

Linear Regression using Gradient Descent

Rodriguez Noh Santiago Miguel

A16016346@alumnos.uady.mx

Professor: Ph.D. Anabel Martin Gonzalez

link to code: https://github.com/Santiagomrn/Linear_regression

I. INTRODUCTION

Implement Linear Regression using Gradient Descent to predict weights of males and females based on their heights. We will use height as the input variable and weight as the output variable Heights ("height.dat") and weights ("weight.dat") are in inches and pounds, respectively. Use the Mean-Square-Error (MSE) function.

I-A. Theoretical framework

Hypothesis:

$$h(x) = b_0 + b_1x \quad (1)$$

Loss Function:

$$L(b_0, b_1) = \frac{1}{2n} \sum_{i=1}^n (h(x_i) - y_i)^2 \quad (2)$$

Update Rule:

for b_0 :

$$b_0 = b_0 - \alpha \frac{dL(b_0, b_1)}{db_0} \quad (3)$$

$$b_0 = b_0 - \alpha \frac{1}{n} \sum_{i=1}^n (h(x_i) - y_i)$$

for b_1 :

$$b_1 = b_1 - \alpha \frac{dL(b_0, b_1)}{db_1} \quad (4)$$

$$b_1 = b_1 - \alpha \frac{1}{n} \sum_{i=1}^n (h(x_i) - y_i)x_i$$

II. DEVELOPMENT OF PRACTICE

II-A. Plot your data set and label the axes ("Heights", "Weights").

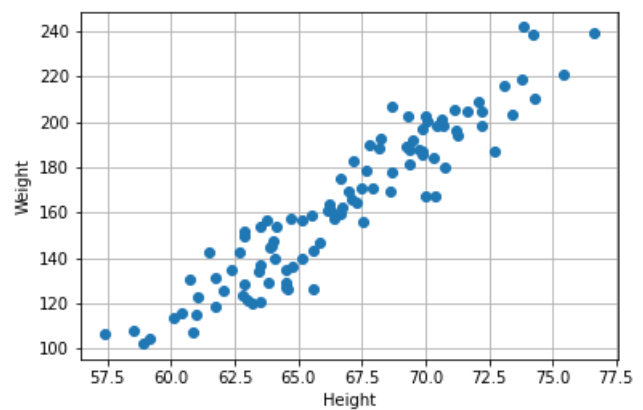


Figura 1. Weight versus height graph.

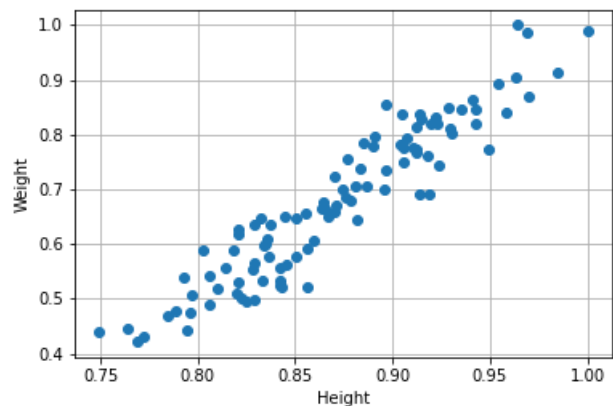


Figura 2. Normalized weight versus height graph.

II-B. Try= 0.001, 0.01, 0.5, 1, 2, 2.5 to perform the following:

1. Initialize the parameters and of your linear model to zero. Run one iteration of gradient descent from this initial starting point. Record the value of your parameters after this first iteration.

Normalized data are used for training because if they are not normalized the training with the proposed learning rates fails.

```

step: 0
learningRate 0.001
loss: 0.2391156542450006
gradient b0: -0.6771449035935605 gradient b1: -0.5970878230785783
b0: 0.0006771449035935605 b1: 0.0005970878230785783

```

```

step: 0
learningRate 0.01
loss: 0.23830133242912868
gradient b0: -0.675947620062591 gradient b1: -0.5960430660863559
b0: 0.007436621104219471 b1: 0.006557518483942137

```

```

step: 0
learningRate 0.5
loss: 0.23025108223982957
gradient b0: -0.663995858736258 gradient b1: -0.5856138852677257
b0: 0.3394345504723485 b1: 0.299364461117805

```

```

step: 0
learningRate 1
loss: 0.010786638838767525
gradient b0: -0.07692623749792095 gradient b1: -0.07333321241276958
b0: 0.41636078797026943 b1: 0.3726976735305746

```

```

step: 0
learningRate 2
loss: 0.00941275842549773
gradient b0: 0.06388245576469212 gradient b1: 0.049546558076126034
b0: 0.2885958764408852 b1: 0.2736045573783225

```

```

step: 0
learningRate 2.5
loss: 0.019272398761340508
gradient b0: -0.15020502913772818 gradient b1: -0.13724472581099267
b0: 0.6641084492852056 b1: 0.6167163719058042

```

Figure 3. Results of the first iterations with various learning rates

2. Continue running gradient descent for more iterations until your parameters converge. Plot your loss values after each iteration and record your parameters final values. The absolute value difference between the current loss and the last loss is used to verify convergence.

$$|loss - lastLoss| < 0,000001 \quad (5)$$

```

step: 1910
learningRate 0.001
loss: 0.00783031216742224
gradient b0: -0.020445838862157863 gradient b1: -0.024033936291377022
b0: 0.36725118369438453 b1: 0.3323162422970489

```

```

step: 262
learningRate 0.01
loss: 0.0075592754273525
gradient b0: -0.003422600167813563 gradient b1: -0.009172091168734943
b0: 0.3747615657595183 b1: 0.3433199346449108

```

```

step: 1420
learningRate 0.5
loss: 0.0016900812528641098
gradient b0: 0.0009289979276489374 gradient b1: -0.0010646352044697284
b0: -0.8881943596815991 b1: 1.7979807487636636

```

```

step: 916
learningRate 1
loss: 0.0013931552405074688
gradient b0: 0.0006558297198415747 gradient b1: -0.0007515833858185326
b0: -1.0503650147758679 b1: 1.9838289443373966

```

```

step: 5
learningRate 2
loss: 2393.788092343407
gradient b0: 69.17000714558156 gradient b1: 60.35146059336287
b0: -98.687968553991 b1: -86.0408594126523

```

```

step: 4
learningRate 2.5
loss: 4137.940690253257
gradient b0: -90.93559823739265 gradient b1: -79.35627679388013
b0: 176.02937015467464 b1: 153.67978293992763

```

Figure 4. Parameters final values with various learning rates

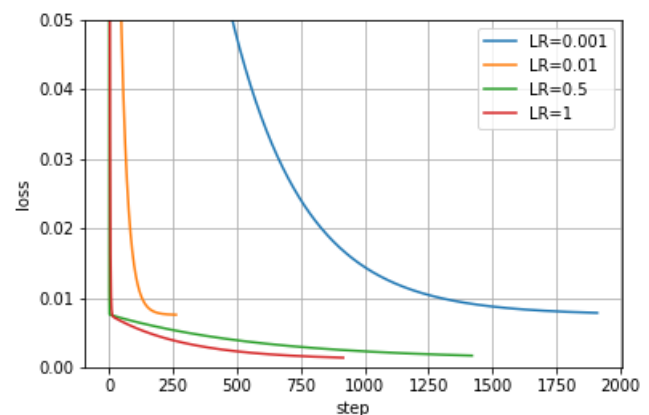


Figure 5. Loss with various learning rates

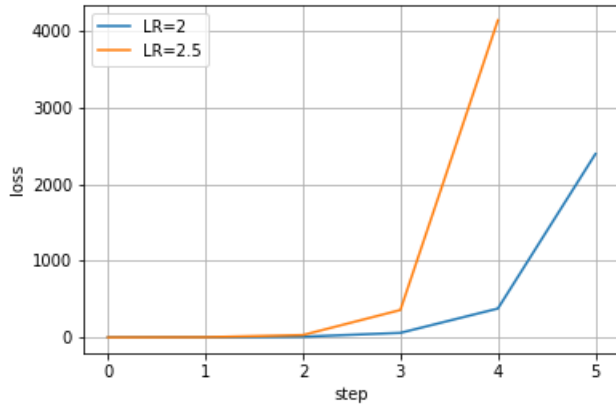


Figura 6. Loss with various learning rates

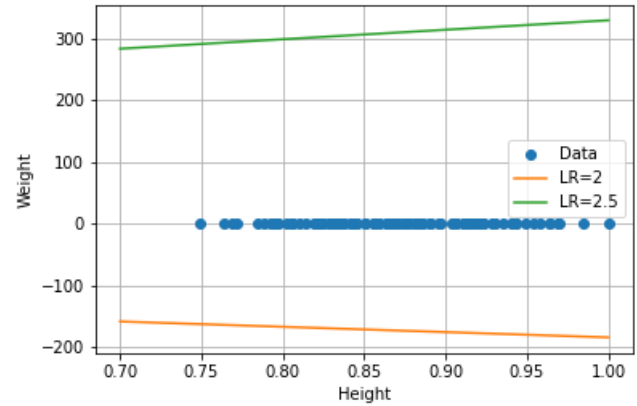


Figura 8. Models with various learning rates.

3. What step size value performs best?.

Based on the loss function the model trained with 1 of learning rate is the best model.

II-C. After convergence, plot the straight line fit from your algorithm on the same graph as your data.

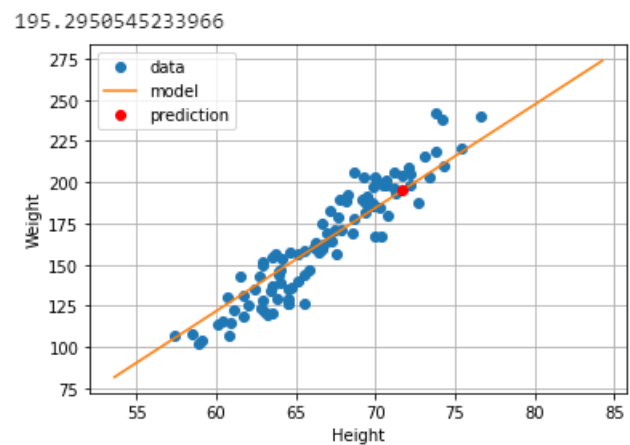


Figura 9. Prediction with data and model.

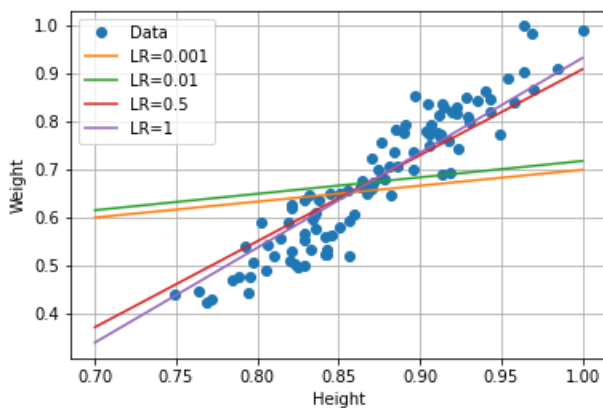


Figura 7. Models with various learning rates.

REFERENCIAS

- [1] Andrew Ng. CS229 Lecture Notes, <http://cs229.stanford.edu/notes2020spring/cs229-notes1.pdf>.