3.线性回归

表示法

我们在接下来的几节先讨论简单线性回归，也就是只考虑一个输入变量，一个输出变量的线性回归。

为了后面解释和使用方便，首先正式定义简单线性回归中的基本元素。

为了后面解释和使用方便，首先正式定义简单线性回归中的基本元素。

$x^{(i)}$ 表示输入变量（自变量），第一部分例子中的X。

$y^{(i)}$ 表示输出变量（因变量），第一部分例子中的Y。

一对 $(x^{(i)}, y^{(i)})$ 表示一组训练样本。

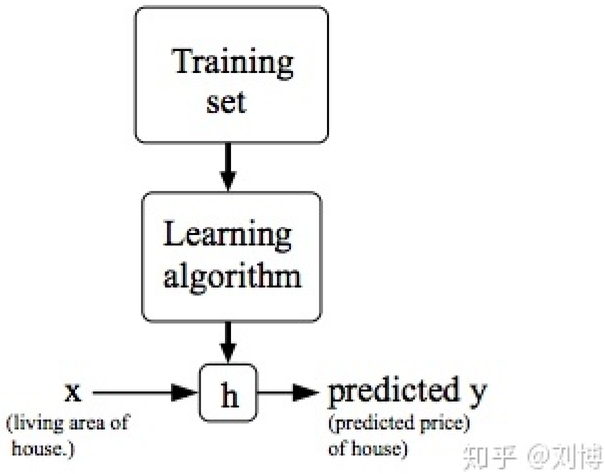
m个训练样本 $(x^{(i)}, y^{(i)}); i = 1, \ldots, m$ 称为训练集。

上面表示法中的i代表第i个样本。

大写的X代表所有输入值组成的空间。

大写的Y代表所有输出值组成的空间。

监督学习的定义是, 给定一个训练集, 我们的目标是“学习”得到一个函数 h : x→y, 使h(x)是真实值y的一个“好的”预测值。这里h叫做模型，也叫做**假设（hypothesis）**.



如果我们要预测的输出值是**连续的**，那么该问题就称作**回归问题**。

对于简单线性回归来说，我们的模型h可以表示如下：

对于简单线性回归来说，我们的模型h可以表示如下：

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

其中的 θ_0 和 θ_1 代表模型的参数。

线性回归的目标是：**求得最合适的 θ_0 和 θ_1 ，使得模型效果最好。**

代价函数（Cost Function）

如何衡量模型效果的好坏？

代价函数的作用是用来测量我们的模型h的准确程度。

以最简单的一个代价函数为例，也就是**均方误差（Mean squared error）**。

假设我们的模型如下： $h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$

均方误差就是求预测值与真实值之间的差值的平方，即：

$$(h_\theta(x^{(i)}) - y^i)^2$$

如果对于训练集中的所有样本求均方误差，就是将每个样本的均方误差求和再求平均，即：

$$\frac{1}{m}\sum^{m} i = 1(h\theta(x^{(i)}) - y^i)^2$$

Ok，这就是均方误差代价函数，想象一下，模型h以及代价函数的h参数是什么？

答案是 θ_0 和 θ_1 ,因为我们的目标是找到最合适的 θ_0 和 θ_1 ，使模型最好。

因此，如果我们使用大写的J来表示代价函数，那么它的定义如下(注意J是关于 θ 的函数)：

$$J(\theta) = \frac{1}{2m}\sum^{m} i = 1(h\theta(x^{(i)}) - y^i)^2$$

这里的一个细节是，公式中的分母由m变成了2m，这里并不影响公式的作用，但是对于后面的工作有帮助，我们后面将会提到这点。

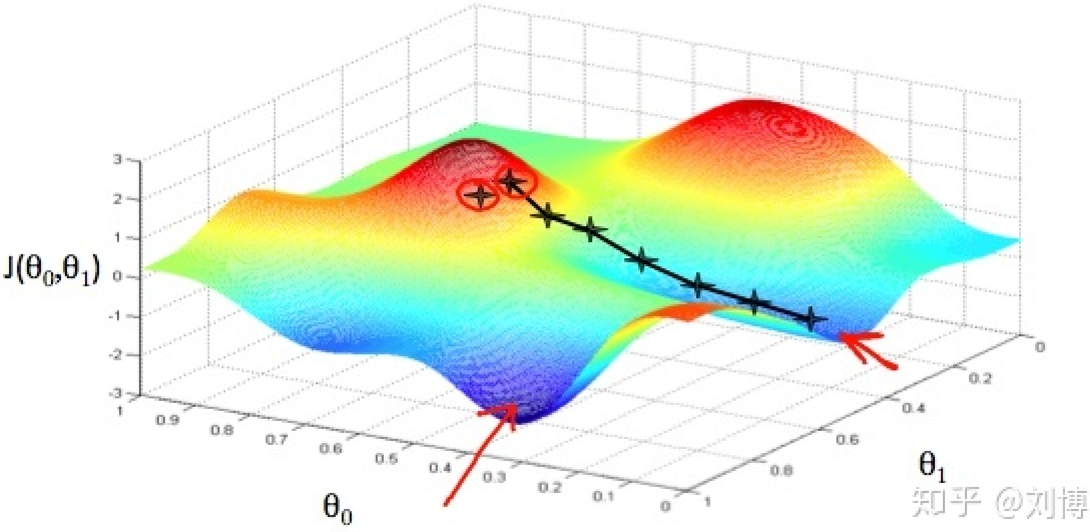
这里需要考虑一个问题，有了代价函数，如何使模型效果最好？

由于代价函数越大，代表预测值与真实值之间的误差就越大，因此，问题的答案是使代价函数最小的参数 theta就是最好的参数。

梯度下降（Gradient Descent）

下面介绍另一种求解参数的方法，梯度下降法。

既然代价函数是关于θ的函数，我们就可以可视化该函数，见下图：



图中的蓝色区域是代价函数最小的点，因此，找到该点对应的θ，即完成了任务。

如何找到最低点对应的θ?

答案是对代价函数求偏导数：

$$\frac{\partial}{\partial \theta_j}J(\theta)=\frac{\partial}{\partial \theta_j}\frac{1}{2}(h_\theta(x)-y)^2\ =\frac{1}{2}\times 2\times(h_\theta(x)-y)\frac{\partial}{\partial \theta_j}(\sum_{i=0}^{n}\theta_i x_i-y)\ =(h_\theta(x)-y)x_j$$

上面的推导中1/2和2互消了，所以给了我们一个更加直观简洁的结果，这就是为什么代价函数公式要乘以一个1/2.

需要注意的是，上面的公式中，我们采用了一个更加通用的(考虑了多个自变量的情况)表示模型h的方式：

$$h(\theta)=\sum_{i=0}^{n}\theta_i x_i=\theta_0 1+\theta_1 x_1+\cdots+\theta_n x_n$$

对于我们的简单线性回归函数来说：

注意，这里 θ_0 乘以一个常数1，也可以认为乘以了常量$x_0 = 1$

所以代价函数对 θ_0 求偏导的结果就是：

$$\frac{\partial}{\partial \theta_0}J(\theta)=(h_\theta(x)-y)x_0=(h_\theta(x)-y)$$

代价函数对 θ_1 求偏导的结果就是：

$$\frac{\partial}{\partial \theta_1}J(\theta)=(h_\theta(x)-y)x_1$$

注意，对 θ 求偏导数的意义是得到**这一点上的切线的斜率**，它将给我们一个向最小值移动的方向。

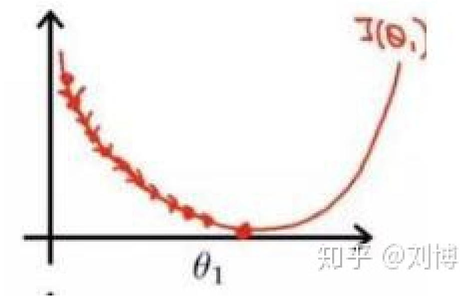
因此，θ 减去偏导数就等于 θ 向最小值的方向移动了一步。这一步的大小由一个参数α决定，也称作学习率。用公式表达如下：

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

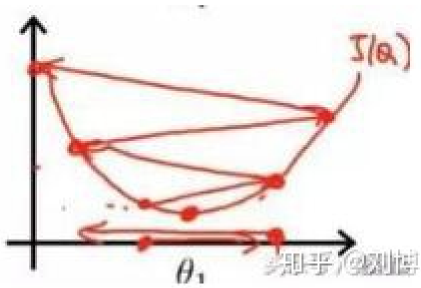
注意，上式中的符号 $:=$ 表示重复该过程，直到收敛。也就是我们的**梯度下降公式。**

学习率

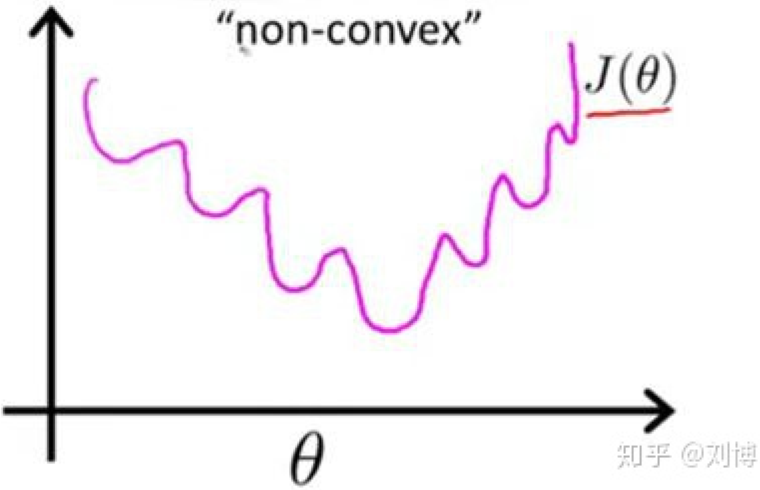
现在我们讨论一下学习率，当学习率比较小时，我们得到最优解的速度将很慢。



当学习率比较大时，我们很容易无法得到全局最优解。



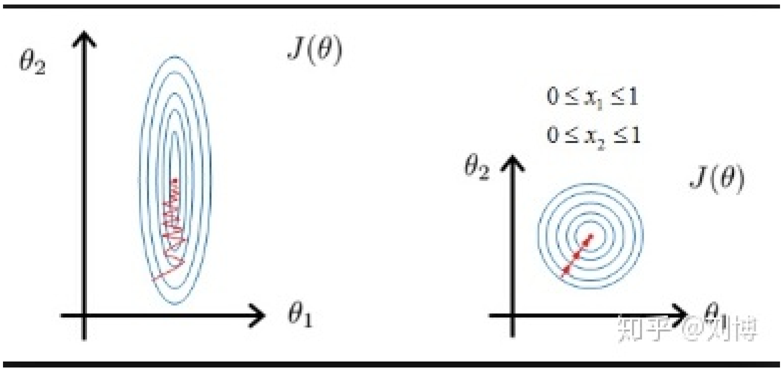
另外一种情况是，我们的学习率设置的比较小时，得到了局部最优解，而不是全局最优解的情况。



最后一种情况，对于我们使用的代价函数来说并不成立，因为我们使用的均方误差MSE是一个凸函数，也就是说说，该函数任意两点的连线都不会与该函数交叉，所以，该函数不存在局部最优解。

数据归一化

为了提高求解效率，我们还需要进行数据归一化处理。为什么要这么做？我们先看下面的图。



左边是没有经过归一化处理，右边是经过归一化处理，两张同都使用梯度下降，很明显右边的方法通过更少的步骤得到了解。

梯度下降法的三种方式

第一种方式就是全量梯度下降，也就是说每一次梯度下降都利用全部的数据。这种方法简称为BGD（Batch Gradient Descent）。该算法的缺点是当数据量m较大时，速度很慢。

第二种方式是随机梯度下降，即随机梯度下降（Stochastic Gradient Descent），简称SGD。这种方法的意思是， 每次梯度下降的过程使用一个随机的样本，但是该方法有一个问题，就是每次选取的学习率如果太小则速度很慢，太大怎么无法得到全局最优解，解决该问题的方法就是灵活设置学习率，让学习率一开始比较大，随后逐渐减小，这种方法也称作模拟退火（simulated annealing）。

第三种方法介于上面两种方法之间，即mini-batch Gradient Descent，简称mini-batch GD，该方法是每次梯度下降过程使用一个比较小的随机数据集。该方法的好处是能够利用计算机对于矩阵求解的性能优化，进而加快计算效率。

矩阵表示

Algebra Perspective

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(n)} \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \dots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \dots & x_d^{(n)} \end{bmatrix} \qquad \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- Prediction $\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{\theta} \\ \boldsymbol{x}^{(2)}\boldsymbol{\theta} \\ \vdots \\ \boldsymbol{x}^{(n)}\boldsymbol{\theta} \end{bmatrix}$

- Objective $J(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top(\boldsymbol{y} - \hat{\boldsymbol{y}}) = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})$

Matrix Form

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) \qquad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

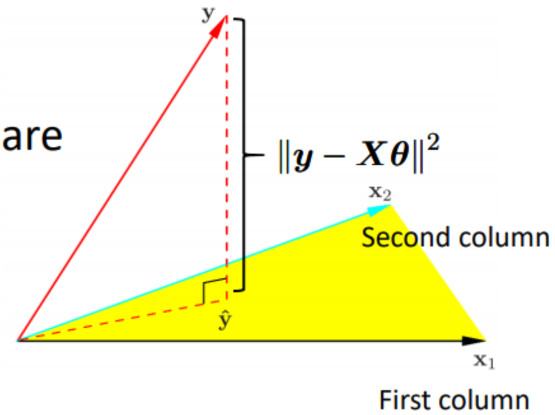
- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})$$

- Solution

$$\begin{aligned} \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} \;&\Rightarrow\; \boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) = \mathbf{0} \\ &\Rightarrow\; \boldsymbol{X}^\top\boldsymbol{y} = \boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{\theta} \\ &\Rightarrow\; \hat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y} \end{aligned}$$

Matrix Form



- Then the predicted values are

$$\hat{\boldsymbol{y}} = \boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$$
$$= \boldsymbol{H}\boldsymbol{y}$$

H: hat matrix

- Geometrical Explanation
  - The column vectors $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d]$ form a subspace of $\mathbb{R}^n$
  - *H* is a least square projection

$$\boldsymbol{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \cdots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{(n)} & x_2^{(n)} & x_3^{(n)} & \cdots & x_d^{(n)} \end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d] \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$\boldsymbol{X}^\top\boldsymbol{X}$ Might be Singular

- When some column vectors are not independent
  - For example, $\mathbf{x}_2 = 3\mathbf{x}_1$

  then $\boldsymbol{X}^\top\boldsymbol{X}$ is singular, thus $\hat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$ cannot be directly calculated.

- Solution: regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta})^\top(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2$$

Matrix Form with Regularization

- Objective

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \qquad \min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

- Gradient

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) + \lambda\boldsymbol{\theta}$$

- Solution

$$\begin{aligned}
\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0} \;\rightarrow\; & -\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) + \lambda\boldsymbol{\theta} = \mathbf{0} \\
\rightarrow\; & \boldsymbol{X}^\top\boldsymbol{y} = (\boldsymbol{X}^\top\boldsymbol{X} + \lambda\boldsymbol{I})\boldsymbol{\theta} \\
\rightarrow\; & \hat{\boldsymbol{\theta}} = (\boldsymbol{X}^\top\boldsymbol{X} + {\color{red}\lambda\boldsymbol{I}})^{-1}\boldsymbol{X}^\top\boldsymbol{y}
\end{aligned}$$