机器学习 2012 笔记（Coursera）

Teacher- Andrew NG

在那追逐的道路上，踏碎过多少梦想

Author E-mail: tong.Elvis@gmail.com

2013-06-01

V0.1

目录

1 单变量的线性回归

1.1 关于监督学习

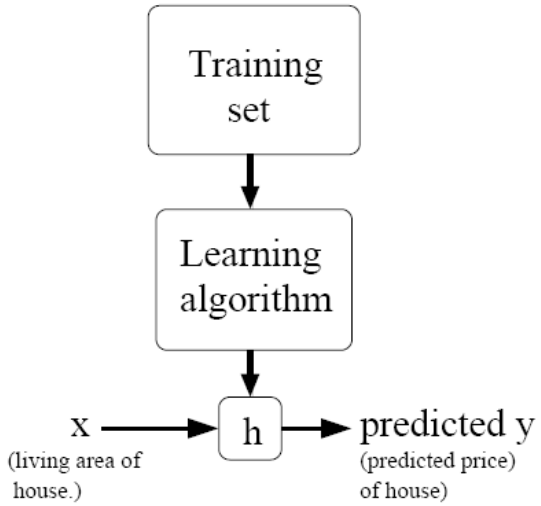
这里首先给出一个监督学习问题的例子：这里有来自 Portland 地区 47 所房子的面积和价格数据，通过上述给出的数据，我们是否可利用机器学习来根据房子的面积大小来预测房子价格。

监督学习就是对训练数据给出了正确结果，这里所给出了 47 所房子的真实价格(相对于预测价格来说)就是正确结果。而线性回归就是要根据输入的特征来进行相应的预测。

这里，我们把上述问题进一步阐述和抽象。m 表示训练集大小，x 表示特征属性，y 表示输出或目标属性。此外，$x^{(i)}$ 表示的是训练集中第 i 个例子的特征向量，$x_i^{(j)}$ 表示第 i 个例子的第 i 个特征值。这里，房子的大小就是特征属性，每个房子的特征向量中只有一个特征值，我们要预测的目标属性 y 是房子价格。

1.2 监督学习的形式化描述

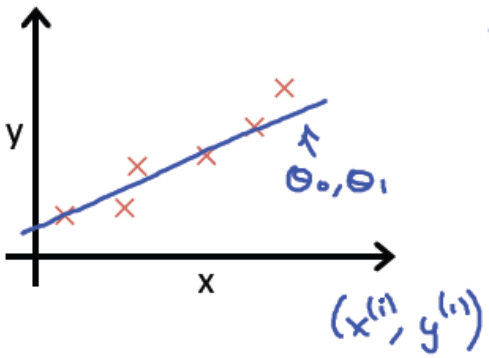
监督学习问题是对给定的训练数据集，找到较为准确的预测函数 $h：\mathrm{X} \mapsto \mathrm{Y}$。通常我们把该函数成为一个假设，过程可以表示如下：



对于单变量的线性规划，假设一般表示为: $h_\theta(x) = \theta_0 + \theta_1 x$,这里 x 是单标量，需要对其中的参数 θ_0 和 θ_1 的求解。

1.3 参数求解

在监督学习中，数据集一般分为测试数据集和训练数据集两个部分，要使得假设函数有较好的预测效果，为达到加好效果来进行参数的求解。



对于单变量的线性回归来说，就是选择合适的 θ_0 和 θ_1，使得 $h_\theta(x)$ 可以较好利用 x 来预测 y。

这里将损耗函数(cost function)定义为 $J(\theta_0,\theta_1) = 1/2m\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$，这里进行参数 θ_0 和 θ_1 求解时，其目标就是最小化损耗函数即 $\min_{\theta_0,\theta_1} J(\theta_0,\theta_1)$。

利用梯度下降方法可以进行参数求解，其核心思想就是：对 θ_0 和 θ_1 进行一定初始化，不断对 θ_0 和 θ_1 进行修改和调整以求得 $J(\theta_0,\theta_1)$ 的最小值。

该算法可以形式化表述为：循环{$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta_0,\theta_1)$}一直到收敛，对于单变量的线性回归而言这里的 j 可取的值为 0 和 1。（注意：这里的参数需要同时进行更新，因为参数会影响 $J(\theta_0,\theta_1)$，所以先用临时变量存储新的参数，最后统一更新；此外，关于判定收敛的问题，一般可以认为 $J(\theta_0,\theta_1)$ 在经过一次迭代或循环后，其下降程度不超过 ε 即认为收敛，一般 ε 可设为 10^{-3}）。

将损耗函数带入，得到如下过程:

1. 判断是否收敛（即 $J(\theta_0,\theta_1)$ 的改变是否小于 ε），否则，执行 2；

$$tmp0 = \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})$$

2. 更新参数，然后执行 1。 $tmp1 = \theta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \bullet x^{(i)}$

$$\theta_0 = tmp0$$

$$\theta_1 = tmp1$$

梯度下降方法可以收敛到局部最优解，即使学习参数 α 是固定值。一般说 α 是变化的，其值越大，即下降速度较快，可能会跳过局部最优；其值较小，即下降速度较低，收敛较慢。一般说来都是先快后慢，可以以 90%左右的速度进行 α 的下降。

在进行参数更新时，如果是以整个训练集为基础，则称为 Batch 梯度下降。还有增量式更新，表示如下（只修改步骤 2）：

2.For i =1 to m, $\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$(一般我们对训练集中任意记录 $x^{(i)}$ 增加第一维属性 $x_0^{(i)} = 1$，这样预测函数为 $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1$)。继续执行 1。

2 多变量线性回归

还是以预测房子价格为例，房子的属性除了其面积大小外，这里还有房子楼层数、房子年龄、厅室数等。也就是说一个房子的特征将不只一个，而是多个。

这里用 n 表示特征数目，$x^{(i)}$ 表示的是训练集中第 i 个例子的特征向量，$x_j^{(i)}$ 表示第 i 个例子的第 j 个特征值。这是目标函数变为：$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$，一般说来，会为每个记录增加属性 $x_0 = 1$。

在进行多变量线性回归中，对目标函数利用梯度下降进行参数求解的过程为：

1. 判断是否收敛，收敛则终止；否则，转到 2。

2. 更新参数。对 n 进行同步更新，$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \bullet x_j^{(i)}$。

2.1 利用梯度下降求解

属性的 scaling

对属性进行 scaling 的目标是让属性的范围基本类似，而不会存在数量级差别过多的情况。Scaling 的好处是让图像变得好看，而且也利于数据处理，如果数据范围过大不仅画图难看而且占用较多存储。

几种典型的数据 scaling 方法：

（1） min-max normalization

min-max 标准化也成为离差标准化，是对原始数据的线性变换，使得结果落在[0,1]区间，转换函数如下：$x^* = (x - \min)/(\max - \min)$。其中 max 和 min 分别为样本中数据的最大值和最小值，这种方法的缺陷就是当有新数据加入时，可能导致 max 和 min 发生变化，需要重新定义。

（2） log 函数转换

变换函数如下：$x^* = \log_{10}(x) / \log_{10}(\max)$。这里，max 为样本数据最大值，而且所有的数据都需要大于等于 1。

（3）z-score normalization

也叫标准差标准化，经过处理的数据符合标准正态分布，即均值为 0，标准差为 1，转换函数为：$x^* = \frac{x - \mu}{\delta}$。其中，$\mu$ 为所有样本的均值，δ 为所有样本数据的标准差。

学习速率α

和单变量回归一样，学习速率α的取值对收敛速度和最优解的求解都会有影响，在单变量的参数求解中，只是说到α要按照 90%的比率进行下降，但是没有说如何进行α的初始化，需要进行一定的调试，Andrew 这里给了他的建议，α的初始值尝试为：0.001,0.003,0.01,0.03…，即以 3 倍的方式进行递增。

2.2 利用归一化方程(Normal Equation)求解

除了利用梯度下降进行参数求解外，还可以利用 Normal Equation 进行参数求解。这里训练集中有 m 个记录数$(x^{(1)}, y^{(1)}), \ldots (x^{(m)}, y^{(m)})$，每个记录有 n 个特征属性。对任意$x^{(i)}$进行补全，转换入下：

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1}$$

即在第一行上加入$x_0^{(i)} = 1$。

矩阵 X 定义为 $\begin{bmatrix} \text{—} (x^{(1)})^T \text{—} \\ \text{—} (x^{(2)})^T \text{—} \\ \vdots \\ \text{—} (x^{(m)})^T \text{—} \end{bmatrix}$，Y 定义为 $\begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$。则参数矩阵$\theta = (X^T X)^{-1} X^T Y$。

梯度下降需要 scaling、多次迭代还有预设学习速率，但是 Normal Equation 就不用考虑那么多东东，但是 Normal Euaqtion 虽然简单方便但是当 Features 的数量即 n 较大时，会很慢，所以其 scale 没有梯度下降方法好。

3 逻辑斯谛回归(Logistic Regression)

除了进行连续值的预测，还有对离散值的预测，一个典型的应用或者场景就是分类，这里首先考虑简单的二元分类（之后会提到更复杂的多元分类）。在现实应用的二元分类有：

垃圾/非垃圾邮件的分类、良性/恶性肿瘤等。可以认为这里的 y 值范围为{0,1}。

可以粗略对 $h_\theta(x)$ 进行阀值设定，如果 $h_\theta(x) \geq 0.5$ 则预测 y=1；如果 $h_\theta(x) \prec 0.5$ 则预测 y=0。

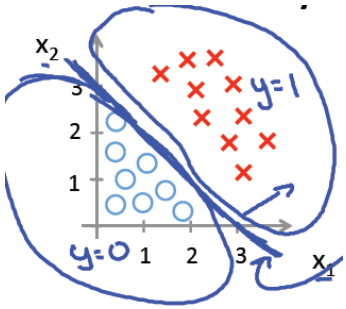
3.1 预测函数的表示

逻辑斯谛回归中 $h_\theta(x)$ 的范围区间是 [0,1]，这里用到 Sigmoid 函数表示为：

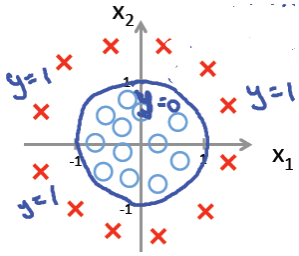
$g(z) = \frac{1}{1+e^{-z}}$，预测函数表示为：$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$。

这里根据不同的 $\theta^T x$，可以认为有不同的边界情况（Decision Bound）。例如：

1. 线性边界。例如 $\theta^T x = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$ 且 $\theta_0 = -3, \theta_1 = 1, \theta_2 = 1$，这样判定 y 值的边界就是一条直线，如下图所示。



2. 非线性边界。例如 $\theta^T x = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$ 且 $\theta_0 = -1, \theta_1 = 0, \theta_2 = 0, \theta_3 = 1, \theta_4 = 1$，这样进行判定时如果 $-1 + x_1^2 + x_2^2 \geq 0$ 则 y=1。如下图所示。



3.2 损耗函数的表示

逻辑斯谛回归的损耗函数为

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))]$$

。

在利用梯度下降方法进行参数求解时：

1. 判断是否收敛，收敛则终止；否则，转到 2。

2. 更新参数。对 n 进行同步更新，$\theta_j = \theta_j - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \bullet x_j^{(i)}$。

此处的 $h_\theta(x^{(i)}) = 1/(1 + e^{-\theta^T x^{(i)}})$。

3.3 多元分类

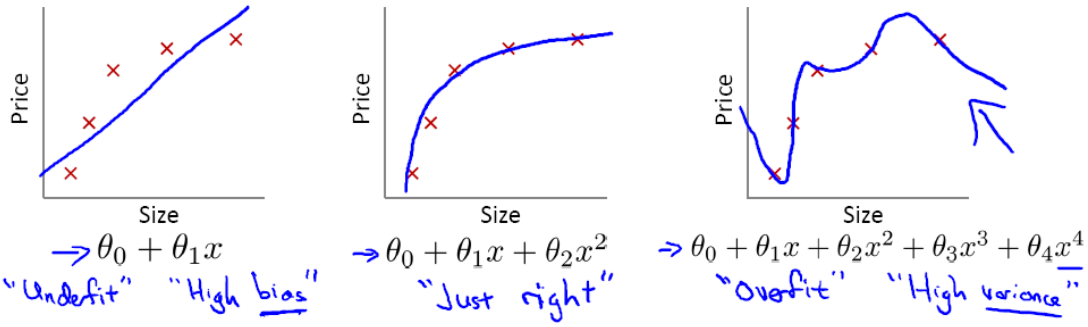
当分类场景变为多分类，即不再是二元分类时，对类别 i 建立一个分类器 $h_\theta^{(i)}(x)$ 来预测 y=i 的概率。对于任意输入 $x^{(i)}$ 对其分类时选择使得 $h_\theta^{(i)}(x)$ 最大化的 i。

4 正则化

4.1 过拟合问题

过拟合问题描述：当有较多的属性时，通过学习获得的假设函数可以较好的拟合训练集，但是却在通用性方面大大降低而无法较好的预测新实例。

例如在基于房子大小的房价预测中（线性回归）。如下图所示。



第一幅图的曲线没有较好拟合训练集数据，称为欠拟合；第二幅图不仅仅可以较好拟合训练集也能较好拟合测试集；第三幅图对训练集拟合的程度比第二个还好，但是此时却无法较好拟合测试集数据，称为过拟合。

解决过拟合的方法有：

1 减少特征数目：手动选择特征、选择合适的模型

2 正则化。采用正则化方法时，保留了所有的属性，但是却降低了各个属性的贡献值即可以认为是其 θ_j 减少了

4.2 采用正则化时的损耗函数

采用正则化来解决过拟合问题的思想就是：将相应的参数值变小。具体例子如下。

例如，在之前的基于房间大小来预测房价例子中，正则化的线性回归损耗函数为：

$J(\theta_0,\theta_1) = 1/2m(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2)$。这里$\lambda$值的设定也非常重要，如果其值过大，会导致无法解决过拟合问题、会产生欠拟合问题和训练过程可能无法收敛。

4.3 正则化举例

（1） 正则化的线性回归

首先，正则化的损耗函数为$J(\theta_0,\theta_1) = 1/2m(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2)$。基于此损耗函数，进行参数求解时，利用梯度下降方法，参数更新过程如下：

Repeat{

$$\theta_0 = \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

$$\theta_j = \theta_j(1 - \alpha\frac{\lambda}{m}) - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}) \bullet x^{(i)}(j = 1,2,\ldots,n)$$

}

如果利用正则方程进行参数的求解，则相应的算式为

If $\lambda > 0$,

$$\theta = \left(X^TX + \lambda\begin{bmatrix}0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1\end{bmatrix}\right)^{-1}X^Ty$$

（2） 正则化的逻辑斯谛回归

正则化逻辑斯谛回归的损耗函数为：

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

基于梯度下降的方法，进行参数的求解。

Repeat{

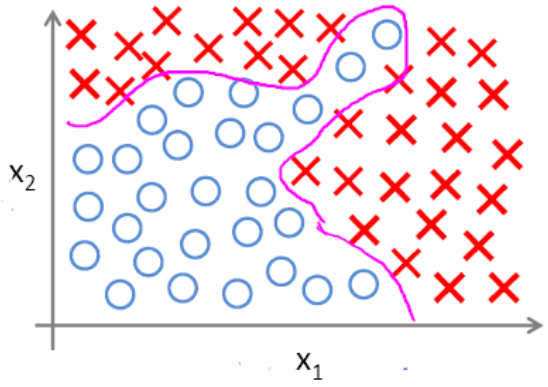
$$\theta_0 = \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

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

}

5 非线性假设之神经网络

逻辑斯谛回归也是一种非线性回归，如下图所示。其假设函数为 $h_\theta(x) = g(\theta^T x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \ldots)$。



当只有两个特征时，逻辑斯谛回归如上曲线所示；当特征较多，例如达到 100 个时，其预测（假设）函数中多项式个数大约有奖金 5000 个，即 $O(n^2)$。

特征和多项式数量的增长大大增加了算法的复杂性，但是如果减少特征的数量，就可能带来欠拟合的问题，那么就需要在不减少特征数量情况下注意效率。

神经网络孕育而生。

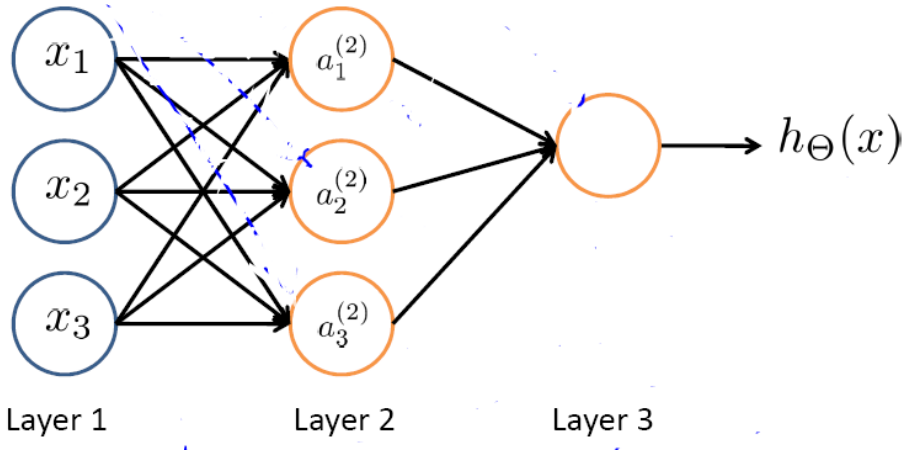
5.1 神经网络中的学习

当人耳听到声音时，大脑皮层的某个部位负责处理，并能不断学习和改进。

现代科学表明，通过一定的刺激来增强大脑的学习，使得人具备处理在处理某方面的能力增强（eg：方向感）。

5.2 模型解释

如下图所示是一个简单的神经网络模型。其中这里的 Layer1 是输入层，Layer3 是输出层，在输入层和输出层之间的是隐层，图中只有一个隐层 Layer2。一般由 j 层向 j+1 层进行特征传递和相关计算时，会给 j 层添加一个特征值，例如在计算 Layer2 中参数时，会给 Layer1 中的特征添加 x_0，一般地会将 x_0 设为 1。



这里 $a_i^{(j)}$ 表示的是第 j 层第 i 个单元的值，$\Theta^{(j)}$ 表示的是从 j 层向 $j+1$ 层进行映射时的参数。

图中 Layer1 到 Layer2 之间进行映射的关系如下：

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

图中最后输出为：

$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{12}^{(2)}a_3^{(2)})$$

这里用 s_j 表示第 j 层的单元个数，s_{j+1} 表示第 $j+1$ 层单元个数，那么 $\Theta^{(j)}$ 的维度为 $s_{j+1} * (s_j + 1)$。

一般，为了更好的进行相关计算，这里我们把该过程向量化：

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = a^{(1)}$$

$$z^{(2)} = \Theta^{(1)}x = \Theta^{(1)}a^{(1)}$$

$$a^{(2)} = g(z^{(2)})$$

Add $a_0^{(2)} = 1$ to $a^{(2)}$

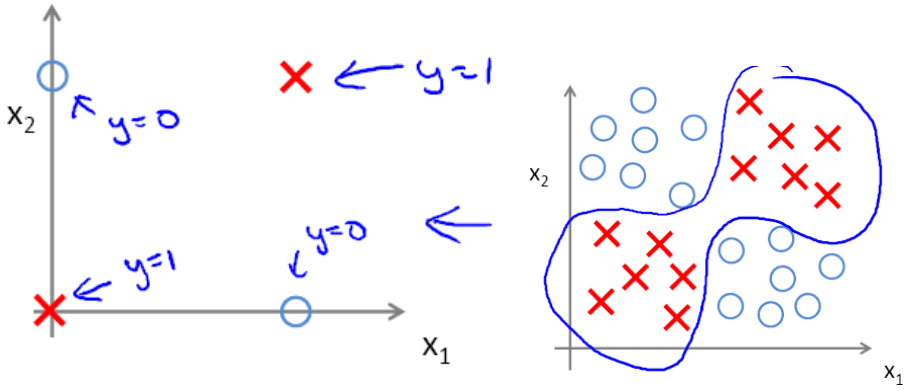
$$z^{(3)} = \Theta^{(2)}a^{(2)}$$

$$h_\Theta(x) = a_1^{(3)} = g(z^{(3)})$$

5.3 模型示例

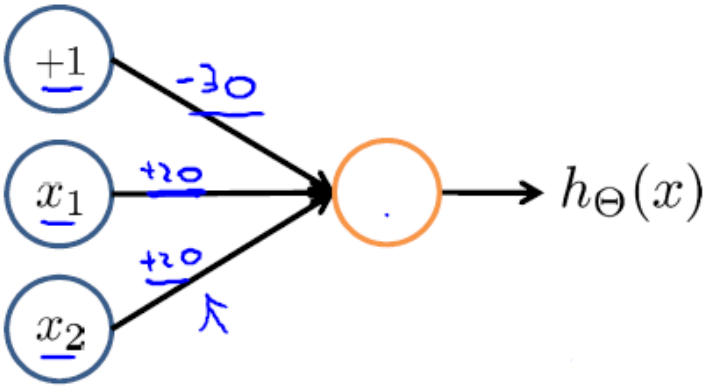
异或

可以将神经网络模型用于非线性分类：异或和异或非。这里以异或为例，$y = x_1 XOR x_2 (x_1 XNOR x_2)$。

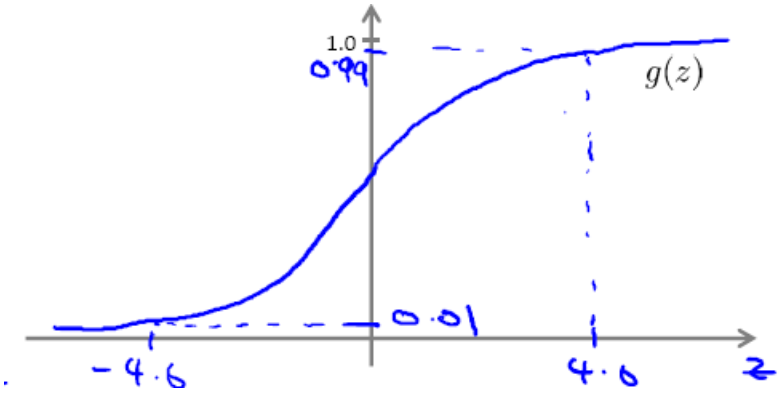


与

$x_1, x_2 \in \{0,1\}$
$y = x_1 AND x_2$ 可以用如下的神经网络进行表示。$h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$。



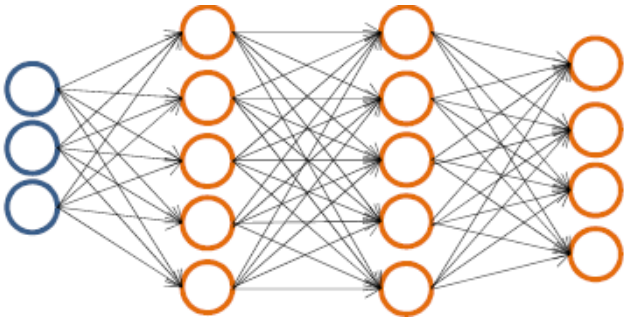
其中，对于 g（z）的曲线，如下图所示。



多分类

例子描述：基于对图像的特征提出，通过神经网络模型参数训练，来识别图像中的目标是行人(Pedestrian)、小汽车(Car)、摩托车(Motocycle)还是卡车(Truck)。

这里的假设函数 $h_\Theta(x) \in \Re^4$ 即该假设函数是四维的。利用神经网络进行相关参数训练的过程可以表示如下：



$$h_\Theta(x) \in \mathbb{R}^4$$

当 $h_\Theta(x) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ 时，判断该图像中的目标是行人；当 $h_\Theta(x) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ 时，判断图像中的目标是小汽车。等等，一次类推。

6 机器学习实践指导

6.1 调试学习算法

假设利用正则化的线性回归来预测房子价格。我们的目标函数如下：

$$J(\theta_0, \theta_1) = 1/2m(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2)$$

在进行机器学习时，需要进行一些诊断，从而来改进算法的性能。

假设评估

为了对假设进行评估，一般会将数据集分为两个部分：一个是训练集(Training Set)，另外一个是测试集(Test Set)（一般比例是 7:3）。

为了对假设进行评估，一般分为基于训练数据的参数学习和测试集错误计算两个步骤。针对线性回归中的训练和测试来说，测试集的错误为：

$$J_{test}(\theta) = 1/2m_{test}\sum_{i=1}^{m_{test}}(h_\theta(x^{(i)}) - y^{(i)})^2$$

对于逻辑斯谛回归的分类来说，首先从训练集中学习参数，然后计算测试集的错误：

$$J_{test}(\theta) = -\frac{1}{m_{test}}[\sum_{i=1}^{m_{test}} y_{test}^{(i)} \log h_\theta(x_{test}^{(i)}) + (1 - y_{test}^{(i)}) \log(1 - h_\theta(x_{test}^{(i)}))]$$

；最后计算误分类率。

模型选择及训练集、校验集和测试集

利用训练集中数据基于模型进行参数训练时，会最小化训练错误 $J_{train}(\theta)$，但是该模型及参数不一定适合其它新的数据记录。针对新的记录用相关假设进行预测时，其错误比 train 的错误大。

我们可以选择的模型很多，例如：

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

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

$h_\theta(x) = \theta_0 + \theta_1 x + \ldots + \theta_3 x^3$

……

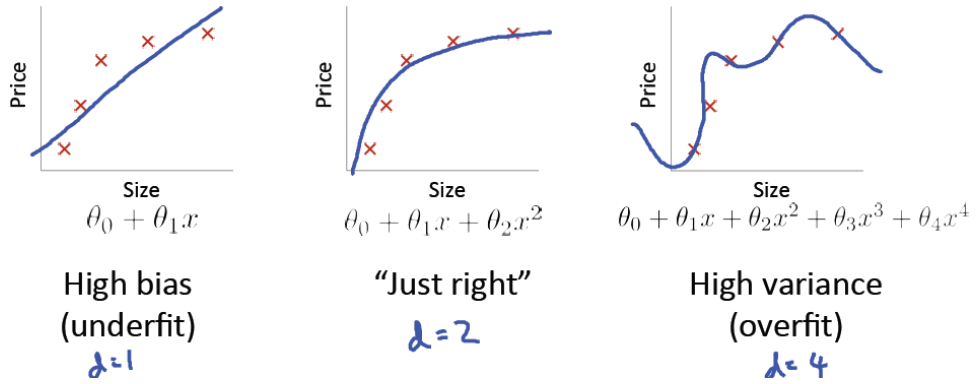
在进行模型选择时，不能依靠其 $J_{train}(\theta)$ 最小，而选择相应的模型，因为该模型并不能保证对 test 集的错误也较低。

为了进行理性的模型选择，可以将数据集划分为 train 训练集、validate 验证集合和 test 测试集合，比例可以选择为 6:2:2（说明：在机器学习中，往往因为训练集不够多而导致无法得到满意结果，所以相关选择要考虑实际数据量）。这样我们就可以通过 validate 验证集的错误来预测对 test 的错误，进行相关的模型选择（选择使得 validate 错误较小的模型）。

对偏倚(Bias)和方差(Variance)的诊断

d 表示假设中多项式次数（多相似的次数等于最大的单项式的次数）。如下图，现实了

在对房价进行预测时，不同d下的拟合情况。



High bias (underfit)

d=1

"Just right"

d=2

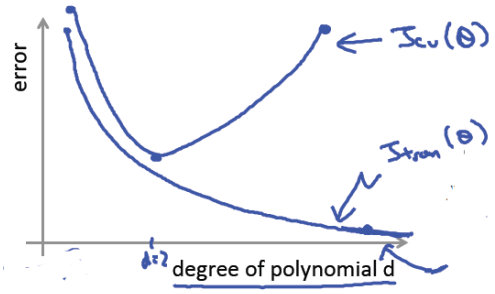
High variance (overfit)

d=4

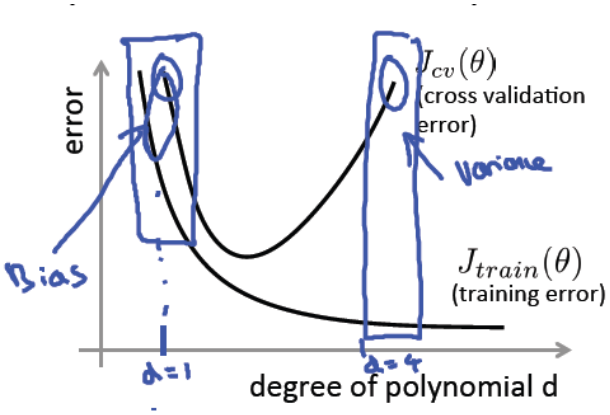
为了较好的判别偏倚和方差，首先表示出训练错误和交叉验证错误。训练错误表示如下：$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$；交叉验证错误表示如下：

$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$。

一般说来，训练错误、交叉验证错误和d之间的变化关系如下图所示。

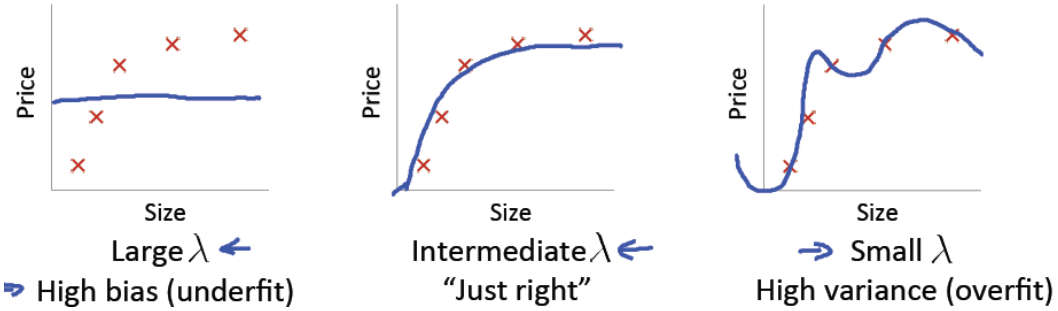


那么，当学习算法不能较好的满足需求时，如何判断到底是偏倚还是方差问题呢。如果是偏倚，通常$J_{train}(\theta)$较大且$J_{train}(\theta) \approx J_{cv}(\theta)$；如果是方差，通常$J_{train}(\theta)$较小且$J_{cv}(\theta) \succ J_{train}(\theta)$。判别如下图所示：



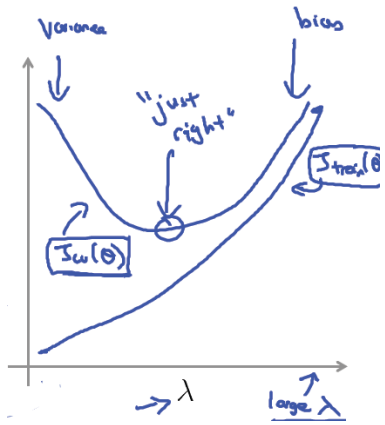
正则化和偏倚/方差

这里我们以线性回归模型为例，假设为 $h_\theta(x) = \theta_0 + \theta_1 x + \theta_3 x^3 + \theta_4 x^4$，采用正则化的损耗函数为 $J(\theta) = 1/2m(\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2)$。那么 λ 和偏倚/方差的关系如何呢？先让我们看下面一组图。



如上图所示，当 λ 过大时，可以看到参数都基本为 0，这样必然导致偏倚较大；而当 λ 较小，即参数的求解主要考虑的训练集，那么就会导致方差问题。

那么如何选择合适的 λ？一种合适的选择方式就是，选择使得 test 错误或交叉错误最小的 λ 值。test 错误和交叉错误与 λ 之间的变量关系如下图所示。

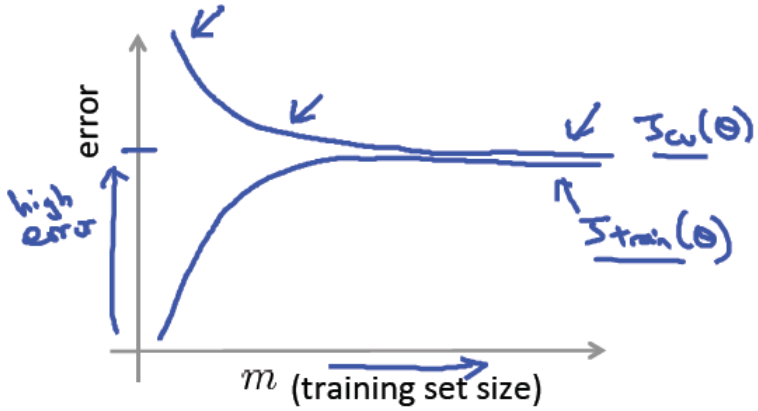


学习曲线

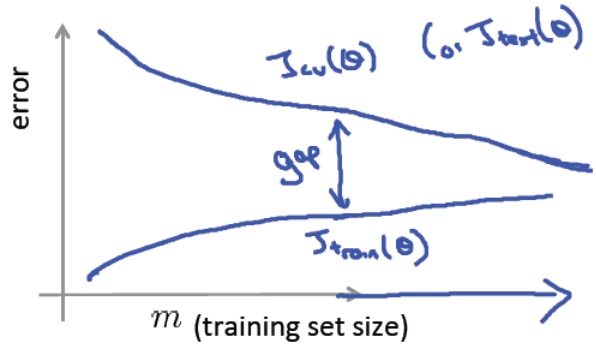
为了较好的掌控学习算法运行的正确性，对学习曲线进行绘制是一个不错的选择，该方法是一种避免学习算法偏倚、方差或两者都有的方法。对较少数据集的实验，就知道到底是偏倚、方差或者两者都有，所以需要通过刻画训练错误（$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$）、验证错误（$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$）和训练集中记录数目 m 之间的变量关系。

如果是高偏倚的话，三者之间的变化关系如下图所示。两者都非常高，增加 m 都无法解

决该问题。



如果是高方差的话，三者之间的关系如下图所示。$J_{cv}(\theta) \succ J_{train}(\theta)$，但是增加 m 却可以进行一些改进。



6.2 调试建议

措施	效果/解决的问题
获取更多的训练记录	高方差
尝试更小的特征数	高方差
尝试添加新的特征	高偏倚
降低 λ	高偏倚
增加 λ	高方差

6.3 基于机器学习的实际系统设计

这里以开发一款辨别垃圾邮件的分类器为例。这里建立垃圾邮件分类器，基于监督学习（在训练集中给出了该邮件是否是垃圾邮件）。需要基于邮件的特征 x 来判别 y =spam(1) 还是 not spam（0）。

这里的 feature 是固定的 100 个单词。其中对邮件进行词语抽取的时候，在实际中，通常提取最经常出现的单词。此外，为了更好的建立分类器，对**特征抽取方面**如下实际建议：

- 有较多的训练数据
- 提取邮件的路由信息作为邮件特征

• 提取出消息中的复杂特征，例如 discount 和 discounts 是否等同处理？deal 和 dealer 是否等同处理等

• 可以发现相应的拼写错误

在对算法进行误差分析时，建议如下：

• 快速的算法实现，开始不用太复杂

• 学习曲线的绘制，从而决定是否需要更多的数据或特征等

• 错误分析：手动检测

（word stemming 可以减少错误率，word stemming 是词汇形态转换，例如 counts、counter 都转换为 count）。

在应对 skewed class 时候

其实，大家平时遇到的垃圾邮件并不多，可能只占 1%甚至更少，这样对整体的 error 根本影响不大，从而通过训练集、验证集或测试集的错误是不能加好评估模型算法的有效性的，这个时候使用的算法评估参数一般改为 precision 和 recall。

针对某个分类，实际正确的集合为 B，分类的结果为 A，A 和 B 的交集为 C，则 precision=C/A，recall=C/B。需要对 precision 和 recall 做一个平衡，从而有一个更新的评价参数为：

$$f-score = \frac{2 * precision * recall}{precision + recall}$$

7 支持向量机（SVM-support vector machine）

7.1 优化目标

这里首先回顾一下逻辑斯谛回归，其假设（Hypothesis）是 $h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$。

如果 y=1，我们希望 $h_\theta(x) \approx 1, \theta^T x \succ 0$；如果 y=1，我们希望 $h_\theta(x) \approx 0, \theta^T x \prec 0$。

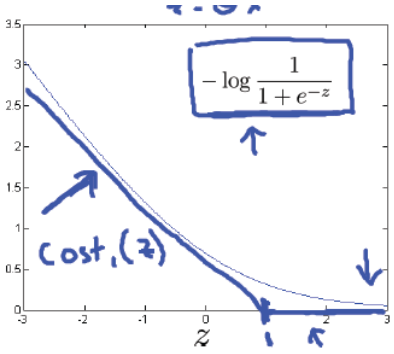
逻辑斯谛回归的损耗函数为

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))]$$

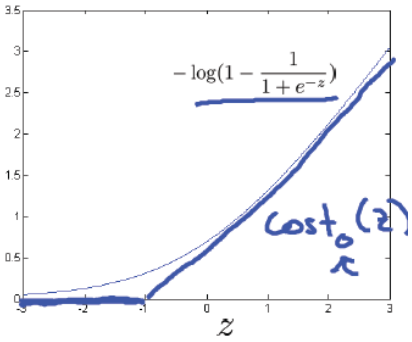
$$= 1/m \sum_{i=1}^{m} [-y \log \frac{1}{1+e^{-\theta^T x^{(i)}}} - (1-y) \log(1 - \frac{1}{1+e^{-\theta^T x^{(i)}}})]$$

。

要使得损耗函数小，那么当 y=1 的时候，我们希望 $-\log \frac{1}{1+e^{-\theta^T x}}$ 较小即 $\theta^T x \succ 0$，如下图所示（说明：蓝色曲线为近似曲线，可以相近的认为，之后会用到）。



当 y=0 的时候，我们希望$-\log(1-\frac{1}{1+e^{-\theta^T x}})$较小即$\theta^T x \prec 0$，如下图所示。

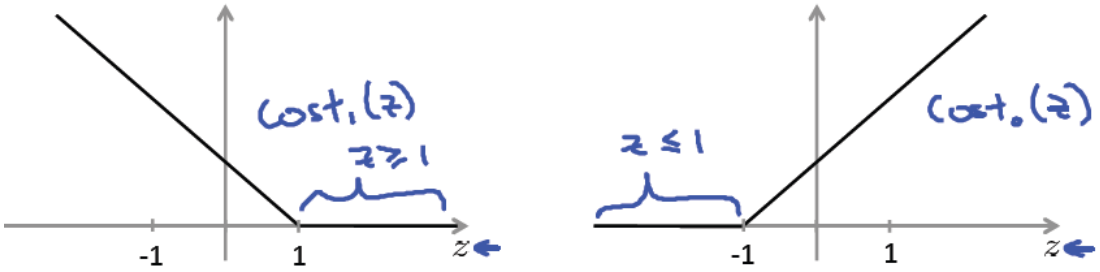


这里我们用$\cos t_1(z) = \cos t_1(\theta^T x) = -\log\frac{1}{1+e^{-\theta^T x}}$，用$\cos t_0(z) = \cos t_0(\theta^T x) = -\log(1-\frac{1}{1+e^{-\theta^T x}})$。SVM 的损耗/目标函数和逻辑斯谛较为相近，可以表示为：

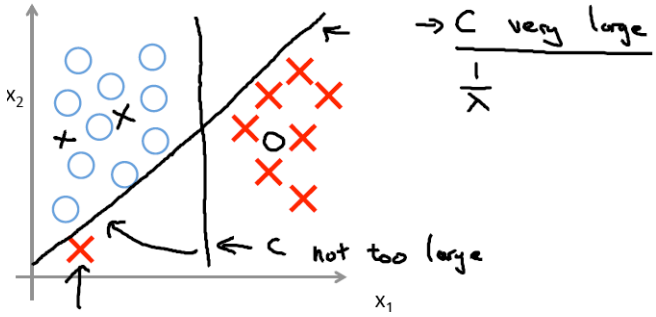
$$\min_\theta C\sum_{i=1}^{m}[y^{(i)}\cos t_1(\theta^T x^{(i)}) + (1-y^{(i)})\cos t_0(\theta^T x^{(i)})] + 1/2\sum_{j=1}^{n}\theta_j^2$$

7.2 SVM 中的间隔(Margin)

在上一小节中，我们提出了 SVM 的目标损耗函数$\min_\theta C\sum_{i=1}^{m}[y^{(i)}\cos t_1(\theta^T x^{(i)}) + (1-y^{(i)})\cos t_0(\theta^T x^{(i)})] + 1/2\sum_{j=1}^{n}\theta_j^2$。这里采用近似的函数曲线，如下图所示。



为了使得损耗函数最小，那么当 y=1 时，需要$\theta^T x \geq 1$而不是≥0；当 y=0 时，需要$\theta^T x \leq -1$而不是＜0。可以看到 SVM 的条件加强了，其核心思想是在两个类别之间建立一个分割线，该分割线离两个分类尽量远，当然这个远近距离会影响分类结果。如下图所示。



7.2 核函数

在很多情况下，我们会碰到非线性的分割边界，而不是之前的线性。例逻辑斯谛回归，假设为$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2$，那么对于多项式中的四个特征是否有更好的选择呢？

可选择的方法就是确定若干个 landmark，然后计算各个记录或点与这些 landmark 的距离作为该记录或点的新特征（feature）。这里举个简单的例子进行说明。

将上面的假设修改为$h_\theta(x) = \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \theta_4 f_4$，那么我们这里将重新定义新的 Features 即f_1、f_2、f_3和f_4。第一步就是选择 landmark，注意 landmark 和之前的记录或点的维度是一样的，这样才能进行相关的比较，例如我们选择了$l^{(1)}$、$l^{(2)}$、$l^{(3)}$和$l^{(4)}$作为 landmark；接着，对于任意的 x，计算其新的特征（这里我们以高斯内核为例），计算公式如下：

$$f_i = similairty(x, l^{(i)}) = k(x, l^{(i)}) = \exp\left(-\frac{\left\| x - l^{(i)} \right\|^2}{2\delta^2}\right)$$

在进行核函数选择的时候，对于给定的数据集$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$，**相关步骤**如下：

选择 landmark。设置 $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$；

对任意记录的特征 $x^{(i)}$，进行特征重新计算。$f^{(i)} = \begin{bmatrix} f_1^{(i)} = sim(x^{(i)}, l^{(1)}) \\ f_2^{(i)} = sim(x^{(i)}, l^{(2)}) \\ . \\ . \\ f_m^{(i)} = sim(x^{(i)}, l^{(m)}) \end{bmatrix}$（相似度如何计算和选择的核函数有关）；

重新建立 training 集的特征后，就能用 svm 的损耗函数来训练参数了，此时的损耗函数可以表示为：

$$\min_{\theta} C\sum_{i=1}^{m} y^{(i)} \cos t_1(\theta^T f^{(i)}) + (1 - y^{(i)}) \cos t_0(\theta^T f^{(i)}) + 1/2\sum_{j=1}^{n=m} \theta_j^2$$

SVM 的相关参数设定

参数名称	参数调整
C（=$1/\lambda$）	C 较大：lower 偏倚，high 方差
	C 较小：higher 偏倚，low 方差
δ^2	较大：特征比较平滑；higher 偏倚，lower 方差
	较小：特征相对不平滑；lower 偏倚，higher 方差

模型的选择

这里设定 n 为特征数目且 $n \in \Re^{n+1}$， m 为训练集中的记录个数。

情况	适用算法
n 较大（相对于 m 来说）	使用逻辑斯谛回归或者 SVM（不适用核函数）
n 较小，m 适中	利用 SVM+高斯核函数
n 较小，m 较大	增加新的特征，然后利用逻辑斯谛或 SVM（不用核函数）
说明：神经网络适合上面所有情况，but 其 slow to train	

8 聚类

这里我们将接触新的一类模型，那就是无监督学习模型，聚类问题是典型的无监督学习例子。在监督学习中，测试集的每一条记录或例子都有 x 和 y；但是无监督学习中，测试集的组成将只是特征 $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$。聚类应用广泛，从社交网络圈子发现、计算机集群组织到商业群体分类等。

8.1 经典的 k-means

k-means 算法的输入：K -类别的数目；训练集 $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$。这里 $x^{(i)} \in \Re^n$。

算法的过程：

- 随机选取 K 个类别的中心点 $\mu_1, \mu_2, \ldots, \mu_K \in \Re^n$
- 重复{

  For $i = 1$ to m：

  计算 $x^{(i)}$ 到各个类中心的距离，将距离最近的类别赋给 $x^{(i)}$ 即 $c^{(i)}$

  For $k = 1 to K$：

  重新计算类别的中心质点 μ_k

}

这里用 $\mu_{c^{(i)}}$ 表示 $x^{(i)}$ 所在类别 $c^{(i)}$ 的中心质点特征。那么，我们分类的目标优化函数就是：

$$\min_{\substack{c^{(1)}\ldots c^{(m)} \\ \mu_1 \ldots \mu_K}} J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \min_{\substack{c^{(1)}\ldots c^{(m)} \\ \mu_1 \ldots \mu_K}} \frac{1}{m} \sum_{i=1}^{m} \left\| x^{(i)} - \mu_{c^{(i)}} \right\|^2$$

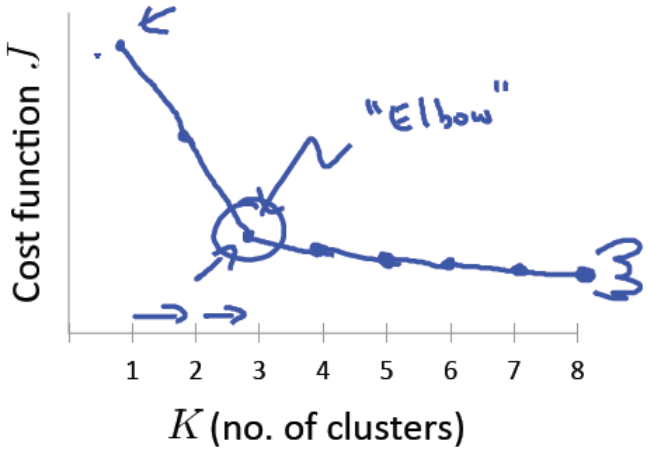
中心点的随机初始化

一般说来，K 一定是小于 m 的，从训练集中随机的选择 K 个记录作为相应的中心质点。为了更好的选择中心质点，可以重复进行{random；利用 k-means 进行聚类（两步：类别分配+类中心调整）}，然后选择其中使得 J 最小的情况作为最后结果。

类别数 K 的选择

上面说到了中心质点的选择，K 也是需要进行事先定义的。

方法之一：实验不同的 K 值，选择使得 J 拐点的情况。一般两者的变量关系如下图所示。



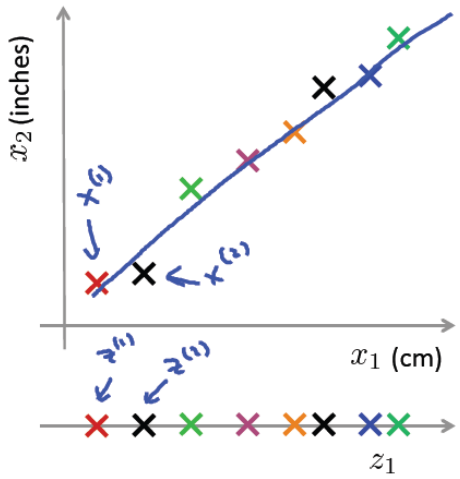
方法之二：根据聚类的目的来选择，看相应的 K 值在最终目的上的表现如何。例如你想对衣服尺码进行聚类，其实就那么几种，事先心里是有数的。

8.2 降维

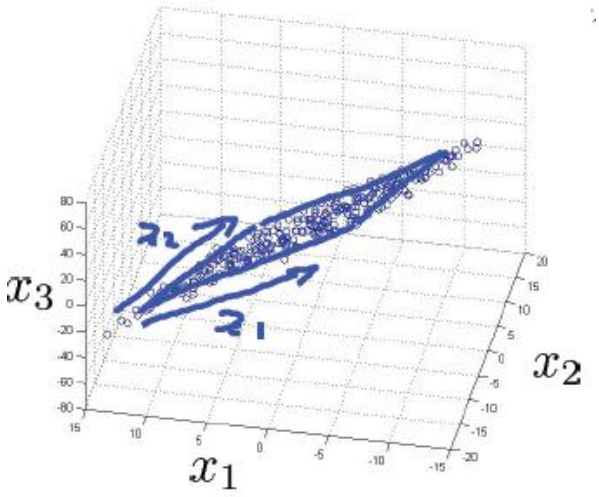
数据降维的驱动力

1.数据压缩

如下图所示，将可以将二维的数据降维为一维的数据。(看到这里，有人可能认为此时的 PCA 好像是在做 regression，其实两者有本质区别，regression 和 PCA 的目标函数是不一样的，这里以二维为例，PCA 是节点到直线距离最小，regression 则是相同横坐标下的纵轴距离)



如下图所示，可以将三维的数据降为二维的数据。



2.数据可视化

如果对某一类事物的特征超过 3 个，即 3D 都无法展示了，就需要对其特征进行降维以便展示。

降维方法之 PCA

本小节将讲述降维方法之一 PCA(Principal Component Analysis)。这里假设训练集合为

$\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$，将每个记录的特征降到 k 维。

第一步：数据预处理

预处理包含两个部分：属性范围调整和均值归一化。

首先是均值归一化。计算每个属性的均值 $\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$，然后对任意记录的相应属性进行调整 $x_j^{(i)} = x_j^{(i)} - \mu_j$。

其次是属性范围调整。例如在之前的房价预测中，有属性房子的空间大小和房子的房间数，空间大小可能达到几百，而房间数通常在十以内，所以需要进行一些调整，例如 $x_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{s_j}$。

第二步：PCA 过程

- 计算方差矩阵 $\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T$；

- 利用 SVD 即矩阵分解对方差矩阵进行处理。$[U, S, V] = svd(\Sigma)$（U 为对角矩阵，S 与 V 正交）；

- $U_1 = U(:, 1:k)$（即截取前 k 个，维度变为 k）；

- $z = U_1^{'} \bullet x$（$U_1^{'}$ 是 U_1 的转置矩阵）。

选择合适的 k

在进行降维后，可以通过新得数据对原始数据进行还原，当然中间存在损耗，还原数据为：

$$x_{approx} = U_1 \bullet z$$

平均错误方差为：$\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - x_{approx}^{(i)}\right\|^2$

数据总的方差为：$\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}\right\|^2$

一般地，选择合适的 k 使得：

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - x_{approx}^{(i)}\right\|^2}{\frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)}\right\|^2} \leq 0.01$$

9 异常检测

场景描述：每一架飞机都需要一些检测来保证其在实际使用中的安全性，那么其中检测的因素非常之多，例如发动机的温度、震动强度等，需要根据实际检测的数据来保证该飞机出现故障的概率低于一个特定的值。

此外在数据中心的机器，监控其运行参数例如内存大小、硬盘大小、CPU 负载、网络带宽等来预测该机器出现故障的概率，以保证该概率值小于一定阀值。

9.1 算法描述

• 选择合适的特征用于进行异常诊断，形成训练集$\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$（说明：在选择合适的特征时，一个是注意特征之间的独立性，另外一个是可能需要一些预处理使得特征分布呈现正态性）

• 对于每个特征x_j，找出其分布参数：

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$$

$$\delta_j^2 = \frac{1}{m}\sum_{i=1}^{m} (x_j^{(i)} - \mu_j)^2$$

• 对于一个新的记录或特征x，计算$p(x)$：

$$p(x) = \prod_{j=1}^{n} p(x_j; \mu_j, \delta_j^2) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\delta_j} \exp(-\frac{(x_j - \mu_j)^2}{2\delta_j^2})$$

如果$p(x) \prec \varepsilon$，则诊断为异常。

9.2 实践指导

为了保证算法的可靠性，需要对其进行评估。这里我们拥有一些已经被标识过的数据，即知晓相应特征下是否异常。这里分为了训练集、验证集和测试集。

集合的划分

在总的集合中，异常出现的次数一般并不多。这里假设有 10000 个正常记录+20 个异常记录。集合划分可以为：训练集 6000 正常、验证集 2000 正常+10 异常、测试集 2000 正常+10 异常；训练集 6000 正常、验证集 4000 正常+10 异常、测试集 4000 正常+10 异常。

可以看到尽量保证训练中足够正常的例子，而验证集和测试集中均有异常点。

评价指标

由于异常出现的次数不多，即出现了 skewed class 情况，那么就可以之前说过的 precision、recall 和 F-score 进行评价。（可以通过这些指标，一方面是判定算法有效性，另一方面可用于选择参数ε）

9.3 异常发现 vs 监督学习

异常发现	监督学习
类别非常 skew，即 y=1 的情况非常至少而 y=1 的情况占绝大多数，例如异常点个数通常在 0-20 左右	y=0 和 y=1 的情况都较多
异常类别非常多，无法实现一一标识	
以后出现的特征可能和之前的特征相去甚远	以后个体出现的特征和之前的相似
例如：制造业中的安全检测、数据中心的监控	**例如：邮件分类、天气预测、癌症分类**

9.4 多变量高斯分布

在之前的例子中，我们有一个重要的假设即实例 x 的各个特征之间是独立的，这里我们考虑它们之间是相互联系的。

多变量高斯分布中的异常检测：

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

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

- 对于一个新的实例特征 x，计算概率

$$p(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(x-\mu)^{T}\Sigma^{-1}(x-\mu)\right)$$

如果 $p(x) \prec \varepsilon$，则诊断为异常。

说明：多变量高斯分布融入了各个特征变量之间的关联性，这导致其相比于原来的独立模型复杂度增加，此外，由于涉及到 Σ 的逆计算所以这里需要 $m \succ n$。

10 推荐系统

10.1 基于内容的推荐

这里我们有电影和用户，用户对自己观看过的电影会有的一个评分即 rating 而对自己尚未观看的电影则没有评分。推荐过程变成了基于历史来预测用户对尚未评分的电影的评分预测，从而来向用户进行相关推荐。

这里用 n_u 表示用户的个数，用 n_m 表示电影的个数，$r(i,j) = 1$ 如果用户 j 评论了电影

i，当用户评论了电影时用 $y^{(i,j)}$ 表示相应的评分，$\theta^{(i)}$ 表示用户 i 的参数，$x^{(i)}$ 表示电影 i 的特征向量，m^j 表示用户 j 评分的电影数目。对于用户 j 电影 i，根据 $(\theta^{(i)})^T(x^{(i)})$ 来预测评分。基于内容的推荐，就是我们基于电影的特征即其特征已知，来求解用户参数。

相关的优化目标函数

学习参数 $\theta^{(j)}$ 的目标函数为：

$$\min_{\theta^{(j)}} \frac{1}{2}\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

求解所有参数的目标函数为：

$$\min_{\theta^{(1)},\dots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

利用梯度下降进行相关参数的求解：

Gradient descent update:

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} \quad (\text{for } k=0)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\left(\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} + \lambda\theta_k^{(j)}\right) \quad (\text{for } k\neq 0)$$

10.2 协同过滤算法

目标函数：

$$J(x^{(1)},\dots,x^{(n_m)},\theta^{(1)},\dots,\theta^{(n_u)}) = \frac{1}{2}\sum_{(i,j):r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}(x_k^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

算法流程：

- 初始化 $x^{(1)},\dots,x^{(n_m)},\theta^{(1)},\dots,\theta^{(n_u)}$ 为较小的随机值
- 利用梯度下降使得目标函数最小化

$$\text{every } j=1,\dots,n_u, i=1,\dots,n_m:$$

$$x_k^{(i)} := x_k^{(i)} - \alpha\left(\sum_{j:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})\theta_k^{(j)} + \lambda x_k^{(i)}\right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha\left(\sum_{i:r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})x_k^{(i)} + \lambda\theta_k^{(j)}\right)$$

- 对于特定用户的 θ 和电影的 x，可进行评分预测