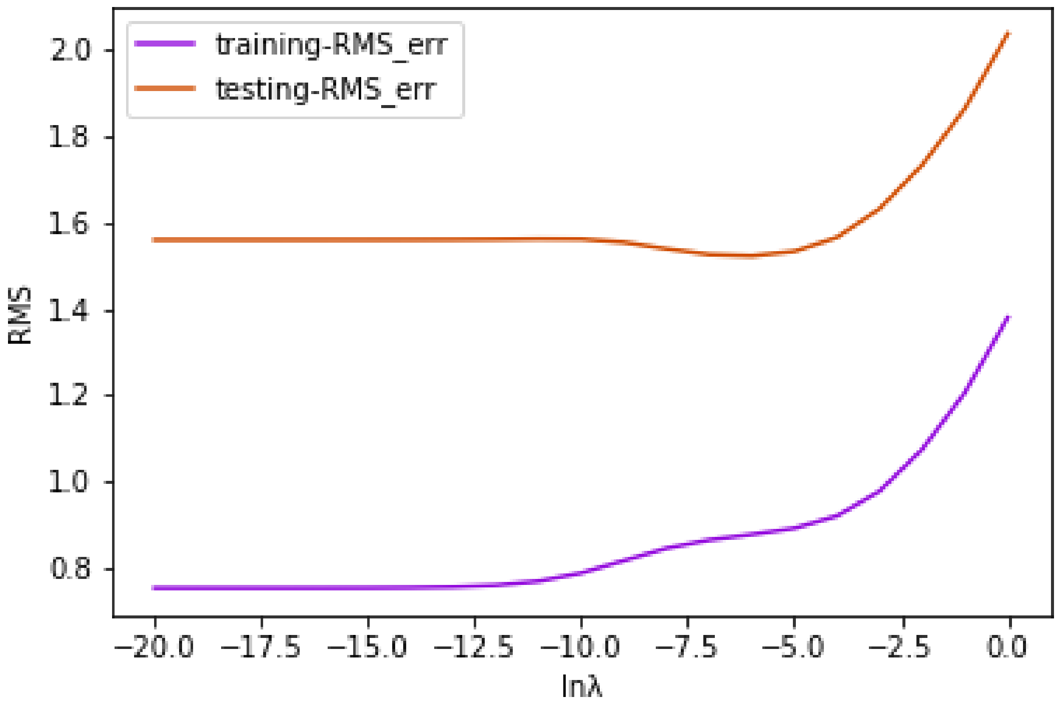

Figure 2: RMS error of training and testing set depends on different λ. We fix M $=$ 9, and set the regularized parameter, λ, from e^{-20} to e^0

4.3 Regularized Error function Discussion

As λ goes higher, the value of W will become smaller, which means that the function y(x,w) will turn to more smooth instead of oscillating. Since we set order M $=$ 9, we know that there is overfitting problem happens here as shown in Figure1. Now, when our polynomial function turns to be more smooth, it is possible to solve the overfitting problem, which λ at some point should let the RMS error of testing set to decrease.

From Figure2, we can see that RMS error between $ln\lambda$ = -10 to $ln\lambda$ = -5 decrease a little bit. We believe that is the proper range for λ.

When λ goes even higher($ln\lambda >$ -2.5), the RMS error increase, which means that it lets the polynomial function turn to be too smooth to fit the data, and so the error increases.

5 Polynomial Regression

Here, we wrote a polynomial regression program to estimate the class from Iris data set provided by TA.

5.1 Apply non-regularized error function

Although the data x now turn to be 4 dimensions, the solution of best W still remain the same:

$$W^* = (\phi^T \phi)^{-1} \phi^T t \tag{7}$$

Here we fix M=1 and M=2 and we can see the result of RMS error as below:

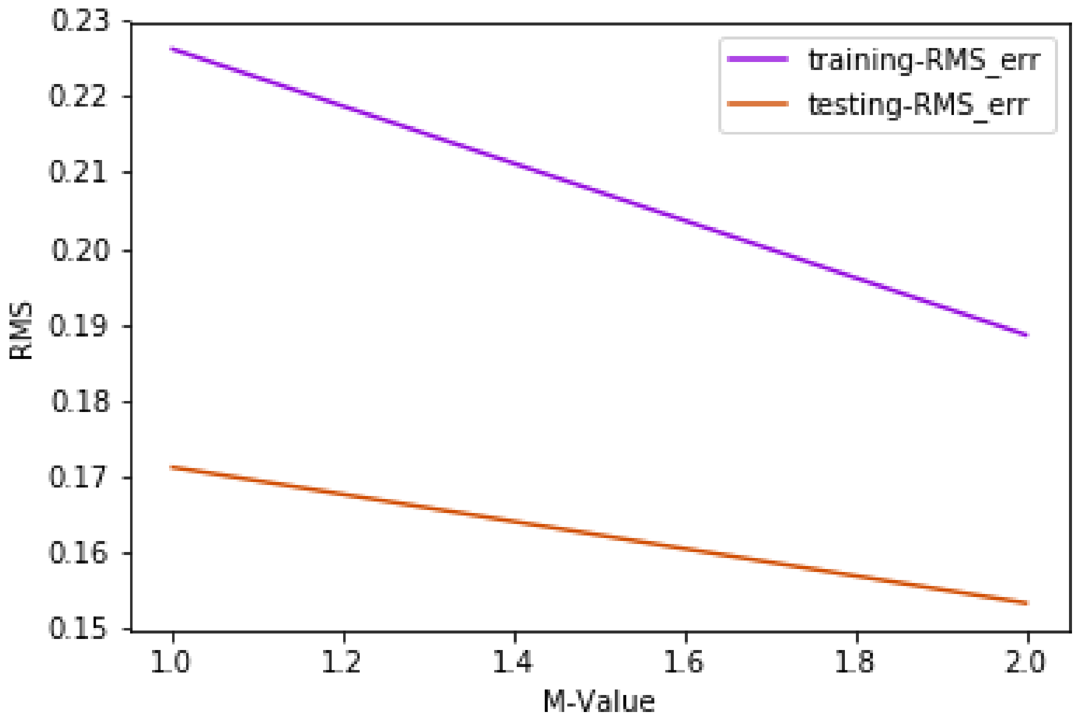

Figure 3: RMS error of training and testing set depends on M = 1 and M = 2.

The ϕ here is the same as equation(6) while the x turns to be 4 dimensions vector, and the way on creating the ϕ matrix is a little bit tricky. I use recursive way to create the ϕ, which is more general and more neat, the more detail please go to my github repository[1] to see the code.

5.2 Select the most contributive attribute

To figure out which dimension's information is more contributive, we simply remove 0nd, 1st, 2nd, 3rd dimension one by one and apply the remaining 3-dimension data to calculate its RMS error on training data.

The result is as follows:

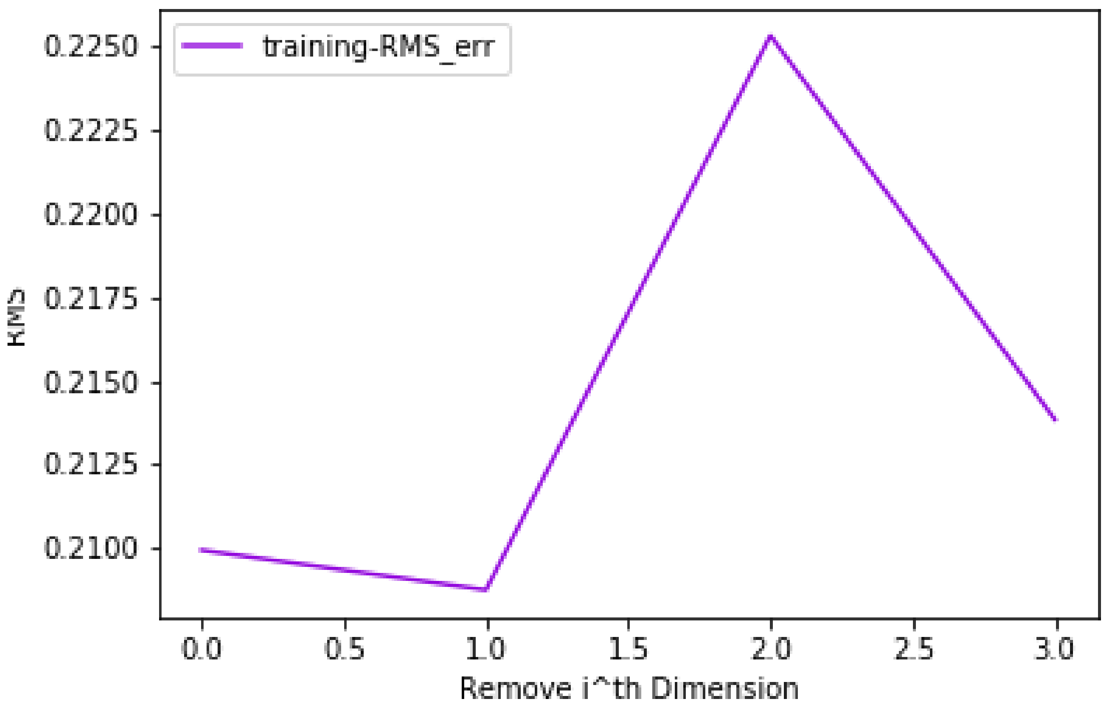

Figure 4: RMS error of training set depends on removing different dimension.

We can see that from the Figure4, when we remove the 2nd dimension(start from 0th), the RMS error result is the worst, which means that the information of the 2nd dimension is pretty important. Without 2nd dimension information we can't achieve good enough RMS error.

References

[1] `https://github.com/w102060018w/2017_MachineLearning_hw1`.

[2] *Batch learning and Online learning* `https://en.wikipedia.org/wiki/Online_machine_learnin`

[3] *Bias-Variance-dilemma* `http://scott.fortmann-roe.com/docs/BiasVariance.html`.