Use of Taylor Series in Machine Learning Algorithms

1. Introduction to Taylor Series

1.1. Introduction

While conducting research for my extended essay in biology, I stumbled upon a model that predicted cell fate in human diseases using polynomials, derived from the Taylor series. Intrigued by this application of mathematics in biology, I began to research the Taylor series; the perfect amalgamation of functions, series and calculus. I then discovered that this theorem is applied in a multitude of fields such as machine learning, cryptography, and as aforementioned, biological models. As an enthusiast of these areas of knowledge, my curiosity regarding the theorem grew.

I quickly learnt that the Taylor series is used to approximate functions that play a key role in the machine learning procedure. In other words, the use of this series helps machines to become more accurate in their independent predictions of outcomes. In view of the increasing relevance of machine learning in the present day, there is an exponential increase in the significance of the Taylor series as well. This incited me to explore the uses of the Taylor Series in machine learning algorithms and learn more about how mathematics is revolutionizing the modern world.

1.2. Aim of Investigation

This investigation sets out to mathematically display and determine how the Taylor series is used in machine learning algorithms with focus on Gradient Descent.

1.3. Power Series

In mathematics, we commonly work with series involving integers. The Power Series, as shown below, can be thought of as an infinite series made up of terms involving a variable[1]. It converges at x = c and can take any form. This series can be used to represent, as well as define functions.

$$\sum_{n=0}^{\infty} c_n(x-a)^n = c_0 + c_1(x-a) + c_2(x-a)^2 + \ldots + c_k(x-a)^k + \ldots \quad (1)$$

1.4. Taylor Series

The Taylor series is a special type of power series solely defined for functions that are infinitely differentiable on an interval. As shown below, it is a method by which functions are expanded as an infinite sum of terms derived by the function's derivative at a particular point, a[2]. In other words, assume that we have a function $f(x)$ that is differentiable on a given interval. The Taylor series generated by $f(x)$ at $x = a$ is given by the formula below. With each term, the approximation becomes more accurate.

$$P_n(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \cdots + \frac{f^n(a)}{n!}(x-a)^n \quad (2)$$

The most commonly used Taylor series is known as the McLaurin series. As shown below, it generates $f(x)$ at $a = 0$.

[1] "10.1: Power Series and Functions." Mathematics LibreTexts, 11 July 2016, math.libretexts.org/Bookshelves/Calculus/Book%3A_Calculus_(OpenStax)/10%3A_Power_Series/10.01%3A_Power_Series_and_Functions.
[2] Saeed, Mehreen. "A Gentle Introduction to Taylor Series." Machine Learning Mastery, 19 Aug. 2021, machinelearningmastery.com/a-gentle-introduction-to-taylor-series/.

$$f(x) = f(0) + f'(0)\,x + \frac{f''(0)}{2!}\,x^2 + \frac{f^{(3)}(0)}{3!}\,x^3 + \ldots + \frac{f^{(n)}(0)}{n!}\,x^n + \ldots. \qquad (3)$$

1.7. Taylor Polynomials

The most important application of Taylor series is in the approximation of functions. Suppose that the function of interest is *f(x)* for *x* near a point *a*.

Example:

$f(x) = \frac{x^2 + 7}{x-3}$

$a = 5$

$f(a) = \frac{(5)^2+7}{5-3} = 16$

Such calculations are manageable. However, the computations of functions such as *sin(0.5)* are far more complicated. For these, a new function (*F(x)*) that is easier to work with, and is a fair approximation of *f(x)*, is formed using the Taylor series. The result is known as the Taylor polynomial: an approximation of the function of interest[3].

[3] "AC Taylor Polynomials and Taylor Series." Activecalculus.org, activecalculus.org/single/sec-8-5-taylor.html.

1.8. Error Approximation - Lagrange Error Bound

When approximating functions by means of the Taylor series, it is crucial to gauge the size of the error we may have introduced. As mentioned in *Chapter 1.7*, the original function is *f(x)* and our approximation is *F(x)*. The error in need of calculation will thus be the difference between the two: $f(x) - F(x)$. This difference is denoted by *R(x)*. Since we cannot perfectly deduce *R(x)*, the function gets bounded as shown below[4].

$$|R(x)| = |f(x) - F(x)| \leq M \qquad (4)$$

M in the above function is a small number that is in close range of the real error, $f(x) - F(x)$. However, calculating the error bound in this manner is not precise. Thus, *R(x)* must be redefined. As aforementioned, the Taylor approximation becomes more accurate with each term. This suggests that $P_{n+1}(x)$ is more accurate that $P(x)$.

$$P_{n+1}(x) = P_n(x) + \frac{f^{(n+1)}(a)}{(n+1)!}(x-a)^{n+1} \qquad (5)$$

Seeing that $P_{n+1}(x) - P_n(x) = \frac{f^{(n+1)}(a)}{(n+1)!}(x-a)^{n+1}$, the approximated error of $P_n(x)$ cannot be greater than the last term. Using this knowledge, a new definition of error *R(x)* can be formulated where *z* is the *x* value that yields the greatest derivative between *a* and *x*.

$$R(x) = \frac{f^{(n+1)}(z)}{(n+1)!}(x-a)^{n+1} \qquad (6)$$

[4] "Taylor Series - Error Bounds | Brilliant Math & Science Wiki." Brilliant.org, brilliant.org/wiki/taylor-series-error-bounds/.

Therefore, approximating error in Taylor polynomials is a three step process[5]:

1. Find the $(n + 1)^{th}$ derivative of $f(x)$
2. Calculate the upper bound of $f^{(n+1)}(z)$
3. Deduce $R(x)$

Although the Lagrange error bound has a standard calculation procedure, the type of error approximation conducted depends on the type of equation at hand. Polynomials of different degrees are approximated in varying manners.

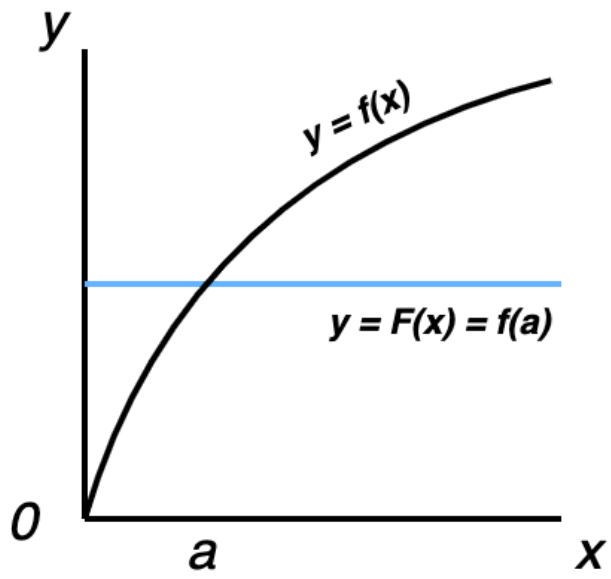
1.9. Types of Approximation

1. Constant Approximation

The first approximated function is one that is constant: a polynomial of degree zero. It will thus take the form, $F(x) = A$. If $F(x) = A$, then $F(a) = A = f(a) \Rightarrow A = f(a)$. Constant approximation is based on the rule that $f(x) \approx f(a)$. The below graph depicts the original and approximated function on the same plane.

[5] "The Error in the Taylor Polynomial Approximations." Personal.math.ubc.ca, personal.math.ubc.ca/~CLP/CLP1/clp_1_dc/ssec_taylor_error.html. Accessed 6 August. 2023.\

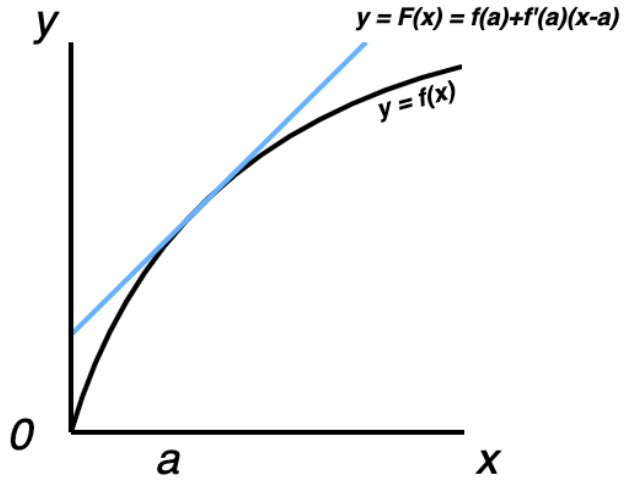
Graph 1.1: Constant Approximation



Here, *F(x)* horizontally interests f(x). As *x* moves away from point *a*, the accuracy of approximation greatly reduces. To ameliorate this, mathematicians conceptualized linear approximation.

2. Linear Approximation

Graph 1.2: Linear Approximation



In linear approximation, *F(x)* takes the form $A + Bx$. In addition to an intersection at $x = a$, *f(x)* and *F(x)* have the same gradient at point a. This further denotes that the functions have the same derivative at a[6].

Function	**First Order Derivative**
$F(x) = A + Bx$	$F'(x) = B$
$F(a) = A + Ba = f(a)$	$F'(a) = B = f'(a)$

Using the equations in the table above, it can be deduced that:

$F(x) = f(a) + f'(a)(x - a)$

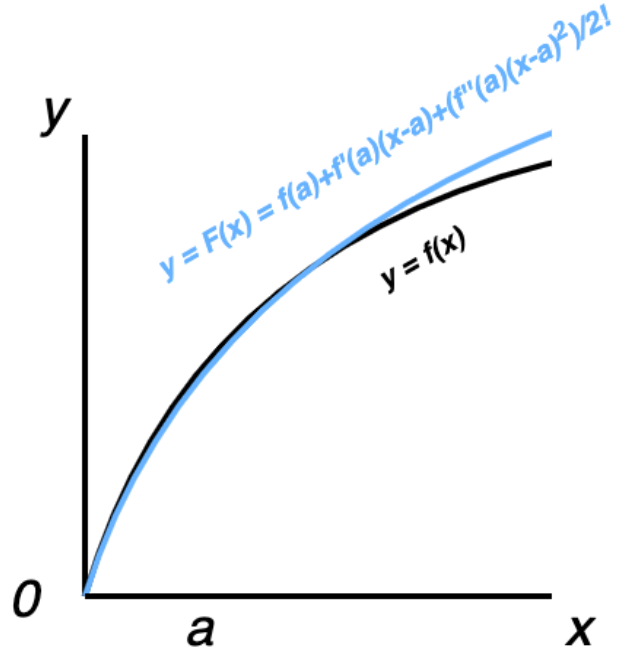
$\Rightarrow f(x) \approx f(a) + f'(a)(x - a)$

As shown in the above graph, this *F(x)* will form a tangent to *f(x)*. Nevertheless, this approximation can still be improved.

[6] Weisstein, Eric W. "Taylor Series." Mathworld.wolfram.com, mathworld.wolfram.com/TaylorSeries.html.

3. Quadratic/Polynomial Approximation

Graph 1.3: Polynomial Approximation



In quadratic approximations, *F(x)* takes the form of $A + Bx + Cx^2$. The condition at play here is that $f''(a) = F''(a)$.

Function	**First Order Derivative**	**Second Order Derivative**
$F(x) = A + Bx + Cx^2$	$F'(x) = B + 2Cx$	$F''(x) = 2C$
$F(a) = A + Ba + Ca^2 = f(a)$	$F'(a) = B + 2Ca = f'(a)$	$F''(a) = 2C = f'(a)$

Using the equations in the table above, it can be deduced that:

$$F(x) = f(a) + f'(a)(x - a) + \frac{f''(a)(x-a)^2}{2!}$$

$$\Rightarrow f(x) \approx f(a) + f'(a)(x - a) + \frac{f''(a)(x-a)^2}{2!}$$

This working helps prove the Taylor series, and increases the accuracy of the previous approximations. It can thus be gathered that higher order derivatives thus produce more accurate approximations[7].

2. Taylor Series and Machine Learning

2.1. Machine Learning and Gradient Descent Algorithm

Machine learning (ML) is the field of study that enables computers to imitate human learning patterns by means of models and algorithms, rather than explicit programming. Over the last few years, the advancement of ML has provided us with tools such as speech recognition and email spam filtering that we use on a regular basis. This attests to the utmost relevance of this field in the present day. The algorithms used in ML predict values based on the data they are fed[8]. As new data is inputted, they optimize their operations and help computers perceive more complicated datasets by identifying patterns in past data. Therefore, the information processing and decision making of machines improve with time and the introduction of various types of data. A series of algorithms is known as a neural network. Such networks are trained using optimization algorithms such as Gradient Descent.

Gradient Descent helps machines increase their efficacy by the minimization of prediction error[9]. It is commonly used to find the minimum of the loss function (differentiable and convex) by iterative optimization. The loss function calculates the difference between actual and expected

[7] Banerjee, Amarabha. "Iterative Machine Learning: A Step towards Model Accuracy." Packt Hub, 1 Dec. 2017, hub.packtpub.com/iterative-machine-learning-step-towards-model-accuracy/.
[8] "Machine Learning Algorithm - an Overview | ScienceDirect Topics." Www.sciencedirect.com, www.sciencedirect.com/topics/engineering/machine-learning-algorithm.

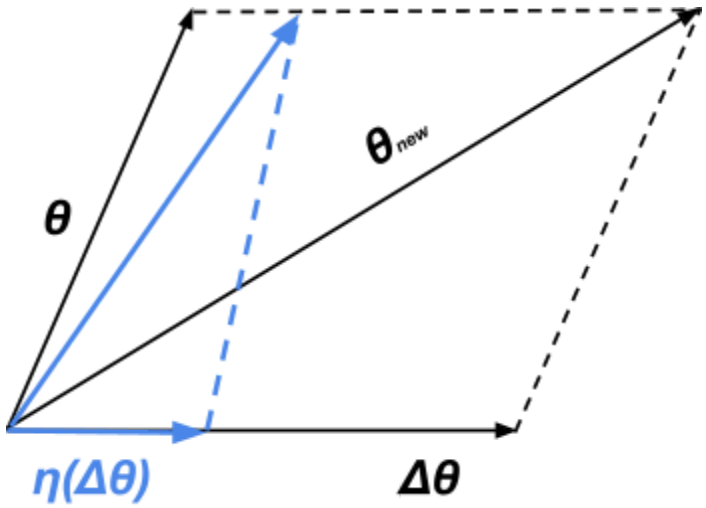
[9] Donges, Niklas. "Gradient Descent: An Introduction to One of Machine Learning's Most Popular Algorithms." Built In, 23 July 2021, builtin.com/data-science/gradient-descent.

values (prediction error) at a quantifiable position. Thus, it is a parameter that determines how well a ML model is performing. The Gradient Descent follows these steps to minimize the loss function[10]:

1. Choose initial parameter (θ = [w,b])
2. Calculate gradient at the chosen starting point
3. Make incremental step downwards (along negative gradient)

- The step size taken is known as the learning rate (η)

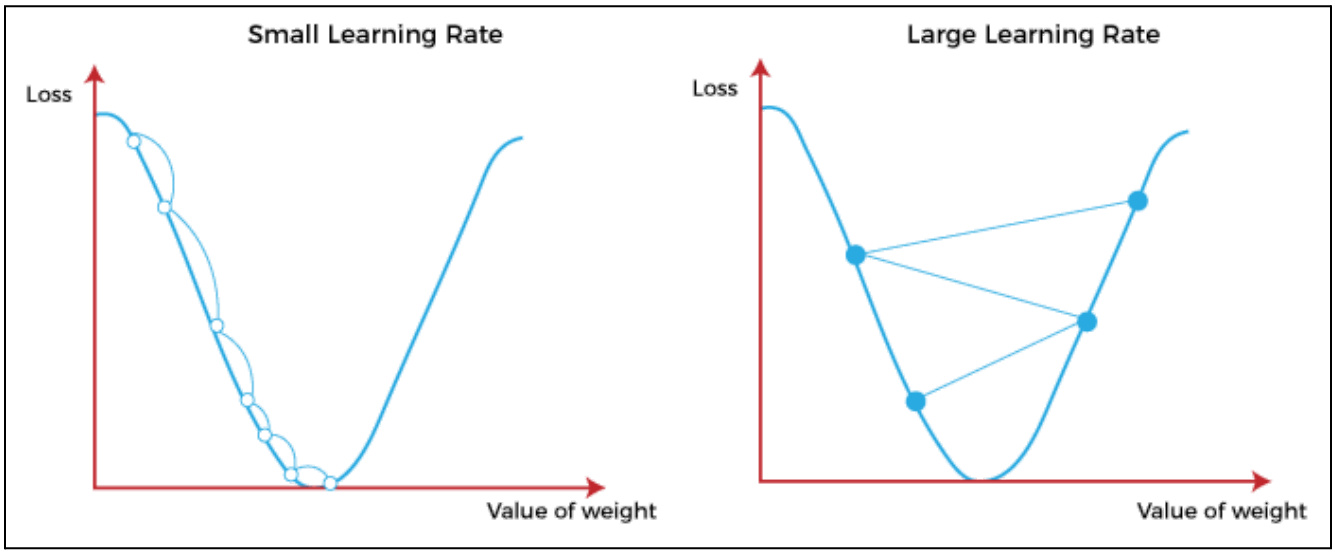
- $\theta_{new} = \theta + \eta(\Delta\theta)$



4. Iterate until arrival at the function's minima

[10] "Gradient Descent in Machine Learning - Javatpoint." Www.javatpoint.com, www.javatpoint.com/gradient-descent-in-machine-learning.

About Learning Rate[11]



[12]

Nevertheless, the question still remains of how the Taylor Series plays a role in the Gradient Descent algorithm to assist in the minimization of error by machines.

2.2. Application of Taylor Series in Gradient Descent

In the Gradient Descent Optimization Algorithm, the Taylor Series is used to approximate a function as a distance Δx from a point x, by differentiating the function at x. The Taylor Series can hence be rewritten as shown below.

$$f(x+\Delta x) = f(x) + f'(x)\Delta x + \frac{1}{2!}f''(x)\Delta x^2 + \frac{1}{3!}f'''(x)\Delta x^3 + \dots \qquad (7)$$

[11] "What Is Learning Rate in Machine Learning." Deepchecks, deepchecks.com/glossary/learning-rate-in-machine-learning/.
[12] "Gradient Descent in Machine Learning - Javatpoint." Www.javatpoint.com, www.javatpoint.com/gradient-descent-in-machine-learning.

The function of interest is the loss function. Therefore, the above Taylor Series must be written in terms of W.

Taylor Series for loss function

$$L(w + \Delta w) = L(w) + L'(w)\Delta w + \frac{1}{2!}L''(w)\Delta w^2 + \frac{1}{3!}L'''(w)\Delta w^3 + \ldots \quad (8)$$

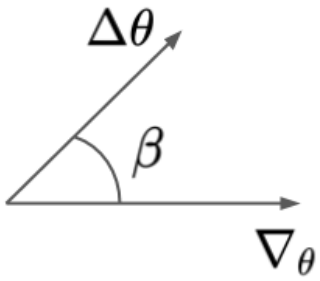
Updated loss function

$$(\theta + \eta\Delta\theta) \approx L(\theta) + \eta\Delta\theta^T \nabla_\theta L(\theta) \quad (9)$$

Since our aim is to minimize loss, we need the updated loss ($L(\theta + \eta\Delta\theta)$) to be less than the initial loss ($L(\theta)$). In other words, the below value must be negative.

$$\eta\Delta\theta^T \nabla_\theta L(\theta) < 0.$$

Now, let β be the angle between $\Delta\theta$ and ∇_θ.



Considering the cos of an angle between two vectors is product ÷ product of magnitudes:

$$cos\beta = \frac{\Delta\theta^T \nabla_\theta}{|\Delta\theta||\nabla_\theta|}$$

Since cos lies between -1 and +1:

$$-1 \leq cos\beta = \frac{\Delta\theta^T \nabla_\theta}{|\Delta\theta|| \nabla_\theta|} \leq +1$$

Multiplication by denominator ($k = |\Delta\theta|| \nabla_\theta|$) throughout:

$$-k \leq \Delta\theta^T \nabla_\theta \leq +k$$

The lowest value of $\Delta\theta^T \nabla_\theta$ is -k.

Thus, $cos\beta = -1$ and $\beta = 180°$.

Since the vectors have opposite directions, we can conclude that:

$$\Delta\theta = -\nabla_\theta L(\theta)$$

Substitution of this into a general Learning Algorithm (set of instructions given to machine) updates the involved parameters. Like the Taylor Series, each iteration produces a more desirable output. In this case, reiteration minimizes the loss function, thus reducing prediction error. This is repeated until loss equals zero and the global minima is located. **It can therefore be stated that the more the iterations, the lesser the prediction error made by the machine, increasing its efficiency.**

3. Evaluation and Conclusion

3.1. Evaluation

Strengths:

1. The investigation entailed an exhaustive explanation of the Taylor Series, Machine Learning and their relevance in mathematics, as well as the current world.
2. Numerous methods of approximation were explained.
3. All calculations made in the investigation were mathematically reasoned.
4. Graphical representations were used to supplement mathematical explanations.
5. Implementation of multiple mathematical concepts (functions, calculus, vectors, trigonometry, sequences and series),

Weaknesses:

1. There are various other applications of the Taylor Series in ML (i.e. nonlinear parametric regression) that this paper does not discuss.
2. The calculations were only done mathematically. More complex computations would require the use of a program, which this paper lacks.
3. Sample calculations of Gradient Descent would involve programs. They were not included to maintain the length of the paper.

3.2. Conclusion

This investigation explored the use of Taylor Series in machine learning; specifically in the Gradient Descent Optimization Algorithm. The mathematical concepts of functions and calculus were greatly useful in our discussion of the Taylor Series. Upon the introduction of Gradient

Descent, vectors and trigonometry also played a vital role in the empirical analysis. In company with our graphical means of explanation, the implementation of these concepts deem this investigation reliable.

We can also conclude that mathematical reiterations yield values of increasing precision. In terms of the Taylor Series, a polynomial approximation of the highest degree possible would produce the most accurate value of $F(x)$.

In terms of its application in ML, the more times the process of Gradient Descent is repeated, the better the minimisation of the loss function. It is intriguing to see that the utilization of mathematics in such fields of inquiry produce marvels such as self-driving cars and speech recognition.

All in all, it can be concluded that this investigation was successful in its aim to mathematically display and determine how the Taylor series is used in machine learning and the Gradient Descent Optimization Algorithm.

4. Bibliography

1. Khan, Shahbaz. “Mathematical Intuition behind Gradient Descent.” Medium, 6 May 2021, towardsdatascience.com/mathematical-intuition-behind-gradient-descent-f1b959a59e6d#b45d. Accessed 19 May 2022.

2. “Second Approximation — the Quadratic Approximation.” Personal.math.ubc.ca, personal.math.ubc.ca/~CLP/CLP1/clp_1_dc/ssec_second_approx.html. Accessed 3 May 2022.

3. Weisstein, Eric W. “Taylor Series.” Mathworld.wolfram.com, mathworld.wolfram.com/TaylorSeries.html. Accessed 4 May 2022.

4. Hassoun, Mohamad. ECE 3040 Lecture 8: Taylor Series Approximations I. Accessed 1 May 2022.

5. “Multivariable Calculus - Second-Order Taylor Series Terms in Gradient Descent.” Mathematics Stack Exchange, math.stackexchange.com/questions/2957673/second-order-taylor-series-terms-in-gradient-descent. Accessed 19 May 2022.

6. “Taylor Series - Math Images.” Mathimages.swarthmore.edu, mathimages.swarthmore.edu/index.php/Taylor_Series. Accessed 4 May 2022.

7. x-engineer.org. Taylor Series Approximation – X-Engineer.org. x-engineer.org/taylor-series-approximation/. Accessed 2 May 2022.

8. Kwiatkowski, Robert. “Gradient Descent Algorithm — a Deep Dive.” Medium, 24 May 2021, towardsdatascience.com/gradient-descent-algorithm-a-deep-dive-cf04e8115f21#:~:text=Gradient%20descent%20(GD)%20is%20an. Accessed 5 May, 2022.

9. Saeed, Mehreen. “A Gentle Introduction to Taylor Series.” Machine Learning Mastery, 19 Aug. 2021, machinelearningmastery.com/a-gentle-introduction-to-taylor-series/. Accessed 5 May 2022.