Supervized Learning: Assignment 1

Thomas Nedelec
MSc Machine Learning
thomas.nedelec.15@ucl.ac.uk

Michal Daniluk
MSc Machine Learning
michal.daniluk.15@ucl.ac.uk

1 Exercise 1: Least Square Regression: effect of the training set size

We generated a noisy random data set, containing 600 samples, as $y_i = x_i' w + n_i$, where each x_i and n_i are drawn from the standard normal distribution. We splitted the data into a training set of size 100 a test set of size 500. We compute the mean squared error on both the training and test sets. Figure 1 shows examples of different training sets and estimated regression w.

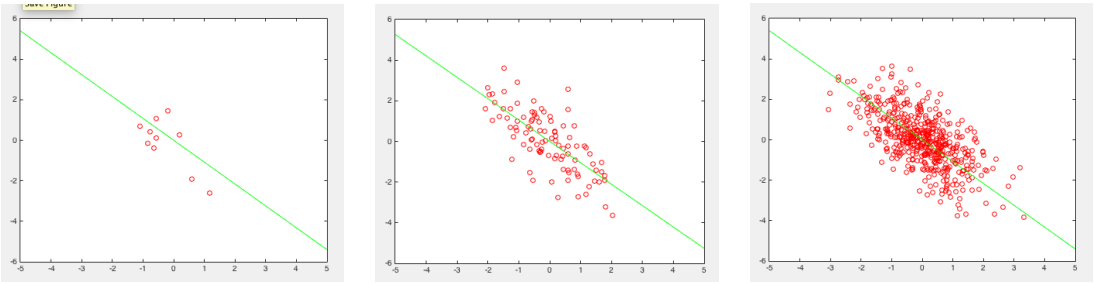


Figure 1: (a) training set of 10 points (b) training set of 100 points (c) the entire data set

Figure 2 illustrates the "2 x 2" table of averages mean square error for both training and test sets, on both 10 and 100 element-size training set.

We can observe that increasing the size of training set lets to decrease the average mean error on the test set. It's because, the larger training set prevents us from overfitting our model and give us a better overview of data. However, it increased the average mean error on train set. With increasing the size of training set, the average error on the test and train set became similar. The average error on test set is larger than on train set. It is clear, because we have seen the train data before and haven't seen the test data.

2 Exercise 2: Least Square Regression: effect of dimensionality

We repeated the task in exercise 1, but with 10-dimensional data sets. Figure 3 illustrates the "2 x 2" table of averages mean square error for both training and test sets, on both 10 and 100 element-size training set.

	Train	Test
10	0.89	1.12
100	1.00	1.02

Figure 2: Average mean square error for both training and test sets, on both 10 and 100 element-size training set.

	Train	Test
10	0.0	2505.1
100	0.9	13.6

Figure 3: Average mean square error for 10-dimensional data for both training and test sets, on both 10 and 100 element-size training set.

For 10-sample training set, we obtain the average mean square error 0 for train set and 2505.1 for test set. We have 10-dimensional data and 10 samples, so we can accurately predict y_i without any errors. However, the prediction is overfitted to the training data and we have a large error for test set. Increasing number of samples to 100 helps to decrease average error on test set. Average error for train set is bigger than for 10 samples, but our goal is to decrease test error.

3 Ridge regression

The regularization is a way to reduce the freedom of the classifier in order to improve the generalization and reach a better test set average error.
To implement ridge regression, we would like to minimize according to w the cost function,

$$\gamma w^T w + \frac{1}{l}\sum_{i=1}^{l}(x_i^T w - y_i)^2. \tag{1}$$

Using the notation $X = (x_1, x_2, ..., x_l)^T$, a matrix containing the training sample vectors as its rows, we can rewrite the cost function as:

$$\gamma w^T w + \frac{1}{l}Tr((Xw - Y)^T(Xw - Y)) \tag{2}$$

Taking derivative according to w, we reach:

$$2\gamma w + 2\frac{1}{l}(X^T X w - 2Y^T X) = 0 \tag{3}$$

To reach the conditions for optimality, we set the previous equation to zero and we reach:

$$w = (\gamma l Id + X^T X)^{-1} Y^T X \tag{4}$$

because if $\gamma * l \neq 0$ $(\gamma l Id + X^T X)$ is non-singular.

$\gamma l Id + X^T X$ is symetric.

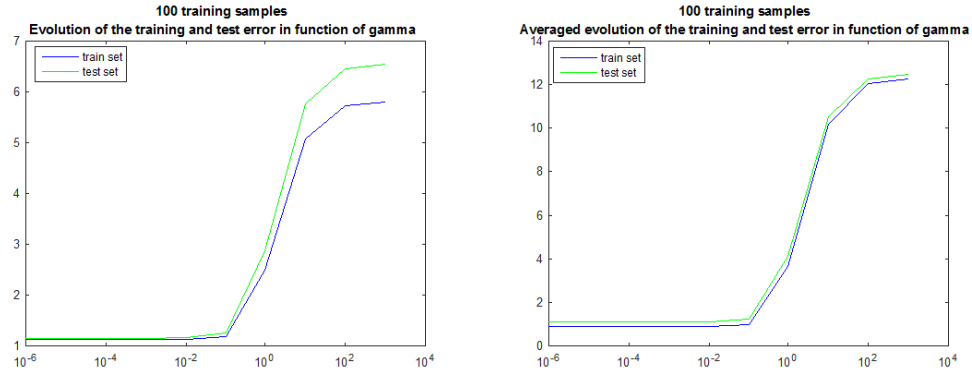
We consider $u \in \mathbb{R}^d$:

$$\begin{aligned}\langle(\gamma l Id + X^T X)u, u\rangle &= \langle \gamma l u, u\rangle + \langle X^T X, u\rangle \\ &= \gamma l \langle u, u\rangle + \langle Xu, Xu\rangle \\ &= \gamma l ||u||^2 + ||Xu||^2 > 0 \; for \; u \neq 0\end{aligned}$$

Thus $\gamma l Id + X^T X$ is definite positve.

4 Effect of the regularisation parameter

In this exercise we perform ridge regression on the data sets with the regularisation parameter ranging from 10^{-6} up to 10^3. First, the graph 4 shows the training error and test error in function of gamma for 100 training samples. If gamma is too high, the training error and test error increase dramatically. It is intuitive because when gamma is high the algorithm is far more likely to minimize $||w||$ rather than the training error.



(a) One run.

(b) Averaged on 200 runs.

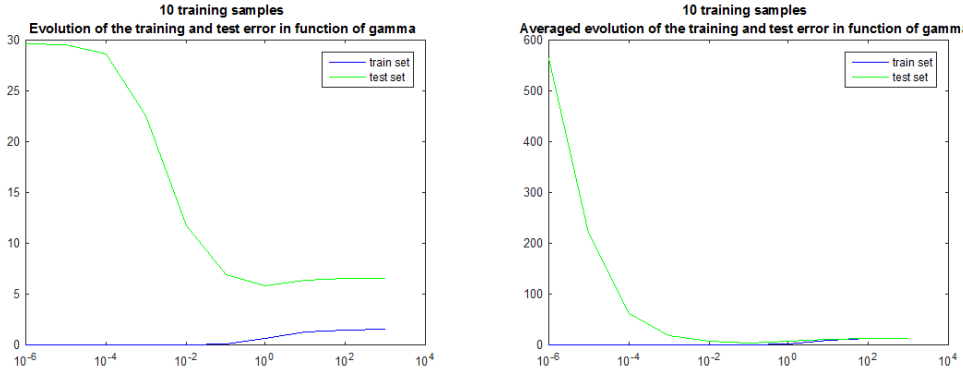
Figure 4: Evaluation of the training error and test error in function of gamma for 100 training samples.

We have an optimal point corresponding to inflection point of the curve in order to select γ. Nevertheless, the training set error is not a sufficiently good guidance to select the regularization parameter because the charts are quite different.

However, when we average the training error on 200 runs, we can observe that now the test error and the training error are now almost identical. A way to select the optimal γ would be to plot

the error on the training set averaged on a certain number of runs of the algorithm and select the optimal γ on this curve.

Figure 5 shows the training error and test error in function of gamma for 10 training samples. It



(a) One run.

(b) Averaged on 200 runs.

Figure 5: Evaluation of the training error and test error in function of gamma for 10 training samples.

looks different than for 100 training samples. Regularization prevented model from overfitting. The test error decreased with increasing regularization parameter until it reached the minimal value for $\gamma = 10^{-1}$.

Regularization prevents algorithm to overfit to data set. However, it is difficult to choose the optimal value of regularization parameter γ by minimizing the training error, because it's almost always have the minimum error for $\gamma = 0$. We don't have access to the test set, so we can only use information about error on training set. In our opinion, we can find optimal value of γ by looking the biggest value of γ, which doesn't dramatically increase training error.

This exercise also shows that it is easier to overfit model for smaller data set than for bigger one and the regularization becomes less important for big data sets.

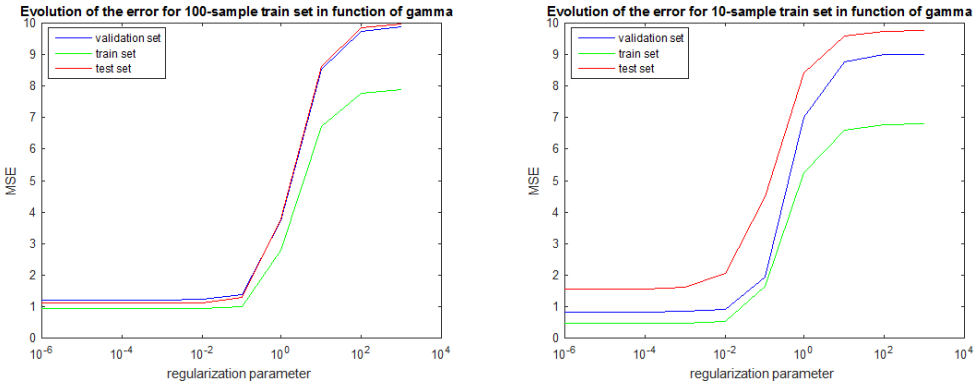
5 Tuning the regularization parameter using a validation set

The goal of this exercise is to use a validation set to tune the regularization parameter for training set with 10 and 100 samples. Figure 6 shows the result error on training, validation and test sets for both data sets.

For 100 sample data set, the averaged validation error is almost identical as averaged test error, so it is a good approximation of test set.

The averaged γ for 200 experiments are : $\gamma^{(100)} = 0.0417$ and $\gamma^{(10)} = 50.1787$. The averaged $\gamma^{(10)}$ is much bigger than $\gamma^{(100)}$. It is because the size of training set is very small and it's very prone to overfitting. Notice that we the training set has only 8 samples and the dimension is 10, so we have less data than dimensions. The high regularization parameter prevent overfitting.

We performed RR on the full training set with 100 samples selecting the regularization parameter that gives rise to the smallest validation set error.Training error and validation error are equal to



(a) 100 sample set (b) 10 sample set

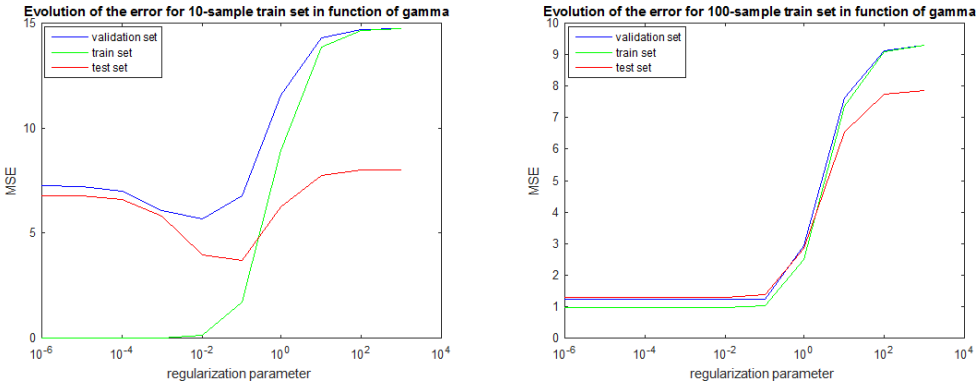
Figure 6: Average MSE in function of gamma .

0.2027 and 3.2162 respectively for 10 sample set and 0.8513 and 1.1106 for 100 sample set. It's obvious that for bigger training set we reached better results.

It is better to tune regularization parameter on validation set, which is independent from training set.

6 Tuning the regularization parameter using cross validation

We use 5-fold cross validation to tune the regularization parameter. Figure 7 shows cross-validation score on top of the training and test set error for different values $\gamma = \{10^{-6}, 10^{-5}, \ldots, 10^{3}\}$ of the regularization parameter.



(a) 10-sample training set (b) 100-sample training set

Figure 7: Evaluation of the mean square error in function of gamma for different number of training examples.

Cross validation error is a good estimation of error for test set for both 10 and 100 samples.

7 Comparison of γ tuning methods

The goal of this exercise is to generate 200 such data as in above exercises and tune the regularization parameter γ using three methods. The results of average test error and standard deviation of the errors for 10 sample set-up are presented below:

1. By minimizing the training error
   - Test set error: 163.9892
   - Standard deviation of error: 508.3517
2. By minimizing the validation error (80% / 20 % split)
   - Test set error: 35.5974
   - Standard deviation of error: 192.3498
3. By minimizing the 5-fold cross validation error
   - Test set error: 6.2890
   - Standard deviation of error: 10.1677

The best results gives tunning regularization by minimizing the 5-fold cross validation error. Our model has only 10 samples and it's very prone to overfitting, so tunning the regularization parameter is very important in this case. Minimizing the 5-fold cross validation error gives us reasonable results despite the fact that training set is very small. Minimizing the training error doesn't seem to be a good approach, because the test set error and standard deviation of error are very big. Using only one validation set gives us better results but still not such good as using cross validation method.

The results of average test error and standard deviation of the errors for 100 sample set-up are presented below:

1. By minimizing the training error
   - Test set error: 1.1202
   - Standard deviation of error: 0.0831
2. By minimizing the validation error (80% / 20 % split)
   - Test set error: 1.1529
   - Standard deviation of error: 0.1283
3. By minimizing the 5-fold cross validation error
   - Test set error: 1.1208
   - Standard deviation of error: 0.0848

First of all, the results are much better than for 10 sample set-up, but is is because we increase training set. Secondly, differences between results for each method are very small. It's because, we have a big enough train set and regularization isn't really necessary. The model isn't prone to overfitting.

8 Load the data

We loaded the boston data set into Matlab.

9 Baseline versus full linear regression.

In this exercise we try a baseline method works for a problem instead of use all of our attributes for prediction. Firstly, we implemented Naive Regression, which predict with the mean y-value on the training set. The averaged training and test errors (after 20 runs) are equal to 84.5706 and 84.3933 respectively. Then Linear Regression with single attributes was implemented. For each of the thirteen attributes, we performed a linear regression using only the single attribute but incorporating a bias term. Results are shown in figure 10.

In general, results are better than for Naive Regression, but they depends on which attribute we selected. It means that some attributes are more important (have more information that we can use to distinguish classes) that others. For example, Linear Regression with 13th attribute has error on test set: 39.1051 in contrast to with 4th attribute: 83.5478.

Linear Regression using all attributes outperforms any of the individual regressors (MSE on test set: 27.8889.

10 Kernel Ridge Regression

In this exercise we will perform kernel ridge regression on the Boston data set with the Gaussian kernel, which is defined as:

$$K(x_i, x_j) = exp(-\frac{||x_i - x_j||^2}{2\sigma^2}) \tag{5}$$

We created a function *kridgereg.m* to perform kernel ridge regression using equation $\alpha^* = (K + \gamma l I_l)^{-1} y$. We don't explicitly use a matrix inverse instead use the matrix left division operator. According to the Matlab documentation, *mldivide* is a more efficient way to solve a linear equation $Ax = b$. If the solution does not exist or if it is not unique, the *mldivide* operator issues a warning.

Then, we created a function called *dualcost.m* to calculate the Mean Squared Error (MSE) using equation:

$$mse = \frac{1}{l}(K_{test}\alpha - y)'(K_{test}\alpha - y) \tag{6}$$

We perform kernel ridge regression on the training set using five-fold cross-validation with different values of γ and σ. Figure 8 shows the cross-validation error as a function of γ and σ. The best results are for values $\gamma = 2^{-31}$ and $\sigma = 2^{12}$. The MSE on the training data = 8.8392 and on the test set = 13.8215.

Then, we repeat those steps over 20 random splits of data. Our results for exercise 9 and 10 are summarized in the table:

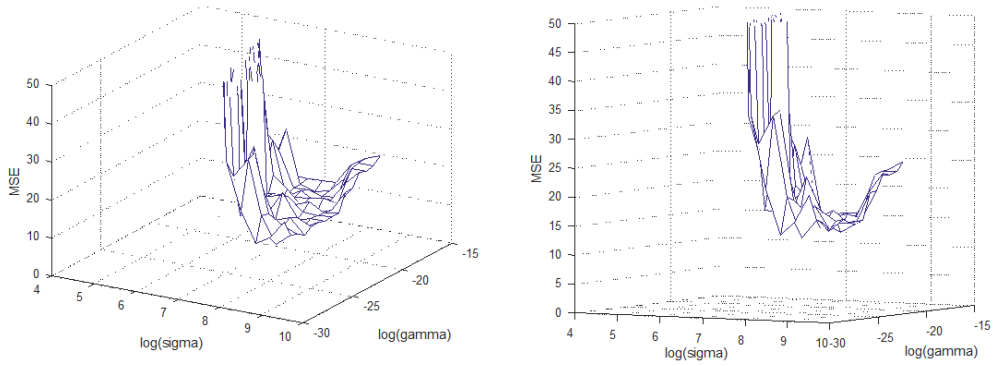


Figure 8: Cross-validation error as a function of γ and σ.

Method	**MSE train**	**MSE test**
Naive Regression	84.5706 ± 3.7489	84.3933 ± 7.3810
Linear Regression (attribute 1)	71.9206 ± 3.7157	71.7623± 7.2524
Linear Regression (attribute 2)	73.4863 ± 4.2998	73.7122± 8.4777
Linear Regression (attribute 3)	64.8467 ± 4.6972	64.7529± 9.1959
Linear Regression (attribute 4)	81.3032 ± 3.2466	83.5478± 6.3118
Linear Regression (attribute 5)	69.1346 ± 3.6979	69.0250± 7.8448
Linear Regression (attribute 6)	43.0045 ± 4.6585	45.4416± 9.3804
Linear Regression (attribute 7)	72.6169 ± 4.3741	72.4109± 8.5975
Linear Regression (attribute 8)	79.1138 ± 4.3852	79.6780± 8.5893
Linear Regression (attribute 9)	72.4303 ± 4.2039	71.9495± 8.2573
Linear Regression (attribute 10)	66.3728 ± 4.5274	65.3491± 8.9139
Linear Regression (attribute 11)	63.3372 ± 3.9821	61.7821± 7.9450
Linear Regression (attribute 12)	75.1591 ± 3.8102	75.1225± 7.5406
Linear Regression (attribute 13)	38.2716 ± 2.1729	39.1051± 4.2303
Linear Regression (all attributes)	23.1232 ± 2.9711	27.8889± 6.2950
Kernel Ridge Regression	7.9308±1.3363	13.7928± 2.4398

Part 2

1