Gradient Descent Algorithm (optimizasyon Algoritması)

Gradient Descent'e başlamadan önce;

Objective function could define anything with respect to the problem you are optimizing. It could be cost for a company, losses for another or even revenue etc.

(objective function genelde "Cost Function" dır)

This function (objective) is optimal at a specific point X^*. This X^* is the optimal point. finding X^* for which $f(x)$ is minimum/maximum. This can also be written as argmin$(f(x))$ = argument where function $f(x)$ is minimum (or argmax$(f(x))$.

Closed form Solution, Non closed form solution?

$S = \sum_{i=0}^{\infty} \frac{x}{2^i}$ } non-closed (There is an infinite number of them. Dünyanın en iyi PC'si de gelse final sonucu veremez)

Closed form ise ⇒ Solves a given problem in terms of functions and mathematical operations from a given generally-accepted set.

Örneğin

Quadratic Polynomial $ax^2 + bx + c = 0$ is closed form because its solution $x = \frac{-b \mp \sqrt{b^2-4ac}}{2a}$ can be expressed in finite numbers of elementary operations.

Neden anlattık Bunları?

S/ OLS yöntemi special case of an optimization problem. The objective function is the sum of squared distances. The solution can be found analytically. Ordinary Least Squares Solution is the analytical solution, as we find the least value of squares of the error. Model parametrelerini analytically bulmak işine biz Closed-form Solution diyoruz, ve OLS solution analitik bir solution'dur. Fakat bu yöntemi non linear ve large datasetlere uygulamak feasible değildir. Bu yüzden, we find numerical approximation of this solution by iterative method (Gradient Descent). Yani; Liner Regresyon parametreleri hem OLS hem de Gradient Descent ile tahmin edilebilir. Gradient Descent, optimizasyon algoritması olup, ana hedefi cost function'ı minimize etmektir.

Closed form solutions (OLS) are solved analytically. But it is not optimization method.

MLE: We can not write down the MLE in closed form unlike linear Regression (OLS solution). So we need to use an optimization algorithm to compute it.

MLE specifies the objective function (the likelihood function); Gradient Descent finds the optimal solution to a problem once the objective function is specified. We can use GD (or other optimization algori to solve Maximum Likelihood problem, and the result will be maximum likelihood estimator.

Şimdi Gradient Descent'e Başlayalım;

Data Science field'da bir çok şeyi optimize ederi

* Liner Regresyonda Intercept ve Slope'ı
* Logistic Reg'de parametreleri
* t-SNE'de cluster'ları

Sadece birkaç örnek

"Gradient Descent" can optimize all these things and much more.

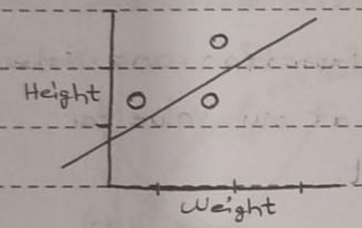
Types of Gradient Descent:

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

Örnek olarak; Eğer ki Regresion Line'ın nasıl optimize edildiğini öğrenirsek, diğer optimizasyon problemlerinin de nasıl çözüldüğünü öğrenmiş olacağız.

Konuyu Liner Regression Üzerinden Anlayalım

Predicted Height = Intercept + slope × Weight



Konuyu daha iyi anlamak için şu anlık "slope"ı sabit alalım ve "Intercept"in optimal değerini bulalım. Slope = 0.64 olsun.

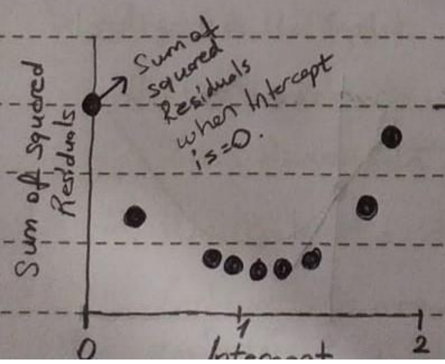
Predicted Height = Intercept + 0.64 × Weight

1) İlk olarak random bir "Intercept" seçilir. Bu sadece başlangıç random değeri ve GD bunu optimize edecek. Random olarak Intercept'i 0 alalım (herhangi bir değer alabilir)

2) Gradient Descent is an optimization algorithm used for minimizing the cost Function in various ML algorithms.

Liner Reg için Loss Function => Sum of Squared Residuals.

Bu durumda, 3 data noktası için "Predicted Height"ler bulunur ve actual'dan ne kadar saptıkları hesaplanır.



→ Gradient Descent, optimal noktaya yakın yerlerde daha fazla iterasyon yapar.

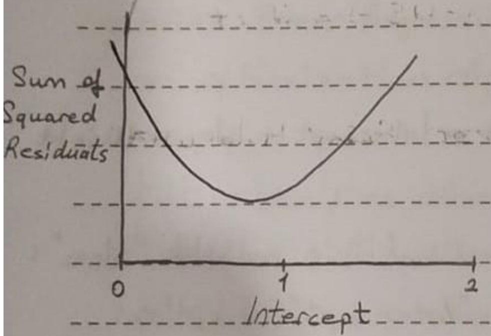
1. data noktası

Sum of Squared Residuals = (observed − Predicted)^2 +

2. nokta (observed − Predicted)^2 +

(observed − Predicted)^2

3. gözlem +



→ Take the Derivative of the loss function and determine the slope at any value for Intercept.

(Kısmi Türev)

Loss fonksiyonun "Intercept"e göre türevi alınır ve Intercept değerleri için curve'ün eğimi bulunur.

Örneğin

Random Intercept = 0 almıştık, başlangıç değerini.

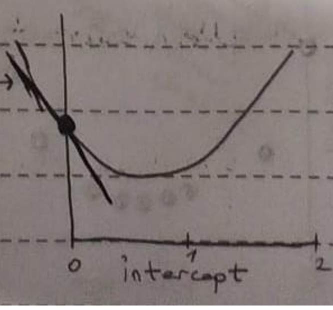
Bu durumda

$$\frac{d}{d\,intercept} \text{Sum of squared Residuals} = \frac{d}{d\,intercept}\left(1.4 - (intercept + 0.64 \times 0.5)\right)^2 +$$

$$\frac{d}{d\,intercept}\left(1.9 - (intercept + 0.64 \times 2.3)\right)^2 +$$

$$\frac{d}{d\,intercept}\left(3.2 - (intercept + 0.64 \times 2.9)\right)^2$$

Intercept = 0 iken eğim −5.7



Step Size = Eğim x Learning Rate

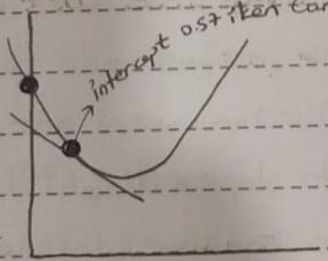
↳ Yeni Initial değer = 0 iken, bir sonraki değer ne olacak.

İlk değer 0 idi → Step Size = $-5.7 \times \overset{\text{olsun}}{0.1} = -0.57$

New Intercept değeriniz ⇒ $0 - (-0.57) = 0.57$

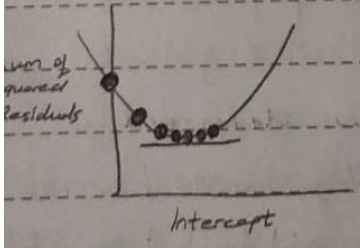
Şimdi ise 0.57'yi Intercept alarak Eğimi bulacağız.

Intercept 0.57 iken Curve eğimi = -2.3



→ Yeni Intercept seçilir ⇒ Step Size = $-2.3 \times 0.1 = -0.23$

Yeni Intercept = $0.57 - (-0.23) = 0.8$



→ Gradient Descent stops when the Step Size is so close to 0.

Step Size = Slope (Eğim) x Learning Rate

↳ Slope 0'a yakın oldukça Step Size'da yakınsa

Şimdiye Dek Ne Yaptık:

1) We decide to use Sum of Squared Residuals as "Loss Function".

2) We took the derivative of the Sum of Squared Residuals. In other words, we took derivative of the "Loss Function".

../..../....

3) We picked the Random value for Intercept, in this case we set the Intercept = 0.

4) We calculate the derivative when Intercept = 0

5) Plugged that Slope (eğim) into Step Size calculation

Step Size = Slope x Learning Rate

6) We calculated the "New Intercept" = old intercept - Step Size

7) We plugged the New Intercept values into derivative and repeated everything until Step Size was close to 0.

Buraya Kadar, Predicted Height = Intercept + slope x Weight

Optimum değerini bulduk

Peki hem Intercept hem de Weight değişkeninin katsayısını bulmak isteseydik ne yapacaktık??

Tamamen aynı ⇒ loss function'ının different Intercept ve Slope değerlerine karşılık 3-D grafiğindeki best nokta bulunacaktı

Yani 2 yerine 3 eksenli grafiğimiz olacaktı ve parametrelerin farklı değerlerine karşılık Minimum Sum of Squared Residuals bulunacak.

Bu kez yine; loss fonksiyonun Intercept'e ve Slope'a göre ayrı ayrı türevleri alınır. (Partial Derivatives)

Yani;

$$\frac{d}{d\,\text{intercept}} \text{Sum of squared Residuals} = \ldots$$

$$\frac{d}{d\,\text{slope}} \text{Sum of squared Residuals} = \ldots$$

Just like before, we start by picking a random number for the Intercept = 0 alalım

and

we pick random number for the Slope = 1 alalım.

↳ Bu durumda 2 eğim elde ederiz, biri Intercept - loss function; diğeri slope - loss function. Sonra bu eğimleri Step Size formülüne koyarız ayrı ayrı.

↳ Bu adımları tüm Step Size'ları çok küçük olana dek (veya maksimum number of steps'de olabilir) tekrarlarız. Ve en sonunda Best Fit line bulunur.

Regardless of which Loss Function you use, Gradient Descent works the same way.

2 veya daha fazla optimize edilecek parametre var ise ayrı adımlar izlenir.

2 veya daha fazla parametre durumunda; her parametre için ayrı ayrı loss function Türevi alınır. Her adımda parametreler için yeni değerler elde edilir. Taki tüm parametreler için Step Sizelar çok küçük olana dek.
Yani Parametrelerin ayrı ayrı eğimlerinin 0'a yaklaşmasını isteriz

→ Step Size'ın çok küçük olması demek, cost func- çok küçük veya 0 eşit olması demek.

Bu örneklerde

Batch Gradient Descent kullanıldı (Yani tüm gözlem noktaları kullanılarak loss func minimize edildi.

Ancak;

Data size çok fazla ise "Stochastic GD" her adımda 1 gözlem alarak bu işlemleri yapar.

"Mini Batch GD" ise her iterasyonda belli sayıda gözlem alınarak bu işlemler gerçekleşir.

Gradient Descent aims to minimize objective func.

Gradient Ascent aims to maximize objective func

Yani; We use gradient ascent to maximize a likelihood function, and Gradient Descent to minimize Cost function.

Note

2 or more parameters

When we have 2 or more derivatives of the same function, they are called "Gradient". We use this "Gradient" to descend to lowest point in the Loss function. This is why this algorithm called "Gradient Descent".