Assignment 2

Parameter estimation and response predictions related to multivariate linear functions

2018-09-27

Kristina Benevides

Automated predictive modelling 1MB516, 5.0 c

Innehåll

Abstract 3
1. Introduction 3
2. Results 4
2.1 Task 2a 4
2.1.1 i. 4
2.1.2 ii 5
2.2 Task 2b 5
2.3 Task 3b 7
2.4 Task 3c 8
3. Discussion and conclusions 9
4. Self-reflections 10

Abstract

In this homework we use three different multivariate models to generate noisy data and to estimate their weight vectors and predict response values. We use ridge regression to estimate the parameters and also perform double k-fold cross validation to estimate the average performance of this linear regression method. There is also an attempt to use maximum likelihood which is equivalent to ordinary least squares solution in this case to estimate the parameters and an explanation to why that did not work. The results indicate that ridge regression works well to estimate these models weight vectors.

1. Introduction

In this assignment we used machine learning methods to build linear prediction models. The three multivariate linear functions used was the following written as (i = 1,2,3):

$$y_i(n) = w_i^T x_n + \varepsilon_n,$$

where w_i is a 2000-dimensional parameter vector and ε_n is normal distributed noise. For the first model (i = 1) the weight vector is:

$$w_{1j} = \begin{cases} \frac{1}{500}, & if\ j \leq 500 \\ 0, & otherwise \end{cases}$$

For the second model (i = 2):

$$w_{2j} = \begin{cases} \frac{1}{10}, & if\ j \leq 10 \\ 0, & otherwise \end{cases}$$

For the third model (i = 3), the weights follow an exponential function that:

$$w_{3j} = 2^{-j}$$

From each of these three models 90 training examples were generated by using a 2000-dimensional column vector x_n containing values uniformly distributed on the interval [-1,1]. The generated column vectors x_n were then stored as rows in a 90 x 2000-dimensional matrix and for each row in this matrix observations were generated as: $y_{obs,i} = X_n w_i + \varepsilon$, where ε is zero mean normal distributed random error. The standard deviations for the noise term varied between models; in order to get standard deviation around 20% of the standard deviation for the noise free term y-values the following was suggested: $\sigma_1 = 0.005$, $\sigma_2 = 0.04$ and $\sigma_3 = 0.07$ for respective model.

We calculate the loss of our models with root mean square error (RMSE).

2. Results

The results for the different task are summarized below.

2.1 Task 2a

2.1.1 i.

In Bayesian inference we have Bayes' rule:

$$P(w \mid D) = \frac{P(D \mid w)\, P(w)}{P(D)},$$

that derives the posterior probability, *P(w|D),* from two antecedents: the prior probability, *P(w)*, and the likelihood function *P(D|w)*. *P(D)* is just the model evidence and is the same for all w. From a perspective of Bayesian inference, Maximum likelihood (ML) is a special case of maximum a posterior estimation which assumes uniform prior distribution of the parameters. In our case when we have errors as follows $\varepsilon_n = y_{obs,i} - X_n w_i$ and $\varepsilon_n \sim N(0, \sigma^2)$, the likelihood function can be written as:

$$L = P(D \mid w) = \prod_{i=1}^{2000} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(y_n - w_i^T x_n)^2 \frac{1}{\sigma^2}},$$

and we regard the *P(w)/ P(D)* factor in the equation as constant. In this case, we have 2000 observations which we assume are independent of one another hence we take the product of all samples. Thus, we want to minimize the likelihood function above by setting the partial derivatives to zero. For practicality, we take the natural logarithm of the likelihood function, resulting in;

$$N log \frac{1}{\sigma\sqrt{2\pi}} - \frac{1}{2\sigma^2} \sum_{i=1}^{2000} (y_n - w_i^T x_n)^2,$$

which gives us the derivative;

$$\frac{\partial L}{\partial w} = \frac{2}{2\sigma^2} \sum_{i=1}^{2000} x_n (y_n - w_i^T x_n).$$

If we set the above derivative to zero, we see that we can discard the factor in front of the summation operator and are left with;

$$\sum_{i=1}^{2000} x_n (y_n - w_i^T x_n) = 0 \Leftrightarrow \sum_{i=1}^{2000} x_i \varepsilon_i = 0 .$$

This result is equivalent to the first order condition in ordinary least squares (OLS) regression.

The ML estimation, i.e. the OLS regression, proved to be useless in the original case when we used 90 training examples and had weight vectors with 2000 coefficients. This problem is ill-posed due to the fact that the number of examples is less than the number of unknown coefficients, meaning that the training examples does not provide enough

information to safely specify the unknown parameters in the model. In other terms, the rank of matrix X will be less than the number of parameters, so the inverse of X^TX does not exist, which results in the OLS estimate may no be unique.

2.1.2 ii.

One advantage of using ridge regression (RR) is that it does work when we have ill-posed problems as the one described above. Since we are adding the penalty term λ along the diagonals of X^TX which makes the matrix $X^TX + \lambda I$ invertible. In this assignment we used following reformulation of the RR solution:

$$wRR = (X^TX + \frac{\lambda}{1-\lambda} I)^{-1} X^T y,$$

where X is the data matrix with examples as rows and we have restricted the penalty parameter to the interval [0,1]. The variable y is our observed Y matrix with values for each model stored as rows.

2.2 Task 2b

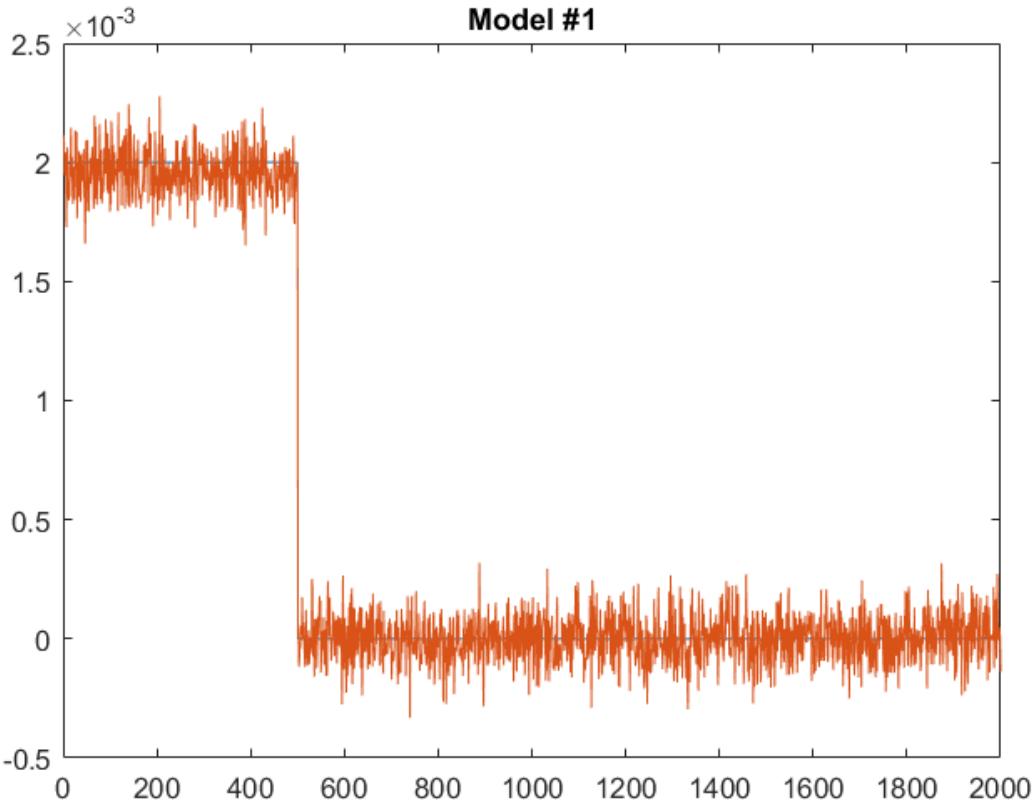

Figure 1. The resulting $w_{1,RR}$-vector (orange) compared to the true w_1 vector (blue).

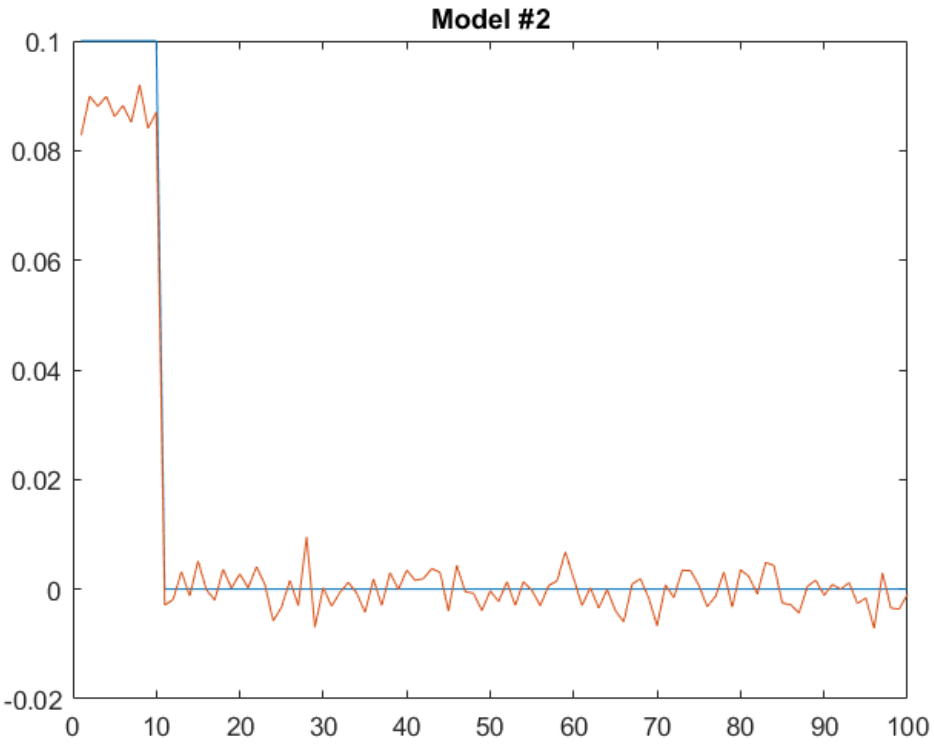

Figure 2. The resulting $w_{2,RR}$ -vector (orange) compared to the true w_2 vector (blue) for the first 100 parameters.

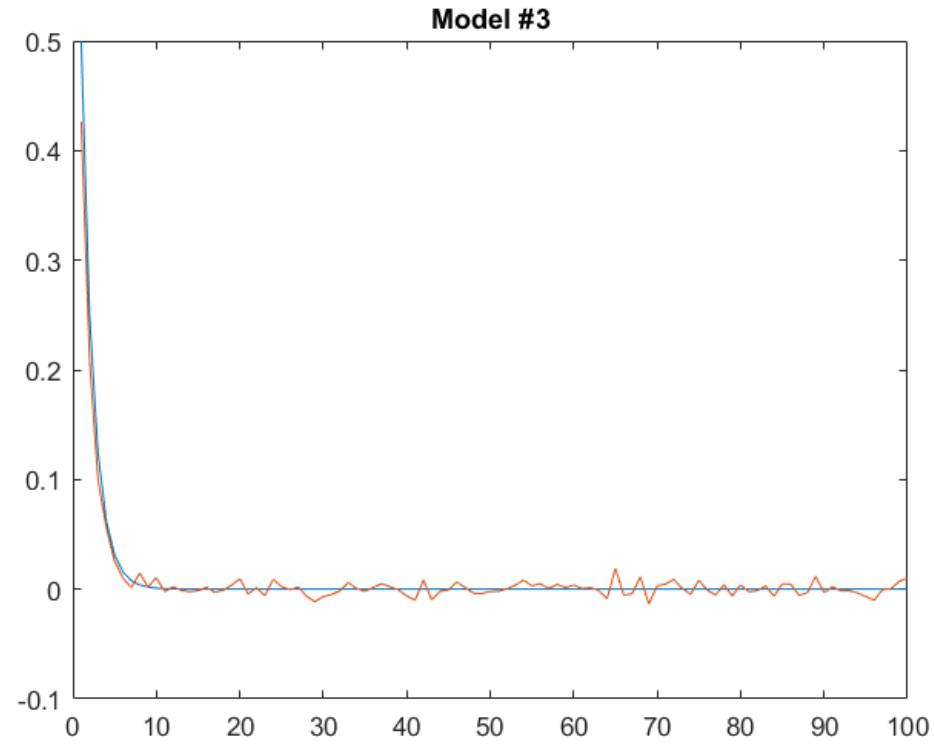

Figure 3. The resulting $w_{3,RR}$ -vector (orange) compared to the true w_3 vector (blue) for the first 100 parameters.

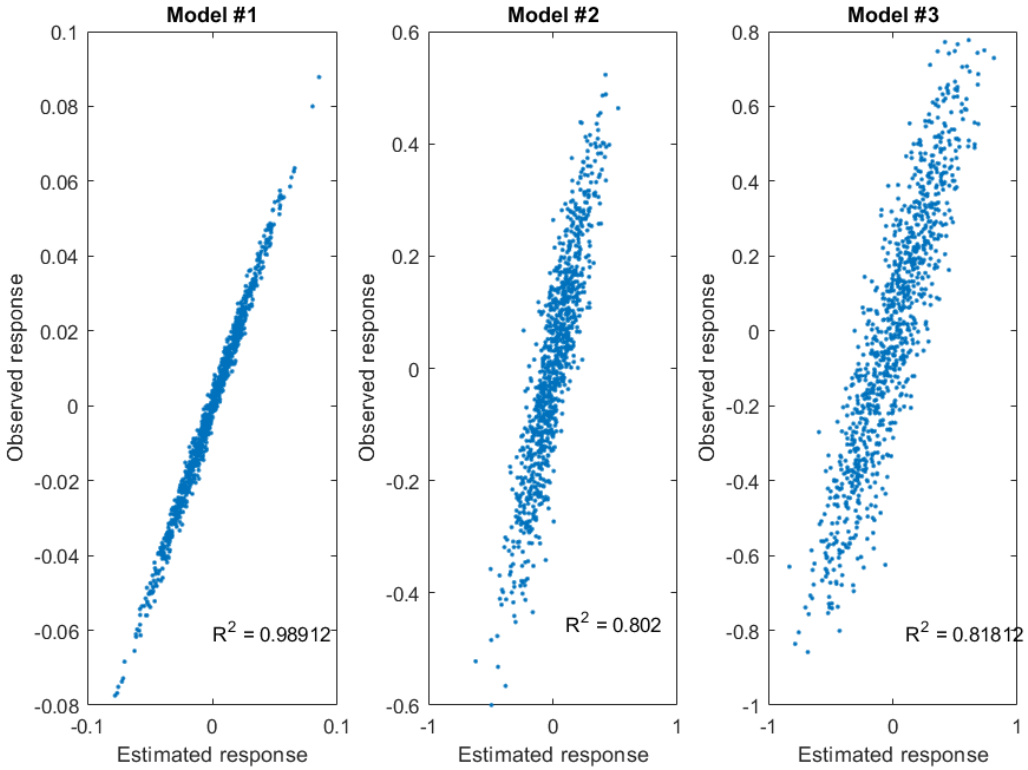

Figure 4. Scatterplot over estimated response and observed response for the three models, generated with 2000 training examples.

2.3 Task 3b

In Task 3 we only used the first model (i=1).

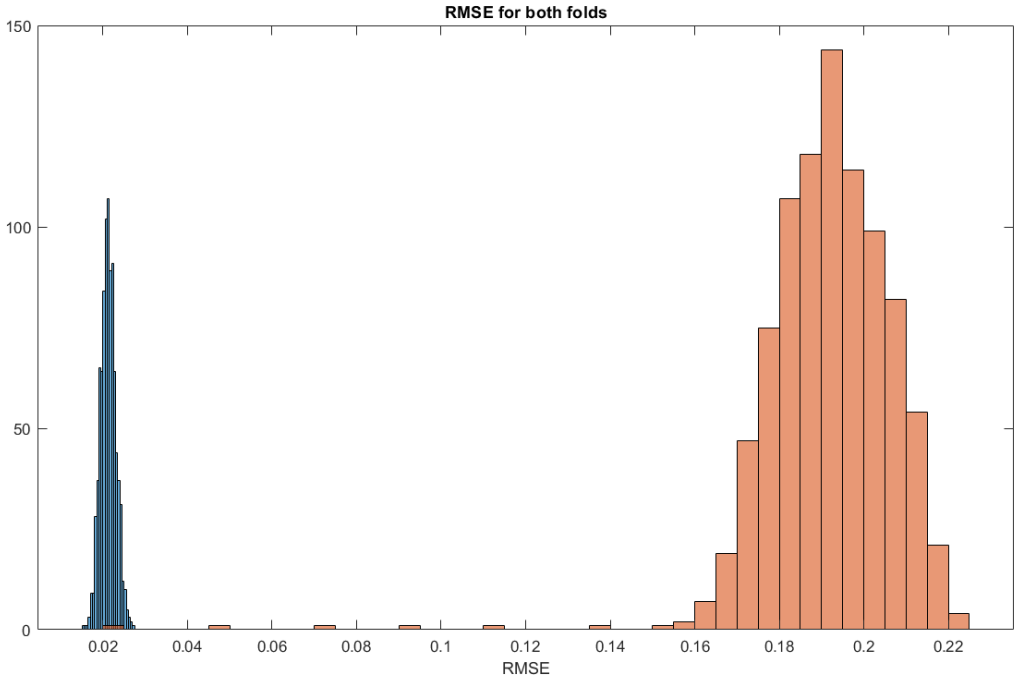

Figure 5. Average RMSE for inner fold (to the left) and RMSE for outer fold (to the right), generated with 90 training examples.

2.4 Task 3c

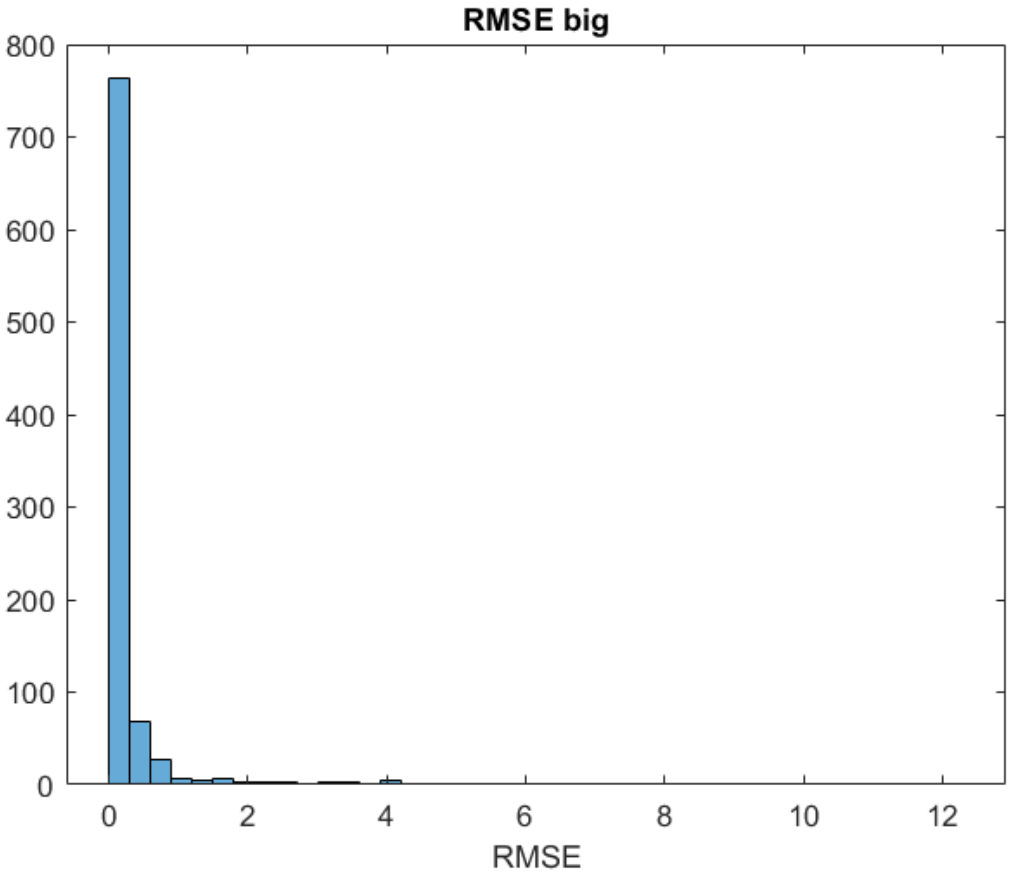

Figure 6. RMSE for $w_{opt,RR}$ tested with the big external data set.

3. Discussion and conclusions

In Task 2b the estimates for all three models mostly agree well with the true values (Figure 1,2 and 3). For model 2 and 3 there is a quite bad fit for the first few variables, probably because the matrix X^TX is singular but Matlab gives us inaccurate results anyways. For model 1 you can clearly see that variables with indices above 500 are very small compared to the ones with indices below 500. It is also possible to conclude that all parameters with indices below 500 are equally important for prediction of y, which is what we expect seeing as they all have the same weight. The standard deviation used in the noise terms were important for the results here. If standard deviation of the noise was around 20% of the standard deviation for the noise free term for the first model, you could not clearly distinguish between the first 500 variables and the rest. Therefore, I lowered the standard deviation for this model to get a better result.

In Figure 4 we can see that first model perform better compared with the other two in predicting response values ($R^2 > 0.98$). The second and third model are not terrible but still have quite low R^2-values ($R^2 > 0.80$). This is probably due to the higher standard deviations for the noise terms for these models compared to the first one.

The results in Task 3b indicate that there is a difference between the RMSE values generated in the outer loop and average RMSE values from the inner loop in terms of mean and variability. As we can see in Figure 5 the mean for RMSEavg_inner is much smaller on average and have lower variability than RMSE_outer which as expected leads us to conclude that values of RMSEavg_inner are biased. This is because we take the average of the inner RMSE, so it smooths out the variability we see in the RMSE_outer.

Figure 6 shows the result from Task 3c where we see much less variability and a smaller mean compared to in Task 3b. However, from this result I can't really conclude that the values in RMSE_outer are unbiased, but rather the fact that RMSEavg_inner are biased. I suppose this is because the optimal weight vectors are generated with the optimal λ from the inner loop. To further investigate this, I would error check the code, repeat the run and perhaps change some parameters, for example increase training examples.

4. Self-reflections

I found this practical useful because I got to review the concepts of linear regression like ridge regression and ordinary least squares regression. I appreciated that a reformulation of the ridge regression solution was provided and guide lines for which λ to test for. Including a part were got estimate the performance for one of the regression methods was valuable since you always must do this. Although our approach, using double cross validation to estimate the performance of the ridge regression method proved to be quite costly in time. I also thought it was good to include the proof that maximum likelihood is the same as the OLS solution here, that gave me a better understanding of why that is the case. Even though it might not have been intended, this assignment also refreshed my memory of when we have an ill-posed problem and OLS does not work. This assignment had an appropriate amount of work-load, but it took a long time to run the script when we used double cross validation. One suggestion for improvement is to make suggestions in the instructions for how one can speed up the running time of one's script, for example decrease the number of dimensions or use something else than ridge regression solution which must calculate the inverse of large matrices and is very time consuming.