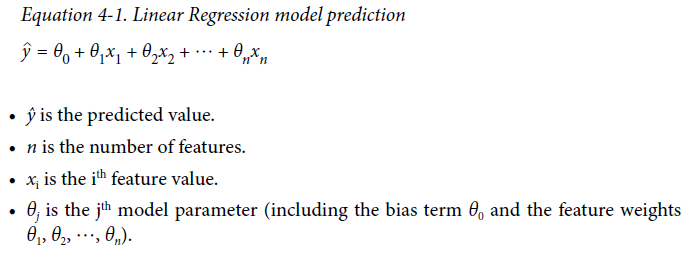
Training Model

A: direct ‘closed-form’直接通过解析解公式求解

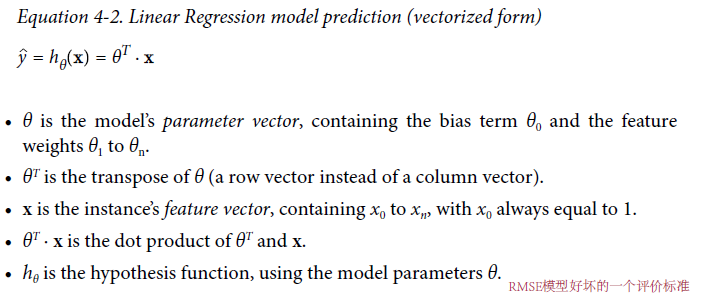
B: gradient descent通过迭代优化算法求解

# Linear Regression

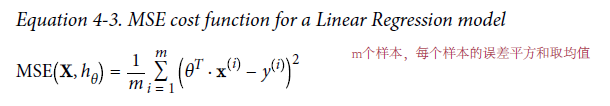
一般函数形式



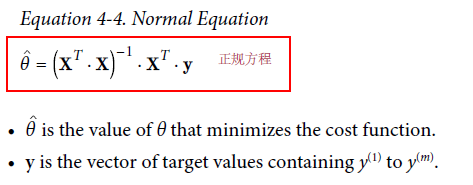
向量形式线性回归模型：



均方误差：



正规方程：



# Computational Complexity

Normal equation: 正规方程，当数据特征少时，计算很方便快速，但是当特征多时计算就非常的慢。

# Gradient Descent

## 3.1 Batch Gradient descent

利用代价函数对每个参数求导，得到梯度向量，结合学习率来逐步优化参数

缺点：使用整个训练集来计算每一步迭代的梯度

## 3.2 Stochastic Gradient Descent

随机选择数据集中的样本来计算梯度。

SGD算法每步通过在数据集中随机选择一个样本，并通过该样本计算梯度。这种方式计算量小，速度快。

但是由于随机选择样本，随着时间它会逐步向最优解靠近，但是由于随机性，在最优解附近仍然会波动

当cost function is very irregular, SGD算法能跳出局部最优解，从而找到全局最优解。但是难以找到最优解，因为它在最优解附近徘徊。解决这个问题的方法之一是：调整学习率，在迭代过程逐步减小学习率，从而最终收敛到最优解。

The steps start out large (which helps make quick progress and escape local minima), then get smaller and smaller, allowing the algorithm to settle at the global minimum. This process is called simulated annealing, because it resembles the process of annealing in metallurgy where molten metal is slowly cooled down.

**Learning Rate: 梯度的变化率**

**Simulated Annealing**: The steps start out large (which helps make quick progress and escape local minima), then get smaller and smaller, allowing the algorithm to settle at the global minimum.

**Learning schedual:** The function that determines the learning rate at each iteration

## 3.3 Min-batch Gradient Descent

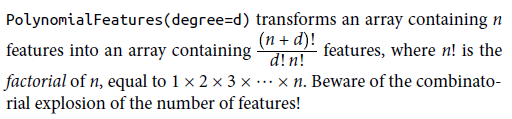
Min-batch Gradient Descent: at each step, instead of computing the gradients based on the full training set (as in Batch GD) or based on just one instance (as in Stochastic GD), Mini- batch GD computes the gradients on small random sets of instances

The main advantage of Mini-batch GD over Stochastic GD is that you can get a performance boost from hardware optimization of matrix operations, especially when using GPUs.(方便使用基于硬件的矩阵运算，如通过GPU计算)

# Polynomial regression

将变量的高次项也转化为一个特征，

当包含多个变量时的，不仅包含每个变量的多次项，还包含变量之间的组合项。



# Learning Curve

## 5.1 underfitting and overfitting

在多项式回归模型中，degree越高，越容易过拟合。

过拟合，欠拟合

如何判断模型时过拟合或者欠拟合？ 通过交叉验证或者学习曲线来初步判断。

通过交叉验证法：

过拟合：如果在训练集上，模型表现非常好，但是在交叉验证中表现很差，那么就是过拟合了。

欠拟合：如果在训练集上，以及交叉验证中都表现很差，那么就是欠拟合了。

通过学习曲线法：

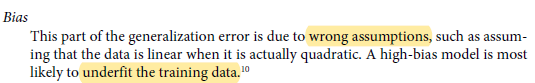
以训练集和验证集的误差关于训练集大小的函数作为学习曲线的。

If your model is underfitting the training data, adding more training examples will not help. You need to use a more complex model or come up with better features.

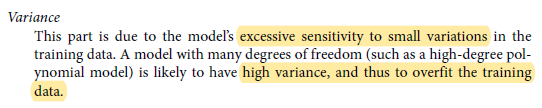
One way to improve an overfitting model is to feed it more training data until the validation error reaches the training error

## 5.2 Bias , Variance , Inrreducible error

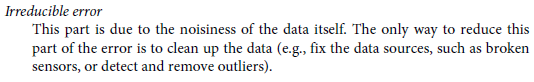
Bias (偏差)： 偏差很大，表明模型欠拟合



Variance(方差): 方差很大，表明过拟合



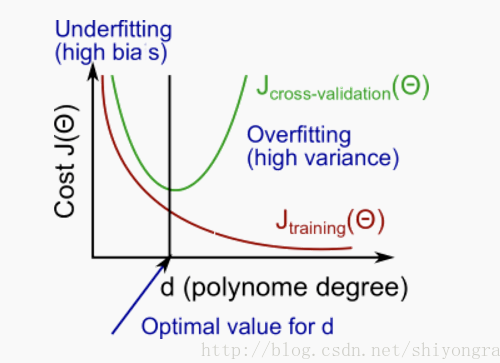
Inrreducible error: 这种误差主要源于噪声，错误数据等，要解决这个问题就需要提供干净的数据，清除异常值。



Increasing a model’s complexity will typically increase its variance and reduce its bias. Conversely, reducing a model’s complexity increases its bias and reduces its variance. This is why it is called a tradeoff.

增加模型复杂度，可以增大方法，减小偏差。

即当训练误差和验证误差近似相等且都比较大时，判断为高偏差、欠拟合（high bias、underfitting）   
当训练误差非常小，验证误差非常大且远远大于训练误差，判定为高方差、过拟合（high variance、overfitting）



模型复杂度对学习曲线的影响

# Regularized Linear models

多项式回归：过拟合问题可以通过减小自由度

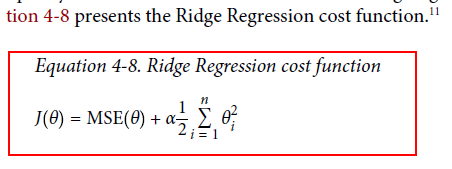
对于linear model ,regularizatfion 一般是通过约束模型权值实现的。

## 6.1 ridge Regression

脊回归：

降低w的权值减少过拟合。

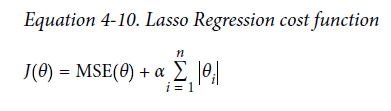
增加正则项（regularization term）到代价函数中，从而使得不仅拟合数据还保证权值很小。



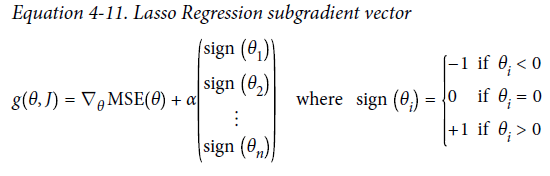
## 6.2 lasso regression

套索回归：

添加一范数来作为正则化项

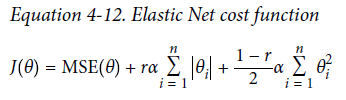


这种正则化方式的重要特点之一是：它趋向于完全消除（减少）最不重要的权值



## elastic net

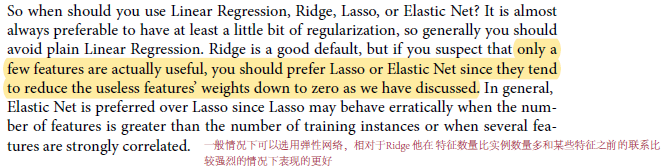
同时有一范数和二范数的正则项



## selection of the three regression method

在使用线性回归模型时，一般情况下都要加一个正则项。使用脊回归是一个很好的默认正则详细，但是如果，只有很少的特征时有用，那么应该采用套索回归或者弹性回归方法，从而去除那些无用的权值。

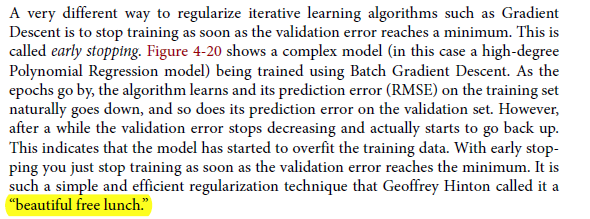
一般情况下，当特征数大于样本数，或者几个特征之间有很强的关联的话，弹性网络比套索回归表现更好的

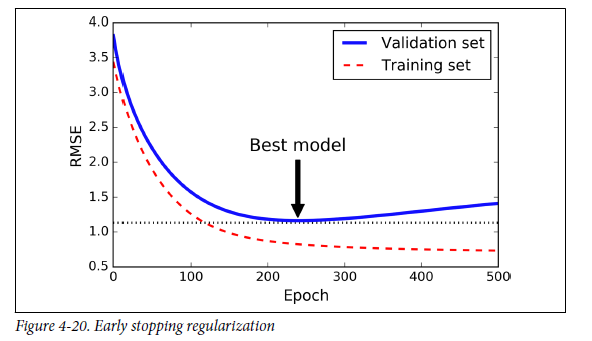


# early stopping

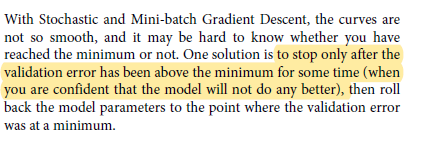
在迭代计算过程中，当误差达到最小是能尽快结束迭代。

对SGD, mini-batch GD, 由于随机性，因而始终存在在最优值附近波动，为了知道能否到达最优值。一种解决办法是：One solution is to stop only after the validation error has been above the minimum for some time (when you are confident that the model will not do any better), then roll back the model parameters to the point where the validation error was at a minimum.



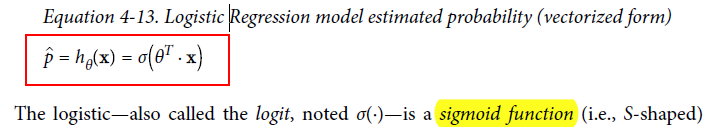


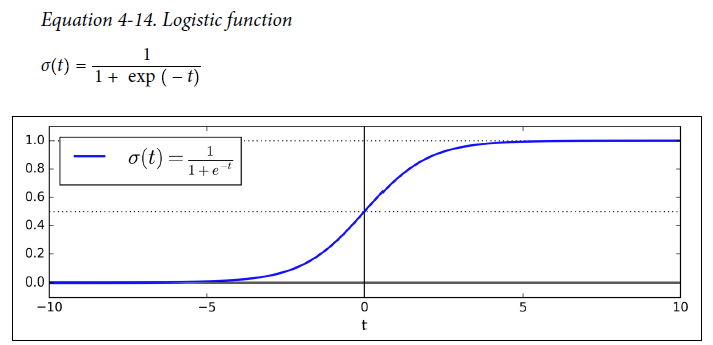
注意：



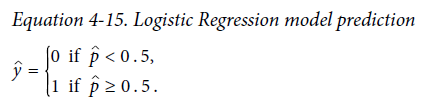
# logistic regression

## 8.1 Estimating Probabilities（模型）



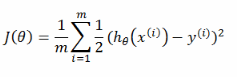


预测模型



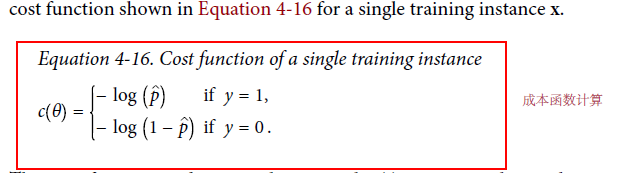
## 8.2 Training and Cost Function

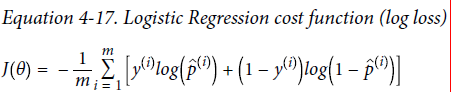
由于假设模型h(x) sigmoid 函数，所以直接使用均方误差作为代价函数的化，该代价函数是一个非凸函数。这会导致使用梯度下降法时收敛到局部最优值。



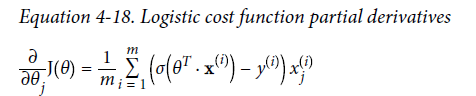
因此改进代价函数将sigmoid函数形式下的代价函数同求对数，从而将代价函数转化为一个凸函数。

代价函数（策略）





对代价函数的参数θj求导，构造梯度向量



## 8.3 Decision Boundaries

分类边界

## 8.4 算法

模型学习算法，如梯度下降等

# softmax regression

## 9.1 process

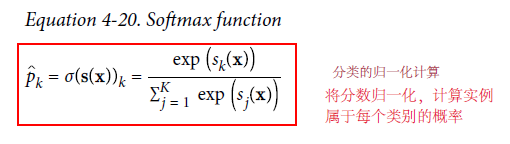
The Logistic Regression model can be generalized to support multiple classes directly, without having to train and combine multiple binary classifiers (as discussed in Chapter 3). This is called Softmax Regression, or Multinomial Logistic Regression.

The idea is quite simple: when given an instance x, the Softmax Regression model first computes a score sk(x) for each class k, then estimates the probability of each class by applying the softmax function (also called the normalized exponential) to the scores. The equation to compute sk(x) should look familiar, as it is just like the equation for Linear Regression prediction。

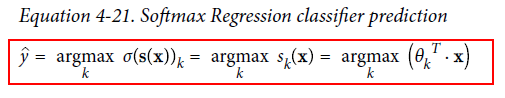
第一步：计算实例属于每个类别的分数



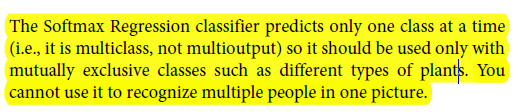
第二步：利用softmax function 计算实例属于每个类别的概率



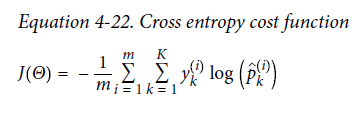
第三步：根据最大类概率来分类



局限：一次只能预测一个类别，无法同时分类多个对象



## 9.2 Cross Entropy





真实分布y下假设分布P的交叉熵。

