Machine Learning Lab 4

Zhikun Zhu; ID: 29356822

November 3, 2017

I Liner Regression

In this section Liner Regression method was taken to training the data from Boston Housing price dataset. Here define $y = (x\ 1)^T$ and $a = (w\ w_0)$, which is same with lecture. From the defination, we have error function:

$$E = \sum_{n=1}^{N} \{y_n^T a - f_n\}^2 \tag{1}$$

In order to minimize error, we can take derivatives of E on each direction of a_i, which, in other word, is gradient of E. Then let the gradient equals zero to achieve minimum error.

$$\frac{\partial E}{\partial a_i} = 2\sum_{n=1}^{N}\{(\sum_{j=1}^{p+1} a_j y_{nj}) - f_n\} y_{ni} = 0$$

$$\sum_{n=1}^{N} y_{ni}\{(\sum_{j=1}^{p+1} a_j y_{nj})\} = \sum_{n=1}^{N} f_n y_{ni}$$

$$\sum_{j=1}^{p+1} a_j \sum_{\mathbf{n=1}}^{\mathbf{N}} (\mathbf{y_{ni} y_{nj}}) = \sum_{n=1}^{N} f_n y_{ni} \tag{2}$$

By simply changing the sequence of two sums, the left side equation can be separated into two part,$\sum_{j=1}^{p+1} a_j$ and $\sum_{n=1}^{N}(y_{ni}y_{nj})$.
Then, firstly, for all $i = 1 : (p+1)$, we have (p+1) equations. Secondly, for each $j = 1 : (p+1)$, $\sum_{n=1}^{N}(y_{ni}y_{nj})$ means dot product of i-th column and j-th column of y. This is, dot product of i-th row of y^T and j-th column of y. Then it becomes the form of $y^T y$, and a_j is the coefficient of liner combination of columns of $y^T y$. Finally, the (p+1) equations from eq.2 can be written in matrix form:

$$y^T y a = y^T f$$

Multiple inverse matrix $(y^T y)^{-1}$ on both side and we get:

$$a = (y^T y)^{-1} y^T f \tag{3}$$

It is worth to notice that there is no inverse matrix for a non-square matrix. Then, $(y^T)^{-1}$ can not eliminate with y^T.
The result of applied such liner regression to Boston time dataset is shown in fig.1. And the final sum of error in each direction via all elements in training set is 130.9755.

Figure 1: True house price and its liner regression prediction

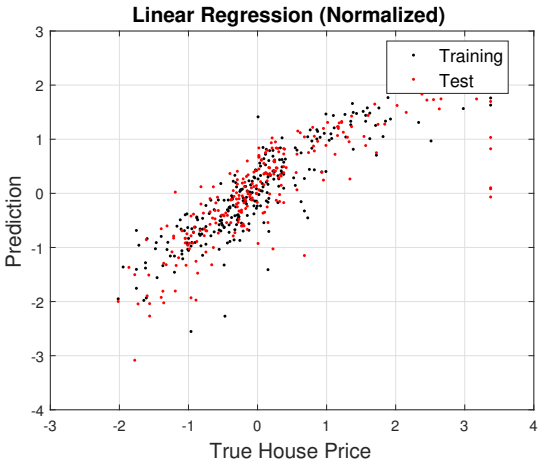

Figure 2: Training and Test sets with thier respctive predictions.

In the following question, the Boston Housing dataset(506×14 $matrix$) was splitted into two equal length row matrixes for traning and test respectively. It should be noticed that, means and variance of training set was used to normalize both classes, since we cannot use the information from test set to adjust the model. The predicted results for both set was shown in fig.2. And their error sum same with mentioned above are 55.1467 and 73.6336 for training and test set, respectively. The algorthim was runned 10 times and the final result is the average of these 10 results. It is obvious that the magnitude of error of training set is lower than test set, since the algorithm is trained to minimize the error for training set.

II k-fold Cross Validation

In this section, 10-fold cross validation was impledmented to the Boston Housing dataset. Briefly, k-fold Cross Validation algorithm divide the dataset into k subsets with equal cardinality. And for each subset k_i, k_i was used as test set to test the training result of the remaining (k-1) subsets. Finally, we have k errors and take average of them we get the algorithm's error.