Gradient Descent Derivation

04 Mar 2014

Andrew Ng's course on Machine Learning at Coursera provides an excellent explanation of gradient descent for linear regression. To really get a strong grasp on it, I decided to work through some of the derivations and some simple examples here.

This material assumes some familiarity with linear regression, and is primarily intended to provide additional insight into the gradient descent technique, not linear regression in general.

I am making use of the same notation as the Coursera course, so it will be most helpful for students of that course.

For linear regression, we have a linear hypothesis function, $h(x) = \theta_0 + \theta_1 x$. We want to find the values of θ_0 and θ_1 which provide the best fit of our hypothesis to a training set. The training set examples are labeled x, y, where x is the input value and y is the output. The ith training example is labeled as $x^{(i)}, y^{(i)}$. Do not confuse this as an exponent! It just means that this is the ith training example.

MSE Cost Function

The cost function J for a particular choice of parameters θ is the mean squared error (MSE):

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Where the variables used are:

m	The number of training examples
$x^{(i)}$	The input vector for the i^{th} training example
$y^{(i)}$	The class label for the i^{th} training example
Θ	The chosen parameter values or "weights" (Θ_0, Θ_1, Θ_2)
$h_\theta\left(x^{(i)}\right)$	The algorithm's prediction for the i^{th} training example using the parameters Θ.

The MSE measures the average amount that the model's predictions vary from the correct values, so you can think of it as a measure of the model's performance on the training set. The cost is higher when the model is performing poorly on the training set. The objective of the learning algorithm, then, is to find the parameters θ which give the minimum possible cost J.

This minimization objective is expressed using the following notation, which simply states that we want to find the θ which minimizes the cost $J(\theta)$.

$$\min_\theta J(\theta)$$

Gradient Descent Minimization - Single Variable Example

We're going to be using gradient descent to find θ that minimizes the cost. But let's forget the MSE cost function for a moment and look at gradient descent as a minimization technique in general.

Let's take the much simpler function $J(\theta) = \theta^2$, and let's say we want to find

the value of θ which minimizes $J(\theta)$.

Gradient descent starts with a random value of θ, typically $\theta = 0$, but since $\theta = 0$ is already the minimum of our function θ^2, let's start with $\theta = 3$.

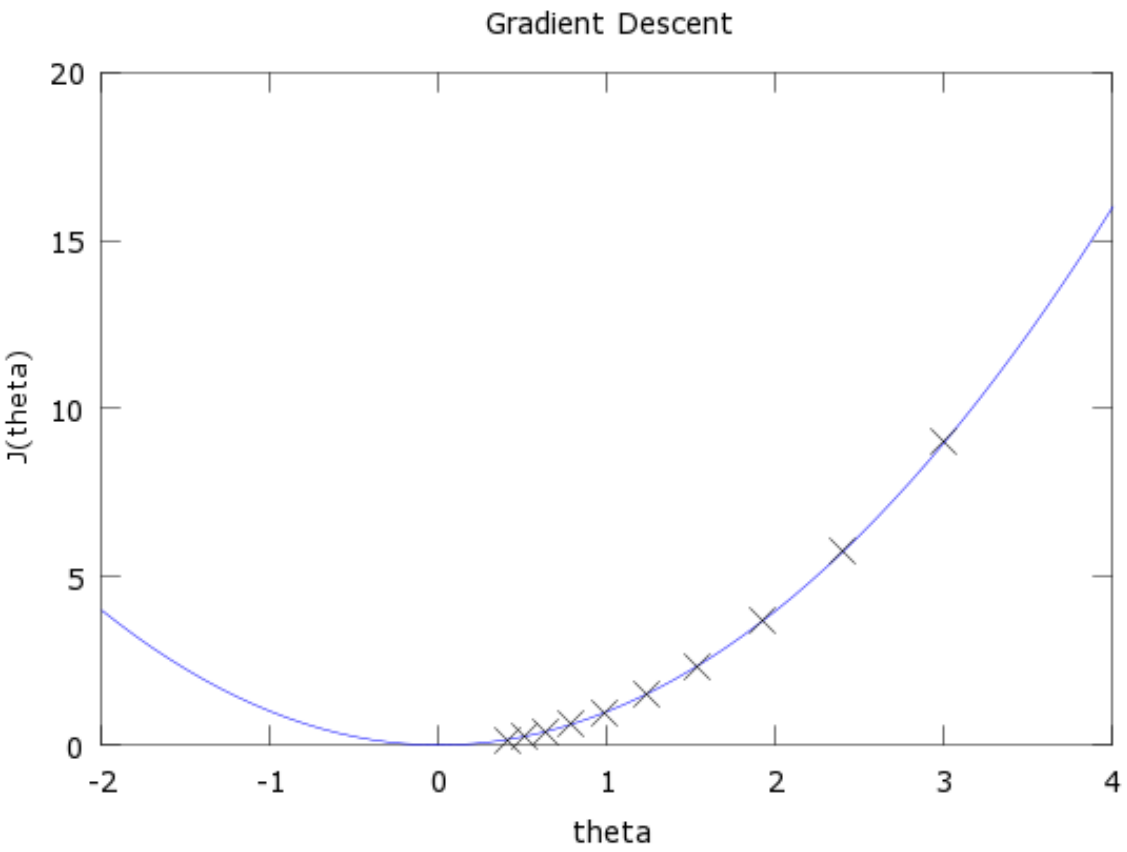
Gradient descent is an iterative algorithm which we will run many times. On each iteration, we apply the following "update rule" (the := symbol means replace theta with the value computed on the right):

$$\theta := \theta - \alpha \frac{d}{d\theta} J(\theta)$$

Alpha is a parameter called the learning rate which we'll come back to, but for now we're going to set it to 0.1. The derivative of $J(\theta)$ is simply 2θ.

$$\alpha = 0.1, \qquad \frac{d}{d\theta} J(\theta) = 2\theta$$

Below is a plot of our function, $J(\theta)$, and the value of θ over ten iterations of gradient descent.



Below is a table showing the value of theta prior to each iteration, and the update amounts.

Itera-tion	θ	$\alpha \frac{d}{d\theta} J(\theta)$
1	3	0.6
2	2.4	0.48
3	1.92	0.384
4	1.536	0.307
5	1.229	0.246
6	0.983	0.197
7	0.786	0.157
8	0.629	0.126
9	0.503	0.101
10	0.403	0.081

Cost Function Derivative

Why does gradient descent use the derivative of the cost function? Finding the slope of the cost function at our current θ value tells us two things.

The first is the direction to move theta in. When you look at the plot of a function, a positive slope means the function goes upward as you move right, so we want to move left in order to find the minimum. Similarly, a negative slope means the function goes downard towards the right, so we want to move right to find the minimum.

The second is how big of a step to take. If the slope is large we want to take a large step because we're far from the minimum. If the slope is small we want to take a smaller step. Note in the example above how gradient descent takes

increasingly smaller steps towards the minimum with each iteration.

The Learning Rate - Alpha

The learning rate gives the engineer some additional control over how large of steps we make.

Selecting the right learning rate is critical. If the learning rate is too large, you can overstep the minimum and even diverge. For example, think through what would happen in the above example if we used a learning rate of 2. Each iteration would take us farther away from the minimum!

The only concern with using too small of a learning rate is that you will need to run more iterations of gradient descent, increasing your training time.

Convergence / Stopping Gradient Descent

Note in the above example that gradient descent will never actually converge on the minimum, $\theta = 0$. Methods for deciding when to stop gradient descent are beyond my level of expertise, but I can tell you that when gradient descent is used in the assignments in the Coursera course, gradient descent is run for a large, fixed number of iterations (for example, 100 iterations), with no test for convergence.

Gradient Descent - Multiple Variables Example

The MSE cost function includes multiple variables, so let's look at one more simple minimization example before going back to the cost function.

Let's take the function:

$$J(\theta) = {\theta_1}^2 + {\theta_2}^2$$

When there are multiple variables in the minimization objective, gradient

descent defines a separate update rule for each variable. The update rule for θ_1 uses the partial derivative of J with respect to θ_1. A partial derivative just means that we hold all of the other variables constant–to take the partial derivative with respect to θ_1, we just treat θ_2 as a constant. The update rules are in the table below, as well as the math for calculating the partial derivatives. Make sure you work through those; I wrote out the derivation to make it easy to follow.

Function:

$$J(\theta_1, \theta_2) = {\theta_1}^2 + {\theta_2}^2$$

Objective:

$$\min_{\theta_1, \theta_2} J(\theta_1, \theta_2)$$

Update rules:

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1, \theta_2)$$
$$\theta_2 := \theta_2 - \alpha \frac{\partial}{\partial \theta_2} J(\theta_1, \theta_2)$$

Derivatives:

$$\frac{\partial}{\partial \theta_1} J(\theta_1, \theta_2) = \frac{\partial}{\partial \theta_1} {\theta_1}^2 + \frac{\partial}{\partial \theta_1} {\theta_2}^2 = 2\theta_1$$

$$\frac{\partial}{\partial \theta_2} J(\theta_1, \theta_2) = \frac{\partial}{\partial \theta_2} {\theta_1}^2 + \frac{\partial}{\partial \theta_2} {\theta_2}^2 = 2\theta_2$$

Note that when implementing the update rule in software, θ_1 and θ_2 should not be updated until *after* you have computed the new values for both of them. Specifically, you don't want to use the new value of θ_1 to calculate the new value of θ_2.

Gradient Descent of MSE

Now that we know how to perform gradient descent on an equation with multiple variables, we can return to looking at gradient descent on our MSE cost function.

The MSE cost function is labeled as equation [1.0] below. Taking the derivative of this equation is a little more tricky. The key thing to remember is that x and y are *not* variables for the sake of the derivative. Rather, they represent a large set of constants (your training set). So when taking the derivative of the cost function, we'll treat x and y like we would any other constant.

Once again, our hypothesis function for linear regression is the following:

$$h(x) = \theta_0 + \theta_1 x$$

I've written out the derivation below, and I explain each step in detail further down.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 \quad [1.0]$$

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1) = \frac{\partial}{\partial\theta_0}\left(\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right) \quad [1.1]$$

$$\frac{d}{d\theta_1}J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial\theta_0}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 \quad [1.2]$$

$$\frac{d}{d\theta_1}J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}2\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)\frac{\partial}{\partial\theta_0}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \quad [1.3]$$

$$\frac{d}{d\theta_1}J(\theta_0,\theta_1) = \frac{2}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \quad [1.4]$$

To move from equation [1.1] to [1.2], we need to apply two basic derivative rules:

Scalar multiple rule: $\frac{d}{dx}(\alpha u) = \alpha\frac{du}{dx}$

Sum rule: $\frac{d}{dx}\sum u = \sum\frac{du}{dx}$

Moving from [1.2] to [1.3], we apply both the power rule and the chain rule:

Power rule: $$\frac{d}{dx}u^n = nu^{n-1}\frac{du}{dx}$$

Chain rule: $$\frac{d}{dx}f(g(x)) = f'\big(g(x)\big)g'(x)$$

Finally, to go from [1.3] to [1.4], we must evaluate the partial derivative as follows. Recall again that when taking this partial derivative all letters except θ_0 are treated as constants (θ_1 , x, and y).

$$\frac{\partial}{\partial\theta_0}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) = \frac{\partial}{\partial\theta_0}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right) = 1$$

Equation [1.4] gives us the partial derivative of the MSE cost function with respect to one of the variables, θ_0 . Now we must also take the partial derivative of the MSE function with respect to θ_1 . The only difference is in the final step, where we take the partial derivative of the error:

$$\frac{\partial}{\partial\theta_1}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) = \frac{\partial}{\partial\theta_1}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right) = x$$

One Half Mean Squared Error

In Andrew Ng's Machine Learning course, there is one small modification to this derivation. We multiply our MSE cost function by 1/2 so that when we take the derivative, the 2s cancel out. Multiplying the cost function by a scalar does not affect the location of its minimum, so we can get away with this.

Alternatively, you could think of this as folding the 2 into the learning rate. It makes sense to leave the 1/m term, though, because we want the same learning rate (alpha) to work for different training set sizes (m).

Final Update Rules

Altogether, we have the following definition for gradient descent over our cost function.

Cost Function – "One Half Mean Squared Error":

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

Objective:

$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Update rules:

$$\theta_0 := \theta_0 - \alpha\frac{\partial}{\partial\theta_0}J(\theta_0, \theta_1)$$

$$\theta_1 := \theta_1 - \alpha\frac{\partial}{\partial\theta_1}J(\theta_0, \theta_1)$$

Derivatives:

$$\frac{\partial}{\partial\theta_0}J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)\cdot x^{(i)}$$

Training Set Statistics

Note that each update of the theta variables is averaged over the training set. Every training example suggests its own modification to the theta values, and then we take the average of these suggestions to make our actual update.

This means that the statistics of your training set are being taken into account during the learning process. An outlier training example (or even a mislabeled / corrupted example) is going to have less influence over the final weights because it is one voice versus many.

Chris McCormick Mod ↗ Akshay • a year ago

Hi Akshay,
The difference is that, in our situation, u is a function of x.
So the derivative of u^n is nu^(n-1) *multiplied by the derivative of u with respect to x.*

1 ^ | v • Reply • Share ›

NAZRUL HASSAN • a year ago

U were heaven sent.Seriously

4 ^ | v • Reply • Share ›

Chris McCormick Mod ↗ NAZRUL HASSAN • 10 months ago

Haha, awesome!

^ | v • Reply • Share ›

Vaibhav Jaiswal • 2 years ago

Good explanation but please change the symbol of derivative (d) to partial derivative(somewhat J). Because it changes the whole meaning of the equation and can be confusing to anyone who knows the difference between both.Please change it.

4 ^ | v • Reply • Share ›

Chris McCormick Mod ↗ Vaibhav Jaiswal • a year ago • edited

Thanks Vaibhav, I wasn't familiar with that distinction.

For anyone interested, from here:

∂ is used when a function; say f(x,y,z) depends on more than 1 variable.

d is used when a function; say f(t) depends on only 1 variable.

^ | v • Reply • Share ›

AshtrayK • 3 years ago

Perfect step by step presentation!
Thank you!

4 ^ | v • Reply • Share ›

Chris McCormick Mod ↗ AshtrayK • 3 years ago

Thank you! Glad it was helpful :)

2 ^ | v • Reply • Share ›

Giuseppe Di Pellegrino • a year ago

Great explanation, indeed! Thank you Chris!

3 ^ | v • Reply • Share ›

Chris McCormick Mod ↗ Giuseppe Di Pellegrino • 10 months ago

Thanks, Giuseppe! Glad it was helpful :)

^ | v • Reply • Share ›

Hyper Safi • a year ago

Hi, Awesome work. I just joined the course on machine learning. I am lacking my math skills, any recommendations what I should start! I do get programming though, but struggling with math.
Appreciate your help...

2 ^ | v • Reply • Share ›

Weronika • a year ago

Thanks for that article, Chris! One thing I don't understand: Why the derivative of J(Θ) is 2Θ?n Where did that come from?
I would be really grateful for an explanation!

2 ^ | v • Reply • Share ›

Weronika ↗ Weronika • a year ago

Thank you so much **@Trigarius** and @RedNam for your help!

^ | v • Reply • Share ›

Trigarius ↗ Weronika • a year ago

Hi Weronika,

This is differentiation. J(theta) is a function where theta is equal to theta^2. In order to differentiate this we need only multiply the coefficient of x by the power which is two and then take away one from the power eg. n*1x^n-1 or 2*1x^2-1 which comes to 2theta.

From this, we are able to calculate the minimum of the curve theta^2

^ | v • Reply • Share ›

RedNam ↗ Weronika • a year ago

You can take a look at Wikipedia here

Reply • Share ›

Mah. E. • 2 years ago

Thanks Chris, i searched a lot to get quick explanation of multivariate derivative of cost function and found this piece of art
Thanks a lot man, keep up the great work

2 Reply • Share ›

Chris McCormick Mod ↗ Mah. E. • a year ago

High praise! Thank you :)

Reply • Share ›

Hyper Safi • a year ago

Hi Awesome work,
I am super passionate about Machine Learning and joined Andrew's course, but I am lacking math skills. What would you recommend to start with?. I get programming very easy, but do not have the fundamentals of math skills . Should I start with linear Algebra or? Thx a lot

1 Reply • Share ›

Chris McCormick Mod ↗ Hyper Safi • 10 months ago

Hi Hyper,
If I remember right, Andrew's course has a nice PDF that's a crash course in linear algebra. Linear algebra is a big topic and you need very little of it--you just need to understand what a vector and a matrix are (and the notation used to describe them), and how to multiply them with each other. It's not as bad as it may sound--don't let it stop you! :)
Thanks,
Chris

Reply • Share ›

keerthana Manjunatha • a year ago

Thanks for the article. Perfect explanation.

1 Reply • Share ›

Chris McCormick Mod ↗ keerthana Manjunatha • 10 months ago

Thanks, Keerthana!

Reply • Share ›

lucas van der zee • 2 years ago

Beautiful! This helps me a lot with brushing up my calculus while doing the Machine Learning course

1 ^ | v • Reply • Share ›

disqus_5kgkbbaz50 • 2 years ago

Big Thanks! Been trying to understand this for some time and no one has given such a lucid explanation. Sir many of us owe you a big Thanks

1 ^ | v • Reply • Share ›

Rohan Damodar • 2 years ago

This helped a lot!! Thank You :)

1 ^ | v • Reply • Share ›

Musharaf Baig • 3 years ago • edited

such a helpful article ! thanks Chris.

1 ^ | v • Reply • Share ›

Yu Zhang • 3 years ago

very helpful !!!!! Thank you vey much!!!

1 ^ | v • Reply • Share ›

Mohammad Husain • 3 years ago

Awesome! Thanks for the step by step explanation!

1 ^ | v • Reply • Share ›

Ashish Jat • 3 years ago

Thank yiu

1 ^ | v • Reply • Share ›

Hannah • 2 months ago

This is just the best explanation of gradient descent and cost function I have ever seen by far.

^ | v • Reply • Share ›

Azizul Haq • 3 months ago • edited

Thanks. I can not understand moving from 1.2 to 1.3 where do you use the chain rule? It seems only using power rule. Would you please explain?

^ | v • Reply • Share ›

Neeloor Palai • 4 months ago

Thanks a lot, I never undstood Gradient Descent even after reading so many article on about it. This is perfect and excellent, once again thanks

about it. This is perfect and excellent, once again thanks

Reply • Share ›

mamun • 5 months ago

nice explanation https://engineering-and-sci...

Reply • Share ›

Muhammed Eldeeb • 6 months ago

thank you so much for your simple explanation

Reply • Share ›

Dinesh Muniandy • 10 months ago

Hi Chris,

Thanks for writing out this really helpful article.

I have a few questions (hopefully you could help answer):

(1) On the equation 1.4:
"difference is in the final step, where we take the partial derivative of the error"

I am not able to see this in the equation, could you please
let me know if I'm missing any information ?

(2) Throughout the equation, m = 2, am I correct to assume that,
you've randomly picked a "2" size training set for the equation ?

(3) "Multiplying the MSE cost function by 1/2". How does the 2s cancel out
when it is multiplied with the derivative ? Has this got to do with the
number of training size ?

Many thanks!

Reply • Share ›

ben ➦ Dinesh Muniandy • 10 months ago • edited

Dinesh, I'll try to answer this for you.

1. Remember that h(x)=h_θ(x)=θ_0+θ_1(x) - which is the linear formula for this regression. It's a party of changing symbols so it can throw you off if you miss any of these equivalencies in the article.

So in the last equation (1.4) the term h_θ(x)^(i) =h_θ(x) (since we compute one x at a time). So for each x: h_θ(x) = θ_0+θ_1(x), which gives us the new look 1.4 equation: 2/m∑(θ_0+θ_1(x)-y). Now we take partial derivative for θ_0 and θ_1.

2. m = total number of rows in the dataset. I'm not sure why you are equating it with the training dataset size. If you are referring to the 2/m, that 2 is a result of

executing the derivative in equation 1.2 - the power 2 comes down to the front of the term, and since it is not affected by the sigma's summation we can pull it out in front of the sigma in 1.4 .

3. Now that we know that the 2 is a result of the derivative, multiplying by 1/2 just gives us a simplified function (like Ng video says) and the minimum of our function is not affected (try graphing (1/2)x^2 and x^2, they have the same minimum, even though their change rates are different).

Reply • Share ›

Trigarius • a year ago

This is Brilliant. Thank you so much.

Reply • Share ›

Chris McCormick Mod Trigarius • a year ago

Thanks, glad it helped!

Reply • Share ›

Fahim Ali Zain • a year ago

I missed the SUM rule! :p

Reply • Share ›

Jeetendra Dhall • 2 years ago • edited

I liked the last paragraph very much, of course, in addition to the complete article. Thank you so much!

Reply • Share ›

giza • 2 years ago

This is brilliant! I understood the intuition and the formulas but the derivation was a little bit elusive. Thank you for this!

Reply • Share ›

Thava Alagu • 2 years ago

Clean step by step explanation. very helpful reference. Thanks for writing this up!

Reply • Share ›

Wira Maharddhika • 2 years ago

Thank you very much, that "power rule" save my life

Reply • Share ›

DataBeta • 2 years ago • edited

Thanks for the explanations! I don't understand one part though: how you went from 1.3 to 1.4 to get 1 and x(i) for the partial derivative of theta0 and theta1, respectively. I read what

you wrote but I don't get how you got dtheta0 (dtheta0+theta1X(i)-y(i). I do understand how that turns into 1 but I just don't get how you got to the equation above from the 1.4

^ | v • Reply • Share ›

Mathias Sunardi DataBeta • 2 years ago • edited

I was stumped by that too. My guess is that h_theta(x(i)) = theta0 + theta1*x(i), which was not explicitly stated before eq [1.0] (probably because it's assumed from the Coursera course).

Thus, d(h_theta(x(i)) - y(i))/dtheta0 = 1, and d(h_theta(x(i)) - y(i))/dtheta1 = x(i).
Since the derivatives have the form: dJ(theta0, theta1)/dtheta# = (1/m)*SUM(...)*d(h_theta(x(i)) - y(i))/dtheta# (substitute # for 0 and 1 for each partial derivative), we get:
dJ(theta0, theta1)/dtheta0 = (1/m)*SUM(...)*d(h_theta(x(i)) - y(i))/dtheta0 = (1/m)*SUM(...)
dJ(theta0, theta1)/dtheta1 = (1/m)*SUM(...)*d(h_theta(x(i)) - y(i))/dtheta1 = (1/m)*SUM(...)*x(i)

^ | v • Reply • Share ›

Chris McCormick Mod Mathias Sunardi • a year ago

Hey guys, sorry for the confusion. As Mathias said, h(x) is the hypothesis function for linear regression. I've added it into the post there to hopefully clear up the confusion. Thanks!

^ | v • Reply • Share ›

duhaime • 2 years ago

Thanks very much for this post! I'm a little confused though--Wikipedia says the argument to MSE is (y-yhat)**2 but you have (yhat-y)**2. Is the Wiki article wrong?
https://en.wikipedia.org/wi...

^ | v • Reply • Share ›

sina duhaime • 2 years ago

Any number squared is equal to its negative squared. The two terms are equivalent.

^ | v • Reply • Share ›

duhaime sina • 2 years ago

Haha, amen!

^ | v • Reply • Share ›

Jude duhaime • 2 years ago

yeah, I think you are right, in statistics way(acc to wiki). I guess, part of the reason it won't matter too much to keep this style, is in cost function(which is applied in training), only used as predictor(in sample), and there's no difference in result

Related posts

Question Answering with a Fine-Tuned BERT 10 Mar 2020
BERT Research - Ep. 1 - Key Concepts & Sources 11 Nov 2019
GLUE Explained: Understanding BERT Through Benchmarks 05 Nov 2019