

Machine learning Assignment2

Koushikmani Maskalmatti Lakshman
S5053566

Abstract This assignment aims to implement in MATLAB three different linear regression models and to also calculate their mean square error objective. Results are tested on two different datasets and on different percentages of the data.

1 Introduction

1.1 Linear regression problem

Linear regression is a machine learning tool whose purpose is to model the relationship between a dependent variable t (target) and a set of one or more dependent variables X (data). To do so, it looks for a linear model $y(x)$ that predicts t given X :

$$y = wX \quad (1)$$

Where the w vector represents the weights that must be calculated. In matrix form:

$$\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix} \simeq \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} x_1 1 & \dots & x_1 d \\ x_2 1 & \dots & x_2 d \\ \vdots & \ddots & \vdots \\ x_N 1 & \dots & x_N d \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} \quad (2)$$

Since the problem above would have no closed form solution, we look for a solution based on optimization: we try to minimize an objective function J which is the expectation of a chosen loss function (the mean square error function).

So we calculate the mean square error (loss function):

$$\lambda_S E = (t - y)^2 \quad (3)$$

We then calculate the function that minimizes the mean value of the loss over the whole dataset (objective function):

$$J = \frac{1}{N} \sum_{i=1}^N (t_i - y_i)^2 \quad (4)$$

And we look for the minimum of this function, which is a quadratic function. [1]

1.2 The datasets

The datasets are:

- Motor Trend Car Road Tests set, made of 32 observations and 4 variables.
- Turkish stock exchange dataset, made of 536 observations and 2 variables.

2 Task 1 and 2

The main loads the two datasets through two different functions. For the turkish dataset it uses the MATLAB `load()` function, while for the car dataset it uses the MATLAB `readtable()` function, since it also contains string data together with numeric data.

Then it calls three functions to solve the linear regression problem with different models and plots the results.

Firstly it calls `oneDimLinReg()` to compute the slope for the linear model y of the first dataset. Secondly it does the same but on different subsets created by randomly picking 10% of the original dataset.

Thirdly it calls `oneDimLinReg_intercept()` to compute the slope and intercept of the model of the second dataset by considering two of its variables only.

Lastly it normalizes the values of the Car data set, calls `multiDimLinReg()` on the entire second dataset and shows a table of the de-normalized results.

2.1 `oneDimLinReg.m`

Input:

- x : data
- t : target

Output:

- w : weights

This function computes the weights for the 1D linear model (i.e. the slope of $y = w_1 x$). In formula:

$$w = \frac{\sum_{i=1}^N (x_i t_i)^2}{\sum_{i=1}^N (x_i)^2} \quad (5)$$

2.2 `oneDimLinReg_intercept.m`

Input:

- x : data
- t : target

Output:

- w_1 : slope
- w_0 : intercept

his function computes the weights for the 1D linear model (i.e. the slope and intercept of $y = w_1 x + w_0$). In formula:

$$w_1 = \frac{\sum_{i=1}^N (x_i - \bar{x})(t_i - \bar{t})}{\sum_{i=1}^N (x_i - \bar{x})^2} \quad (6)$$

$$w_0 = (t_i - w_1 \bar{x}) \quad (7)$$

2.3 `multiDimLinReg.m`

Input:

- X : data matrix
- t : target vector

Output:

- W : weights vector

This function computes the weights vector for the multi dimensional linear regression problem. In formula:

$$w = (X^T X)^{-1} X^T t \quad (8)$$

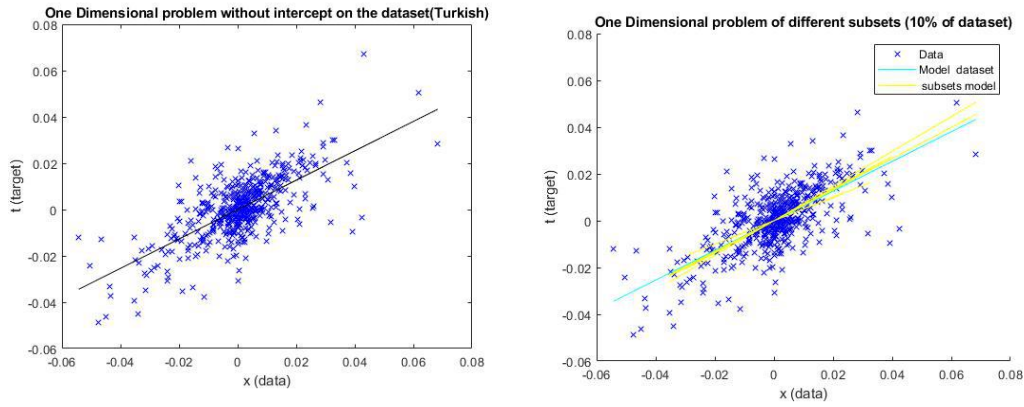


Figure 1: 1D problem

3 Task 3

The main divides the datasets in two parts: 5% is used to build a training set, while 95% is used to build a test set. In this particular case, the training set of the Turkish set is made of 27 lines, while the one of the car set is made of 2 lines only. The code then enters a loop and re-computes the 1-D problem and multi-D problems using the training sets. It subsequently tests the model on the test sets and computes the objective functions J for all cases. Finally it averages the results of all the J and shows them.

3.1 meanSquareError.m

Input:

- x: data
- t: target
- w1: slope
- w0: intercept
- problem: type of linear regression problem. can be 1 for 1-D, 2 for multi-D.

Output:

- objective: objective function

This function computes the mean square error objective of a linear regression problem. In particular, based on the *problem* variable it computes the error for either the one or the multi dimensional problem. The 1D problem's objective is calculated as:

$$J = \frac{1}{N} \sum_{i=1}^N (t_i - y_i)^2 \quad (9)$$

While the multi dimensional problem's one is calculated through the MATLAB function `immse()`.

4 Results

Hereby are shown the results that were obtained. Figure 1 (a) shows the solution to the linear regression problem applied to the entire Turkish dataset.

Figure 1 (b) shows the different results obtained by applying the same problem to subsets created by randomly picking 10% of the original dataset. The red line is the model computed on the entire dataset, while the green lines are the ones computed on the subsets. In the second case the results are less accurate then the first case, since the number of observations is lower (54 observations each). Figure 2 shows the solution to the linear regression problem applied to the car dataset considering

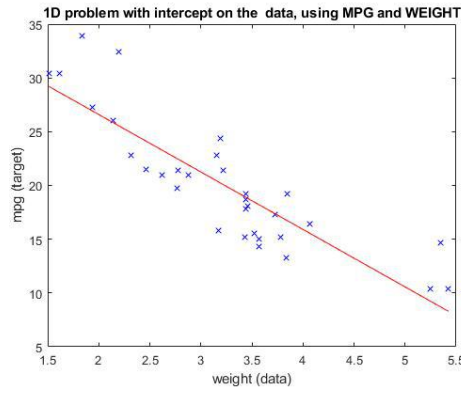


Figure 2: 1D problem with intercept

	Real-----Target t	Predicted-----Target y
1	21	23.5700
2	21	22.6008
3	22.8000	25.2887
4	21.4000	21.2167
5	18.7000	18.2407
6	18.1000	20.4722
7	14.3000	15.5656
8	24.4000	22.9115
9	22.8000	22.0409
10	19.2000	20.0411
11	17.8000	20.0411
12	16.4000	15.7693
13	17.3000	17.0616
14	15.2000	16.8715
15	10.4000	10.3215
16	10.4000	9.3598
17	14.7000	9.2115
18	32.4000	26.6135
19	30.4000	29.2760
20	33.9000	28.0391
21	21.5000	24.6016
22	15.5000	18.7549
23	15.2000	19.0911
24	13.3000	14.5488
25	19.2000	16.6639
26	27.3000	27.6204
27	26	26.0236
28	30.4000	27.7450
29	15.8000	16.5025
30	19.7000	20.9888
31	15	12.8168
32	21.4000	23.0296

(a) multi-D problem intercept

	Train	Test
1-D	1.4730e-04	4.1671e-04
1-D offset	2.0763e-04	1.3823e-04
multi-DIMENSIONAL	2.1109e-04	2.5638e-04

(b) Training-Test results

Figure 3

just the mpg as data and the weight as target.

Figure 3 (a) shows the t and y of the multi-D problem applied to the car set. Results are denormalized. Figure 3 (b) shows the results on the training set and on the test set, which have in general the same magnitude. The 1-D model is computed with more observations with respect to the other two, so it should have lower errors. Here, this only happens in the test set. The multi-D model has higher errors since the W is computed through the Moore-Penrose inverse which can bring to numerical errors. To solve this problem it would be useful to make an iterative computation by successive approximations. Finally, results on the training sets are mostly lower than results on the training set, which is reasonable since the model was computed on the training sets.