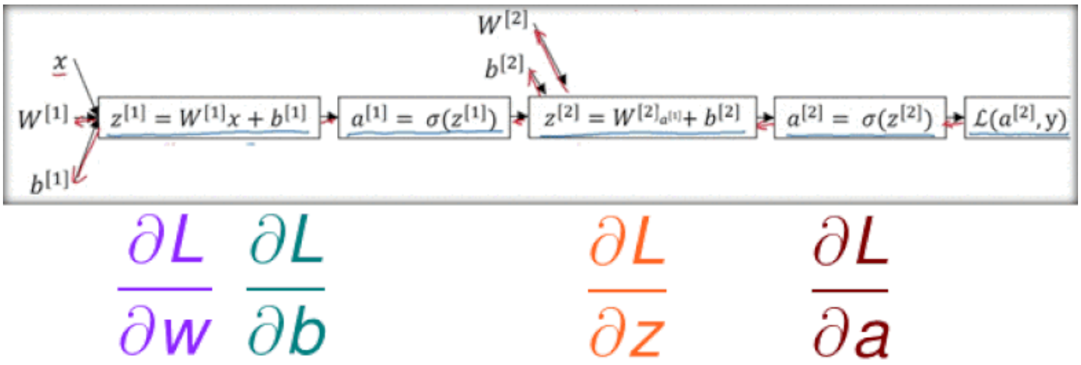
来自链接：https://medium.com/@pdquant/all-the-backpropagation-derivatives-d5275f727f60

By Patrick David
Machine Learning , Deep Learning, AI, Python, Quant Trading -ML Researcher-follow for walkthroughs, tutorials, proofs, research etc. Also on twitter @pdquant
Jun 8

All the Backpropagation derivatives



All the Backprop Derivations

我们知道前馈神经网络是长这样的：

$$
\begin{aligned}
Z^{[1]} &= W^{[1]}X + b^{[1]} \\
A^{[1]} &= g^{[1]}(Z^{[1]}) \\
Z^{[2]} &= W^{[2]}A^{[1]} + b^{[2]} \\
A^{[2]} &= g^{[2]}(Z^{[2]}) \\
&\vdots \\
A^{[L]} &= g^{[L]}(Z^{[L]}) = \hat{Y}
\end{aligned}
$$

Forwardpropagation Equations

然后我们知道反向传播算法是长这样的：

$$dZ^{[L]} = A^{[L]} - Y$$
$$dW^{[L]} = \frac{1}{m} dZ^{[L]} A^{[L]^T}$$
$$db^{[L]} = \frac{1}{m} np.\text{sum}(\text{d}Z^{[L]}, axis = 1, keepdims = True)$$
$$dZ^{[L-1]} = dW^{[L]^T} dZ^{[L]} g'^{[L]}(Z^{[L-1]})$$
$$\vdots$$
$$dZ^{[1]} = dW^{[L]^T} dZ^{[2]} g'^{[1]}(Z^{[1]})$$
$$dW^{[1]} = \frac{1}{m} dZ^{[1]} A^{[1]^T}$$
$$db^{[1]} = \frac{1}{m} np.\text{sum}(\text{d}Z^{[1]}, axis = 1, keepdims = True)$$

Backpropagation Equations

问题是你知不知道这些式子是怎样推导出来的呢?

如果你已经知道了反向传播，但是又不知道诸如

$$A - Y$$

The derivative of our linear function - dz

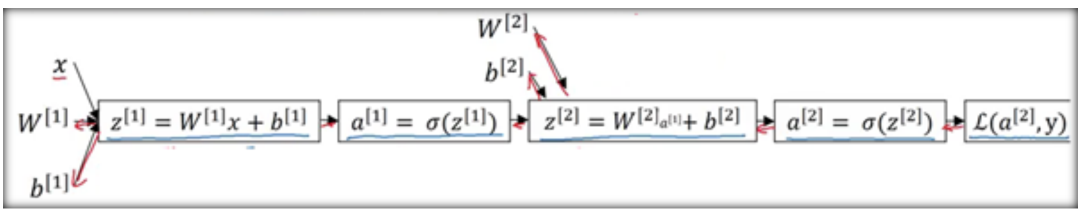
和

$$-\sum \left[\frac{y}{a} + \frac{(1-y)}{1-a}\right]$$

The derivative of our activation function - da

这样的式子是怎样推导出来的, 如果你希望了解直接推导或使用链式法则推导的过程, 请读下去。。。

神经网络



Neural Net taken from Coursera Deep Learning.

这是一个简单的神经网络模型，这里x, W和b是我们的输入，这里的z是这些输入的线性函数变换，这里的a是我们的激活函数（sigmoid），然后最后

$$L(a, y)$$

Cross Entropy cost function

是模型的交叉熵或负对数形式的代价函数。

我们将会从代价函数开始反向传播

$$L(a, y) = -\frac{1}{m}\left[\sum_{i=1}^{m} yln(a) + (1-y)ln(1-a)\right]$$

Our cost function

然后直接进行计算，对代价函数

$$L(a, y)$$

Cross Entropy cost function

进行求导可分为如下几个部分：

$$\frac{\partial L}{\partial a} \frac{\partial L}{\partial z} \frac{\partial L}{\partial w} \frac{\partial L}{\partial b}$$

The derivatives of L(a,y) w.r.t each element in our NN

对上述导数的求解过程，本文将采用直接求解和链式法则两种方式进行讲述。

 # Note: we don’t differentiate our input ‘X’ because these are fixed values that we are given and therefore don’t optimize over.

注：我们不对输入**'X'**进行求导因为**X**是给定的固定的

数值，因此我们不需要在此进行求导学习

---

[1] 代价函数求导

$$\frac{\partial L}{\partial a}$$

[1] derivative of our activation function

首先，我们会对代价函数进行求导

$$L(a, y)$$

激活函数定义为

$$a^{[2]}$$

Activation function 2

因此我们是在对负对数形式的函数（交叉熵）进行求导，可以推导为：

$$\frac{\partial L}{\partial a} - [yln(a) + (1 - y)ln(1 - a)]$$

Taking derivative of our cost function

会得到：（注：上式图像有误）

$$-\left[y\frac{1}{a} - (1 - y)\frac{1}{(1 - a)}\right]$$

derivative of cost function w.r.t 'a2'

在推导过程中，我们用到了log求导的链式法则：

$$\frac{d}{dx}[\ln g(x)] = \frac{1}{g(x)} g'(x)$$

the derivative of a log

ln(x)的导数为x的倒数的证明过程如下：

Proof for the derivative of the natural logarithm:

Let $f(x) = \ln(x)$. To be found: $f'(x) = [\ln(x)]'$.

The natural logarithm is the inverse function of the exponential function e^x, and vice versa, therefore:

$$e^{f(x)} = x$$

We differentiate both sides of the equation:

$$[e^{f(x)}]' = [x]'$$
$$e^{f(x)} \cdot f'(x) = 1$$
$$x \cdot f'(x) = 1$$

Now, we solve for $f'(x)$:

$$f'(x) = \frac{1}{x}$$
$$[\ln(x)]' = \frac{1}{x}$$

proof for the derivative of a log

重写一下上面的结果可以得到：

$$-\sum\left[\frac{y}{a} - \frac{(1-y)}{(1-a)}\right]$$

derivative of L(a,y) w.r.t a2

---

[2] Derivative of sigmoid

[2] 激活函数求导（sigmoid求导）

$$\frac{\partial a}{\partial z}$$

[2] derivative of sigmoid

现在我们来计算一下sigmoid激活函数的导数，后续我们将要用到它。

先来看sigmoid（logistic函数）的定义：

$$\frac{1}{1+e^{-z}}$$

Sigmoid (Logistic) function

首先将式子改写成下面的样子，方便我们使用链式法则进行微分：

$$(1+e^{-z})^{-1}$$

Rearranged sigmoid function

然后使用链式法则：外层函数的导数与内层函数导数的乘积

$$-(1+e^{-z})^{-2}\cdot\left[e^{-z}\cdot -1\right]$$

outer derivative x inner derivative

重写一下得到如下形式：

$$\frac{e^{-z}}{(1+e^{-z})^{2}}$$

put RHS over LHS

下面是一个技巧性的部分。我们将上式变成两个部分的乘积，分母都是1+e^-z，第二部分的分子e^-z，我们进行加一减一的操作，得到：

$$\frac{1}{1+e^{-z}}\cdot\frac{1+(e^{-z})-1}{(1+e^{-z})}$$

add a ‘1’ and subtract a ‘1’ on RHS

第二部分（RHS）现在简化为：

$$\left[1 - \frac{1}{1+e^{-z}}\right]$$

你应该立刻发现，上式其实就是：

$$(1 - sig(z))$$

1 minus our sigmoid

而第一部分本来就是sig(z)，因此，整个结果就是：

$$sig(z) \cdot (1 - sig(z))$$

或者用a表示激活函数（sigmoid）时，简化为：

$$a(1 - a)$$

This notation will be easier

---

[3] 线性函数求导

$$\frac{\partial L}{\partial z}$$

[3] derivative of our linear function (z = wX + b)

To get this result we can use chain rule by multiplying the two results we’ve already calculated [1] and [2]

要得到上式的导数结果，根据链式法则我们可以将[1]和[2]乘起来

multiply derivative [1] by derivative [2]

$$-\left[\frac{y}{a} - \frac{(1-y)}{(1-a)}\right] \bullet a(1-a)$$

der[1] x der[2]

通分：

$$-\left[\frac{y(1-a)}{a(1-a)}-\frac{(a)(1-y)}{(a)(1-a)}\right]\cdot a(1-a)$$

add ‘(1–a)’ and ‘a’ to get common denominator

得到：(注：下图有错误，负号应移入括号中，1-y也应有括号）

$$-\left[\frac{y(1-a)+(a)1-y}{a(1-a)}\right]\cdot a(1-a)$$

common denominator

第一部分分母和第二部分约去，得到：（注：1-y应有括号）

$$-y(1-a)+(a)1-y$$

the remaining numerator

展开得到：

$$-y+ya+a-ay$$

expanded out

最终得到：

$$-y+a$$

所以最终我们算得

$$\frac{\partial L}{\partial z}$$

[3]

的导数是：

$$a - y$$

derivative of our linear function (z = wX +b)

---

[4] 权重值的求导

$$\frac{\partial z}{\partial w}$$

[4] derivative of linear func ‘z’ w.r.t weights ‘w’

这个导数非常容易得到，因为z定义为：

$$w^T X + b$$

linear function ‘z’

因此求导得到：

$$X$$

derivative of ‘z’ w.r.t ‘w’

---

[5] 权重值求导（2）

$$\frac{\partial L}{\partial w}$$

[5] derivative of cost func w.r.t weights ‘w’

我们将采用两种方式计算这个导数，链式法则和直接计算。

链式法则：我们注意到我们已经计算了如下的导数

$$\frac{\partial L}{\partial a} \quad \frac{\partial a}{\partial z} \quad \frac{\partial z}{\partial w}$$

previously computed

Noting that the product of the first two equations gives us

注意到前面两项的乘积其实就是:

$$\frac{\partial L}{\partial z}$$

继续应用链式法则，我们乘上

$$\frac{\partial z}{\partial w}$$

我们将得到

$$\frac{\partial L}{\partial z} \cdot x$$

即:

$$x(a-y)$$

The final result for ‘dw’

或者用链式法则写出来:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}$$

chain rule result for ‘dw’

下面我们直接计算这个导数，还是回到交叉熵代价函数:

$$-\sum[yln(a) + (1-y)ln(1-a)]$$

Cross Entropy cost function

展开第一个log项，代入激活函数得到：

$$ln(1+e^{-z})^{-1}$$

expanding ‘ln(a)’

即：

$$-ln(1+e^{-z})$$

展开第二个log项，带入激活函数得到：

$$ln\frac{e^{-z}}{(1+e^{-z})}$$

ln(1-a)

由log运算法则可得：

$$(-z-ln(1+e^{-z}))$$

log of the numerator

将上面两个中间结果代入到原来的代价函数得到：

$$\sum -yln(1+e^{-z})\left[(1-y)(-z-ln(1+e^{-z}))\right]$$

1*od0TgKCRalB_rbL0rmqkUA.png

plugged back into cost function

将中括号中的乘法进行多项式展开，得到：

$$\sum -yln(1+e^{-z})-z-ln(1+e^{-z})+yz+yln(1+e^{-z})$$

terms inside bracket expanded

消去第一项和最后一项，得到：

$$-z-ln(1+e^{-z})+yz$$

重写一下顺序：

$$yz - \left[z + ln(1 + e^{-z})\right]$$

这里又有一个小技巧，对中括号的第一项z进行求幂然后求对数操作，得到：

$$yz - \left[ln(e^{z}) + ln(1 + e^{-z})\right]$$

we exponentiate ‘e^z’ then take its log

next we can take advantage of the rule of sum of logs: ln(a) + ln(b) = ln(a.b) combined with rule of exp products:e^a + e^b = e^(a+b) to get

运用log运算规则，得到：（注：中括号中的第一个加号应为乘号）

$$yz - \left[ln((e^{z}) + (1 + e^{-z}))\right]$$

summing the logs

然后得到：

$$yz - [ln(1 + e^{z})]$$

summing the exps

重写得到：

$$[yz - ln(1 + e^{z})]$$

最终我们得到代价函数对权重W的偏导数为：

$$\frac{\partial L}{\partial w}[yz - ln(1 + e^{z})]$$

take derivative w.r.t W

第一项yz变为yx，应用链式法则计算第二项后，得到：

$$yx - \frac{x \cdot e^{wx}}{1+e^{wx}}$$

taking derivative of logs again

根据激活函数的定义，实际上第二项就是：

$$x \cdot sig(z)$$

代入后得到：

$$-[yx - x \cdot sig(z)]$$

将x因子提出来：

$$-x\,[y - sig(z)]$$

最终得到：

$$\frac{\partial L}{\partial w} = x\,[a - y]$$

final result

---

[6] 偏置项的导数

$$\frac{\partial L}{\partial b}$$

[6] derivative w.r.t bias b

我们还是可以采用链式法则：

$$\frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial b}$$

chain rule for ‘db’

这个是非常容易计算的，因为z=wX+b对b的偏导数就是1，因此结果就是代价函数对z的偏导数：

$$\frac{\partial L}{\partial z}$$

这个值我们前面已经计算得到：

$$a - y$$

derivative of our linear function (z = wX +b)

这里我们同样使用直接计算推导偏置项的导数。这里我们用到前面算权重值导数时的中间结果：

$$[yz - ln(1 + e^{z})]$$

note: z = wX + b

将z=wX+b代入上式，得到

$$\frac{\partial L}{\partial b}y(wX + b) - \frac{\partial L}{\partial b}ln(1 + e^{wX+b})$$

0*VgWLH92SBmRefdWY

note the parenthesis

左边求导得到y，因为第一项不含b，右边求导得到：

$$\frac{e^{wX+b}}{1 + e^{wX+b}}$$

derivative of ln(1+e^wX+b)

分子分母同除e^(wX+b)，就可以得到激活函数（sigmoid），即：

$$a$$

左右组合在一起得到

$$[a-y]$$

final result

其实就是代价函数对z的偏导数：

$$\frac{\partial L}{\partial z}$$

'db' = 'dz'

这样我们就推到了神经网络的所有导数。我们计算了一下的所有偏导数：

$$\frac{\partial L}{\partial a} \frac{\partial L}{\partial z} \frac{\partial L}{\partial w} \frac{\partial L}{\partial b}$$

The derivatives of L(a,y) w.r.t each element in our NN
