Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Automação e Sistemas

Aprendizado supervisionado
Prof. Eric Antonelo

# Treinamento de Modelo

## Descenso do Gradiente

Já temos nossa função de hipótese $h_\theta(x)$ e um jeito de medir o quanto esta se ajusta aos dados. Agora, precisamos estimar os parâmetros da função de hipótese. Isto será feito através do método do descenso do gradiente.

Imagine que nós plotemos nossa função de de custo $J(\theta_0, \theta_1)$ em função dos parâmetros $\theta_0, \theta_1$ , onde cada combinação de valores para $\theta_0, \theta_1$ define uma certa hipótese $h_\theta(x)$ .

Colocamos $\theta_0$ no eixo x, e $\theta_1$ no eixo y, e a função de custo no eixo vertical z. No gráfico abaixo, cada ponto no plano x-y define uma hipótese, de infinitas possíveis. A superfície do gráfico representa o custo de usar uma certa hipótese $\theta_0, \theta_1$ como modelo para os dados de treinamento. Este custo é computado pela função de custo $J(\theta_0, \theta_1)$ .

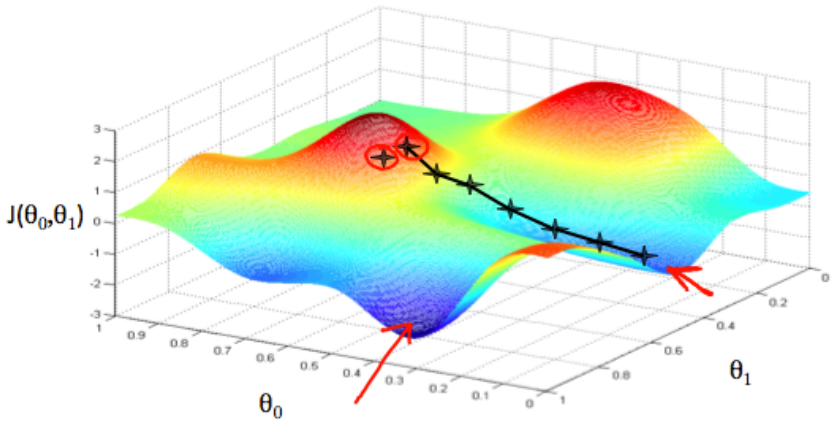


Figura 1:

Aqui, o objetivo é encontrar valores de $\theta_0, \theta_1$ tal que a função de custo seja minimizada, ou seja, que seu valor esteja no fundo dos vales do gráfico acima. As flechas em vermelho marcam os pontos mínimos no gráfico.

Para fazer isso, precisamos calcular a derivada (a reta tangencial a uma função) de nossa função de custo com respeito aos parâmetros $\theta_0, \theta_1$ . A inclinação da tangente é a derivada naquele ponto e que nos fornecerá uma direção para mover $\theta_0, \theta_1$ . Descemos gradativamente na função de custo na direção da descida mais íngrime. O tamanho de cada passo é determinado por um parâmetro $\alpha$ chamado **taxa de aprendizagem**.

Por exemplo, a distância entre cada "estrela"no gráfico acima representa um passo determinado pelo parâmetro $\alpha$. Um $\alpha$ baixo resulta num passo menor, enquanto um $\alpha$ mais alto num passo maior. A direção em que o passo é feito é determinada pela derivada parcial de $J(\theta_0, \theta_1)$ . Dependendo de onde o processo é iniciado no gráfico, ou seja, dos valores iniciais de $\theta_0, \theta_1$ , pode-se terminar em diferentes pontos e, consequentemente, com diferentes valores da função de custo. Ou seja, o valor inicial de $\theta_0, \theta_1$ pode resultar em hipóteses ($\theta_0, \theta_1$ ) que se ajustam melhor ou pior aos dados de treinamento.

O algoritmo de descenso do gradiente é:

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

onde $j = 0, 1$ representa o índice da característica de entrada (*input feature*).

A cada iteração $j$, deve-se atualizar os parâmetros $\theta_0, \theta_1, \theta_2, ..., \theta_n$ simultaneamente. Atualizando um parâmetro antes dos outros na $j$-ésima iteração resultaria em uma implementação errônea:

Incorreto:

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

Correto: atualização simultânea

$\texttt{valor0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$
$\texttt{valor1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$
$\theta_0 = \texttt{valor0}$
$\theta_1 = \texttt{valor1}$

## Entendendo Descenso do Gradiente

Supondo que nossa hipótese contenha somente um parâmetro $\theta_1$, a fórmula do método fica:

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

Independente do sinal da inclinação dado por $\frac{\partial}{\partial \theta_1} J(\theta_1)$, $\theta_1$ converge eventualmente para o valor mínimo. O gráfico seguinte mostra que, quando a inclinação é negativa, o valor de $\theta_1$ aumenta e, quando é positiva, o valor de $\theta_1$ diminui.

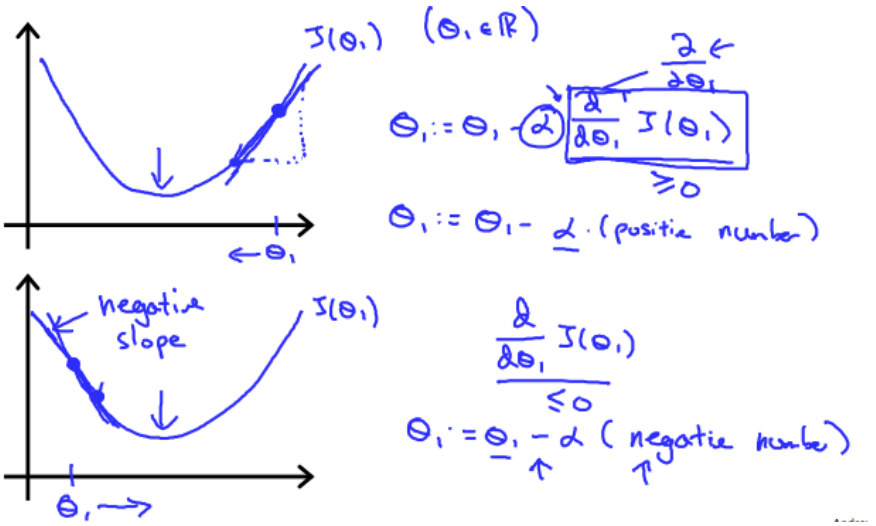


Figura 2:

Devemos escolher o valor de $\alpha$ de tal modo que o algoritmo do descenso do gradiente convirja em um tempo razoável. O tamanho do passo estará incorreto se o método não convergir (até mesmo divergir) ou demorar demasiadamente para obter o valor mínimo.

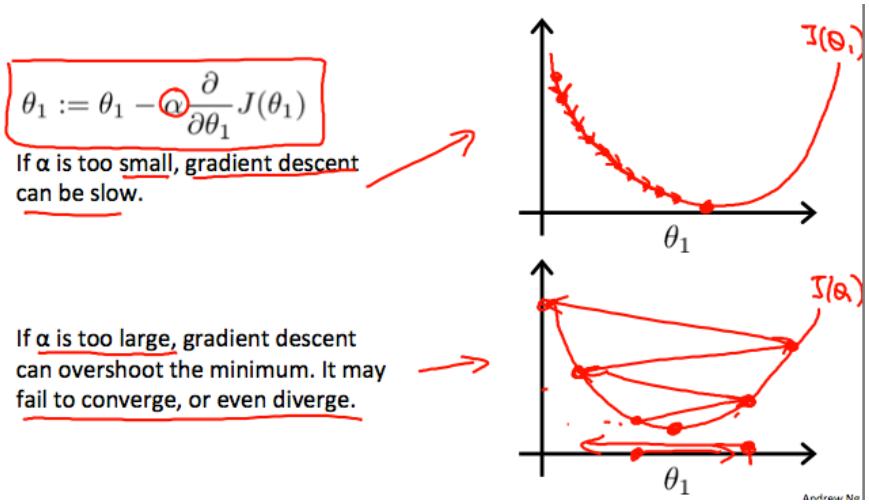


Figura 3:

### Como o método do descenso do gradiente converge com um tamanho de passo fixo $\alpha$ ?

A compreensão advém do fato de que $\frac{\partial}{\partial \theta_1} J(\theta_1)$ tende a zero à medida que nos aproximamos do fundo de nossa função convexa. No mínimo da função, a derivada será sempre 0 e, consequentemente:

$\theta_1 := \theta_1 - \alpha * 0$

Quanto mais próximo estamos de um mínimo local, o método de descenso do gradiente automaticamente emprega tamanho de passos menores. Em geral, não precisamos diminuir $\alpha$ ao longo do tempo (a não ser em problemas específicos).

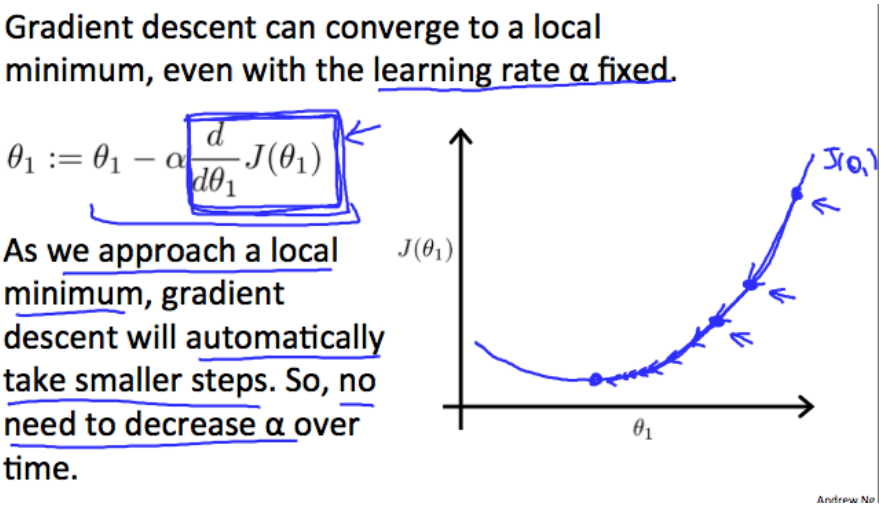


Figura 4:

## Descenso do Gradiente para Regressão Linear

Quando aplicado especificamente para o caso de regressão linear, podemos expandir o cálculo da derivada parcial da função de custo para o modelo linear em questão. onde $m$ é o tamanho do conjunto de treinamento, $\theta_0$ um parâmetro que mudará simultaneamente com $\theta_1$, e $x^{(i)}, y^{(i)}$ são valores de entrada e saída desejada do conjunto de dados de treinamento.

**Algorithm 1:** Descenso do gradiente para uma hipótese linear $h_\theta(x) = \theta_0 + \theta_1 * x$

**while** *não convergir* **do**
  $\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$;
  $\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( (h_\theta(x^{(i)}) - y^{(i)}) * x^{(i)} \right)$;
**end while**

Note que separamos os dois casos para $\theta_j$ em equações individuais para $\theta_0$ e $\theta_1$; e para $\theta_1$, nós multiplicamos $x^{(i)}$ no final devido à derivada.

A seguir, calculamos $\frac{\partial}{\partial \theta_j} J(\theta_j)$ para somente um exemplo $(x, y)$ do conjunto de treinamento:

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta_j) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 && (1)\\
&= 2.\frac{1}{2} (h_\theta(x) - y) \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) && (2)\\
&= (h_\theta(x) - y) \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) && (3)\\
&= (h_\theta(x) - y) \frac{\partial}{\partial \theta_j} \left( \sum_{i=0}^{n} \theta_i x_i - y \right) && (4)\\
&= (h_\theta(x) - y) x_j && (5)
\end{aligned}$$

onde $x$ é um vetor de características de entrada para um exemplo, e $x_i$ é o $i$-ésimo componente do vetor. Veja que o Algoritmo 1 passa por todos os $m$ exemplos de treinamento. A expressão em (5) entra dentro da somatória $\sum_{i=1}^{m}$ para todos exemplos, oriunda da definição da função de custo.

Se começarmos com uma estimativa para nossa hipótese e, em seguida, aplicarmos repetidamente essas equações do método do descenso do gradiente, nossa hipótese se tornará cada vez mais precisa. Isto é o que o método do descenso do gradiente faz quando aplicada na função de custo original $J(\theta)$. Este método passa por todos os exemplos do conjunto de treinamento em cada iteração do *loop* do Algoritmo 1. Essa passagem de uma iteração do método por todos exemplos é chamada de **época**, resultando **no descenso do gradiente em lote** (*batch gradient descent*). Observe que, embora esse algoritmo possa ser suscetível a mínimos locais em geral, o problema de otimização que colocamos aqui para a regressão linear tem apenas um ótimo global e nenhum outro local; assim, o descenso do gradiente sempre converge (assumindo que a taxa de aprendizado $\alpha$ não seja muito grande) para o mínimo global. Para o caso linear, $J$ é uma função quadrática convexa. Aqui está um exemplo de descenso do gradiente executado para minimizar uma função quadrática.

As elipses mostradas acima são os contornos de uma função quadrática. Também é mostrada a trajetória tomada pelo descenso do gradiente, que foi inicializado em (48,30). Os pontos x's na figura (unidos por linhas retas) marcam os valores sucessivos de $\theta$ pelos quais o descenso do gradiente passou à medida que convergiu para seu mínimo.

# Bibliografia


- Curso de Machine Learning de Andrew Ng.
- Bishop, Christopher M. Pattern recognition and machine learning. springer, 2006.
- Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016.
- Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: data mining, inference, and prediction. Springer Science & Business Media, 2009. (`https://web.stanford.edu/~hastie/Papers/ESLII.pdf`)

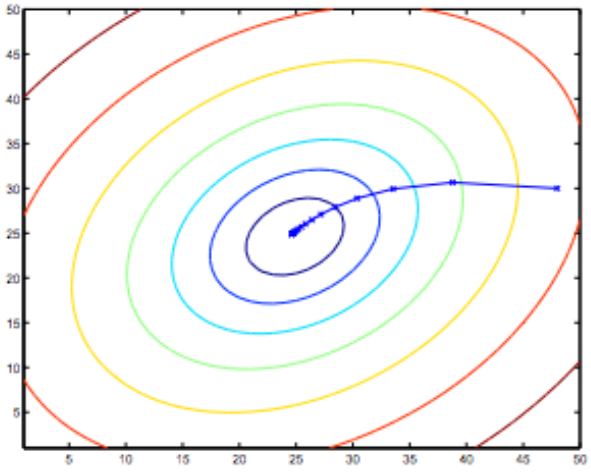


Figura 5: