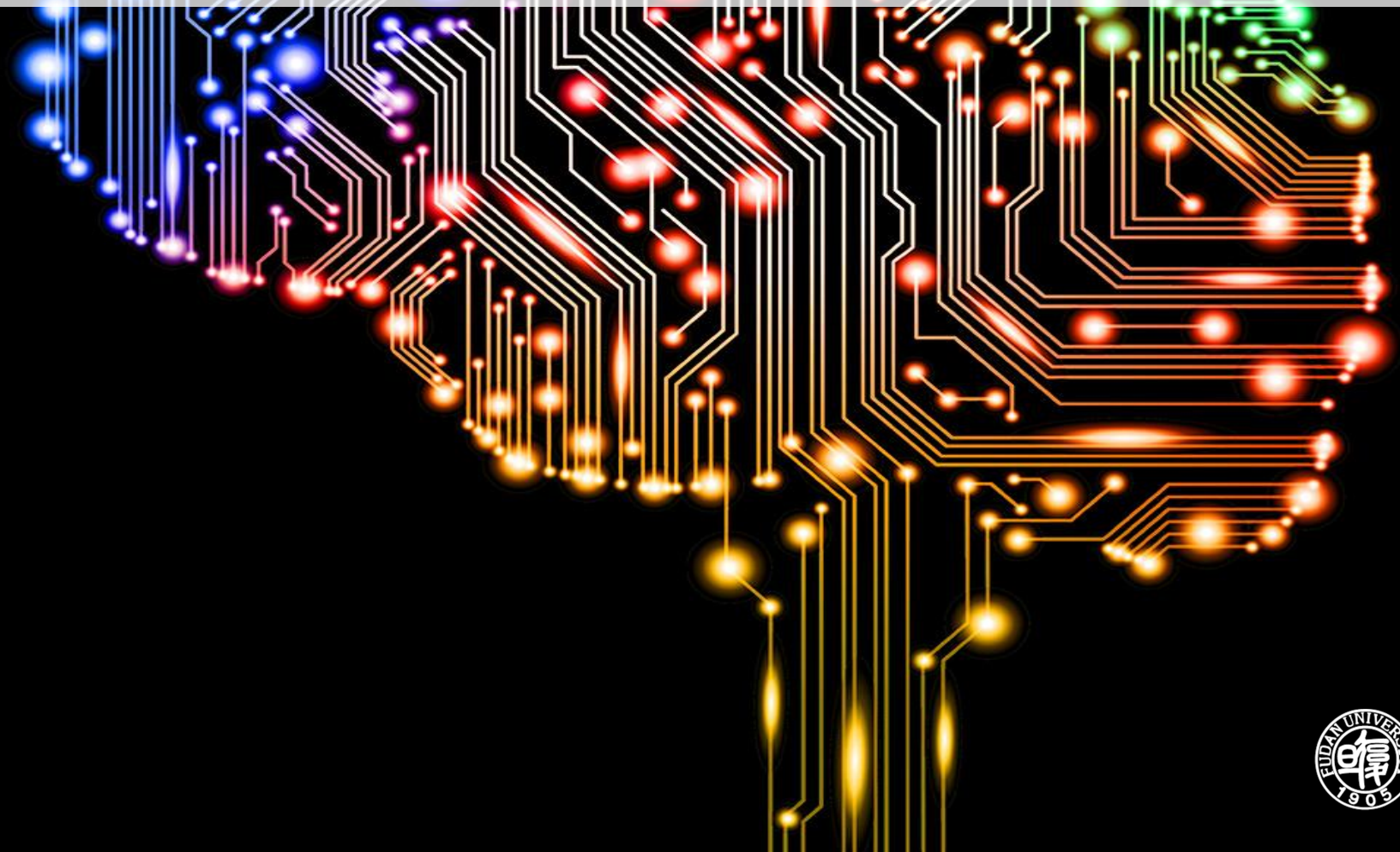
数据驱动的人工智能（2a）机器学习基础

Data Driven Artificial Intelligence

邬学宁 SAP硅谷创新中心

2017 / 02



目录

日程

Linear Regression
Cost Function
Gradient Descent / Normal Equation
Logistic Regression
Perceptron
Machine Learning System Design

变量类型

Categorical 类别 (Qualitative 定性)	Numerical 数值 (quantitative 定量)
Nominal 名词性	Interval 区间性
Ordinal 顺序性	Ratio 比例性

Discrete 离散

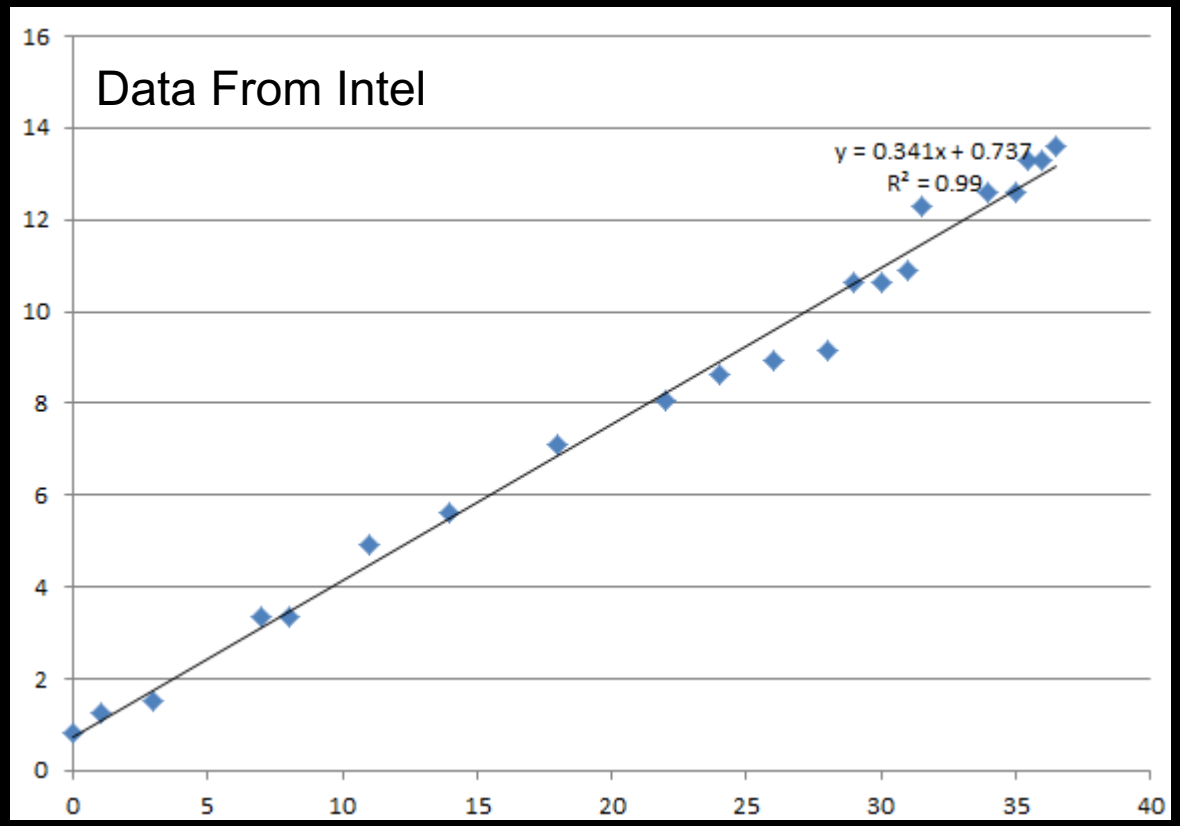
Continuous 连续

回归／分类

	分类 **Classification**	回归 **Regression**
输出类型	离散（分类标签）	连续（数值）
分析目标	Decision Boundary	Best Fit Line
评价	Accuracy / Confusion Matrix	Mean Squared Error / R^2

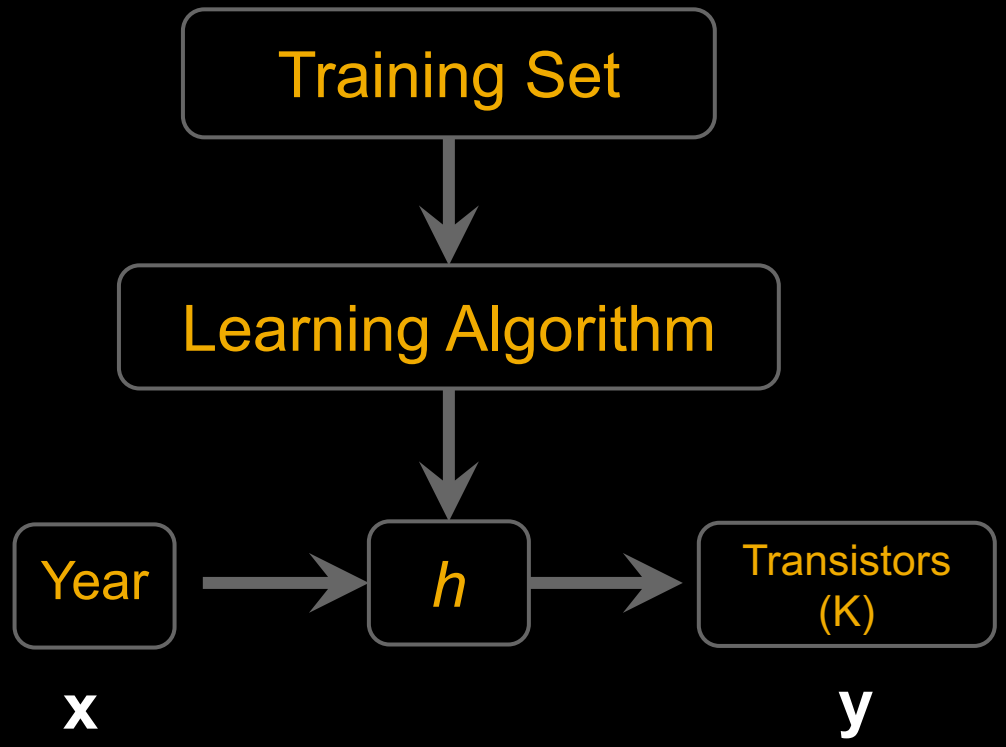
Linear Regression (摩尔定律的线性回归)

Time(years)	Transistors (1K)	log(Transistors)
0	2.3	0.83
1	3.5	1.25
3	4.5	1.50
7	29	3.37
8	29	3.37
11	134	4.90
14	275	5.62
18	1200	7.09
22	3100	8.04
24	5500	8.61
26	7500	8.92
28	9500	9.16
29	42000	10.65
30	42000	10.65
31	55000	10.92
31.5	220000	12.30
34	291000	12.58
35	291000	12.58
35.5	582000	13.27
36	582000	13.27
36.5	820000	13.62



Univariate Linear Regression 单变量线性回归

Model Representation 模型表达与标记说明



如何表达H?

$$H_\theta(x) = \theta_0 + \theta_1 x \qquad (1)$$

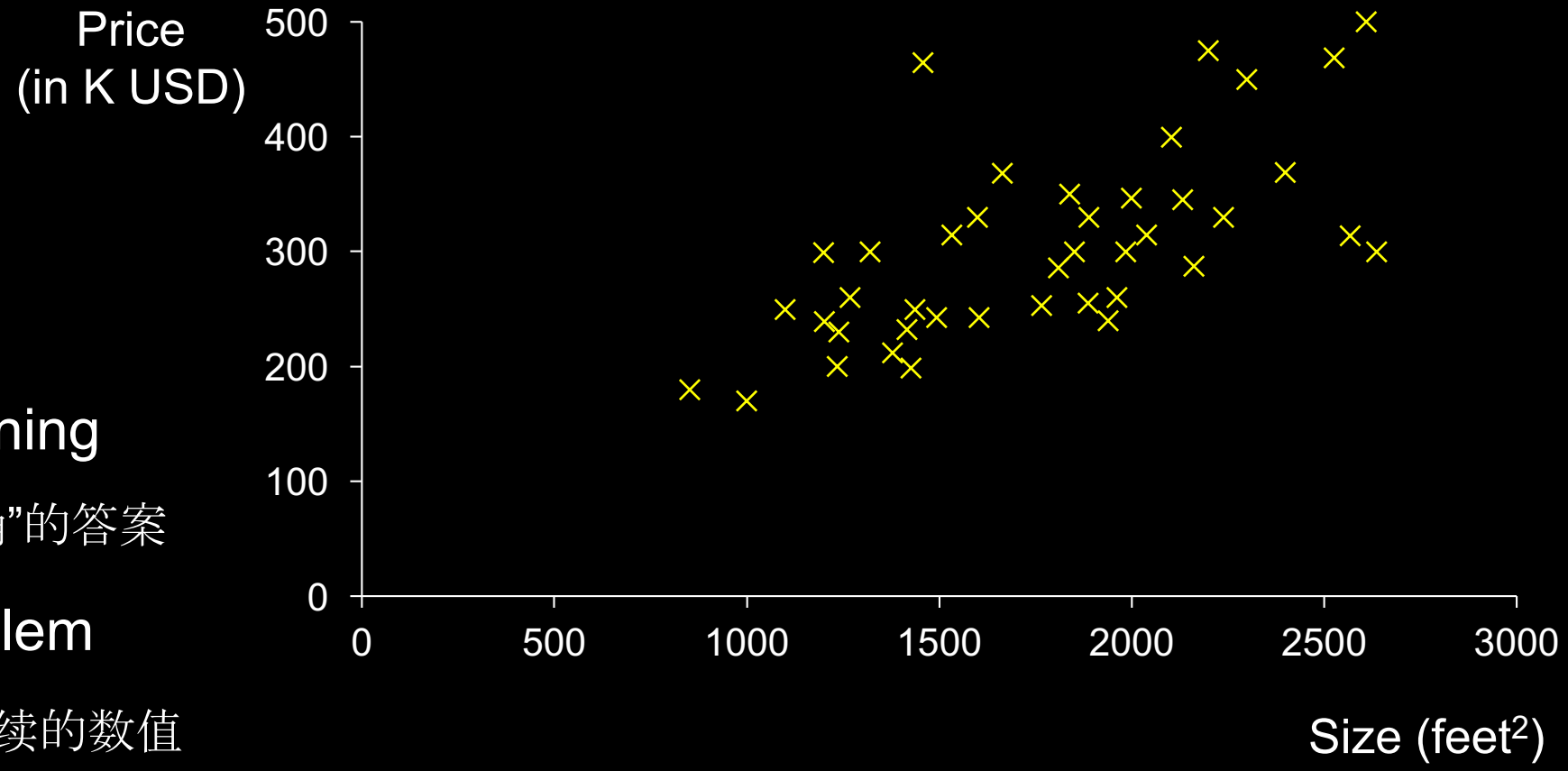
Notation:

- **m** = Number of training examples (rows)
- n = Number of features (columns)
- **x**'s = "input" variable / features (independent)
- **y**'s = "output" / "target" variable (dependent)
- Turple $(x^{(i)}, y^{(i)})$ represent the i^{th} training example

房价训练数据集 (Portland)

Size in feet^2 (x)	Price in 1K USD (y)
2104	460
1416	232
1534	315
852	178
…	…

房价训练数据集 (Portland)



- Supervised Learning

  给每个数据样本“正确”的答案

- Regression Problem

  预测目标：房价是连续的数值

房价训练数据集 (Portland)

Size in feet² (x)	Price in 1K USD (y)
2104	460
1416	232
1534	315
852	178
…	

Hypothesis: $H_\theta(x) = \theta_0 + \theta_1 x$

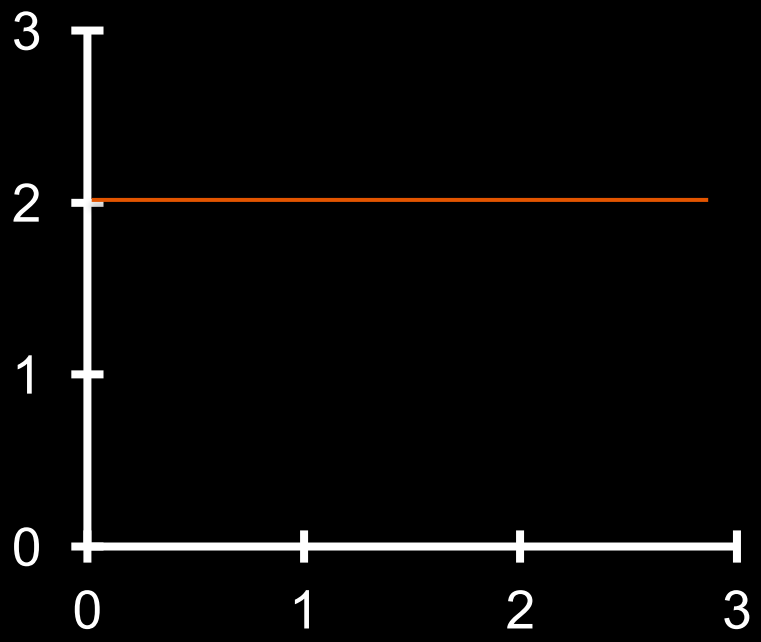
θ‘s: Parameters / Weight

如何选择θ ‘s ?

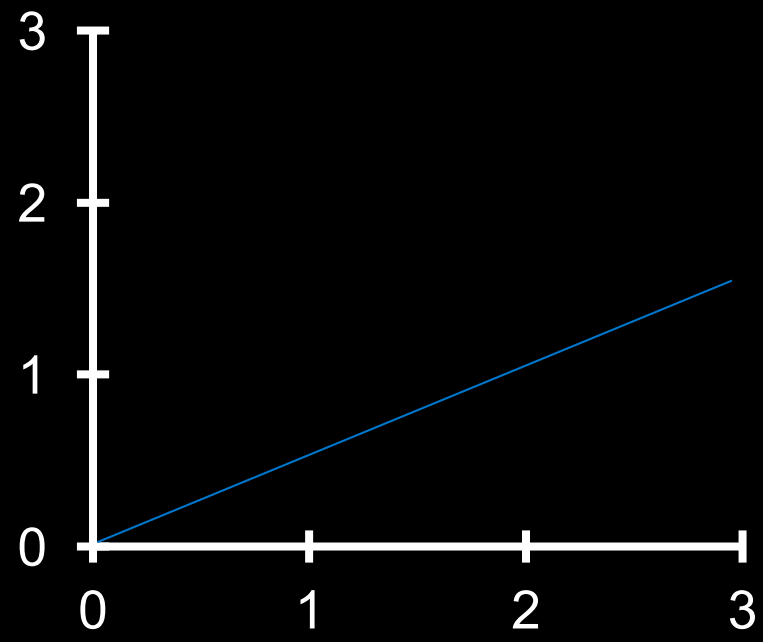
$$h_\theta(x) = \theta_0 + \theta_1 x$$

房价训练数据集 (Portland)

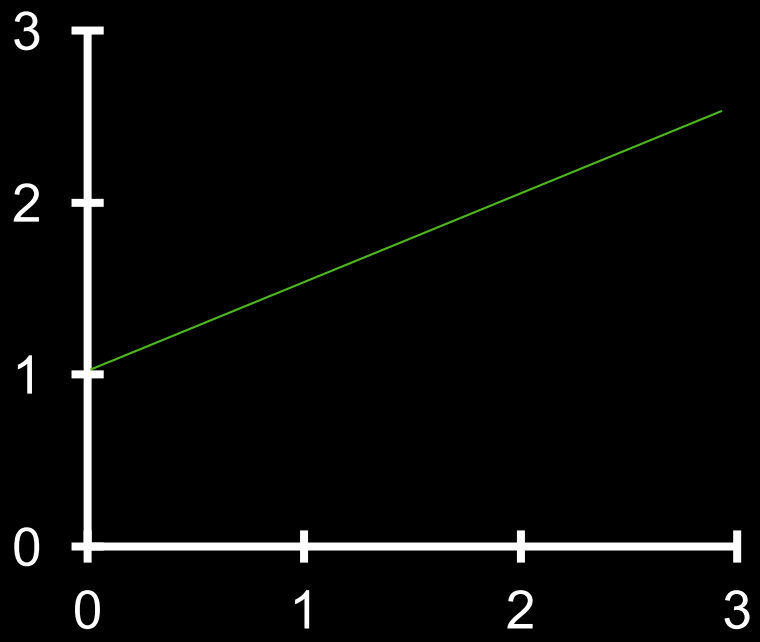
Hypothesis:$H_\theta(x) = \theta_0 + \theta_1 x$



$\theta_0 = 2$
$\theta_1 = 0$



$\theta_0 = 0$
$\theta_1 = 0.5$



$\theta_0 = 1$
$\theta_1 = 0.5$

SAP

FUDAN UNIVERSITY 1905 復旦大學

Univariate Linear Regression

Cost Function 代价函数 / 损失函数

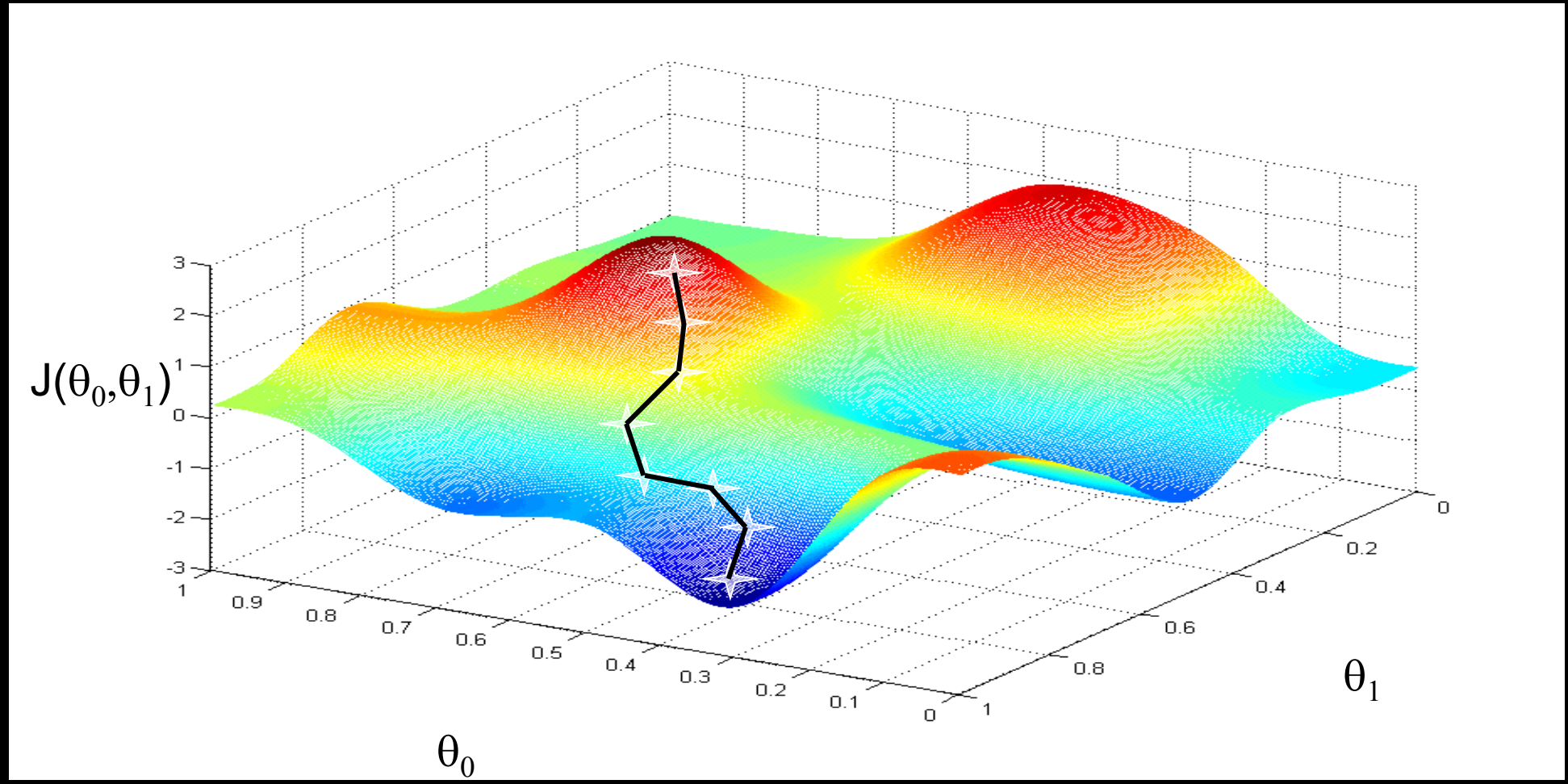
如何表达模型的误差?

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

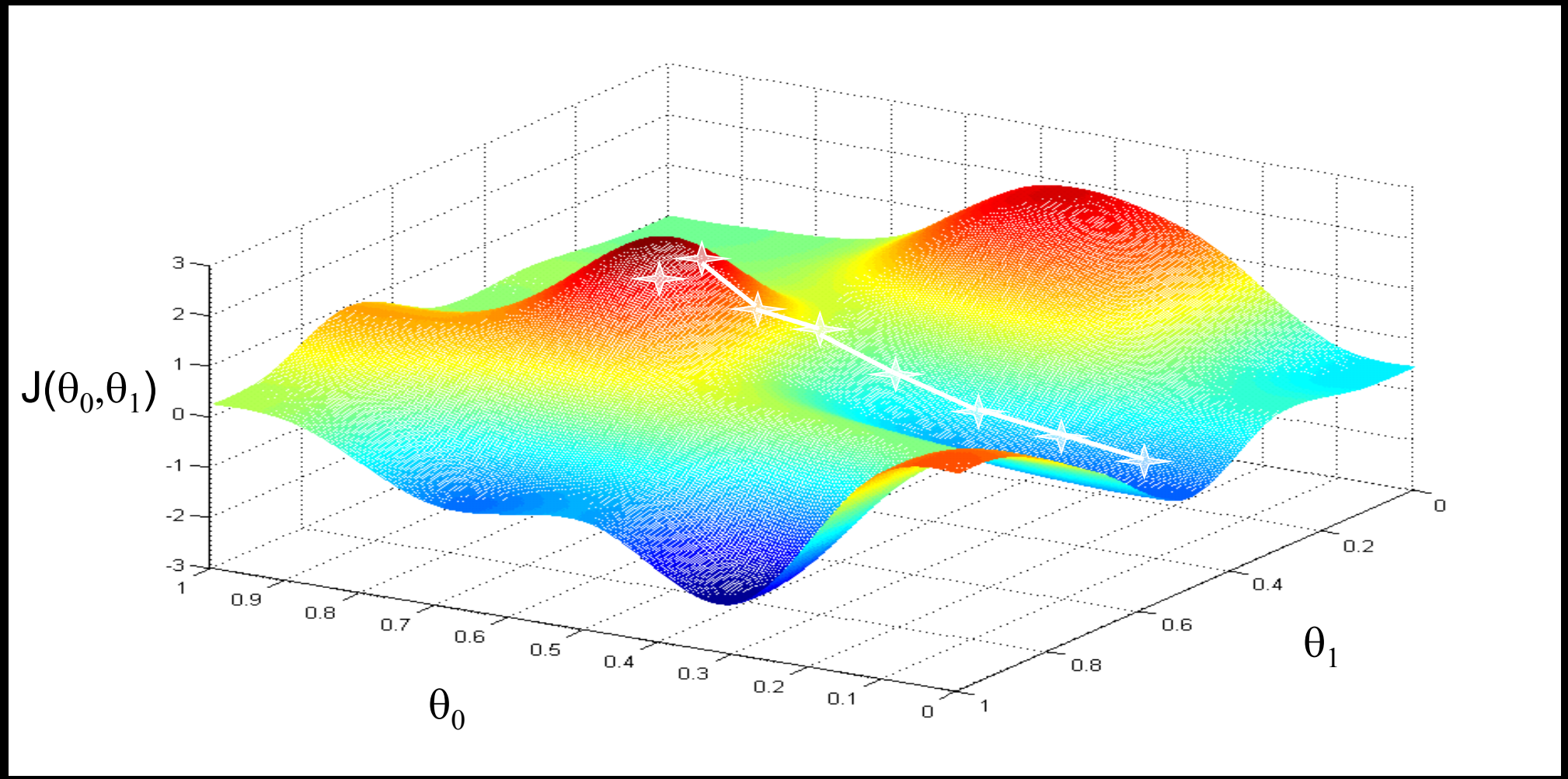
目标: 通过调整 θ_0, θ_1 ，最小化 J (θ_0, θ_1)

方法: Gradient Descent

梯度下降 Gradient Descent



梯度下降 Gradient Descent



Univariate Linear Regression

Gradient Descent

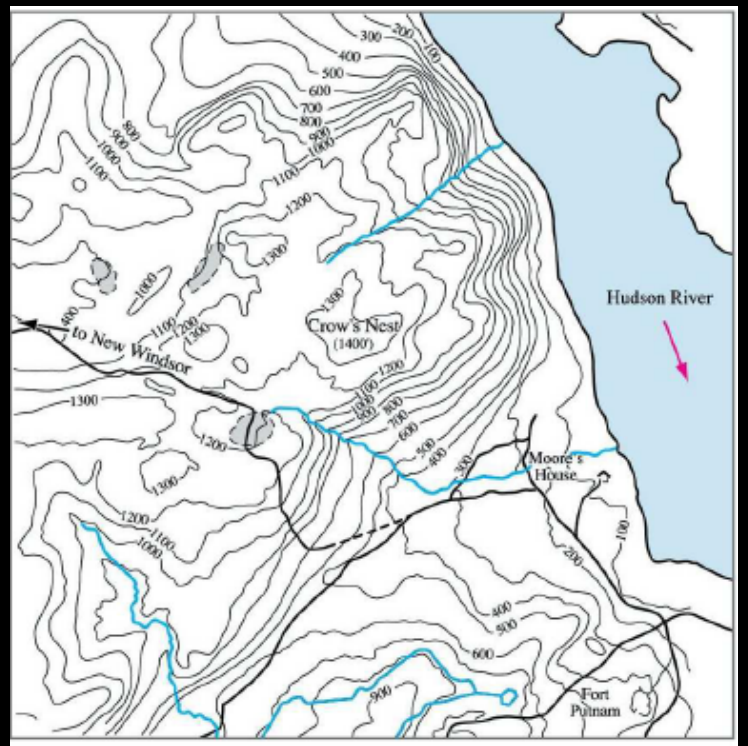
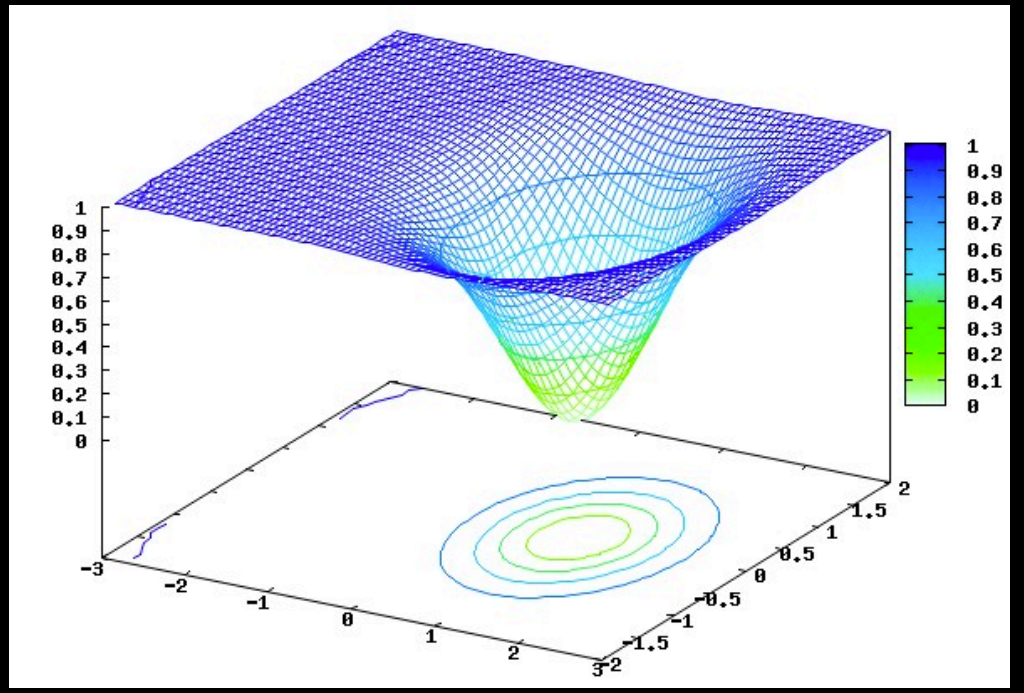


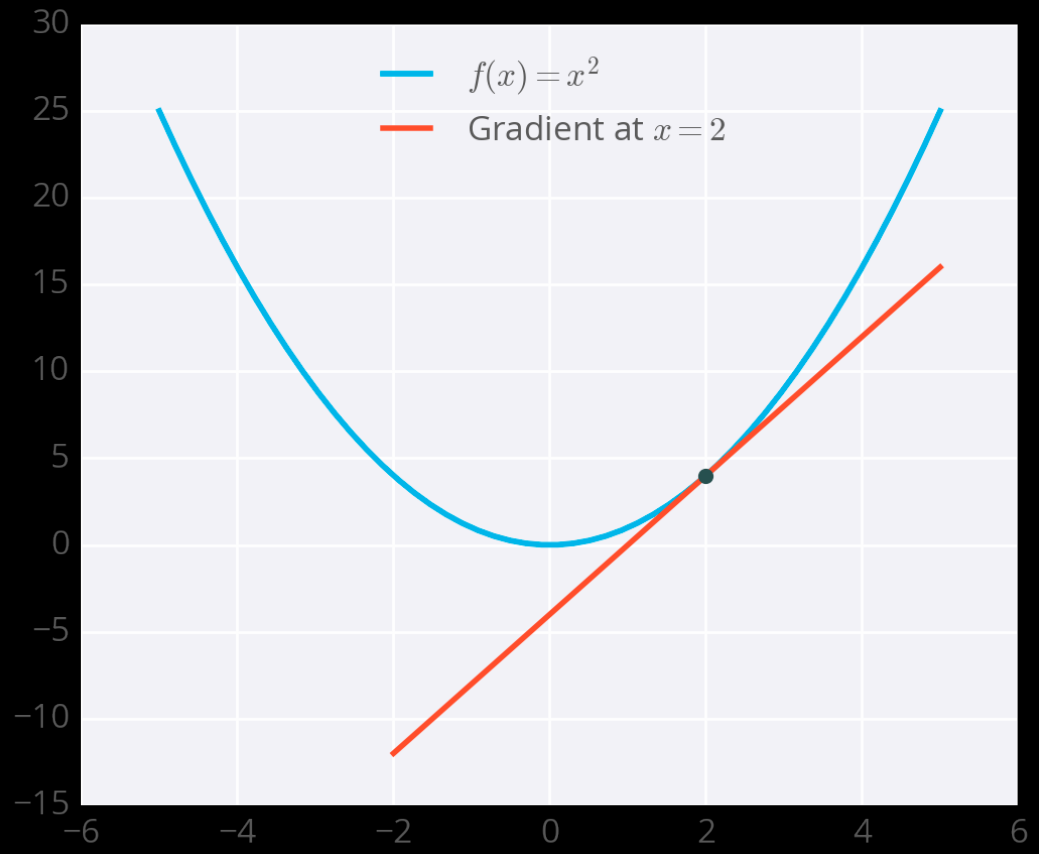
FIGURE 14.25 Contours along the Hudson River in New York show streams, which follow paths of steepest descent, running perpendicular to the contours.



Credit: Thomas Calculus

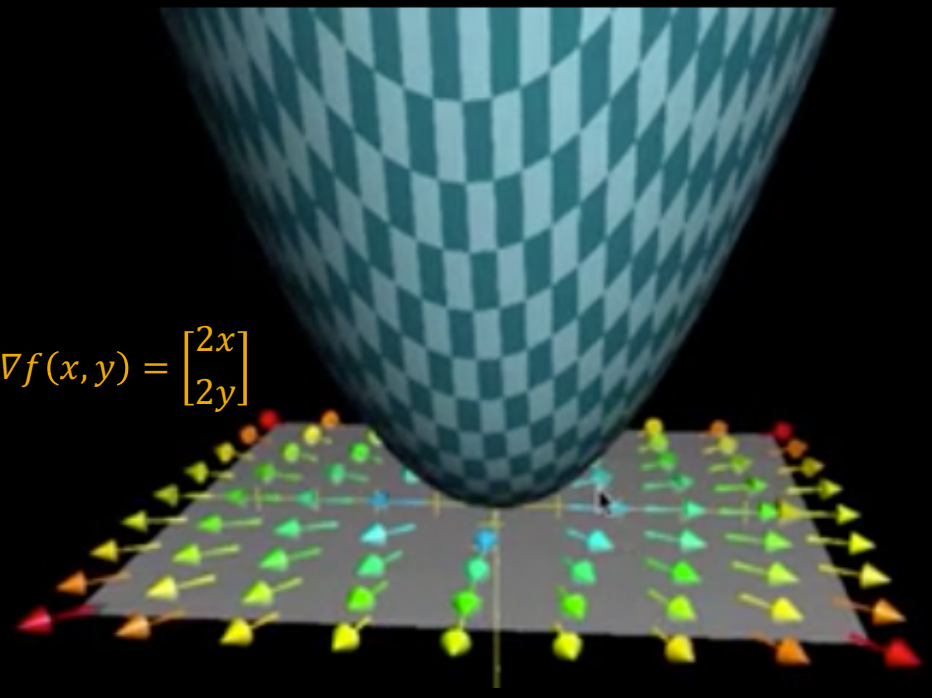
Univariate Linear Regression

Gradient is the direction of steepest ascent 梯度是最陡峭的上升方向



$f(x, y) = x^2 + y^2$

$\nabla f(x, y) = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$



Credit: Khan Academy

Univariate Linear Regression

Gradient Recap

$$f(x, y) = x^2 \sin y$$

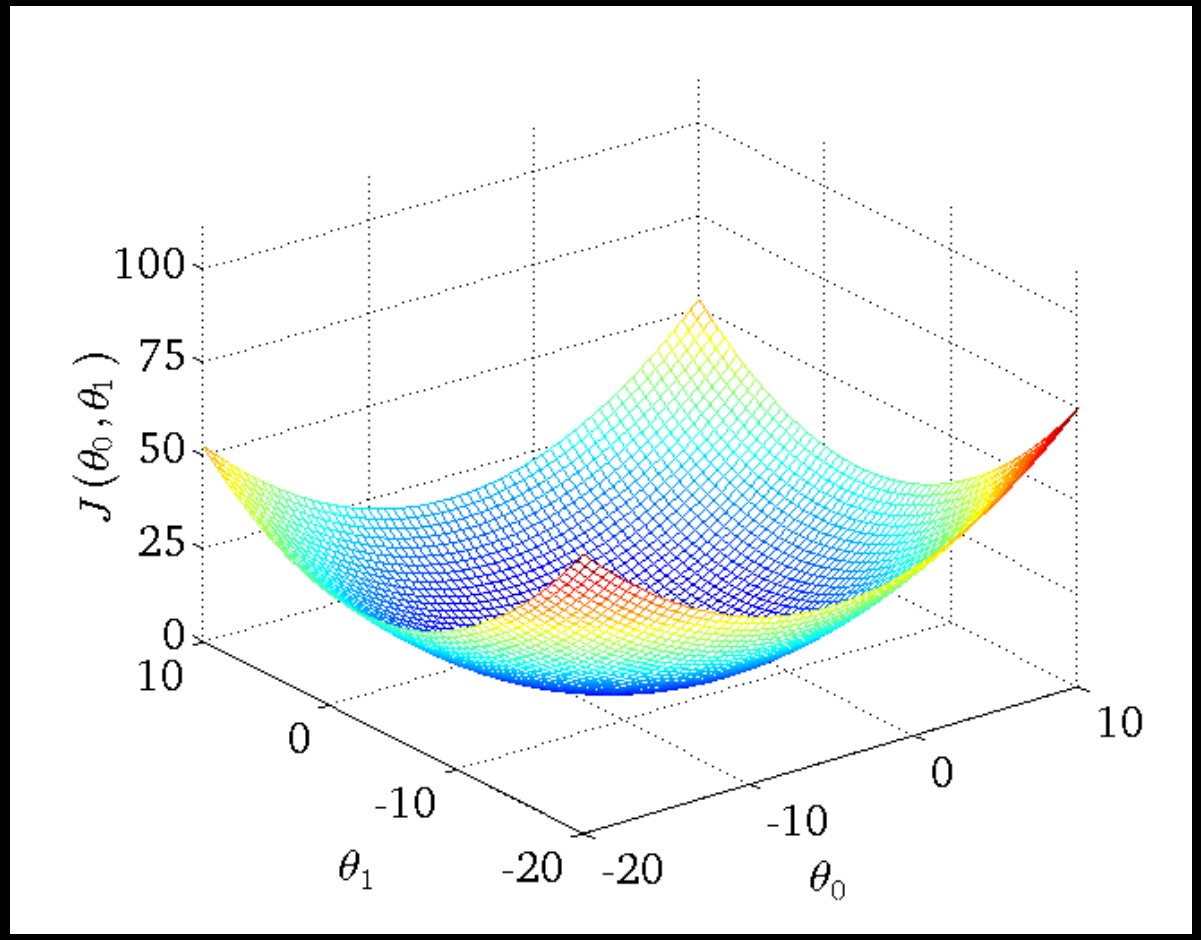
$$\frac{\partial f}{\partial x} = 2x \sin y$$

$$\frac{\partial f}{\partial y} = x^2 cos\, y$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

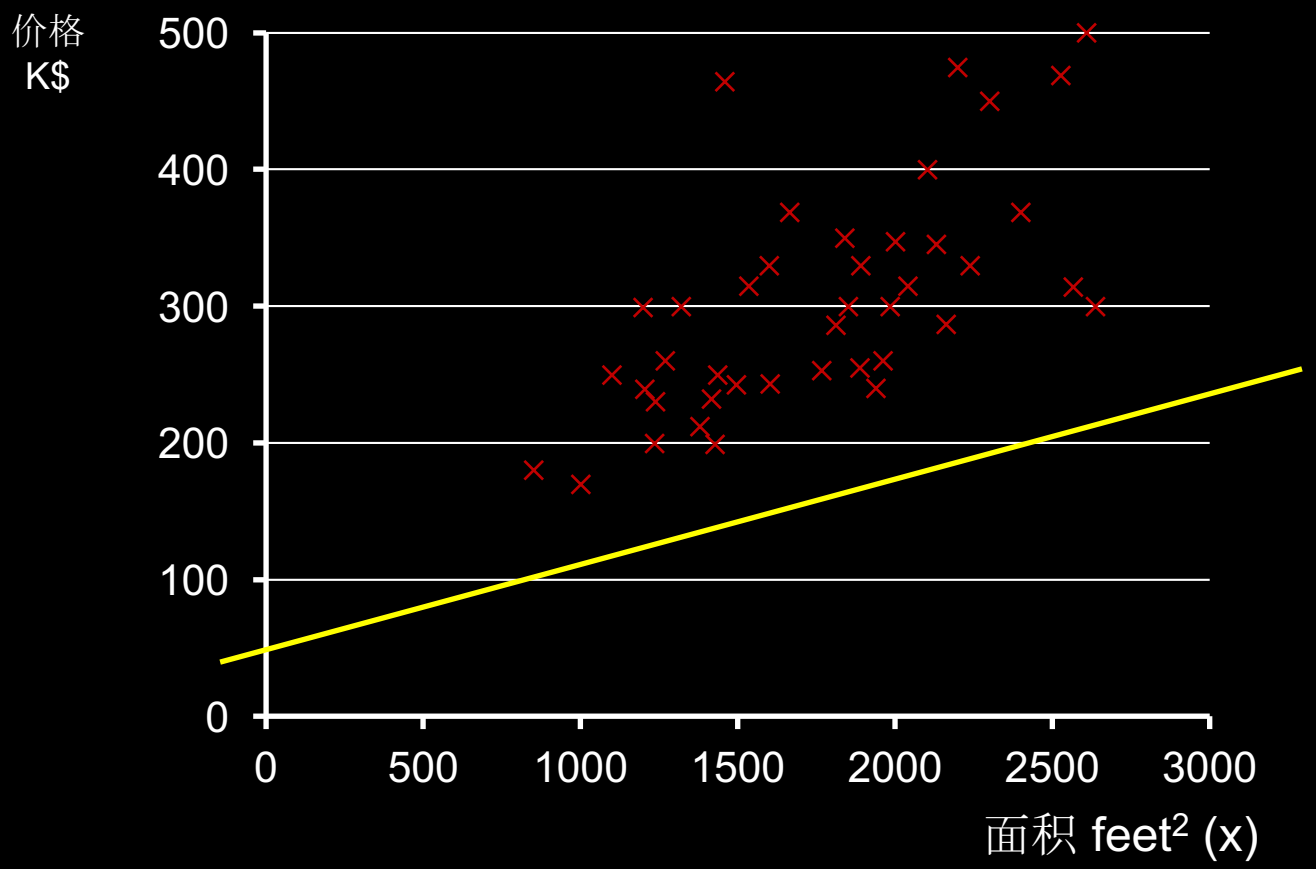
$$\nabla f(x, y) = \begin{bmatrix} 2x \sin y \\ x^2 cos\, y \end{bmatrix}$$

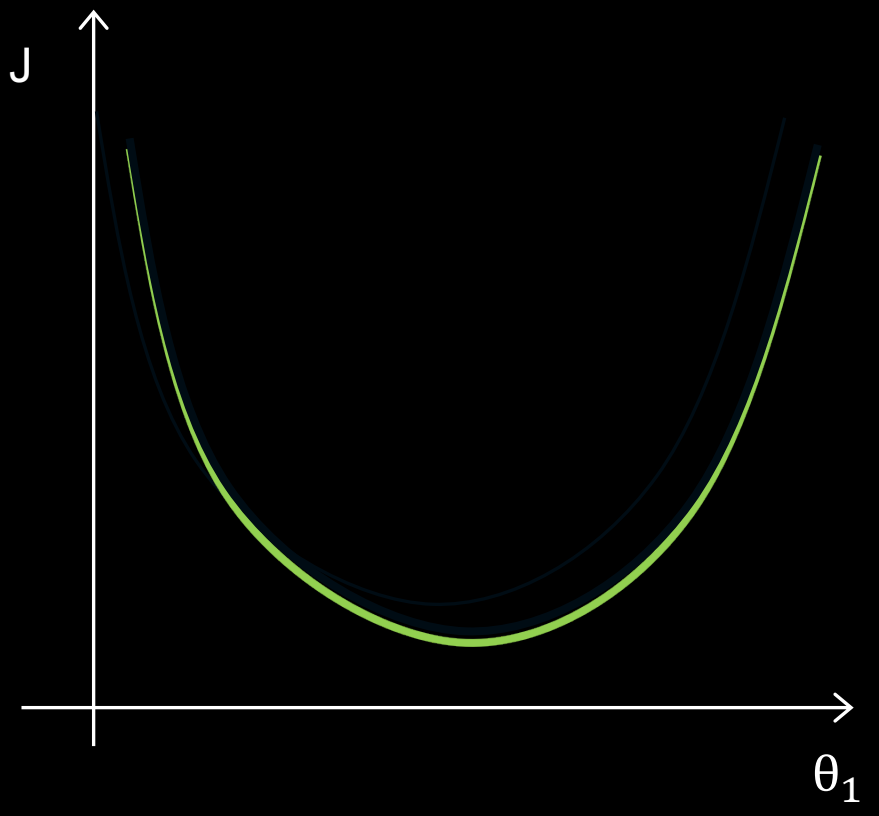
凸优化 Convex Optimization



代价函数

关于参数 θ_1 的代价函数





$$h_\theta(x) = 50 + 0.1x$$

代价函数

问题：对于某代价函数 $\mathcal{J}(\theta_0, \theta_1)$ ：

我们希望能 minimize $\mathcal{J}(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

步骤：

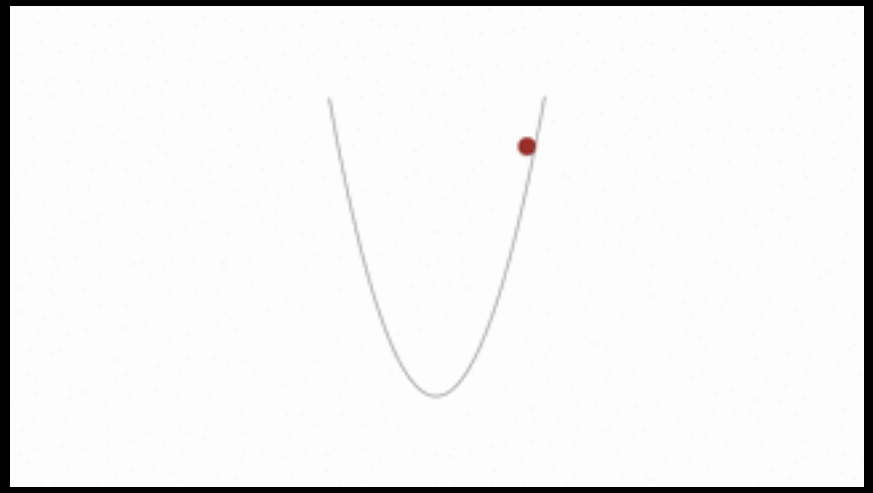
- 从某任意值的θ_0, θ_1开始
- 不断改变θ_0, θ_1以降低J (θ_0, θ_1)
- 直到最小值停止

调整参数

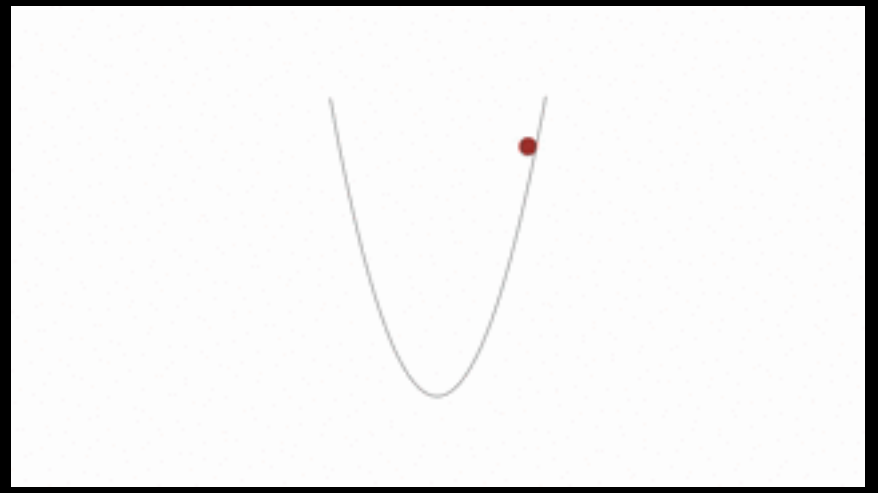
$$\theta_j := \theta_j - \text{learning\_rate} * \text{gradient\_of\_}\theta_j$$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} \mathcal{J}(\theta_0, \theta_1) \quad (j = 0,1) \qquad (3)$$

Learning Rate 学习率



理想情况收敛 Convergence



学习率过大导致发散 Divergence

- 经验值：0.1到0.0001之间，很多时候在0.001到0.0001之间；
- 当越逼近（局部）最优解时，梯度越小，参数更新的步幅也越小，因此，可能无需手动缩小步幅。

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} \mathcal{J}(\theta_0, \theta_1) \quad (j = 0,1)$$

LMS算法 （Widrow-Hoff Learning Rule)

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

先计算$\frac{\partial}{\partial \theta_j} \mathcal{J}(\theta_0, \theta_1)$，我们先计算只有一个数据样本的情况，即m=1的情况:

$$\frac{\partial}{\partial \theta_j}\mathcal{J} = \frac{\partial}{\partial \theta_j}\left(\frac{1}{2}(h_\theta(x) - y)^2\right)$$

$$= 2 \cdot \frac{1}{2} \cdot (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j}(h_\theta(x) - y)$$

$$= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j}\left(\sum_{i=0}^{n} \theta_i x_i - y\right)$$

$$= (h_\theta(x) - y) \cdot x_j$$

LMS算法 （Widrow-Hoff Learning Rule)

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

$$\theta_j := \theta_j + (y - h_\theta(x)) \cdot x_j$$

显然：参数更新的幅度（$Magnitude$）与误差$(y - h_\theta(x))$呈正比。

LMS算法 （Widrow-Hoff Learning Rule）

对于 m个训练样本：

循环至收敛{

$$\theta_j := \theta_j + \alpha \cdot \frac{1}{m} \cdot \sum_{i=1}^{m} (y^{(i)} - h_\theta\left(x^{(i)}\right)) \cdot x_j^{(i)}$$

}

显然：

- 参数更新的幅度（$Magnitude$）与误差$(y - h_\theta(x))$呈正比；
- 每次循环中，都遍历所有的训练样本，因此被称为Batch Gradient Descent

LMS算法 （Widrow-Hoff Learning Rule)

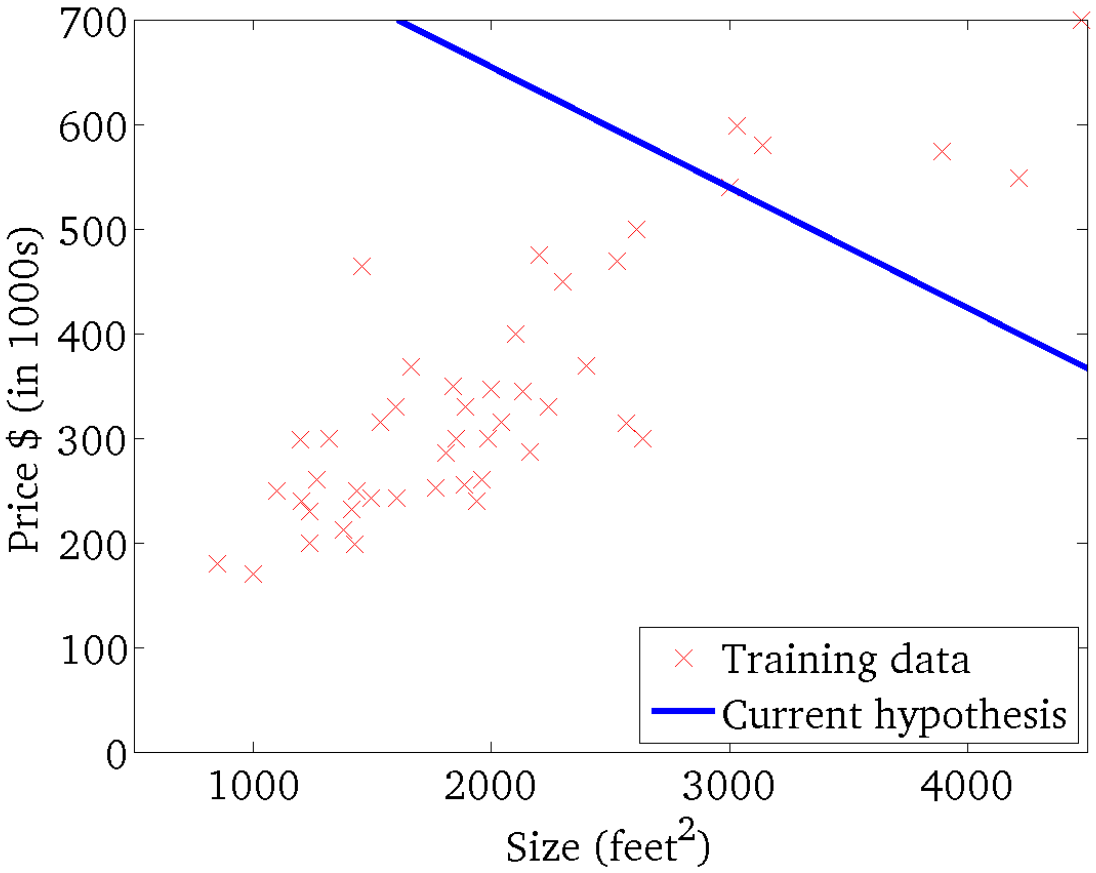
$$\theta_0 := \theta_0 + \alpha \cdot \frac{1}{m} \cdot \sum_{i=1}^{m} (y^{(i)} - h_\theta\left(x^{(i)}\right))$$

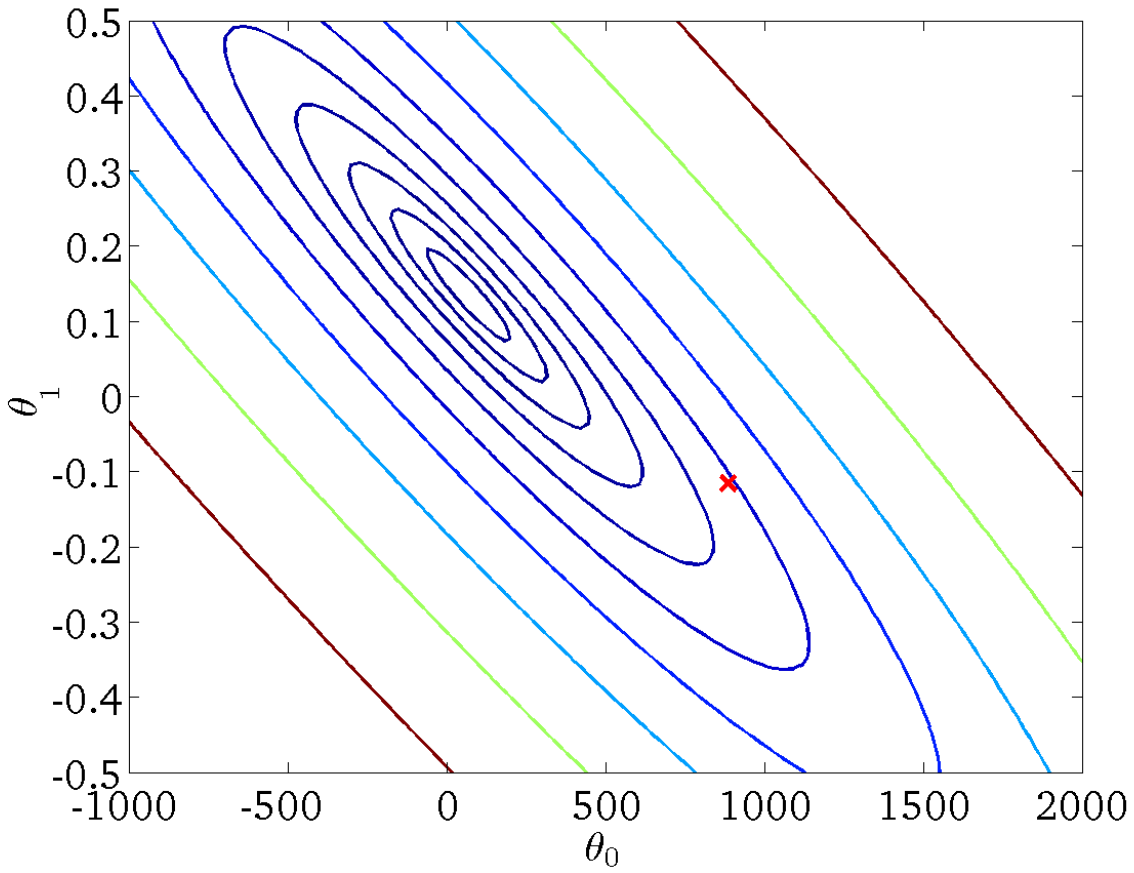
$$\theta_1 := \theta_1 + \alpha \cdot \frac{1}{m} \cdot \sum_{i=1}^{m} (y^{(i)} - h_\theta\left(x^{(i)}\right)) \cdot x^{(i)}$$

显然：参数更新的幅度（$Magnitude$）与误差$(y - h_\theta(x))$呈正比。

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

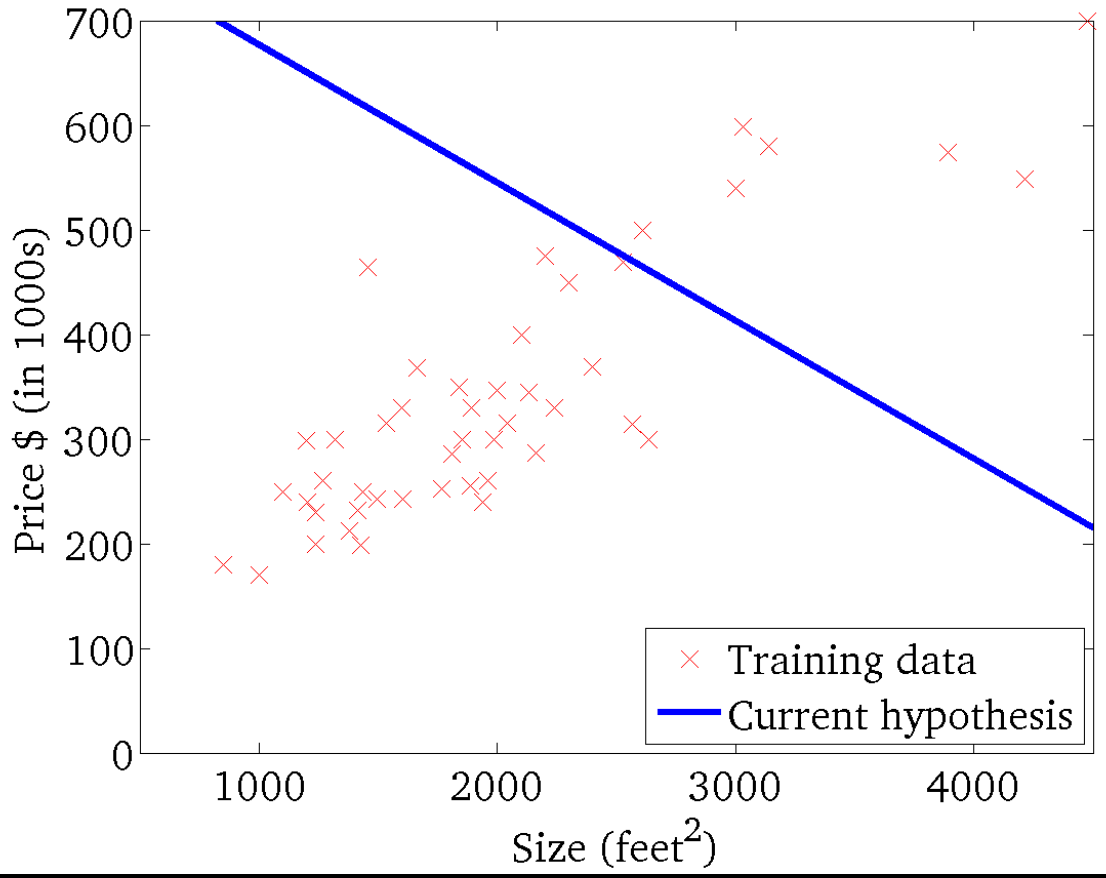


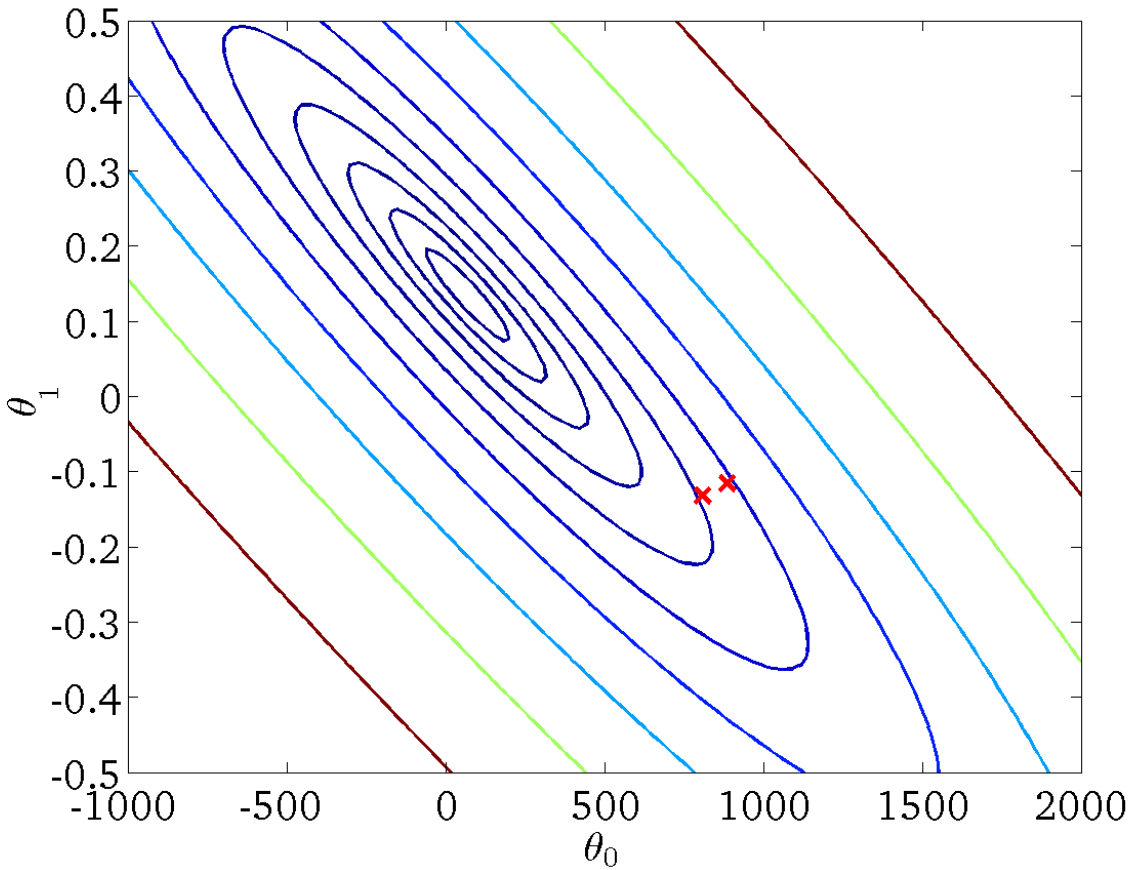


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

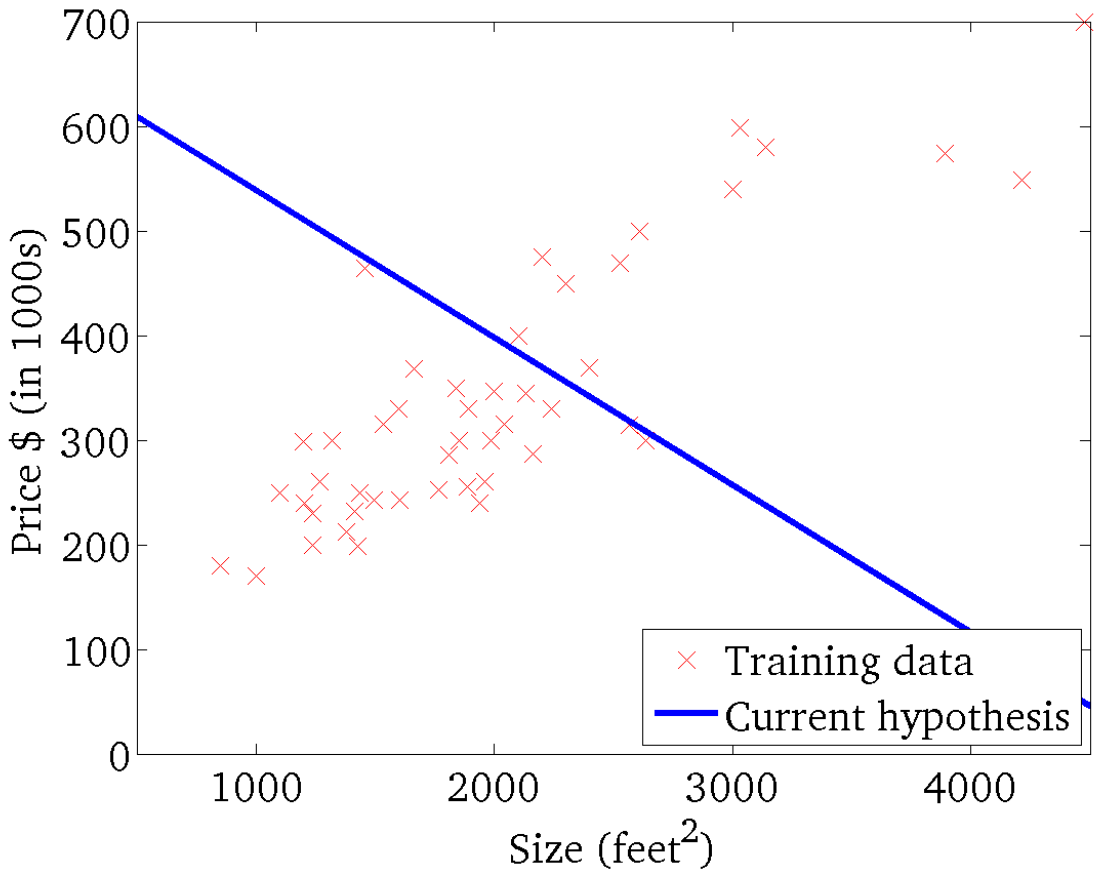


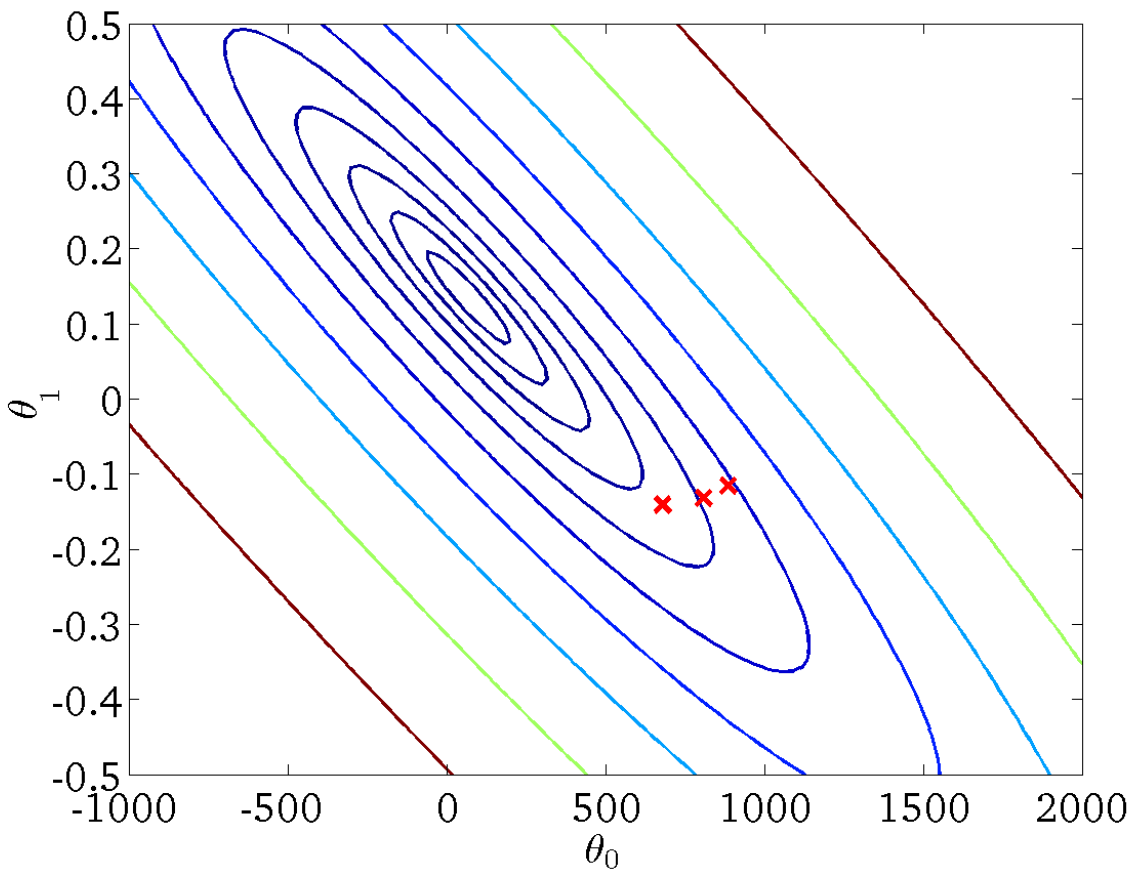


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

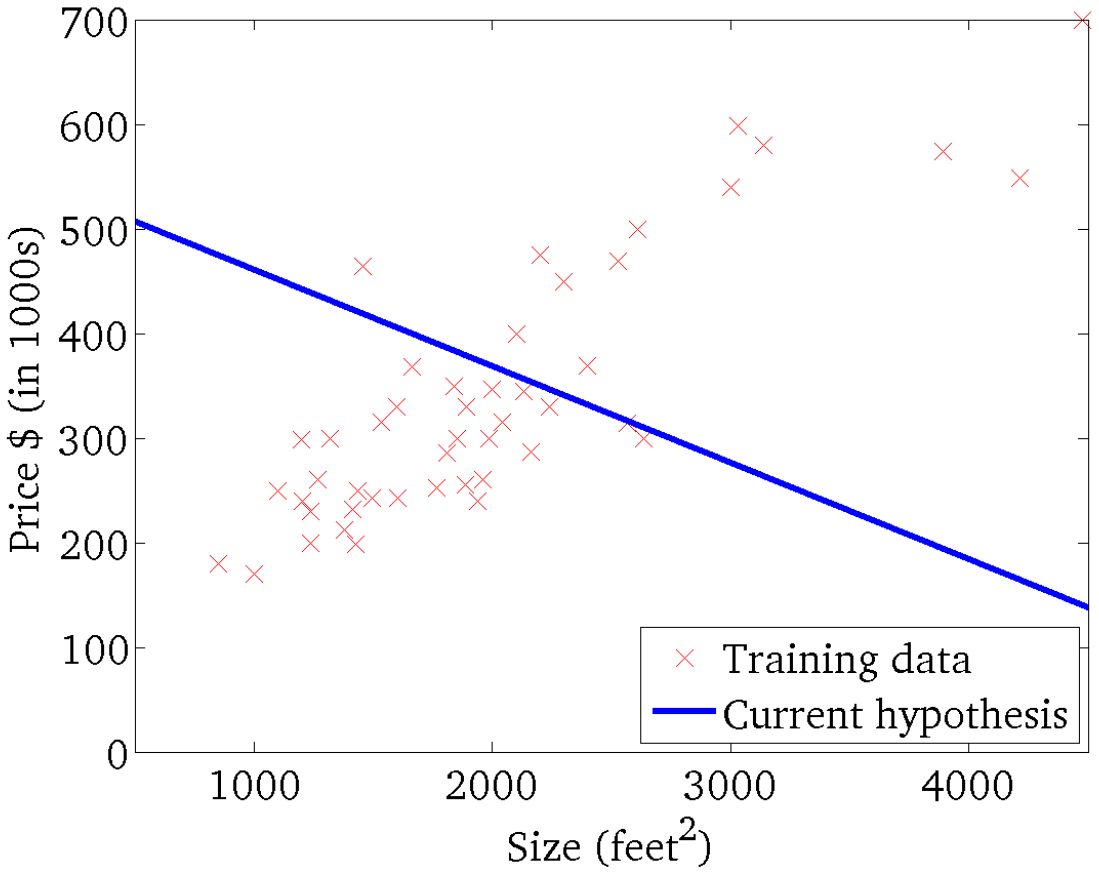


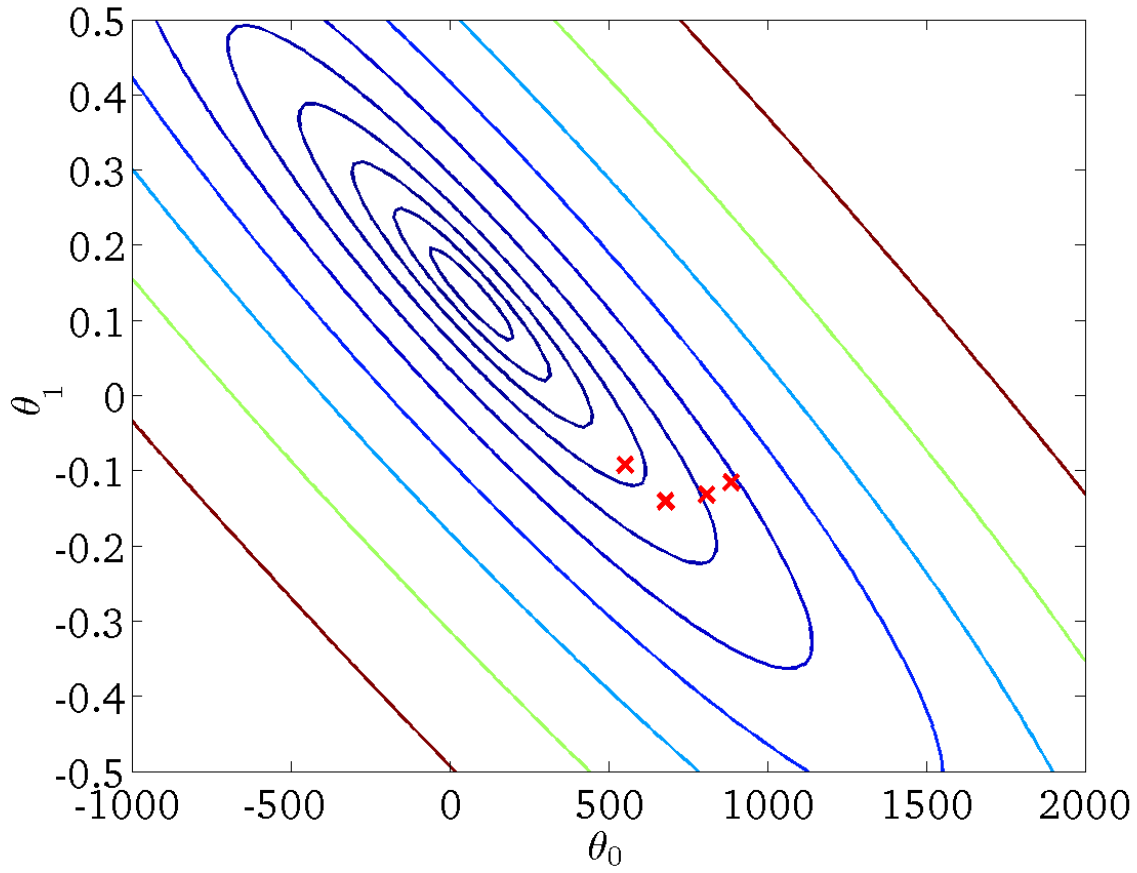


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

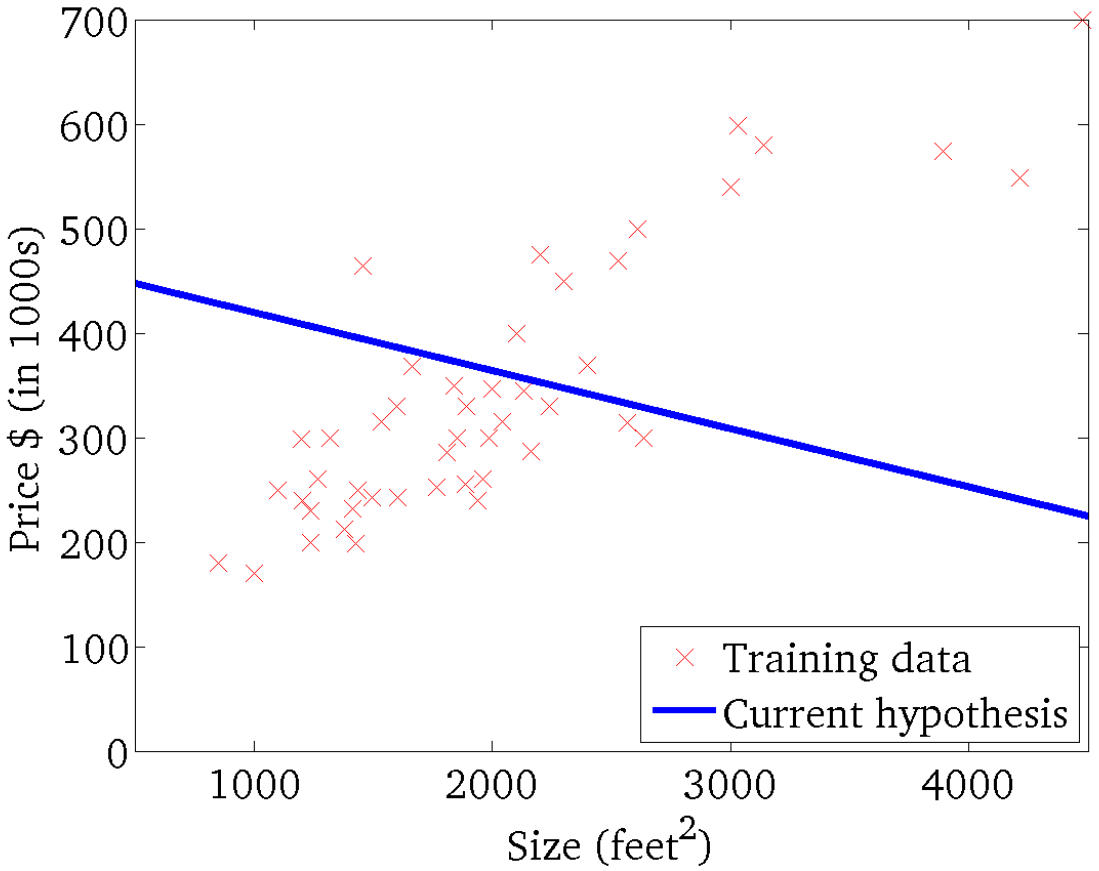


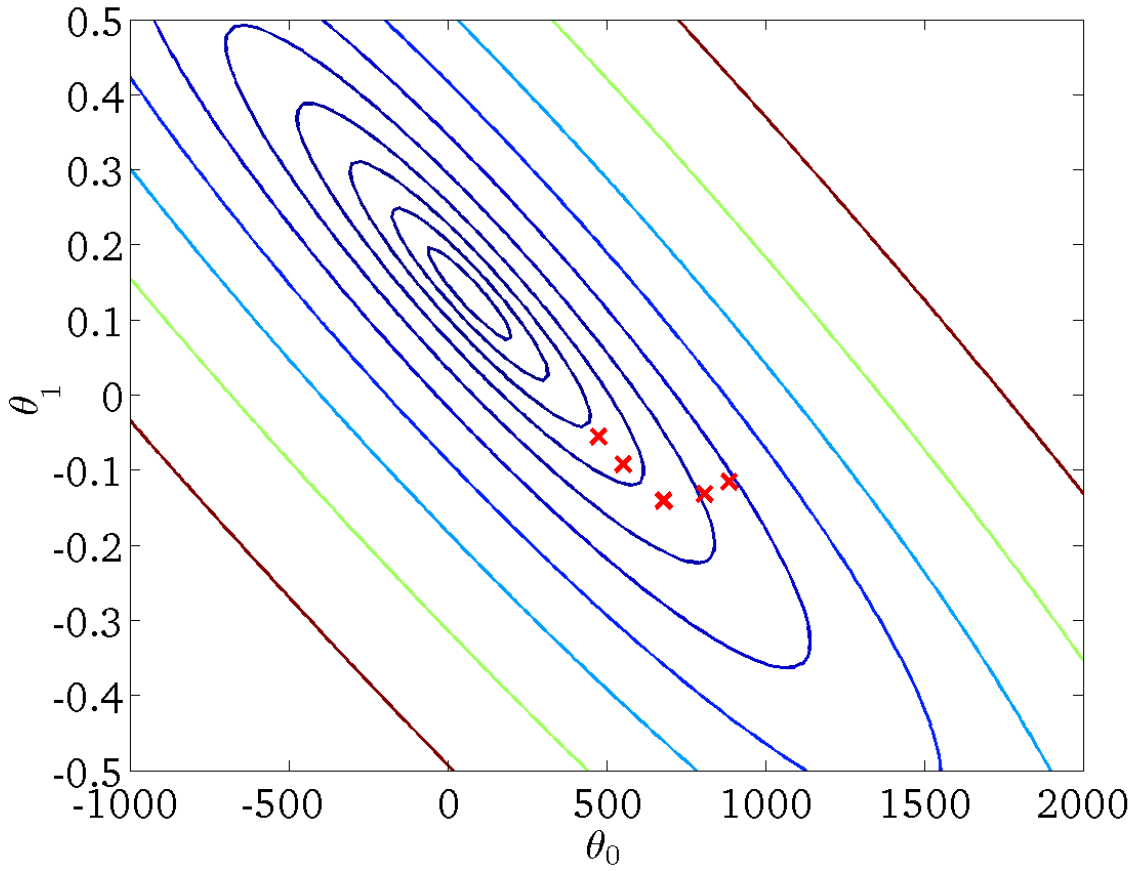


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

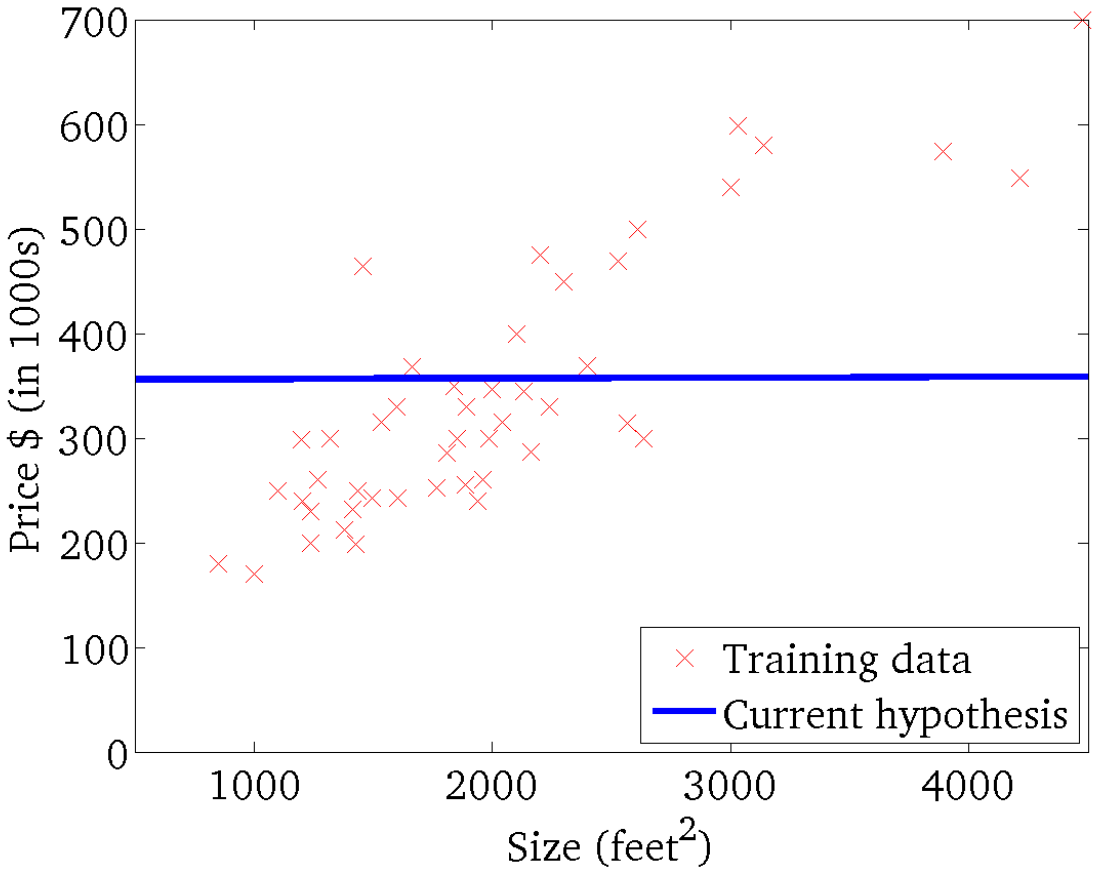


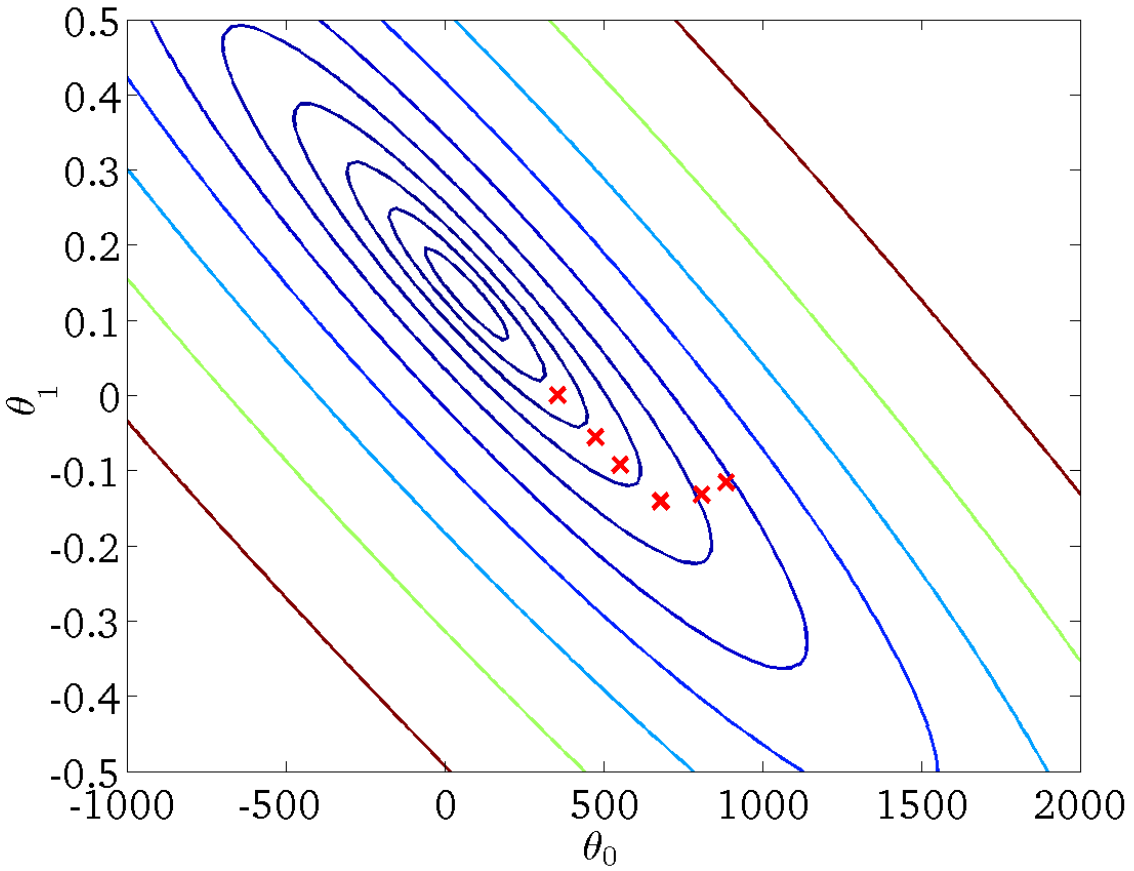


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

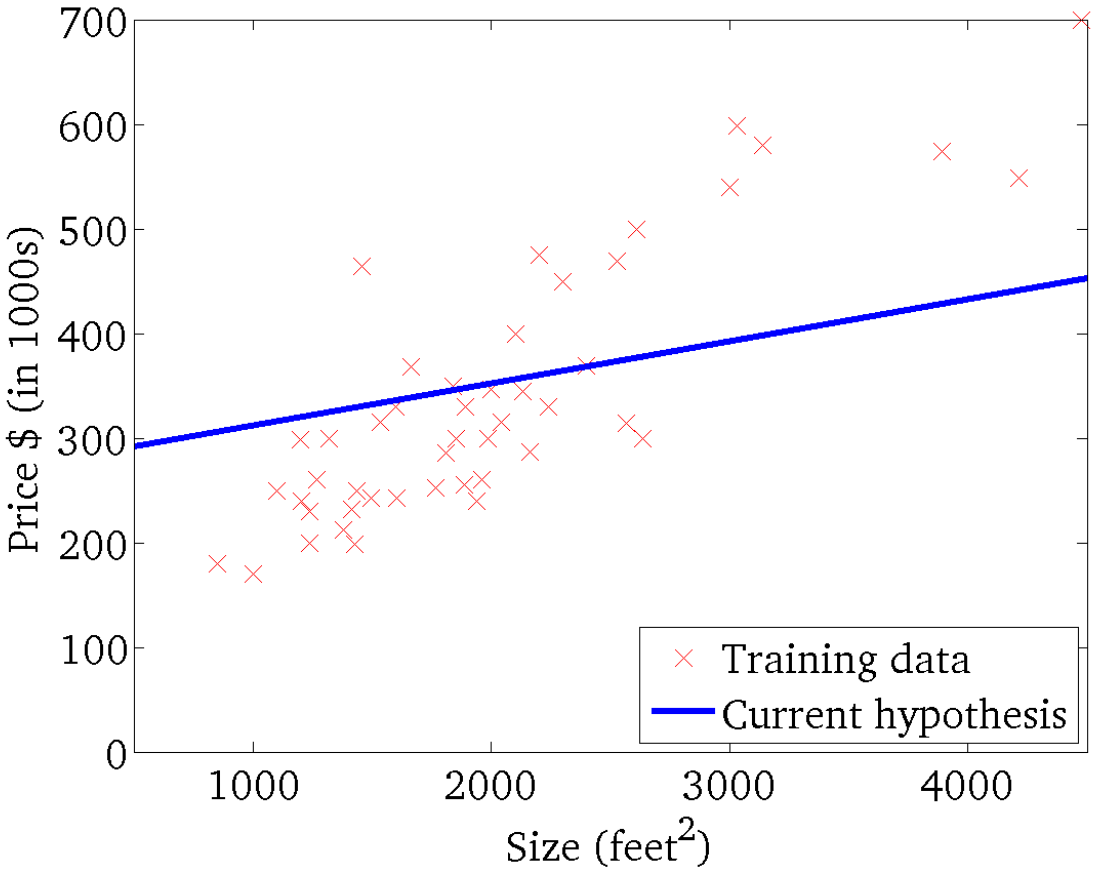


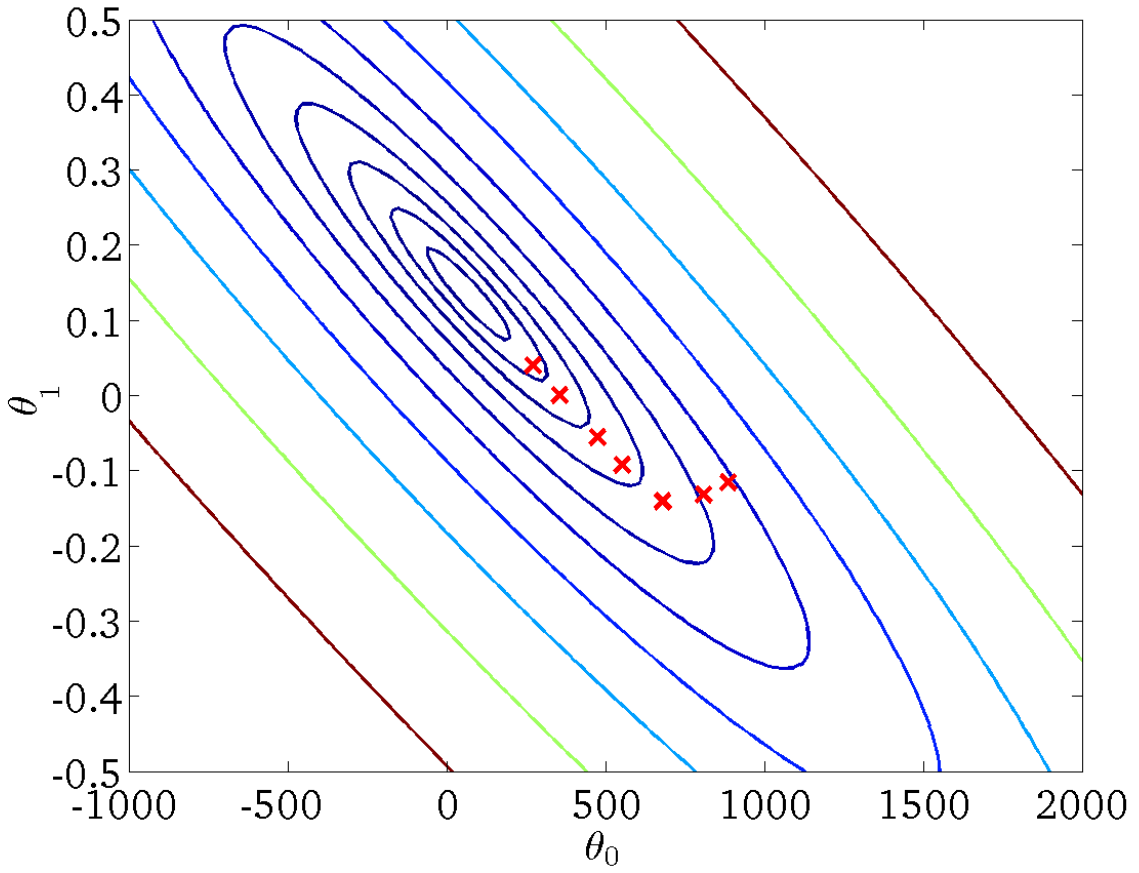


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$

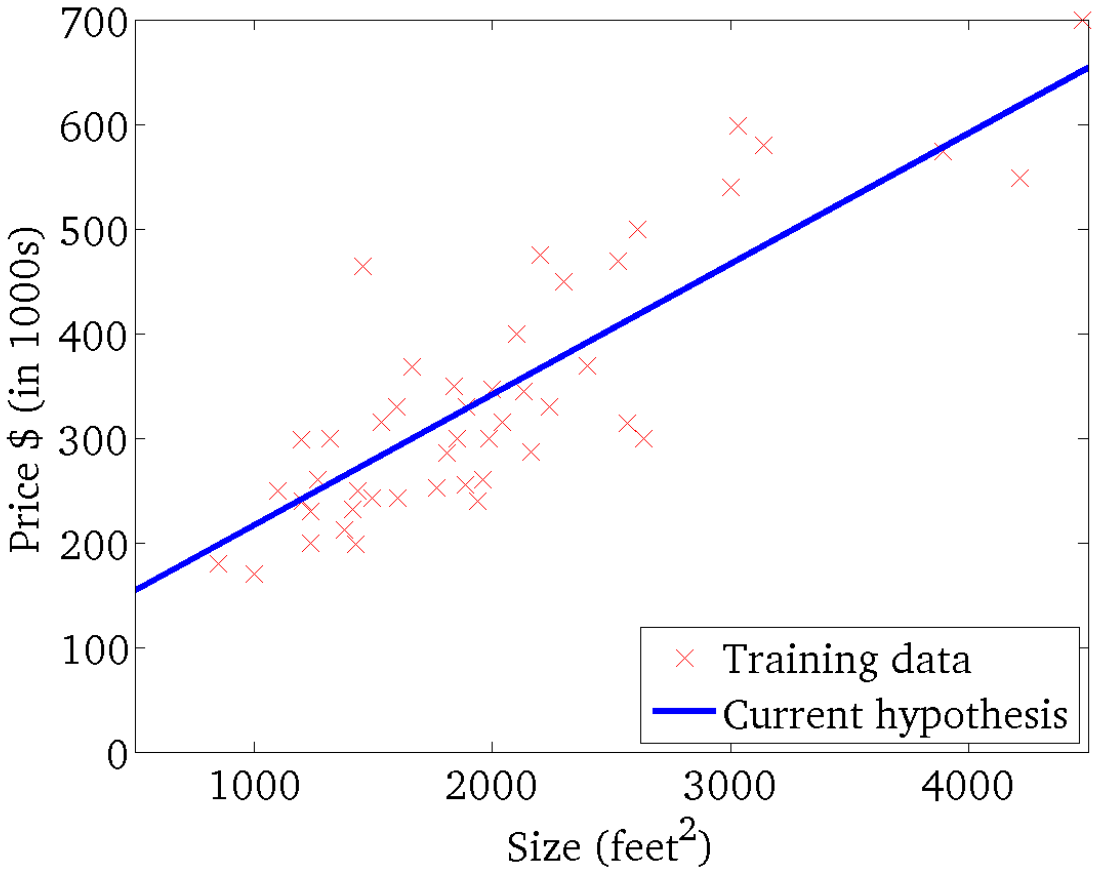


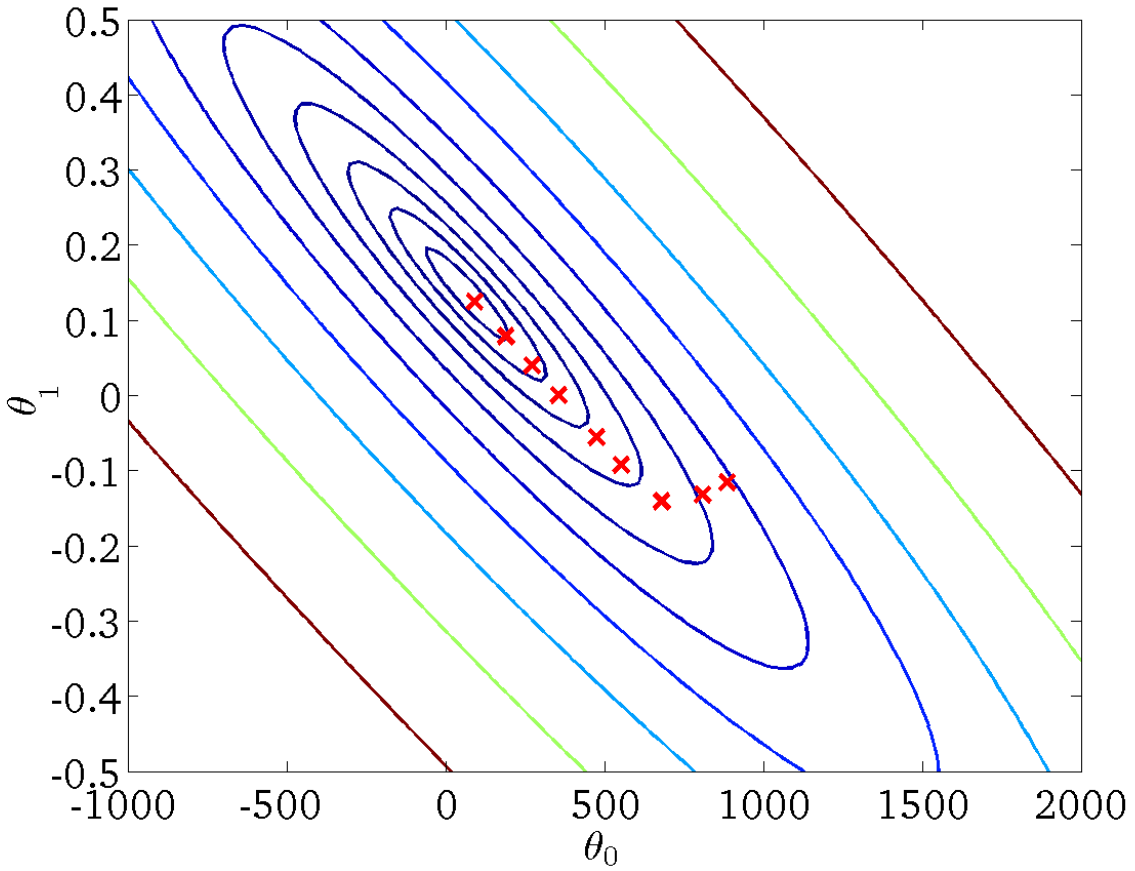


Credit: Andrew Ng

线性回归 / 梯度下降

$J(\theta_0, \theta_1)$





Credit: Andrew Ng

随机梯度下降 Stochastic Gradient Descent

1. 对训练样本的数据进行随机重排序
2. 利用单个训练样本就可以进行权重更新

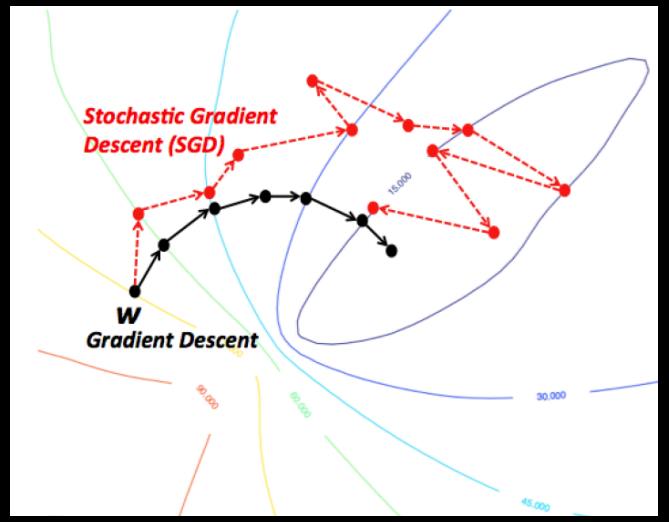
Loop {

for i=1 to m, {

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}

}



Source: https://wikidocs.net/3413

思想：以随机梯度替代真正的梯度。在足够多步后，真正梯度的均值≈随机梯度均值。

优点：简单，计算量小，对于大数据量有用

缺点：稳定性欠缺一些

Mini-Batch Gradient Descent

$J(\theta_0, \theta_1)$

Say $b = 10, m = 1000$.

Repeat {

  for $i = 1, 11, 21, 31, \ldots, 991$ {

$$\theta_j := \theta_j - \alpha \frac{1}{10} \sum_{k=i}^{i+9} (h_\theta(x^{(k)}) - y^{(k)}) x_j^{(k)}$$

(for every $j = 0, \ldots, n$)

  }

}

Credit: Andrew Ng

Multivariate Linear Regression

Model Representation

$$H_\theta(x) = \theta_0 + \theta_1 x_1 + \bullet\bullet\bullet + \theta_n x_n$$
$$= \sum_{i=0}^{n} \theta_i x_i \;=\; \theta^T \bullet X \qquad (x_0 = 1)$$

Multivariate Linear Regression

Multiple features

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

Credit: Andrew Ng

Notation:

- n = number of features
- $x^{(i)}$= input (features) of the i^{th} training example.
- $x_j^{(i)}$= value of feature j in i^{th} training example.

Multivariate Linear Regression

Multiple features（多变量线性回归）

Model:

$$h_\theta(\mathrm{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \qquad (x_0 = 1)$$
$$= \theta^T x$$

Parameters: $\theta_0 \ , \theta_1 \ , \cdots \ , \theta_n$

Cost Function: $\mathcal{J}(\theta_0, \theta_1, \cdots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

Gradient Descent: (♻迭代至收敛)

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

Multivariate Linear Regression

Gradient Descent（梯度下降）

♻迭代至收敛，每 Iteration / epoch 内使用矢量表达避免循环

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

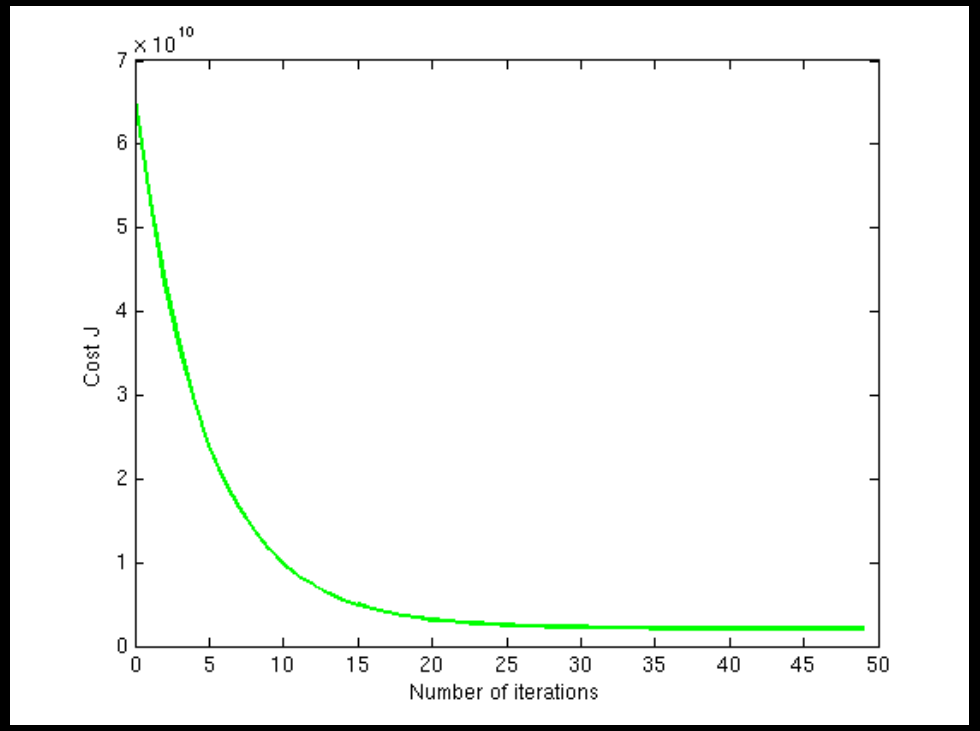
$$= \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

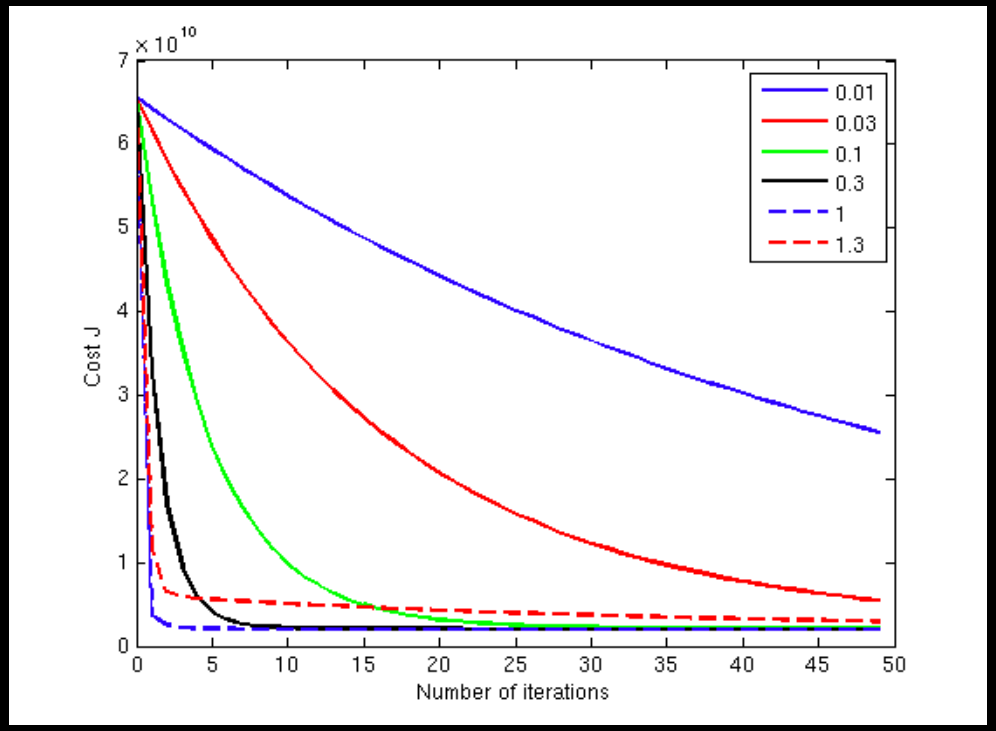
- 程序运行期间，需监控代价函数，确保其不但不断下降，而且还不至于太慢
- 合理选择α学习率
- Declare $\mathcal{J}(\boldsymbol{\theta})$ is convergence if decreases by less than 10^{-3} in one iteration

SAP

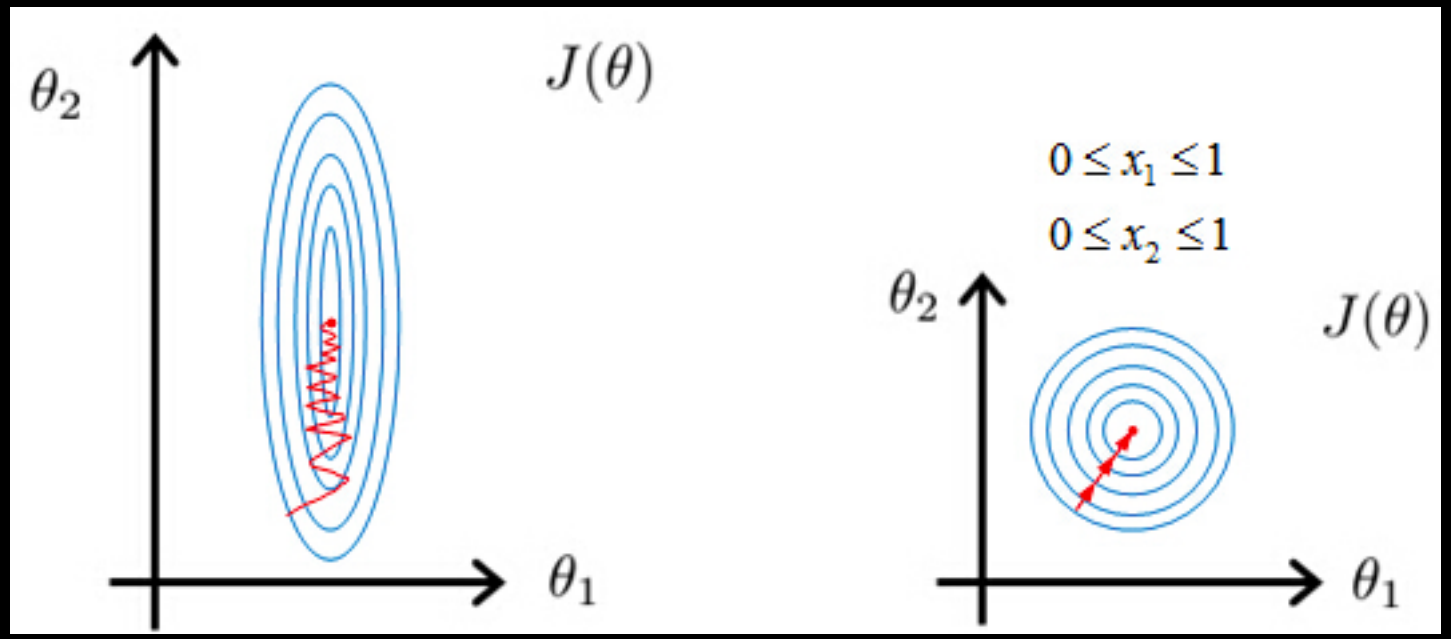
FUDAN UNIVERSITY 1905 復旦大學

Learning Rate & Cost





Feature Scaling



公式：$x' := \dfrac{x - x_{min}}{x_{max} - x_{min}}$ 或 $x' := \dfrac{x - \bar{x}}{\sigma(x)}$

Thank you!

Contact information:

邬学宁（i025497)
Data Scientist, SAP Silicon Valley Innovation Center
Address: No. 1001, Chenghui Road, Shanghai, 201023
Phone number: +8621-6108 5287
Email: x.wu@sap.com