Machine Learning - Robotics Engineering
Report Assignment 2
Linear Regression

Francesca Canale 4113133

4/11/2019

Abstract

This assignment is based on linear regression that is a machine learning algorithm which aims to model the relationship between a dependent variable y and one or more independent variables x. More specifically, it assumes that y can be calculated from a linear combination of the input variables x. The requirement of the assignment was to implement three different linear regression models in MATLAB. We had also to test them evaluating their mean square error.

1 Introduction

The goal of this lab assignment was to build three different types of linear regression models: one-dimensional linear regression with and without intercept, and a multi-dimensional one. It was also required to test them with two data sets.

2 Data set

We have been given two different data sets to work with. The first one is about the variation of the MSCI Turkish index with respect to Standard and Poor's 500 return index; it is composed of two columns (SP500 and MSCI) and 536 observations. The second data set is about a survey on some car models that takes into account four variables: the miles-per-gallon (mpg), the displacement (disp), the horse-power (hp) and the weight; it is composed of four columns, one for each variable, and 32 observations.

3 Linear regression

The goal of regression is to predict the value of one or more target variables t given the value of a D-dimensional vector X of input variables comprising of N observations:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_{\mathrm{N}} \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & ... & x_{1,\mathrm{D}} \\ x_{2,1} & x_{2,2} & ... & x_{2,\mathrm{D}} \\ & & . & \\ & & . & \\ & & . & \\ x_{\mathrm{N},1} & x_{\mathrm{N},2} & ... & x_{\mathrm{N,D}} \end{pmatrix}, \qquad t = \begin{pmatrix} t_1 \\ t_2 \\ . \\ . \\ . \\ t_{\mathrm{N}} \end{pmatrix}$$

so given a training data set x_{n}, where n = 1,...,N, together with corresponding target values t_{n}, the goal is to predict the value of t for a new value of x.
The simplest linear model for regression is one that involves a linear combination of the input variables:

$$y(x, w) = w_0 + w_1 x_1 + ... + w_{\mathrm{D}} x_{\mathrm{D}} \tag{1}$$

This is often simply known as linear regression. [1]
Since generally it is not possible to find values of w_i, with i = 1,...,D, that are good for all points of a data set, it is sufficient to choose their values that minimize the cost of a loss function. A common choice of loss function in regression problems is the squared error loss given by:

$$\lambda_{\mathrm{SE}}(t, y) = (t - y)^2 \tag{2}$$

that has the interesting properties of being even, of growing more that linearly, so giving heavier weight to a larger error, and of being differentiable with respect to the model output. The objective function (or cost function) that we want to minimize represents the mean value of the loss over the whole data set:

$$J_{\mathrm{MSE}} = \frac{1}{N}\sum_{l=1}^{N}\lambda_{\mathrm{SE}}(t_l, y_l) = \frac{1}{N}\sum_{l=1}^{N}(t_l - y_l)^2 \tag{3}$$

and it is called *mean square error objective*. It is a quadratic function and hence its minimum always exists. [2]

3.1 One-dimensional linear regression

In a one-dimensional linear regression we have that D equals 1 so our observation vector has Nx1 dimensions. In this case the equation (1) becomes the equation of a straight line:

$$y(x, w) = w_0 + w_1 x_1 \tag{4}$$

where w_1 is called *slope* and w_0 is called *intercept*.

3.1.1 Without intercept

If the intercept is not present then the straight line $y(x, w) = w_1 x_1$ passes through the origin of the axes. In this case the only parameter that we have to compute to build the linear regression is the slope w_1.
The mean square error objective described in equation (3) is minimized when:

$$w_1 = \sum_{l=1}^{N} x_l t_l / \sum_{l=1}^{N} {x_l}^2 \tag{5}$$

To implement this type of linear regression I implemented a MATLAB function *linearRegression()* that takes in input a data set composed of N rows and 2 columns. The first column represent the observation vector and the second one is the target. The function computes the value of the slope w_1 implementing equation (5). The results obtained on the Turkish stock exchange data using this function are illustrated in Fig. 1: the red x represents the data set and the blue line is the one of equation $y = w_1 x$. If instead of giving as input the whole data set I divide it into different random subsets having dimension the 10% of N, I obtain the lines illustrated in Fig. 2: these lines are characterized by different values of the slope and they all intersect in the origin.

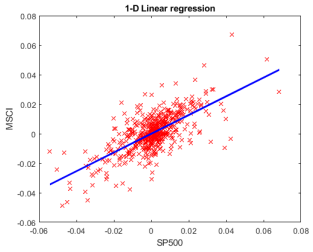

Figure 1: One-dimensional linear regression on the Turkish stock exchange data

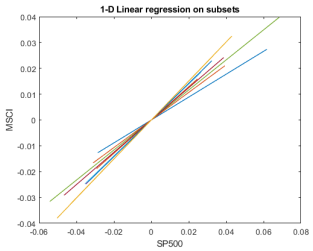

Figure 2: One-dimensional linear regression on subsets of the data set

3.1.2 With intercept

In this case the mean square error objective described in equation (3) is minimized when:

$$w_1 = \sum_{l=1}^{N}(x_l - \overline{x})(t_l - \overline{t}) / \sum_{l=1}^{N}(x_l - \overline{x})^2 \tag{6}$$

$$w_0 = \overline{t} - w_1\overline{x} \tag{7}$$

with $\overline{x} = \frac{1}{N}\sum_{l=1}^{N} x_l$ and $\overline{t} = \frac{1}{N}\sum_{l=1}^{N} t_l$. To implement this type of linear regression I implemented a MATLAB function *linearRegressionIntercept()* that takes the same inputs of the first function already explained. The function computes the value of the slope w_1 implementing equation (6) and the value of the intercept w_0 implementing equation (7). The results obtained on the Motor Trends car data using this function and considering as observation vector the weight column and as target the mpg one, are illustrated in Fig. 3: the red x represents the data set and the blue line is the one of equation $y = w_1 x + w_0$.

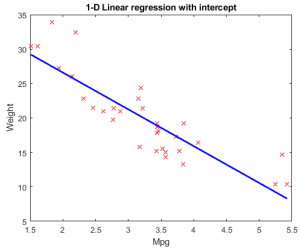

Figure 3: One-dimensional linear regression with intercept on the Motor Trends car data

3.2 Multi-dimensional linear regression

In a multi-dimensional regression problem we have that D>1 and so we have to find as many w as the value of D. In this case the mean square error objective described in equation (3) is minimized when:

$$w = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}t = X^{\dagger}t \tag{8}$$

To implement this type of linear regression I implemented a MATLAB function *linearRegressionMultiD()* that takes in input a matrix X composed of D columns and a target vector. The function computes the values of the slope vector implementing equation (8). The results obtained on the Motor Trends car data using this function and considering as matrix X the columns disp, hp and weight and as target the mpg column, are shown in Fig. (4): the first column of the table contains the real value of the target while the second column contains the values predicted with the linear regression.

Mpg real (t)	Mpg estimated (y)	Mpg real (t)	Mpg estimated (y)
21	17.7300	14.7000	23.1416
21	20.8063	32.4000	20.2359
22.8000	19.4427	30.4000	12.9155
21.4000	13.4556	33.9000	16.6769
18.7000	7.0986	21.5000	19.9532
18.1000	20.0486	15.5000	11.8763
14.3000	11.7344	15.2000	12.4870
24.4000	24.0573	13.3000	16.1603
22.8000	25.7103	19.2000	7.3100
19.2000	27.3040	27.3000	17.0039
17.8000	27.3040	26	15.7461
16.4000	24.7577	30.4000	12.0911
17.3000	20.6560	15.8000	8.7933
15.2000	21.2592	19.7000	24.1410
10.4000	17.1604	15	22.5733
10.4000	21.1001	21.4000	24.1740

Figure 4: Multi-dimensional linear regression on the Motor Trends car data

4 Testing the regression models

In order to test the regression models I have implemented one MATLAB function *MSE()* that computes the mean square error (MSE) defined in equation (3). I have implemented a MATLAB script that execute the three models of linear regression on some random training sets composed of 10% of the original data sets and computes the relative MSE. After that it computes the MSE between the regression obtained with the slope (and also the intercept when needed) just computed and the target of the test sets obtained considering the remaining 90% of the original data sets. The script repeats all this steps 100 times and the resulting average mean square error values are shown in Fig. (5).

	Training	Test
Model 1	8.5757e-05	9.2638e-05
Model 2	4.2887	18.3895
Model 3	23.4931	1.2551e+03

Figure 5: Values of the average MSE

5 Results and conclusion

As it can be seen in Fig. (5) the average mean square errors obtained on the test sets are always higher than the ones obtained on the training sets: this is reasonable because the regression is build on the training set so obviously it fits better on it than on the test one. Moreover, we can notice that the errors obtained on the Turkish stock exchange data (Model 1) are lower than the ones obtained for the Motor Trends car data (Model 2 and 3): this because the training set built on the first one is made of 53 observations while the one built on the second data set is made of only 4 observation, a little too few to get satisfactory results. Lastly, the results obtained on the third model are worse than the others because in this case the regression problem is solved using the Moore-Penrose pseudo-inverse that leads to a solution that does not always exist and that can be of low-quality; this problem can be solved using some iterative approximation methods like the gradient descent.

References

[1] Pattern Recognition and Machine Learning - C. M. Bishop

[2] The Elements of Statistical Learning - T. Hastie, R. Tibshirani, J. Friedman