Gradient Descent

This lesson will focus on the intuition behind the gradient descent algorithm.

WE'LL COVER THE FOLLOWING

In the last lesson, we minimized a loss function to find the best model to predict the tip paid by customers. But there was a drawback with the approach. We manually entered the values of the model parameter θ and compared the losses. But this approach of manually choosing the values of the model parameters is not scalable because:

- It works only on predetermined values of θ
- Most models have many model parameters and complex structures of the prediction function, for which it will require a lot of time to choose parameters manually.
- We may not choose the best set of model parameters, and then we will not get to the best model.

We need some approach that chooses the model parameters automatically and then arrives at the best model.

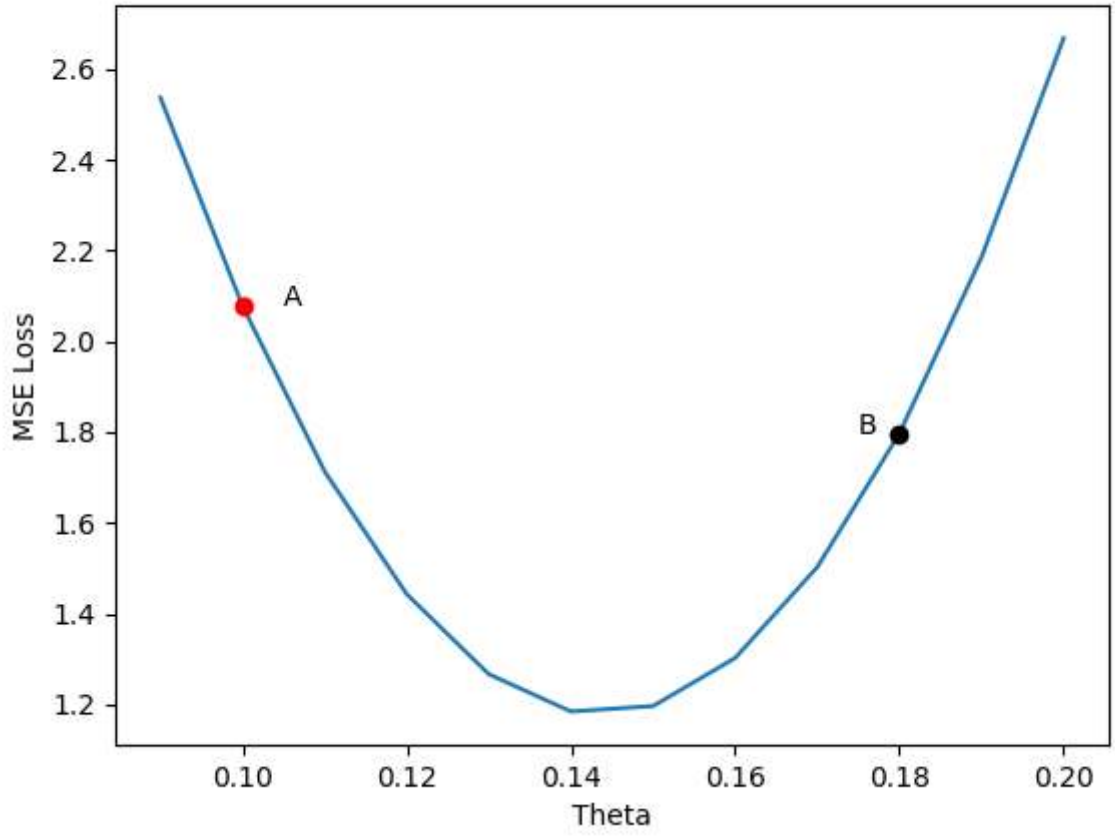
Intuition

Since we need a method in which we do not use predetermined values of θ, let's start by picking a random value of θ and see what our loss is. After this, we will decide whether to increase the current value of θ or decrease it and

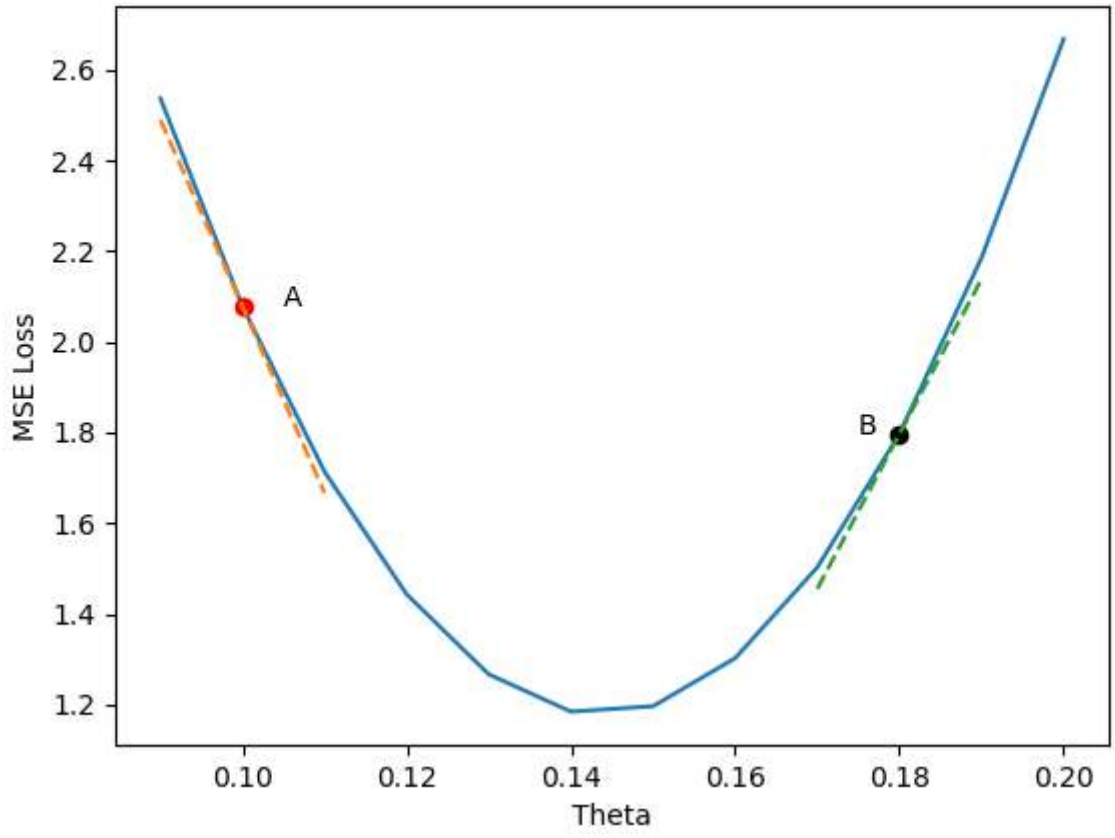
the amount to increase or decrease. Let's look at the direction and amount separately.

Direction of change in θ

Look at the error surface of the example in the previous lesson below.



We have highlighted two points in this curve. The red highlighted point A is the value of the loss at $\theta = 0.10$. If we choose this as our starting point for θ, then at this point, we need to choose a new value for θ that is closer to the minimum of this curve. Let's look at the slope of the line at this point.



The slope at point A is **negative**, which means that if we increase θ from this point, the loss will decrease. Therefore, we need to *increase* the value of θ from here to reach the minimum of the curve.

Now let's consider another situation where we start at $\theta = 0.18$. This point B is highlighted in black. Looking at the slope of the line at point B, we can see that the slope is **positive**, which means that if we increase θ from this point, the loss will increase. Hence, we need to *decrease* θ from here to reach the minimum.

Therefore, we establish the rule that if the slope is **negative**, we need to *increase* the value of θ, and if the slope is **positive**, we need to *decrease* the value of θ. So, now that we have established the direction in which we need to move, the question of what amount should we move remains.

Amount of change in θ

The natural approach would be to increase or decrease θ by the amount of gradient at this point. Since we need to compute the gradient to capture the direction of the slope, why not increase or decrease by the gradient amount

itself? Let's see how we will do this mathematically.

If at instant t, we call our model parameter θ_t, then at the next instant $t+1$, we call it θ_{t+1}. To update the value of θ we do,

$$\theta_{t+1} = \theta_t - \frac{\partial}{\partial \theta} L(\theta_t, Y)$$

where $L(\theta_t, Y)$ is the loss function which depends on the values of θ and the set of actual values Y to give us a loss value. $\frac{\partial}{\partial \theta}$ means that we are taking the partial derivate of the loss function with respect to θ. We subtracted the gradient instead of adding to apply the rule we made above of going opposite to the direction of the gradient.

We multiply the gradient term with a constant α which ranges from 0 to 1, so that we can control the rate at which we update the model parameter θ. This term is known as the **learning rate**. Hence, the above expression becomes

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial}{\partial \theta} L(\theta_t, Y)$$

Hence, we have come up with a way to update the model parameters based on how they perform. This procedure is known as **Gradient Descent**.

Gradient Descent

It is an algorithm to update the values of model parameters to find the best model. It works as

- Start with random initial value of θ.
- Compute $\theta_t - \alpha \frac{\partial}{\partial \theta} L(\theta_t, Y)$ to update the value of θ.
- Keep updating the value of θ until it stops changing values. This can be the point where we have reached the minimum of the error function.

The process of using algorithms like gradient descent to minimize a function is sometimes called **optimization**.

This was the gradient descent algorithm. In the next lesson, we will implement it in our example of predicting tips given by the customer.