**Understanding the optimizers**

**Introduction:** Optimizers play very important role in Machine learning. Understanding them will be helpful for us, helps in building the good machine learning model.

I will be covering most commonly used optimizers, which are used frequently in model building.

1. Gradient Descent
2. Stochastic Gradient Descent

**Gradient Descent:**

Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. In machine learning, we use gradient descent to update the parameters of our model. Parameters refer to coefficients in Linear regression and weights in neural networks.

**A close up of a map

Description automatically generated**

**Learning rate:**

The size of these steps is called the *learning rate*. With a high learning rate we can cover more ground each step, but we risk overshooting the lowest point since the slope of the hill is constantly changing. With a very low learning rate, we can confidently move in the direction of the negative gradient since we are recalculating it so frequently. A low learning rate is more precise, but calculating the gradient is time-consuming, so it will take us a very long time to get to the bottom.

## Cost function:

## A Loss function tells us “how good” our model is at making predictions for a given set of parameters. The cost function has its own curve and its own gradients. The slope of this curve tells us how to update our parameters to make the model more accurate.

## Step-by-step:

## Now let’s run gradient descent using our new cost function. There are two parameters in our cost function we can control: mm (weight) and bb (bias). Since we need to consider the impact each one has on the final prediction, we need to use partial derivatives. We calculate the partial derivatives of the cost function with respect to each parameter and store the results in a gradient.

**Given the cost function**:

**The gradient can be calculated as:**

To solve for the gradient, we iterate through our data points using our new ***m*** and ***b*** values and compute the partial derivatives. This new gradient tells us the slope of our cost function at our current position (current parameter values) and the direction we should move to update our parameters. The size of our update is controlled by the learning rate.

**Stochastic Gradient Descent:**

## The word “stochastic” means random probability. In SGD the method uses randomly selected (or shuffled) samples to evaluate the gradients. Here all training points are not used to get the gradients, only a single training point or subset of training point is used to get the gradients.

## Step-by-step:

## Choose an initial vector of parameters {\displaystyle w}*w* and learning rate *n* {\displaystyle \eta }.

## Repeat until an approximate minimum is obtained

## Randomly shuffle examples in the training set.

## For *i* = 0, 1, 2 . . . . . . . n do:

## .

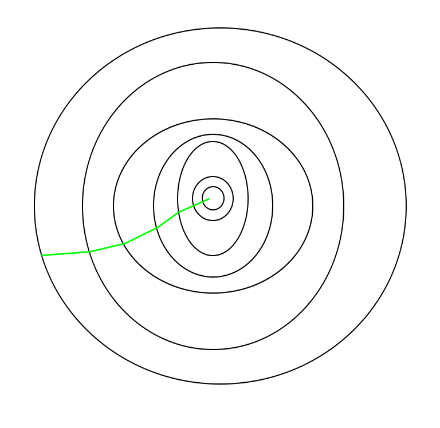
## Here *w* =

**Advantages**:

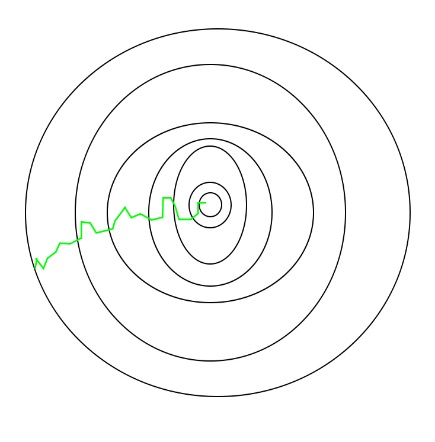
computationally very fast when we have very large data compared to gradient descent.

Now let us see how it looks.

1. Path taken by Batch Gradient Descent.



1. Path taken by the SGD.



## By seeing above pictures, we can see that noise is more in SGD to reach minimal, as it makes more adjustments.

## Well now I will tell you the beautiful intuition, which I read in Quora which helps in remembering this.

**How to remember this?**

Let’s say you are about to start a business that sells t-shirts, but you are unsure what are the best measures for a medium sized one for males. Luckily you have gathered a group of men that have all stated they tend to buy medium sized t-shirts. Now you figure you're going to use a gradient descent type method t get the size just right.

**Batch Gradient Descent**

1. Tailor makes initial estimate.
2. Each person in the batch gets to try the t-shirt and write down feedback.
3. Collect and summarize all feedback.
4. If the feedback suggests a change, let the tailor adjust the t-shirt and go to 2.

**Stochastic Gradient Descent**

1. Tailor makes initial estimate.
2. A random guy (or a subset of the full group) tries the t-shirt and gives feedback.
3. Make a small adjustment according to feedback.
4. While you still have time for this, go to 2.

**Final take out**

* Batch gradient descent needs to collect lots of feedback before making adjustments but needs to do fewer adjustments.
* Stochastic gradient descent makes many small adjustments, but spends less time collecting feedback in between.
* Batch gradient descent preferable if the full population is small, stochastic gradient descent preferable if the full population is very large.

Hope this helps in understanding the basic optimizers, had a great time writing this cheers 😊

## 