UNDERSTANDING LINEAR REGRESSION

We must decide how we're going to represent functions/hypotheses h.

As an initial choice, lets say we decide to approximate y as a linear function of x:

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

Here, the θi 's are the parameters (also called weights) parameterizing the space of linear functions mapping from X to Y. Here we are considering only one independent variable (the variable on which the output depends). In case we have multiple variables, this function can be modified.

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

Now, given a training set, how do we pick, or learn, the parameters θ?

What is Cost Function?

It is a **function** that measures the performance of a model for any given data.
Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number.

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

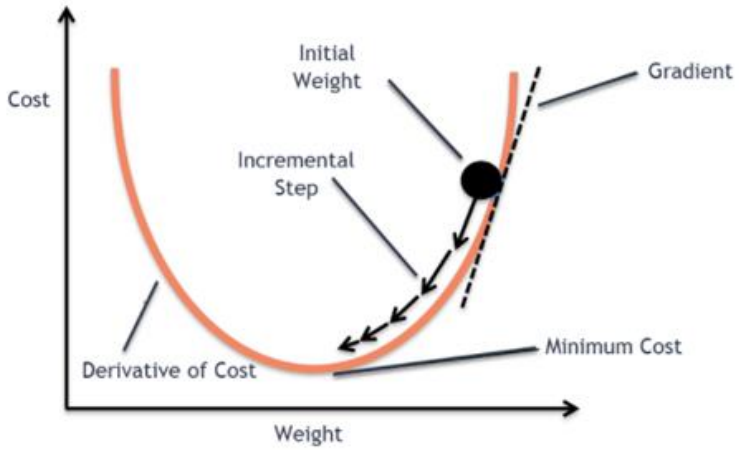
After making a hypothesis with initial parameters, we calculate the Cost function. And with a goal to reduce the cost function, we modify the parameters by using the Gradient descent algorithm over the given data. We define the cost function as the sum of squares of the estimate that our parameters/weights produce, and the value that we were expecting. The difference is squared, and the total quantity is later halved for optimization reasons.

Source: Coursera

What is Gradient Descent?

Gradient descent is an iterative optimization algorithm for finding the local minimum of a function.

The goal of the gradient descent algorithm is to minimize the given function (say cost function). To achieve this goal, it performs two steps iteratively:
Compute the gradient (slope), the first order derivative of the function at that point
Make a step (move) in the direction opposite to the gradient, opposite direction of slope increase from the current point by alpha times the gradient at that point



Gradient descent algorithm

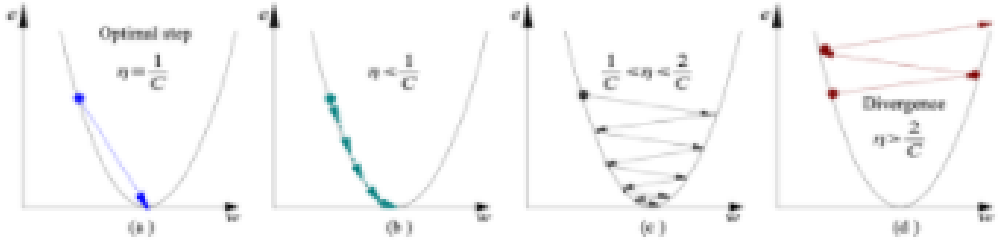
$$\text{repeat until convergence } \{$$
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$
$$(\text{for } j = 1 \text{ and } j = 0)$$
$$\}$$

Alpha is called **Learning rate** – a tuning parameter in the optimization process. It decides the length of the steps

Source: Coursera

Alpha – The Learning rate

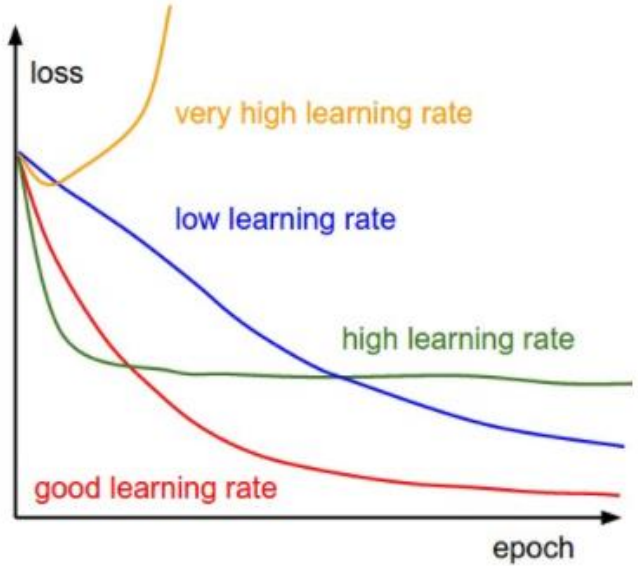
We have the direction we want to move in, now we must decide the size of the step we must take.
*It must be chosen carefully to end up with local minima.
If the learning rate is too high, we might OVERSHOOT the minima and keep bouncing, without reaching the minima
If the learning rate is too small, the training might turn out to be too long



1.a) Learning rate is optimal, model converges to the minimum
2.b) Learning rate is too small, it takes more time but converges to the minimum
3.c) Learning rate is higher than the optimal value, it overshoots but converges ($1/C < \eta < 2/C$)
4.d) Learning rate is very large, it overshoots and diverges, moves away from the minima, performance decreases on learning

Image Source: Coursera

Alpha Learning Rate



Source: researchgate

Note: *As the gradient decreases while moving towards the local minima, the size of the step decreases. So, the learning rate (alpha) can be constant over the optimization and need not be varied iteratively.*

- https://cs231n.github.io/neural-networks-3/ Stanford University