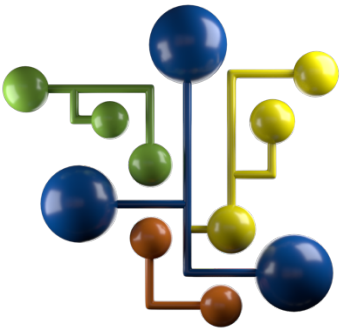


Data Science Academy

www.datascienceacademy.com.br

Deep Learning I

Compreendendo o Conceito de Derivada e Chain Rule

Agora vamos discutir o gradiente em maior profundidade.

A fim de descobrir como devemos alterar um parâmetro para minimizar o custo, primeiro devemos descobrir o efeito desse parâmetro no custo. Isso faz sentido. Afinal, não podemos simplesmente mudar cegamente os valores dos parâmetros e esperar obter resultados significativos. O gradiente leva em conta o efeito que cada parâmetro tem no custo, então é assim que encontramos a direção da subida mais íngreme.

Como determinamos o efeito que um parâmetro tem no custo? Esta técnica é conhecida como backpropagation ou diferenciação de modo reverso. Esses nomes podem parecer intimidantes, mas, por trás disso, é apenas uma aplicação inteligente da regra da cadeia ou Chain Rule (acesse esse link caso queria mais detalhes matemáticos sobre esse assunto: http://tutorial.math.lamar.edu/Classes/CalcI/ChainRule.aspx).

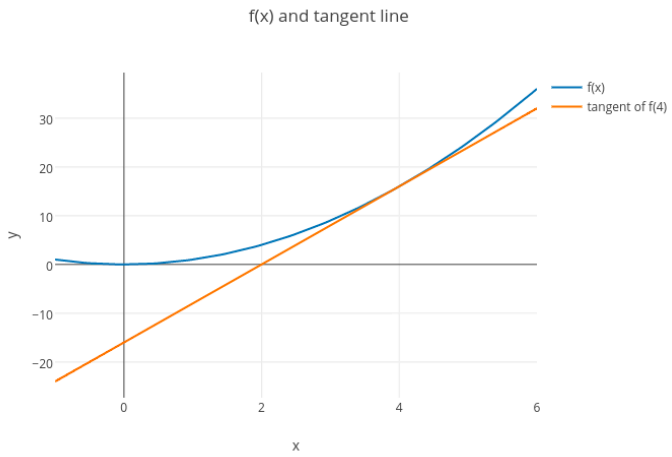
Mas antes de falarmos um pouco mais sobre a Chain Rule, vamos visitar o conceito de Derivadas.

Derivadas

Em cálculo, a derivada nos diz como algo muda em relação a outra coisa. Ou, de forma diferente, quão sensível é algo para outra coisa.

Vamos considerar a função f(x) = x^2 como exemplo. Neste caso, a derivada de f(x) é 2x. Outra maneira de declarar isso é "a derivada de f(x) em relação a x é 2x".

Usando a derivada, podemos dizer o quanto uma mudança em x afeta f(x). Por exemplo, quando x é 4, a derivada é 8 (2x = 2 * 4 = 8). Isso significa que se x for aumentado ou diminuído em 1 unidade, então f(x) aumentará ou diminuirá em 8.



Observe que f(4) = 16 e f(5) = 25 e se subtrairmos 25 - 16 = 9, percebemos que obviamente isso não é o mesmo que 8.

Mas acabamos de definir no exemplo anterior, que o aumento de x por 1 unidade mudaria f(x) em 8. O que aconteceu?

A resposta é que a inclinação (ou derivada) em si muda à medida que x muda. Se calculamos a derivada quando x é 4.5, o resultado será 9, que corresponde à diferença entre f(4) = 16 e f(5) = 25.

Chain Rule

Voltemos às redes neurais e o objetivo original de descobrir o efeito que um parâmetro tem no custo. Nós simplesmente calculamos a derivada do custo em relação a cada parâmetro na rede. O gradiente é um vetor de todas essas derivadas.

Na realidade, as redes neurais são uma composição de funções, de modo que a computação da derivada do custo de um parâmetro não é tão simples quanto o cálculo da derivada de uma função polinomial como f (x) = x^2. É aí que a regra da cadeia entra em jogo.

Para explicar esse conceito, que tem um nível de complexidade relativamente alto, vamos reproduzir em português parte do paper de Erik Learned-Miller da Universidade de Stanford.

O paper original em inglês você encontra em: http://cs231n.stanford.edu/vecDerivs.pdf. Recomendamos a leitura do paper original!

Digamos que temos uma nova função f◦g (x) = f (g (x)). Podemos calcular a derivada de f◦g w.r.t x, indicado por $\frac{\partial f \circ g}{\partial x}$, aplicando a regra da cadeia:

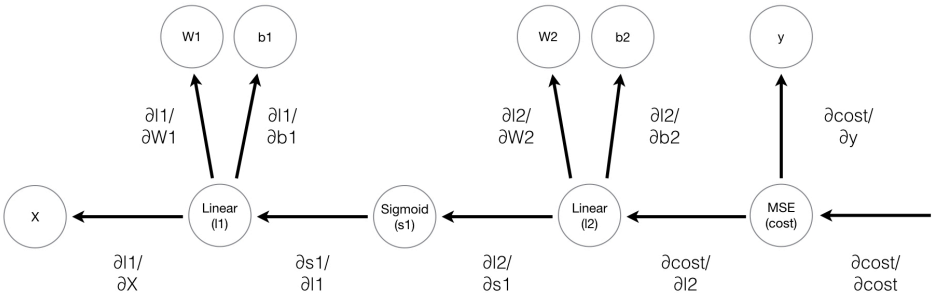
$$\frac{\partial f \circ g}{\partial x} = \frac{\partial g}{\partial x} \frac{\partial f}{\partial g}$$

A maneira de pensar sobre isso é: para conhecer o efeito que x tem em f, primeiro precisamos saber o efeito que x tem em g, e então o efeito que g tem em f. Em nosso algoritmo, isso seria equivalente a:

```
X, y = Input(), Input()
W1, b1 = Input(), Input()
W2, b2 = Input(), Input()

l1 = Linear(X, W1, b1)
s1 = Sigmoid(l1)
l2 = Linear(s1, W2, b2)
cost = MSE(l2, y)
```

Isso também pode ser escrito como uma composição da função MSE (Linear (Sigmoid (Linear (X, W1, b1)), W2, b2), y). Nosso objetivo é ajustar os pesos e os bias representados pelos nós de entrada W1, b1, W2, b2, de modo que o custo seja minimizado.



Neste grafo da rede neural acima, vemos a passagem para trás e os gradientes fluindo ao longo do processo. Backpropagation facilita a computação desses valores.

Durante o backpropagation, as derivadas dos nós no grafo são calculadas de volta à frente. Esta visão facilita a implementação de backpropagation. Ao calcular a passagem para trás para um nó, precisamos apenas nos preocupar com a computação desse nó e com as suas entradas.

Agora, vamos adicionar esses cálculos ao nosso algoritmo!

Mais detalhes sobre esse tópico aqui:

http://www.deeplearningbook.com.br