International Journal for Research in Engineering Application & Management (IJREAM)
ISSN : 2454-9150 Vol-07, Issue-02, MAY 2021

Gradient Descent Algorithm an Efficient Approach for Minimizing Error

Ruchi Rathore Research Scholar M. Tech. (C.S.E.) 4th Semester Lakshmi Narain of College Technology Science (RIT) Indore M.P. India ruchir.lkct@gmail.com

Vasudha Sharma Assistant Professor Lakshmi Narain of College Technology Science (RIT) Indore M.P. India vasu.dixit2007@gmail.com

Abstract : Gradient descent is one of the most accepted algorithms to achieve optimization and by far the most general way to optimize. Gradient descent is an iterative optimization algorithm used to discover the minimum value for a function. Gradient Descent Algorithm helps to make these decisions efficiently and successfully, by applying the use of derivatives. Derivative is calculated as the slope of the graph for a particular point. The slope is described by drawing a tangent line to the graph at the point. So, we need to compute this tangent line, we might be able to compute the desired direction to reach the minima. Gradient Descent is the most common optimization algorithm in machine learning. It is a first-order optimization algorithm. We need to update parameters in the opposite direction on every iteration of the gradient of the objective function with respect to the parameters where the gradient gives the direction of the steepest ascent. In the paper we used Gradient descent to optimize the improve linear regression to prediction. With the help of Gradient descent we minimize error by using cost function.

Keywords — Gradient descent, Optimization, Minimize, Cost , Efficiently, Successfully

I. INTRODUCTION

Gradient descent is very popular and most accepted optimization algorithm. **Gradient descent** algorithm is an iterative algorithm to achieve optimization. **Gradient descent** algorithm provides optimization by the minimum value for a cost function. Gradient Descent Algorithm not only helps to make decision efficiently and but provides solution of the problem successfully. by applying use of derivatives. A **derivative** is an important term that comes from calculus. **Derivative** is used to calculate the slope of the graph for a particular point. The tangent line is used to describe the slop by drawing the graph at the point. So, we need to compute this tangent line, we might be able to compute the desired direction to reach the minima. Gradient Descent is the most common optimization algorithm in machine learning. This means it only takes into account the first derivative when performing the updates on the parameters. In mathematical terms, a gradient is a partial derivative with respect to its inputs. Imagine a blindfolded man who wants to climb to the top of a hill with the fewest steps along the way as possible. He might start climbing the hill by taking really big steps in the steepest direction, which he can do as long as he is not close to the top. As he comes closer to the top, however, his steps will get smaller and smaller to avoid overshooting it. This process can be described mathematically using the gradient [11,12].

Before working with gradient descent, it may need to know concepts from linear regression. The linear line is resented using following formula for the slope y = mx + b, in this formula where m represents the slope and b is the intercept on the y-axis. This formula is used in statistics to find the line of best fit, which required calculating the error between the actual output and the predicted output. The gradient descent algorithm works in similarly way, but it is based on a convex function, such as the one below[13,14],

The starting point can be any arbitrary point selected to evaluate the performance. From starting point, we find the derivative and we use a tangent line to observe the steepness of the slope. The slope helps us to inform that we need to updates weights and bias. If the slope at the starting point is steeper and new parameters are generated, we need to reduce the steepness gradually until it reaches the lowest point on the curve. Finding the line of best fit in linear regression in similar way, the goal of gradient descent is to minimize the cost function, or the error between predicted and actual y.

II. DIRECTION AND LEARNING RATE

A direction and a learning rate are two important things are required in gradient descent. These factors decide the partial derivative for future iterations. By following these factors we gradually arrive at the local or global minimum[15].

 DOI : 10.35291/2454-9150.2021.0301

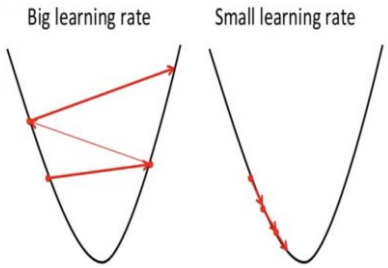

Figure 1 big and small leaning rate

Learning rate (also referred to as step size or the alpha) is the size of the steps that are taken to reach the minimum. This is typically a small value, and it is evaluated and updated based on the behavior of the cost function. High learning rates result in larger steps but risks overshooting the minimum. Conversely, a low learning rate has small step sizes. While it has the advantage of more precision, the number of iterations compromises overall efficiency as this takes more time and computations to reach the minimum.

The cost (or loss) function measures the difference, or error, between actual y and predicted y at its current position. This improves the machine learning model's efficacy by providing feedback to the model so that it can adjust the parameters to minimize the error and find the local or global minimum. It continuously iterates, moving along the direction of steepest descent (or the negative gradient) until the cost function is close to or at zero. At this point, the model will stop learning. Additionally, while the terms, cost function and loss function, are considered synonymous, there is a slight difference between them. It's worth noting that a loss function refers to the error of one training example, while a cost function calculates the average error across an entire training set.

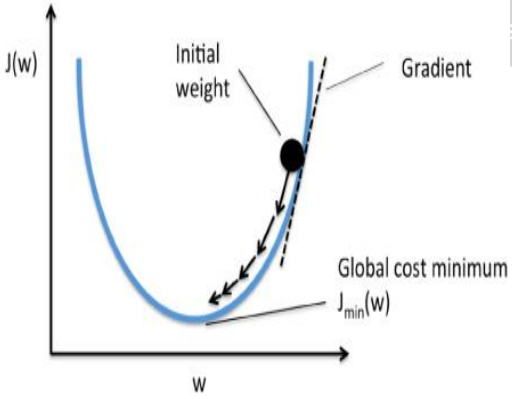

Figure 2 steps in gradient descent

III. VARIANTS OF GRADIENT DESCENT

There are three Variants of Gradient Descent are available [10,12]

1. Batch Gradient Descent
2. Stochastic Gradient Descent
3. Mini-Batch Gradient Descent

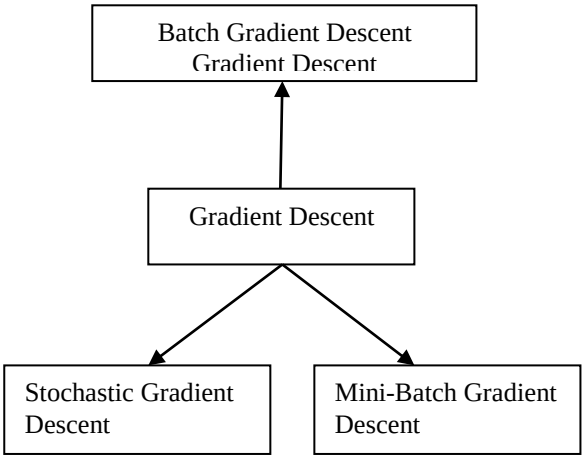

Figure 3 variants of Gradient Descent

A. Batch Gradient Descent

In batch gradient descent we use the complete dataset to compute the cost function. We need to perform just one update for whole dataset. Batch gradient descent is very slow and is inflexible to fit in memory.

B. Stochastic Gradient Descent (SGD)

Batch Gradient Descent is slower to perform faster computation, need to use stochastic gradient descent. The first we need to randomize the whole training set. In Stochastic Gradient Descent we use only one training example to compute the gradient of cost function.

C. Mini-Batch Gradient Descent

Mini batch algorithm is the most widely used algorithm. Mini batch algorithm gives faster results by using training examples. Standard mini-batch sizes range between 50 and 256. Sometimes, it can vary for different applications.

IV. LITERATURE SURVEY

In 2010 Leon Bottou proposed "Large-Scale Machine Learning with Stochastic Gradient Descent. They showed that a more precise analysis uncovers qualitatively different tradeoffs for the case of small-scale and large-scale learning problems. The large-scale case involves the computational complexity of the underlying optimization algorithm in non-trivial ways. They presented that optimization algorithms such as stochastic gradient descent show amazing performance for large-scale problems. In particular, second order stochastic gradient and averaged stochastic gradient are asymptotically efficient after a single pass on the training set[1].

In 2011 Feng Niu, Benjamin et al proposed "A Lock-Free Approach to Parallelizing Stochastic Gradient Descent". They presented an update scheme called Hog-wild which allows processors access to shared memory with the possibility of overwriting each other's work. They showed

 DOI : 10.35291/2454-9150.2021.0301

that when the associated optimization problem is sparse, meaning most gradient updates only modify small parts of the decision variable. The proposed algorithm takes advantage of sparsely in machine learning problems to enable near linear speedups on a variety of applications[2].

In 2012 Matthew D. Zeiler proposed "An Adaptive Learning Rate Method". They presented a novel per-dimension learning rate method for gradient descent called ADADELTA. They showed promising results compared to other methods on the MNIST digit classification task using a single machine and on a large scale voice dataset in a distributed cluster environment. They introduced a new learning rate method based on only first order information which shows promising result on MNIST and a large scale Speech recognition dataset[3].

In 2013 Hui Huang proposed "Faster gradient descent and the efficient recovery of images". They examined the effect of replacing steepest descent by a faster gradient descent algorithm in image deblurring and denoising tasks. They proposed several highly efficient schemes for carrying out these tasks independently of the step size selection, as well as a scheme for the case where both blur and significant noise are present. They showed that the effect of replacing steepest descent (SD) by a faster gradient descent algorithm, specifically, lagged steepest descent (LSD)[4].

In 2014 Fayrouz Dkhichi proposed "Neural Network Training By Gradient Descent Algorithms". They deal with the parameter determination of solar cell by using an artificial neural network trained at every time, separately. The training process is insured by the minimization of the error generated at the network output, from the outcomes obtained by each gradient descent algorithm, they conducted a comparative study between the overall of training algorithms in order to know which one had the best performances. With the help of result the Levenberg-Marquardt algorithm presents the best potential compared to the other investigated optimization algorithms of gradient descent[5].

In 2015 Dustin Tran proposed "Stochastic gradient descent methods for estimation with large data sets". They develop methods for parameter estimation in settings with large-scale data sets, where traditional methods are no longer tenable. The proposed methods rely on stochastic approximations, which are computationally efficient as they maintain one iterate as a parameter estimate, and successively update that iterate based on a single data point. Proposed method are numerically stable because they employ implicit updates of the iterates. This shrinkage prevents numerical divergence of the iterates, which can be caused either by excess noise or outliers[6].

In 2016 J V N Lakshmi "Stochastic Gradient Descent using Linear Regression with Python" They give python code for linear regression with stochastic gradient descent. The results and output showed for the code provided. The graph gives the cost function and the scatter plot drafts the dataset point in the plot. 'r' value is given for the correlated data. Information is mounting exponentially and hungry for knowledge. Linear Regression is a statistical method for plotting the line and is used for predictive analysis. Gradient Descent is the process which uses cost function on gradients for minimizing the complexity in computing mean square error[7].

In 2017 O. Majumder proposed "Learning the Learning Rate for Gradient Descent by Gradient Descent". They introduced an algorithm for automatically tuning the learning rate while training neural networks. They formalize this problem as minimizing a given performance metric at a future epoch using its "hyper-gradient" with respect to the learning rate at the current iteration. They presented a comparison between RT-HPO and other popular HPO techniques and show that our approach performs better in terms of the final accuracy of the trained model. They demonstrated that gradient-based HPO techniques like RT-HPO can be made to work and can even outperform current approaches[8].

In 2018 Difan Zou proposed "Stochastic Gradient Descent Optimizes Over-parameterized Deep ReLU Networks". They study the problem of training deep neural networks with Rectified Linear Unit (ReLU) activation function using gradient descent. They studied the binary classification problem and show that for a broad family of loss functions, with proper random weight initialization, both gradient descent and stochastic gradient descent can find the global minima of the training loss for an over-parameterized deep ReLU network, under mild assumption on the training data[9].

In 2019 Matthias Tschöpe "Beyond SGD: Recent improvements of Gradient Descent Methods". They train a model which calculates an output for a given input. The goal is to minimize the error between the actual output (also called label) and the estimated output from the model. Often, the output consists of a sum of differentiable functions.. To find good parameters for a given neural network, they used optimization algorithms. They briefly repeated the idea of Gradient Descent and show why it is equivalent to the steepest descent. They gave enough basics to explain Stochastic Gradient Descent, Momentum, Nesterov, AdaGrad, RMSprop, Adam, and AdamW[10].

In 2020 Yura Malitsky proposed "Adaptive Gradient Descent without Descent" . They presented a strikingly simple proof for two rules are sufficient to automate gradient descent, don't increase the step size too fast and

 DOI : 10.35291/2454-9150.2021.0301

don't overstep the local curvature. They showed that for functional values, no line search, no information about the function except for the gradients. These rules are used to create a method adaptive to the local geometry, with convergence guarantees depending only on the smoothness in a neighborhood of a solution. They examine its performance on a range of convex and non convex problems, including logistic regression and matrix factorization [16].

V. PROBLEM STATEMENTS

Gradient Descent is the most common optimization algorithm in machine learning. It has following problems

1. The objective is to continue to try different values for the coefficients, evaluate their cost and select new coefficients that have a slightly better (lower) cost
2. The objective is to optimization to deal with real life problems.
3. We perform optimization on the training data and check its performance on a new validation data.

VI. PROPOSED APPROACH

let's assume two parameters weight *w* and bias *b*.

1. Initialize weight *w* and bias *b* to any random numbers.
2. Pick a value for the learning rate α. The value of learning rate α determines step size.
3. If α is very small, it would obtain long time to meet and computationally expensive. If α is large, it may fail to meet and exceed.
4. Plot the cost function adjacent to different values of α. Decide the value of α that is right before the first value that didn't converge.
5. The most commonly used rates are : 0.001, 0.003, 0.01, 0.03, 0.1, 0.3.
6. On each iteration, take the partial derivative of the cost function with respect to each parameter. Continue the process until the cost function converges. That is, until the error curve becomes flat and doesn't change.

VII. CONCLUSION

Optimization techniques are performed on the training data and then the validation data set is used to check its performance. It is an iterative optimization algorithm used to find the minimum value for a function. Gradient Descent Algorithm helps us to make these decisions efficiently and effectively with the use of derivatives. In the proposed work we used Gradient descent for Optimization we apply random value for m and c. We continue apply iteration and we found that after some iteration we found best fitted regression line for the given data set. For practical implementation we used house size and house price data set. By the experimental analysis we found that the proposed approach give better regression line for the proposed data set and gets reduce error up to remarks

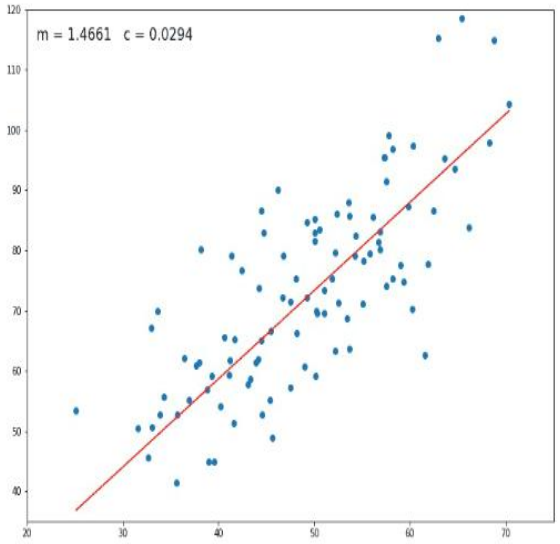

Figure 4 fitting regression line using Gradient Descent

REFERENCES

[1] Leon Bottou "Large-Scale Machine Learning with Stochastic Gradient Descent Proceedings of COMPSTAT'2010,DOI 10.1007/978-3-7908-2604-3 16, c Springer Verlag Berlin Heidelberg 2010.
[2] Feng Niu, "A Lock-Free Approach to Parallelizing Stochastic Gradient Descent" Computer Sciences Department, University of Wisconsin-Madison1210 W Dayton St, Madison June 2011.
[3] Matthew D. Zeiler, Adadelta: An Adaptive Learning Rate Method arXiv:1212.5701v1 Dec 2012.
[4] Hui Huang "Faster gradient descent and the efficient recovery of Images" arXiv:1308.2464v1 [cs.CV] 12 Aug 2013.
[5] Fayrouz Dkhichi "Neural Network Training By Gradient Descent Algorithms" International Journal of Innovative Research in Science, Engineering and Technology Vol. 3, Issue 8, August 2014.
[6] Dustin Tran "Stochastic gradient descent methods for estimation with large data sets" arXiv:1509.06459v1 [stat.CO] 22 Sep 2015
[7] J V N Lakshmi "Stochastic Gradient Descent using Linear Regression with Python" International Journal of Advanced Engineering Research and Applications (IJA-ERA) ISSN: 2454-2377 Volume – 2, Issue – 8, December – 2016.
[8] O. Majumder "Learning the Learning Rate for Gradient Descent by Gradient Descent" {orchid,donini,prtic}@amazon.com Amazon Web Services (AWS).

 DOI : 10.35291/2454-9150.2021.0301

[9] Difan Zou "Stochastic Gradient Descent Optimizes Over-parameterized Deep ReLU Networks" arXiv:1811.08888v3 [cs.LG] 27 Dec 2018.

[10] Matthias Tschöpe "Beyond SGD: Recent improvements of Gradient Descent Methods" https://www.researchgate.net /publication/334372835 July 2019.

[11] Khushbu Kumari, Suniti Yadav "Linear Regression Analysis Study" [Downloaded free from http://www.j-pcs.org on Friday, July 17, 2020, IP: 157.34.76.130]

[12] Samit Ghosal "Linear Regression Analysis to predict the number of deaths in India due to SARS-CoV-2 at 6 weeks" Diabetes & Metabolic Syndrome: Clinical Research & Reviews 14 (2020) 311e315journal homepage: www.elsevier.com/locate/dsx.

[13] Dokkyun Yi , SangminJi and Sunyoung Bu An Enhanced Optimization Scheme Based onGradient Descent Methods for Machine Learning Daegu University, Kyungsan 38453, Korea 8 June 2019; Accepted: 17 July 2019; Published: 20 July 2019.

[14] Jonathan Schmidt, Mário R. G. Marques , SilvanaBotti Recent advances and applications of machine learning in solidstatematerials science 26 February 2019 Accepted: 17 July 2019.

[15] Simon Shaolei Du Gradient Descent for Non-convex Problems in Modern Machine Learning APRIL 2019 CMU-ML-19-102 Machine Learning Department School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213.

[16] Yura Malitsky Konstantin Mishchenko Adaptive Gradient Descent without Descent Proceedings of the 37 th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

 DOI : 10.35291/2454-9150.2021.0301