Assignment - 6

Name: Utkarsh Farkya

Referral ID: SIRSS1114

Q1. Calculate/ derive the gradients used to update the parameters in cost function optimization for simple linear regression.

Simple linear regression is a supervised learning model where a straight line is used to fit to the trend of the data such that the predictions made by the learned model has maximum possible accuracy. The hypothesis used by this regression model has only one feature (input variable) and two parameters that needs to be optimized such that the cost of the function is minimized. The algorithm that we use to minimize the cost function is called gradient descent.

Gradient Descent Algorithm:

Gradient descent algorithm is an iterative optimization approach to find the minima of a given differentiable function. Gradient descent is simply used to find optimal values of the parameters (coefficients) of an equation in such a way that minimizes the value of cost function as much as possible. It is similar to person walking down a valley deciding direction of walking based on his current location. Taking large steps when the slope is steep and small steps when slope is gentle and when he reaches down he stops moving. Similar thing happens in gradient descent where if the value of gradient is large steps are taken to optimize the value of parameters and as the value of gradient reduces small steps are taken to optimize the parameters.

The hypothesis used in simple linear regression:

$$Y' = \Theta_0 + \Theta_1 x_1$$

Here, Y' is the calculated (predicted) output from the hypothesis function and Θ_0 and Θ_1 are the parameters which needs to be optimized and X_1 is the feature or input variable to the function. This function is known as the hypothesis function that is used to predict values for given input.

To check the accuracy of our model a cost function is defined with the help of hypothesis function which gives us a measure of the accuracy of our model. Cost function is nothing but the mean square error (MSE) of different between the predicted value Y' and the actual value Y for a given set of training samples. Cost function is defined as:

$$J\,(\Theta) = \frac{\sum_{i=1}^{m} (Y' - Y)^2}{2m}$$

J(Θ) is said to be the representation of cost function, **summation** refers to sum of squares of difference between predicted and actual value of all training examples, **m** refers to the number of training examples. The general equation for gradient descent looks something like this:

$$\Theta_j = \Theta_j - \alpha \frac{\delta\, J(\Theta)}{\delta\, \Theta j}$$

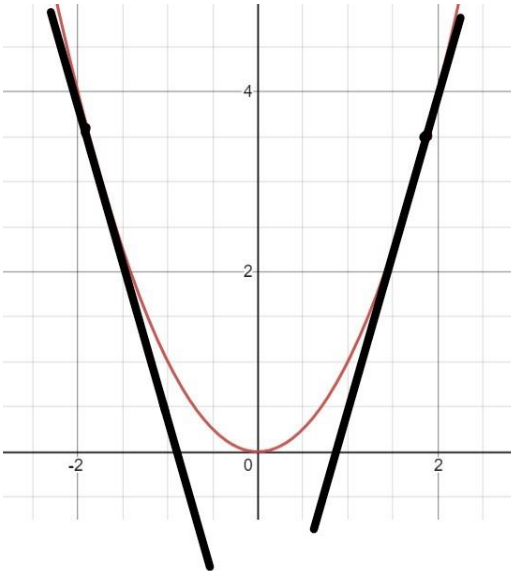
Here Θ_j refers to every parameter that is present in the hypothesis function and the last term in colour red is referred as gradient which is partial differential of the cost function w.r.t every parameter Θ_j, α is known as learning rate which decides the size of step. Since we have only two parameters, the partial differentiation of the cost function will be:

$$\frac{\delta\, J(\Theta)}{\delta\, \Theta 0} = \frac{\sum_{i=1}^{m}(Y'-Y)}{m} \qquad \frac{\delta\, J(\Theta)}{\delta\, \Theta 1} = \frac{\sum_{i=1}^{m} Xi\,(Y'-Y)}{m}$$

When substituted these values in gradient descent equations we get:

$$\Theta_0 = \Theta_0 - \alpha \frac{\sum_{i=1}^{m}(Y'-Y)}{m} \qquad \Theta_1 = \Theta_1 - \alpha \frac{\sum_{i=1}^{m} Xi\,(Y'-Y)}{m}$$

Q2. What does the sign of gradient say about the relationship between the parameters and cost function?



Assume this as a plot of cost vs. Θ_1 where cost is on y-axis and parameter-1 is on x-axis. The plot between cost and parameter is always convex sort of curve where there is a point when cost is minimum. Let assume $\Theta_1 = 2$, the gradient/slope at that point is positive and to decrease cost we have to decrease the value of Θ_1 as well. That means when gradient is positive there is direct relation of decreasing the value of Θ_1 to reduce cost as well. Similarly, when $\Theta_1 = -2$, we get negative slope which indicates inverse relation that means to decrease the cost we need to increase the value of Θ_1. The same can be seen from the gradient descent equation of Θ_1 as well which is:

$$\Theta_1 = \Theta_1 - \alpha \frac{\sum_{i=1}^{m} Xi\,(Y'-Y)}{m}$$

The red part in equation is the gradient which when positive reduces the value of Θ_1 and if negative increase the value of Θ_1.

Q3. Why Mean squared error is taken as the cost function for regression problems.

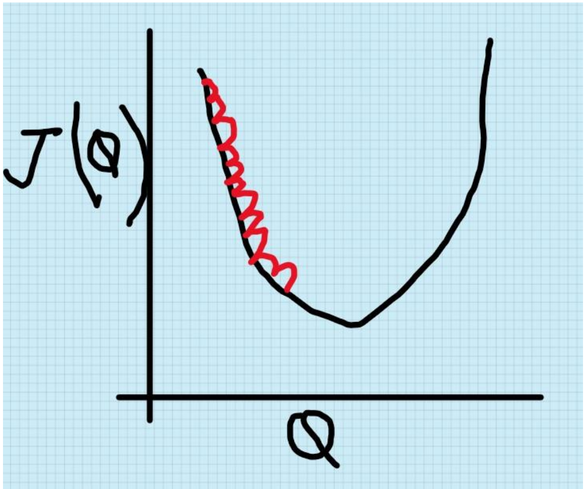
Cost function in regression problems is used to check the accuracy of our hypothesis function i.e. how accurate predictions are done by the hypothesis function. In regression problems MSE is taken into account as the cost function over MAE and mean average value. The reason of not considering mean average error as cost function is because the error might become zero because of positive and negative values added together and will lead to bad optimization. MAE is also used in predictive modelling but calculating the gradient in case of MAE requires complicated methods. Also because of square in MSE large errors have much more influence on MSE than smaller errors and of course finding gradient of square function is easier than the function used in MAE because of taking absolute values into account. Considering MSE as cost function as compared to cube or power 4 function, square function gives considerably small errors among all and calculating gradient of square function is much easier than other. There might be a possibility that using a cube function or some other degree function we might now be able to find minima which is the requirement of optimization process. These are some of the reasons of using MSE as compared to the other cost functions.

Q4. What is the effect of learning rate on optimization, discuss all the cases?

$$\Theta_1 = \Theta_1 - \alpha \frac{\sum_{i=1}^{m} Xi\,(Y' - Y)}{m}$$

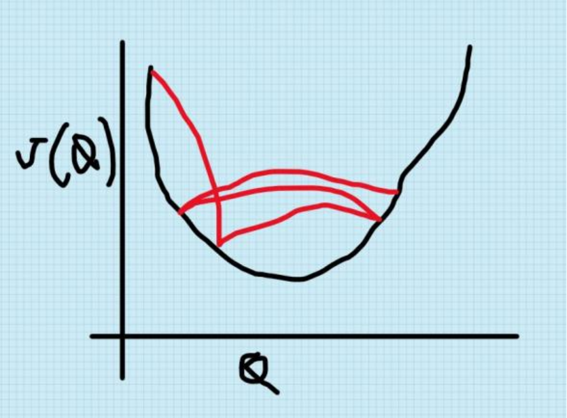
In the gradient descent optimization approach, alpha is the learning rate hyper parameter that decides how much the parameters are going to be updated in each iteration during training. There can be three cases of learning rate's effect on optimization process:

- Small learning rate



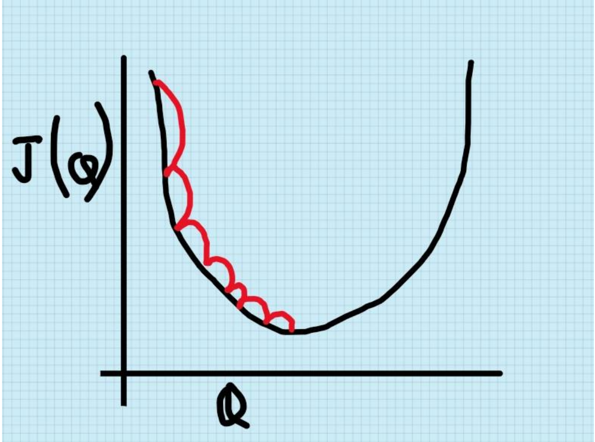
When learning rate is really small the optimization process takes a lot of time to get optimal values because the steps taken to update parameters are very small and it leads to longer duration for optimization process to complete.

- Large learning rate



When learning rate becomes large it leads to dangling or divergence situation because the steps taken to update parameters are really large because of which it doesn’t converge and might even lead to divergence of the parameters.

- Optimal learning rate



When the value of learning rate is optimal i.e. not too big and not too small the cost function is minimized rather quickly and steadily. A general range of alpha is considered between [0,1] so that our model does not suffer with divergence problem or taking too long to optimize the parameters.