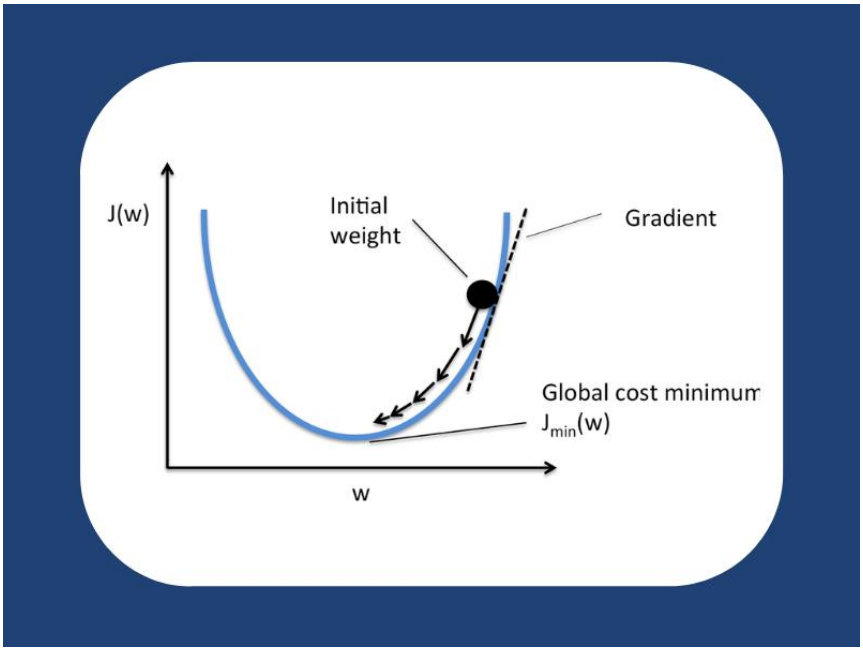
DataJerms.

17 Unique Machine Learning Interview Questions about Gradient Descent



Introduction

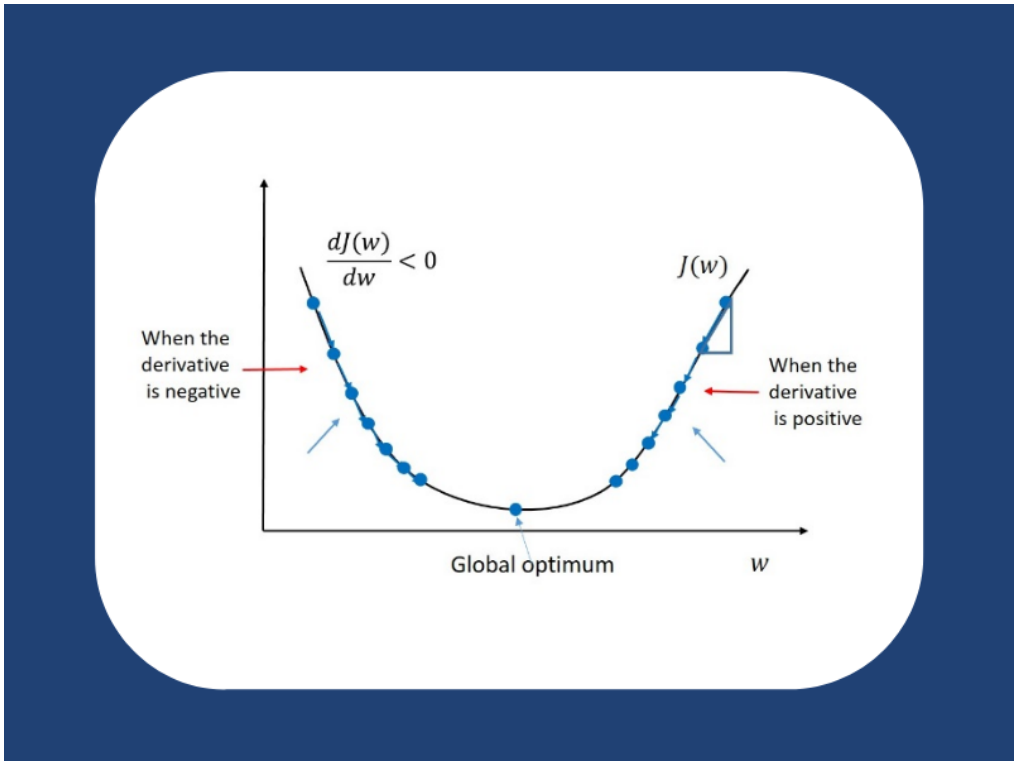
In life, sometimes we come across some algorithms that just make sense. The type whose logic is just one that anyone and everyone can relate to. Machine Learning has its complications and overwhelming technicalities. However, the Gradient Descent algorithm is not one of them. In fact, it falls in the basics of anyone even thinking of taking their talents to the field of Artificial Intelligence. The word 'optimization' will make much more sense to you as an ML enthusiast once you explore what it is.

Article Overview

- What is the Gradient Descent Algorithm?
- How does the Gradient Descent Algorithm work?
- Why do we need to Use the Gradient Descent algorithm?
- Gradient Descent algorithm ML Interview Questions/Answers

What is the Gradient Descent Algorithm?

Gradient Descent is an algorithm which is used to train most Machine Learning models and Neural Networks. It is the algorithm that reduces the error in the cost function using the training data. In doing so, it optimizes the model by increasing its accuracy and updating its parameters so that they result in the smallest possible error. It basically finds the overall minimum of the cost function, or any function.



How does the Gradient Descent Algorithm work?

The working of Gradient Descent is quite simple to understand. Let's use the scenario of performing linear regression to understand the steps involved in gradient descent. Recall the equation **y = mx + b.**

1. Initially let the gradient m and y-intercept c equal to zero. Let L be the learning rate. L should be a small value like 0.0001 for good accuracy.
2. Then, compute the partial derivative of the loss function with respect to m. Now plug in the current values of x, y, m and c in it to obtain the derivative value D.
3. Here the actual optimization part takes place where we update the current value of m and c using the following equations:

$$m = m - L \times D_m$$

$$c = c - L \times D_c$$

4. We then repeat the above until the calculated gradient becomes very small or even 0. Finally, the value of m and c that we are left with now will be optimized to their peak.

Why do we need to Use the Gradient Descent algorithm?

The reason is simple – computational efficiency. It gets the work done faster and cheaper than an approach like the Linear Algebra one. In fact, there are also iterations and types of the Gradient Descent algorithm that allow parallel (simultaneous) calculations to take place and approaches that further enhance the computational time.

Gradient Descent algorithm ML Interview Questions/Answers

After having a look at all the above, it should be safe to say that Gradient Descent is quite vital for any Data Scientist or ML related person to have knowledge of. Henceforth, let us dive into interview questions related to it. The above should have already got anyone excited regarding Gradient Descent and their importance and use. Therefore, it makes perfect sense for us to now transition onto interview questions related to them. The above should have already got anyone excited regarding Support Vector Machines and their importance and use. Therefore, it makes perfect sense for us to now transition onto interview questions related to them. Try to answer them in your head before clicking the arrow to reveal the answer

1. What is the difference between the Gradient Descent method and the Ordinary Least Squares method? Which is better?

In Gradient descent:

- We need to perform hyper-parameter tuning for the learning parameter.
- We need to iterate.
- We have a time complexity of O(kn2)

in Ordinary Least Squares:

- There is no need for any hyperparameter.
- No iteration is needed.
- We have a time complexity of O(n3)

Looking at the above differences, it is evident that the Ordinary Least Squares method is the better and swifter option for a smaller training data (smaller n). However, for larger n (larger training data), due to the higher time complexity of the Ordinary Least Squares method, the Gradient Descent algorithm must be preferred.

2. What are the different types of Gradient Descent methods?

The methods are:

- Batch Gradient Descent: where the entire training data is taken into account to take a single step. Its cost function is convex and relatively smooth.
- Stochastic Gradient Descent: where only one data is taken into account to take a single step. Its cost function is a fluctuating one whose fluctuations eventually decrease as the number of iterations pass.
- Mini-batch Gradient Descent: where a batch of a fixed number of data is considered. Its cost function is also a fluctuation one.

3. What is the difference between Gradient Descent and Cost Function?

The Cost Function is something that we intend to minimize. For example, our cost function might be the sum of squared errors over the training set. However, the Gradient Descent is a method or algorithm for finding the minimum of that cost function of multiple variables.

4. What is the Learning Rate in Gradient Descent Algorithms?

Learning Rate is the value used to update the bias and weights after the derivatives have been found out in Gradient Descent.

5. Is it better to use a larger Learning Rate or a smaller one?

Using either type of Learning Rate could be beneficial so long as a balance is maintained whilst keeping the points below in mind:

- A small learning rate may lead to the model taking a longer time to learn
- A large learning rate will make the model converge as our pointer will shoot and we may not be able to get to a minima

6. Can the Cost Function ever have a value of 0?

Yes, for sure, but it does not necessarily mean that the model being used is the best.

7. What is the difference between Backpropagation and Gradient Descent?

Backpropagation is the process of computing the derivatives while Gradient Descent is the process of descending through the gradient to minimize the loss and update the relevant model parameters.

8. Name a type of Error Function used during Gradient Descent.

MSE – Mean Squared Error function

9. How does Gradient Descent Power Neural Networks?

- The neural network is initialized with a set of random weights.
- We ask the neural network to make a prediction on given data points from a training set.
- We compute the prediction and then the loss/cost function which tells us how good or bad of a job our neural network did at making predictions.
- We compute the gradient of the loss.
- We then ever-so-slightly tweak the parameters of the neural network such that our predictions are better.

10. What approaches can we take to lessen the effects of overfitting?

- Regularization: L1, L2 regularization (commonly called weight decay), ElasticNet
- Adjustment of learning rate
- Dropout (technically a type of regularization)
- Implicit Regularization: Early Stopping, Data Augmentation
- Batch Normalization

11. What is the difference between a global minimum and a local minima?

Local minima are peaks (troughs) across a landscape that represent many small regions of loss. Our global minimum is the local minimum with the least loss over the loss landscape. This point ensures our parameters take on the most optimal solutions.

12. What are the problems with Vanilla Gradient Descent?

- If the number of training samples is large, then the vanilla gradient descent is going to take a long time to converge because a weight update is only happening once per data cycle.
- The larger your dataset, the more nuanced your gradients become, the more time related computation is used and eventually, there will not be much learning.

13. If we have 7 independent variables and 4 feature interactions, how many of these terms will be included in the gradient calculation?

All (11) of them. Because each given feature will have its own coefficient which determines effect or relevance of the feature on the output predictions. Although Gradient descent can weaken or diminish or strengthen the effects of these features.

14. Outline Gradient Descent Optimization Algorithms

- Momentum: It helps accelerate Stochastic Gradient Descent in the relevant direction and dampens irregular movements of the SGD as it navigates to the local optimum.
- Nesterov Accelerated Gradient: This method works directly with Momentum. In layman's terms, it makes the descent smarter. Typically, the momentum first computes the gradient and then makes a movement in the direction of the updated accumulated current gradient. Nesterov Accelerated Gradient (NAG) also makes a movement (not as much) in the direction of the momentum (previous accumulated gradient), measures the gradient, and makes a correction which results in the complete NAG update. This anticipatory update prevents us from going too fast and results in increased responsiveness which is suitable for the models.
- AdaGrad: It allows the learning rate adapt to parameters, performing smaller updates (i.e. low learning rates) for parameters associated with frequently occurring features and larger updates (i.e. large learning rates) for parameters associated with infrequent features With all these stated, it is suitable for dealing with sparse data.
- Adadelta: It is the extension of Adagrad that seeks to reduce its aggressive, monotonically decreasing learning rate. Adagrad accumulates squared gradients which makes the learning rate shrink to uselessness. Adadelta restricts the window of accumulated passed gradients to a particular size.
- RMSProp: It is an adaptive learning rate method that stemmed from the need to resolve Adagrad's radically diminishing learning rates. RMSprop divides learning rate by an exponentially decaying average of squared gradients.
- Adaptive Moment Estimation (Adam): Adam is a method that computes adaptive learning rates for each parameter
- Nadam
- Adamax
- AdamW: This fixes weight decay in Adam

15. What are the merits and demerits of Batch Gradient Descent?

Advantages

- It is computationally efficient
- It has stable performance (less noise)

Disadvantages

- It requires a lot of memory
- It has a slower learning process
- It may become caught in local minima

16. What are the merits and demerits of Stochastic Gradient Descent?

Advantages

- It has faster learned on some problems
- The algorithm is simple to understand
- It provides immediate feedback

Disadvantages

- It is computationally intensive
- There's a definite probability it won't settle in the global minimum
- The performance will be very noisy
-

17. What are the merits and demerits of Mini-Batch Gradient Descent?

Advantages

- It avoids getting stuck in local minima
- It is more computationally efficient than Stochastic Gradient Descent (SGD)
- It does not need as much memory as Batch Gradient Descent (BGD)

Disadvantages

- Hyperparameter Tuning (batch size)
- It can be highly expensive and intractable for datasets that are too large to fit in memory.