Basics and Beyond: Gradient Descent

Decoding gradient descent step by step

Kumud Lakara · Follow
Published in The Startup · 6 min read · Dec 25, 2020

Photo by Jake Hills

This post aims to take you from the very basics to advanced concepts in gradient descent. When starting off with machine learning, Gradient Descent is probably one of the initial concepts one comes across. Let's get started!

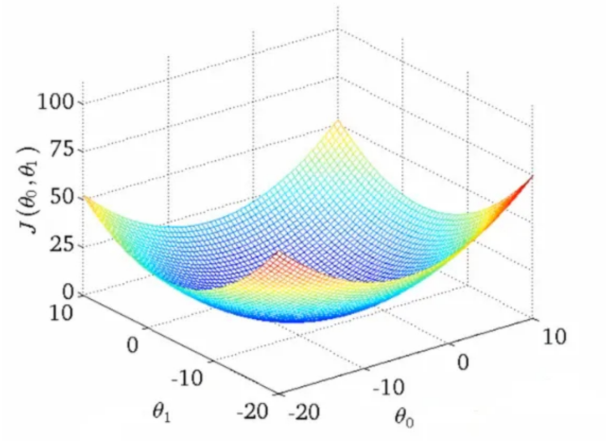
All machine learning algorithms require an optimization algorithm and Gradient Descent is just that. It is an optimization algorithm that aims to adjust parameters in order to minimize the cost function .

$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)

Cost function is a function of the parameters used to evaluate quality of predictions

Lets think about it this way, imagine a bowl. Any point on this bowl is the current cost and our aim is to reach the bottom of the bowl which is called the "global optimum". This is exactly what gradient descent tries to achieve. It selects parameters, evaluates the cost and then adjusts these parameters so as to get a lower cost than the previous one hence inching a step closer to the minimum. Once we reach the global minimum or intuitively the bottom of the bowl we will have the best parameters for our hypothesis function and hence be able to make accurate predictions.



Plot of cost vs parameters

How It Works

Alright, now we have a basic idea of what gradient descent is so now lets take a look at how it works.

Well we start off by initializing our parameters. This initialization can be zero value or a small random value.

After this we calculate the cost function by plugging in the values of parameters. Now, we find the derivative of the cost function. But wait...why? Well our aim with gradient descent is to reach the global minimum and continuing our bowl analogy we always want to be moving in the direction that takes us closer to the "bottom of the bowl" or the global minimum. Finding the derivative of the cost function simply gives us the direction in which we need to move. Mathematically, the derivative gives us the slope and hence we can use it to find the direction in which we need to "move" or actually the direction in which we need to update our parameters so that we arrive at minimum cost.

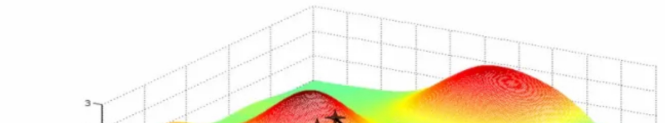
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(for } j = 0 \text{ and } j = 1\text{)}$$

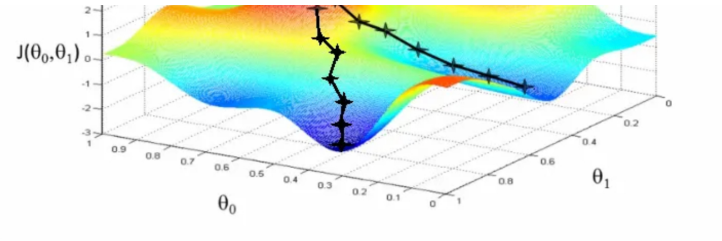
Equation for gradient descent

$$\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

The derivative of the cost function

Now that we have a direction to walk in the next thing we need to decide is how big of a step do we want to take in that direction. Remember that we don't know where the actual global minimum exists, we are merely trying to analyze our surrounding slopes and trying to choose the one that could potentially get us there. Since we are not very certain we don't want to take extremely confident steps or in our context: we don't want to update our parameters in the (supposedly) correct direction by a very overconfident or optimistic margin or step-size as we call it. This is where Learning Rate comes in.





Different possible paths to reach minimum which may be local or global

Learning Rate

Honestly, learning rate can be a post of its own but our purpose here is to focus on learning rate keeping gradient descent as our main point of interest. So, in simple words learning rate is a hyper-parameter that controls how big of a step we need to take when doing gradient descent. Step here means updating our parameters in a certain direction.

It is also quite clear from the name that it is in fact the rate at which your model can be intuitively assumed to be learning at.

We have already seen the basic equation for gradient descent:

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

α here is the learning rate. We see that it is a multiplying factor to the derivative of the cost function. This is the exact mathematical form of our intuition so far.

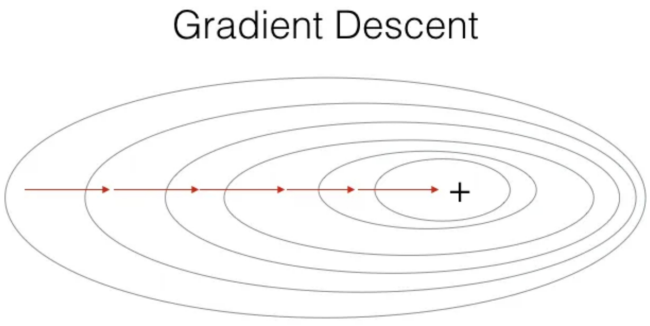
One thing to keep in mind with regard to learning rate is that if it is too low then gradient descent can be slow to converge and if its too large, gradient descent can overshoot the minimum. It may fail to converge or may even start diverging.

Now that we have a basic idea of what gradient descent is and how it works lets take a look at the different types of gradient descent:

1. Batch Gradient Descent

This is vanilla gradient descent which uses all the training examples to make a single update. This means that in order for this algorithm to take a single step in a certain direction it uses all the training examples in the data-set to make the update.

It becomes quite obvious that if our data-set is very large then this form of gradient descent can be computationally expensive.

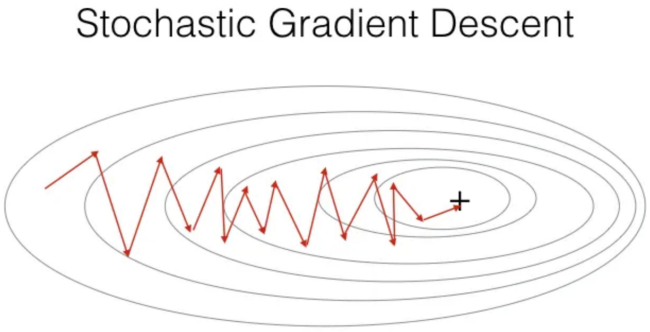


Batch gradient descent

2. Stochastic Gradient Descent

In cases where we have a large number of training examples we can make use of stochastic gradient descent or SGD which can prove to be computationally lighter.

SGD is different from batch gradient descent in the sense that it makes an update to the parameters or it takes a single step after *each* training example. Contrary to batch gradient descent SGD will evaluate the cost function, find the derivative and make the update for every training example instead of the whole data-set.



Stochastic gradient descent

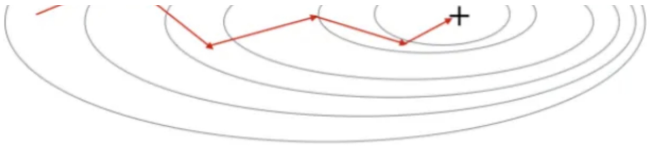
The process for updating is the same but the cost is calculated over one training example and not the entire data-set. In the case of large datasets the learning can be faster with SGD as compared to batch gradient descent.

It is also not very difficult to understand why SGD is faster than batch gradient descent. Batch gradient descent performs redundant computations for large datasets as it finds the gradients for similar values which are very close to each other and then makes an update by considering all the training examples in the data-set. SGD avoids this by making updates after every training example. SGD might also overshoot because its updates have a high variance. However it has been demonstrated that with a degrading learning rate SGD converges in a way quite similar to batch gradient descent.

3. Mini-batch Gradient Descent

This is actually the best of both worlds. It accounts for the computational expenses in case of batch gradient descent and the high variance in case of SGD. Mini-batch gradient descent is usually the method of choice when it comes to gradient descent optimization algorithms.

It works just like batch gradient descent but obviously with “mini” batches. Instead of making updates after evaluating cost and derivative for the *whole* data-set like in batch gradient descent or by considering *only one* training example like stochastic gradient descent; it makes the updates by considering a smaller batch instead.



Mini-batch gradient descent

Common mini-batch size range is from 50–250 but it usually depends on the application. It is also interesting to note that batch gradient descent and SGD are in fact just the two extremes of mini-batch gradient descent.

When our mini-batch size is the total number of training examples then it becomes batch gradient descent and when the min-batch size is 1 then it becomes stochastic gradient descent.

That's it!

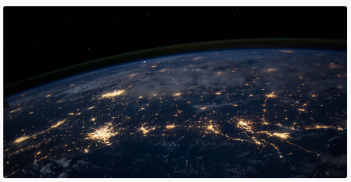
Congratulations on making it to the end of the post. You should now have a pretty good understanding of what gradient descent is and how it works. I suggest you look at some implementations of machine learning algorithms where you can see gradient descent in action.

. . .

References

1. https://www.holehouse.org
2. https://machinelearningmastery.com/gradient-descent-for-machine-learning/
3. https://ruder.io/optimizing-gradient-descent/
4. https://www.coursera.org/learn/machine-learning/home/

Machine Learning | Aritificial Intelligence | AI | Deep Learning | Technology



Kumud Lakara in Towards Data Science

Coding Linear Regression from Scratch

Implementing Linear Regression from absolute scratch using python

9 min read · Jan 5, 2021

343 7

Neeramitra Reddy in The Startup

3 Advanced (and Unique) ChatGPT Uses You've Likely Not Seen Before

Valuable "meta" use cases I've found in 10 months of tinkering with ChatGPT

11 min read · 5 days ago

3.7K 51

Jano le Roux in The Startup

ChatGPT Just Silently Rolled Out A (No-Code) Feature That Will Wrec...

What would you do if it was your startup?

4 min read · Nov 9

2.2K 43

Kumud Lakara in Analytics Vidhya

Basics and Beyond: Linear Regression

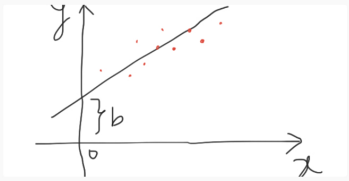
This post will walk you through linear regression from the very basics. When...

8 min read · Dec 26, 2020

264 1

See all from Kumud Lakara | See all from The Startup

Recommended from Medium



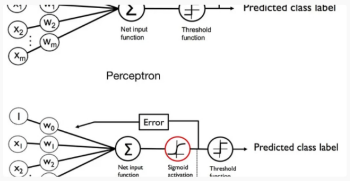
Dilip Kumar

Linear Regression Model using Gradient Descent algorithm

What is Machine learning?

4 min read · Jul 4

11



AmyChu

#11 What is a linear classifier (Logistic Regression)

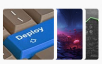
The previous Perceptron can successfully achieve binary classification, but it can only...

5 min read · Aug 20

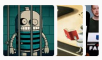
1

Lists

The New Chatbots: ChatGPT, Bard, and Beyond
12 stories · 226 saves

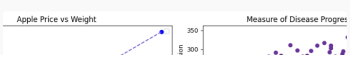


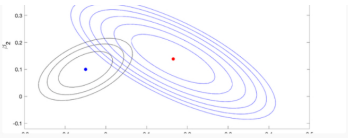
Predictive Modeling w/ Python
20 stories · 658 saves



AI Regulation
6 stories · 208 saves

Generative AI Recommended Reading
52 stories · 463 saves





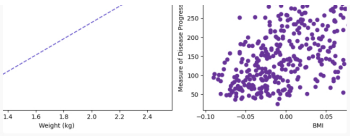
Tahera Firdose

Ridge Regression -Derivation and code from Scratch

Ridge Regression is a linear regression technique that incorporates a regularization...

6 min read · Jun 27

3



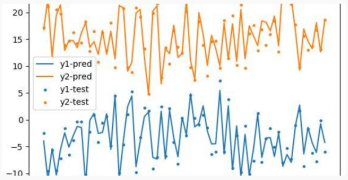
Mohammad Jasir in Python in Plain English

Understanding and Implementing Linear Regression | From Scratch

Every regression story starts with a dataset. The first thing a data scientist looks into afte...

7 min read · Oct 19

17



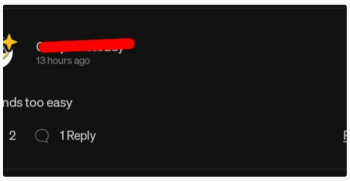
Francesco Franco

Multioutput Regressions with SVMs in Python

Support Vector Machines can be used for performing regression tasks- we know that...

10 min read · Oct 18

15



Paul Rose

I Found A Very Profitable AI Side Hustle

And it's perfect for beginners

6 min read · Oct 18

12.6K 221

See more recommendations