第1周

一、引言

1.3 监督学习

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。在房价的例子中，我们给了一系列房子的数据，我们给定数据集中每个样本的正确价格，即它们实际的售价。

然后运用学习算法，算出更多的正确答案。比如你朋友那个新房子的价格。用术语来讲，这叫做回归问题。我们试着推测出一个连续值的结果，即房子的价格。

1.4 无监督学习

无监督学习，它是学习策略，交给算法大量的数据，并让算法为我们从数据中找出某种结构。

二、单变量线性回归

2.1 模型表示

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

一种可能的表达方式为：$h_\theta\left(x\right)=\theta_0+\theta_1 x$，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。

2.2 代价函数

我们的目标便是选择出可以使得建模误差的平方和能够最小的模型参数。 即使得代价函数
$J\left(\theta_0,\theta_1\right)=\frac{1}{2m}\sum\limits_{i=1}^{m}\left(h_\theta(x^{(i)})-y^{(i)}\right)^2$最小。

2.5 梯度下降

梯度下降背后的思想是：开始时我们随机选择一个参数的组合$\theta_0,\theta_1.....$，计算代价函数，然后我们寻找下一个能让代价函数值**下降最多**的参数组合。我们持续这么做直到到到一个局部最小值。

因为我们并没有尝试完所有的参数组合，所以不能确定我们得到的局部最小值是否便是全局最小值，选择不同的初始参数组合，可能会找到不同的局部最小值。

批量梯度下降（**batch gradient descent**）算法的公式为：

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{(for } j = 0 \text{ and } j = 1) \\
&\}
\end{aligned}
$$

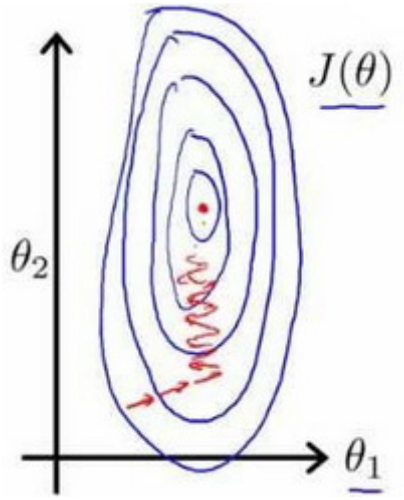
其中α是学习率，它决定了我们沿着能让代价函数下降程度最大的方向向下迈出的步子有多大，在批量梯度下降中，我们每一次都同时让所有的参数**减去学习速率乘以代价函数的导数。**

第2周

四、多变量线性回归

4.3 梯度下降法实践1-特征缩放

以房价问题为例，假设我们使用两个特征，房屋的尺寸和房间的数量，尺寸的值为 0-2000平方英尺，而房间数量的值则是0-5，以两个参数分别为横纵坐标，绘制代价函数的等高线图能，看出图像会显得很扁，梯度下降算法需要非常多次的迭代才能收敛。



解决的方法是尝试将所有特征的尺度都尽量缩放到-1到1之间。

最简单的方法是令：$x_n = \frac{x_n - \mu_n}{s_n}$，其中 μ_n是平均值，s_n是标准差。

第3周

六、逻辑回归

6.1 分类问题

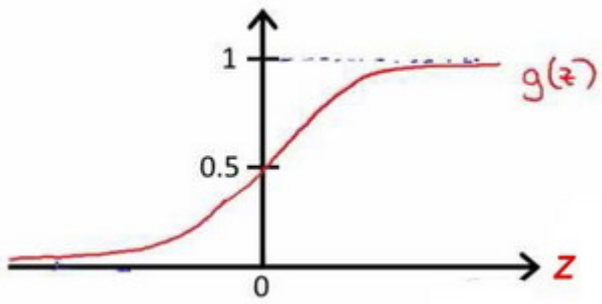
我们将因变量可能属于的两个类分别称为负向类和正向类，其中 0 表示负向类，1 表示正向类。

如果我们要用线性回归算法来解决一个分类问题，对于分类， 取值为 0 或者1，但如果你使用的是线性回归，那么假设函数的输出值可能远大于 1，或者远小于0，即使所有训练样本的标签 都等于 0 或 1。尽管我们知道标签应该取值0 或者1，但是如果算法得到的值远大于1或者远小于0的话，就会感觉很奇怪。

所以我们在接下来的要研究的算法就叫做逻辑回归算法，这个算法的性质是：它的**输出值永远在0到 1 之间。**

6.2 假说表示

> 我们引入一个新的模型，逻辑回归，该模型的输出变量范围始终在0和1之间。 逻辑回归模型的假设是：
> $h_\theta(x) = g(\theta^T X)$ 其中： X 代表特征向量 g 代表逻辑函数（**logistic function**)是一个常用的逻辑函数为S形函数（**Sigmoid function**），公式为： $g(z) = \frac{1}{1+e^{-z}}$。



6.4 代价函数

y=1，预测值越接近1，loss越小

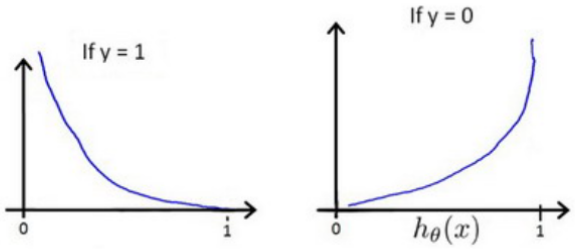
y=0，预测值越接近0，loss越小

线性回归的代价函数为：$J(\theta)=\frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2$。我们重新定义逻辑回归的代价函数为：

$J(\theta)=\frac{1}{m}\sum_{i=1}^{m}Cost\left(h_\theta\left(x^{(i)}\right),y^{(i)}\right)$，其中

$$\mathrm{Cost}(h_\theta(x),y)=\begin{cases}-\log(h_\theta(x)) & \text{if } y=1\\ -\log(1-h_\theta(x)) & \text{if } y=0\end{cases}$$

$h_\theta(x)$与$Cost\left(h_\theta(x),y\right)$之间的关系如下图所示：



$$Cost\left(h_\theta(x),y\right)=-y\times log\left(h_\theta(x)\right)-(1-y)\times log\left(1-h_\theta(x)\right)$$

七、正则化

7.1 过拟合的问题

1. 丢弃一些不能帮助我们正确预测的特征。可以是手工选择保留哪些特征，或者使用一些模型选择的算法来帮忙（例如**PCA**）
2. 正则化。 保留所有的特征，但是减少参数的大小。

7.2 代价函数

上面的回归问题中如果我们的模型是：$h_\theta(x)=\theta_0+\theta_1x_1+\theta_2x_2^2+\theta_3x_3^3+\theta_4x_4^4$ 我们可以从之前的事例中看出，正是那些高次项导致了过拟合的产生，所以如果我们能让这些高次项的系数接近于0的话，我们就能很好的拟合了。 所以我们要做的就是在一定程度上减小这些参数θ 的值，这就是正则化的基本方法。我们决定要减少θ_3和θ_4的大小，我们要做的便是修改代价函数，在其中θ_3和θ_4 设置一点惩罚。这样做的话，我们在尝试最小化代价时也需要将这个惩罚纳入考虑中，并最终导致选择较小一些的θ_3和θ_4。 修改后的代价函数如下：

$$\min_\theta \frac{1}{2m}[\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2+1000\theta_3^2+10000\theta_4^2]$$

通过这样的代价函数选择出的θ_3和θ_4 对预测结果的影响就比之前要小许多。假如我们有非常多的特征，我们并不知道其中哪些特征我们要惩罚，我们将对所有的特征进行惩罚，并且让代价函数最优化的软件来选择这些惩罚的程度。这样的结果是得到了一个较为简单的能防止过拟合问题的假设：

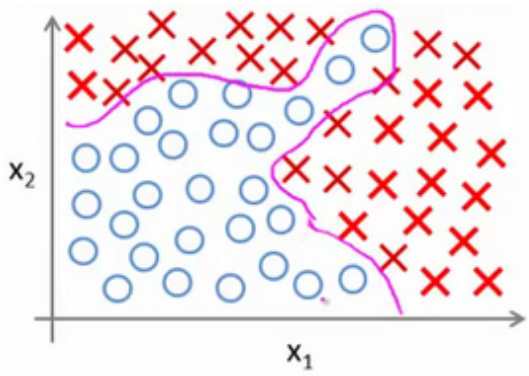
$$J(\theta)=\frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2+\lambda\sum_{j=1}^{n}\theta_j^2]$$

第4周

第八、神经网络：表述

8.1 非线性假设

无论是线性回归还是逻辑回归都有这样一个缺点，即：当特征太多时，计算的负荷会非常大。



$$g(\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_1x_2+\theta_4x_1^2x_2+\theta_5x_1^3x_2+\theta_6x_1x_2^2+\dots)$$

因此考虑神经网络

第5周

九、神经网络的学习

前面的层，链式求导可以复用后面的层

第8周

十三、聚类

13.2 K-均值算法

K-均值是一个迭代算法，假设我们想要将数据聚类成n个组，其方法为:

首先选择K个随机的点，称为**聚类中心**；

对于数据集中的每一个数据，按照距离个中心点的距离，将其与距离最近的中心点关联起来，与同一个中心点关联的所有点聚成一类。

计算每一个组的平均值，将该组所关联的中心点移动到平均值的位置。

重复步骤直至中心点不再变化。

第9周

十五、异常检测

15.2 高斯分布

我们可以利用已有的数据来预测总体中的μ和σ^2的计算方法如下： $\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$

$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \mu)^2$