Linear Regression
Assignment 2 of Machine Learning I, 2022/2023

Mura Alessio

DIBRIS - Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi
Università Degli Studi di Genova

1 Introduction

Linear regression is an algorithm used to predict, or visualize, a relationship between two different features/variables. In linear regression tasks, there are two kinds of variables being examined: the dependent variable and the independent variable.

The independent variable is the variable that stands by itself, not impacted by the other variable. As the independent variable is adjusted, the levels of the dependent variable will fluctuate. The dependent variable is the variable that is being studied, and it is what the regression model solves for/attempts to predict. In linear regression tasks, every observation/instance is comprised of both the dependent variable value and the independent variable value.

Linear means that the approximation uses a first order polynomial function to approximate the data. This implies that, in the case of a one-dimensional problem, the approximation will be a line.

2 Theory of Linear Regression

2.1 One-dimensional problem

Keeping in mind that the linear regression defines a linear approximation, the problem can be formalized in the following way:

Define a linear function such that, given an input *x* (observation), computes the target *t* that best approximates the real value *y*.

$$t \approx y, where \rightarrow t = wx \tag{1}$$

w is the linear regression parameters, and associate with a linear function the input and the target.

Suppose there are two different observation x_0 and x_1.
Arranging the equation 1 it is simple to get the value of *w*, which is:

$$w_0 = \frac{x_0}{t_0} \tag{2}$$

However, with the second observation we get:

$$w_1 = \frac{x_1}{t_1} \tag{3}$$

It is clear that, comparing the equation 2 and 3, the two values of *w*, w_0 and w_1, can not be the same if the observation are different, so we understand that the approximation can't ever be *perfect*, and it introduces a small error to the target result.

Linear regression aims to compute the best value of *w* that better approximates every observation. *w* must then minimize the mean error.

Square error (SE) is then the best solution for finding the best value of w, because it grows more than linearly, giving heavier weight to larger errors. It is also even, meaning that $(y-t)^2 = (t-y)^2$, and it is differentiable with respect to the model output:

$$\frac{d}{dx}\lambda_{SE}(y,t) = 2(y-t) \tag{4}$$

Since we need to take the *mean* SE, we can use the Mean Square error (MSE), that can be computed as:

$$J_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2 \tag{5}$$

After some computation, it is proven that the best value of *w* that approximates the best the value of the set of observation is:

$$w = \frac{\sum_{l=1}^{N} x_l t_l}{\sum_{l=1}^{N} {x_l}^2} \tag{6}$$

With the equation 6 is then possible to get the best value of *w* for the equation 1.

2.2 One-dimensional problem with intercept

It is possible to build a better regression model using an interception factor, w_0, that can move to the left or to the right the approximation line, The equation 1 becomes then:

$$t = w_0 x + w_1 \tag{7}$$

The two parameters, w0 and w1 can then be computed as follows:

$$w_1 = \frac{\sum_{l=1}^{N} (x_l - \bar{x}) \left(t_l - \bar{t}\right)}{\sum_{l=1}^{N} (x_l - \bar{x})^2} \tag{8}$$

$$w_0 = \bar{t} - w_1 \bar{x} \tag{9}$$

2.3 Multi-dimensional problem

The equations above can be generalized for the multidimensional case. Let **X** the set of observation, each of them with multiple variables. Let **w** the vector of all the approximation parameters, one for each variable, including w_0 intercept parameter:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & . & . & x_{1,n} \\ x_{2,2} & x_{2,2} & . & . & x_{2,n} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{n,n} & x_{n,2} & . & . & x_{n,n} \end{bmatrix} \tag{10}$$

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ . \\ . \\ w_n \end{bmatrix} \tag{11}$$

The final output oh the linear regression will be:

$$y_i = w_0 + x_{i,1} w_1 + x_{i,2} w_2 + ... + x_{i,n} w_n \tag{12}$$

We then obtain:

$$\mathbf{y} = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & . & . & x_{1,n} \\ 1 & x_{2,1} & x_{2,2} & . & . & x_{2,n} \\ . & . & . & . & . & \\ . & . & . & . & . & \\ 1 & x_{n,1} & x_{n,2} & . & . & x_{n,n} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ . \\ . \\ w_n \end{bmatrix} = \mathbf{Xw} \tag{13}$$

Note that a constant column of ones is added in the first matrix, in order to incorporate w_0.

As for the one-dimensional case, also here it is necessary to minimize the MSE. Since we are dealing with matrices and not one-dimensional variables, now we must set $\nabla J_{MSE} = 0$. We can then find **w** as follows:

$$\mathbf{w} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{t} = \mathbf{X}^{\dagger} \mathbf{t} \tag{14}$$

Where $\mathbf{X}^{\dagger}$ is the Moore-Penrose pseudoinverse of **X**. Notice that computing the pseudoinverse can result in introducing a small but notable error in the calculation. This small error however can be quickly amplified by the condition number and cause large errors on the result.

3 THE ASSIGNMENT

The given assignment consist of three tasks:

- **Task 1**: Get data
- **Task 2**: Fit a linear regression model
- **Task 3**: Test regression model

3.1 Task 1: Get data

This assignment takes into account two different sets:

- turkish-se-SP500vsMSCI (Turkish stock exchange data);
- mtcarsdata-4features (1974 Motor Trend Car Road Tests).

The first set is composed by 536 observations and 1 variable. The second set is composed by 32 observations and 3 variables.

The Two sets are initially stored as .csv files. To import them in Python it is required to use the function read-csv(), inside Pandas library, in order to store them into dataframes.

3.2 Task 2: Fit a linear regression model

In the second tasks, we have to compute the linear regression parameters in four different cases:

1. One-dimensional problem without intercept on the Turkish stock exchange data;
2. Compare graphically the solution obtained on different random subsets (10%) of the whole data set;
3. One-dimensional problem with intercept on the Motor Trends car data, using columns mpg and weight;
4. Multi-dimensional problem on the complete MTcars data, using all four columns (predict mpg with the other three columns).

3.2.1 1

In order to compute and visualize the linear regression, we created a function, *BuildGraph()*, in which we passed: the turkish dataset, the two columns, and a boolean value, which could be *yes* [TRUE] or *no* [FALSE].

In this case, we passed *yes*, meaning that we want to compute the linear regression without intercept. In fact, inside *BuildGraph()*, after creating a vector with the the variables we need by *create-vector()* function and thanks to the boolean value yes, we call three functions: *estimate-coef-without-intercept()*, *OneDimensionalMeanSquareError()* and *plot-regression-line-with-intercept()*.

By these functions, we estimated the *w* coefficient without intercept, we computed the mean square error and, finally, we plotted the result into a graph. All these steps have been done following the theory explained in the *Theory of Linear Regression* section.

3.2.2 2

In order to get different subsets with 10% of the whole data, we created a function, *randomSubset()*, where we compute the linear regression without intercept in the same way as the previous step. The final results will be shown in a subplot.

3.2.3 3

In order to compute the linear regression with intercept on MotorTrendsCar dataset, we reused *BuildGraph()* function, but, this time, the boolean value is *no*, in order to compute it taking into account of intercept. By computing it, we used mpg and weight columns.

3.2.4 4

In this last step, we created a new function, *MultiDimensional()*, in order to predict *mpg* with the other three columns. Obviously, the dataset we used is MotorTrendsCar. According with the theory, we compute the mean square error for the multi-dimensional case by *MultiDimensionalMeanSquareError()* function and, finally, printed our prediction.

3.3 Task 3: Test regression model

The last task requires to re-run 1,3 and 4 from task 2 using only 5% of the data, compute the objective (mean square error) on the training data, compute the objective of the same models on the remaining 95% of the data and repeat for different training-test random splits.

4 RESULTS

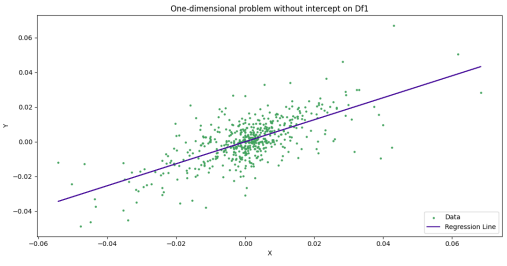

Figure 1. One-dimensional problem without intercept.

As we can clearly see in Figure 1, we found the best value of *w* that approximates our observation. The fact that we did not use the intercept is denoted by the passage of the regression line through the origin.

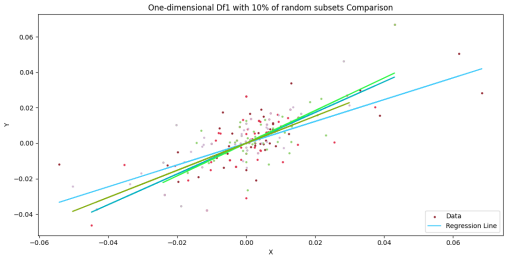

Figure 2. One-dimensional problem without intercept with random subsets using 10% of the whole data.

The figure 2 shows instead a graphically comparison between different random subsets (10%) of the whole data set. we can observe that all the graphs are similar to each other and also similar to the result obtained using the whole dataset.

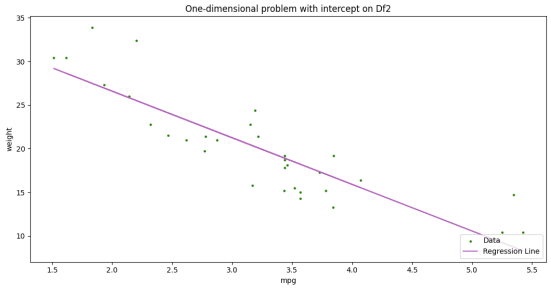

Figure 3. One-dimensional problem with intercept.

In the figure 3 we can see the linear regression with the intercept factor. This is still a one-dimensional problem: only the car weight is used as variable. In this case, the dataset we used is MotorTrendsCar.

```
mpg predicted:  [17.7300375  20.80633344 19.44268689 13.45563111  7.09858125 20.04860722
 11.7344481  24.057253   25.71031483 27.30398202 27.30398202 24.75769425
 20.65596634 21.25916162 17.16042693 21.10010764 23.14162385 20.23585976
 12.91554774 16.67685959 19.95321775 11.87632126 12.48695075 16.16031795
  7.31000171 17.00386631 15.74614247 12.09107429  8.79326283 24.1409941
 22.57328313 24.17403992]
```

Figure 4. mpg prediction using the Multi-dimensional regression

By Figure 4, It is clearly visible that our mpg prediction, using the multi-dimensional linear regression on MotorTrendsCar dataset, is quite successful. The prediction we obtained is similar to the real values of mpg. Obviously our prediction is affected by error.

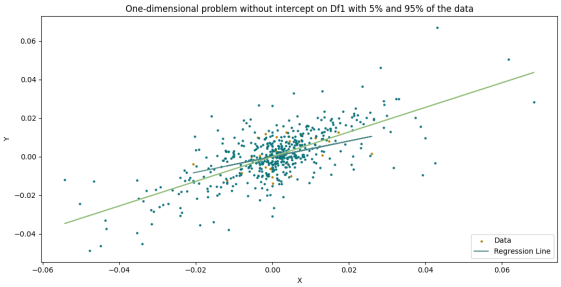

Figure 5. One-dimensional problem without intercept using 5% and 95% of the data.

As in Figure 1, in Figure 5 we found the best value of w that approximates our observation by using 5% and 95% of the whole dataset. They are quite similar to each other and to the result using the whole dataset: also the slops of the regression lines are similar.

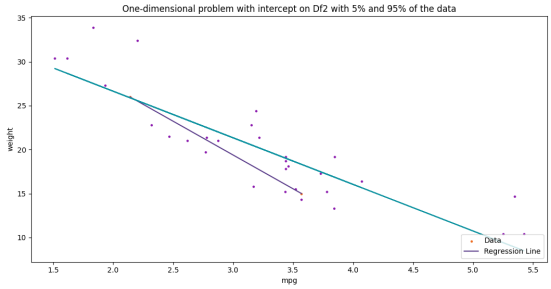

Figure 6. One-dimensional problem without intercept using 5% and 95% of the data.

As in Figure 3, in Figure 6 we can see the linear regression with the intercept factor by using 5% and 95% of the whole dataset. If the subset should correspond to the 5% of the total set, it means that the cars data set should be composed by only 2 observations. This has some consequences on the test outcome: If a set is made of only two observation, the computed factor w will then be perfect. As a proof, consider that w is the slope of the correspondent regression line. Given only two points, there's one and only one line that passes through them, and that line matches the regression line. In this case $J_{MSE} = 0$.

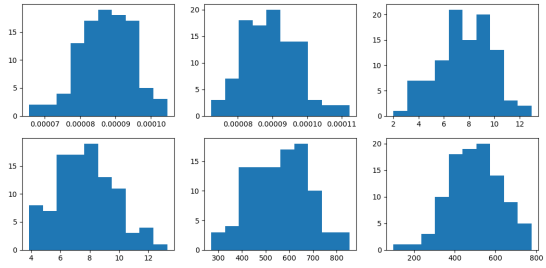

Figure 7. Hist of different training-test random splits.

The figures 7 shows a comparison of the computed objectives obtained with 100 random % of subsets, and the objectives of their consequent remaining subsets.

We created a subset of histograms for every case: first column for One-dimensional problem on Turkish dataset, second column for One-dimensional problem on MotorTrendscar dataset and the third column for Multi-dimensional problem MotorTrendscar dataset.

We can infer that our analysis has been successful by the fact that our histograms present a bell-shaped graphs.