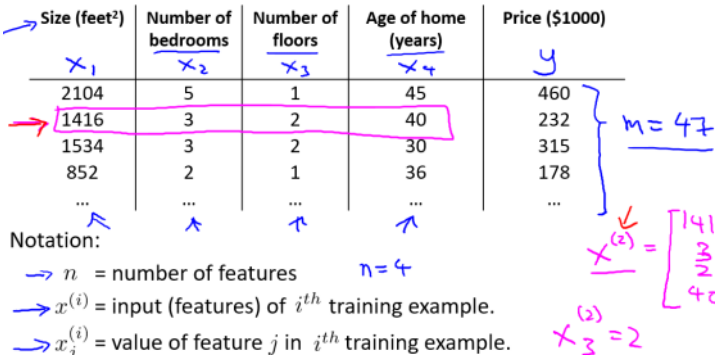
4 多元线性回归

2022年10月19日　19:20

4.1多特征量

多特征量(Multiple Features (variables)):比单一变量更多变量的描述。



Size (feet²) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($1000) y
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

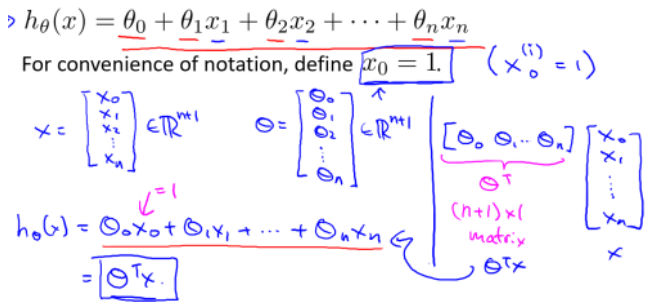
$m = 47$

Notation:

- n = number of features　$n = 4$
- $x^{(i)}$ = input (features) of i^{th} training example.
- $x_j^{(i)}$ = value of feature j in i^{th} training example.

$x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$　$x_3^{(2)} = 2$

多变量的假设函数：



$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

For convenience of notation, define $x_0 = 1$. $(x_0^{(i)} = 1)$

$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$　$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$

$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$　$(x_0 = 1)$

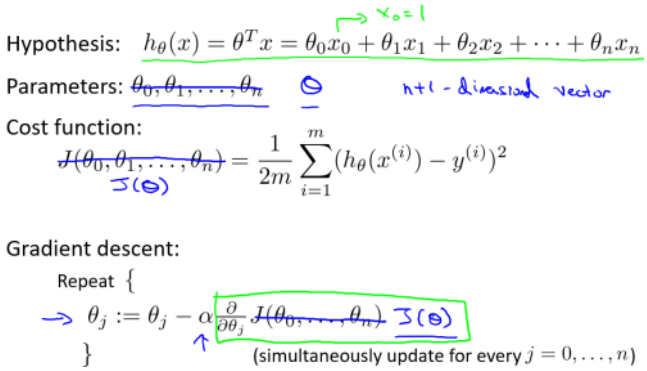
$= \theta^T x$

θ^T: $(n+1) \times 1$ matrix

可以表示为参数向量的转置与变量向量的相乘。也即多元线性回归(Multivariate linear regression)。

4.2梯度下降法

多元函数的代价函数可以用一个参数向量表示出来：



Hypothesis: $h_\theta(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$　$x_0 = 1$

Parameters: ~~$\theta_0, \theta_1, \ldots, \theta_n$~~ θ　$n+1$-dimensional vector

Cost function:

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

$J(\theta)$

Gradient descent:

Repeat {

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} ~~J(\theta_0, \ldots, \theta_n)~~ J(\theta)$

}

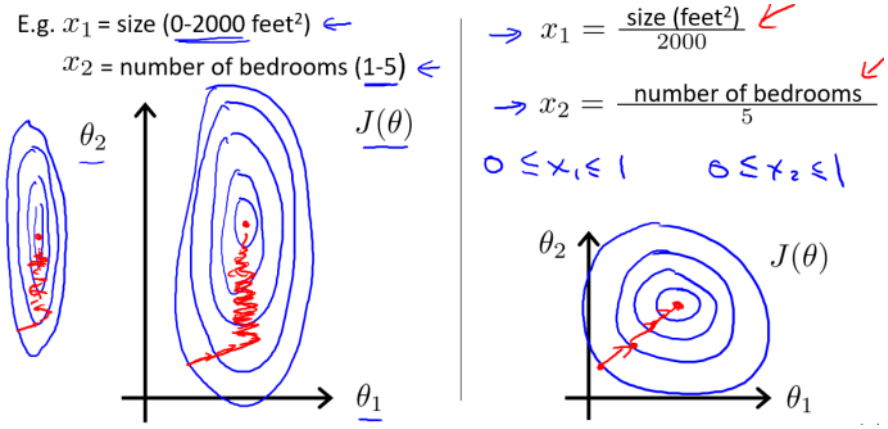
(simultaneously update for every $j = 0, \ldots, n$)

多元函数梯度下降算法的实现：

4.3梯度下降技巧：特征缩放

特征缩放(Feature Scaling):确保特征都处在一个相近的范围内，梯度下降法能更快地收敛。

一个例子：房屋面积和卧室数量分别为两个特征变量



左图中由于2000：1的范围比例，让等高线近似成一个椭圆形，在其上进行梯度下降是很耗时的；通过特征缩放，让变量除以它们的范围，如右图所示，代价函数的等值线不会偏移得那么严重，可以证明其算法的耗时会更小一些，从而更快地收敛。

一般执行特征缩放能够将变量的范围约束在[-1,1]。当然，范围也不能太小，比如[-0.0001,0.0001]，它也严重偏离了[-1,1]的范围大小。

均值归一化(Mean normalization):通过令变量减去其均值后除以范围大小(最大值减去最小值)，使特征值具有为0的平均值。

Replace x_i with $x_i - \mu_i$ to make features have approximately zero mean (Do not apply to $x_0 = 1$).

E.g. $x_1 = \frac{size - 1000}{2000}$

$x_2 = \frac{\#bedrooms - 2}{5}$ 4

Average size = 1000

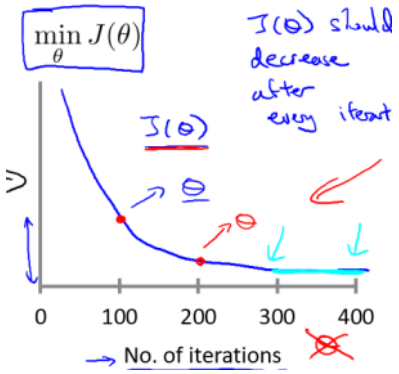
1-5 bedrooms

$-0.5 \le x_1 \le 0.5$, $-0.5 \le x_2 \le 0.5$

这里的x_2虽然没有除以4，但只要是相近的范围，算法都能够很好运行。

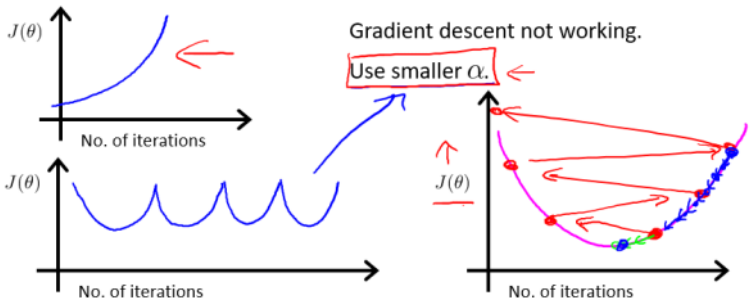
4.4学习率

调试(Debugging):确保梯度下降正确工作。



“代价函数-迭代次数”曲线反映了每增加一次迭代代价函数都会减小。
一种自动判断收敛的方法：当某一次迭代后代价函数的减小值小于10^(-3)，则判断其已经收敛。由于这种方法的阈值很难确定，从而通过查看曲线的方法更为准确高效。

- 如果“代价-迭代次数”曲线随着迭代次数增加而上升，那么意味着很有可能是学习率设置的太大了，也就是说每一步的步长太大，从而越过了局部最小值点而导致恶化。



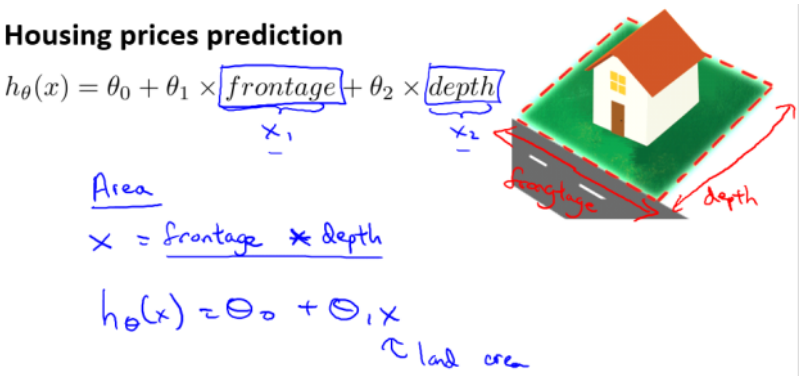
- 若曲线是上图第二种所示上下起伏波动，则一般也是由学习率设置太大导致的。

但是学习率也不能设置得太小，这会使算法收敛得很慢。

在实际的应用中，老师一般会尝试很多个学习率的设置，找到一个能又快又好的值。如0.001，0.003，0.01，0.03，0.1，0.3，1，…

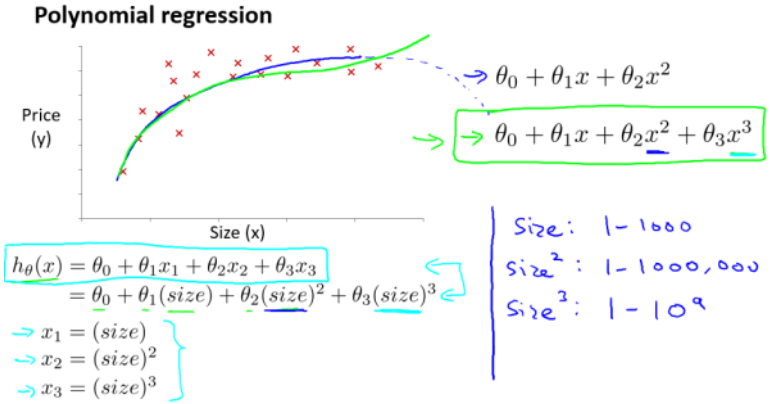
4.5特征和多项式回归

依然是房价预测的例子：



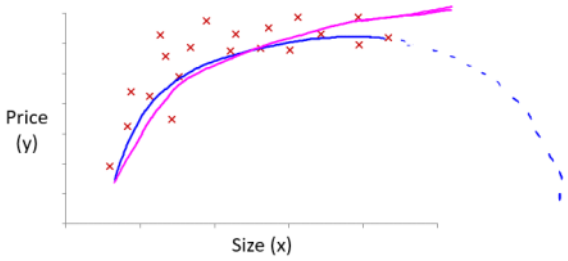
对于一个房子的房价预测，可以将房子的长度和宽度作为其预测特征，也可以将其面积=长度*宽度作为特征，通过定义一个新的特征，我们可能会得到一个更好的模型。

对于一个例子，可能显而易见用线性模型是不合适的，如下图所示，这样就可以想到多项式回归。



考虑到房子的价格不太会随着面积的增加而下降，三次多项式也许是一个很好的拟合方案。而多项式回归可以与之前讲过的多元线性回归联系起来，如上图式所示。可以用类似多元线性回归的方式来求解多项式回归的模型，但是这样一来特征缩放就显得尤为重要。因为三个带有特征变量的项存在幂次的差别，有很大的不同。

根据平方根函数的特征，也可以尝试用平方根函数对上述的例子进行拟合。



$h_\theta(x) = \theta_0 + \theta_1(size) + \theta_2(size)^2$

$h_\theta(x) = \theta_0 + \theta_1(size) + \theta_2\sqrt{(size)}$

我们应该从符合模型的角度考虑来确定选择的特征以及拟合的方案。

4.6正规方程

正规方程(Normal Equation)：一种求模型参数的解析解法。

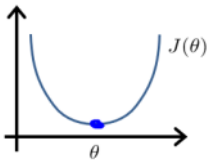
从一个很简单的例子谈起，

Intuition: If 1D ($\theta \in \mathbb{R}$)

$J(\theta) = a\theta^2 + b\theta + c$

$\frac{d}{d\theta}J(\theta) = \cdots \overset{set}{=} 0$

Solve for θ



对于如图所示的一元函数，根据微积分的知识可知，对其进行求导并令其为0后，所解出的参数值极为极值点。然而一般的代价函数没有这么简单，其模型参数为参数向量。

$\theta \in \mathbb{R}^{n+1}$ $\quad J(\theta_0, \theta_1, \ldots, \theta_m) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$

$\frac{\partial}{\partial \theta_j}J(\theta) = \cdots \overset{set}{=} 0$ (for every j)

Solve for $\theta_0, \theta_1, \ldots, \theta_n$

如果依然按照上述方法将向量对应的方程遍历，那么求解微积分的计算会非常复杂。

对于一个含有四个样本的数据集：

x_0	Size (feet²) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($1000) y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \quad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

$m \times (n+1)$ $\qquad$ m-dimensional vector

$\theta = (X^T X)^{-1} X^T y$

构建特征矩阵X和向量y，那么就可以得到使代价函数最小化的 θ ：

$\theta = (X^T X)^{-1} X^T y$

Octave的算法实现：

```
pinv(X'*X)*X'*y
```

使用正规方程法就不用考虑之前梯度下降中所提到的特征缩放的问题。

梯度下降与正规方程的特点和使用场景：

梯度下降：

- 需要选择学习率，即需要运行很多次来找到最好的学习率，这意味着额外的计算

和工作
- 需要很多次迭代，计算可能会很慢
- 在**特征量n非常大**的情况下也能运行得相当好，尤其是对于具有上百万案例的模型很有效

正规方程：
- 不需要选择学习率，非常方便和易于实现
- 不需要多次迭代，也不用采取多余的工作来检查收敛性等
- 需要计算 $(X^TX)^{-1}$，对于大多数计算应用来说计算逆矩阵的代价以矩阵维度的三次方增长，其时间复杂度为O(n^3),所以如果**特征量非常多**的话用正规方程法会很慢，尤其是n上万的情况开始，这种方法可能会很慢

4.7正规方程与不可逆性(选学)

根据线性代数的相关知识，矩阵存在可逆与不可逆的情况，也即奇异矩阵或退化矩阵，那么对于正规方程中的X^TX，如果不可逆该怎么办？
事实上，它不可逆的情况很少发生，而在Octave中，pinv()和inv()函数都可以用来计算矩阵的逆。而pinv()函数又叫伪逆(pseudo-inverse)矩阵函数，使用它可以计算出结果，即使所计算的矩阵是不可逆的。

X^TX出现不可逆的情况可能是下面两个原因：
- 学习过程中包含了冗余的特征量，例如在房价预测的例子里，既有以平方英尺为单位的面积，也有以平方米为单位的面积，两个特征量存在线性变换关系，从而造成冗余
- 运行的学习算法包含太多特征量，也就是m $\leqslant n$。例如我们有包含10个样本的数据集，但是每一个样本包含100个特征，那么10个样本对于101个参数的求解太少了。可以采取的解决办法是看看能否删除某些特征或者使用正则化。