Study notes for machine learning

by QingHong Lin

Week 1 Introduction

1 Introduction

1-1 Welcome

1-2 What is MachineLearning

• Define 在进行特定编程的情况下，给予计算机学习能力的领域。
1959 Arthur Samuel 西洋棋
TomMitchell 一个程序被认为能从**经验 E** 中学习，解决**任务 T** 达到**性能度量值 P**，当且仅当，有了**经验 E 后，经过 P 评判，程序在处理 T 时的性能有所提升。**

监督学习 Supervised learning：我们教
非监督学习 Unsupervised learning：自己学

1-4 Supervised learning

给出一个算法，需要部分**数据集**已经有正确答案。（已经告知了正确答案）
回归问题 regression：预测一个**连续值**的输出
回归意味着预测这类连续值属性的种类
分类问题 classification：处理**离散型**数据
向量机：简洁的数学方法让电脑处理无限多的特征

1-5 Unsupervised Learning

所有数据一样的、把数据分成不同的**聚类**、无反馈
Octave、Matlab

2 Model and Cost Function

2-1 Model Representation

Given the “right answer”

Training set 训练集 （m 训练数目、x 输入、y 输出）$(x^{(i)}, y^{(i)})$

Training Set → Learning Algorithm → h(函数 hypothesis)

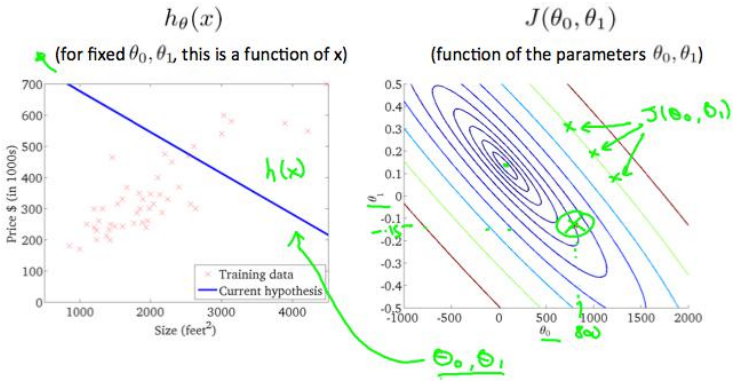
Linear regression 线性回归模型/单变量线性回归 $h_\theta(x) = \theta_0 + \theta_1 x$

2-2 Cost function

平方误差代价函数 $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$ 最小化参数

2-3 Cost Function - Intuition I

Hypothesis: $h_\theta(x)$ Parameters: θ_0, θ_1 Cost Function: $J(\theta_1)$ Goal:minimize $J(\theta_1)$



2-4 Cost Function - Intuition II

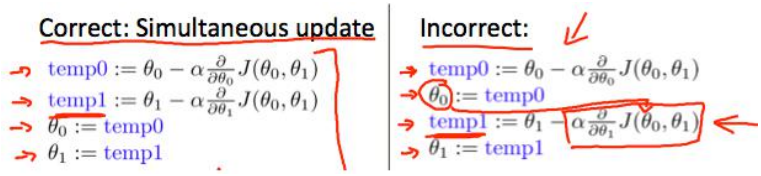
3 Parameter Learning

3-1 Gradient Descent

起始点不同，会得到不同的最优解。

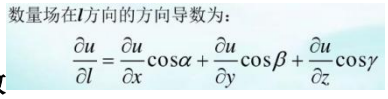
梯度下降(同时更新)

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

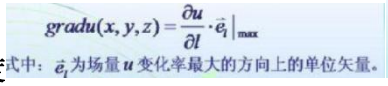


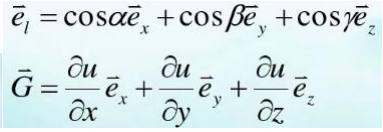
3-2 Gradient Descent Intuition

a 太小，移动太慢。a 太大，可能无法收敛。
移动的过程中，导数值将减少，变化趋势变缓；达到局部最低点后，导数值为零，将保持不变。
补充：方向导数、梯度、散度、旋度

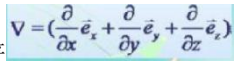
方向导数 

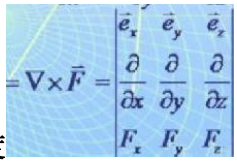
(cos α　cos β　cos γ 是自由指定的)

梯度 



(cos α cos β cos γ 确定)

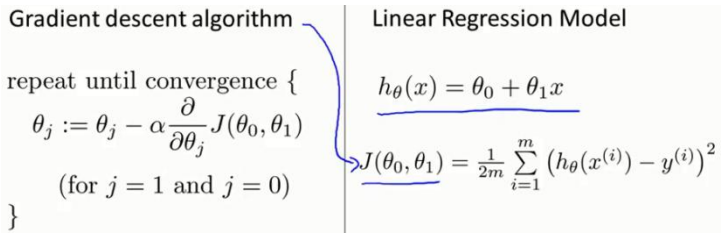
哈密顿算符 

旋度 

梯度是方向导数最大的一点，标量场的梯度是矢量。
矢量场的散度是标量，旋度是矢量。

3-3 Gradient Descent For Linear Regression

梯度下降与代价函数结合



$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$
更容易得到最优解。

- 凸函数没有局部最优解，只有全局最优解。
- 一般我们把初始值设置为 0。
- 批量梯度法 Batch Gradient descent

4 Linear Algebra Review

4-1 Matrices and Vectors

- **Matrix**(2-dimensional arrays)

Dimension of matrix = rows * columns $R^{2行\times3列}$

- **Vector**

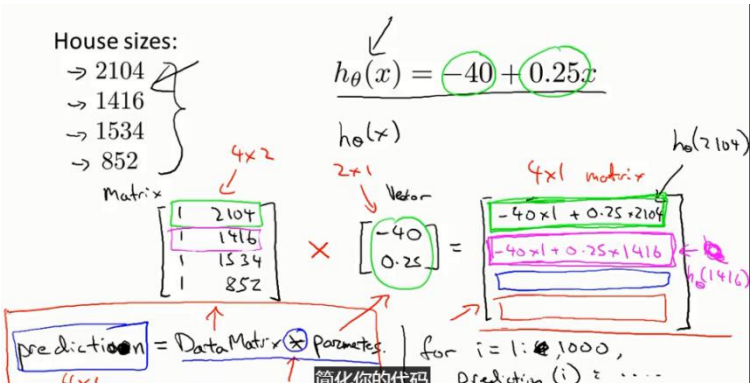
N * 1 列的矩阵 $R^{4行}$

1 索引与 0 索引

4-2 Addition and Scalar Multiplication

- A+2

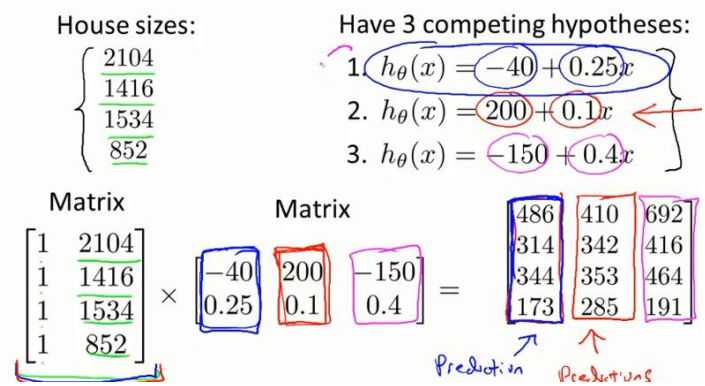
4-3 Matrix Vector Multiplication



4-4 Matrix Matrix Multplication

Details:

$$\underset{\text{m x n matrix (m rows, n columns)}}{A} \times \underset{\text{n x o matrix (n rows, o columns)}}{B} = \underset{\text{m x o matrix}}{C}$$



4-5 Matrix Mult Properties

- **identity matrix**

- Matrices are not commutative: $A * B \neq B * A$
- Matrices are associative: $(A * B) * C = A * (B * C)$

For any matrix A,

$$A \cdot I = I \cdot A = A$$

$I_{n \times n}$

Note:
$AB \neq BA$ in general
$AI = IA$ ✓

4-6 Inverse and Transpose

- Inverse

Matrix inverse: (Square matrix (# rows = # columns)) A^{-1}

If A is an m x m matrix, and if it has an inverse,

$$A(A^{-1}) = A^{-1}A = I.$$

零矩阵没有逆矩阵。

Singular/degenerate(奇异矩阵)不存在逆矩阵。

• Transpose

把矩阵的第一行变成第一列

Example: $A = \begin{bmatrix} 1 & 2 & 0 \\ 3 & 5 & 9 \end{bmatrix}$ $B = A^T = \begin{bmatrix} 1 & 3 \\ 2 & 5 \\ 0 & 9 \end{bmatrix}$

Let A be an m x n matrix, and let $B = A^T$.

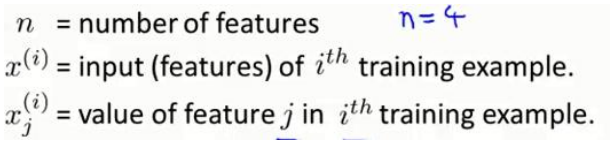
Then B is an n x m matrix, and

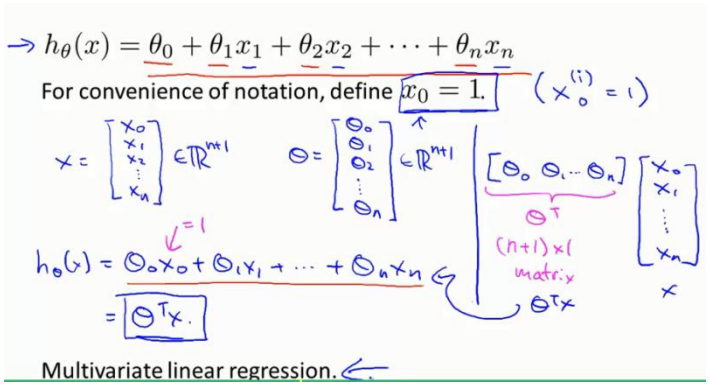
$$B_{ij} = A_{ji}.$$

Week 2 Linear Regression with Multiple Variables

2 Multivariate Linear Regression

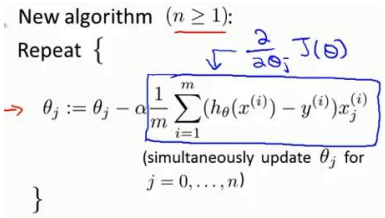
2-1 Multiple Features





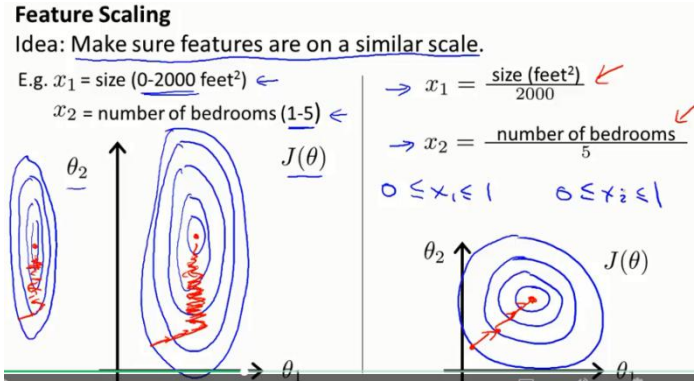
多元线性回归

2-2 Gradient Descent for Multiple Variables

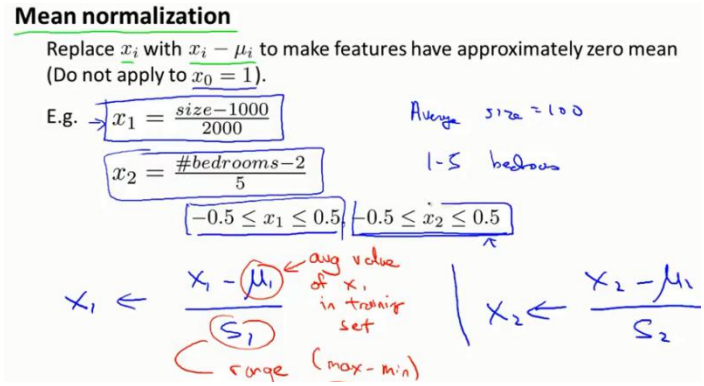


2-3 Gradient Descent in Practice I - Feature Scaling

(目的为了让梯度下降法使用地更快一些。)
特征缩放(feature scaling)



归一化(Mean normalizatio)



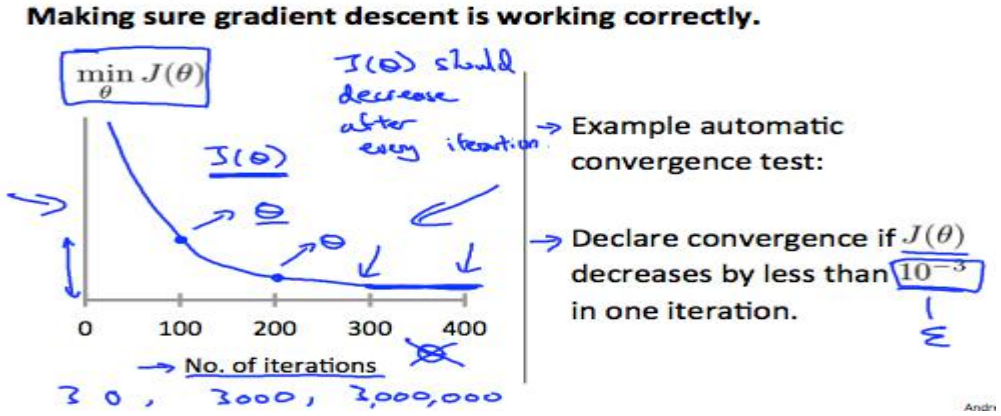
$$x_i := \frac{x_i - \mu_i}{s_i}$$

For example, if x_i represents housing prices with a range of 100 to 2000 and a mean value of 1000, then, $x_i := \frac{price - 1000}{1900}$.

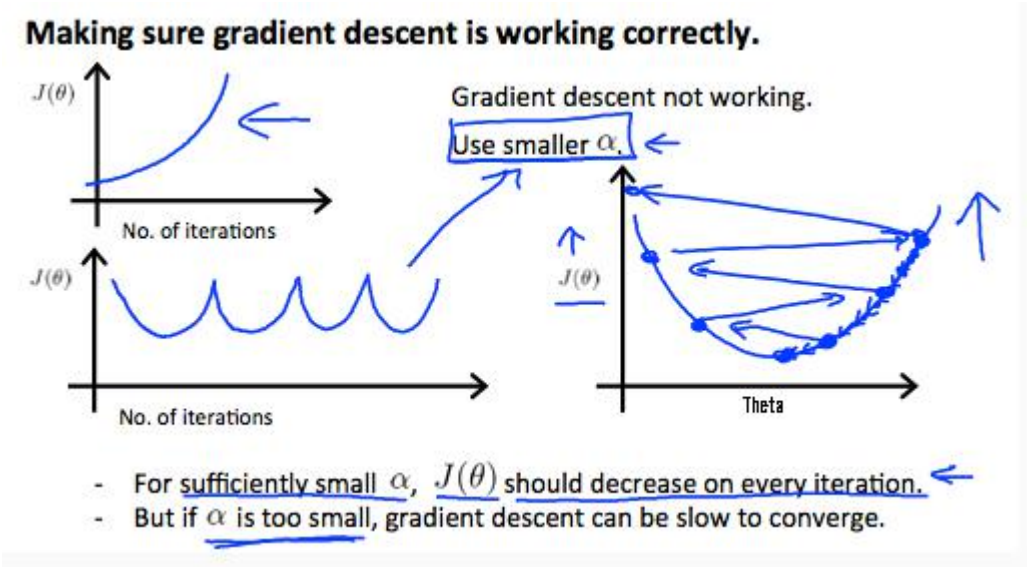
不需要很精确，近似即可。

2-4 Gradient Descent in Practice II - Learning Rate

(Debugging and 如何确定学习率α)



横坐标(迭代次数) 纵坐标(代价函数的误差值)
当每次迭代之间的差值小于 10^-3 次方，说明收敛。

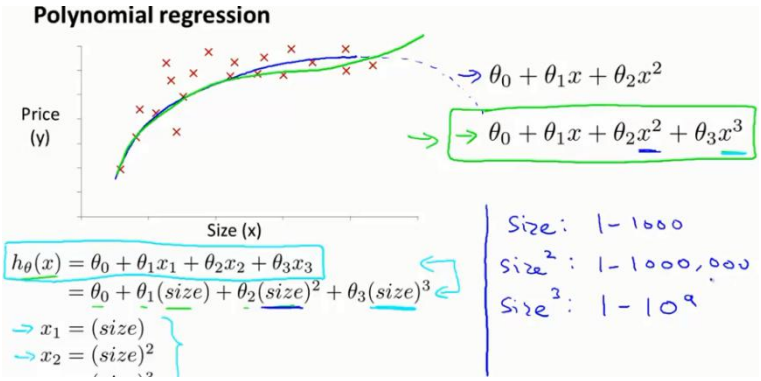


如果 α 太小，收敛会很慢。
如果 α 太大，无法收敛，会出现上升的情况。

2-5 Features and Polynomial Regression

选择特征和得到不同的学习算法、多项式回归。

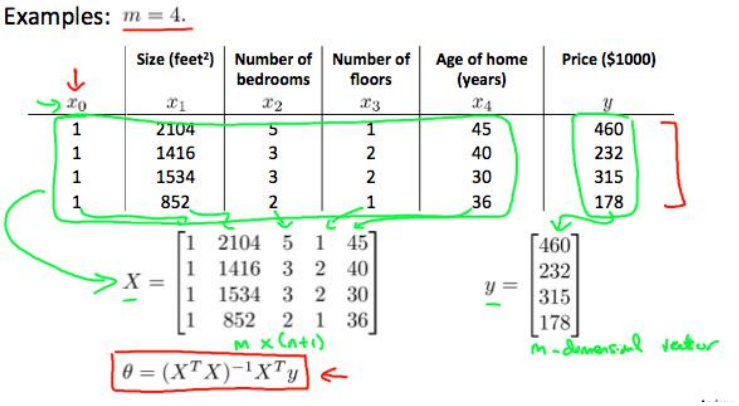
- 自由选择特征 如用面积代替长和宽
- 用多项式拟合 $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^3$ 、 $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 \sqrt{x_1}$
- 多项式拟合情况下，特征缩放就很重要了



3 Computing Parameters Analytically

3-1 Normal Equation

(正规方程法) 一次性求解最优解。



m:样本数 n:特征数 X:设计矩阵

使用正规方程法不需要归一化。

Gradient Descent	Normal Equation
Need to choose alpha	No need to choose alpha
Needs many iterations	No need to iterate
O (kn^2)	O (n^3), need to calculate inverse of $X^T X$
Works well when n is large	Slow if n is very large

当 n(特征数)大于 10000 时，使用正规方程法。

3-2 Normal Equation Noninvertibility

The 'pinv' function will give you a value of θ even if $X^T X$ is not invertible.

- 在 octave 中使用 pinv 可以计算出值即使不可逆。
- 若不可逆，可能有以下的原因：

多余的特征，比如两个特征线性相关。如步数和米。

太多特征，导致样本数<=特征数，删除一些特征或者使用 regularization。

4 Octave/Matlab Tutorial

4-1 Basic Operations

randn %高斯分布 hist %作出曲线图

4-2 Moving Data Around

Size、length、pwd、cd、ls、load、who、whos、clear、save

A = [A, [100; 101; 102]] %附加新的列向量

C = [A,B] %将两个矩阵按列连接在一起。
C = [A;B] %上下排列

4-3 Computing on Data

Log、exp、abs、A’、max、find、magic、prod
Floor、ceil、flipud
A+ones(length(v),1)

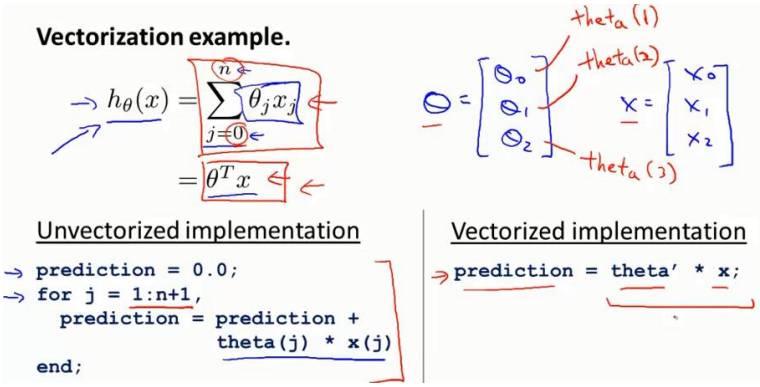
4-4 Plotting Date

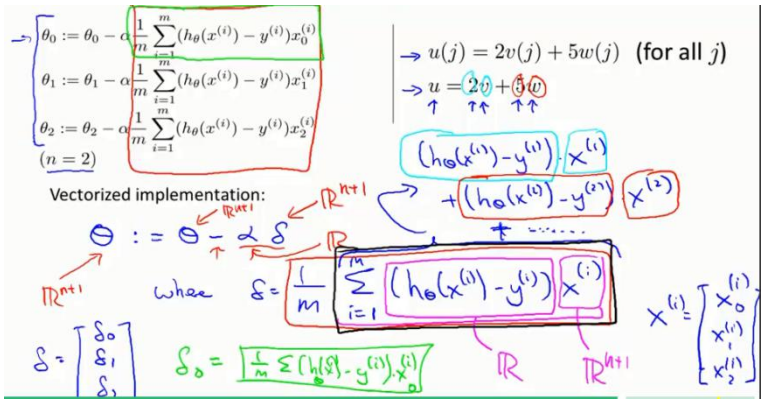
plot、close、figure、subplot、axis、clf
imagesc(A)、colorbar、colormap gray
• a=1, b=2, c=3 %逗号间隔同时执行。

4-5 for,while,if statement

Indices、disp、exit/quit、function
• addpath(‘C:\Users\ang\Desktop’) 添加寻找路径。

4-6 Vectorization





• 向量只有一行。

Exercise

• 向量化计算可以通过矩阵的维数(即行列数)来简单进行判断。

```
X = [ones(m, 1), data(:,1)];
theta = zeros(2, 1);
J = computeCost(X, y, theta);

m = length(y);

J = 1/(2*m)*sum((X*theta-y).^2);
```

$J(\theta) = \frac{1}{2m}(X\theta - \vec{y})^T(X\theta - \vec{y})$ 两者等价。

```
b = 1/m*(X'*(X*theta-y));
theta = theta - alpha*b;
```

• 生成椭圆代价图

```
contour(theta0_vals, theta1_vals, J_vals, logspace(-2, 3, 20))
```

%logspace 用来在 10^-2 到 10^3 生成数目为 20 的等比数列，用来给 contour 表示间隔。

```
xlabel('\theta_0'); ylabel('\theta_1');
hold on;
plot(theta(1), theta(2), 'rx', 'MarkerSize', 10, 'LineWidth', 2);
```

• 补充:给一维的数组作曲线图。

plot(J);

Plot(1:50,J(1:50),'r') %在 1~50 的范围内作图。

Week 3 Logistic Regression

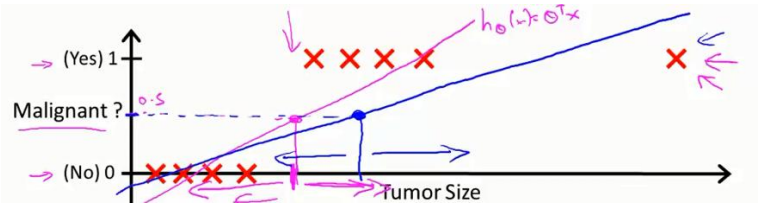
1 Classification and Representation

1-1 Classification

Logistic Regression Model

Want $0 \leq h_\theta(x) \leq 1$

红叉为正样本，〇为负样本。



• 分类问题对大于 0.5 的值为 1，小于 0.5 为 0；故线性回归不好用，其一：样本会导致方程改变。其二是其范围太大。故我们使用 **Logistic regression**

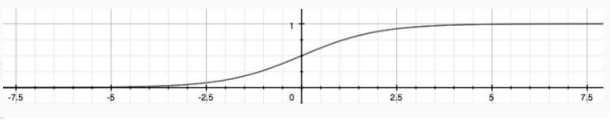
1-2 Hypothesis Representation

Sigmoid Function/Logistic Function

$$h_\theta(x) = g(\theta^T x)$$

$$z = \theta^T x$$

$$g(z) = \frac{1}{1+e^{-z}}$$



$h_\theta(x)$ will give us the **probability** that our output is 1. 函数值表示判定为 1 的概率。

$$h_\theta(x) = P(y=1|x;\theta) = 1 - P(y=0|x;\theta)$$

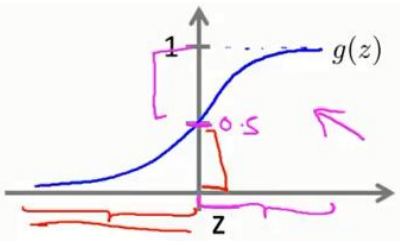
$$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$

1-3 Decision Boundary

Logistic regression

$\rightarrow h_\theta(x) = g(\theta^T x) = P(y=1 \mid x; \theta)$

$\rightarrow g(z) = \frac{1}{1+e^{-z}}$



Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

$\rightarrow \theta^T x \geq 0$

$g(z) \geq 0.5$ when $z \geq 0$

$h_\theta(x) = g(\theta^T x) \geq 0.5$ whenever $\theta^T x \geq 0$ (z)

predict "$y = 0$" if $h_\theta(x) < 0.5$

$h_\theta(x) = g(\theta^T x)$ $\quad g(z) < 0.5$

$\theta^T x < 0$

$h_\theta(x) \geq 0.5 \rightarrow y = 1 \quad g(z) \geq 0.5$

$h_\theta(x) < 0.5 \rightarrow y = 0 \quad when\ z \geq 0$

$z = 0, e^0 = 1 \Rightarrow g(z) = 1/2$

$z \rightarrow \infty, e^{-\infty} \rightarrow 0 \Rightarrow g(z) = 1$

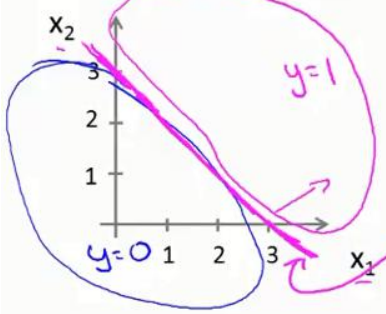
$z \rightarrow -\infty, e^{\infty} \rightarrow \infty \Rightarrow g(z) = 0$

$\theta^T x \geq 0 \Rightarrow y = 1$

$\theta^T x < 0 \Rightarrow y = 0$

Decision Boundary

$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$



$\rightarrow h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

$-3 \quad 1 \quad 1$

Decision boundary

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$ ($\theta^T x$)

$\rightarrow x_1 + x_2 \geq 3$

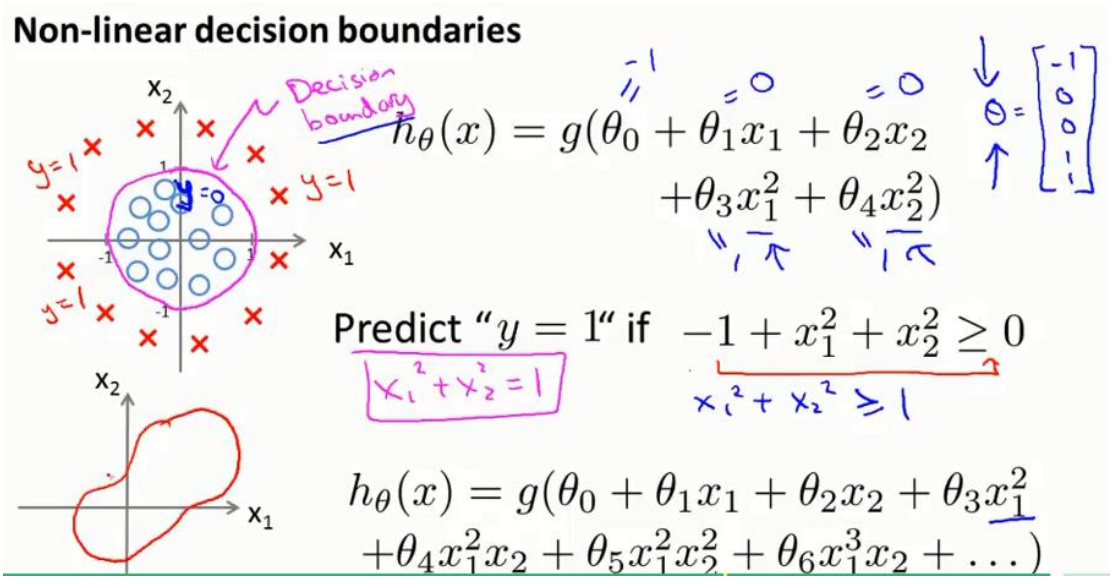
x_1, x_2

$\rightarrow h_\theta(x) = 0.5$

$x_1 + x_2 = 3$

$x_1 + x_2 < 3$

$\rightarrow y = 0$



决策边界由参数给定，参数由数据集给定。

Example:

$$\theta = \begin{bmatrix} 5 \\ -1 \\ 0 \end{bmatrix}$$

$y = 1 \; if \; 5 + (-1)x_1 + 0x_2 \geq 0$

$5 - x_1 \geq 0$

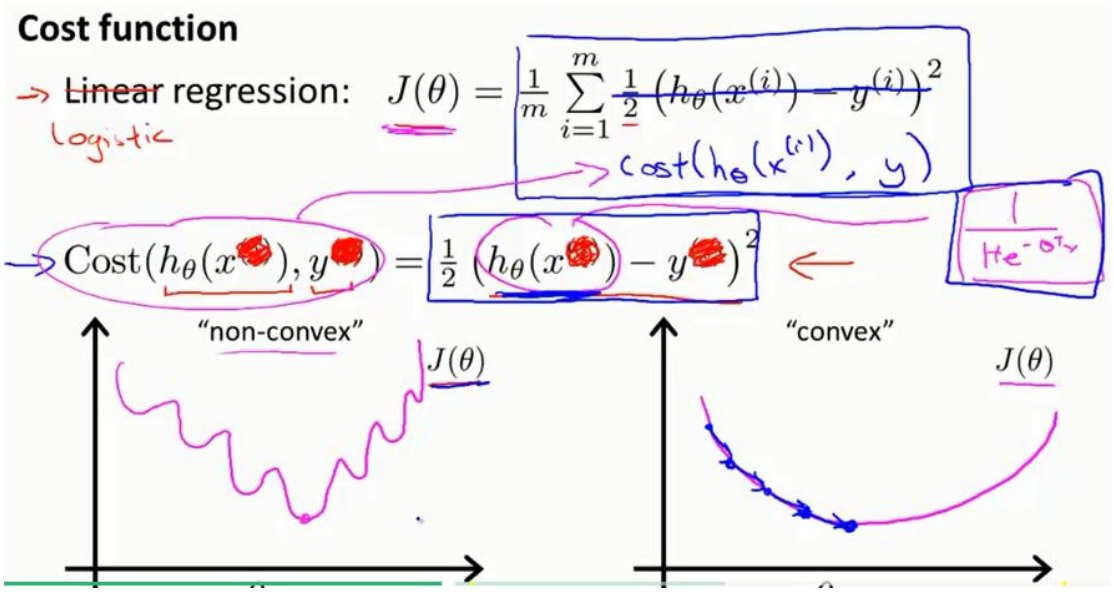
$-x_1 \geq -5$

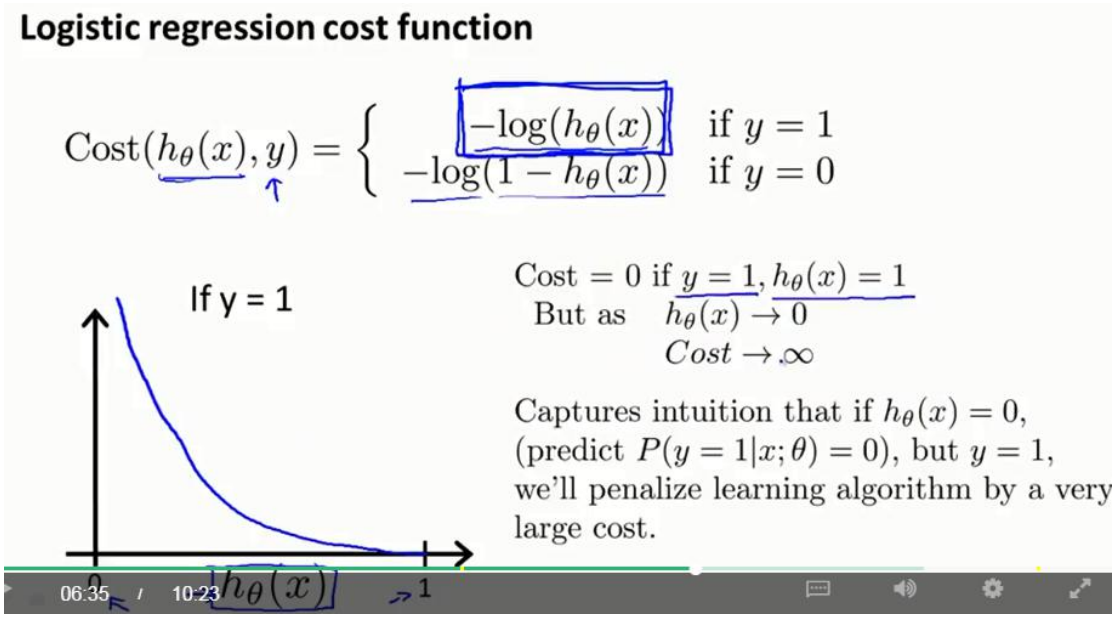
$x_1 \leq 5$

2 Logistic Regression Model

2-1 Cost Function

• 先定义代价函数 Cost，若用线性回归中的代价函数将不适用，因为若非凸函数可能会出现下左图中的情况，无法收敛到全局最优解，于是我们定义新的代价函数。





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

$\text{Cost}(h_\theta(x), y) = -\log(h_\theta(x))$ if y = 1

$\text{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x))$ if y = 0

$\text{Cost}(h_\theta(x), y) = 0$ if $h_\theta(x) = y$

$\text{Cost}(h_\theta(x), y) \to \infty$ if $y = 0$ and $h_\theta(x) \to 1$

$\text{Cost}(h_\theta(x), y) \to \infty$ if $y = 1$ and $h_\theta(x) \to 0$

2-2 Simplified Cost Function and Gradient Descent

(简单的方法来写代价函数并用梯度下降法拟合参数)

• 将两种情况结合在一起

$$\text{Cost}(h_\theta(x), y) = -y\ \log(h_\theta(x)) - (1 - y)\log(1 - h_\theta(x))$$

• 代入梯度下降法，会发现跟线性回归一样。(这其中是有推导过程的。)

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

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) \qquad \theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

补充推导过程

第三周课程中，逻辑回归代价函数的求导过程没有具体展开，在此推导并记录：

逻辑回归的代价函数可以统一写成如下一个等式：

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

其中：$h_\theta(x^{(i)}) = \frac{1}{1+e^{-\theta^{\mathrm{T}}x}}$

为了避免求导过程太冗长复杂，我们做一些显示的简化：

$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} K(\theta)]$

其中：$K(\theta) = y^{(i)}log(h_\theta(x^{(i)})) + (1-y^{(i)})log(1-h_\theta(x^{(i)}))$

$h_\theta(x^{(i)}) = \frac{1}{1+e^{-\theta^{\mathrm{T}}x}}$

OK，下面开始我们的推导过程：如果要求$J(\theta)$对某一个参数θ的偏导数，则：

（1）根据求导公式，可以先把常数项$-\frac{1}{m}\sum_{i=1}^{m}$提取出来，这样就只需要对求和符号内部的表达式求导，即：

$J(\theta)' = -\frac{1}{m}[\sum_{i=1}^{m} K(\theta)']$

$K(\theta)' = (ylog(h_\theta(x)) + (1-y)log(1-h_\theta(x)))'$（为方便显示，先把右上角表示第i个样本的上标去掉）

（2）根据对数复合求导公式，$log(x)' = \frac{1}{x}x'$，对$K(\theta)$继续求导可得：

$K(\theta)' = y\frac{1}{h_\theta(x)}h_\theta(x)' + (1-y)\frac{1}{1-h_\theta(x)}(1-h_\theta(x))'$

（3）根据幂函数复合求导公式，$(y^x)' = xy^{x-1}x'$，及以e为底的指数求导公式，对$h_\theta(x)$继续求导可得：

$h_\theta(x)' = \left(\frac{1}{1+e^{-\theta^{\mathrm{T}}x}}\right)' = -\frac{(1+e^{-\theta^{\mathrm{T}}x})'}{(1+e^{-\theta^{\mathrm{T}}x})^2} = \frac{e^{-\theta^{\mathrm{T}}x}(\theta^{\mathrm{T}}x)'}{(1+e^{-\theta^{\mathrm{T}}x})^2} = \left(\frac{1}{1+e^{-\theta^{\mathrm{T}}x}}(1-\frac{1}{1+e^{-\theta^{\mathrm{T}}x}})\right)(\theta^{\mathrm{T}}x)' = h_\theta(x)(1-h_\theta(x))(\theta^{\mathrm{T}}x)'$

同理，$(1-h_\theta(x))' = -\frac{e^{-\theta^{\mathrm{T}}x}(\theta^{\mathrm{T}}x)'}{(1+e^{-\theta^{\mathrm{T}}x})^2} = -h_\theta(x)(1-h_\theta(x))(\theta^{\mathrm{T}}x)'$

（4）把步骤3的结果带入步骤2，化简后可得：

$K(\theta)' = (y-h_\theta(x))(\theta^{\mathrm{T}}x)'$

再把上面结果带入步骤1，化简后可得：

$J(\theta)' = \frac{1}{m}[\sum_{i=1}^{m}(h_\theta(x)-y)(\theta^{\mathrm{T}}x)']$

最后$(\theta^{\mathrm{T}}x)'$，对第j个θ求偏导，结果即X_j（j表示样本中第几项），得到最终结果：

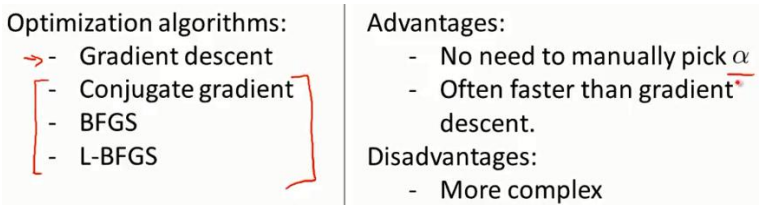
$\frac{\partial J(\theta)}{\partial\theta_j} = \frac{1}{m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right]$

来自 **https://www.cnblogs.com/zhongmiaozhimen/p/6155093.html**

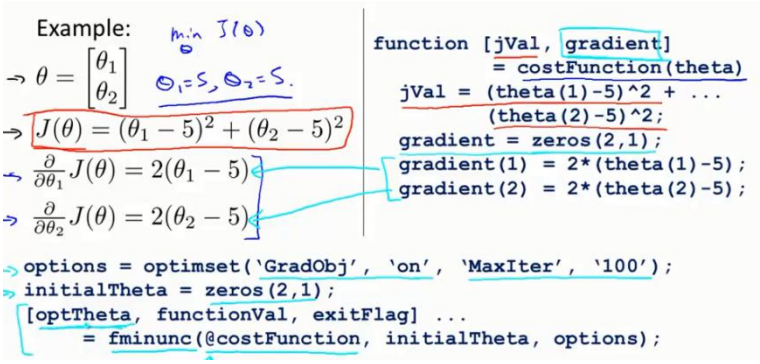
2-3 Advanced Optimization

(优化，提高逻辑回归的速度)

- 调用已有的库函数如 fminunc()。



• 先写一个函数 costFunciton，能返回 J 和 gradient，然后代入 fminunc()函数中。它与梯度下降法的不同在于，不需要定义学习率，并且速度更快。



补充

1)要使用 fminun 函数的如下形式：[x,fval,exitflag] = fminunc(fun,x0,options);

2)该方法需要定义 3 个输入参数 fun,x0 和 options;我们以图片的为例子进行 fminunc 用法的说明。

3)图片中的第一个输入参数 fun 定义为 @costFunction 该“@”符号涉及句柄知识，在另一篇（http://blog.csdn.net/gzp444280620/article/details/49252491）博文中有讲到。

4)第二个输入参数 x0 定义为 initialTheta 该参数为一个 2x1 矩阵（2 个元素的列向量）。该列向量为用户自定义梯度下降法的输出参数。使用前要预先初始化。

5)第三个输入参数为 options 该参数的的作用包括是否使用用户自定义的梯度下降公式（GradObj）以及迭代次数（MaxIter）。%打开自定义梯度下降公式与最大迭代次数 100。

6)例子中使用了自定义的梯度下降公式（costFunction 中的 gradient 返回值），并且把迭代次数设为迭代 100 次。 例： options = optimset('GradObj','on','MaxIter','100');

7)输入参数定义完成后我们来看一下 fminunc 的输出参数[x,fval,exitflag]，其中第一个返回值 x 为上面 function [jVal,gradient] = costFunction(theta)中的第二个返回值 gra，第二个返回值 fval 返回的是 costFunction 函数的第一个返回值 jVal 这个关系是理解 fminunc 的重点。第三个返回值 exitflag 返回值为 0 或 1，表示在 theta 点定义的 Jval 函数是否收敛。

来自 http://blog.csdn.net/gzp444280620/article/details/49272977

3 Multiclass Classification

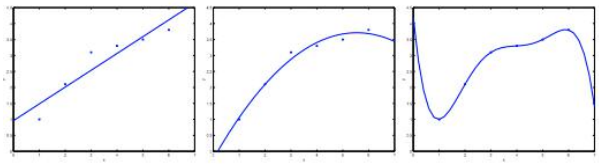
3-1 Multiclass Classification:One-vs-all

$$y \in \{0, 1 \dots n\}$$
$$h_\theta^{(0)}(x) = P(y = 0|x;\theta)$$
$$h_\theta^{(1)}(x) = P(y = 1|x;\theta)$$
$$\dots$$
$$h_\theta^{(n)}(x) = P(y = n|x;\theta)$$
$$\text{prediction} = \max_i(h_\theta^{(i)}(x))$$

4 Regularization

4-1 The Problem of Overfitting

• 欠拟合、高偏差
• 过拟合、高方差、无法很好泛化新样本(特征量太多、样本少将会导致过拟合。)



• 解决方法
1 减少特征数目(人为选择 or 使用模型选择算法)
2 正则化(保持特征数，减少 theta 的大小)

Options:
1. Reduce number of features.
   — Manually select which features to keep.
   — Model selection algorithm (later in course).
2. Regularization.
   — Keep all the features, but reduce magnitude/values of parameters θ_j.
   — Works well when we have a lot of features, each of which contributes a bit to predicting y.

4-2 Cost Function

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

- **注意后面一项是从 theta(1)开始的，没有包括 theta(0)。**
- 正则化参数不能太大，若太大，会导致 theta 值都过小，结果为欠拟合。

4-3 Regularized Linear Regression

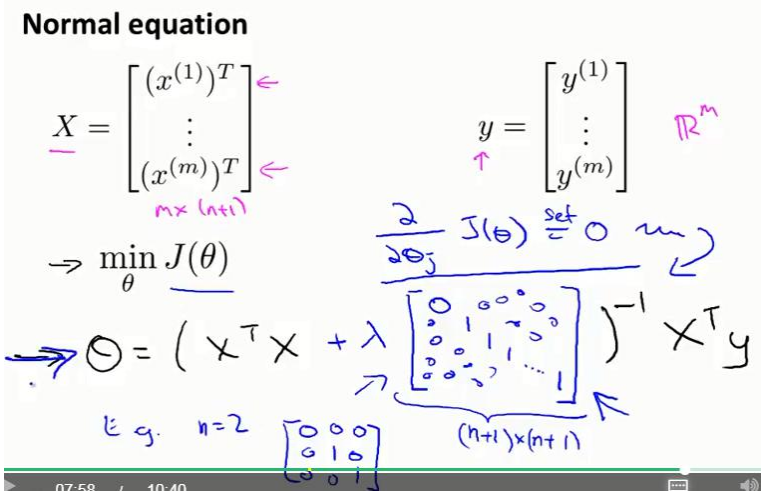
- **梯度下降法**有所不同，theta(0)不变，而 theta(1)多了一项，但是原理都是 J 对 theta 的偏导，于是我们可以将之合并。

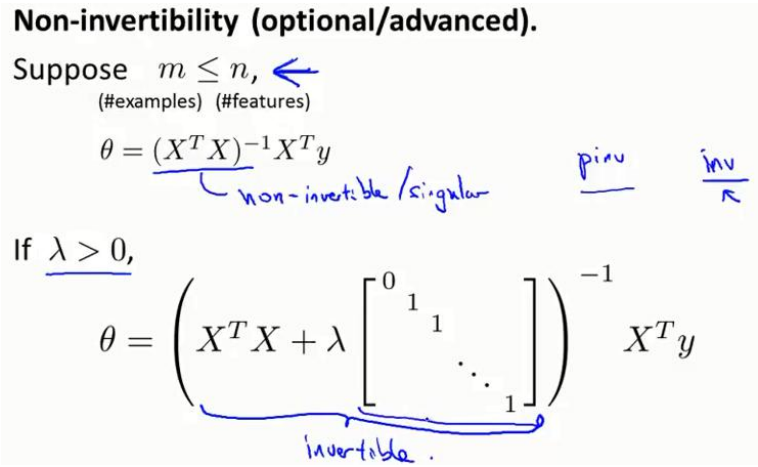
$$\text{Repeat } \{$$
$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$
$$\theta_j := \theta_j - \alpha\left[\left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j\right] \qquad j \in \{1,2...n\}$$
$$\}$$

合并后会发现 $(1-\alpha\frac{\lambda}{m})$ 总是略小于 1，指每次更新 theta 会使得值变为原来 0.99 倍。

$$\theta_j := \theta_j(1-\alpha\tfrac{\lambda}{m}) - \alpha\tfrac{1}{m}\textstyle\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

- **正规方程法**，需要补充多一项。(注意各矩阵的维度)

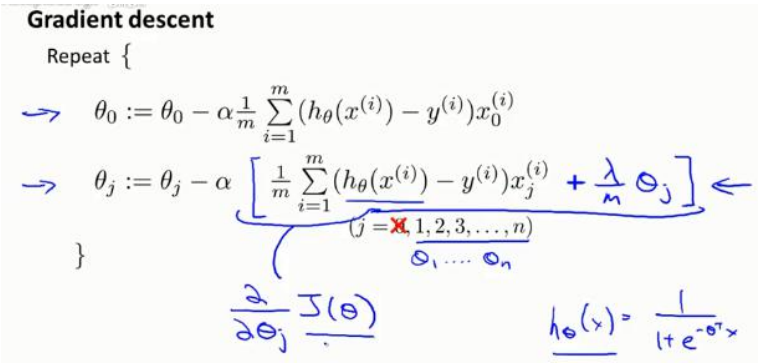




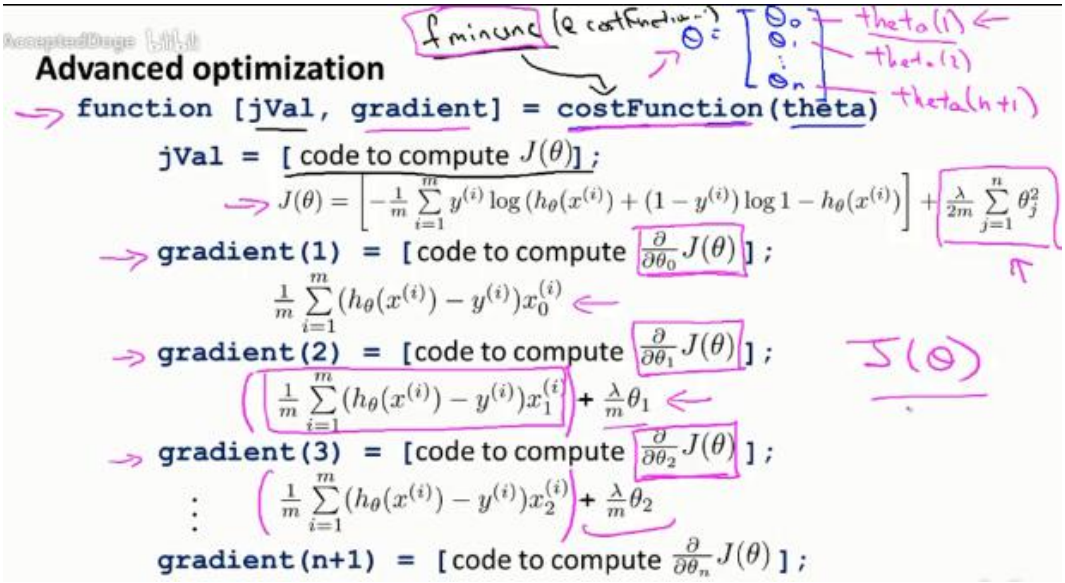
• 当样本数小于特征数的时候，会导致 theta 为不可逆矩阵(虽然 pinv 可以解决该问题。)
但是如果用了正则化，就可以将不可逆矩阵变为可逆矩阵。

4-4 Regularized Logistic Regression

之前提过有两种算法：梯度下降法与自行设计的优化算法。



• 跟线性回归中的梯度下降法很相似，不同之处在于 h 函数是 S 函数。



• 高级优化算法，除了 theta(0)不变，其他都多一项。

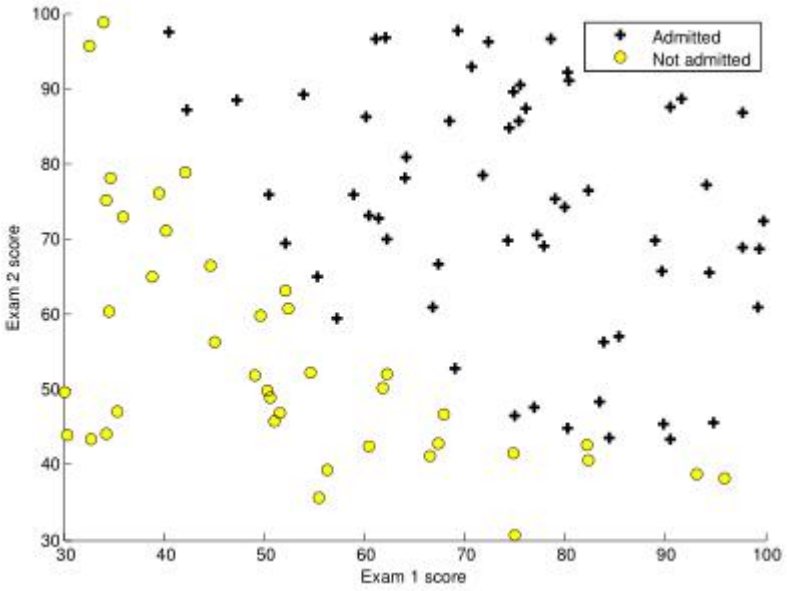
$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\ \log(h_\theta(x^{(i)})) + (1-y^{(i)})\ \log(1-h_\theta(x^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

Exercise3

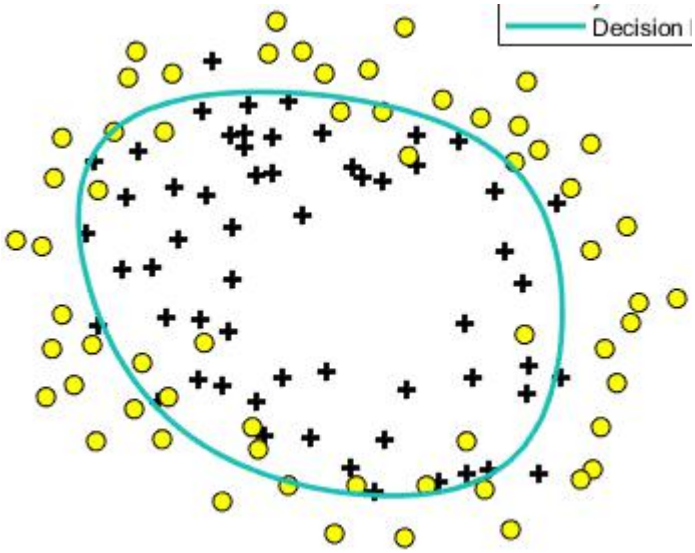
1)如何作图

```
% Find Indices of Positive and Negative Examples
pos = find(y==1); neg = find(y == 0);

% Plot Examples
plot(X(pos, 1), X(pos, 2), 'k+','LineWidth', 2, ...
     'MarkerSize', 7);
plot(X(neg, 1), X(neg, 2), 'ko', 'MarkerFaceColor', 'y', ...
     'MarkerSize', 7);
```



2)补充看懂其是如何作出决策边界的函数图



```
if size(X, 2) <= 3 %If the number of features smaller or equal two
    % Only need 2 points to define a line, so choose two endpoints
    plot_x = [min(X(:,2))-2,  max(X(:,2))+2];

    % Calculate the decision boundary line
    plot_y = (-1./theta(3)).*(theta(2).*plot_x + theta(1));

    % Plot, and adjust axes for better viewing
    plot(plot_x, plot_y)

    % Legend, specific for the exercise
    legend('Admitted', 'Not admitted', 'Decision Boundary')
    axis([30, 100, 30, 100])
```

3)J 和梯度的计算

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

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \qquad \text{for } j \geq 1$$

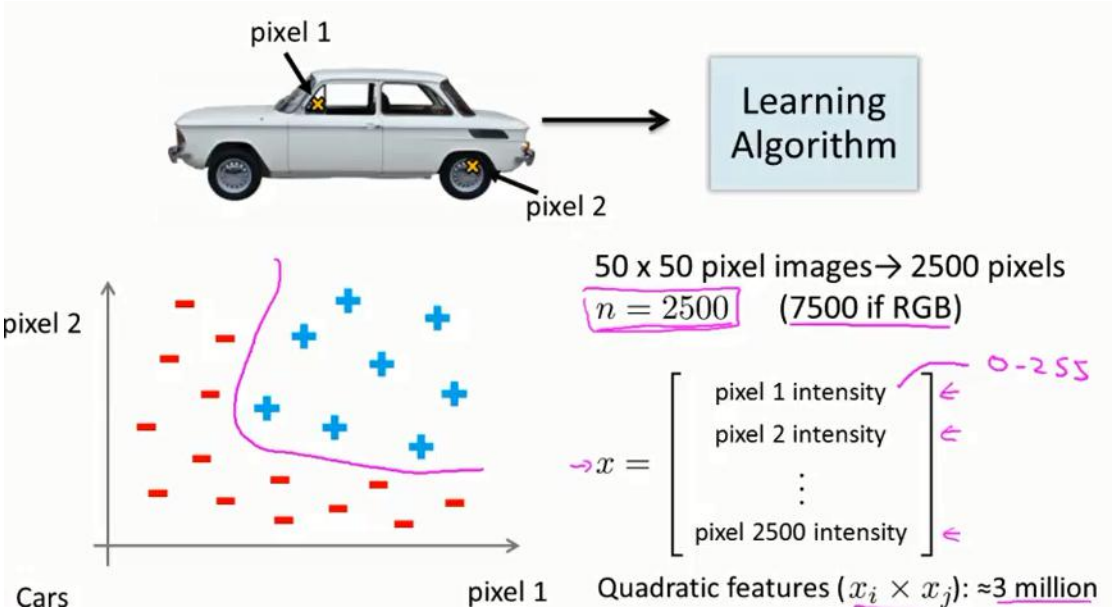
```
J = 1/m *(-y'*log(sigmoid(X*theta))-(1-y)'*log(1-sigmoid(X*theta)))...
    +lambda/(2*m)*(theta(2:end)'*theta(2:end));
grad = 1/m *sum(X'*(sigmoid(X*theta)-y),2)+lambda*[0;theta(2:end)]/m;
```

Week 4 Neural Networks: Representation

1 Motivations

1-1 Non-linear Hypotheses

当特征数量很大的时候，使用逻辑回归算法将产生很多的特征项，故不能很好地解决问题。比如一张汽车把手灰度图的像素是 50x50，那么将会有 2500 个像素点，那么只考虑二次项将会产生 n^2/2 即 300 百万个多项式。



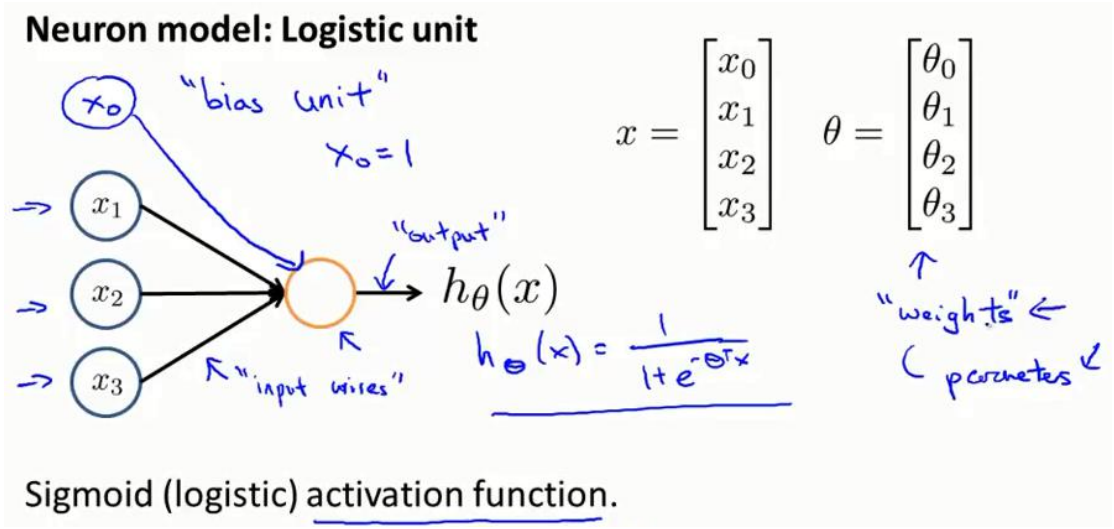
1-2 Neurons and the Brain

• 把传感器接入大脑，大脑通过算法，就能找出学习数据的方法。

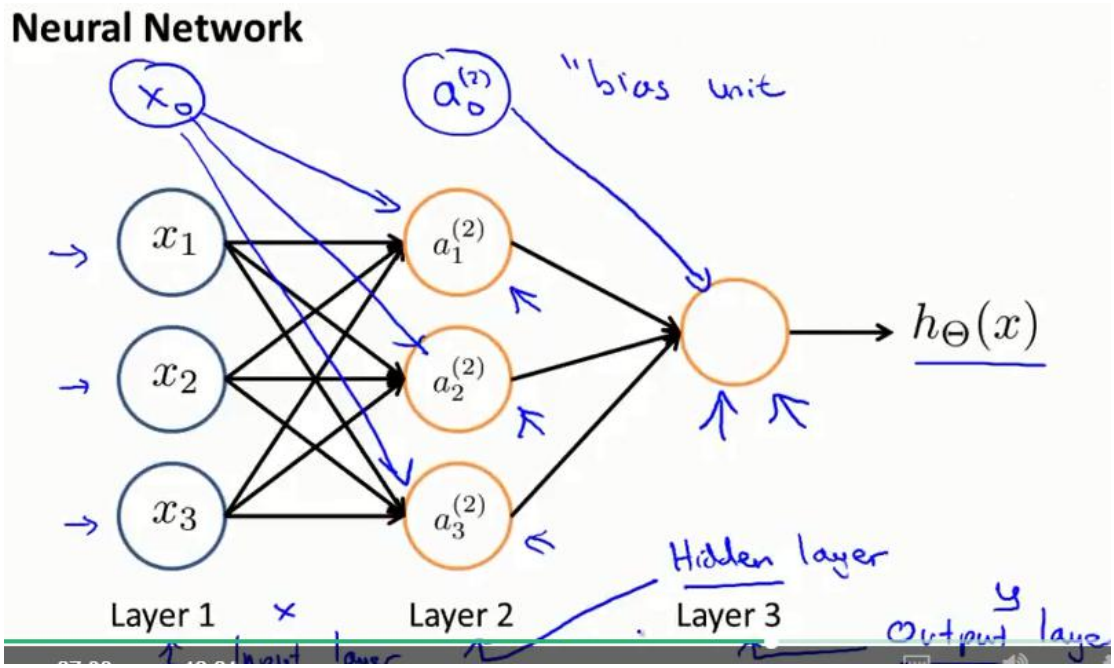
2 Neural Networks

2-1 Model Representation I

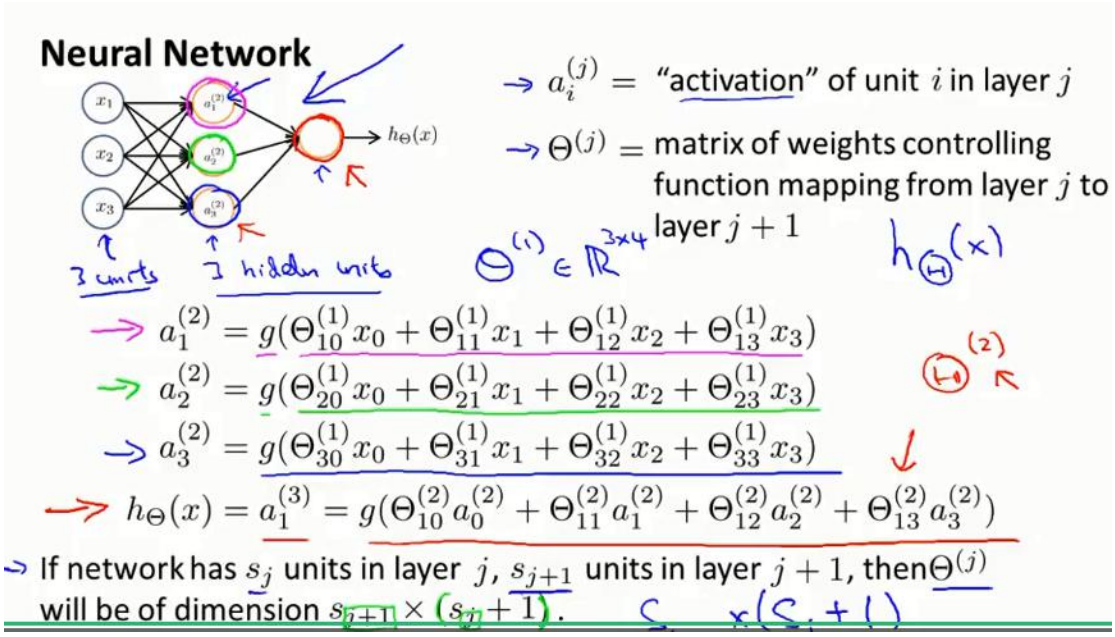
• 模仿大脑中的神经元



• x0 为偏置节点，其值永远为 1



- 输入层、隐藏层、输出层
- **激励**：一个具体神经元读入计算并输出的值
- **参数矩阵：注意下标** $S_{j+1}\times(S_j+1)$

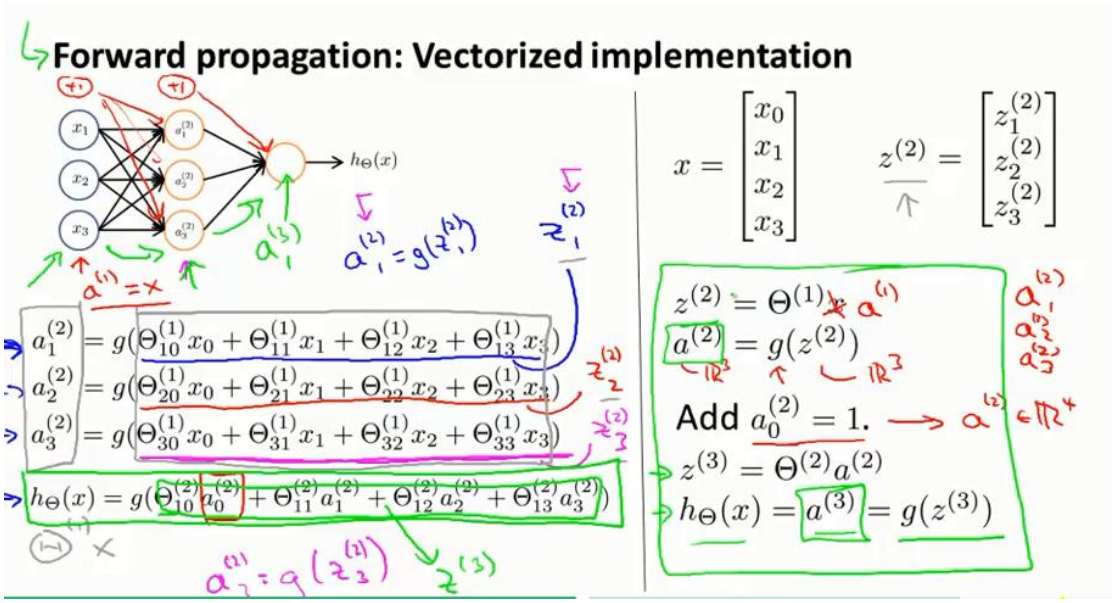


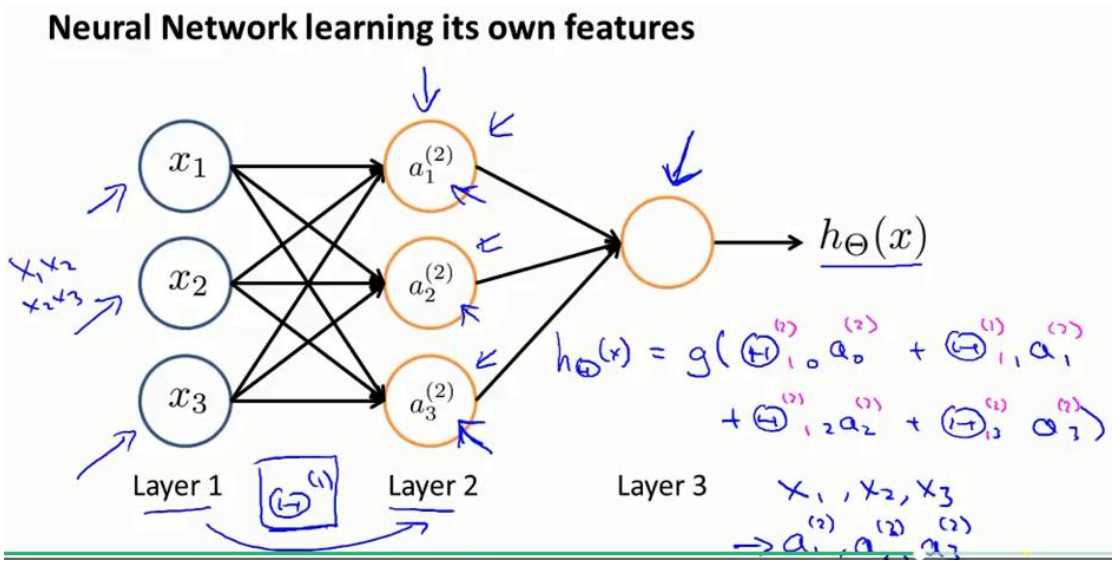
Example: If layer 1 has 2 input nodes and layer 2 has 4 activation nodes. Dimension of $\Theta^{(1)}$ is going to be 4×3 where $s_j = 2$ and $s_{j+1} = 4$, so $s_{j+1} \times (s_j + 1) = 4 \times 3$.

2-2 Model Representation II

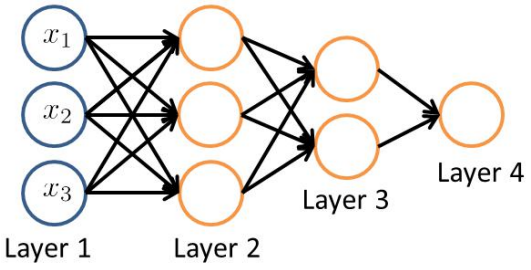
(如何高效地计算)

- 将其向量化





• 逻辑回归是使用 x1、x2、x3 及其多项式作为输入；而神经网络是以 a1、a2、a3 及其多项式作为输入，两者非常相似。



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

• **重要**

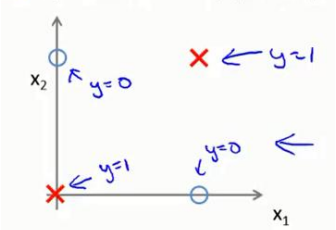
$z^{(j+1)} = \Theta^{(j)}a^{(j)}$ 、 $z^{(j)} = \Theta^{(j-1)}a^{(j-1)}$

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

3 Applications

3-1 Examples and Intuitions I

→ x_1, x_2 are binary (0 or 1).



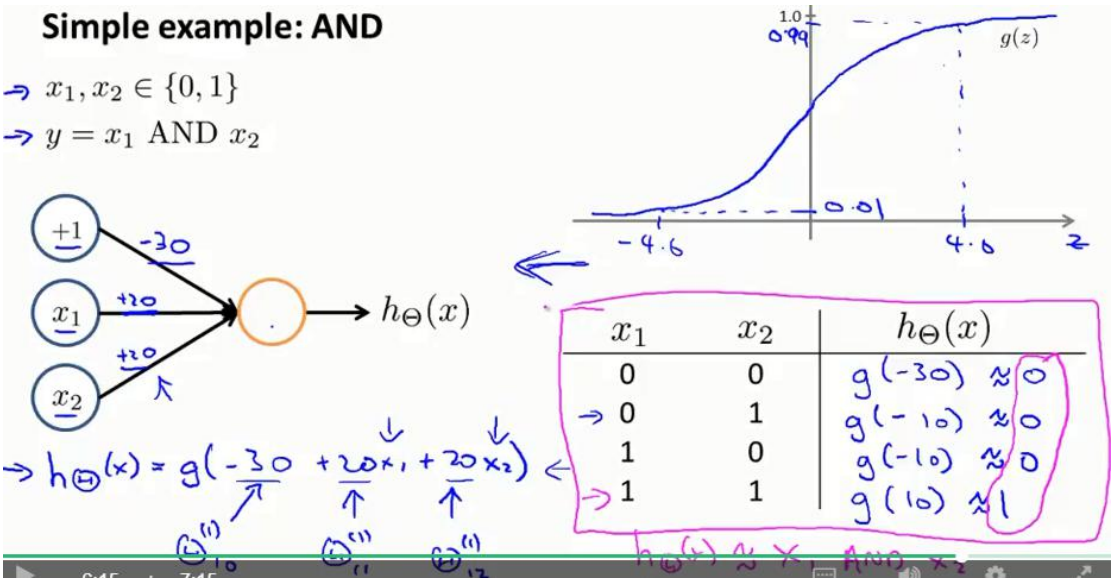
$y = x_1 \text{ XOR } x_2$

$x_1 \text{ XNOR } x_2$

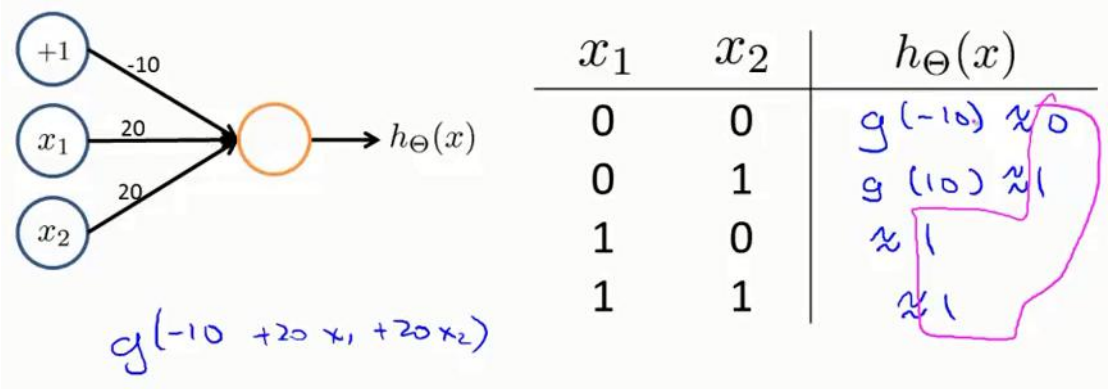
$\text{NOT } (x_1 \text{ XOR } x_2)$

x1 同或 x2，y=1；x1 异或 x2，y=0

• 与运算

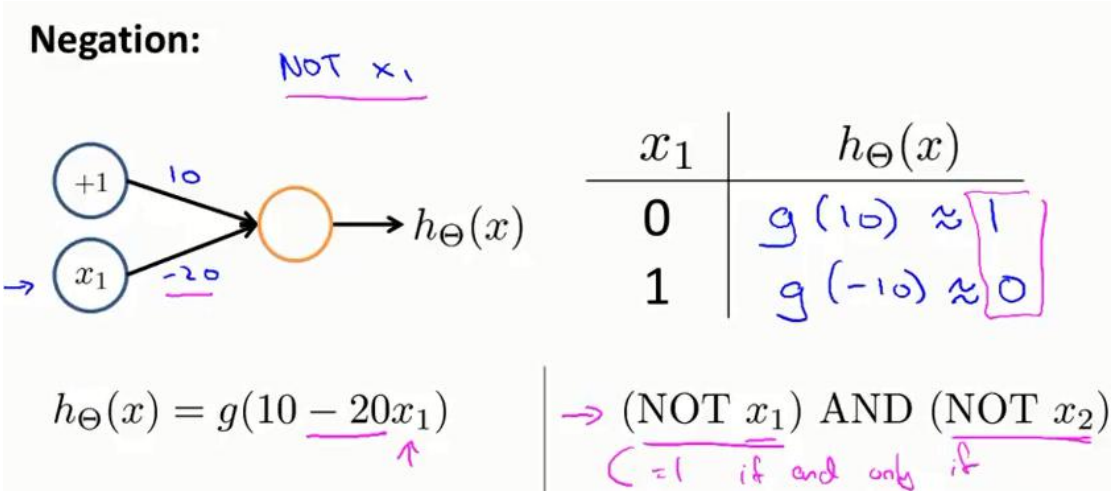


• 或运算

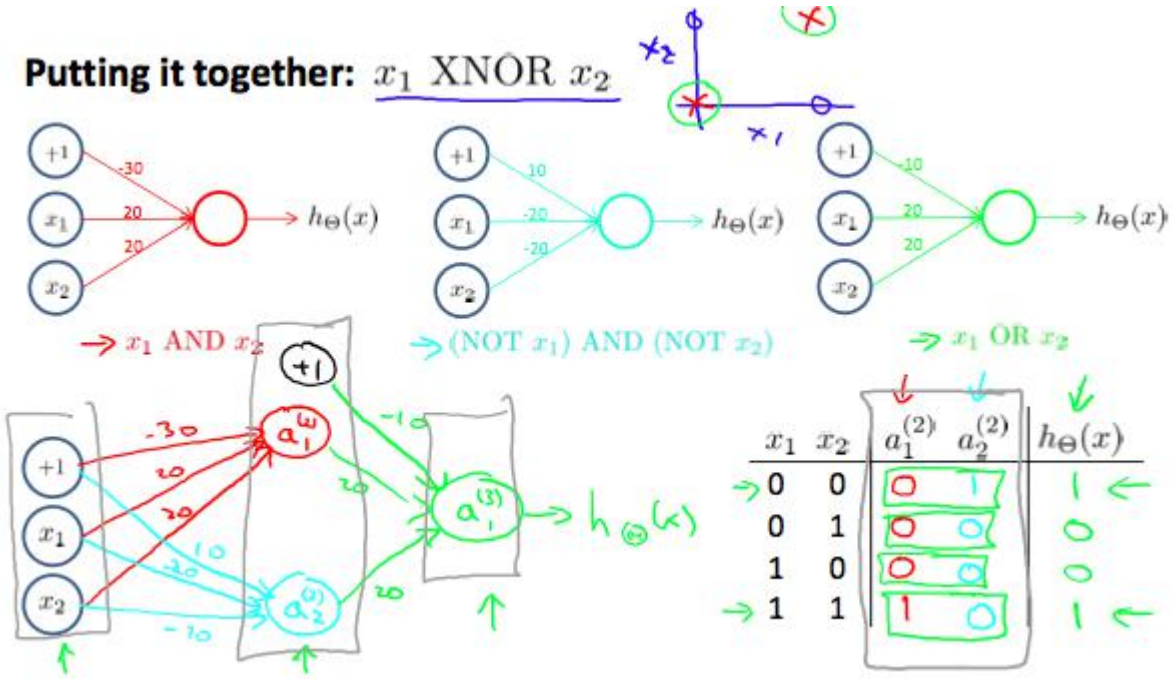


3-2 Examples and Intuitions II

• 取反

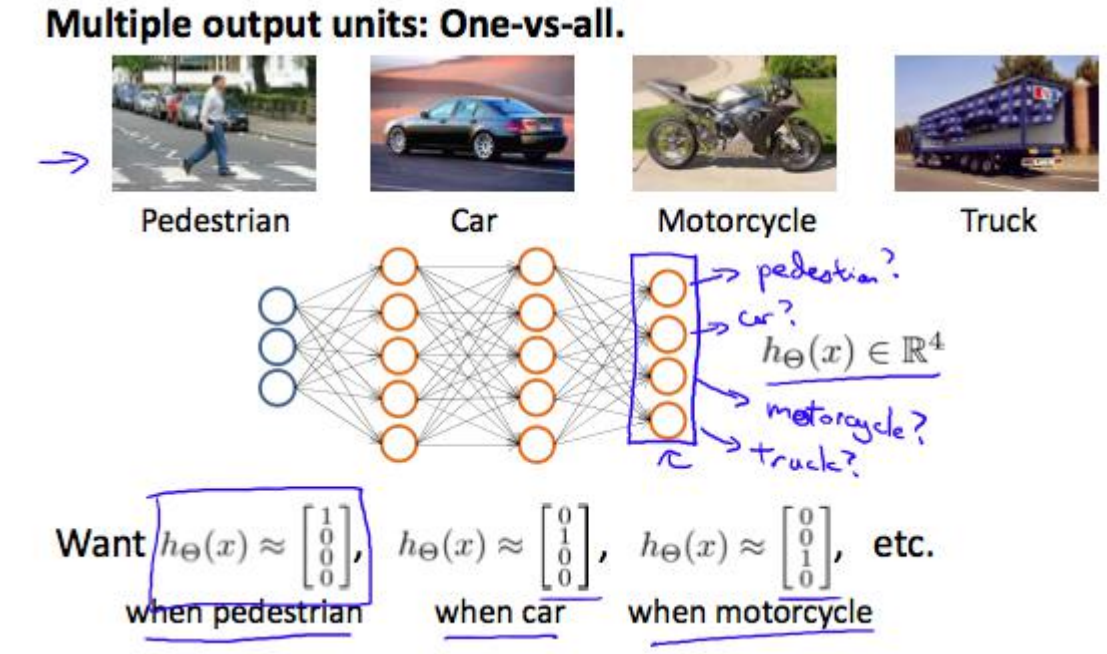


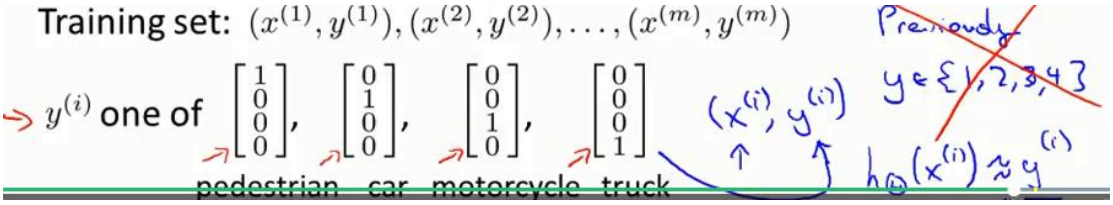
• 同或的搭建　XNOR：同或 XOR：异或



3-3 Multiclass Classification

(多类别分类)





• 图中的例子，h(x)和 y 都是四维向量，h(x)表示是某个事物的概率向量，所以我们希望 h(x)与 y 相等。

4-1 Exercise3

• 让矩阵 A 中大于 2 的数全部变为 1：A(A>2)=1

Week 5 Neural Networks: Learning

1 Cost Function and Backpropagation

• 介绍拟合神经网络参数的算法

1-1 Cost Funtion

• 神经网络有两类(计算层单元不考虑偏置项)

Binary classification(二元分类) y=0 or 1 故输出单元数为 1

Multi-class classification(多元分类) 输出单元一般大于等于 3。

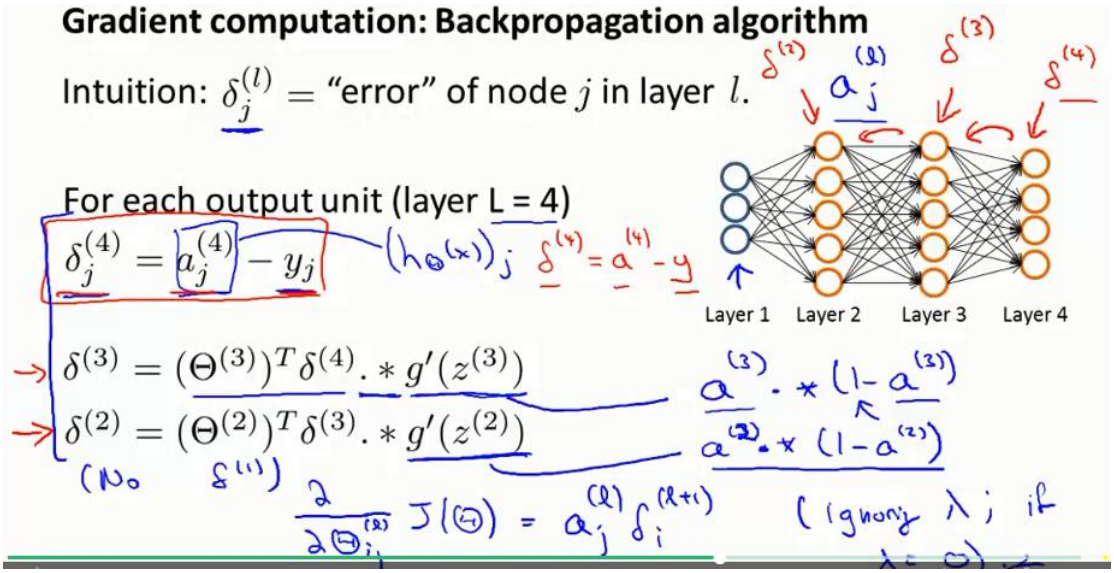
代价函数(第二项不考虑偏置单元)

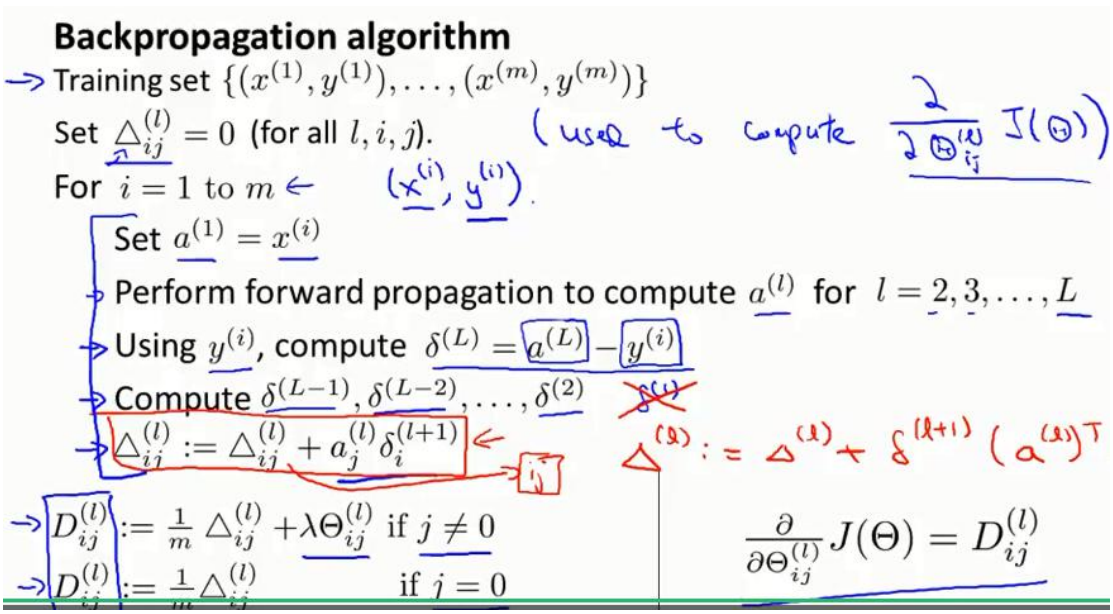
$$J(\Theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^{(i)}\log((h_\Theta(x^{(i)}))_k) + (1-y_k^{(i)})\log(1-(h_\Theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{sl}\sum_{j=1}^{sl+1}(\Theta_{j,i}^{(l)})^2$$

1-2 Backpropagation Algorithm

• 从输出层往前计算偏差项，数学原理相对复杂。

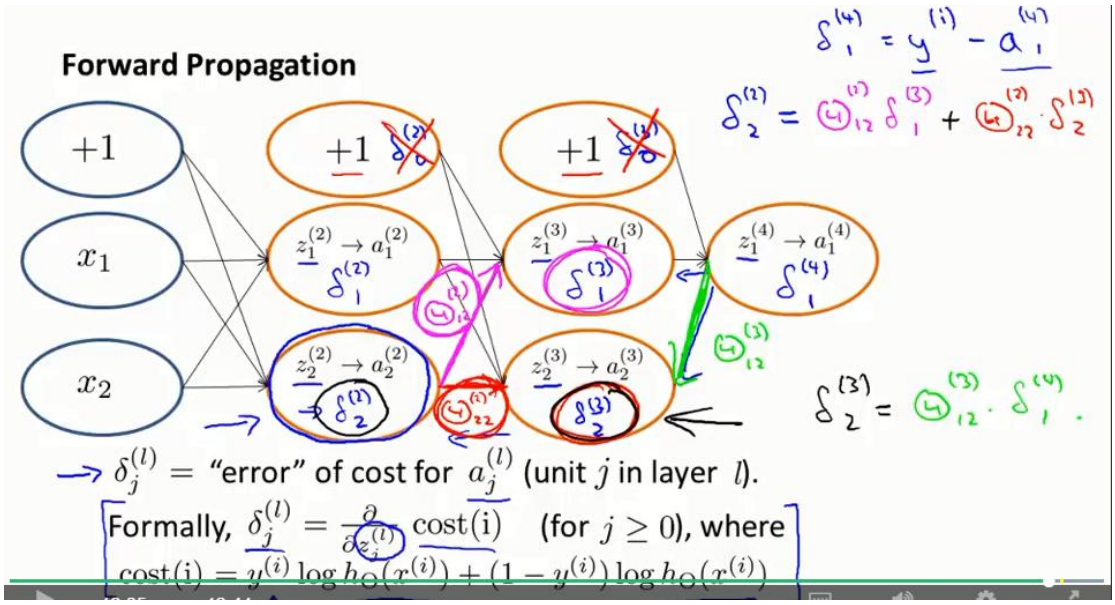
$$\delta^{(l)} = ((\Theta^{(l)})^T\delta^{(l+1)}) .* a^{(l)} .* (1-a^{(l)})$$





补充，只有出现正则化的项才要考虑 theta0 的差异。

1-3 Backprogation Intuition



2 Backpropagation in Practice

2-1 Implementation Note:Unrolling Parameters

• 在之前的线性回归中，theta 是向量，而神经网络中却是矩阵，于是我们需要将矩阵展开成向量的形式。

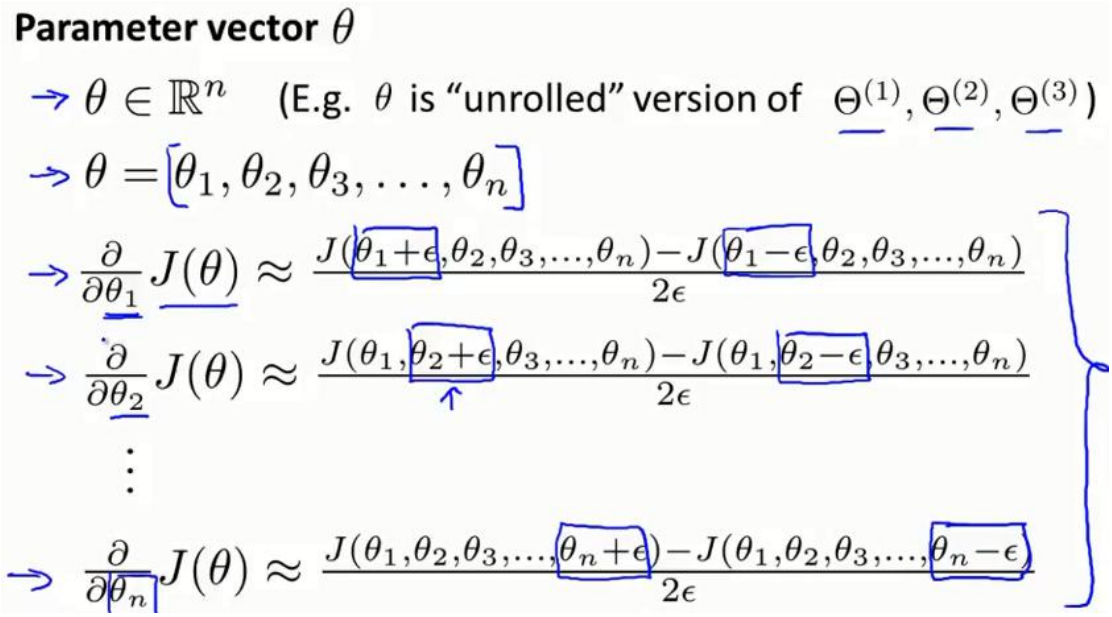
```
thetaVec = [ Theta1(:); Theta2(:); Theta3(:)];
DVec = [D1(:); D2(:); D3(:)];

  Theta1 = reshape(thetaVec(1:110),10,11);
Theta2 = reshape(thetaVec(111:220),10,11);
Theta3 = reshape(thetaVec(221:231),1,11);
```

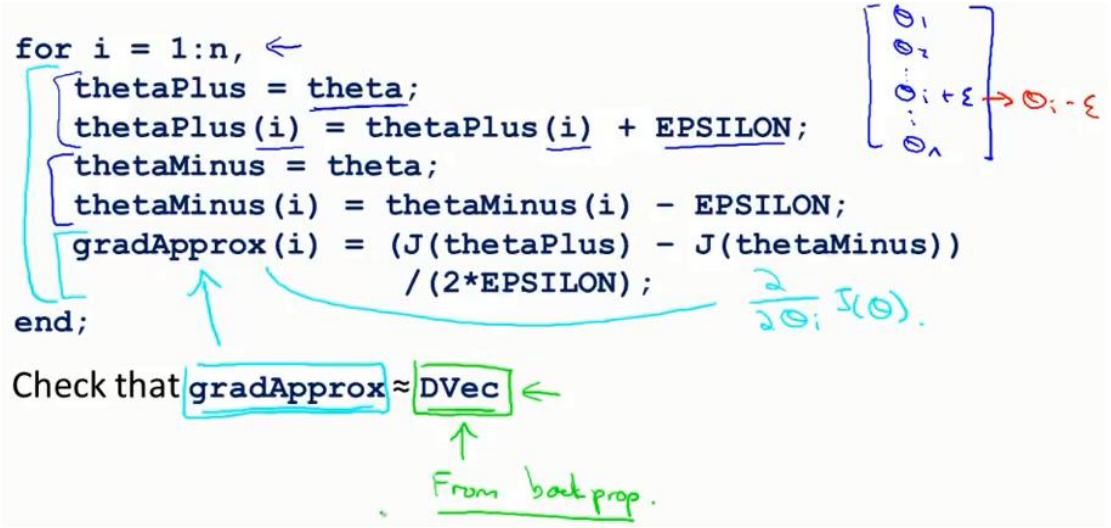
矩阵的好处：方便、充分实现了向量化
向量的好处：某些高级优化算法通常需要向量的形式。

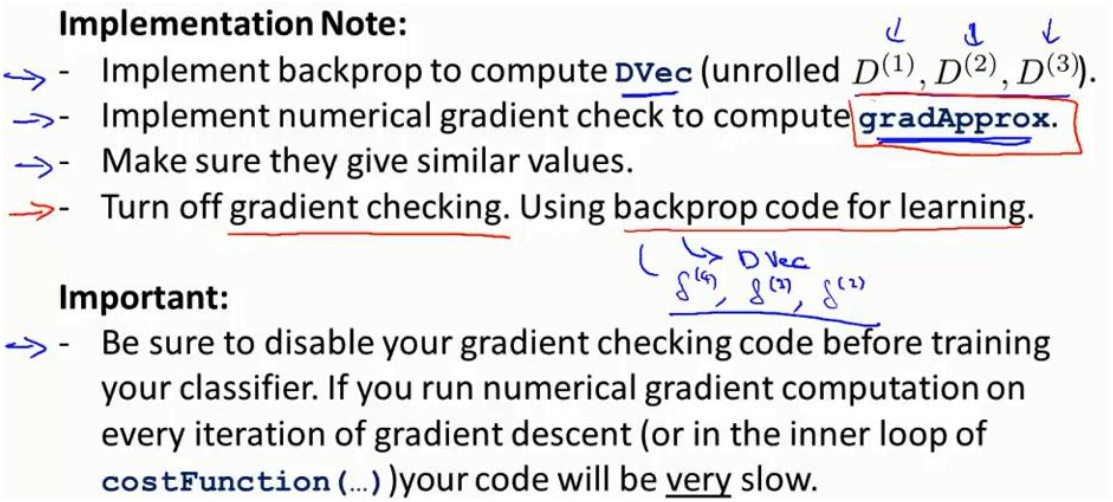
2-2 Gradient Checking

• 因为后向传播比较复杂，往往会有一些 bug 会导致错误，于是要用梯度检验。



• 我们利用微分法来计算梯度的变化量 gradApprox，用它跟反向传播法计算的 DVec 比较，看是否相等。





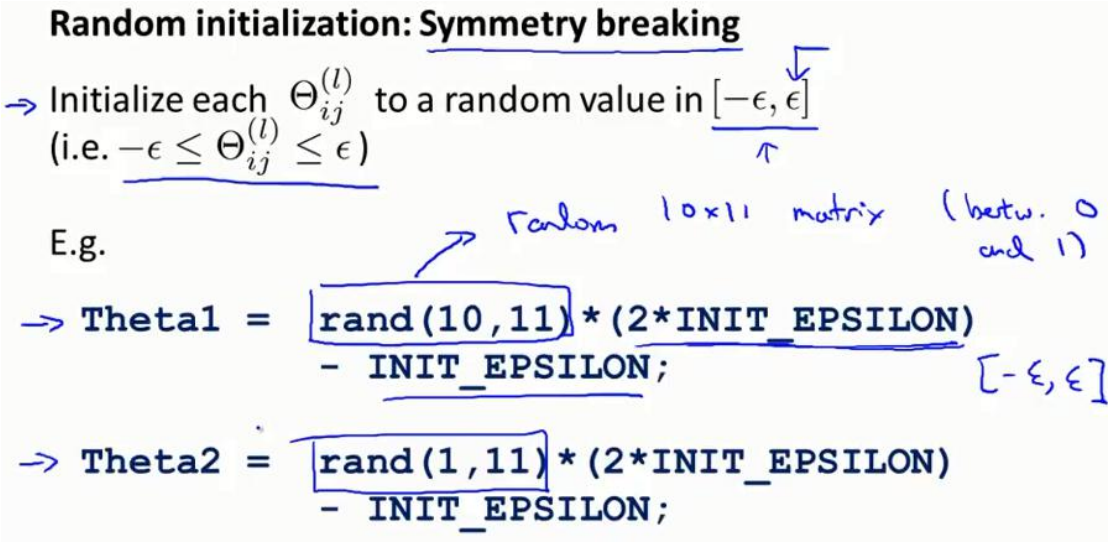
• 步骤：①先用反向传播法计算 DVec ②数值计算出梯度的变化量 gradApprox

$$\frac{\partial}{\partial \Theta_j} J(\Theta) \approx \frac{J(\Theta_1, \ldots, \Theta_j + \epsilon, \ldots, \Theta_n) - J(\Theta_1, \ldots, \Theta_j - \epsilon, \ldots, \Theta_n)}{2\epsilon}$$

③确保两个值相同 ④一定要关掉数值计算方法(其缺点就是计算很慢)

2-3 Random Initialization

• 在神经网络中若初始化参数矩阵为零，将无法正常工作，因为参数将会相同更新，于是我们有必要随机初始化参数矩阵。



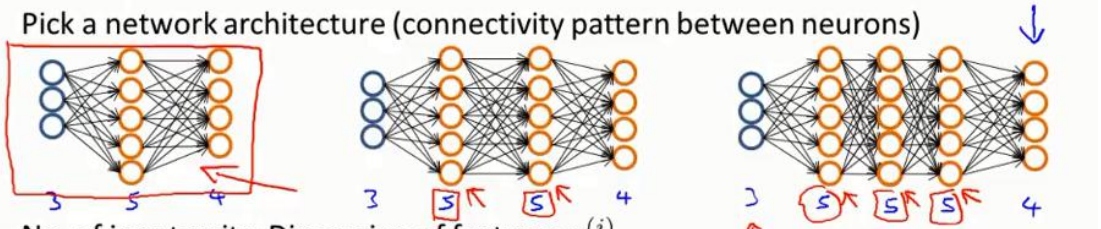
2-4 Putting It Together

(总体回顾)

• **第一步 搭建模型框架** 一般输入层与输出层由条件决定，隐藏层最好一层 or 每层隐藏层拥有相同的节点数。

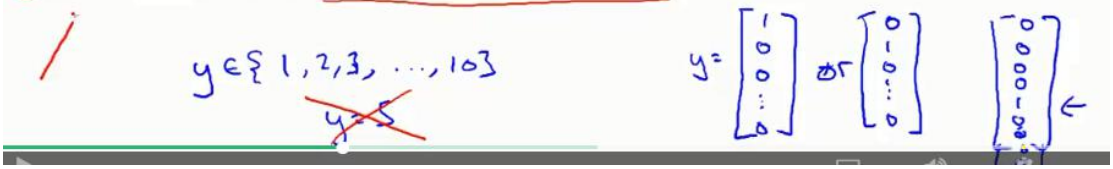
Training a neural network

Pick a network architecture (connectivity pattern between neurons)



- No. of input units: Dimension of features $x^{(i)}$
- No. output units: Number of classes

Reasonable default: 1 hidden layer, or if >1 hidden layer, have same no. of hidden units in every layer (usually the more the better)



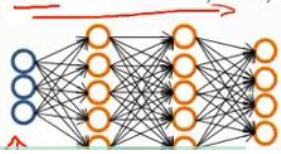
- 第二步 实现训练过程（共六步）

Training a neural network

1. Randomly initialize weights
2. Implement forward propagation to get $h_\Theta(x^{(i)})$ for any $x^{(i)}$
3. Implement code to compute cost function $J(\Theta)$
4. Implement backprop to compute partial derivatives $\frac{\partial}{\partial \Theta_{jk}^{(l)}} J(\Theta)$

for i = 1:m {

- Perform forward propagation and backpropagation using example $(x^{(i)}, y^{(i)})$
  (Get activations $a^{(l)}$ and delta terms $\delta^{(l)}$ for $l = 2, \ldots, L$).



Training a neural network

5. Use gradient checking to compare $\frac{\partial}{\partial \Theta_{jk}^{(l)}} J(\Theta)$ computed using backpropagation vs. using numerical estimate of gradient of $J(\Theta)$.
   - Then disable gradient checking code.
6. Use gradient descent or advanced optimization method with backpropagation to try to minimize $J(\Theta)$ as a function of parameters Θ

1 随机初始化参数矩阵
2 使用前向传播算法计算出预测值。
3 计算出代价 J。
4 使用反向传播算法计算梯度
5 比较数值计算的梯度来检验是否正确（**使用后需要关闭**）
6 使用高级优化算法来最小化代价 J。

3-1 Exercise4
• 计算神经网络的代价值，**好好体会为什么要用点乘，也可以用 for 循环。参见 http://blog.csdn.net/artprog/article/details/51468943**

```
z2 = [ones(m,1) X]*Theta1';
a2 = sigmoid(z2);
z3 = [ones(m,1) a2]*Theta2';
hx = sigmoid(z3);

Y = zeros(m,10);

for i = 1:m
    Y(i,y(i)) = 1;
end

J = 1/m *sum(sum(-Y.*log(hx)-(1-Y).*log(1-hx)));
%To understand why we use ".*" instead of "*".
```

Week 6 Advice for Applying Machine Learning

1 Evaluating a Learning Algorithm

1-1 Machine learning diagnostic

(评估机器学习算法的性能)
例如我们发现住房问题出现了较大的误差，我们该如何解决呢？有几下几种方式该如何选取。

Debugging a learning algorithm:
Suppose you have implemented regularized linear regression to predict housing prices.

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

However, when you test your hypothesis on a new set of houses, you find that it makes unacceptably large errors in its predictions. What should you try next?

- Get more training examples
- Try smaller sets of features
- Try getting additional features
- Try adding polynomial features $(x_1^2, x_2^2, x_1x_2, \text{etc.})$
- Try decreasing λ
- Try increasing λ

1-2 Evaluating a Hypothesis

(使用学过的算法来评估假设函数)

• 如何评估假设函数过拟合 or 欠拟合，在不能画图的情况下。

1 首先我们将数据集按照 7：3 分成**训练集**与**测试集。（最好是随机打乱的）**

2 通过训练集最小化代价函数 J，得到 Theta。

3 将 Theta 代入测试集来使用，然后来计算误分率。

The test set error

1. For linear regression: $J_{test}(\Theta) = \frac{1}{2m_{test}}\sum_{i=1}^{m_{test}}(h_\Theta(x_{test}^{(i)}) - y_{test}^{(i)})^2$

2. For classification ~ Misclassification error (aka 0/1 misclassification error):

$$err(h_\Theta(x), y) = \begin{matrix} 1 & \text{if } h_\Theta(x) \geq 0.5 \text{ and } y = 0 \text{ or } h_\Theta(x) < 0.5 \text{ and } y = 1 \\ 0 & otherwise \end{matrix}$$

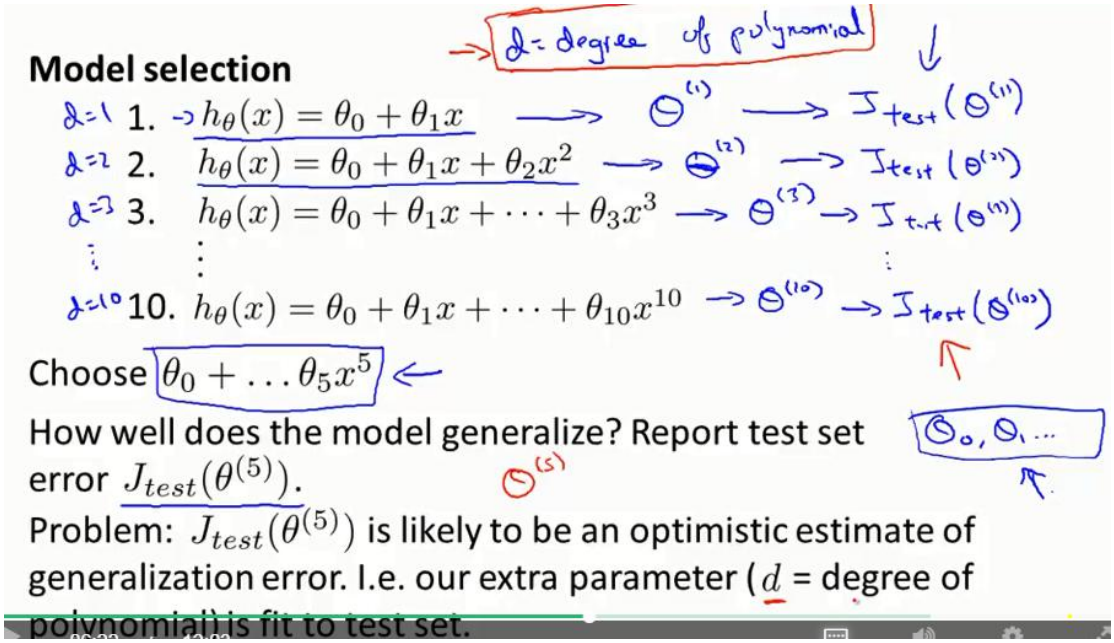
This gives us a binary 0 or 1 error result based on a misclassification. The average test error for the test set is:

$$\text{Test Error} = \frac{1}{m_{test}}\sum_{i=1}^{m_{test}} err(h_\Theta(x_{test}^{(i)}), y_{test}^{(i)})$$

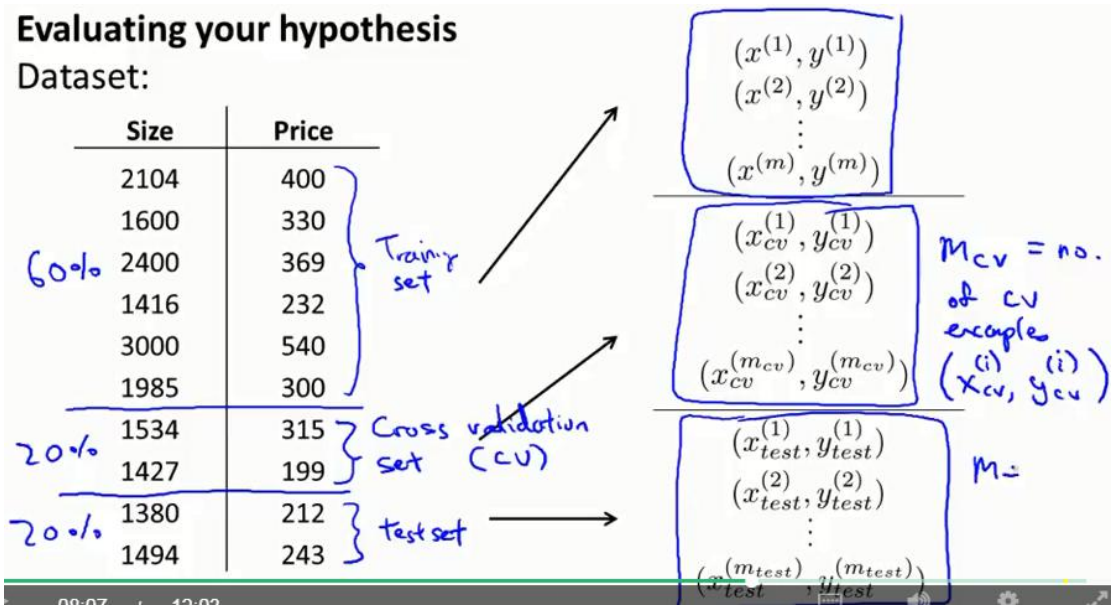
This gives us the proportion of the test data that was misclassified.

1-3 Model Selection and Train/Validation/TestSets

• 假设我们需要选择多项式模型的次数 d，d 从 1-10，我们该如何选择呢？我们按照前面的方法算出了对应的参数 Theta，然后使用该参数对测试集进行计算代价，使用代价最小的参数，**但是这样仍不能说明这组参数是最适用与新样本的。**



• 故我们将数据分成**训练集、交叉验证集、测试集**，比例分别为 60%、20%、20%。



1. Optimize the parameters in Θ using the training set for each polynomial degree.
2. Find the polynomial degree d with the least error using the cross validation set.
3. Estimate the generalization error using the test set with $J_{test}(\Theta^{(d)})$, (d = theta from polynomial with lower error);

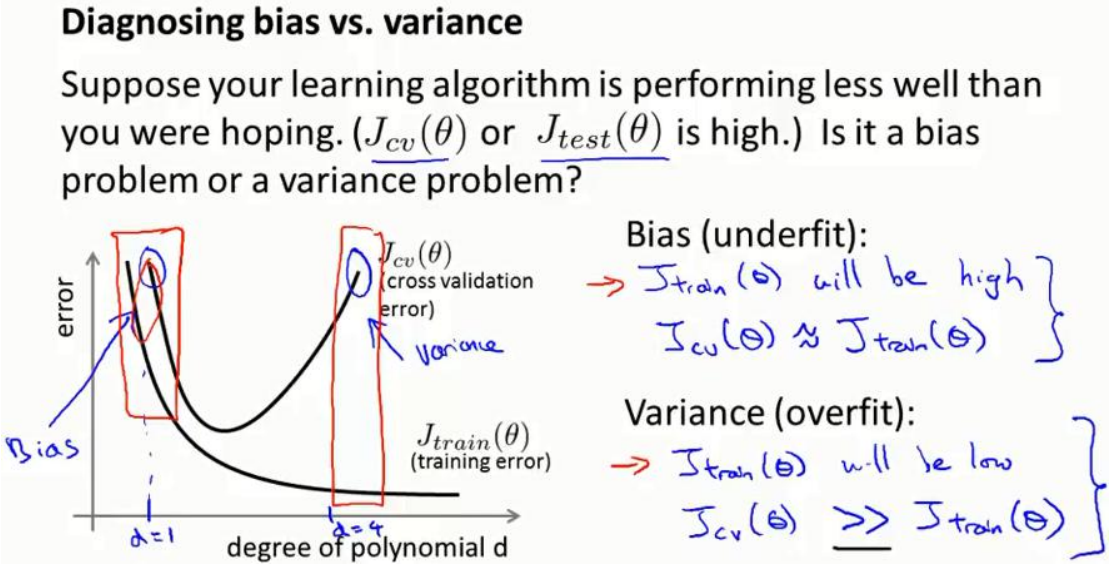
1 先用训练集获得多项式模型的参数。
2 使用交叉验证集选择代价最小的参数。
3 使用测试集来获得最小的参数。

2 Bias vs. Variance

2-1 Diagnosing Bias vs Variance

欠拟合/偏差和过拟合/方差)

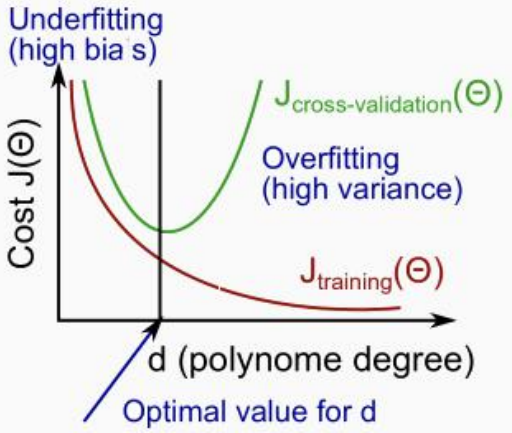
• 判断一个算法是偏差有问题还是方差 观察 Jtrain 与 Jcv 两种数值的关系。



High bias (underfitting): both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ will be high. Also, $J_{CV}(\Theta) \approx J_{train}(\Theta)$.

High variance (overfitting): $J_{train}(\Theta)$ will be low and $J_{CV}(\Theta)$ will be much greater than $J_{train}(\Theta)$.

The is summarized in the figure below:

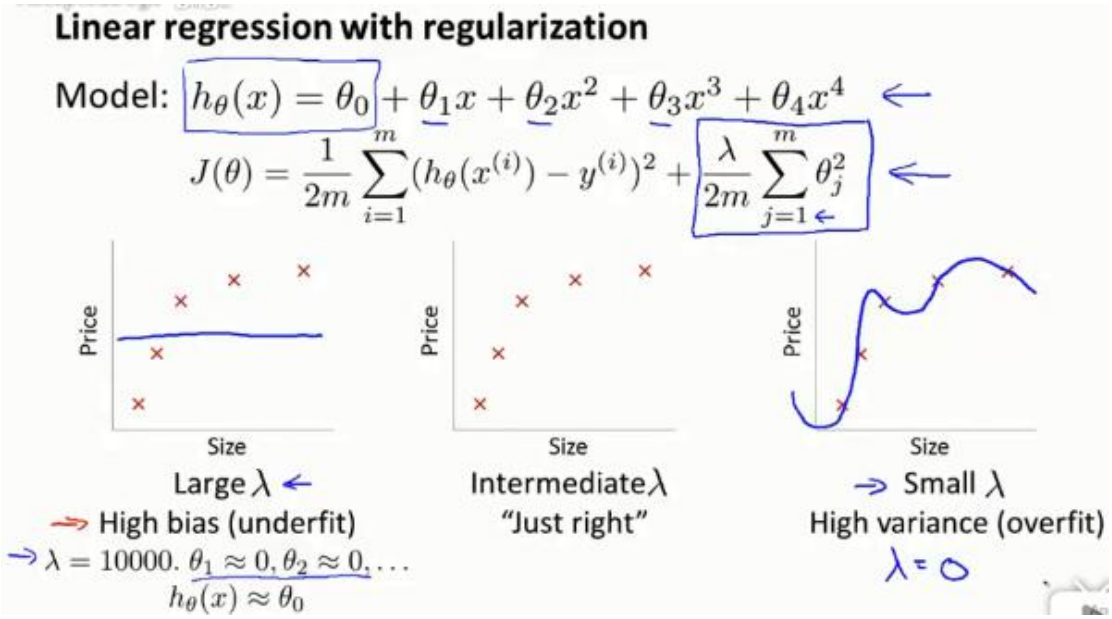


2-2 Regularization and Bias/Variance

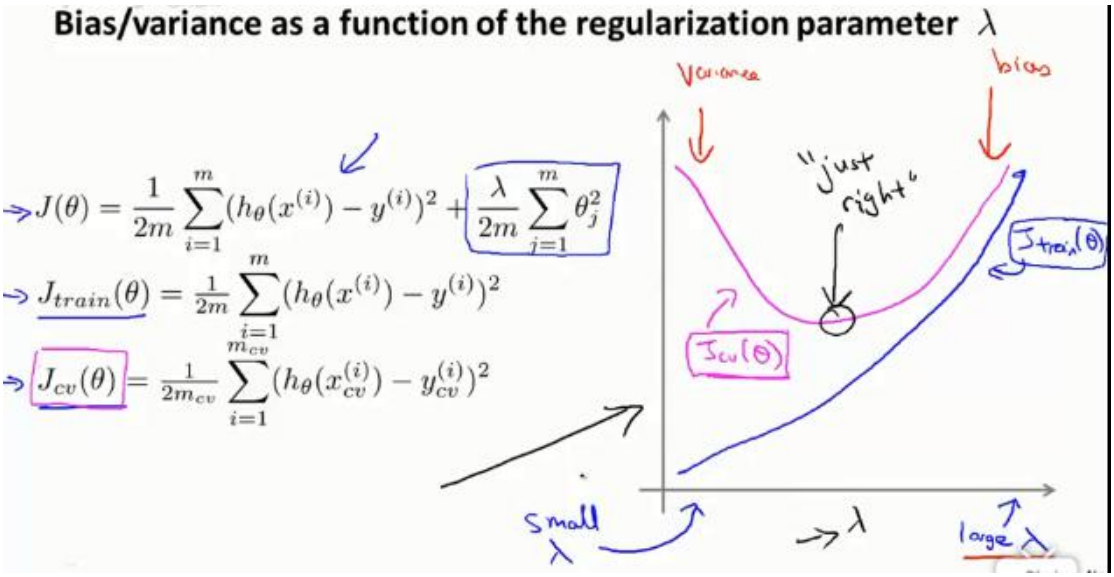
正则化与偏差/方差的关系

• 当 lambda 几乎为 0 时，模型 h=theta(0)，为一条直线，此时欠拟合，偏差很大。

当 lambda 过大时，过拟合，方差很大。



• 做出 Jtrain、Jcv 的曲线随着 lambda 变化的曲线图。



• 具体步骤

1. Create a list of lambdas (i.e. λ∈{0,0.01,0.02,0.04,0.08,0.16,0.32,0.64,1.28,2.56,5.12,10.24});
2. Create a set of models with different degrees or any other variants.
3. Iterate through the λs and for each λ go through all the models to learn some Θ.
4. Compute the cross validation error using the learned Θ (computed with λ) on the $J_{CV}(\Theta)$ **without** regularization or λ = 0.
5. Select the best combo that produces the lowest error on the cross validation set.
6. Using the best combo Θ and λ, apply it on $J_{test}(\Theta)$ to see if it has a good generalization of the problem.

2-3 Learning curves

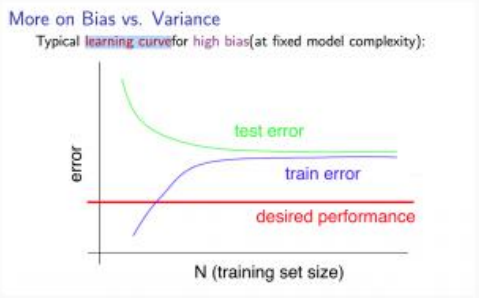
(样本容量与偏差/方差的关系)

• 高偏差下，Jtrain 近似于 Jcv，增加样本容量将不会有太大帮助，因为 Jtrain 与 Jcv 将会趋于一个值。

Low training set size: causes $J_{train}(\Theta)$ to be low and $J_{CV}(\Theta)$ to be high.

Large training set size: causes both $J_{train}(\Theta)$ and $J_{CV}(\Theta)$ to be high with $J_{train}(\Theta) \approx J_{CV}(\Theta)$.

If a learning algorithm is suffering from **high bias**, getting more training data will not **(by itself)** help much.

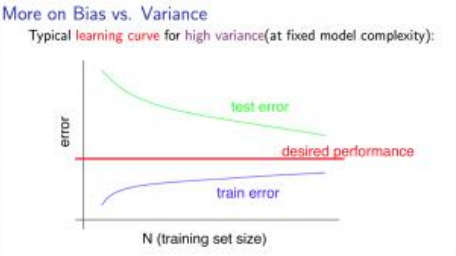


• 高方差下，Jtrain 很小，Jcv 很大，Jcv 一直大于 Jtrain，增加样本容量有帮助，Jtrain 与 Jcv 将会不断地靠近。

Low training set size: $J_{train}(\Theta)$ will be low and $J_{CV}(\Theta)$ will be high.

Large training set size: $J_{train}(\Theta)$ increases with training set size and $J_{CV}(\Theta)$ continues to decrease without leveling off. Also, $J_{train}(\Theta) < J_{CV}(\Theta)$ but the difference between them remains significant.

If a learning algorithm is suffering from **high variance**, getting more training data is likely to help.



2-4 Deciding What to do Next Revisited

• 以下的方式对哪种情况有效

1 Getting more training examples: Fixes high variance

2 Trying smaller sets of features: Fixes high variance

3 Adding features: Fixes high bias

4 Adding polynomial features: Fixes high bias

5 Decreasing λ : Fixes high bias

6 Increasing λ : Fixes high variance.

• 如何诊断神经网络
1 参数太少，欠拟合，计算量小。
2 参数太多，过拟合，计算量大，可以使用正则化来解决。

Diagnosing Neural Networks

- A neural network with fewer parameters is **prone to underfitting**. It is also **computationally cheaper**.
- A large neural network with more parameters is **prone to overfitting**. It is also **computationally expensive**. In this case you can use regularization (increase λ) to address the overfitting.

• 多项式
低次项高偏差、低方差，不能很好地拟合数据。
高次项低偏差、高方差。

3-1 ex5
• 记住了，在正则化的情况下。
使用训练集计算 theta，需要代入 lambda。
计算 Jtrain、Jval 不需要代入 lambda。

Week 6 Machine Learning System Design

3 Building a Spam Class

3-1 Prioritizing What to Work On

头脑风暴，想象出多个方法。
• 收集大量数据
• 开发复杂的功能
• 开发不同方式的算法

- Collect lots of data (for example "honeypot" project but doesn't always work)
- Develop sophisticated features (for example: using email header data in spam emails)
- Develop algorithms to process your input in different ways (recognizing misspellings in spam).

很难说哪种方法更好。

3-2 Error Analysis

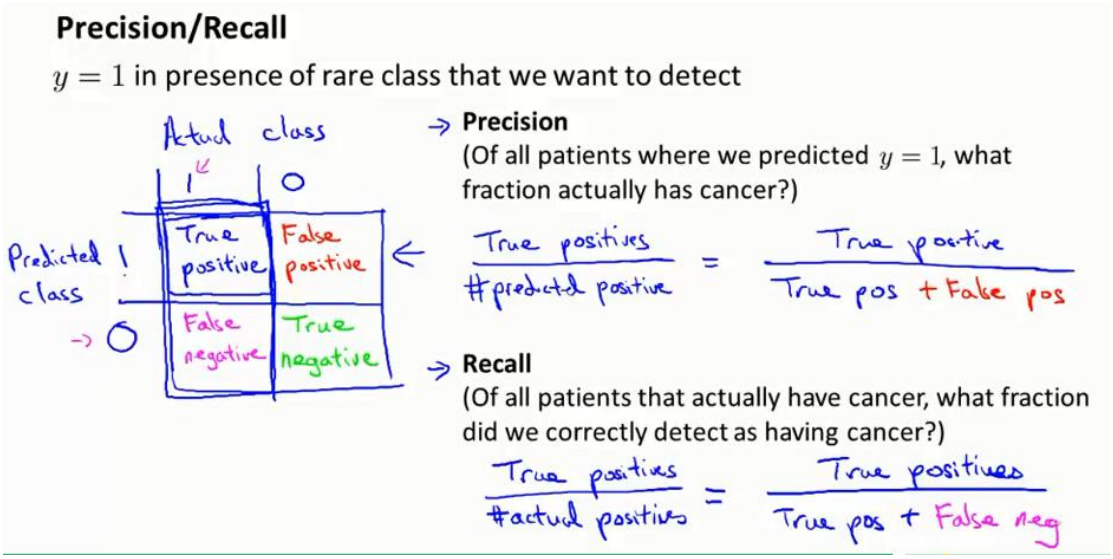
解决机器学习问题的方法：
• 从一个简单的算法开始，快速实施它，并尽早在交叉验证数据上进行测试。
• 绘制学习曲线以确定更多数据，更多功能等是否可能有所帮助。
• 手动检查交叉验证集中示例的错误，并尝试找出发生大多数错误的趋势。

Important 保证有一种**量化的数值计算**的方法来评估算法。
Conclusion 不要耗费太多时间在构建算法上，简单地实现算法后，再来改进。

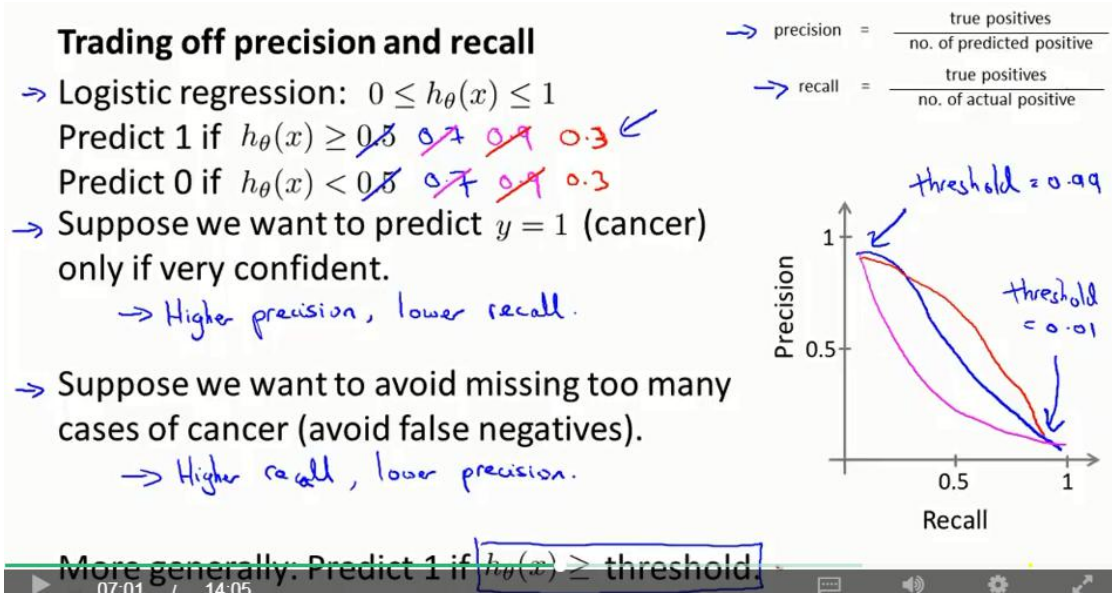
4 Handling Skewed Data

4-1 Error Metrics for Skewed Classes

(斜偏类)
一个类中的样本数比另一个类的数据相比大很多，如某一个算法的准确率是 99.5%，但是他的算法却是一直判断为常数。
precision 查准率 recall 召回率 两者都是越高越好。
Y=1 代表之比较稀少的分类，如患肿瘤的样本。



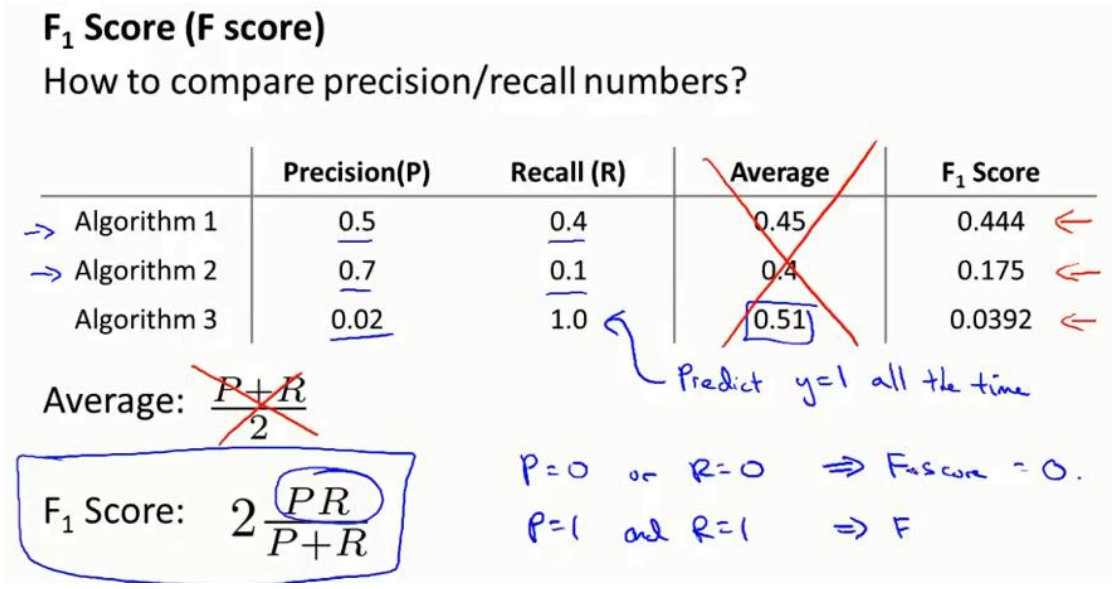
4-2 Trading Off Precision and Recall



- 提升判断癌症的准确率，将阈值定为 0.9，**高准确率、低召回率模型。**
- 担心患有癌症，却没有接受治疗。将阈值定为 0.3，**高召回率、低准确率模型。**

- Accuracy = (true positives + true negatives) / (total examples)
- Precision = (true positives) / (true positives + false positives)
- Recall = (true positives) / (true positives + false negatives)
- F_1 score = (2 * precision * recall) / (precision + recall)

用评估度量值来权衡准确率与召回率。

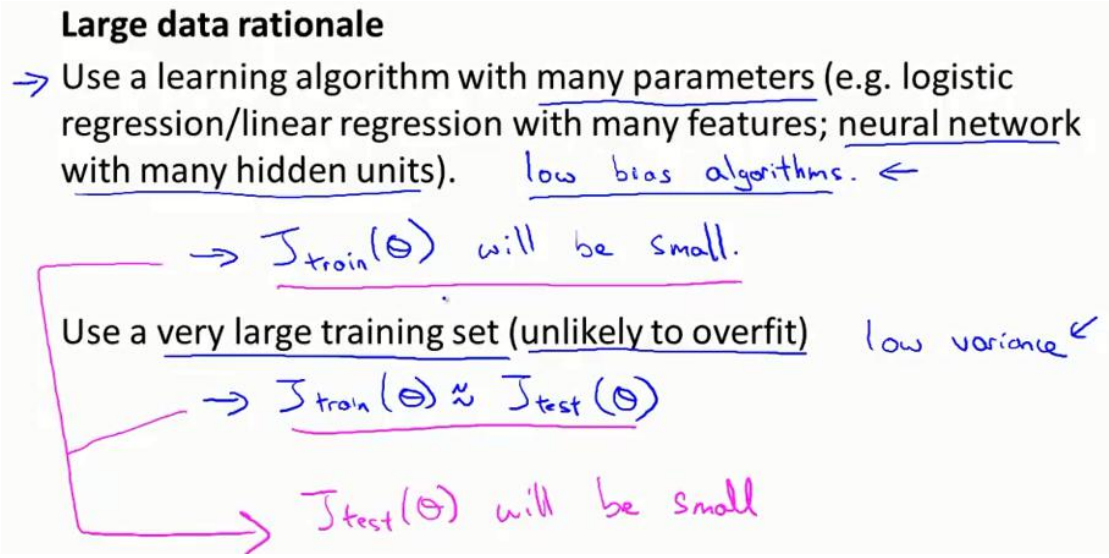


- 一般不能用平均值来权衡，因为平均值最高的也可能是最不实际的。所以我们定义了一个 **F 值**来计算两者的关系。
- 在交叉验证集上跑出最好的结果。

5 Using Large Data Sets

5-1 Data For Machine Learning

取得成功并不是拥有最好的算法，而是拥有最多数据的人。
• 大数据的合理性：假设特征拥有足够的信息去预测结果。例如只给出了房子的尺寸，即使是房地产商也无法判断；给出了一个英语句子，会英语的人可以判断填什么词。
• 庞大的数据可以避免过度拟合。



• **使用了大量的特征参数**（高方差、低偏差）+**大量的数据**（避免过度拟合）=良好的算法。例如，我们先找某一领域的专家得出可以预测结果相关的特征参量，然后寻找大量的数据从而建立性能良好的算法。

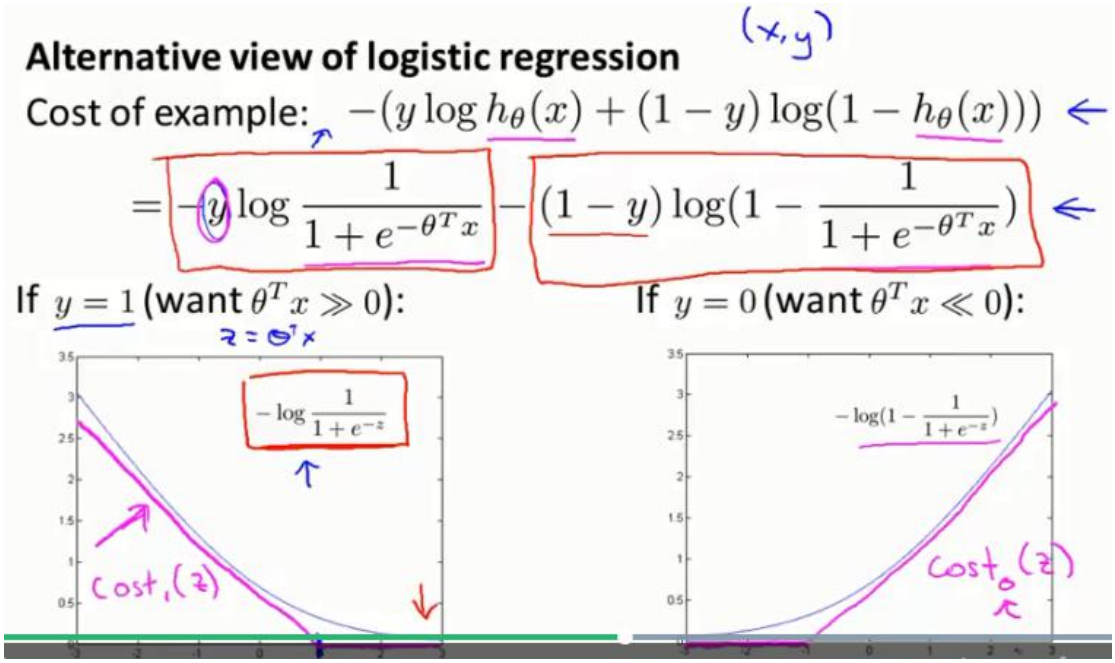
习题分析 http://blog.csdn.net/a1015553840/article/details/50781173

Week 7 Support Vector Machines

1 Large Margin Classification

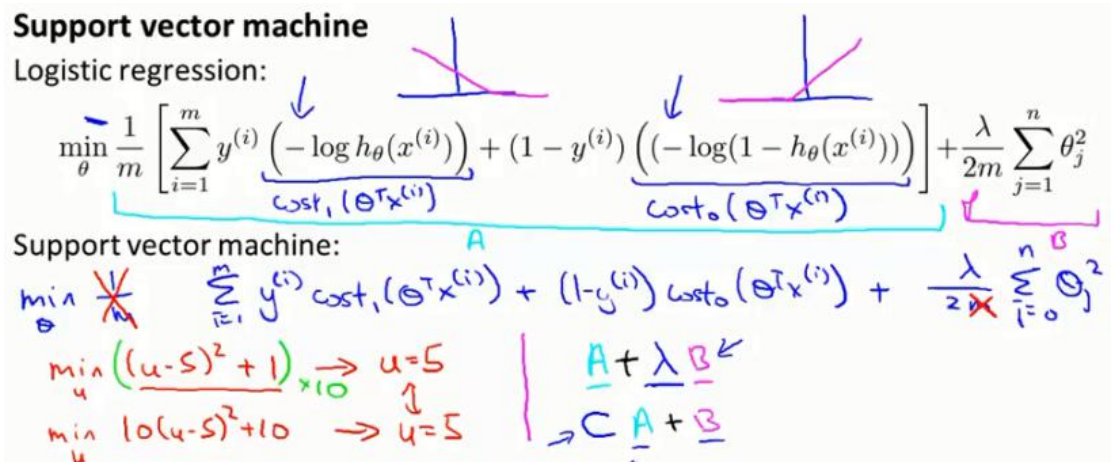
1-1 Optimization Objective

• 代价函数先不看 1/m 这个系数，把它展开，然后分类讨论。当 y=0 or 1 时，对应的曲线下图左右，我们定义为 cost0、cost1。



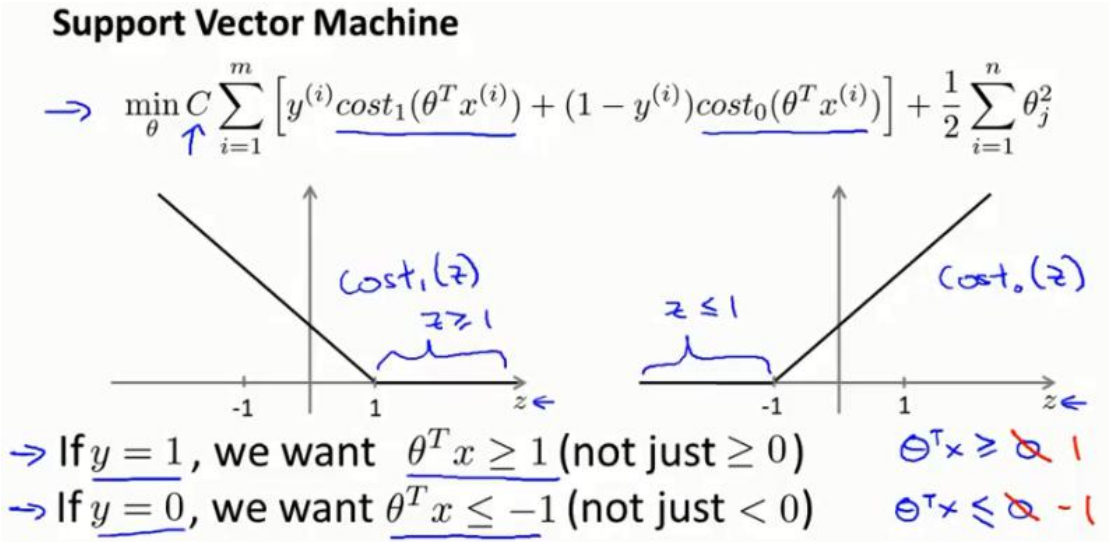
1 考虑系数 1/m 与否是不会影响优化目标的。

2 原始的代价函数可以分为 A 与 B 两项，我们用 A+lambda*B 来权衡这两项，或者用其他表达式。C 代表了正则化参数。

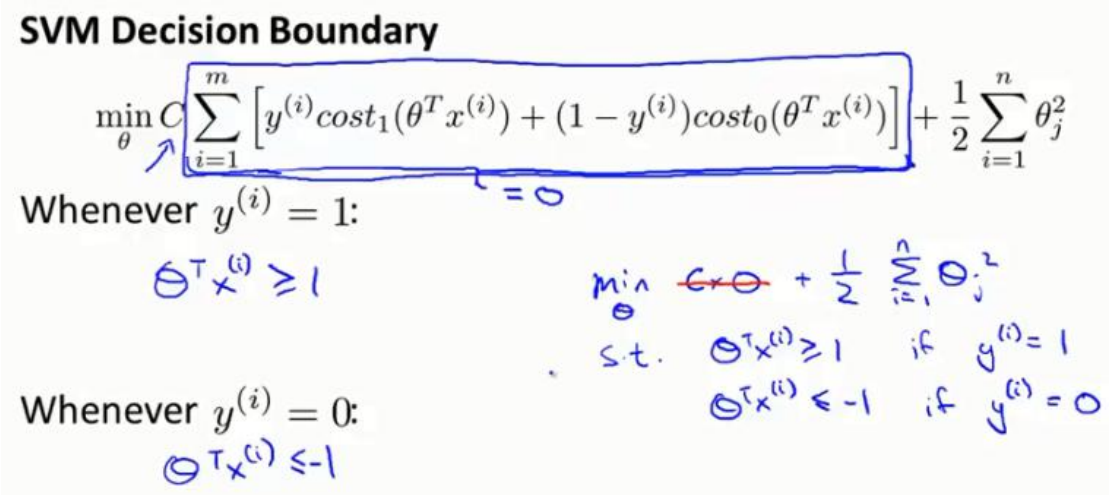


1-2 Large Margin Intuition

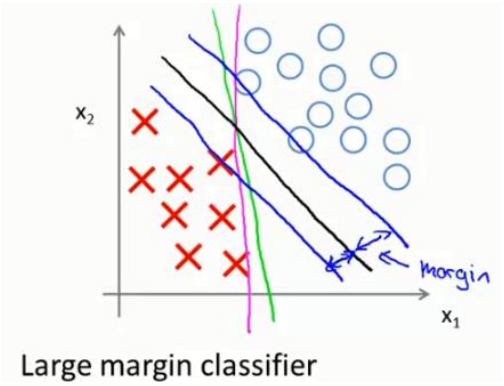
(大间距分类器)



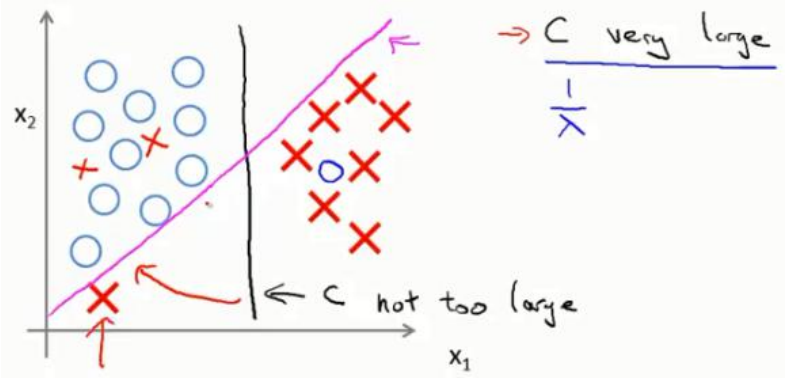
• 逻辑回归只需要满足大于或者小于 0，但是 SVM 的要求更严苛，要大于 1 或者小于-1，这个叫安全间距。



• 当 C 设置为非常大的数如 100,000 时，为了最小化代价，A 项将会变为 0，将发生在以下两个条件。Z 大于 1&y=1 或 Z 小于-1&y=0；



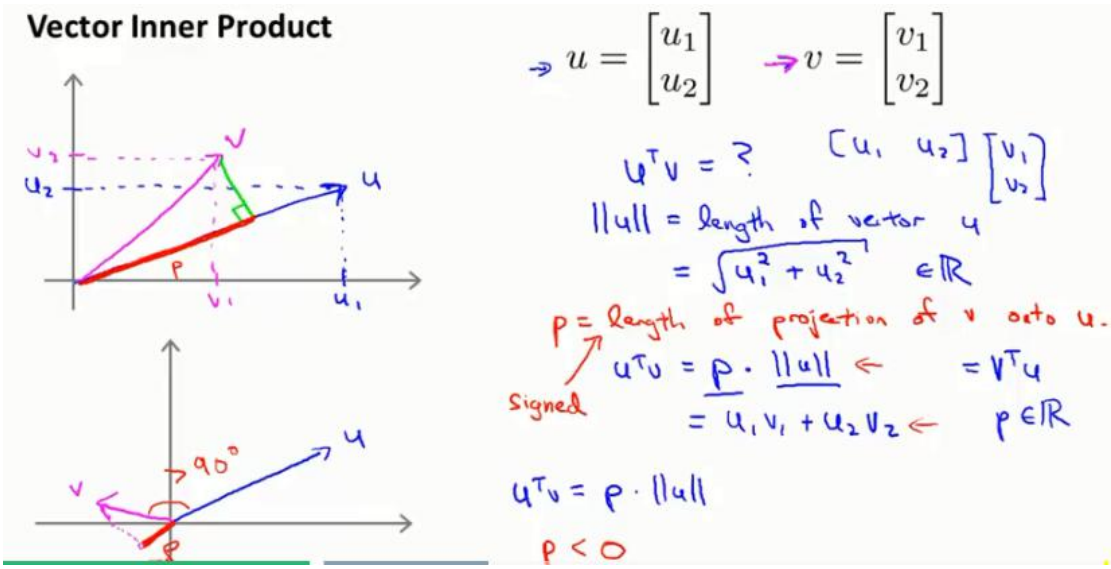
• 黑色的线为支持向量机的决策界，它具有最大的间距，跟蓝色的线之间，这也是它具有鲁棒性的原因。



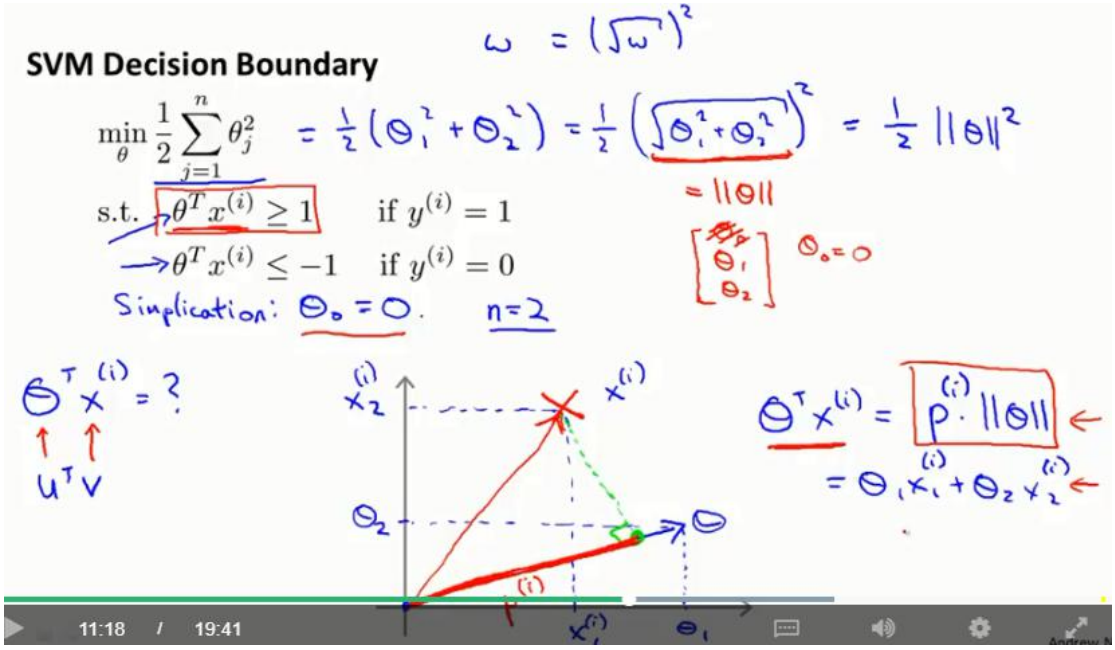
• 当 C 非常大的时候，将具有最大的间距，如图中的黑线。但是如果出现一个异常点如左下角，此时会导致黑线变为粉红线。这是不明智的，所以我们通常不要将 C 设置过大，就算数据是非线性的也可以描述。

1-3 Mathematics Behind Large Margin Classification

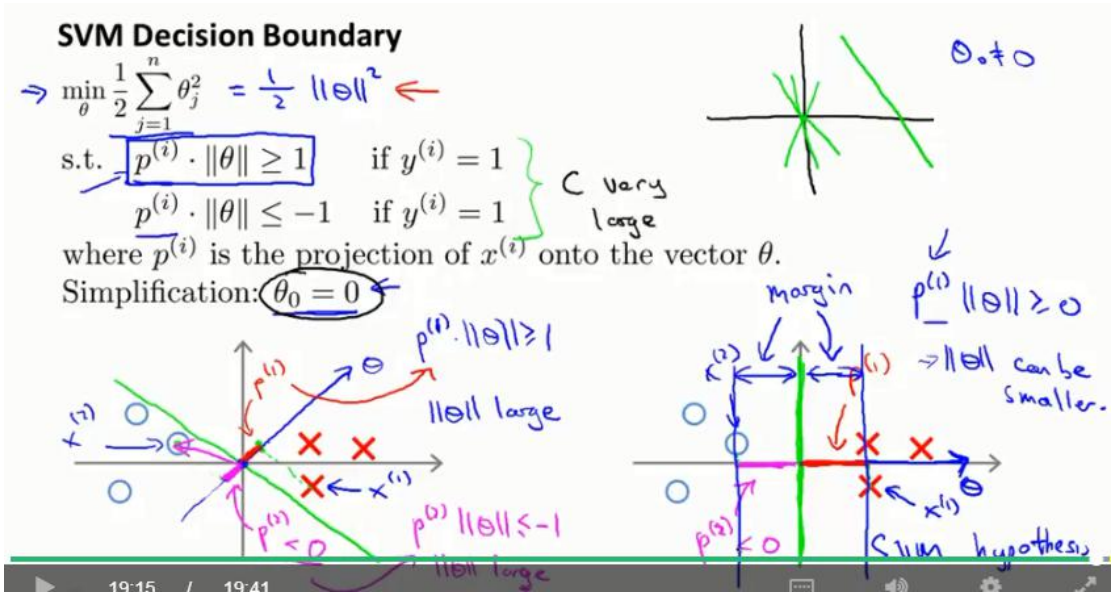
• 向量的内积等于一个向量在另一个向量上的投影 p（p 有正有负）与另一个向量模的乘积。**uTv 的 p 是 v 在 u 上的投影。**



• 假如只有两项，忽略 Theta0，那么 Theta 是一个向量，分量可由 Theta1、Theta2 表示。则说明 $\theta^T x^{(i)} \geq 1$ 可由 $p^{(i)} \cdot \|\theta\|$ 来表示。



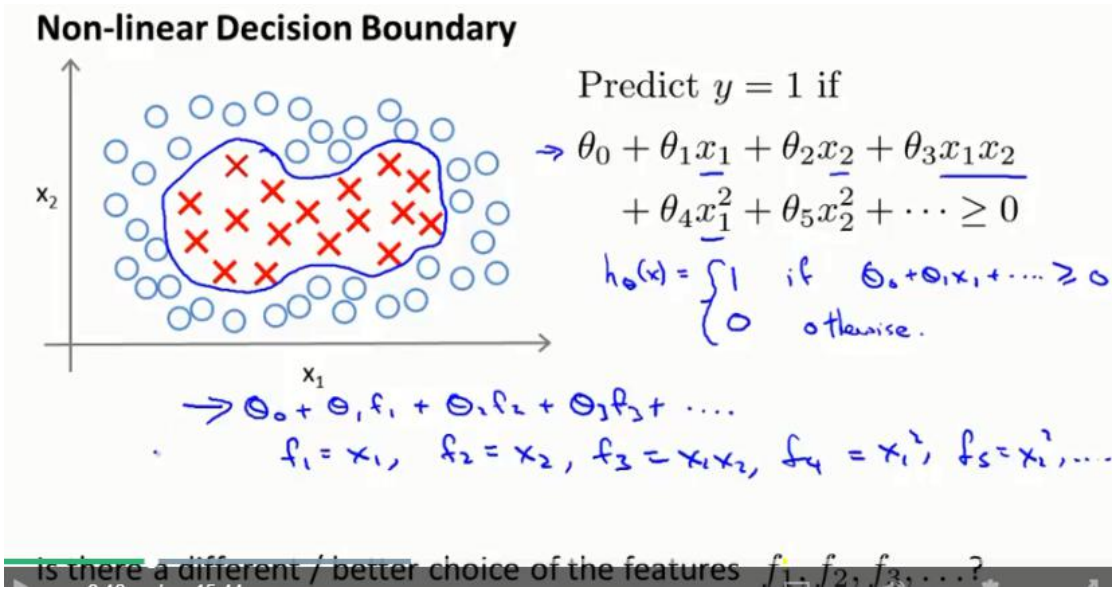
• 在前面的前提（C 非常大）的情况下，假设 Theta0 = 0，即表示决策边界过原点，一条决策边界对应一个 Theta，P(i)则表示 x 在 Theta 上的投影，有正有负。当数据很靠近绿线时，p 很小而 Theta 却很大，这与我们的优化目标矛盾。这正是 SVM 为什么会倾向于选择间距大的决策边界。

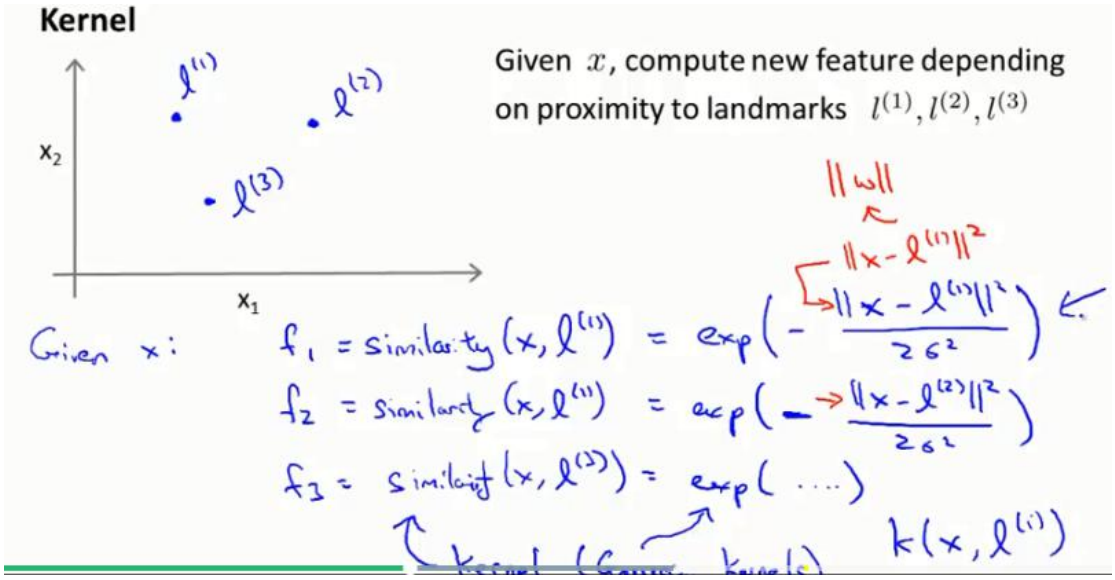


2 Kernels

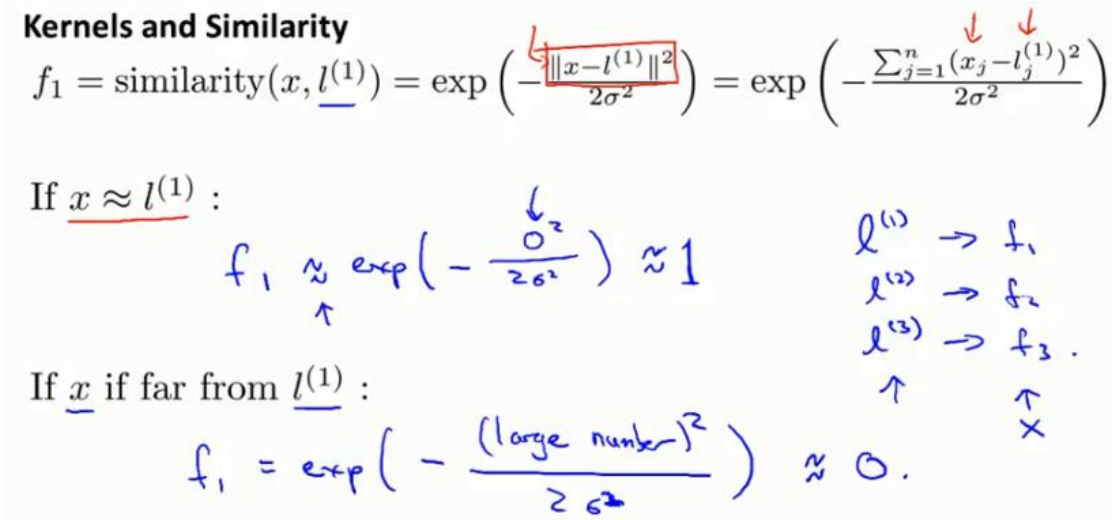
2-1 Kernels I

• 当我们面对一个非常复杂的非线性边界时，我们采用多项式回归，但是多项式非常复杂，我们便使用 f1、f2、f3 来替代他们，但我们仍然不知道多项式该怎么取。

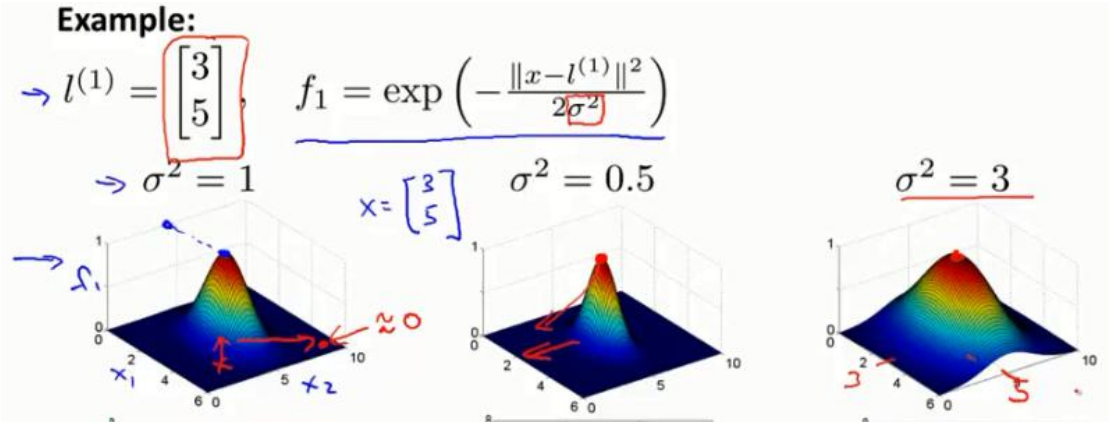




• 我们定义 f1 为相似度，即给定了标志 l，我们使用核函数即高斯函数来算出 f1。

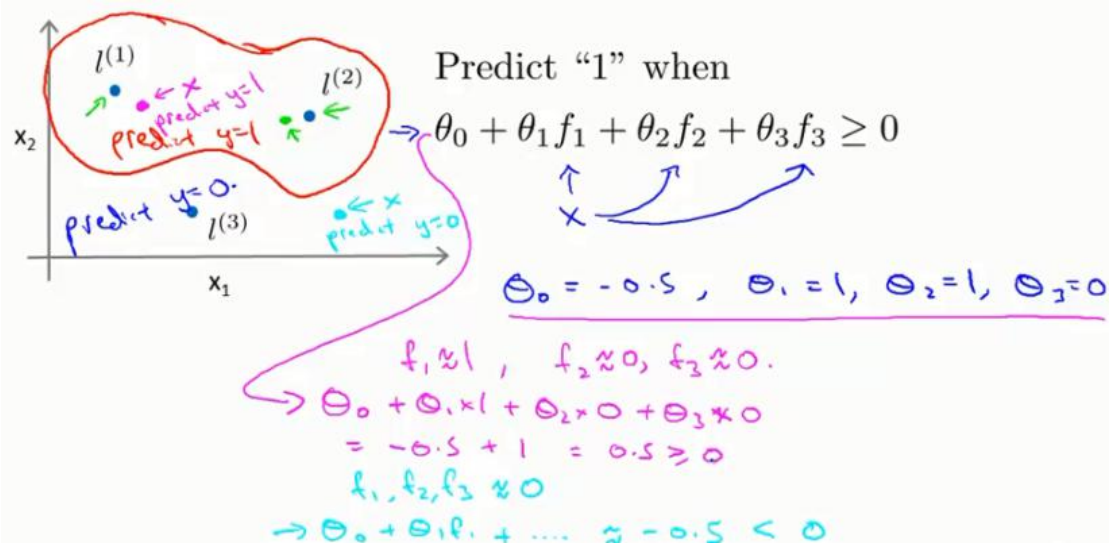


• 用图形化来表示，分母越大，衰减越平缓。



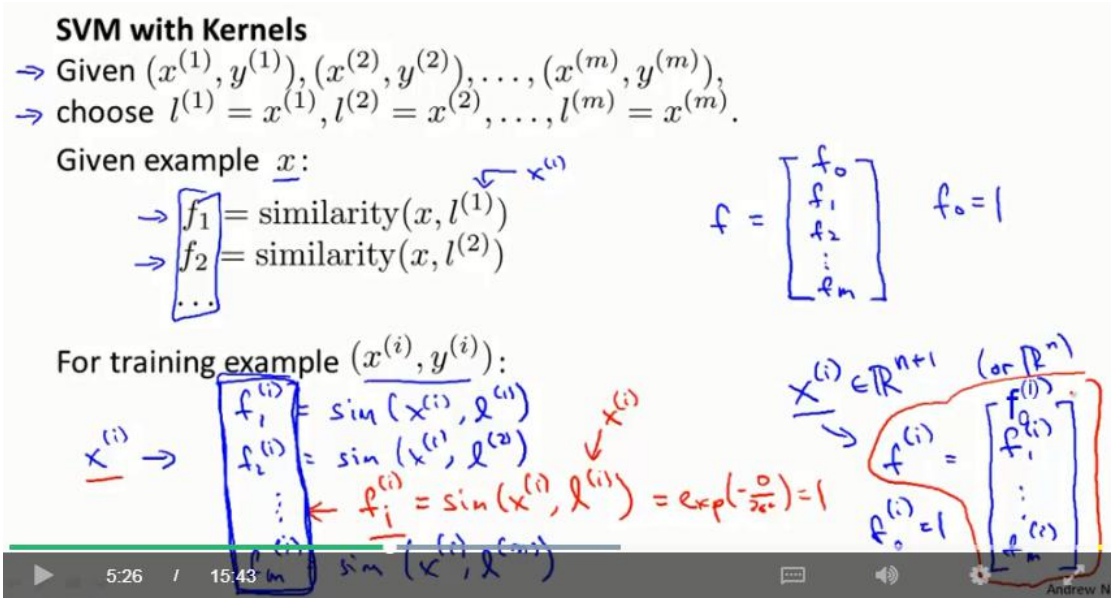
• 通过标记点和核函数来训练非常复杂的非线性边界。

如下，我们定义了一个函数，确定了 Theta 值，由前可知，当 X 接近于 l1、l2 时，f 相似度输出为 1，最终结果为 1；当 X 远离 l1、l2 时，最终结果为 0；由此我们可以做出红色的曲线，即决策边界。



2-2 Kernels II

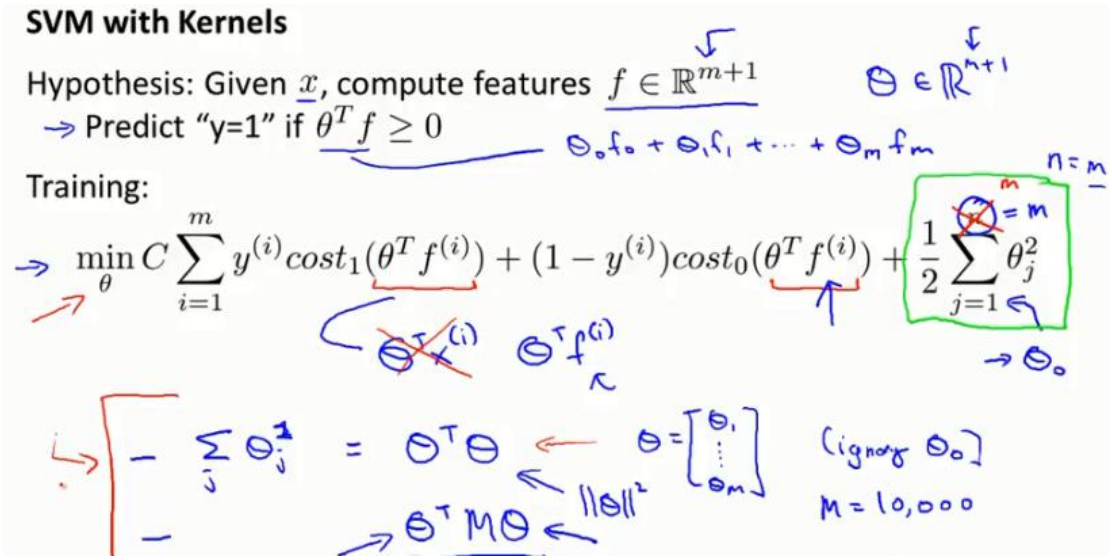
• 选择样本 x 为标记点 l，那么我们可以用一组向量 f 来描述训练样本的特征向量。



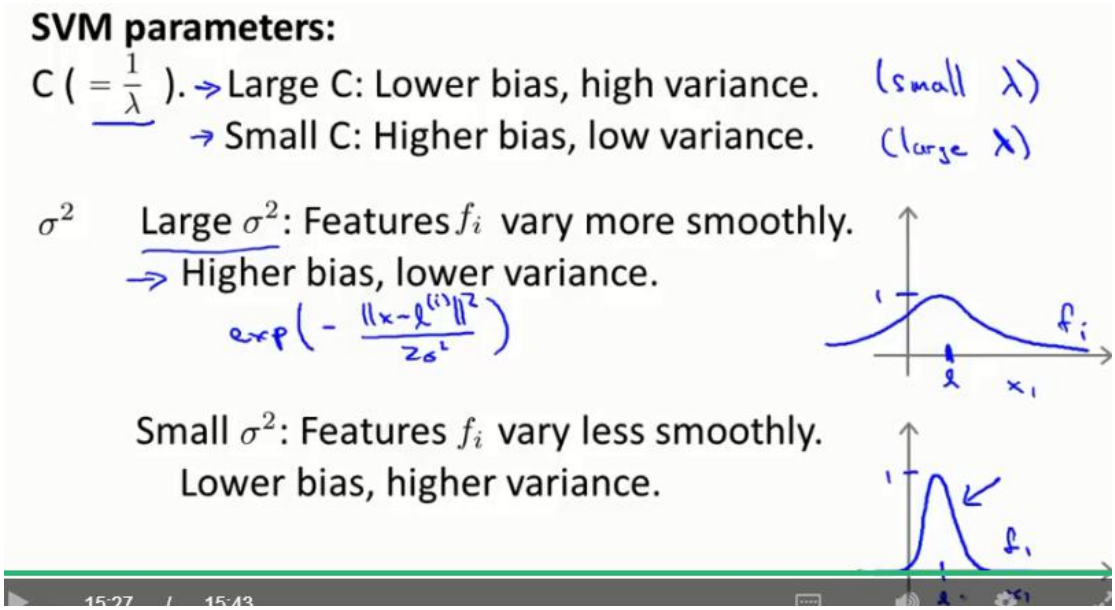
• 1 我们利用优化目标来求解 SVM 的参数

• 2 在计算第二项，Theta 平方求和的时候，我们不能把 Theta0 这一项算入，并且我们可以用 **ThetaT*M*Theta** 来计算（M 依赖于核函数），不直接用||Theta||来计算，这样做的目的是使 SVM 能更有效率地运行，当超大训练集的情况。

• 3 实际上，Theta*f 也可以用在逻辑回归中，但是在逻辑回归中和核函数运算非常慢。



• SVM 中的参数如何选择

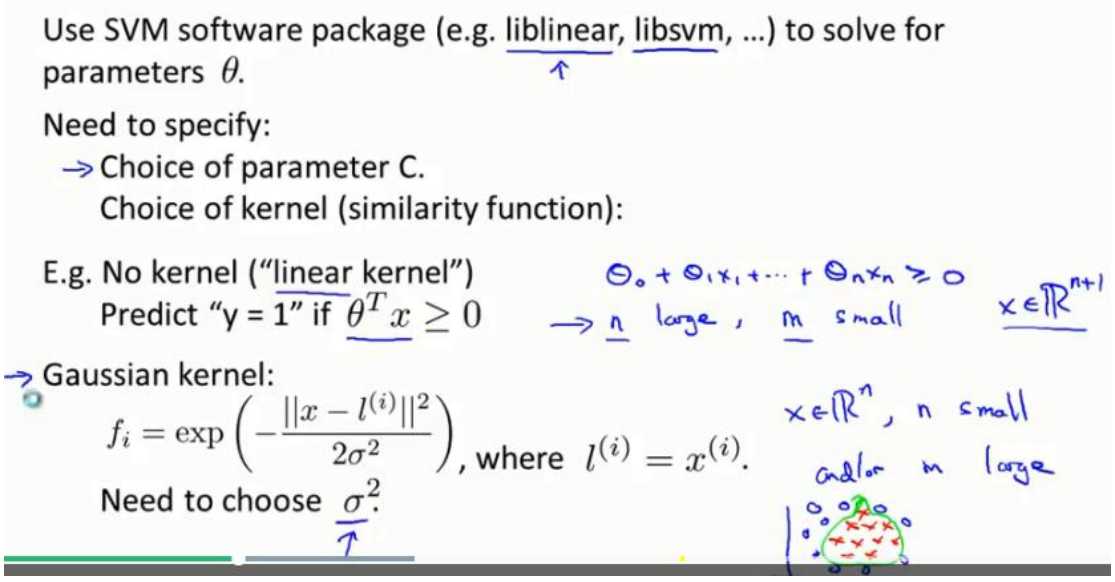


当 a2 过大的时候，变化很平缓，这时候是高偏差，低方差的情况；
当 a2 过小的时候，变化很剧烈，这时候是低偏差，高方差的情况。

3 SVMs in Practice

3-1 Using An SVM

• 已经有很多成熟的软件库，但是我们仍然需要手动选择参数 C、以及核函数。
核函数的选择有线性核函数、高斯核函数。
在特征数量大、样本容量小的时候，我们采用线性核函数。
在特征数量小、样本容量大的时候，我们采用高斯核函数。



• 需要对不同的特征向量归一化，使之能够同等的关注特征向量。（比如房子的尺寸 1000 英尺与卧室的数量 5）

Kernel (similarity) functions:

```
function f = kernel(x1,x2)
```

$$f = \exp\left(-\frac{||x1 - x2||^2}{2\sigma^2}\right)$$

```
return
```

→ Note: Do perform feature scaling before using the Gaussian kernel.

$||x - l||^2$

$v = x - l$

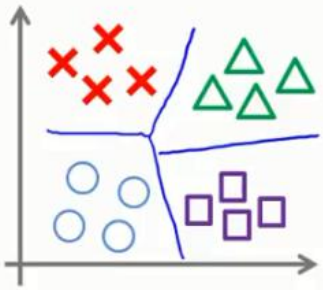
$||v||^2 = v_1^2 + v_2^2 + \cdots + v_n^2$

$= (x_1 - l_1)^2 + (x_2 - l_2)^2 + \cdots + (x_n - l_n)^2$

1000 feet² 1-5 bedrooms

- 所有的核函数都需要遵循 **Mercer's Theorem**
- 多分类与逻辑回归相似，只不过是把 y 换成了 ThetaT*x。

Multi-class classification



$$y \in \{1, 2, 3, \ldots, K\}$$

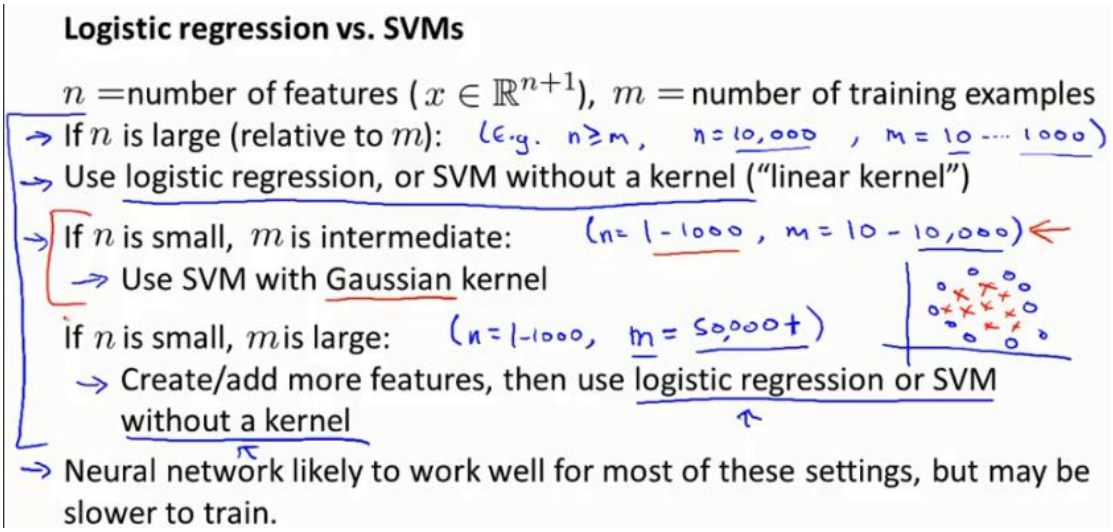
Many SVM packages already have built-in multi-class classification functionality.

→ Otherwise, use one-vs.-all method. (Train K SVMs, one to distinguish $y = i$ from the rest, for $i = 1, 2, \ldots, K$), get $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(K)}$

Pick class i with largest $(\theta^{(i)})^T x$

- 那么逻辑回归和 SVM 该选择哪一个呢？

逻辑回归和不带核函数的 SVM 是非常相似的算法。



当 n 相比 m 很大，我们使用逻辑回归 or 不带核函数的 SVM。

当 n 很小，m 适中，我们使用带高斯核函数的 SVM。

当 n 很小，m 很大，我们使用逻辑回归 or 不带核函数的 SVM。

4-1 Exercise

• 区分是否为垃圾邮件

```
> Anyone knows how much it costs to host a web portal ?
>
Well, it depends on how many visitors youre expecting.  This can be
anywhere from less than 10 bucks a month to a couple of $100.  You
should checkout http://www.rackspace.com/ or perhaps Amazon EC2 if
youre running something big..

To unsubscribe yourself from this mailing list, send an email to:
groupname-unsubscribe@egroups.com
```

经过转换后

```
anyon know how much it cost to host a web portal well it depend on how
mani visitor your expect thi can be anywher from less than number buck
a month to a coupl of dollarnumb you should checkout httpaddr or perhap
amazon ecnumb if your run someth big to unsubscrib yourself from thi
mail list send an email to emailaddr
```

Week 8 Unsupervised Learning

1 Clustering

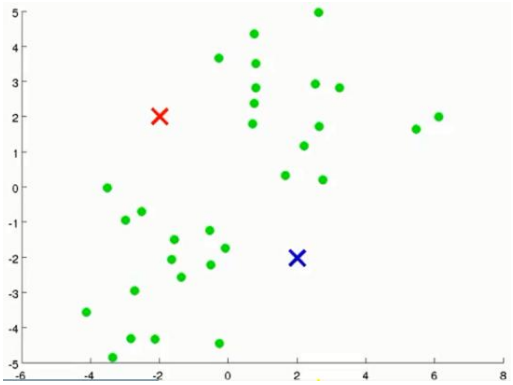
1-1 Unsupervised Learning: Introduction

聚类 Clustering 是非监督学习的一种。

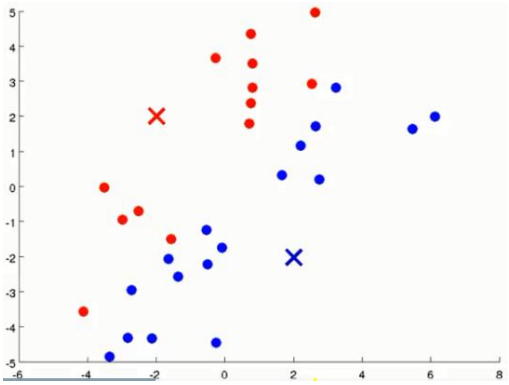
1-2 K-Means Algorithm

(K 均值算法)一种最广泛的聚类方法

①随机寻找两个点作为**聚类中心(cluster centroids)**

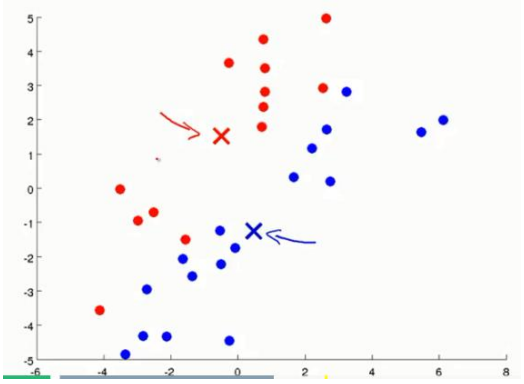


②簇分配：遍历每一个样本(绿点)，依据更接近与哪个中心将点分为两类(红点、蓝点)



③移动聚类中心

计算所有红点、蓝点的平均位置。



④重新进行簇分配、聚类中心移动直到聚类中心不变。

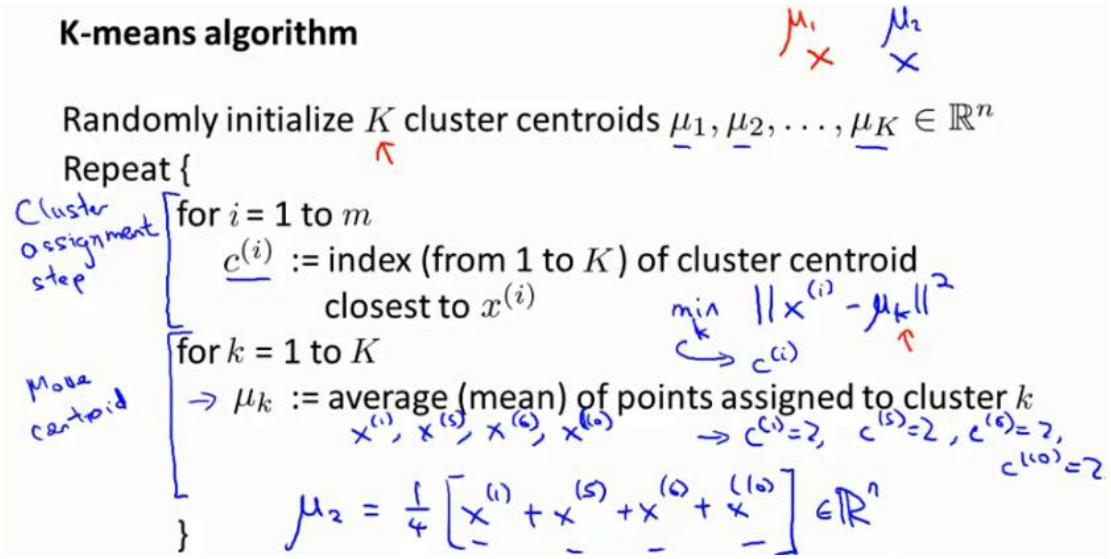
整个过程

K-means algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

$x^{(i)} \in \mathbb{R}^n$ (drop $x_0 = 1$ convention)

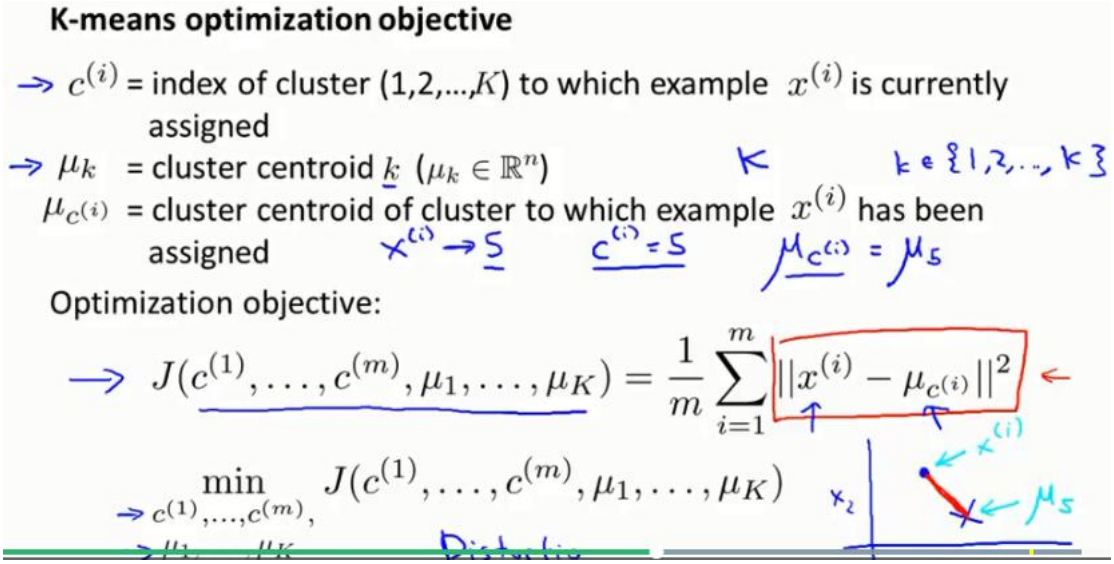


c(2) = 1，表示第二个样本属于第一类

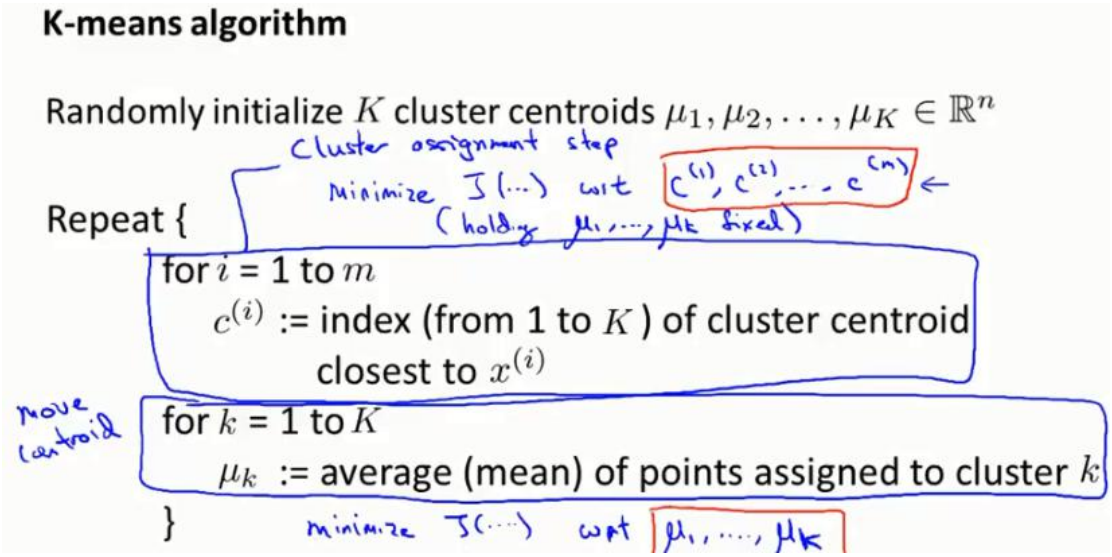
若存在没有点属于的聚类，我们直接移除该聚类中心。

1-3 Optimization Objective

- 定义三个量与代价失真函数。



• 可以看作簇分配与移动聚类中心就**分别以样本 c(i)和 uk 为变量来最小化代价函数 J。**

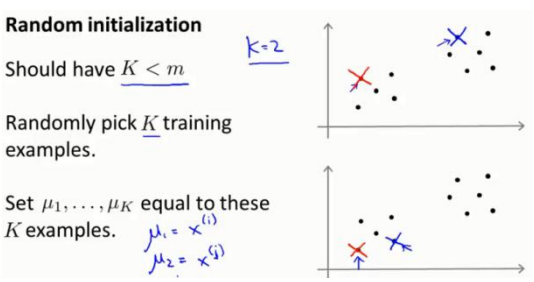


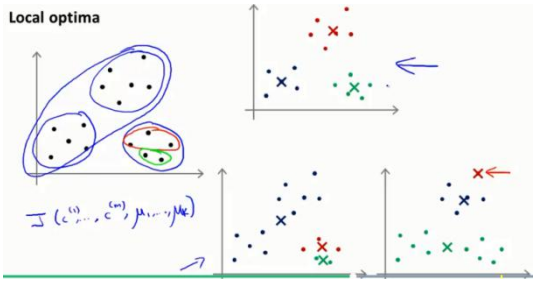
1-4 Random Initialization

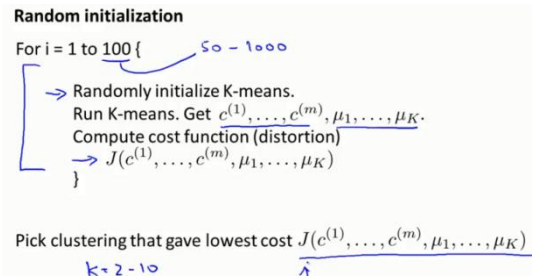
(随机初始化聚类中心)

• 假设有 K 个聚类中心，那么我们随机选择 K 个样本赋值给聚类中心。

• 但是这种随机初始化的弊端是在于 K 均值法最终可能收敛在一个局部最优解。如右图，于是我们想多试几次来确保效果。如循环 100 次，选择 J 值最小的一种。**这种方法在当 K 值较小的时候效果较好。**在 K 值较大的时候，变化不大，但多次循环依然最给出一个较合理的起始点。







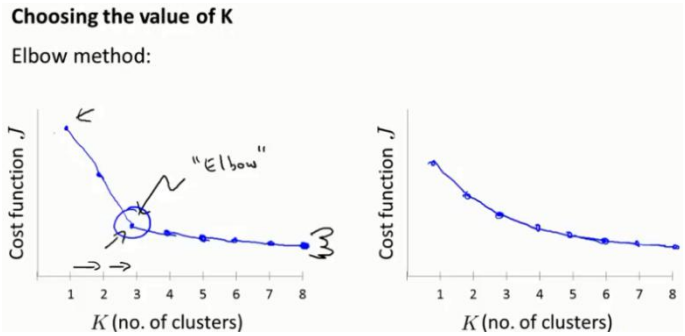
1-5 Choosing the Number of Clusters

(选择 K 的值)

• 最为常见的方法：根据可视化的结果手动选择。

Elbow method(肘部法则)

作出 J 随 K 的变化图，如果可以看到直观的拐点，我们就选择拐点作为我们的 K 值。但是有时候也将无法判断。

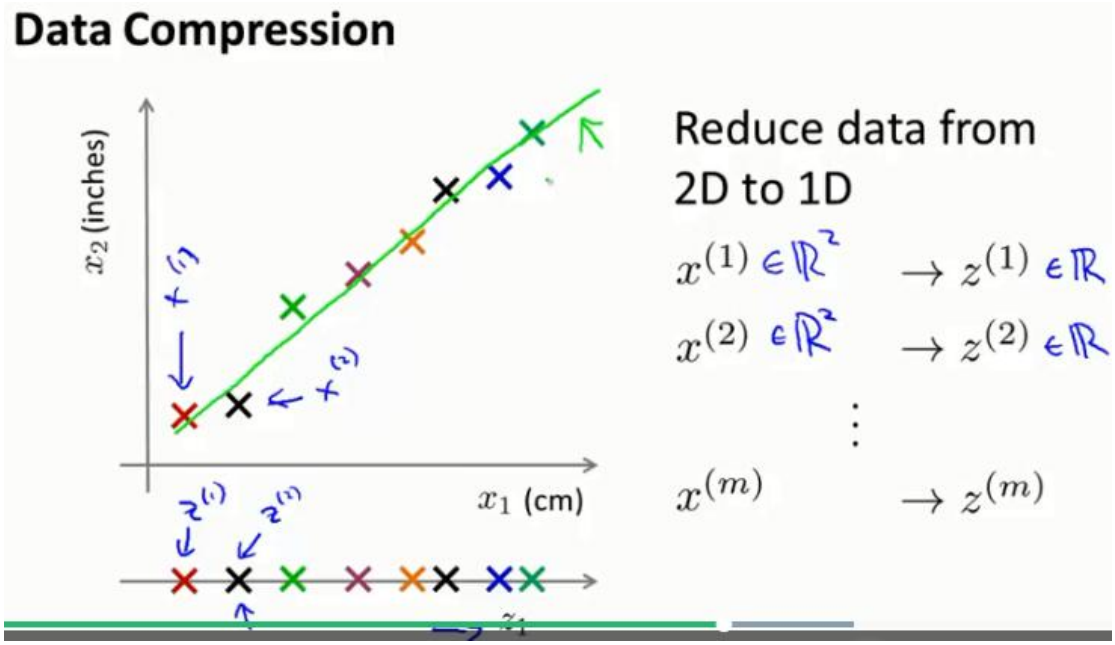


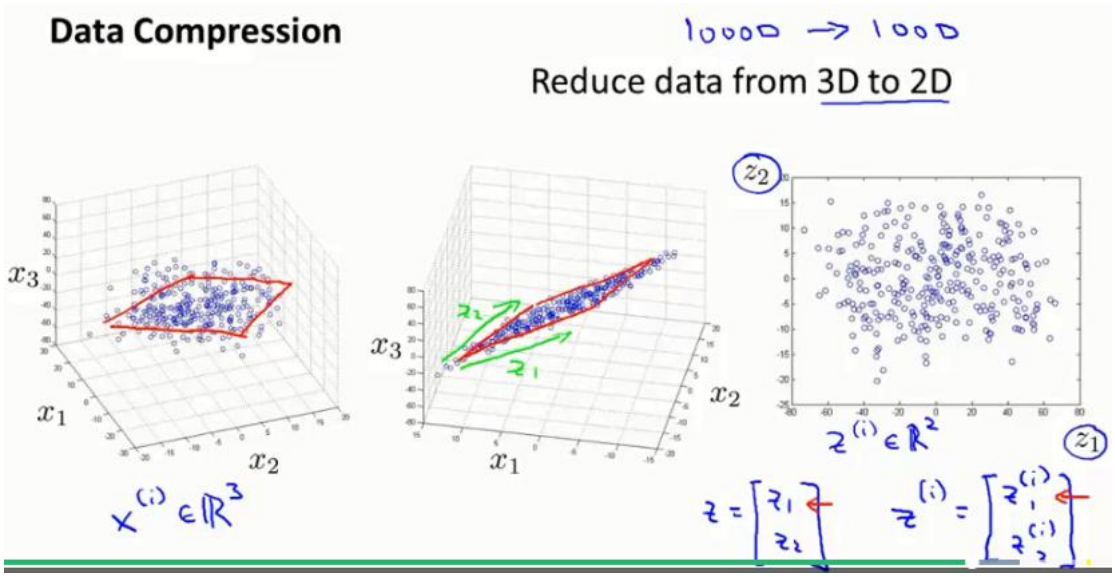
Week 8 Dimensionality Reduction

2 Motivation

2-1 Motivation I:Data Compression

• **Dimensionality Reduction(维数约减)** 优点：数据压缩，占用计算机内存少。





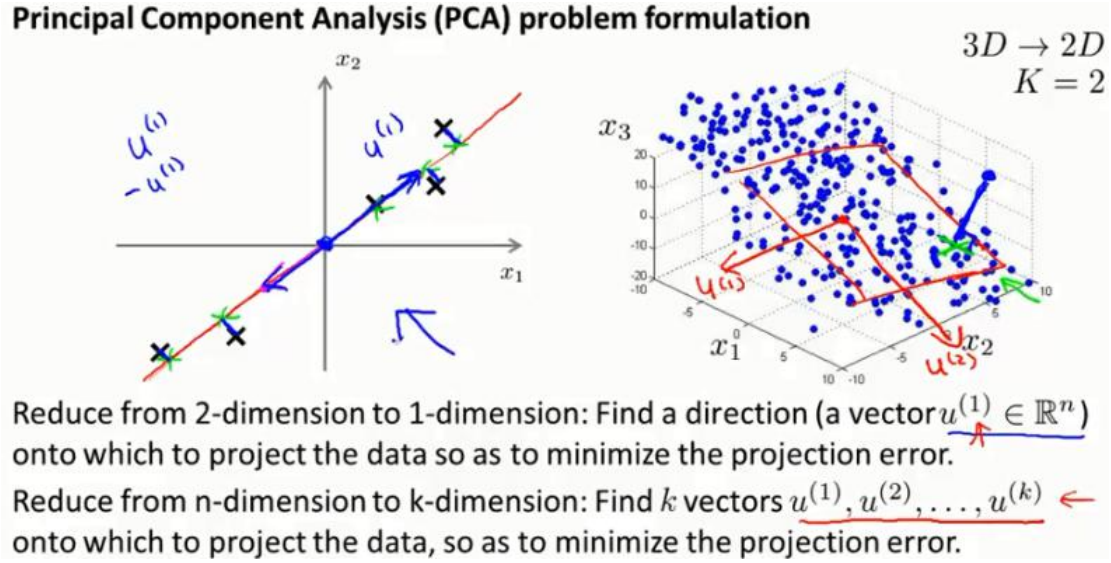
2-2 Motivation II: Visualization

• 假设当我们有 50 个特征参数时，经过计算最终发现样本可以由 2 个主要的特征参数来表征，并且有利于我们将数据可视化。

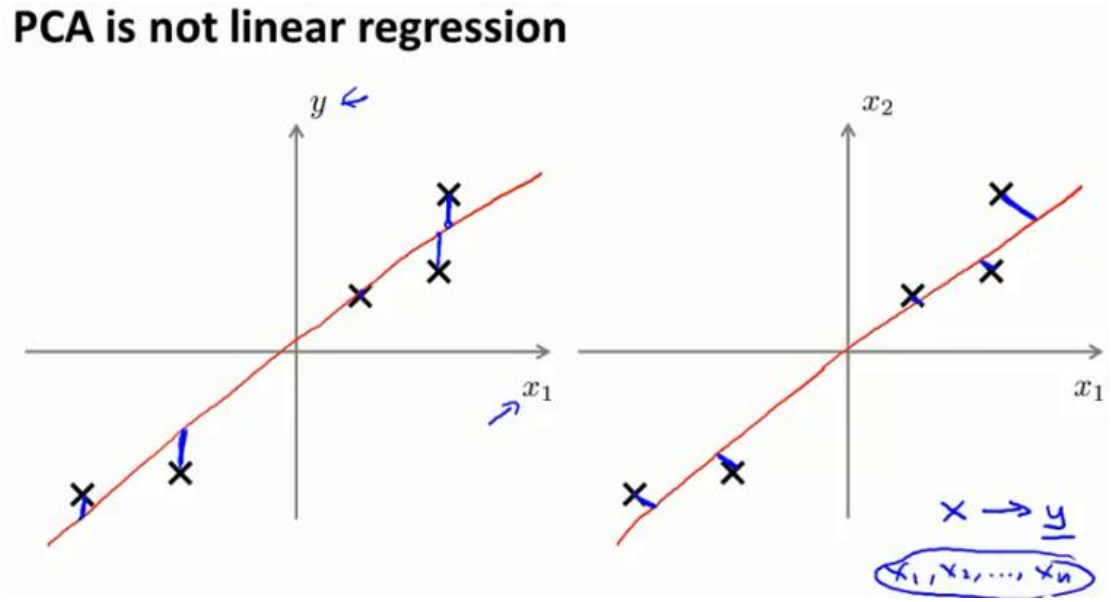
3 Principal Component Analysis

3-1 Principal Component Analysis Problem Formulation

主成分分析法(PCA)：寻找一个投影平面，对样本数据进行投影，使得投影误差能够最小化。



• PCA 与线性回归的关系



• 线性回归是垂直线、而 PCA 是垂线。

并且线性回归的目的是当我们给出一个特殊值来预测结果，而 PCA 中没有，所有的 x 都被平等对待。

3-2 Principal Component Analysis Algorithm

• 数据预处理(均值归一化、特征缩放)

Data preprocessing

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

Preprocessing (feature scaling/mean normalization):

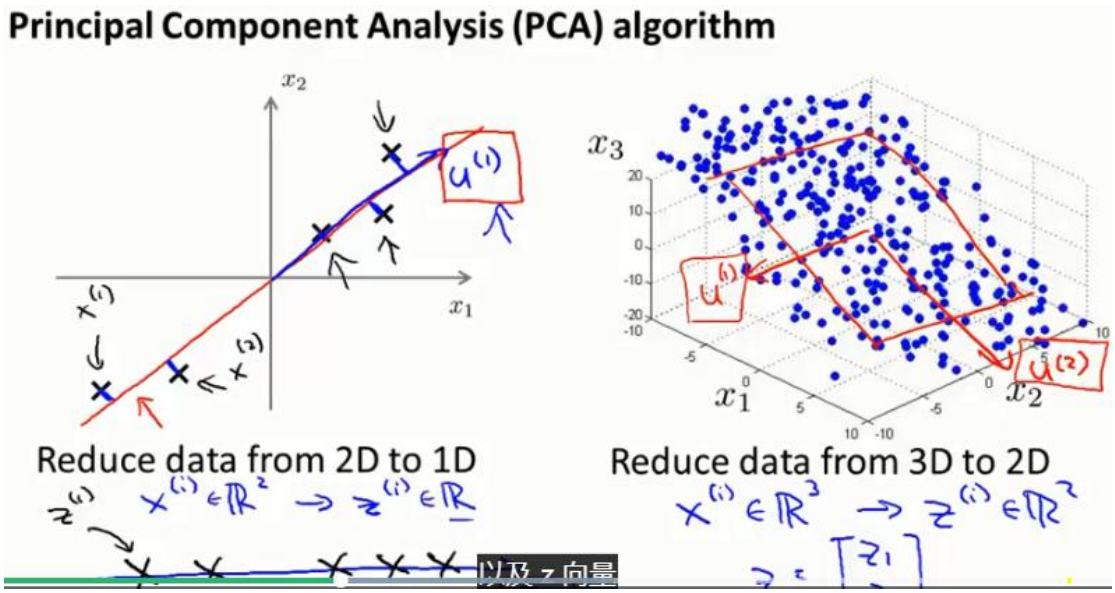
$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$

Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

If different features on different scales (e.g., x_1 = size of house, x_2 = number of bedrooms), scale features to have comparable range of values.

$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$

• PCA 完成了两件事：1 计算出向量 u(i)、2 计算投影后的特征参数 z(i)

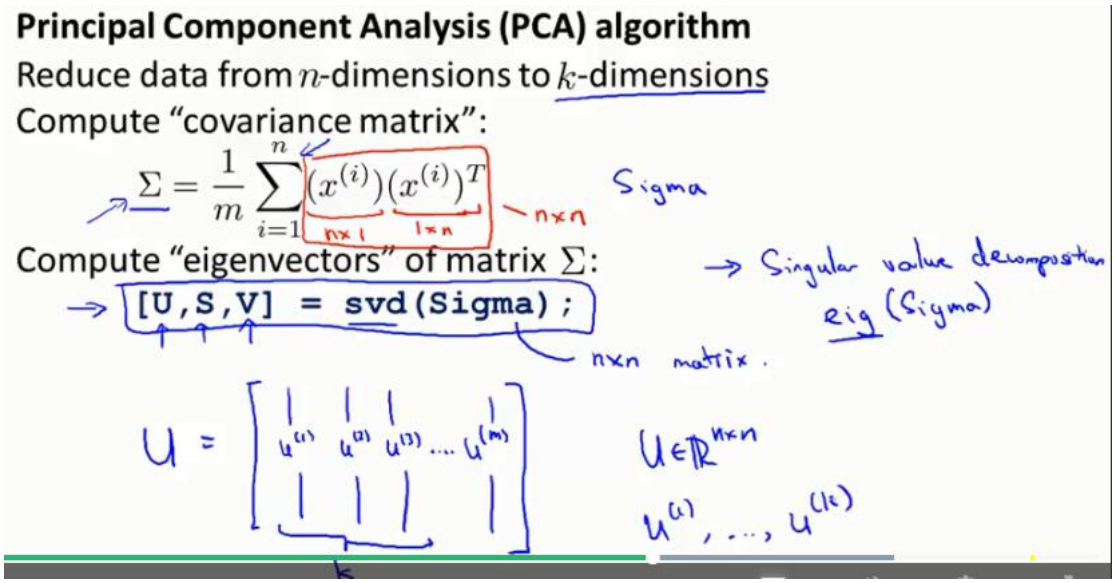


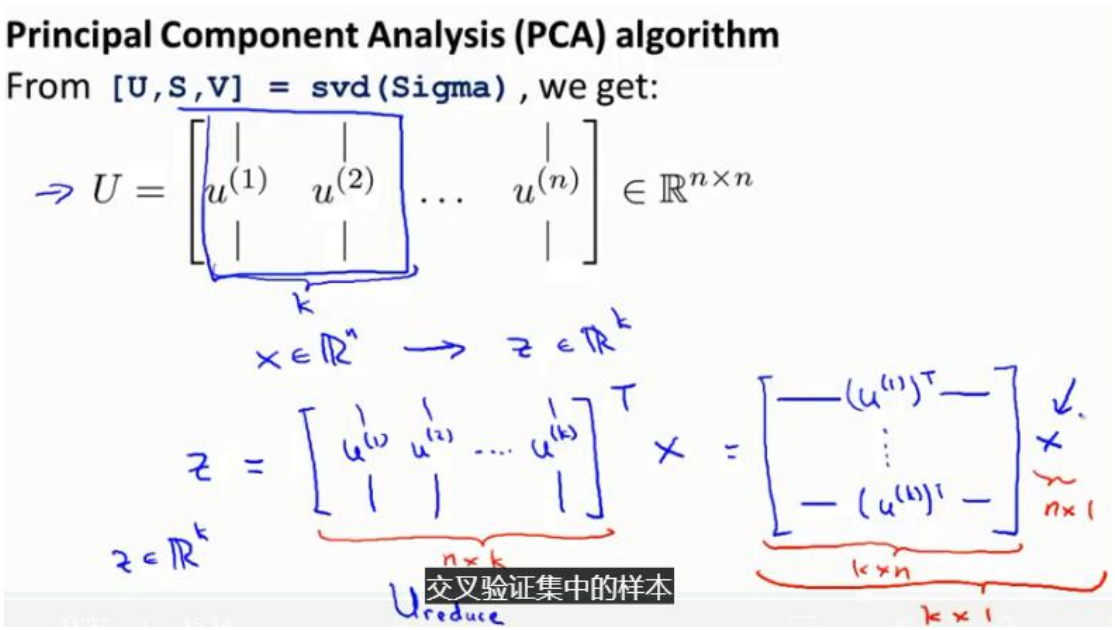
• 如何降维

1 计算出原特征参数的协方差矩阵(应为 n*n 的方阵)。

2 使用 **svd 奇异值分解**来计算它的特征参量。使用 svd 后将返回 U，S，V。我们只需要 U。

假设我们将 n 维的矩阵降为 k 维，那么我们只需要前 k 列向量。





那么那么新的特征参量 z = (Unew)T*x，Unew 为减少列数后的矩阵。Z 的维度为 k*1。

In PCA, we obtain $z \in \mathbb{R}^k$ from $x \in \mathbb{R}^n$ as follows:

$$z = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(k)} \\ | & | & & | \end{bmatrix}^T x = \begin{bmatrix} --- & (u^{(1)})^T & --- \\ --- & (u^{(2)})^T & --- \\ & \vdots & \\ --- & (u^{(k)})^T & --- \end{bmatrix} x$$

Which of the following is a correct expression for z_j?

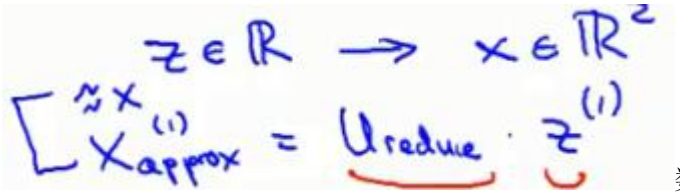
- $z_j = (u^{(k)})^T x$
- $z_j = (u^{(j)})^T x_j$
- $z_j = (u^{(j)})^T x_k$
- $z_j = (u^{(j)})^T x$

假设有 n 维的特征参数，X 是原特征向量，维度为 n*1。
U 为 n*n，我们取前 k 维，则 Ureduce 为 n*k， (Ureduce)T = k*n，z = k*n。
补充：X(m*n)、U(n*n)、U_reduce(n*k)、z(m*k)

4 Applying PCA

4-1 Reconstruction from Compressed Representation

重构数据，从低纬到高纬

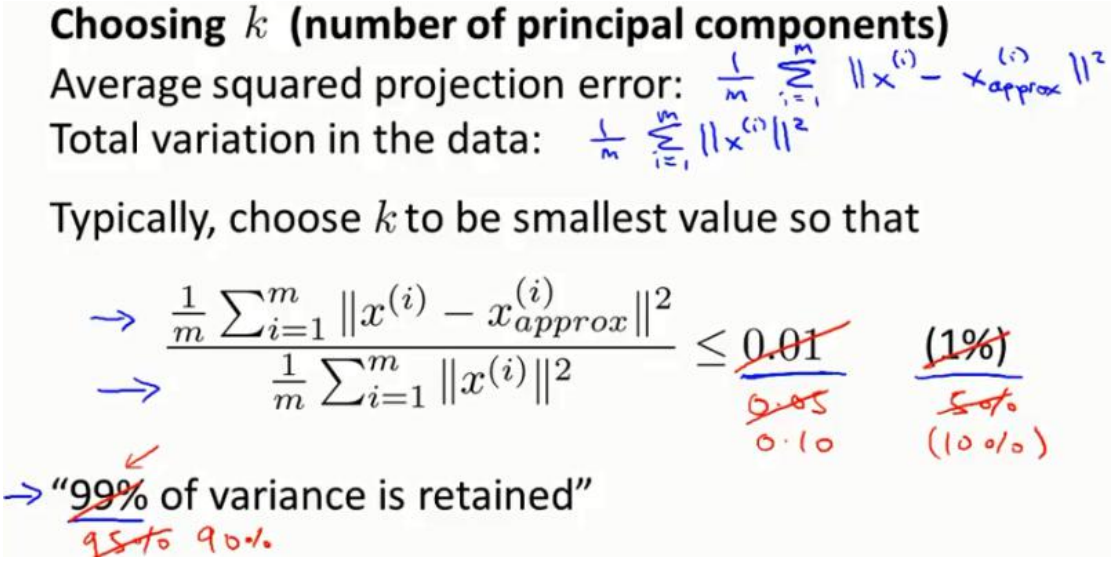


数据从低维到高纬。

X = Ureduce*z。

4-2 Choosing the Number of Principal Components

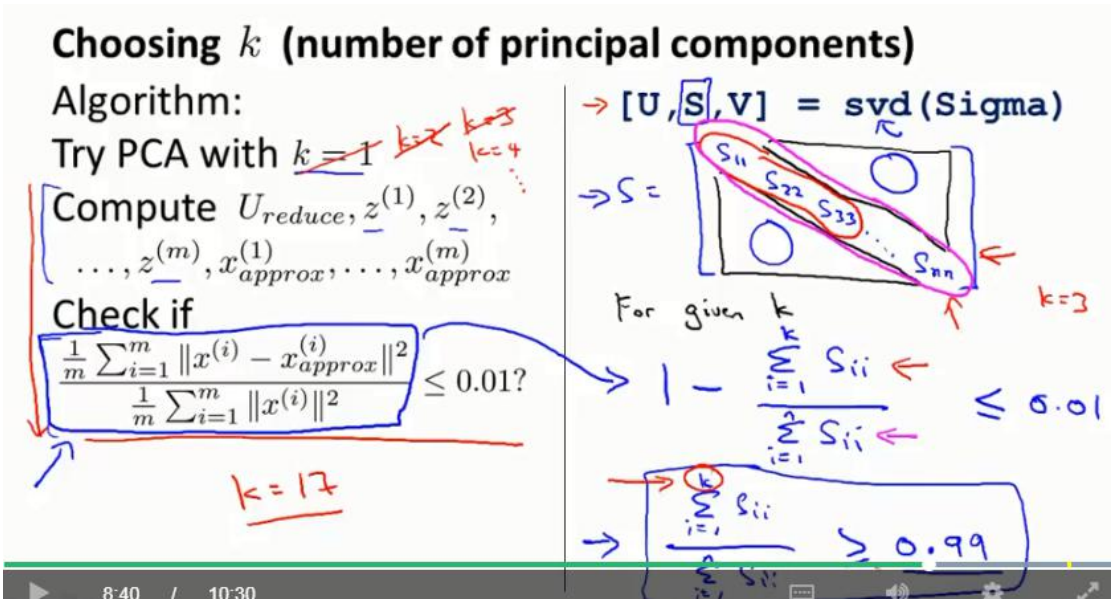
最小化平均平方映射误差、数据的总变差
·降低维度后 平均平方映射误差与数据总变差的比值小于 0.01，说明至少保留了 99%的差异性。

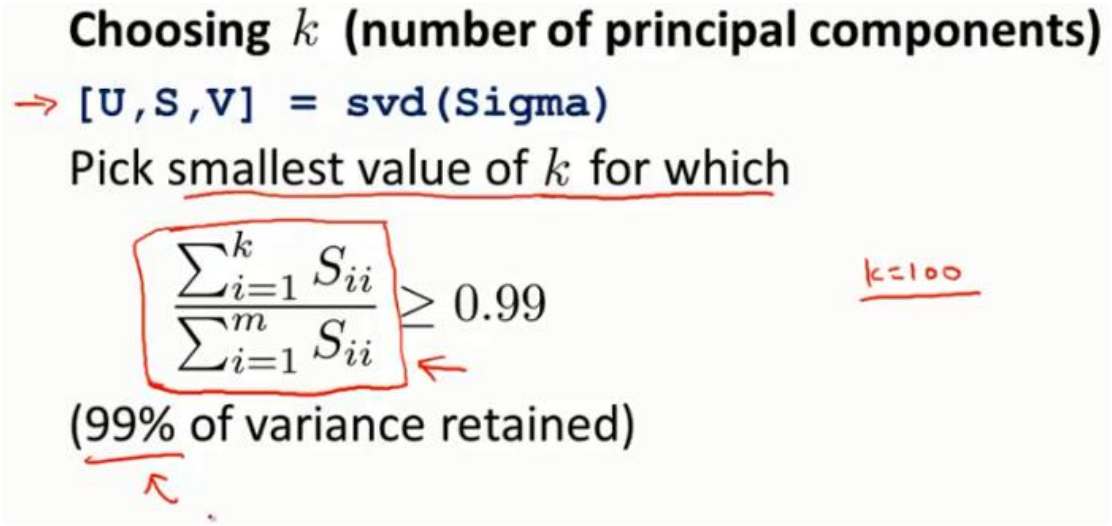


• 选择满足差异值最小 k 值的方法：

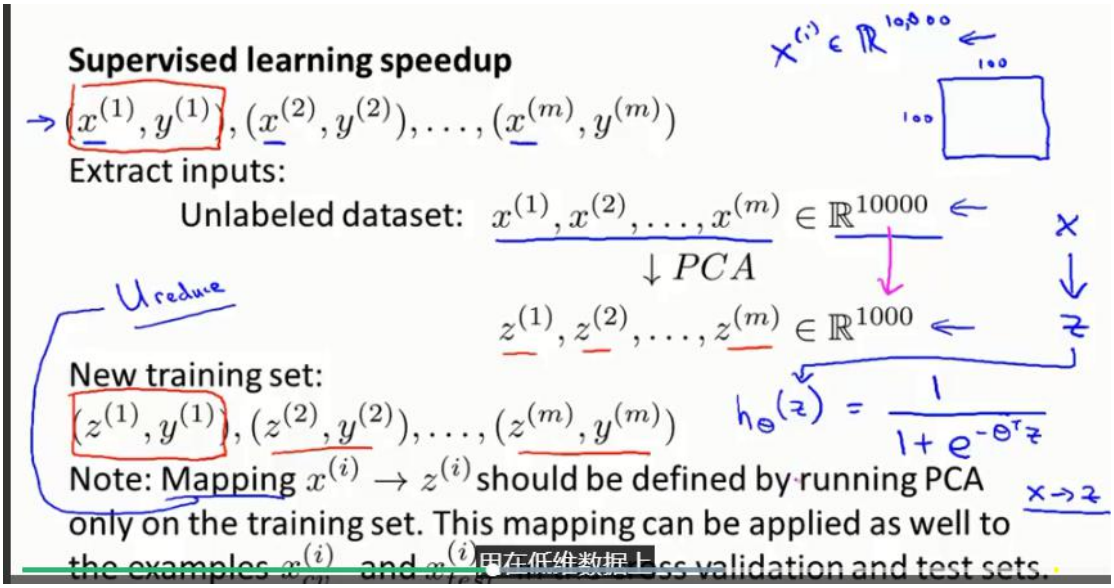
1 从 k=1 开始到 k=2、k=3，不断地重复计算下左图中的项，观察其是否小于 0.01。

2 使用 svd 后会返回 S，我们可以使用其来计算等价于左边的项。

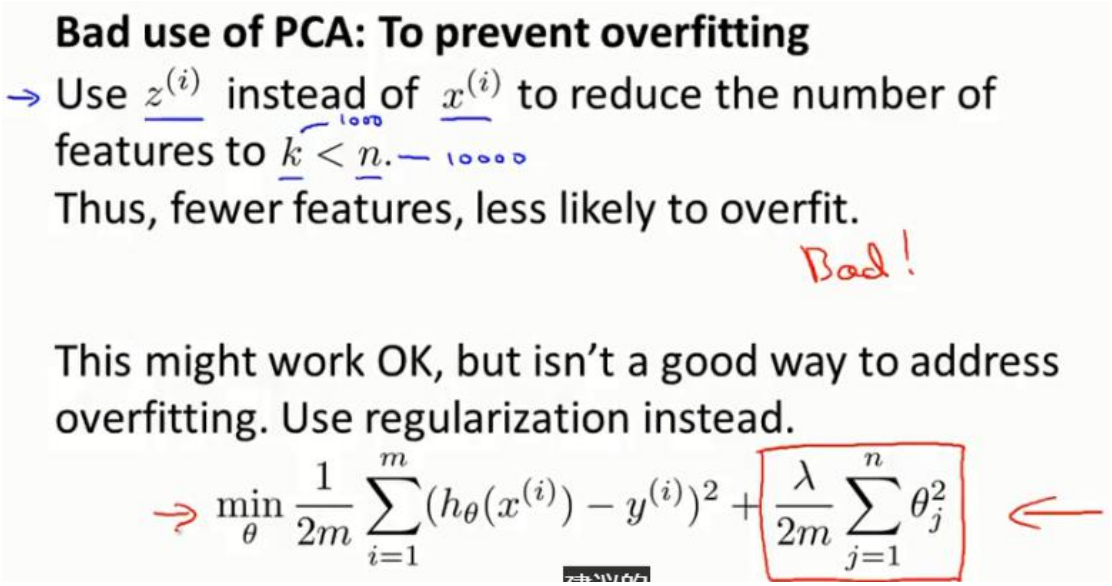




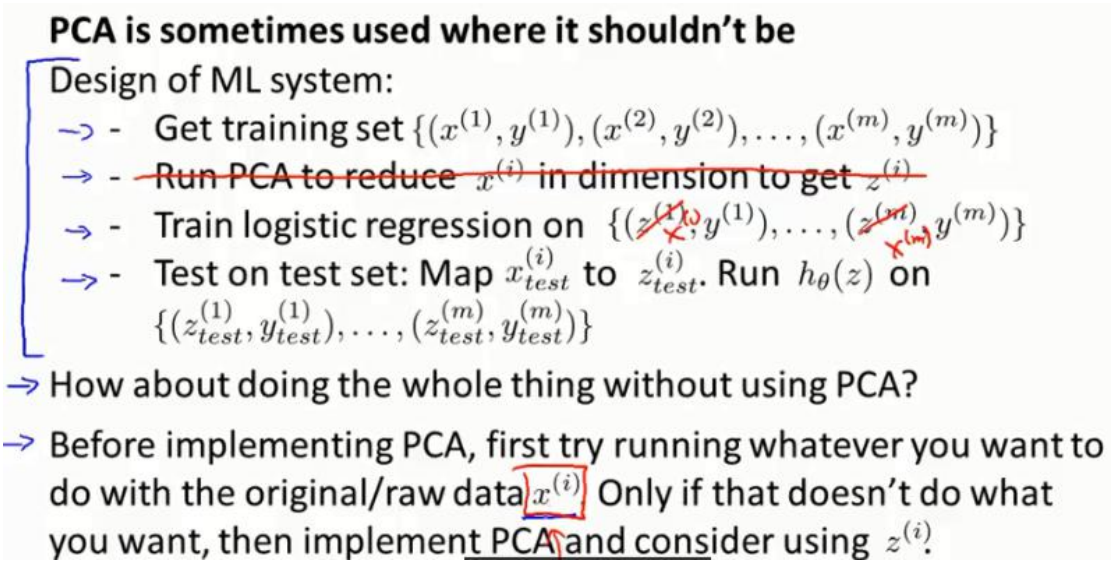
4-3 Advice for Applying PCA



PCA 的应用：**压缩数据/减少内存（加速算法）、可视化数据**。不要使用它来避免过拟合。比如将 1000 维压缩到 100 维，看起来好像是特征数量减少了，但是他实际上舍弃了有用的信息；若要真的避免过拟合，最好采用正则化来解决。



• 不要盲目使用 PCA，我们要先使用 X 跑跑看，如果真的出现了数据内容过大的问题，我们再选用 PCA。



• 运行 PCA 之前需要进行特征缩放。

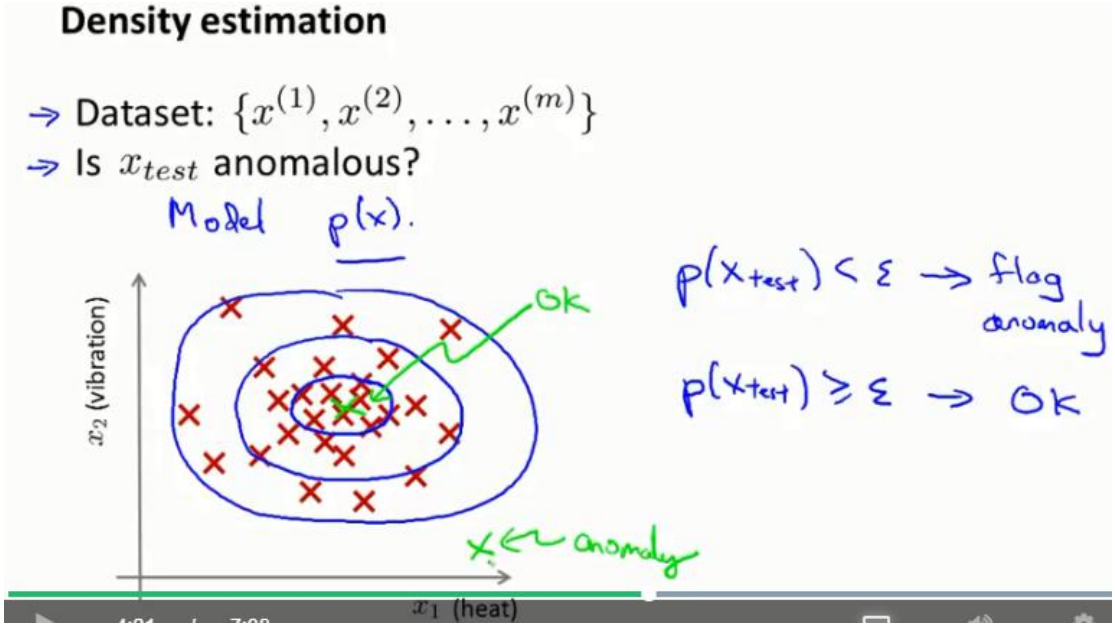
Week 9 Anomaly Detection

1 Density Estimation

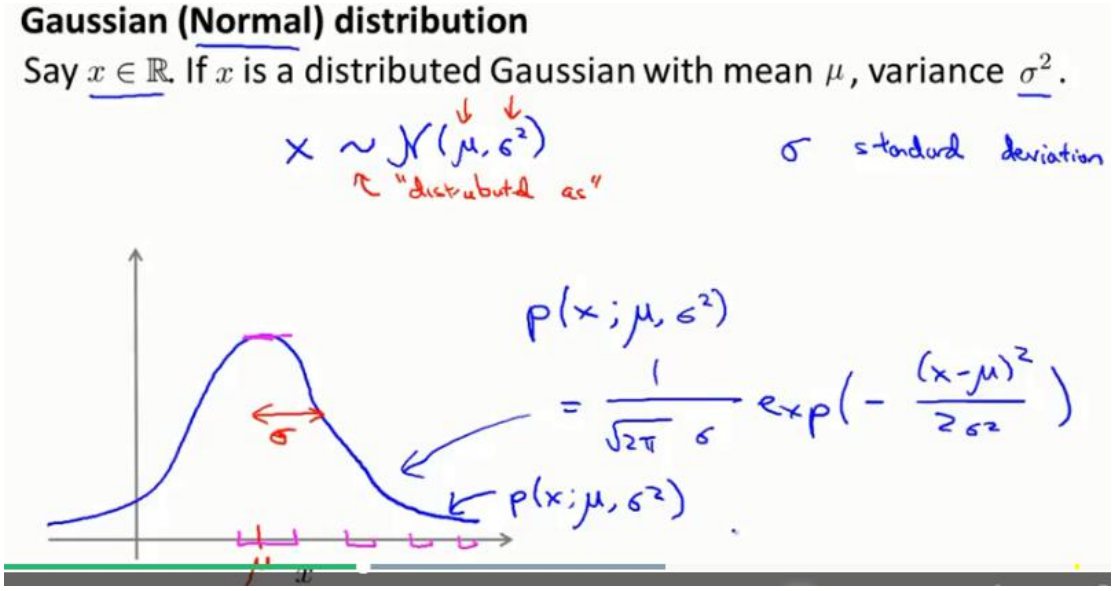
1-1 Problem Motivation

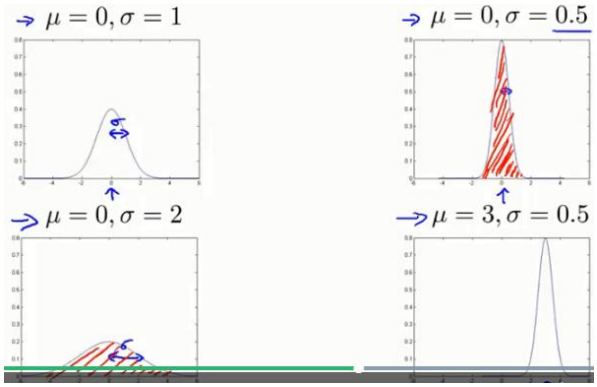
异常检测算法

我们拥有一个模型，这个数据的可能性大于某个阈值时，认为其正常；反之为异常。

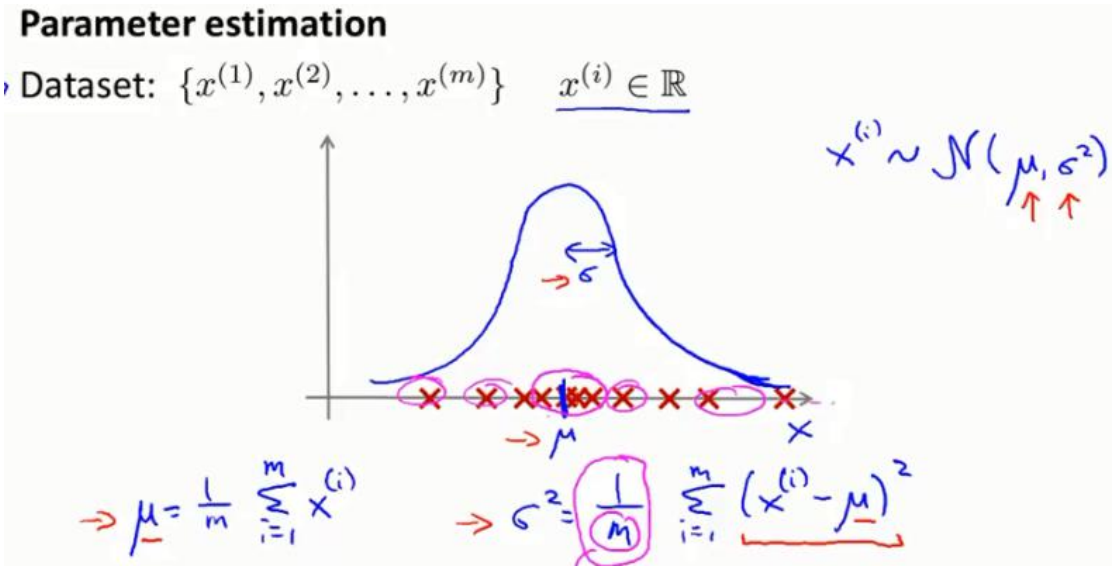


1-2 Gaussian Distribution



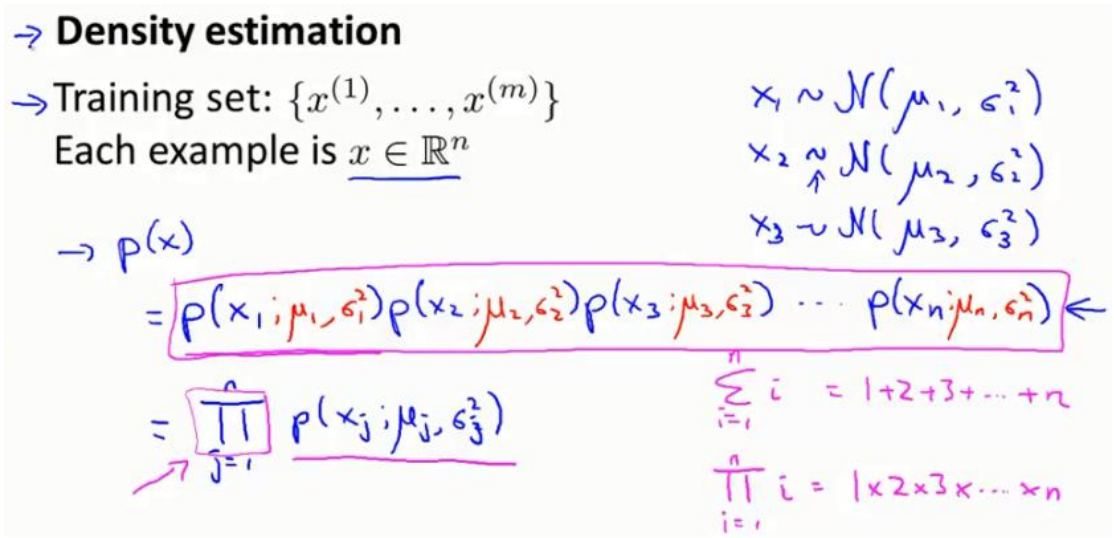


• 如何确定参数平均值与方差。



1-3 Algorithm

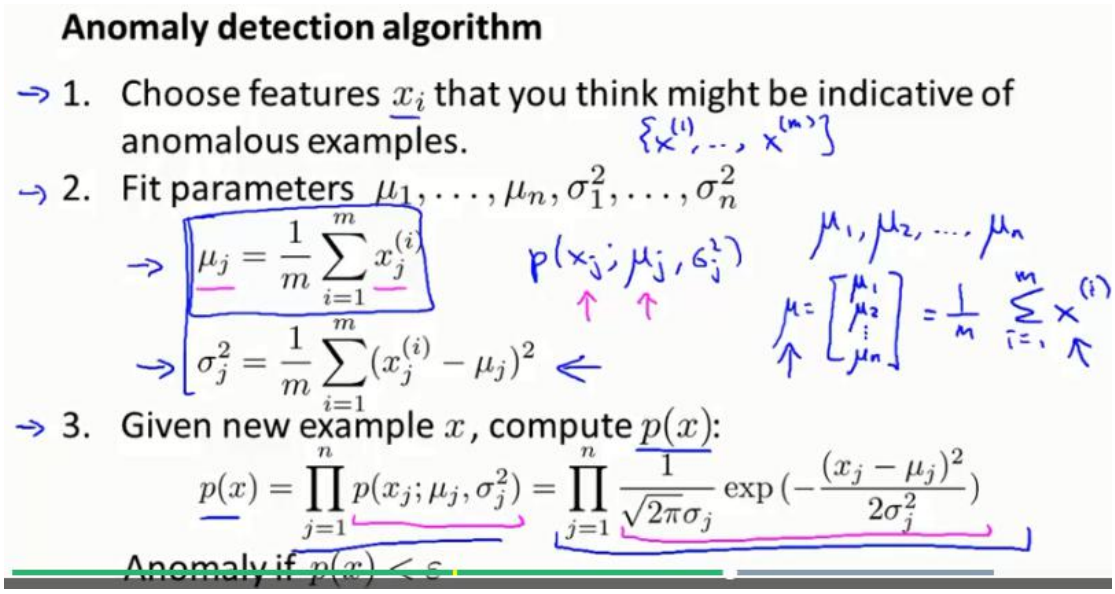
(异常检测方法)



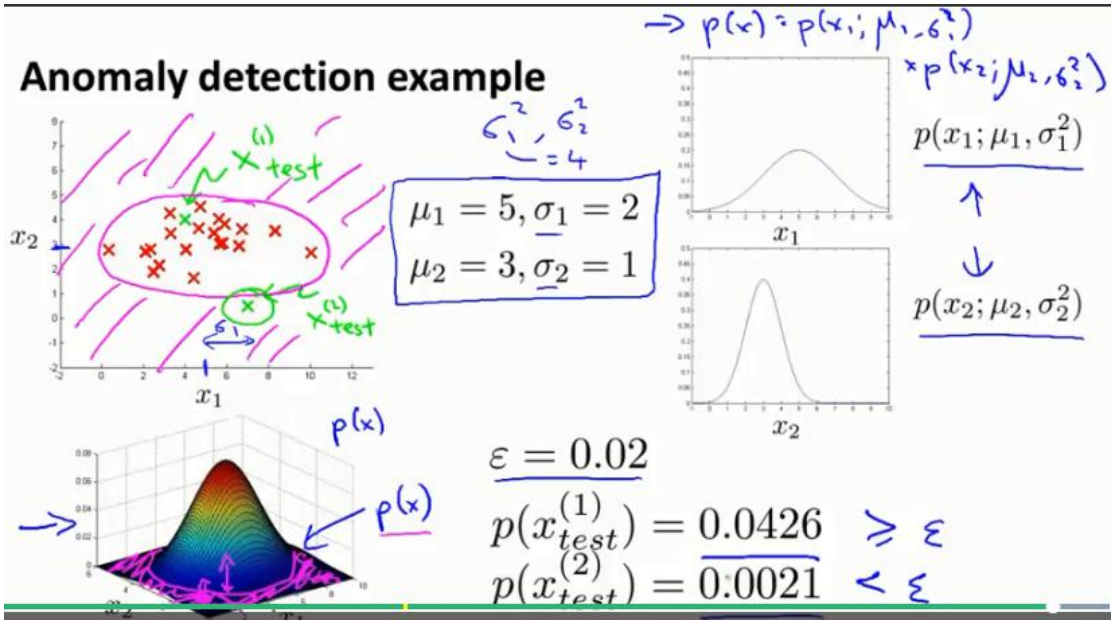
1、选择可能指出异常的特征 xi。

2、计算出每类特征对应的平均值与方差

3、选定一个样本，计算他的期望值。



举例。



2 Building an Anomaly Detection System

2-1 Developing and Evaluating an Anomaly Detection System

开发异常检测的应用

•虽然异常检测是应用于无监督学习，但是我们希望拥有一个具体的数字来评价异常检测的好坏，所以我们可以采用监督学习中的方法，给出一个样本是否为异样来测试这个异常检测模型。

The importance of real-number evaluation

When developing a learning algorithm (choosing features, etc.), making decisions is much easier if we have a way of evaluating our learning algorithm.

→ Assume we have some labeled data, of anomalous and non-anomalous examples. ($y = 0$ if normal, $y = 1$ if anomalous).

→ Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ (assume normal examples/not anomalous)

→ Cross validation set: $(x_{cv}^{(1)}, y_{cv}^{(1)}), \ldots, (x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$

→ Test set: $(x_{test}^{(1)}, y_{test}^{(1)}), \ldots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

• 将 10000 个好样本、20 个坏样本分为三组，**分别给训练集（只有好样本）**、交叉训练集、测试集，注意**测试集和交叉训练集不应该共用**。

Aircraft engines motivating example

→ 10000 good (normal) engines

→ 20 flawed engines (anomalous) 2-50 $y=1$

→ Training set: 6000 good engines ($y = 0$) $p(x) = p(x_1; \mu_1, \sigma_1^2) \cdots p(x_n; \mu_n, \sigma_n^2)$

CV: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Test: 2000 good engines ($y = 0$), 10 anomalous ($y = 1$)

Alternative:

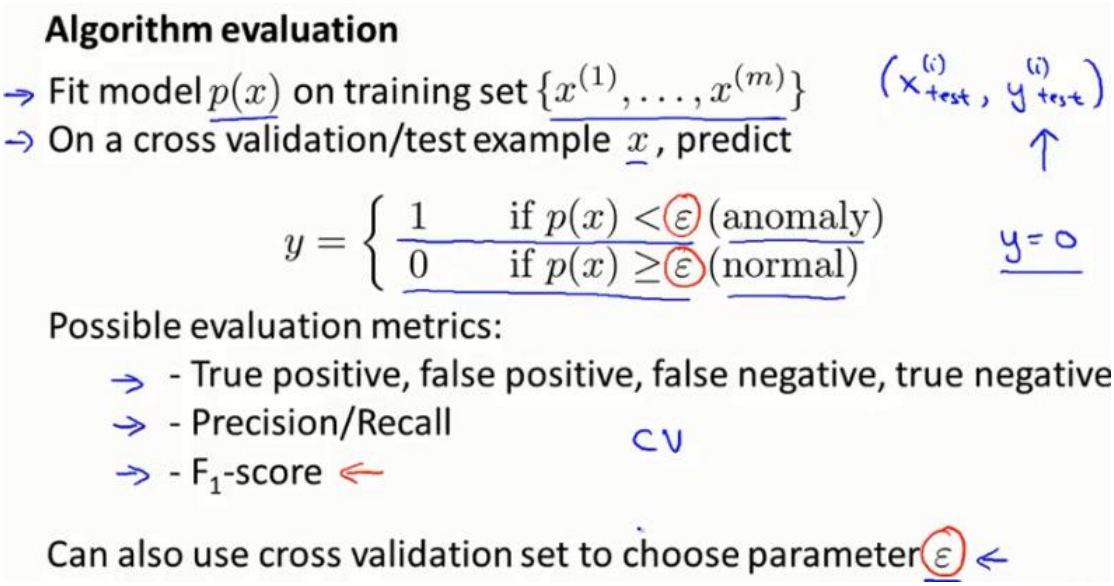
Training set: 6000 good engines

→ CV: 4000 good engines ($y = 0$), 10 anomalous ($y = 1$)

→ Test: 4000 good engines ($y = 0$), 10 anomalous ($y = 1$)

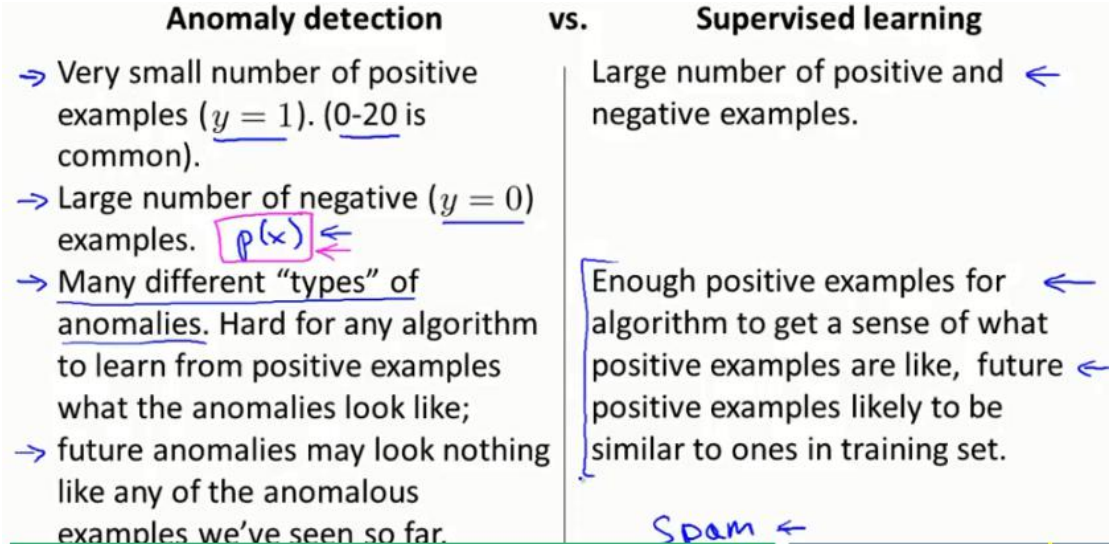
• 在交叉训练集与测试集中，因为正常的样本远多于异常的样本，于是我们需要使用之前所学的四类评价指标来衡量这个评价算法的好坏。

• 至于如何选定参数 Sigma，我们可以使用多组，代入模型，最终选出得分最高的参数。



2-2 Anomaly Detection vs. Supervised Learning

异常检测算法和监督学习算法（逻辑回归、神经网络）我们该用哪个？



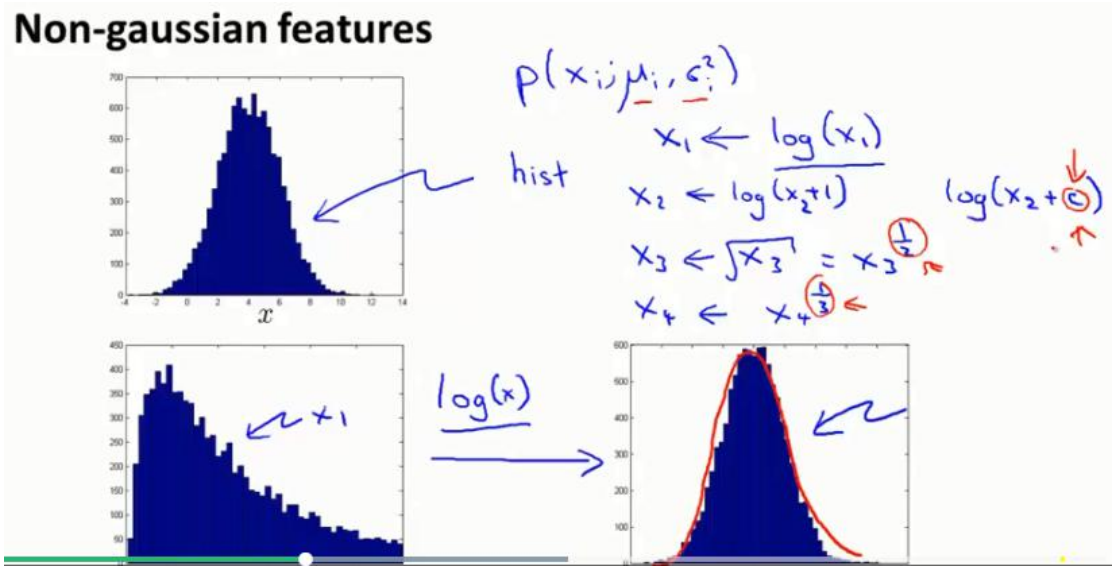
• 若异常样本数非常小，我们应该采用异常检测算法；若非常大，我们采用监督学习算法。

•异常检测算法难以从较少的样本中去预测各种类型的异常样本是怎么样的，也难以预测将来的异常类型；而监督学习算法有大量的数据，所以更有可能预测将来的算法。

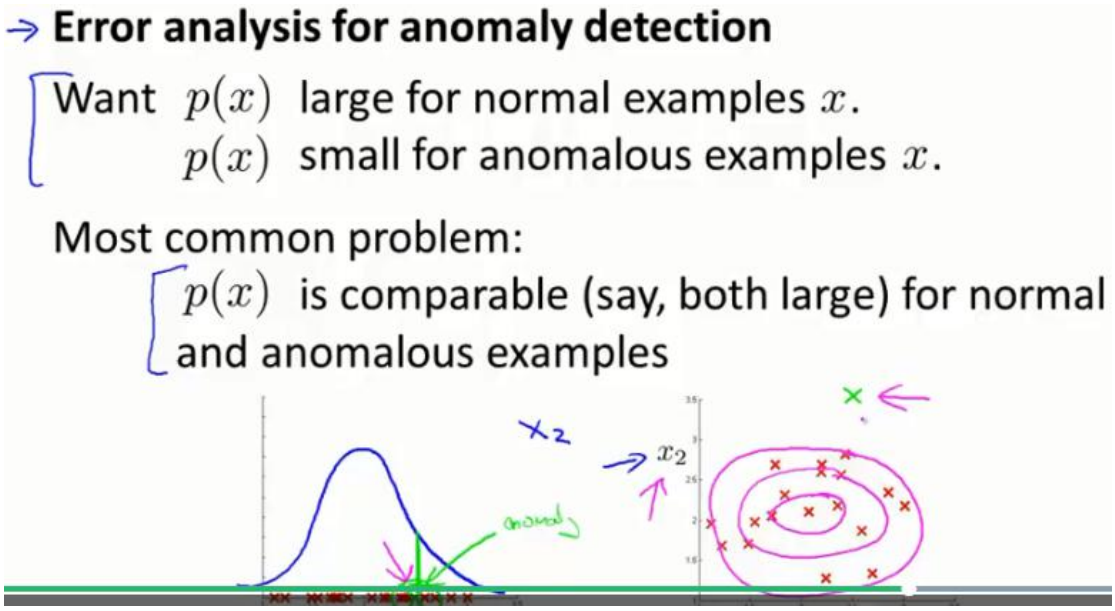
2-3 Choosing What Features to Use

（如何选定特征）

我们希望数据集呈高斯分布，若不是高斯分布也没关系，我们可以将变量进行转换，如取对数，让他近似于高斯分布。



• 正常的样本 p 值大、异常样本 p 值小。但是很可能会有这种情况，对于异常的样本，它的 P 值也很大，这样将无法和正常的样本进行区分开。那么，我们就要进行误差分析，看能不能**创建新的特征**，来区分出这个异常样本。



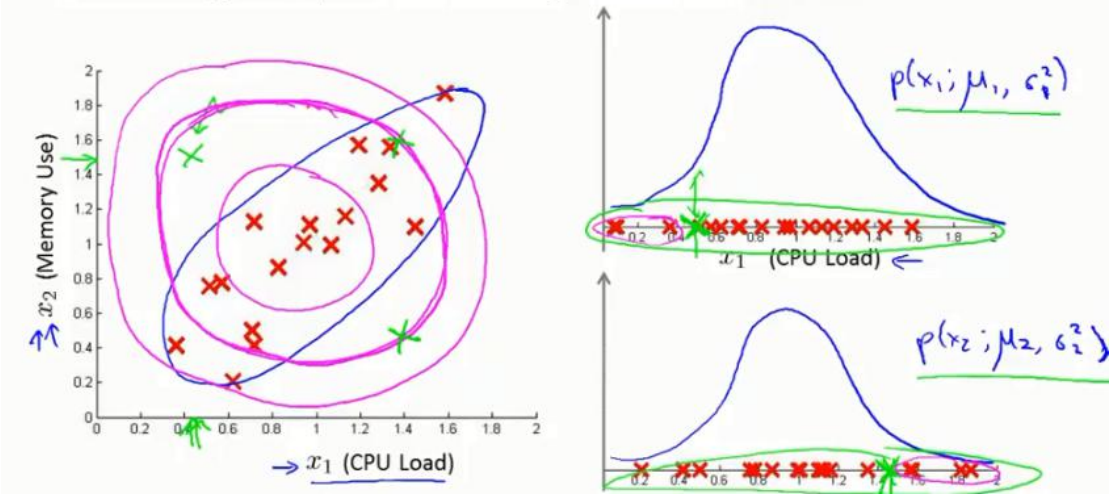
通过不同特征的组合和关系，从而捕捉新的异常类型。

3 Multivariate Gaussian Distribution (Optional)

3-1 Multivariate Gaussian Distribution

• 之前的异常检测算法，会以正常样本为中心作同心圆，而绿点虽然为异常点，但是由于其 x1、x2 的 P 值并不小，所以依然也会被归入正常样本。所以我们需要改良异常算法，采用**多元高斯分布**。

Motivating example: Monitoring machines in a data center



Multivariate Gaussian (Normal) distribution

$x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \ldots$, etc. separately.

Model $p(x)$ all in one go.

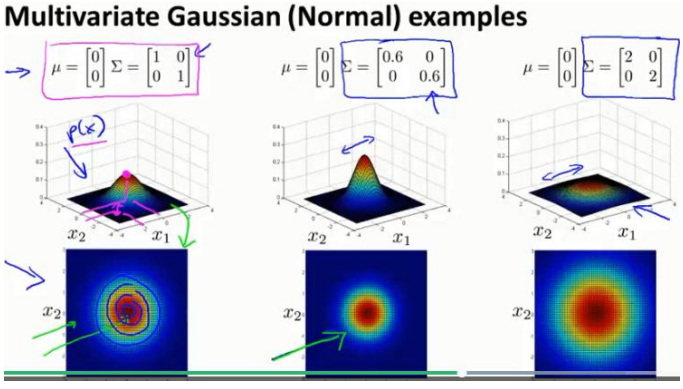
Parameters: $\mu \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

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

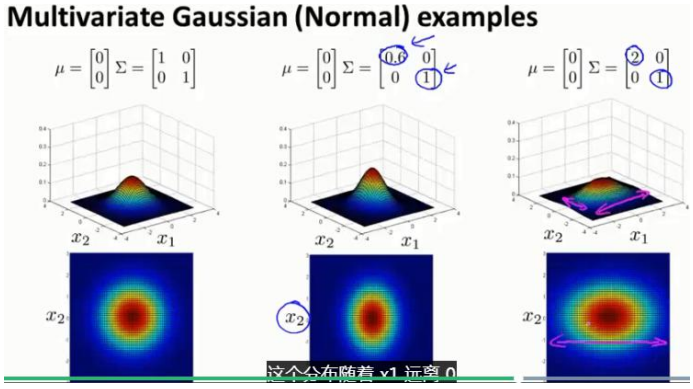
$|\Sigma|$ = determinant of Σ | det (Sigma)

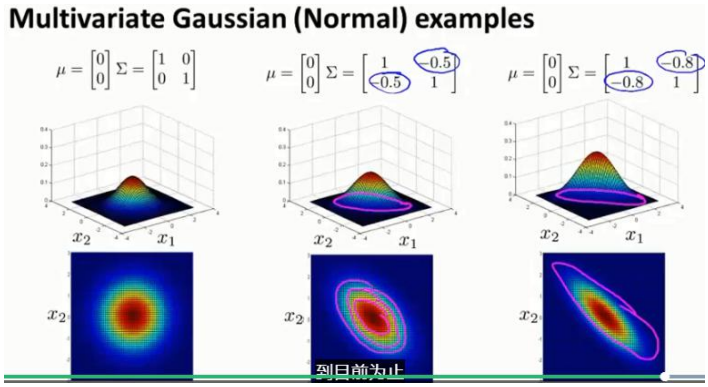
• 对角线表示在 x1、x2 方向上的胖瘦；非对角线若为正，图像沿着 x1=x2 拉伸、否则沿着逆方向拉伸。

Multivariate Gaussian (Normal) examples

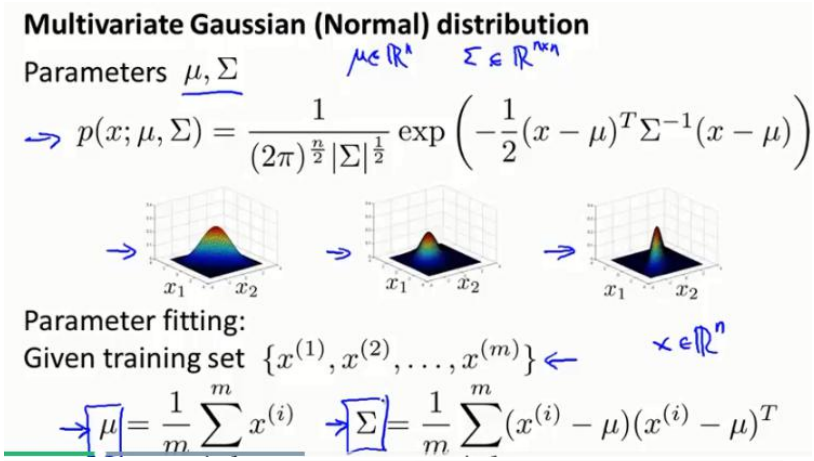


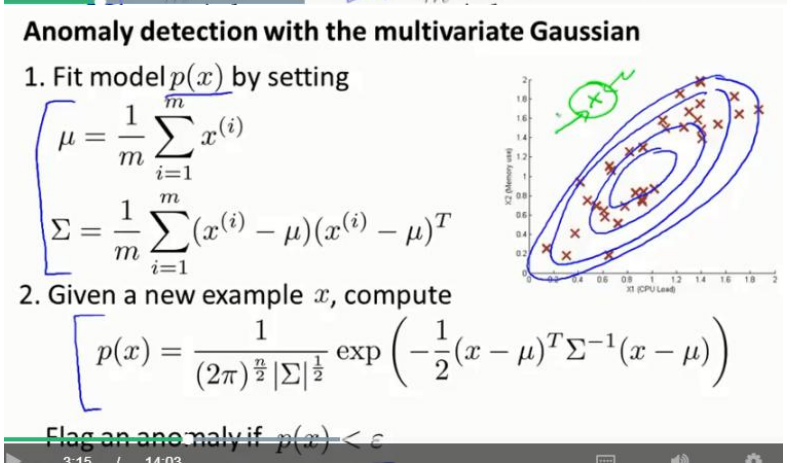
Multivariate Gaussian (Normal) examples



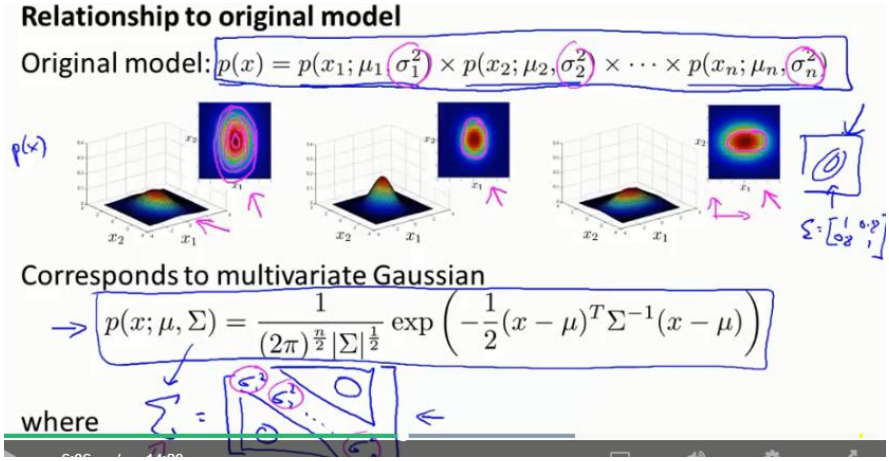


3-2 Anomaly Detection using the Multivariate Gaussian Distribution





之前提到的第一个高斯分布模型其实是多元高斯模型的特殊化，把方差代入协方差矩阵的对角线中。

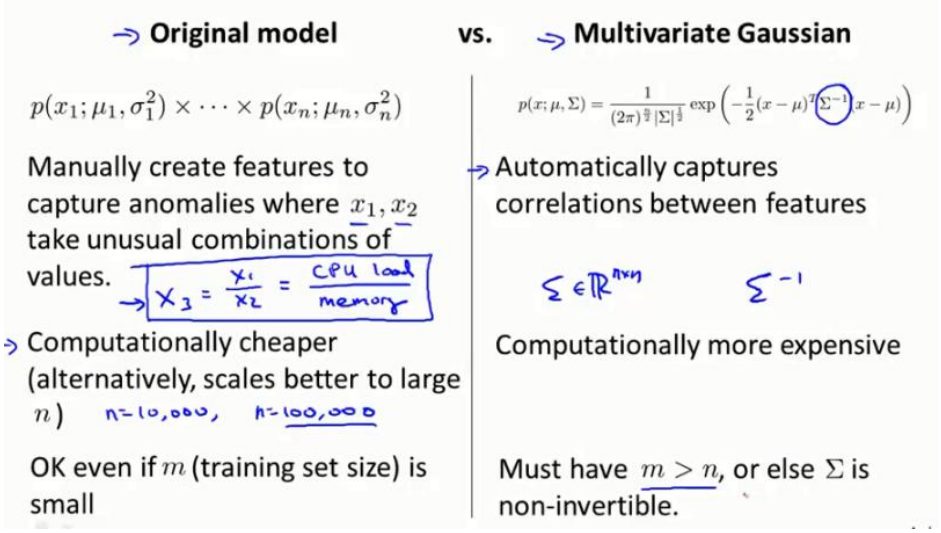


• **两种方法之间的对比**

1、若特征之间有相关性，用高斯分布模型需要创建新的特征参量；而多元高斯分布则会自动地捕捉这种相关性。

2、高斯分布模型计算量更小；多元高斯分布模型计算量更大。

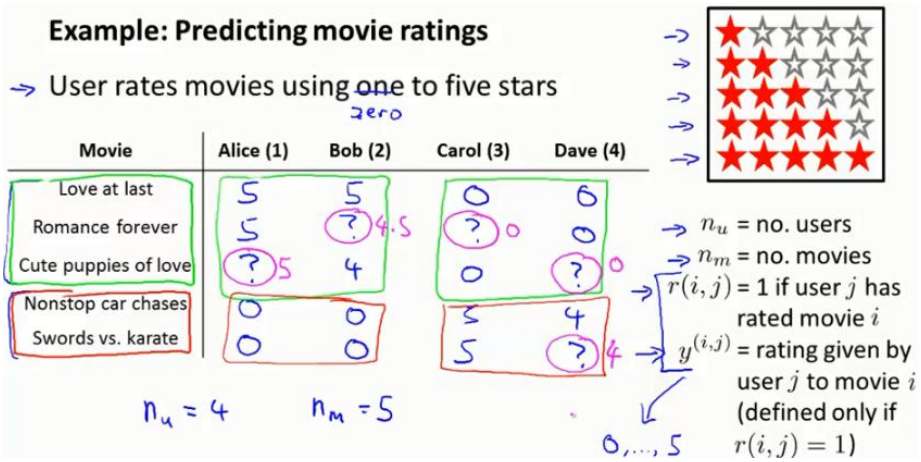
3、即使样本数很小也能运行；多元高斯分布必须要有样本数远大于特征数（m≥10n），或者**协方差矩阵是不可逆(奇异)**的。可能原因：1)不满足 m>n；2)有冗余的特征。



Week 8 Recommender Systems

4 Predicting Movie Ratings

4-1 Problem Formulation



In our notation, $r(i,j) = 1$ if user j has rated movie i, and $y^{(i,j)}$ is his rating on that movie.
Consider the following example (no. of movies $n_m = 2$, no. of users $n_u = 3$):

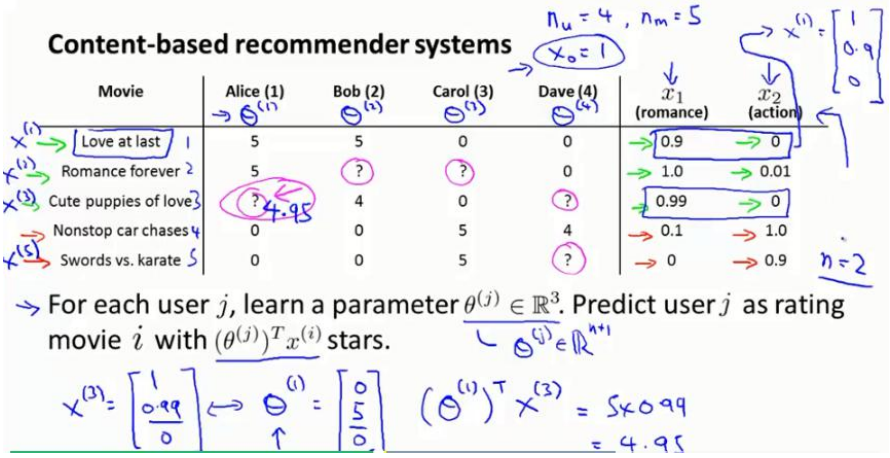
.	User 1	User 2	User 3
Movie 1	0	1	?
Movie 2	?	5	5

What is $r(2,1)$? How about $y^{(2,1)}$?

- $r(2,1) = 0,\ y^{(2,1)} = 1$
- $r(2,1) = 1,\ y^{(2,1)} = 1$
- $r(2,1) = 0,\ y^{(2,1)} = \text{undefined}$

4-2 Content Based Recommendations

(基于内容的推荐)其实是线性回归的一种变形



Theta 为用户的偏好特征、X 为描述电影分类的样本。

Problem formulation

- $r(i,j) = 1$ if user j has rated movie i (0 otherwise)
- $y^{(i,j)}$ = rating by user j on movie i (if defined)
- $\theta^{(j)}$ = parameter vector for user j
- $x^{(i)}$ = feature vector for movie i
- For user j, movie i, predicted rating: $(\theta^{(j)})^T(x^{(i)})$
- $m^{(j)}$ = no. of movies rated by user j

To learn $\theta^{(j)}$:

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

r 是否对电影进行评分、y 具体的评分、Theta 用户的参数、x 电影的特征、m 用户评分的电影数量。与普通的优化代价不同在于我们考虑了所有的用户进去。

Optimization objective:

To learn $\theta^{(j)}$ (parameter for user j):

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

To learn $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$:

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

- 基于内容的推荐系统其实就是一种线性回归。

Optimization algorithm:

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

Gradient descent update:

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

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

5 Collaborative Filtering

5-1 Collaborative Filtering

(协同过滤)

- **我们也可以通过评分值和用户的口味来倒推一部电影的参数。**

Consider the following movie ratings:

.	User 1	User 2	User 3	(romance)
Movie 1	0	1.5	2.5	?

Note that there is only one feature x_1. Suppose that:

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \theta^{(2)} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \theta^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

What would be a reasonable value for $x_1^{(1)}$ (the value denoted "?" in the table above)?

- 0.5

正确

Optimization algorithm

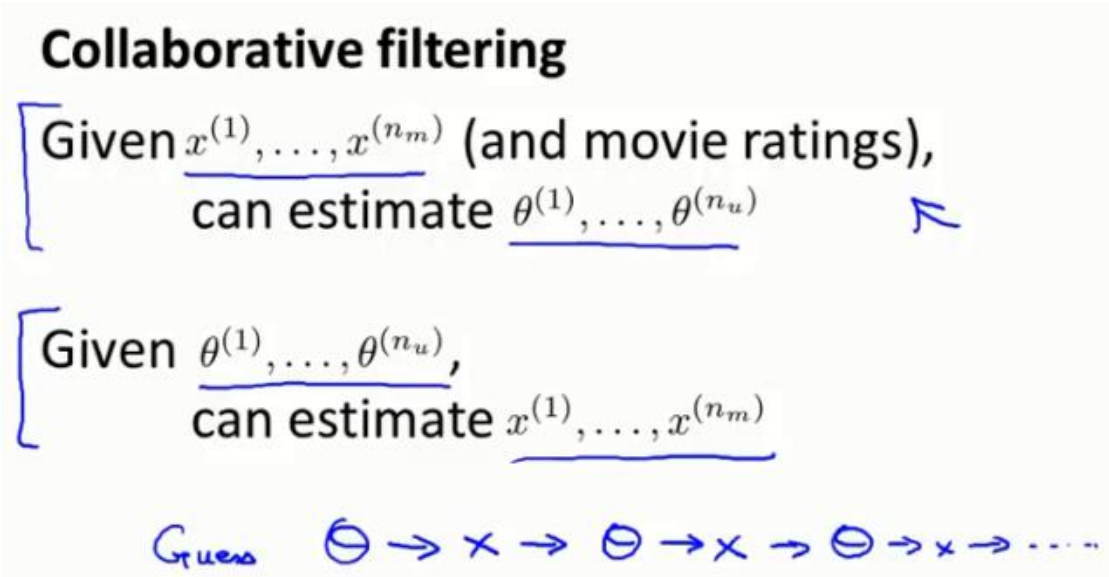
Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(i)}$:

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

Given $\theta^{(1)}, \dots, \theta^{(n_u)}$, to learn $x^{(1)}, \dots, x^{(n_m)}$:

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

- 类似于先有鸡还是先有蛋的问题，我们先随机猜一个 Theta 值然后推导出 X，然后不断地循环，从而收敛到一个良好的 X 与 Theta 值。
- **协同过滤算法：每个用户的评分都是为了集体，从而帮助学习出更好的特征。**

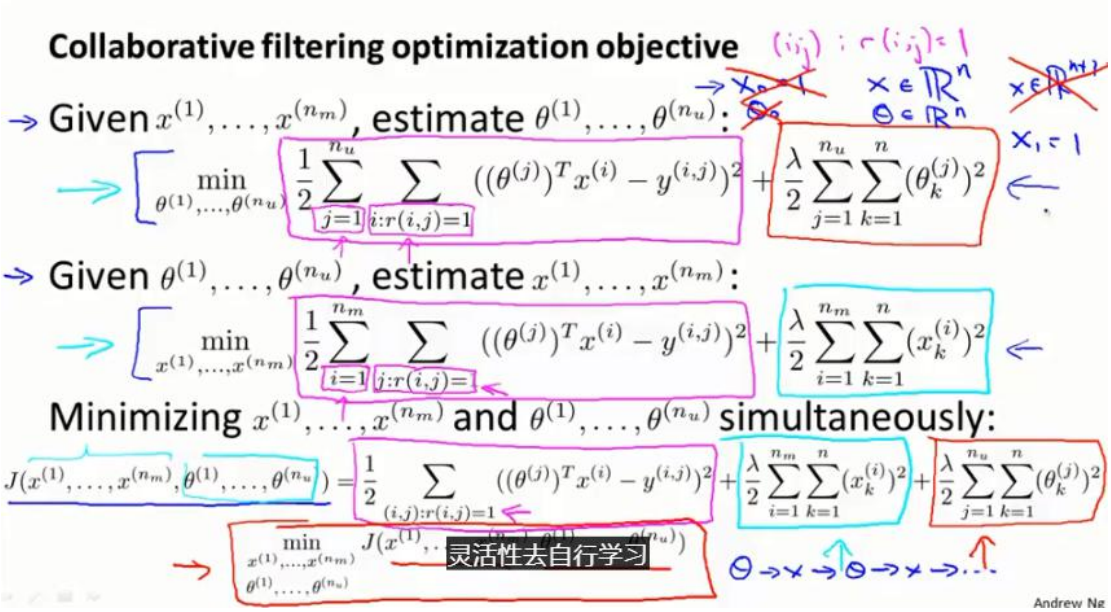


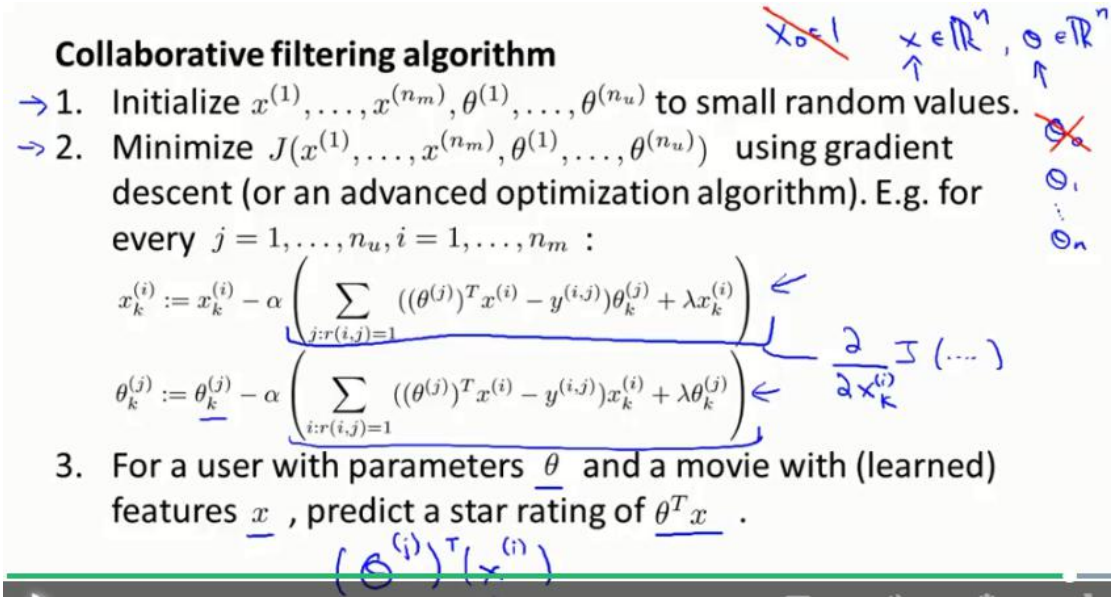
5-2 Collaborative Filtering Algorithm

• 先前我们要么给出 x 计算 Theta、要么给出 Theta 计算 x，现在我们将两则合并。我们发现两者优化目标的第一项其实是相同的。

• 合并后的优化目标把 Theta 或 x 看作不变的就回到了两种 情况中的各一种。

• **值得注意的事，我们的 x 将不再是 n+1 维，而是采用 n 维的向量。**





- 我们需要进行随机初始化破坏对称性，类似于神经网络，然后进行梯度下降法。

This serves as symmetry breaking (similar to the random initialization of a neural network's parameters) and ensures the algorithm learns features $x^{(1)},\dots,x^{(nm)}$ that are different from each other.

6 Low Rank Matrix Factorization

6-1 Vectorization: Low Rank Matrix Factorizatio

低秩矩阵分解

- 将打分情况转化为矩阵。

Users who have not rated any movies

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	Eve (5)
Love at last	5	5	0	0	?
Romance forever	5	?	?	0	?
Cute puppies of love	?	4	0	?	?
Nonstop car chases	0	0	5	4	?
Swords vs. karate	0	0	5	?	?

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix}$$

Let $X = \begin{bmatrix} - & (x^{(1)})^T & - \\ & \vdots & \\ - & (x^{(n_m)}) & - \end{bmatrix}$, $\Theta = \begin{bmatrix} - & (\theta^{(1)})^T & - \\ & \vdots & \\ - & (\theta^{(n_u)} & - \end{bmatrix}$.

What is another way of writing the following:

$$\begin{bmatrix} (x^{(1)})^T(\theta^{(1)}) & \dots & (x^{(1)})^T(\theta^{(n_u)}) \\ \vdots & \ddots & \vdots \\ (x^{(n_m)})^T(\theta^{(1)}) & \dots & (x^{(n_m)})^T(\theta^{(n_u)}) \end{bmatrix}$$

- $X\Theta$
- $X^T\Theta$
- $X\Theta^T$

正确

• **How 推荐相关电影，保证两个特征的差值在一定的范围内。**

Finding related movies

For each product i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$.

→ x_1 = romance, x_2 = action, x_3 = comedy, x_4 = ...

How to find movies j related to movie i?

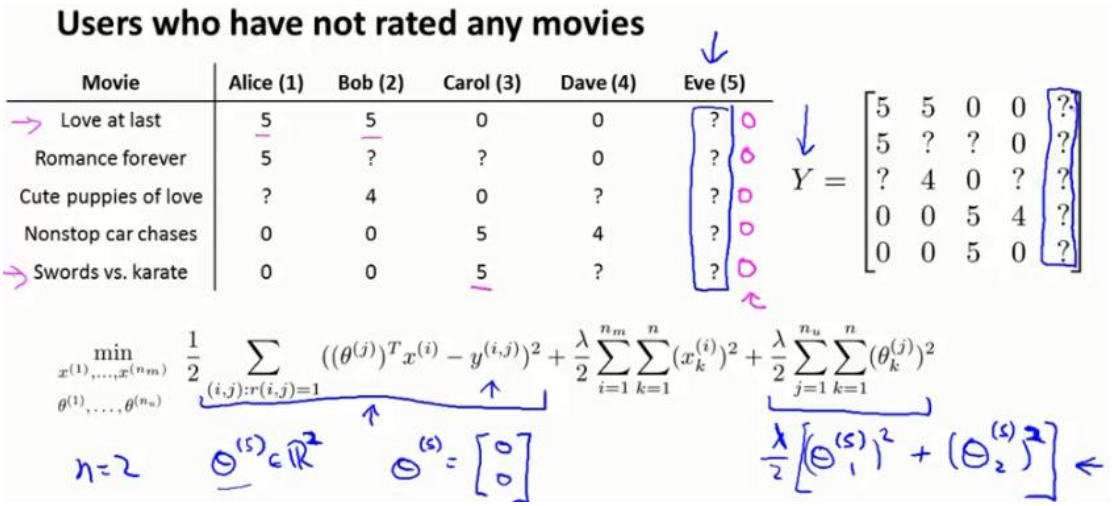
small $\|x^{(i)} - x^{(j)}\|$ → movie j and i are "similar"

5 most similar movies to movie i:
Find the 5 movies j with the smallest $\|x^{(i)} - x^{(j)}\|$.

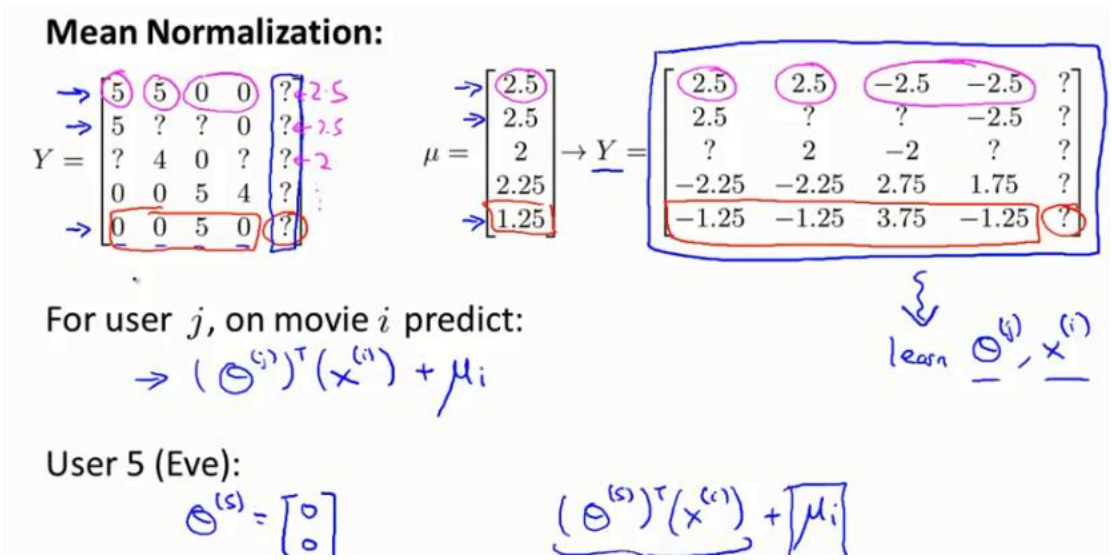
6-2 Implementational Detail: Mean Normalization

(均值归一化)

•我们加入了一个新的用户 Eve，它没有给任何一部电影打过分，所以计算优化目标的时候，第一项不影响，唯一影响的是第三项的归一化，但为了最小化目标函数，将会给 Eve 赋值为 [0;0]



• 为了解决这种情况，我们先计算所有已知值的平均值（行），然后将 Y 减去平均值 u，这样使得 Y 均值归一化，然后计算实际得分的时候，我们再把平均值 u 给加回来，那么若未给电影打过分，该项将会被设置为（0+其他人打分的平均值），这是非常合理的。



• 我们减去了平均值，却没有除去他们的方差。为什么？因为电影的评分标准以及在 0-5 分内，在一个相同的尺度了，所以没有必要进行特征缩放。

We talked about mean normalization. However, unlike some other applications of feature scaling, we did not scale the movie ratings by dividing by the range (max – min value). This is because:

- This sort of scaling is not useful when the value being predicted is real-valued.
- All the movie ratings are already comparable (e.g., 0 to 5 stars), so they are already on similar scales.

正确

7 Test

• 代价函数的表达形式

- [ ] $\frac{1}{m}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}(\sum_{k=1}^{n}(\theta^{(k)})_j x_i^{(k)} - y^{(i,j)})^2$
- [x] $\frac{1}{m}\sum_{i=1}^{n_m}\sum_{j:r(i,j)=1}(\sum_{k=1}^{n}(\theta^{(j)})_k x_k^{(i)} - y^{(i,j)})^2$
- [ ] $\frac{1}{m}\sum_{(i,j):r(i,j)=1}((\theta^{(j)})^T x^{(i)} - r(i,j))^2$
- [x] $\frac{1}{m}\sum_{(i,j):r(i,j)=1}((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2$

• 如何分开处理用户是否对电影进行评分。

```
C = A * B;
total = 0;
for i = 1:5
  for j = 1:5
    if (R(i,j) == 1)
      total = total + C(i,j);
    end
  end
end
```

Which of the following pieces of Octave code will also corr
all that apply. Assume all options are in code.

- [x] total = sum(sum((A * B) .* R))
- [x] C = (A * B) .* R; total = sum(C(:));

Week 10 Large Scale Machine Learning

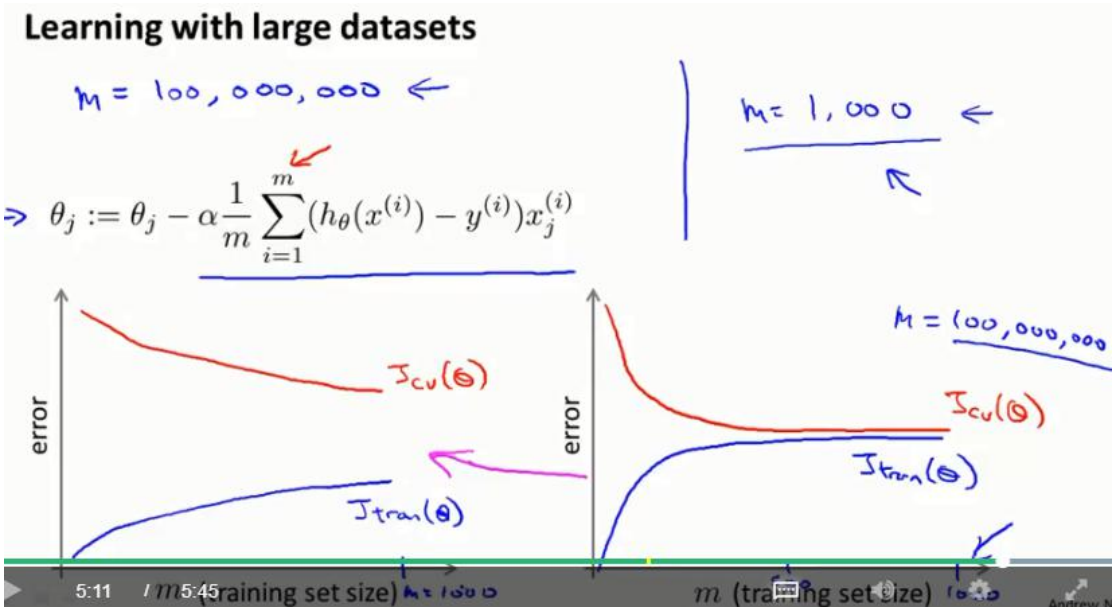
1 Gradient Descent with Large Datasets

1-1 Learning With Large Datasets

使用一个低偏差的学习算法，然后用很多数据来训练它。
• 假设我们拥有一亿个数据，我们可以先拿出 1000 个来训练它，目的是：

Plot a learning curve for a range of values of m and verify that the algorithm has high variance when m is small.

做出其学习曲线，然后判断在样本数小的时候方差是否比较大，若是这样的话，使用大数据就有帮助。



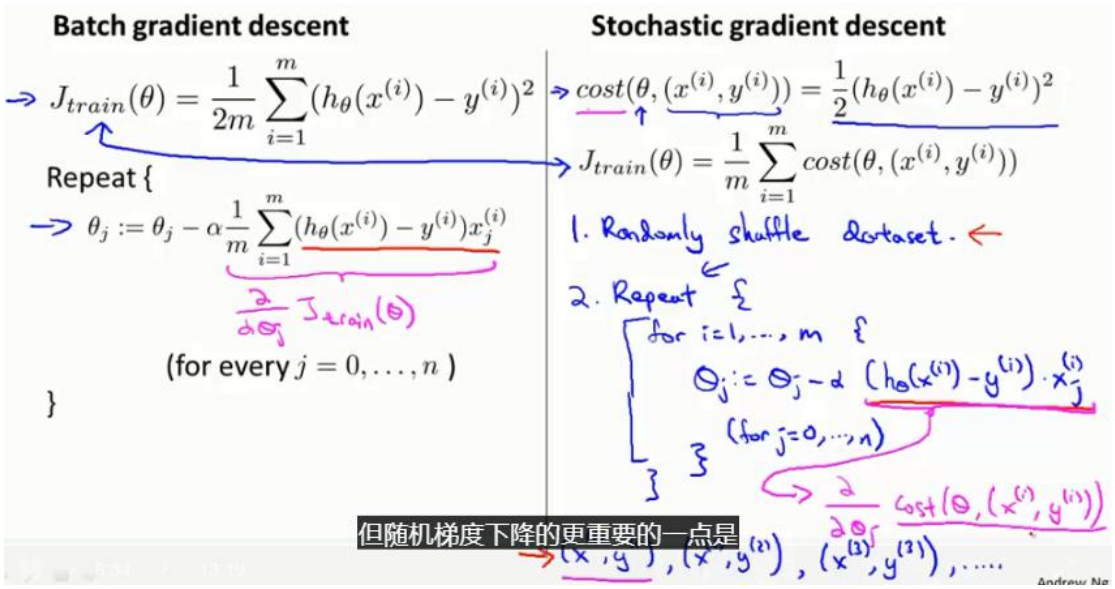
1-2 Stochastic Gradient Descent

(随机梯度下降)

• 普通的梯度下降法**(批量梯度下降法 batch gradient descent)**当样本数较大的时候，计算量会非常大。

• 假设我们有三亿个样本，我们每迭代一次就要重新读取一次数据，费时。

•随机下降法：首先需要将样本随机排布，然后我们定义代价函数为只对**一个样本**进行计算，那么这样在迭代的过程中，代价函数将不断地对参数进行优化、并且计算代价值，不断地收敛。此外，我们不需要使用全部样本，只需要使用一部分样本即可。

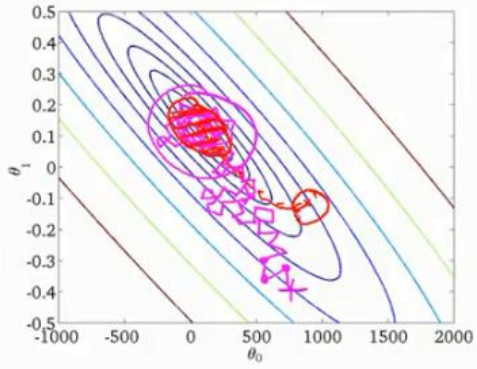


•两种算法比较，批量下降法缓慢地向低代价值移动，而随机下降法可能下降的趋势很曲折。

Stochastic gradient descent

1. Randomly shuffle (reorder) training examples

2. Repeat {
   for $i := 1, \ldots, m$ {
   $\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
   (for every $j = 0, \ldots, n$)
   }
   }



1-3 Mini-Batch Gradient Descent

(小批量梯度下降)

- 三种算法的比较：**小批量梯度下降，每次迭代对 b 个样本求和。**

Mini-batch gradient descent

Batch gradient descent: Use all m examples in each iteration

Stochastic gradient descent: Use 1 example in each iteration

Mini-batch gradient descent: Use b examples in each iteration

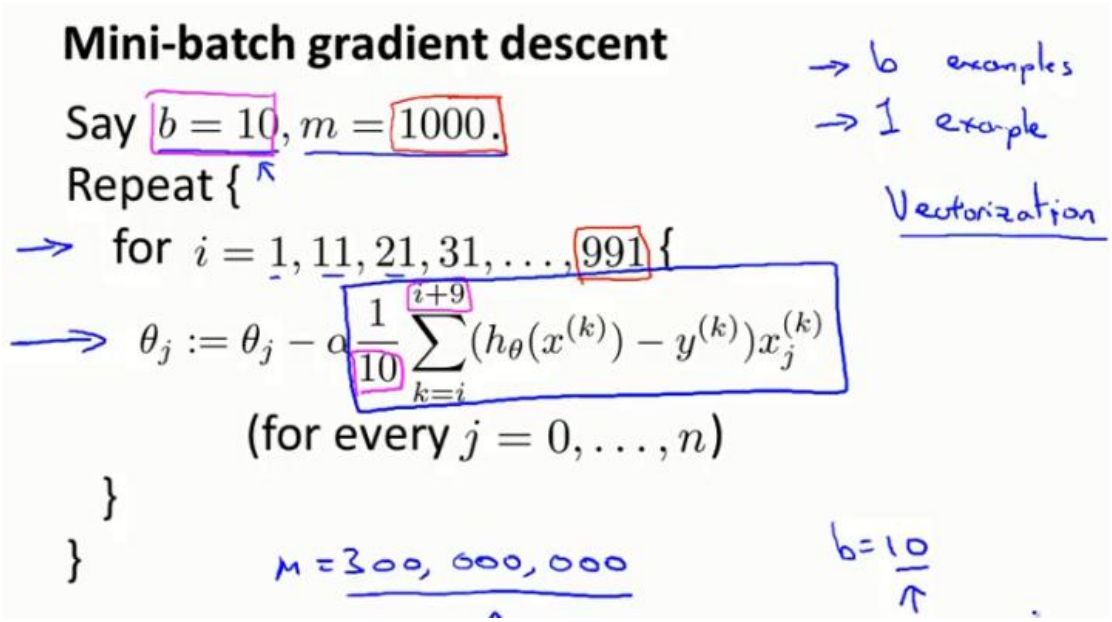
b = mini-batch size. $b = 10$. 2-100

Get $b = 10$ examples $(x^{(i)}, y^{(i)}), \ldots (x^{(i+9)}, y^{(i+9)})$

$$\theta_j := \theta_j - \alpha \frac{1}{10} \sum_{k=i}^{i+9} (h_\theta(x^{(k)}) - y^{(k)}) \cdot x_j^{(k)}$$

$i := i + 10$

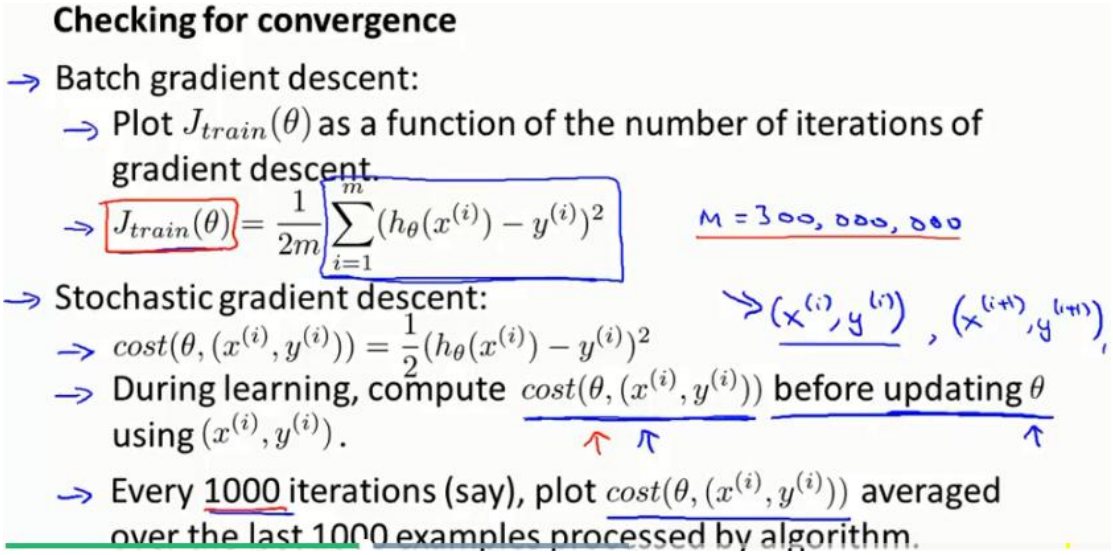
- 具体实现



每 b 个样本，我们迭代一次，更新一次参数，当 **b 等于样本数**的时候，就变成了批量下降法。

1-4 Stochastic Gradient Descent Convergence

• 如何确保随机梯度下降法能正常收敛并且调整随机梯度下降中学习速率 α 的值。



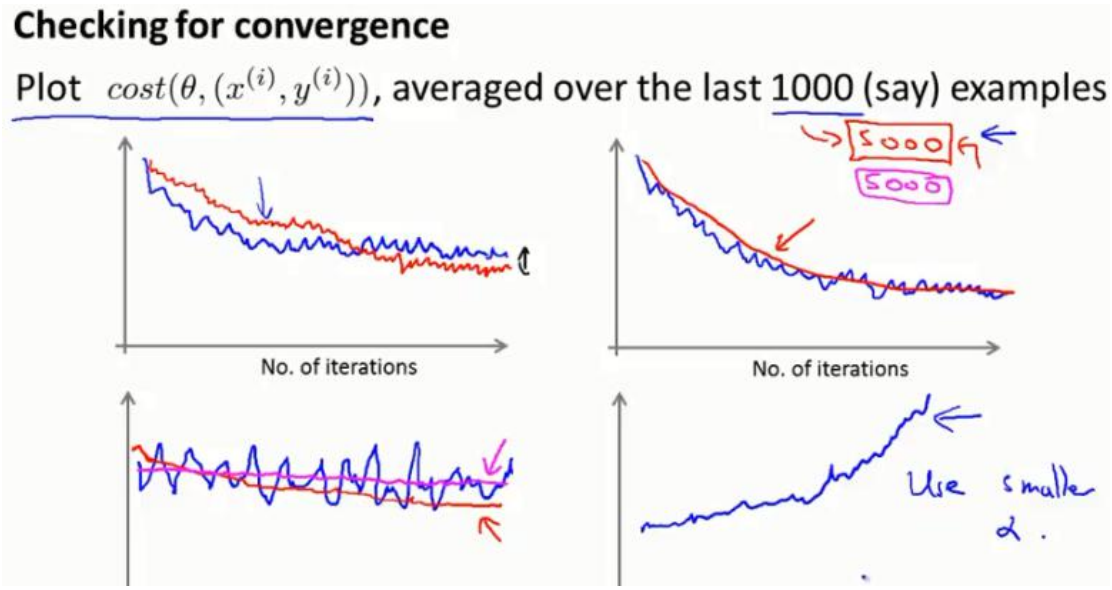
• 我们在更新参数 Theta 之前，需要先计算他对应的代价值；否则更新后计算出来的代价值将会比实际更好。

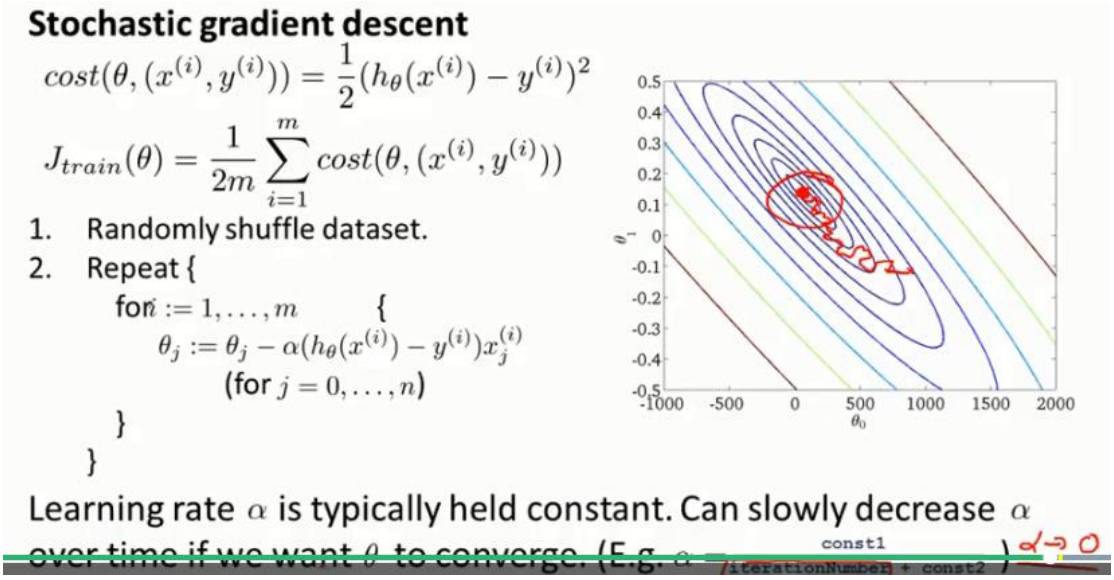
• 在 Batch Gradient Descent 中检测算法是否收敛的方法是绘制一个图像，横坐标是迭代次数，纵坐标是 J(θ)。对于 Stochastic Gradient Descent 同样可以使用这个方法。区别是因为数据量大，每 **k 次迭代（k 个样本）**绘制一个点。

• 注意在 Stochastic Gradient Descent 时，对于第 i 个样本我们是**先计算 cost(i)，再用 x(i)优化模型。**

绘制出的图像会是震荡的。• 若样本看起来非常地粗糙，则需要增大样本数量；若代价函数

在 上 升 ， 则 需 要 减 小 学 习 速 率 α 。





• 使用随机下降法通常非常曲折，并且难以收敛到全局最小值。那么我们可以让学习速率 α 随着迭代次数的增大而不断地下降，如如 α =c1/(c2+迭代次数)。

2 Advanced Topics

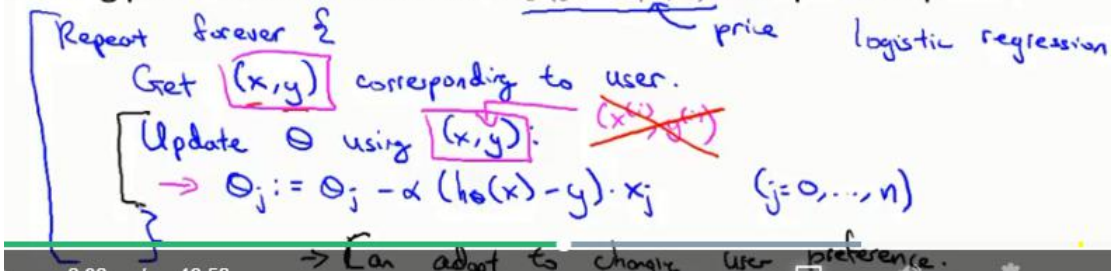
2-1 Online Learning

(在线学习机制)
连续的用户流(数据流)，使用用户的数据优化网站的决策。

Online learning

Shipping service website where user comes, specifies origin and destination, you offer to ship their package for some asking price, and users sometimes choose to use your shipping service ($y = 1$), sometimes not ($y = 0$).

Features x capture properties of user, of origin/destination and asking price. We want to learn $p(y = 1|x; \theta)$ to optimize price.



预测点击率 CTR

Other online learning example:

Product search (learning to search)

User searches for "Android phone 1080p camera"

Have 100 phones in store. Will return 10 results.

→ x = features of phone, how many words in user query match name of phone, how many words in query match description of phone, etc.

→ $y = 1$ if user clicks on link. $y = 0$ otherwise.

→ Learn $p(y = 1|x; \theta)$. predicted CTR

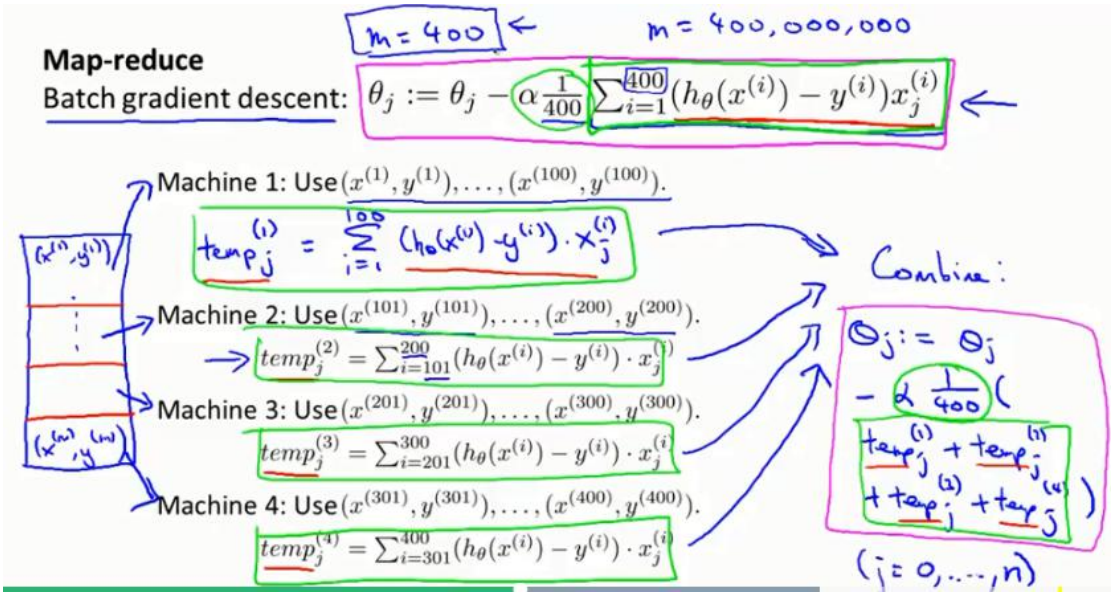
→ Use to show user the 10 phones they're most likely to click on.

Other examples: Choosing special offers to show user; customized selection of news articles; product recommendation; ...

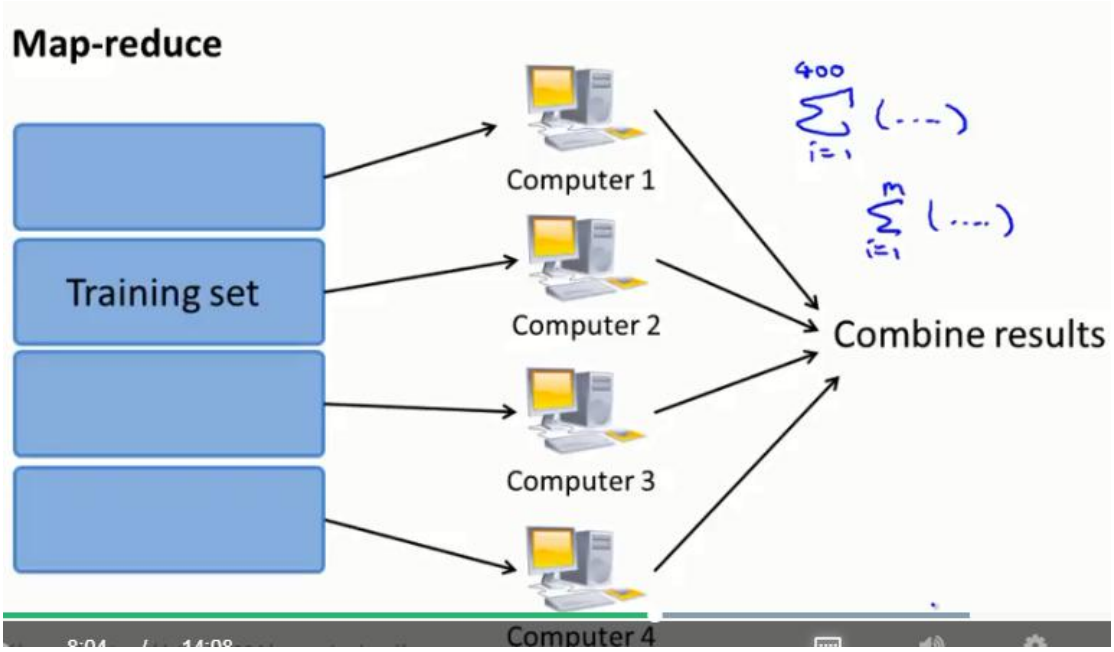
- 与随机梯度下降法的区别在于：我们不会使用固定的数据集。

2-2 Map Reduce and Data Parallelism

(映射约减)



• 数据量很大的话，我们把数据分成多份，让每个计算机并行地计算一部分，最后把数据组合起来更新参数值。



理论上计算速度会提升为只用一台计算机的 1/4，而实际上总是小于这个值，因为有各种原因如网络延迟。

当学习算法的目的是为了计算某种**求和**就可以使用映射约减这种方法。

Map-reduce and summation over the training set

Many learning algorithms can be expressed as computing sums of functions over the training set.

E.g. for advanced optimization, with logistic regression, need:

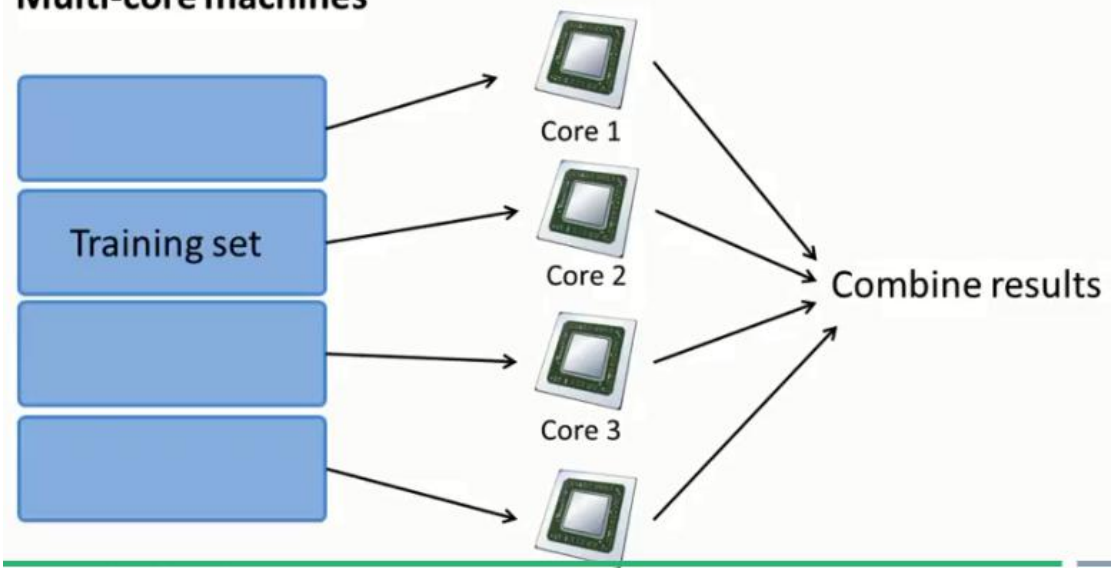
$$J_{train}(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) - (1-y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

$$\frac{\partial}{\partial \theta_j} J_{train}(\theta) = \frac{1}{m}\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

temp(i) temp(i) ←

•除了用多个计算机，我们可以采用使用一台计算机的多核。这样子还可以避免网络延迟的问题，但是需要注意某些线性代数的库已经自动调用了多核并行处理。

Multi-core machines



Week 11 Application Example: Photo OCR

1 Photo OCR

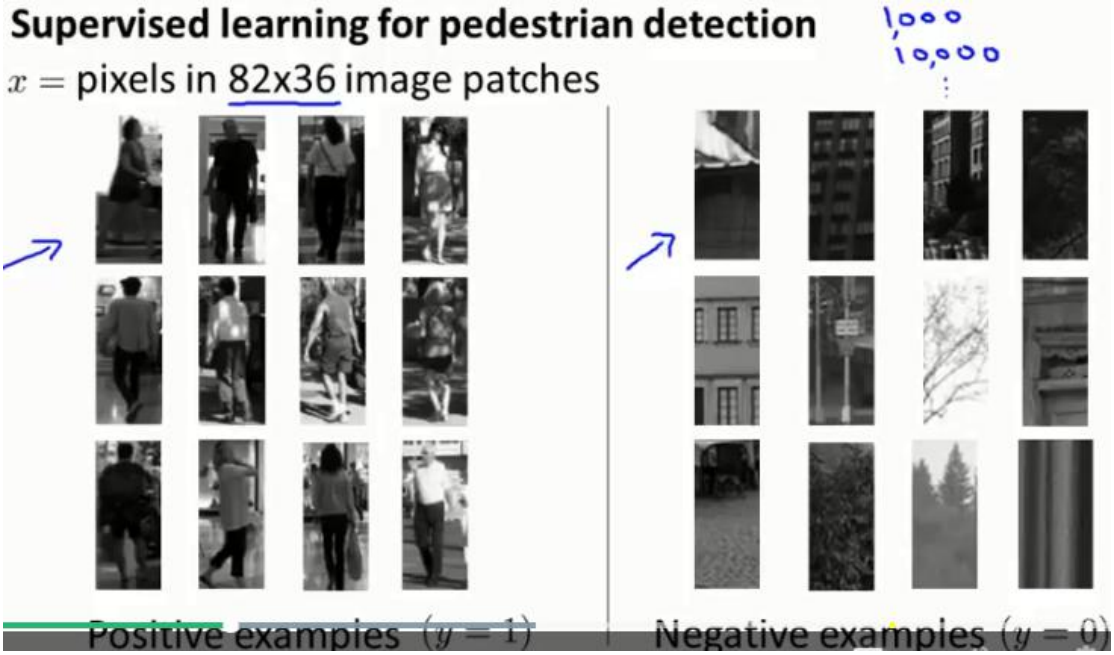
1-1 Problem Description and Pipeline

• 机器学习在照片 OCR 技术中的应用：让计算机光学识别照片中的文字信息。
• **文字检测、字符分割、字符分类**。通过机器学习**流水线 Pipeline**，我们的问题往往是如何设计这个流水线。

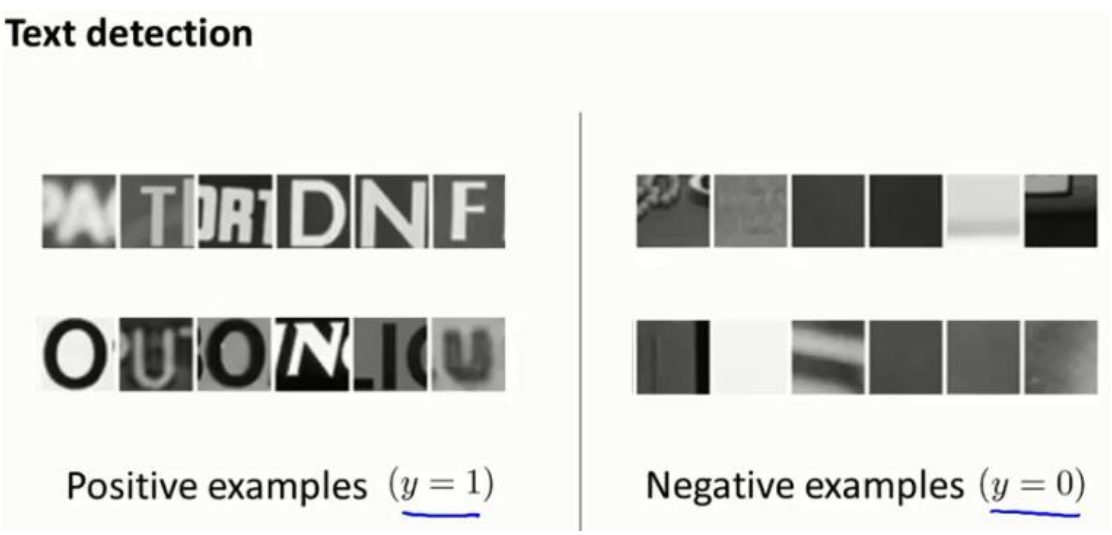
1-2 Sliding Windows

• 以识别路人为例，路人往往非常相似，具有一定的长宽比。

我们收集了具有相同长宽比的样本，然后作为监督学习系统的输入。

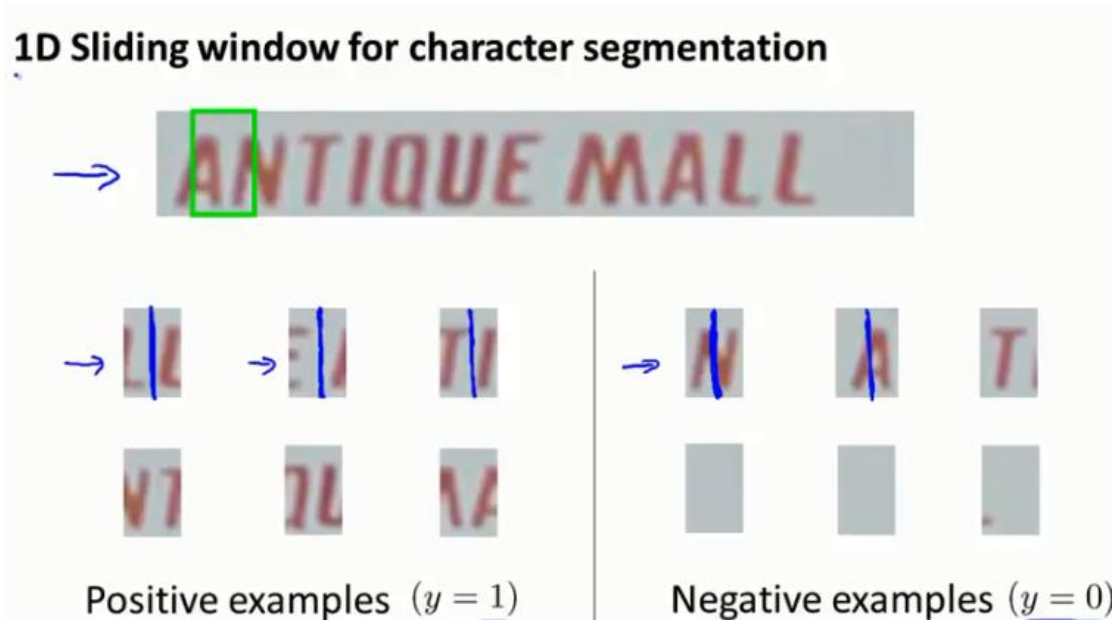


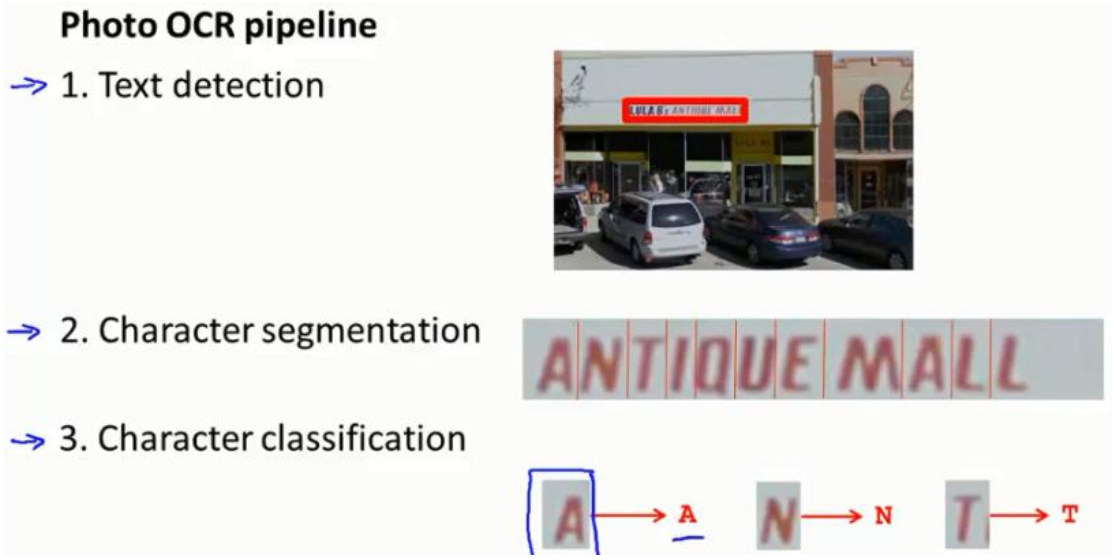
• 先从左上角取一个长宽比一定的矩形框，然后开始以步长为 4 或 8 移动，直至遍布整张图片，从而能够检测出行人来。



• **文字检测：**识别完成后，我们得到左下角的图，白色表示出现文字概率较高；然后将其输入到一个展开器中，展开器将会将白色区域拓展一部分，然后我们通过人为判断选择出有文字的区域块可以作为流水线的下一个输入。

• 字符分割：训练一个分类器，来判断字符中间是否有分界线，依然是通过滑动窗，从左向右。





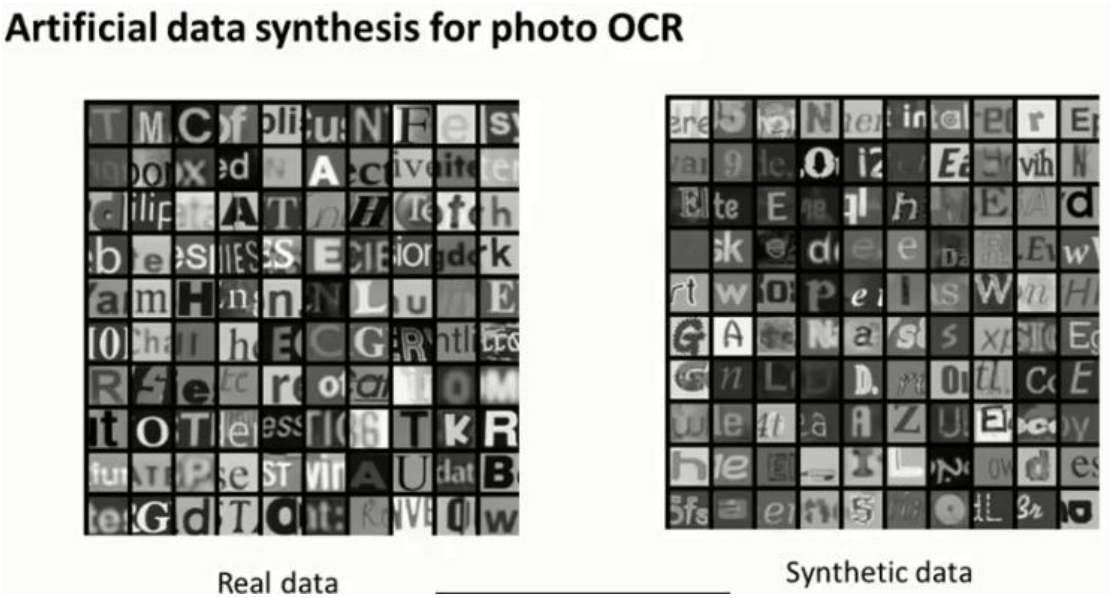
1-3 Getting Lots of Data and Artificial Data

(人工数据合成)

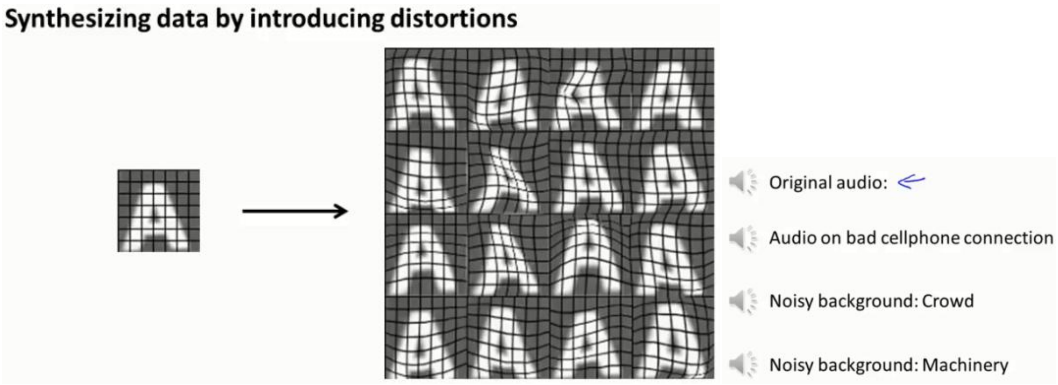
一个高级的机器学习算法最好采用低偏差然后使用大量的数据训练它。

• 人工数据合成通常包含两类：**①完全创造新的数据；②已经有一小部分带标签的训练集，然后我们扩大这个训练集。**

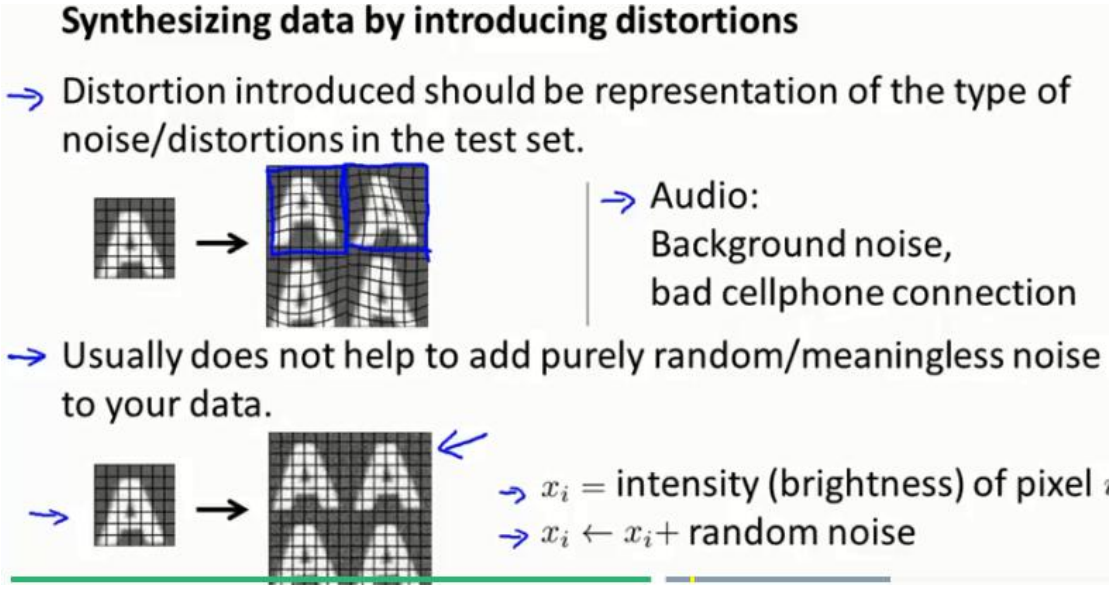
①左图中是我们已有的真实数据集，右图是我们通过左图进行处理如：将字母取出结合不同的背景，进行模糊处理、旋转等等。



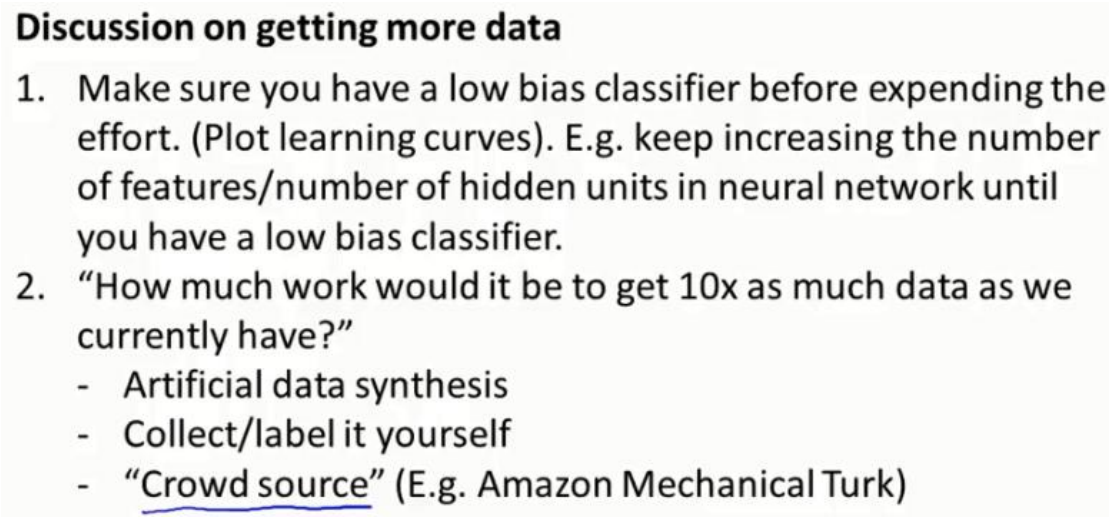
②我们可以取出一个字母，将他进行网格划分，然后对这个图形进行扭曲处理，那么这样一下子就得到了很多个新样本；再比如我们有一个纯净的人声在念数字、我们可以给这个声音加以背景声效模拟不同的情况，从而获得更多的训练样本。



• 需要注意，你所做的这些处理都应该是要有真实意义，这样才有代表性。



• **总结**



1 在使用人工数据合成之前，使用学习曲线去保证模型是正确的，即低偏差 or 高方差模型，这样收集数据才有意义。

2 仔细要花多少工作量才能收集比现在多十倍的数据呢？

人工数据合成、手动收集 or 标记、使用“众包”服务。

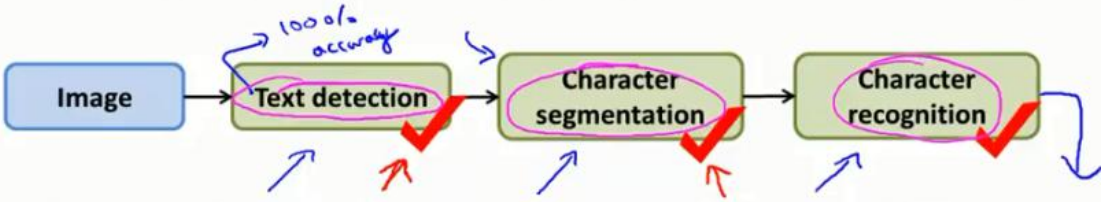
1-4 Ceiling Analysis:What Part of the Pipeline to Work on Next

(上限分析)

• 如何分配流水线中每个环节所需要耗费的时间呢？我们需要一个数值评价量度。

• **即改善每个模块，系统的上升空间是多少？**如下图中的例子，如果我们改善了字符检测，将提升 17%；继续改善字符分割，只提升 1%；若继续改善文字识别，将提升 10%。当这些模块绝对完美时，这些数值就是他们的上限分析。

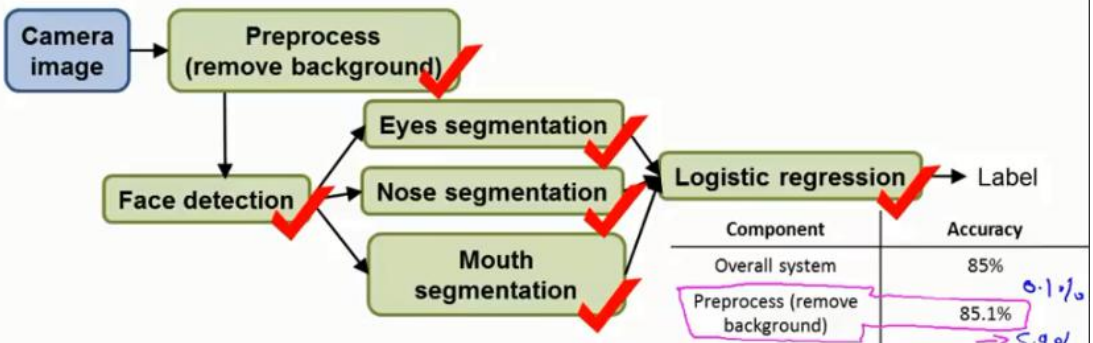
Estimating the errors due to each component (ceiling analysis)



What part of the pipeline should you spend the most time trying to improve?

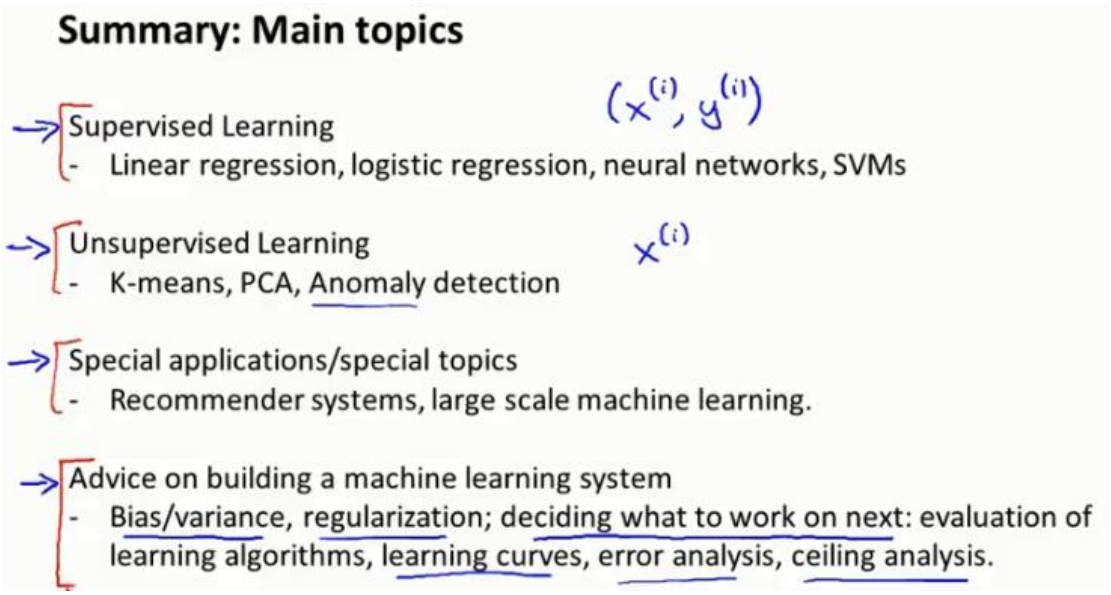
Component	Accuracy
Overall system	72%
Text detection	89%
Character segmentation	90%
Character recognition	100%

Another ceiling analysis example



Component	Accuracy
Overall system	85%
Preprocess (remove background)	85.1%
Face detection	91%
Eyes segmentation	95%
Nose segmentation	96%
Mouth segmentation	97%
Logistic regression	100%

2 Conclusion



习题参考答案：http://blog.csdn.net/a1015553840/article/details/50781173