Machine Learning - Week 1

赵燕

目录

1 Introduction 引言

1.1 Welcome

第一个视频主要讲了什么是机器学习，机器学习能做什么。

- Grew out of work in AI
- New capability for computers

机器学习案例：

- Database mining 数据库挖掘
  Large datasets from growth of automation/web.
  E.g.,Web click data, medical records, biology, engineering
- Applications can't program by hand.
  E.g.,Autonomous helicopter,(直升机自主飞行程序) handwriting recognition, most of Natural Language Processing(NLP), Computer Vision
- Self-customizing programs
  E.g., Amazon, Netflix product recommendations

1.2 What is Machine Learning?

第一个机器学习的定义来自于 Arthur Samuel。他定义机器学习为，在不进行特定编程的情况下，给予计算机学习能力的领域。

Arthur Samuel described it as: "the field of study that gives computers the ability to learn without being explicitly programmed." This is an older, informal definition.

另一个年代比较近的定义，Tom Mitchell定义的机器学习是，一个好的学习问题，定义如下，他说，一个程序被认为能从经验E中学习，解决任务T，达到性能度量值 P，当且仅当，有了经验E后，经过P评判，程序在处理T时的性能有所提升。

Tom Mitchell provides a more modern definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Example: playing checkers.
E = the experience of playing many games of checkers
P = the probability that the program will win the next game.
In general, any machine learning problem can be assigned to one of two broad classifications: Supervised learning and Unsupervised learning.

目前存在几种不同类型的学习算法。主要的两种类型被我们称之为监督学习（supervised learning）和无监督学习（unsupervised learning）。

其他的有，强化学习（Reinforcement learning），推荐系统（recommender systems）。也会提到应用学习算法的实用建议（Practical advice for applying learning algorithms）。

1.3 Supervised Learning

Supervised Learning "right answers"given.监督学习指的就是我们给学习算法一个数据集，这个数据集由"正确答案"组成。

例1：售楼的案例的数据是实现准确得到的，一个学生从波特兰俄勒冈州的研究所收集了一些房价的数据。你把这些数据画出来,看起来是这个样子:横轴表示房子的面积,单位是平方英尺,纵轴表示房价,单位是千美元。那基于这组数据,假如你有一个朋友,他有一套 750 平方英尺房子,现在他希望把房子卖掉,他想知道这房子能卖多少钱。

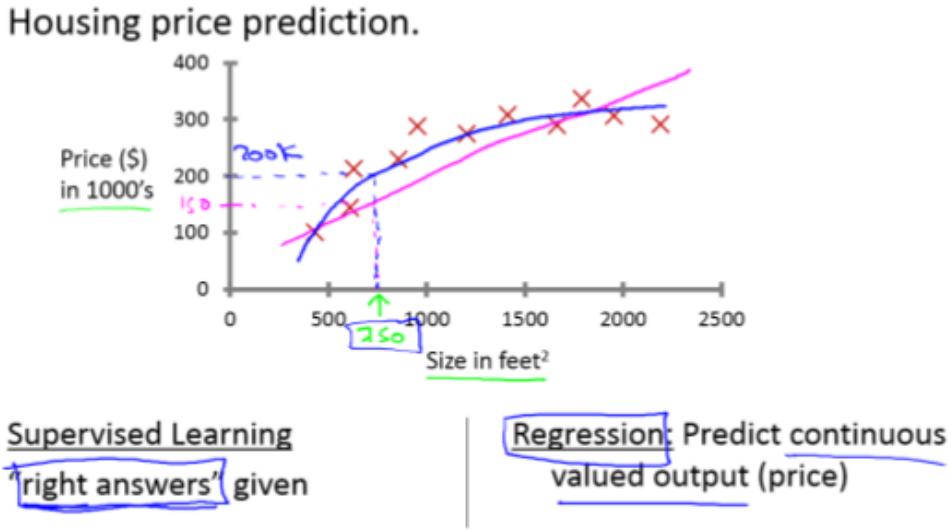

图 1: 房价问题（描述）

这个例子是回归问题。回归(Regression)这个词的意思是，我们在试着推测出这一系列连续值属性(Predict continuous valued outputR(price))。

例2：肿瘤样本的例子属于分类问题，其目标是推出一组离散结果,实际分类问题中输出会不止两个值,预测肿瘤的恶性与否用到的特征，在机器学习问题中，可能会遇到不止一种特征。

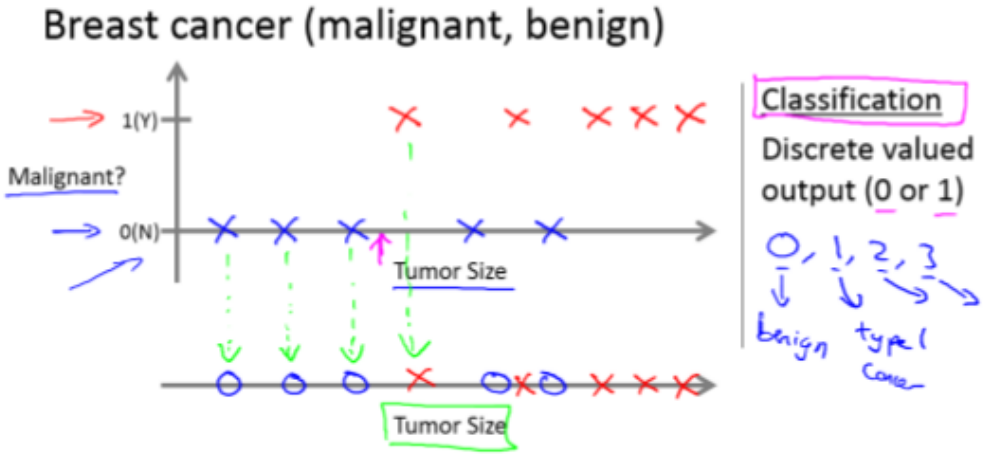

图 2: 肿瘤问题（描述）

现在我用不同的符号来表示这些数据。既然我们把肿瘤的尺寸看做区分恶性或良性的特征,那么我可以这么画,我用不同的符号来表示良性和恶性肿瘤。或者说是负样本和正样本 现在我们不全部画 X,良性的肿瘤改成用 O 表示,恶性的继续用 X 表示。来预测肿瘤的恶性与否。

如果想用无限多种特征好让你的算法可以利用大量的特征， 或者说线索来做推测。如何处理怎么存出这些特征都存在问题，电脑内存会不够。用支持向量机来解决，可以让计算计处理无限多个特征。

在其它一些机器学习问题中,可能会遇到不止一种特征。举个例子,我们不仅知道肿瘤的尺寸,还知道对应患者的年龄。在其他机器学习问题中,我们通常有更多的特征,我朋友研究这个问题时,通常采用这些特征,比如肿块密度,肿瘤细胞尺寸的一致性和形状的一致性等等,还有一些其他的特征。这就是我们即将学到最有趣的学习算法之一。那种算法不仅能处理 2 种 3 种或 5 种特征,即使有无限多种特征都可以处理。

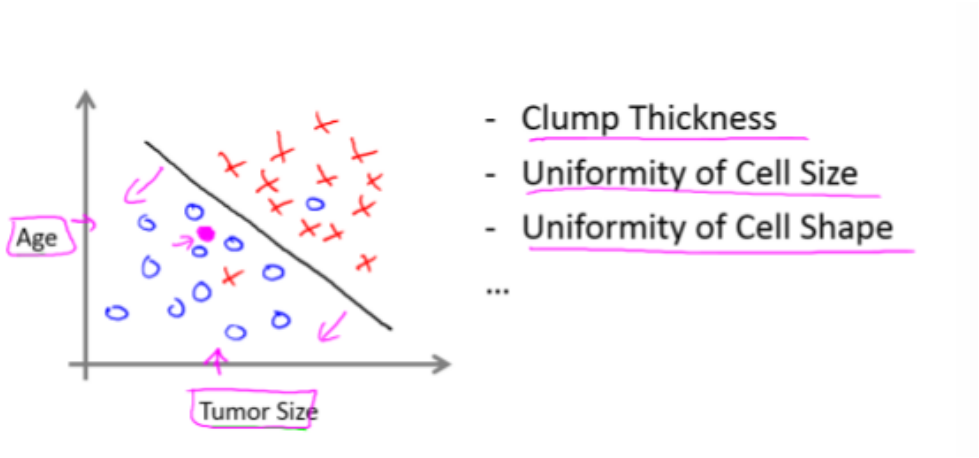

图 3: 机器学习的特征

我列举了总共 5 种不同的特征,坐标轴上的两种和右边的 3 种,但是在一些学习问题中,你希望不只用 3 种或 5 种特征。相反,你想用无限多种特征,好让你的算法可以 利用大量的特征,或者说线索来做推测。那你怎么处理无限多个特征,甚至怎么存储这些特征都存在问题,你电脑的内存肯定不够用。我们以后会讲一个算法,叫支持向量机,里面有一个巧妙的数学技巧,能让计算机处理无限多个特征。

监督学习。其基本思想是,我们数据集中的每个样本都有相应的“正确答案”。再根据这些样本作出预测,就像房子和肿瘤的例子中做的那样。我们还介绍了回归问题,即通过回归来推出一个连续的输出,之后我们介绍了分类问题,其目标是推出一组离散的结果。

1.4 Unsupervised Learning

无监督学习中的数据没有任何标签或者是有相同的标签或者就是没标签，没有基于预测结果的反馈。但无监督学习算法可能会把这些数据分成不同的簇，叫做聚类算法(Clustering)。

Unsupervised learning allows us to approach problems with little or no idea what our results should look like. We can derive structure from data where we don't necessarily know the effect of the variables.

With unsupervised learning there is no feedback based on the prediction results.

无监督学习或者聚类算法在其他领域有着大量应用：组织大型的计算机群(Organize computing clusters)、社交网络分析(Social network analysis)、 市场分割中的应用(Market segmentation)、天文数据分析(Astronomical data analysis)。这些都是聚类的例子，聚类只是无监督学习的一种。

例1：基因芯片问题，找到一种方法，将这些基因自动组合成类似或相关的不同变量，如寿命、位置、角色等。

Clustering: Take a collection of 1,000,000 different genes, and find a way to automatically group these genes into groups that are somehow similar or related by different variables, such as lifespan, location, roles, and so on.

例2：非聚类问题，鸡尾酒宴问题，允许你在混乱的环境中找到结构。，两个人同时说话，每个麦克风都会录到两个人重叠的声音，就是从重叠的音频中分离音频。

Non-clustering: The "Cocktail Party Algorithm", allows you to find structure in a chaotic environment. (i.e. identifying individual voices and music from a mesh of sounds at a cocktail party).

代码：[W, s, v] = svd ((repmat(sum(x.∗x,1),size(x,1),1).∗x)∗x');

使用Octave或者Matlab这类的工具实现这些算法

2 Liner Regerssion with One Variable 单变量线性回归

2.1 Model Representation

通过线性回归算法了解监督学习的例子

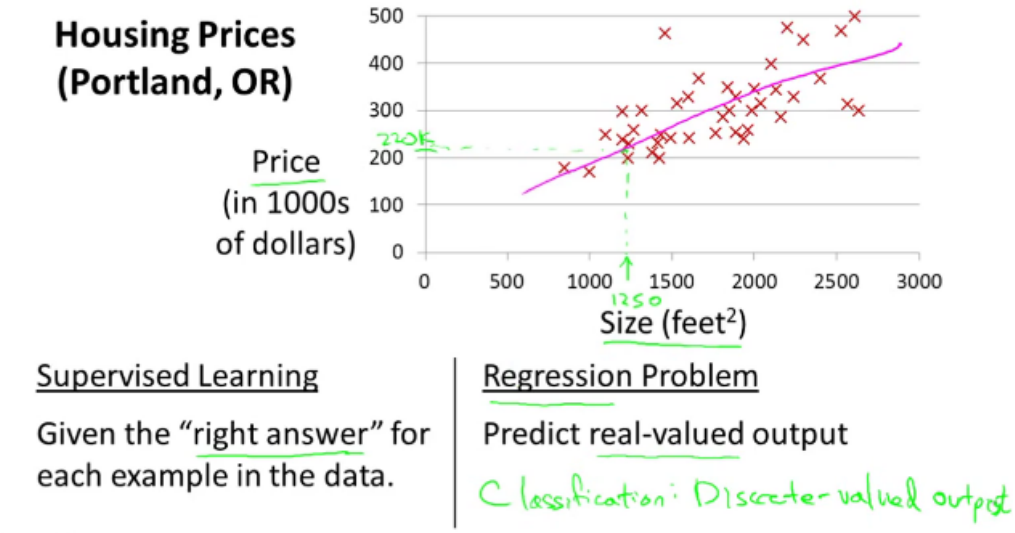

图 4: 房价模型表示

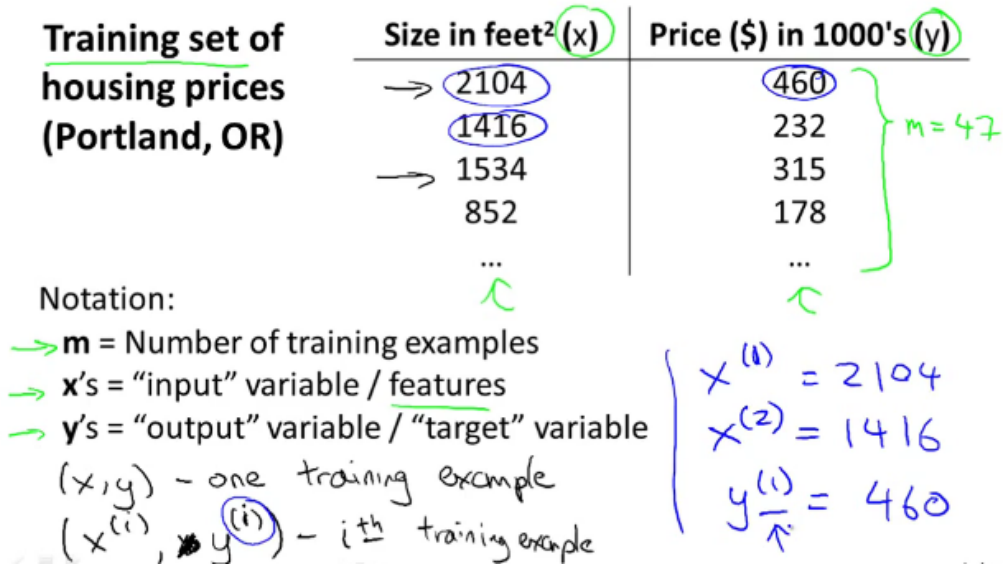

图 5: 监督学习

假使我们回归问题的数量集(Training Set)如图所示:

Size in feet^2 (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

图 6: 数量集 Training Set

我们将要用来描述这个回归问题的标记如下:
m 代表训练集中实例的数量

x 代表特征/输入变量

y 代表目标变量/输出变量

(x,y) 代表训练集中的实例

(x (i) ,y (i)) 代表第 i 个观察实例

h 代表学习算法的解决方案或函数也称为假设(hypothesis)

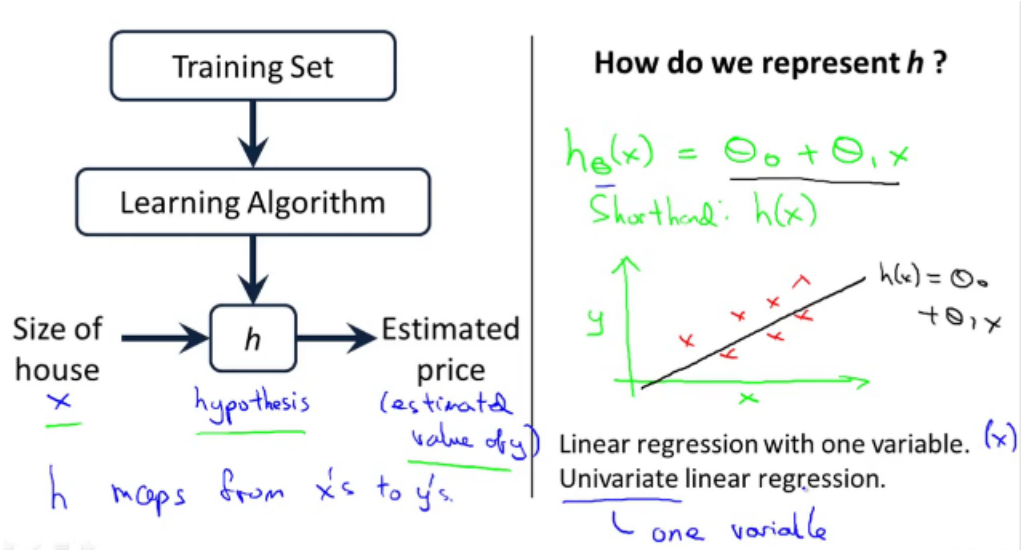

图 7: 假设Hypothesis

如何表达h?

Training Set

Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

m = 47

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

θ_i's: Parameters

How to choose θ_i's ?

图 8: 如何表达h

因为只含有一个特征/输入变量,因此这样的问题叫作单变量线性回归问题

To describe the supervised learning problem slightly more formally, our goal is, given a training set, to learn a function h : X → Y so that h(x) is a "good" predictor for the corresponding value of y. For historical reasons, this function h is called a hypothesis. Seen pictorially, the process is therefore like this:

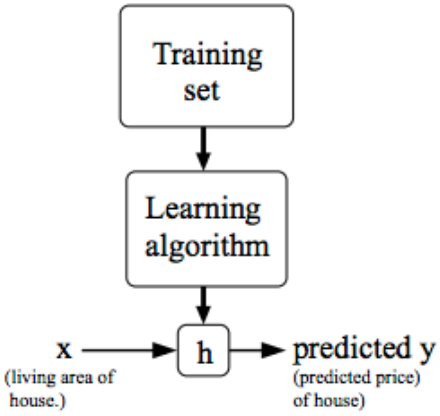

图 9: 假设函数

When the target variable that we're trying to predict is continuous, such as in our housing example, we call the learning problem a regression problem. When y can take on only a small number of discrete values (such as if, given the living area, we wanted to predict if a dwelling is a house or an apartment, say), we call it a classification problem.

2.2 Cost Function

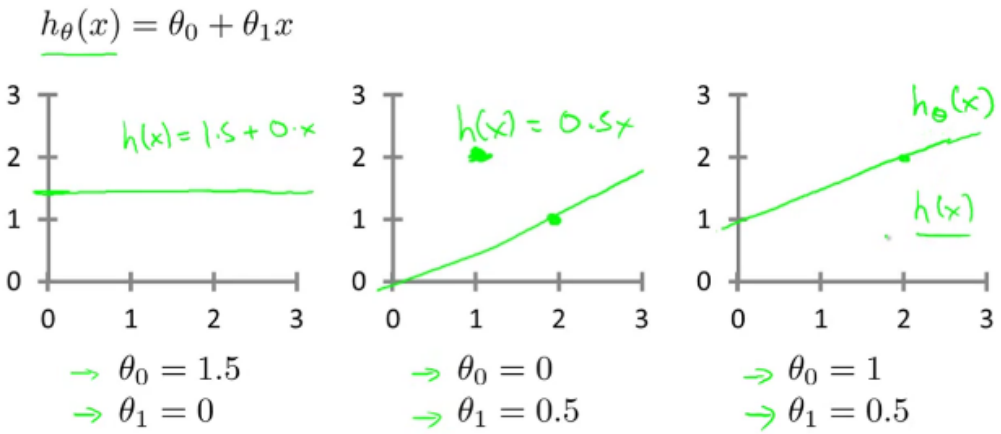

图 10: 代价函数 CostFunction

平方差代价函数是解决回归问题最常用的手段

We can measure the accuracy of our hypothesis function by using a cost function. This takes an average difference (actually a fancier version of an average) of all the results of the hypothesis with inputs from x's and the actual output y's.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{k=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{k=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

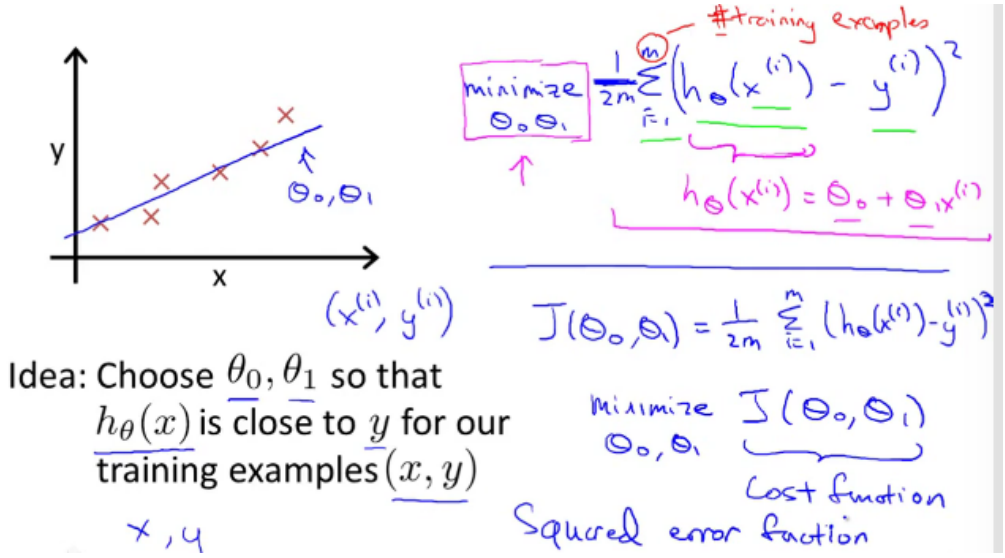

图 11: 代价函数

To break it apart, it is 1/2 $\bar{x}$ where $\bar{x}$ is the mean of the squares of h θ (xi)yi , or the difference between the predicted value and the actual value.

This function is otherwise called the "Squared error function", or "Mean squared error". The mean is halved 1/2 as a convenience for the computation of the gradient descent, as the derivative term of the square function will cancel out the 1/2 term. The following image summarizes what the cost function does:

2.2.1 Cost Function-Intuition I

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

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

Goal:

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

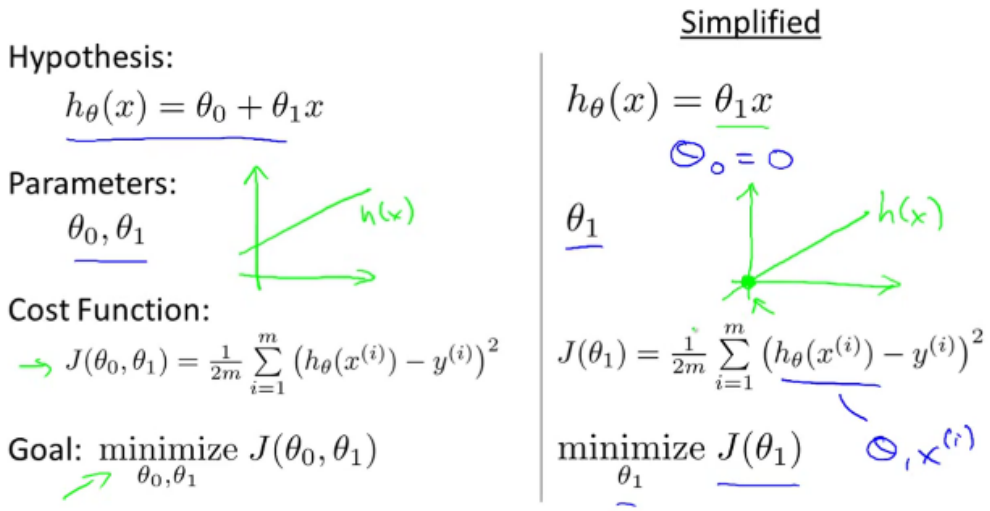

图 12: 假设函数的简化

代价函数我们要理解两个重要的函数：一个是假设函数，一个是代价函数
当$\theta 1$等于1时：

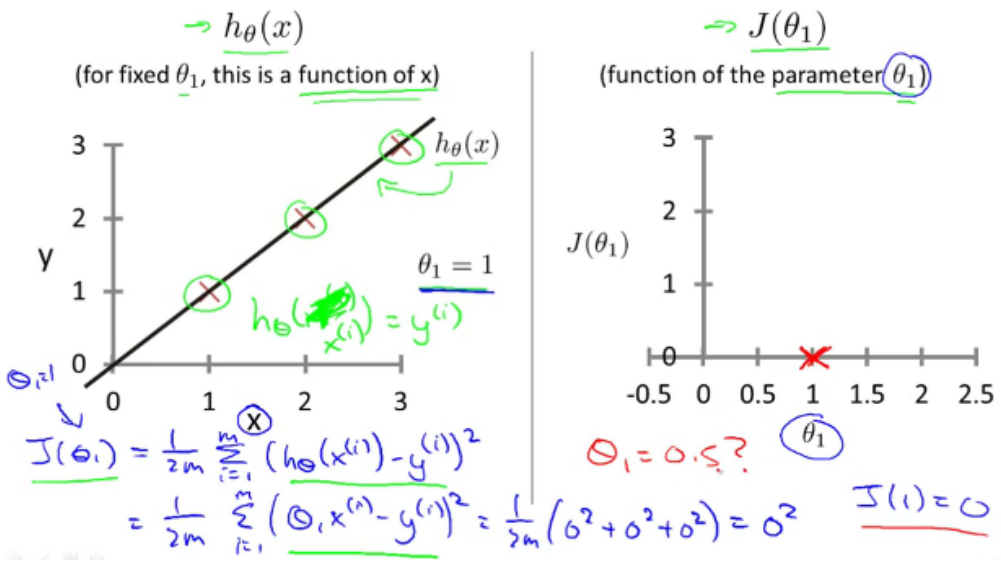

图 13: $\theta 1$等于1

当$\theta 1$等于0.5时：

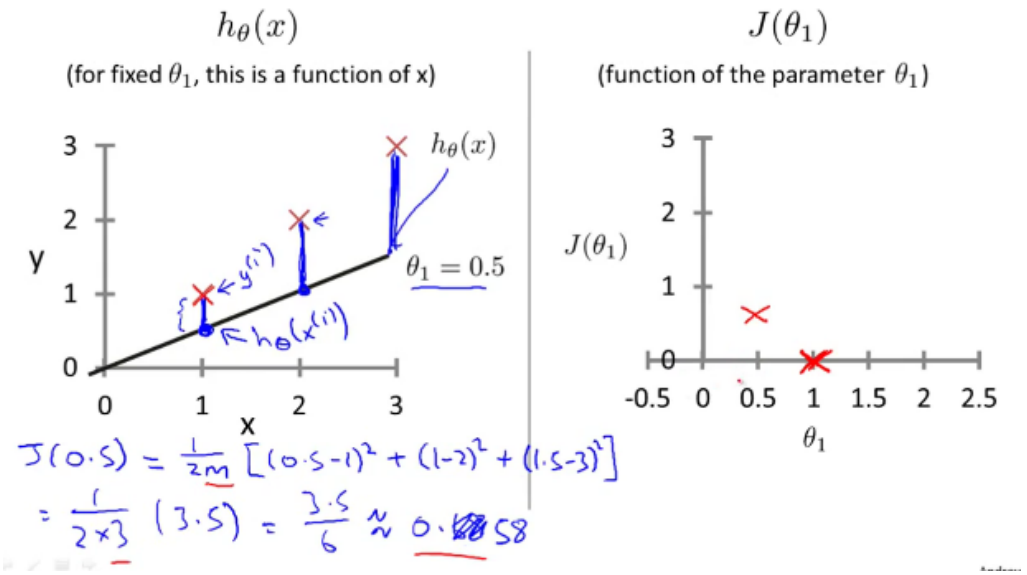

图 14: $\theta 1$等于0.5

当$\theta 1$等于0时：假设函数是一条水平的直线

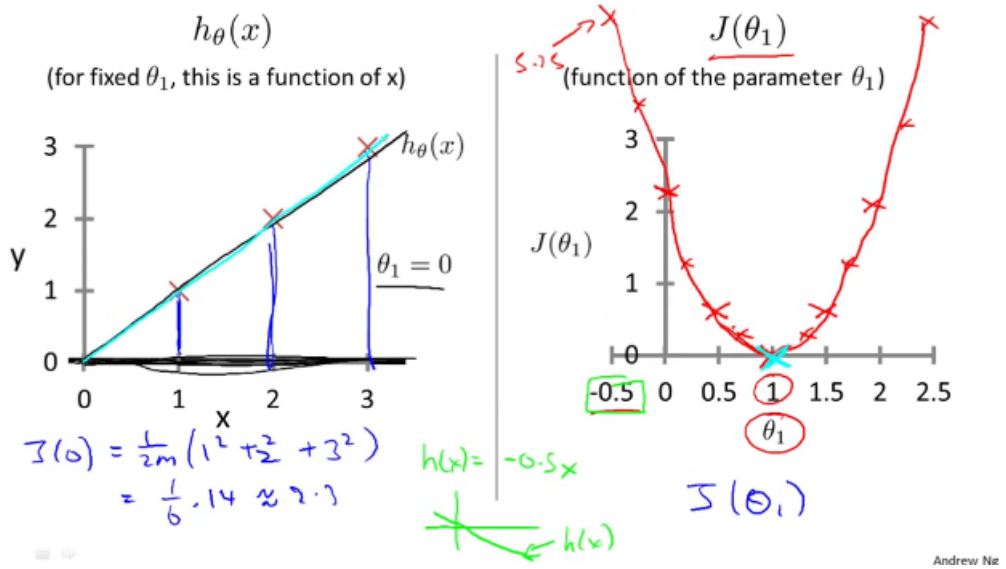

图 15: $\theta 1$等于0

回顾：任何一个$\theta 1$对应着一个不同的假设函数，或者说对应这左边一条不同的拟合直线，对于任意的$\theta 1$，你可以算出一个不同的J($\theta 1$)的值，我们可以利用这些来求右边的值。

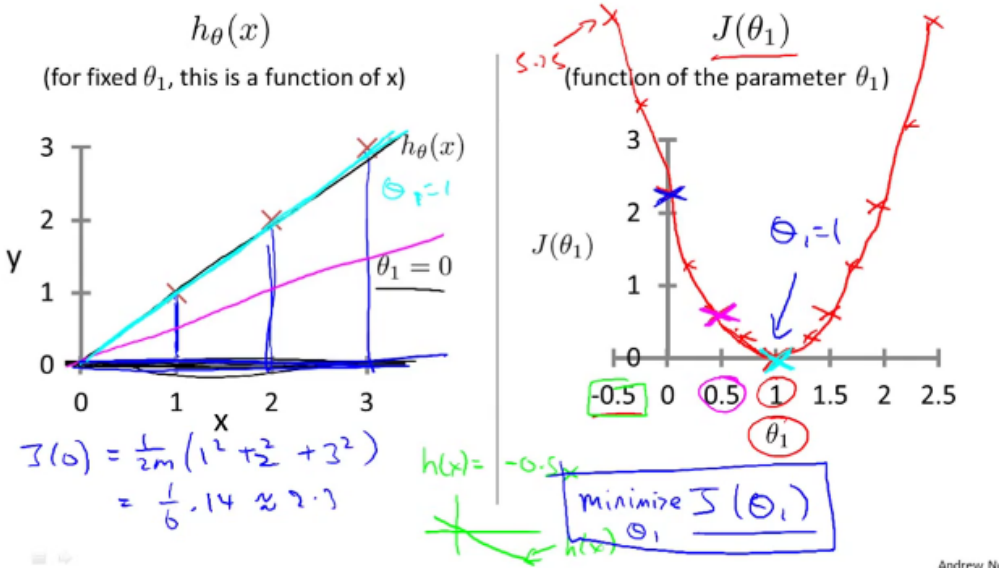

图 16: θ1的各种情况

找到一个θ1的值来将J(θ1)最小化，对于特定的训练样本，我们最后可以完美的拟合，这就是为什么最小化J(θ1)对应着寻找一个最佳拟合直线的目标。

2.2.2 Cost Function-Intuition II

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

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

Goal:

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

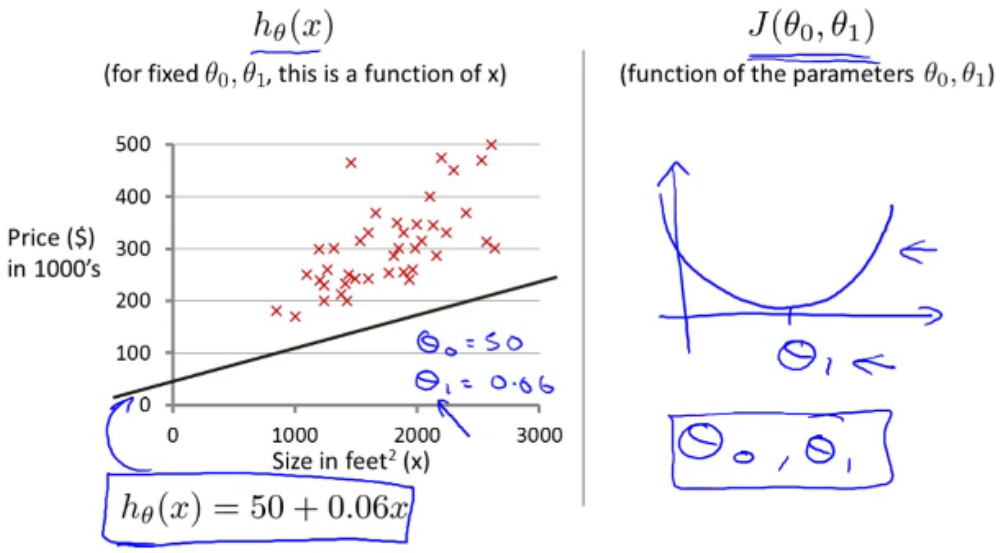

图 17: 代价函数的直观理解

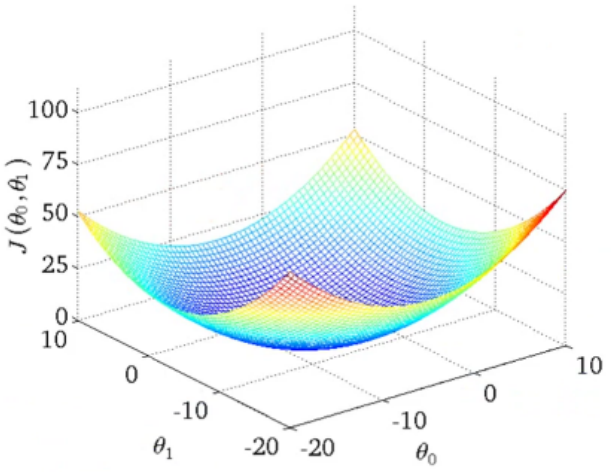

图 18: 三维曲面图

右图会得到一个三维曲面图，改变横坐标轴的两个值会得到不同的代价函数J(θ 0, θ 1)，竖直方向的高度就代表代价函数的值，但是之后用用轮廓图来表示代价函数：

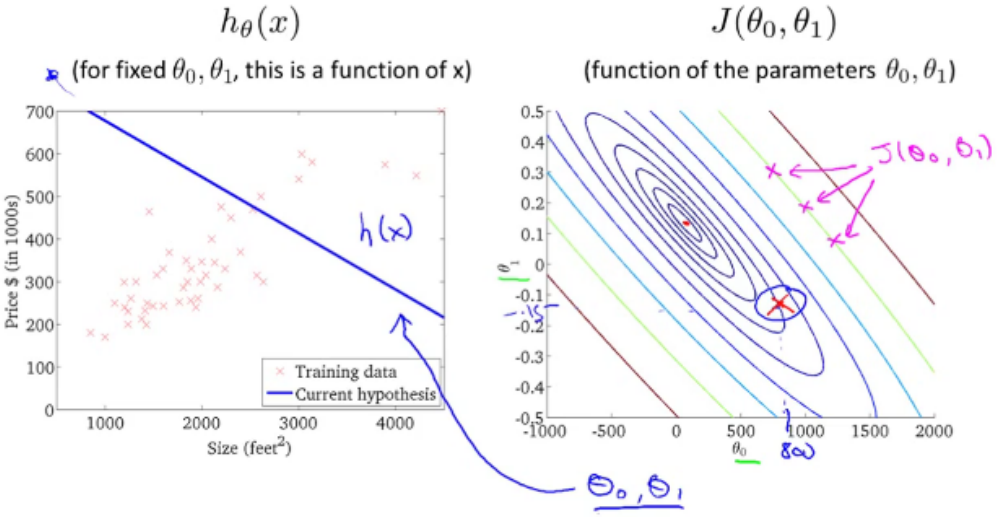

图 19: 轮廓图表示

轮廓图可以直观地观察代价函数，但是上图可以看出来不是很好的拟合，再看一个例子：

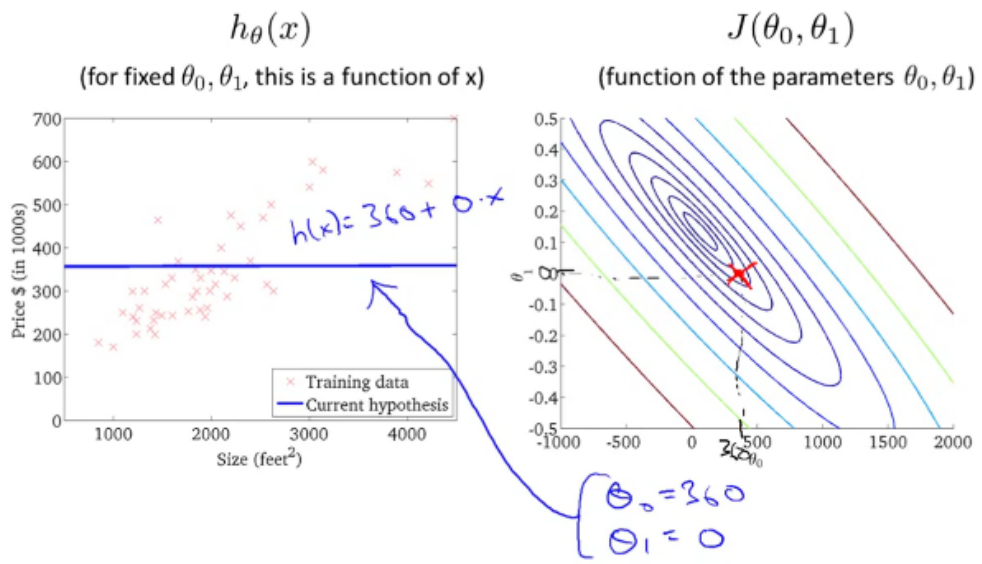

图 20: 拟合1

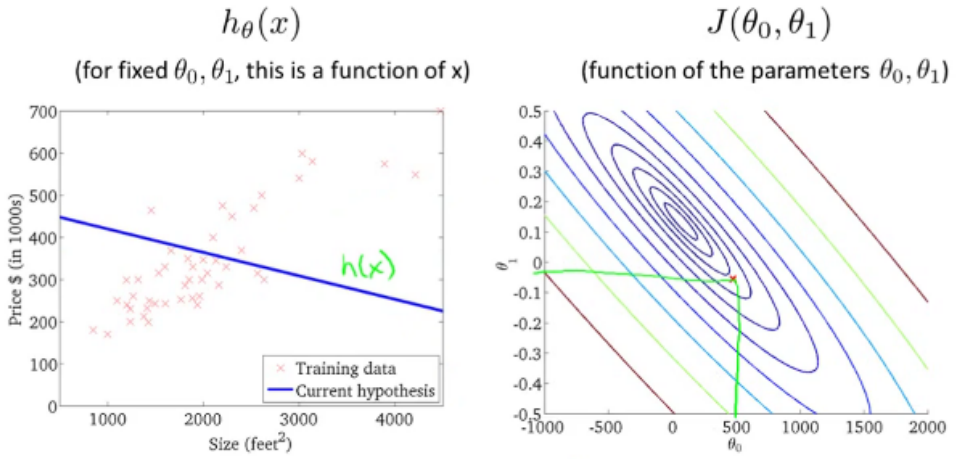

图 21: 拟合2

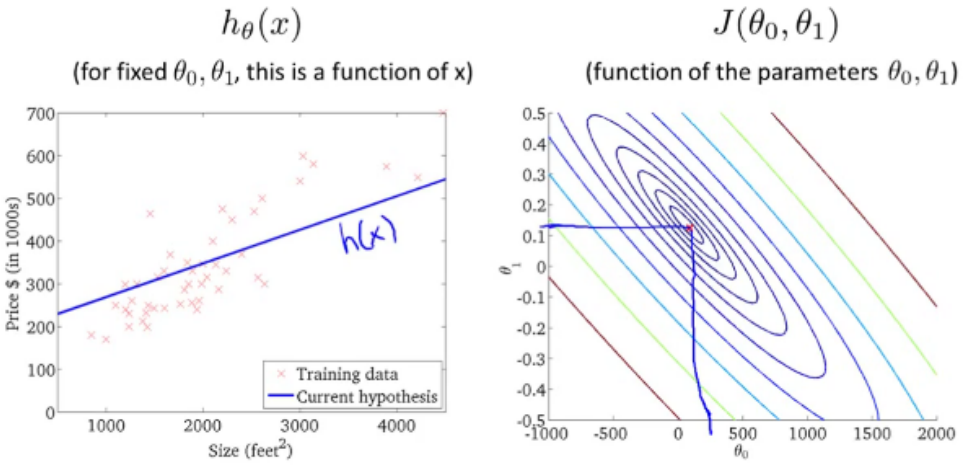

图 22: 拟合3

虽然拟合不好，但是已经离最小值很近了， 因此误差平方和 或者说 训练样本和假设的距离的平方和 这个距离值的平方和 非常接近于最小值 尽管它还不是最小值。
后面我们会遇到更复杂、更高维度、更多参数的情况，而这些情况是很难画出图的 因此更无法将其可视化，因此我们真正需要的是编写程序来找出这些最小化代价函数的 θ 0和 θ 1的值。

2.3 Gradient Descent

梯度下降函数可以适用于多种多样的函数求解
基本构想:

Have some function $J(\theta_0, \theta_1)$ $J(\theta_0, \theta_1, \theta_2, \ldots, \theta_n)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$ $\min_{\theta_0 \cdots \theta_n} J(\theta_0, \ldots, \theta_n)$

Outline:

- Start with some θ_0, θ_1 (say $\theta_0 = 0, \theta_1 = 0$)
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

图 23: 梯度下降算法

梯度下降算法三维图:

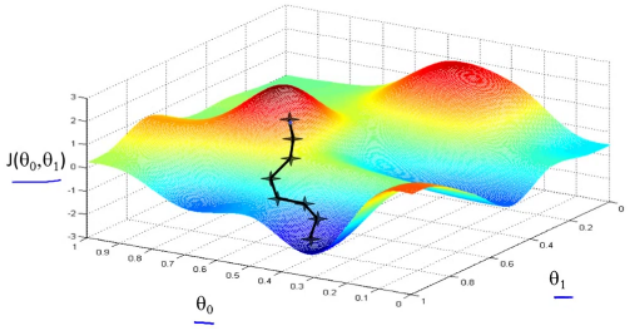

图 24: 三维图一

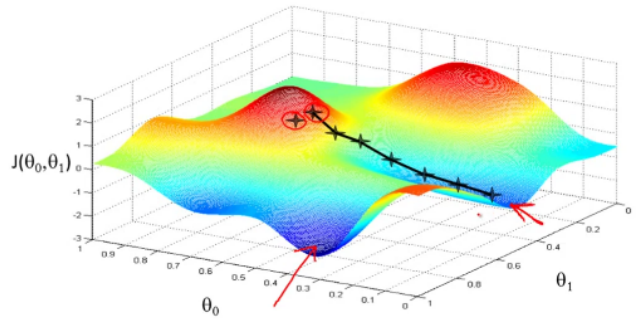

图 25: 三维图二

对梯度下降进行初始化，如此往复，达到第二个局部优处，如果从刚才的第一个点出发，会得到第一张图的局部最优解，但是如果你的起点偏移了一点，起点位置不同，到那时得到的是非常不同的局部最优解，这就是梯度下降的一个特点。

（1）梯度算法的微妙之处：如果想要更新这个等式，需要同时更新θ0和θ1的值。

（2）实现方法：计算公式右边的部分，然后将计算出的结果存入temp0和temp1，然后同时更新θ0和θ1的值.

（3）同时更新是梯度下降中的一种常用方法

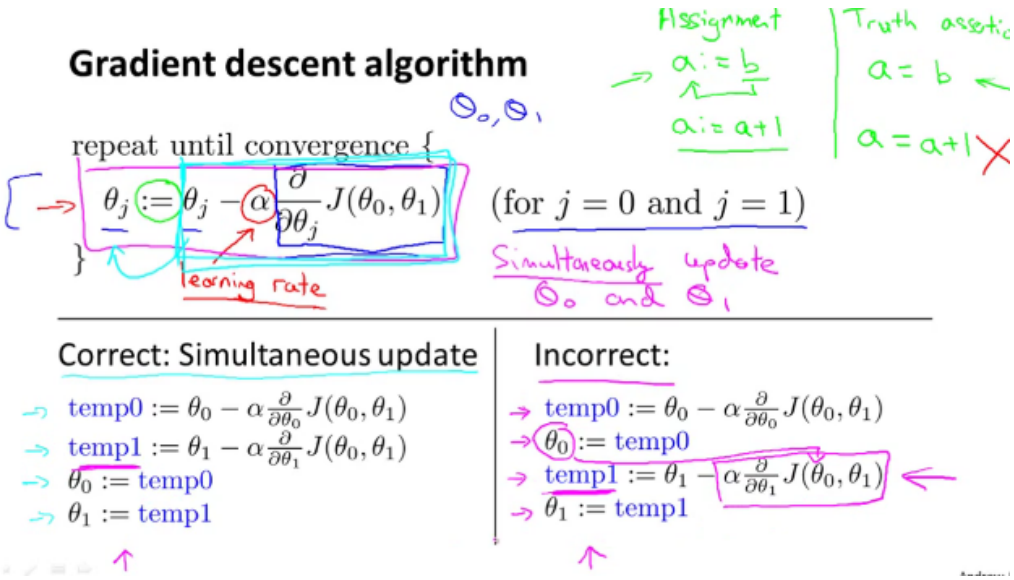

图 26: 梯度下降算法的定义及同时更新

2.3.1 Gradient Descent Intuition

2.3.2 Gradient Descent for Liner Regression