week 1. Machine Learning

1.3 supervised M.L. {regression, classification}

1.4 unsupervised M.L. {clustering}

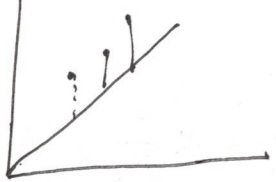
具体的类参见 book: hands-on-sklearn

2. Linear Regression with one variable

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

如何选择合适的 θ_0, θ_1.

modeling error:



选使 modeling error 平方和最小的参数.

即使 cost func 最小

$$J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right]^2$$

自动找出使 $J(\theta_0,\theta_1)$ 最小的参数的算法.

——— Gradient Descent.

不断搜索下降平均方向.

Batch gradient descent:

repeat until convergence { $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0,\theta_1)$, $(j=0$ ✓

α is learning rate.

correct: simultaneous update:

temp 0 := $\theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0,\theta_1)$

temp 1 := $\theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0,\theta_1)$

$\theta_0 :=$ temp 0

$\theta_1 :=$ temp 1 , 同时更新 θ_0, θ_1, 之后也就更新了 $J(\theta_0,\theta_1)$

2.7 Gradient Descent for Linear Regression.

Gradient Descent algorithm	Linear Regression model
repeat until converge { $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0,\theta_1)$. ($j=0$ or 1) }	$h_\theta(x) = \theta_0 + \theta_1 x$ $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right]^2$

$j=0$, $\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right]^2$

$j=1$, $\frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1} = \frac{1}{m}\sum_{i=1}^{m}\left[h_\theta(x^{(i)}) - y^{(i)}\right] \cdot x^{(i)}$

3. Linear Algebra Review

矩阵乘法：

性质：$A \times B \neq B \times A$

$A \times (B \times C) = (A \times B) \times C$

identity matrix：$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$A A^{-1} = A^{-1} A = I$

$AI = IA = A$

转置 transpose：$\begin{vmatrix} a & b \\ c & d \\ e & f \end{vmatrix}^T = \begin{vmatrix} a & c & e \\ b & d & f \end{vmatrix}$

$(A \pm B)^T = A^T \pm B^T$

$(A \times B)^T = B^T \times A^T$

$(A^T)^T = A$

$(kA)^T = kA^T$ ？k是什么？

Week 2

4. Linear Regression with Multiple Variables

$x_j^{(i)}$ 代表第 i 行 [illegible] 第 j 个特征。

$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$

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

cost function：

$J(\theta_0, \theta_1, \cdots \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x^{(i)}) - y^{(i)}]^2$

Gradient Descent：

Repeat {

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

批量的：

$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \cdot [h_\theta(x^{(i)}) - y^{(i)}] x_j^{(i)}$

simutaneously update θ_j for $j = 0, 1, \cdots, n$.

Feature scaling：Make sure features are on a similar scale.

mean normalization：$x_n = \frac{x_n - \mu_n}{s_n}$，$\mu_n$ 是平均值，s_n 是标准差

Learning Rate：太大：不收敛

太小：学习太慢

4.5 features and Polynomial Regression.

非线性：

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

$h_\theta(x)=\theta_0+\theta_1 x_1+\theta_2 x_1^2$

$h_\theta(x)=\theta_0+\theta_1 x_1+\theta_2 x_1^2+\theta_3 x_1^3$

$h_\theta(x)=\theta_0+\theta_1 x_1+\theta_2\sqrt{x_1}$

⋮

若用 polynomial Regression, feature scaling 很重要，eg：

$x_1 \sim 1\sim 1000$

$x_2^2 \sim 1\sim 10^6$

4.6 Normal equation. —只适用 linear model.

有些；①只用 Normal equ. 更好.

Normal equ. 是指解 $\theta=(X^TX)^{-1}X^Ty$ 找出使 $J(\theta)$ 最小的参数.

x_0	x_1	x_2	…	x_n	y

X y

与 Gradient decent 对比：见 P51.

$\theta=(X^TX)^{-1}X^Ty$ 的推导：

~~$J(\theta)=\frac{1}{2m}$~~

$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}[h_\theta(x^{(i)})-y^{(i)}]^2$，$h_\theta(x)=\theta^Tx$

向量形式：$=\theta_0x_0+\theta_1x_1+\dots+\theta_nx_n$

$J(\theta)=\frac{1}{2}(X\theta-y)^2$，$X$ 是 m 行 n 列，m 个样本，n 个特征.

θ 为 n 行 1 列.，y 为 m 行一列.

$J(\theta)=\frac{1}{2}(X\theta-x)^T(X\theta-y)$

$=\frac{1}{2}(\theta^TX^T-y^T)(X\theta-y)$

$=\frac{1}{2}(\theta^TX^TX\theta-\theta^TX^Ty-y^TX\theta+y^Ty)$

对 $J(\theta)$ 求偏导，

n 个性质：$\frac{dAB}{dB}=A^T$

$\frac{\partial J(\theta)}{\partial\theta}=\frac{1}{2}[2X^TX\theta-X^Ty-(y^TX)^T-0]$

$=\frac{1}{2}[2X^TX\theta-X^Ty-X^Ty-0]$

$=X^TX\theta-X^Ty$

$\frac{dX^TAx}{dx}=2AX$

$\frac{dx^TA}{dx}=A$

let $\frac{\partial J(\theta)}{\partial\theta}=0 \Rightarrow \theta=(X^TX)^{-1}X^Ty$

Week 3

6. Logistic Regression.

↑ 处理 classification, y 值是离散的值，通常 0~1, $0 \le h_\theta(x) \le 1$

Logistic Regression 的假设：

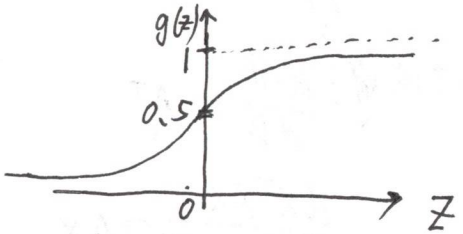
$$h_\theta(x) = g(\theta^T X)$$

X 是特征向量，

g 代表 logistic function

常用的一个 ~~是~~ logistic function 是 sigmoid function：

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



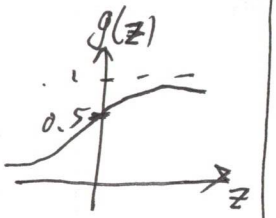
h(x) 的作用是：对于给定输入，计算该输入计算出输出变量=1的可能性，$h_\theta(x) = P(y=1|x;\theta)$

6.3 Decision boundary

Logistic Regression

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

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



当 $h_\theta(x) \ge 0.5$ 时，predic $y=1$

< 0.5 ， $y=0$

or:

$z=0$ 时， $g(z)=0.5$

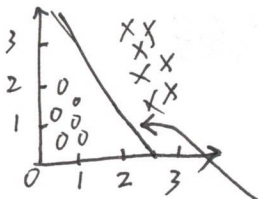
$z>0$ 时， $g(z) \to 0.5$

$z<0$ 时， $g(z) < 0.5$

or $\theta^T x > 0$ 时，predict $y=1$

$\theta^T x < 0$ ， $y=0$

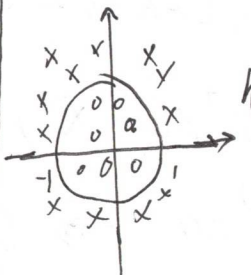
e.g.:



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

$\theta = [-3, 1, 1]$

当 $-3 + x_1 + x_2 > 0$ 即 $x_1 + x_2 \ge 3$ 时，predict $y=1$.



$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$

$\theta = [-1, 0, 0, 1, 1]$

6.4 Cost Function.

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

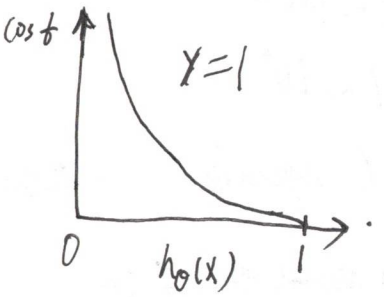
$$Cost\left[h_\theta(x^{(i)}), y^{(i)}\right]$$

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

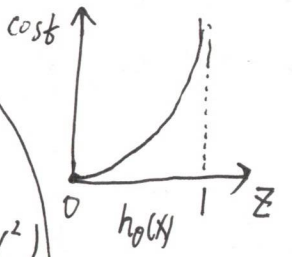
if $y=1$

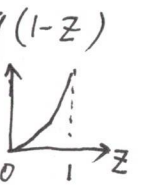
$cost=0$ if $y=1$, $h_\theta(x)=1$

as $h_\theta(x) \to 0$, $cost \to \infty$



if $y=0$





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

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

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

Logistic Gradient Descent:

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

(simultaneously update all θ_j) }

推导后得到：

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

(simultaneously update all θ_j) }

推导：

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

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

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

$$= y^{(i)}\log\left(\frac{1}{1+e^{-\theta^T x^{(i)}}}\right) + (1-y^{(i)})\log\left(1-\frac{1}{1+e^{-\theta^T x^{(i)}}}\right)$$

$$= -y^{(i)}\log(1+e^{-\theta^T x^{(i)}}) - (1-y^{(i)})\log(1+e^{\theta^T x^{(i)}})$$

So $\frac{\partial J(\theta)}{\partial\theta_j}$

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

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

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

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

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

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - \frac{e^{\theta^T x^{(i)}}}{1+e^{\theta^T x^{(i)}}}\right)x_j^{(i)}$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - \frac{1}{1+e^{-\theta^T x^{(i)}}}\right)x_j^{(i)}$$

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

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

这的 Gradient Descent 有 Feature scaling

还有其他方法：conjugate Gradient，BFGS，L-BFGS

6.5 Simplified cost function and Gradient Descent.

Logistic regression cost function:

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

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

合并式：

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

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

找出使 $\min J(\theta)$ 的参数 θ.

画出 gradient Descent:

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

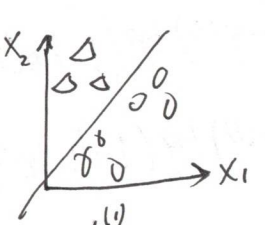
同时更新 all θ_j }

A vectorized implementation is:

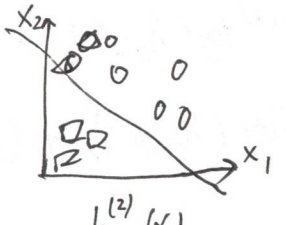
$$\theta:=\theta-\frac{\alpha}{m}X^T(g(X\theta)-\vec{y})$$

6.7 Multiclass Classification: one-vs-all

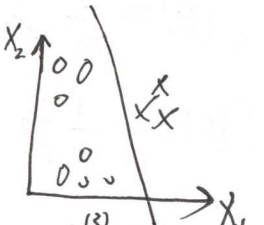
将多个类中的一个类标记为正向类 ($y=1$)，其他所类标记为负向类：



$h_\theta^{(1)}(x)$



$h_\theta^{(2)}(x)$



$h_\theta^{(3)}(x)$

7. Regularization.

7.1 The problem of overfitting.

过拟合怎么办：

① 去一些不能帮助正确预测的特征

② 正则化。保留所有特征，但减少参数的大小 (magnitude)

7.2 Cost Function.

模型 $h_\theta(x)=\theta_0+\theta_1x_1+\theta_2x_2^2+\theta_3x_3^3+\theta_4x_4^4$

对 θ_3,θ_4 进行惩罚.

修改后的代价函数：

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

若特征很多，不知道哪些.

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

λ is Regularization Parameter.

(按惯例不对 θ_0 惩罚)

7.3 Regularized Linear Regression.

Repeat until convergence {

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

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

$$=\theta_j\left(1-\alpha\frac{\lambda}{m}\right)-\alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)})-y^{(i)}\right)x_j^{(i)}$$

}

regularized linear regression 的 Gradient Descent 在每次把刚上一轮 θ 值减小了一个额外的值.

normal equ.

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

$(n+1)\times(n+1)$ 矩阵

if $m<n$, X^TX is non-invertible, when add $\lambda\cdot L$, $X^TX+\lambda\cdot L$ becomes invertible.

7.4 Regularized logistic Regression.

给 logistic regression 加 regularization 项.

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

Gradient Descent:

Repeat until converge. {

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

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

for $j=1,2,3,\cdots n$

}

Week 4

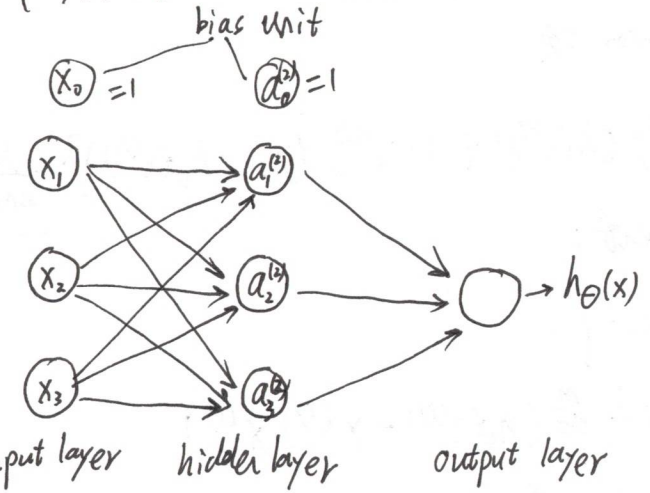
8. Neural Networks: Representation.

8.1 Non-linear Hypotheses. 特征太多，之前的回归模型不适用

8.2 Neurons and Brain

8.3 Model Representation. I.

一个3层的 Neural Network



$a_i^{(j)}$ 代表第 j 层的第 i 个激活单元.

$\Theta^{(j)}$ 代表从 j 层映射到 $j+1$ 层的权重矩阵

↑ size: 以 $j+1$ 层为行数

以 j 层~~为列数~~单元数+1为列数

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

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

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

$$h_\Theta(x)=a_1^{(3)}=g(\Theta_{10}^{(2)}a_0^{(2)}+\Theta_{11}^{(2)}a_1^{(2)}+\Theta_{12}^{(2)}a_2^{(2)}+\Theta_{13}^{(2)}a_3^{(2)})$$

从左向右称为 Forward propagation.

矩阵表示 $\Theta\cdot X=a$

$$X=\begin{matrix}x_0\\x_1\\x_2\\x_3\end{matrix},\ \Theta=\begin{matrix}\Theta_{10} & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \cdots\\ \cdots & \cdots & \cdots & \Theta_{33}\end{matrix},\ a=\begin{matrix}a_1\\a_2\\a_3\end{matrix}$$

8.4 Model Representation II

用向量化计算第二层的值:

$$x=\begin{vmatrix}x_0\\x_1\\x_2\\x_3\end{vmatrix},\ z^{(2)}=\begin{vmatrix}z_1^{(2)}\\z_2^{(2)}\\z_3^{(2)}\end{vmatrix},\ z^{(2)}=\Theta^{(1)}x,\ a^{(2)}=g(z^{(2)})$$

之后, 添加 $a_0^{(2)}=1$

$$g\left([\Theta_{10}^{(2)},\Theta_{11}^{(2)},\Theta_{12}^{(2)},\Theta_{13}^{(2)}]\times\begin{vmatrix}a_0^{(2)}\\a_1^{(2)}\\a_2^{(2)}\\a_3^{(2)}\end{vmatrix}\right)=\cdots=h_\Theta(x)$$

令 $z^{(3)}=\Theta^{(2)}a^{(2)}$, 则 $h_\Theta(x)=a^{(3)}=g(z^{(3)})$

这只是针对一个 training set 中一个 training unit 的计算, 要算整个 training set,

需将训练集特征矩阵转置, 使同一实例的特征在同一列里.

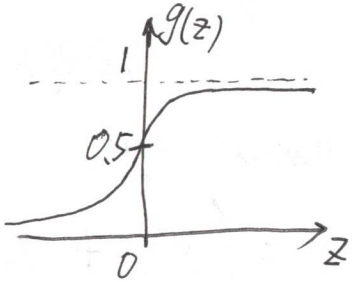
$$z^{(2)}=\Theta^{(1)}X^T$$

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

8.5 Examples and Intuitions I.

神经网络中, 单层神经元的计算

用来表示逻辑运算, And, Or, ...



8.6 Examples and Intuitions II

Binary logical operations 当输入bool值 0, 1, 我们可以用一个单一的激活层作为二元逻辑运算符，为表示不同的运算符(and, or, not)，只需选择不同的权重.

看图，理解.

8.7 Multiclass Classification.

Week 5.

9. Neural Networks: Learning.

9.1 Cost Function.

标记：训练样本 m 个

每个样本包含一组输入 x，一组输出 y.

L 表示神经网络层数，

S_l 表示每层 neuron 个数，S_l 表示 # of units (not contain bias unit) in layer l.

S_L 表示最后一层中处理单元个数

神经网络分为 $\begin{cases} \text{二类分类：} S_L=1,\ y=0 \text{ or } 1 \text{ 表示那一类.} \\ K\text{类分类：} S_L=K,\ y_i=1 \text{ 表示分到第 } i \text{ 类，} (K>2) \end{cases}$

cost function in logistic regression.

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

在 logistic regression 中，只有一个输出变量，只有一个因变量 y.

在 neural network 中，有多个输出，$h_\theta(x)$ 是维度为 K 的向量.

$$h_\theta(x)\in R^K,\quad \left(h_\theta(x)\right)_i = i^{th}\ \text{output}.$$

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

对于每一特征，都会给出 K 个预测.

regularized term 排除了每层的 θ_0，再算 θ_0 经除的和.

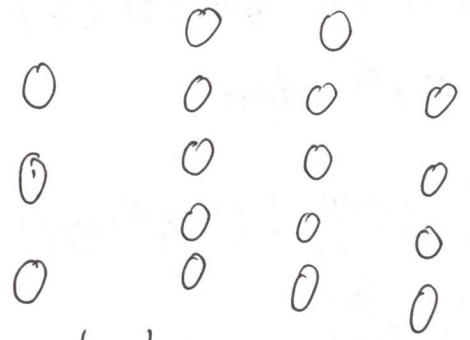
随机化 Θ 时，随机范围：$[-\varepsilon_{init}, \varepsilon_{init}]$，$\varepsilon_{init}=\frac{\sqrt{6}}{\sqrt{L_{in}+L_{out}}}$，$L_{in}=S_l$，$L_{out}=S_{l+1}$

9.2 Back~~war~~ propagation Algorithm

为了计算 $\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta)$，采用反向传播算法。

例子：

已有一个实例 $(x^{(1)}, y^{(1)})$，neural net work 是一个四层，$k=4$，$S_L=4$，$L=4$



Layer 1 layer 2 layer 3 layer 4

$a^{(1)}$ $a^{(2)}$ $a^{(3)}$ $a^{(4)}$

最后一层误差：$\delta^{(4)} = a^{(4)} - y$

$\delta^{(3)} = (\theta^{(3)})^T \delta^{(4)} .* g'(z^{(3)})$ 有没有这一项？？不理解。

其中：$g'(z^{(3)}) = a^{(3)} .* (1 - a^{(3)})$，$a^{(3)} = g(z^{(3)})$

$(\theta^{(3)})^T \delta^{(4)}$ 是权重导致的误差 bias

$\delta^{(2)} = (\theta^{(2)})^T \delta^{(3)} .* g'(z^{(2)})$

第一层是输入，无误差

$$g(z) = \frac{1}{1+e^{-z}}, \quad g'(z) = -\frac{1}{(\ )^2} \cdot (-1) \cdot e^{-z} = \frac{e^{-z}}{(1+e^{-z})^2}$$

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

$$= g(z)(1 - g(z))$$

若 $\lambda = 0$，$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = a_j^{(l)} \delta_i^{(l+1)}$

l：代表当前所在层

j：字母表 activation unit 的下标

i：下一层中误差单元的下标。

有很多 training unit.

用 $\Delta_{ij}^{(l)}$ 来存误差矩阵，

第 l 层的第 i 个激活单元受第 j 个参数影响而导致的误差。

算法为：

for $i = 1 : m$ {

set $a^{(1)} = x^{(i)}$

perform forward propagation to compute $a^{(l)}$ for $l = 1, 2, \dots, L$

Using $\delta^{(L)} = a^{(L)} - y^i$

perform backward prop~~to~~ compute previous layer error vector $\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

求出 $\Delta_{ij}^{(l)}$ 后，$\frac{\partial J}{\partial \Theta}$ will be:

$$P_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} + \lambda \Theta_{ij}^{(l)} \quad \text{if } j \neq 0$$

$$D_{ij}^{(l)} := \frac{1}{m}\Delta_{ij}^{(l)} \quad \text{if } j = 0$$

9.3 Back propagation intuition.

~~看pdf，P146~~

9.4 : Implementation Note _ Unrolling parameters.

~~9.5~~ 在论文中把参数从矩阵展开成向量

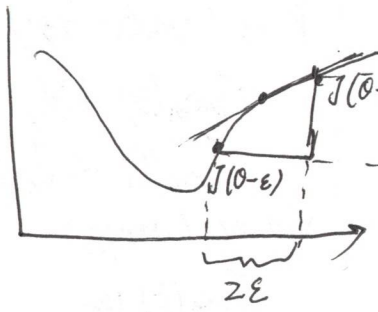
看pdf，P146

9.5 Gradient checking

因为 $J(\Theta)$ 在减小，但结果可能不是最优的

用 ~~Gradient~~ Numerical Gradient checking 方法：

画出估计梯度值来检验。



$J(\Theta+\varepsilon) - J(\Theta-\varepsilon)$

即令 $\varepsilon = 0.001$

$$\frac{d}{d\Theta}J(\Theta) \approx \frac{J(\Theta+\varepsilon) - J(\Theta-\varepsilon)}{2\varepsilon}$$

parameter vector Θ

$\Theta \in R^n$, $\Theta = [\Theta_1, \Theta_2, \cdots, \Theta_n]$

$$\frac{\partial}{\partial \Theta_1} J(\Theta) \approx \frac{J(\Theta_1+\varepsilon, \Theta_2, \cdots \Theta_n) - J(\Theta_1-\varepsilon, \Theta_2, \cdots \Theta_n)}{2\varepsilon}$$

$\frac{\partial}{\partial \Theta_2}$;

$$\frac{\partial}{\partial \Theta_n} J(\Theta) \approx \frac{J(\Theta_1, \Theta_2, \cdots \Theta_n+\varepsilon) - J(\Theta_1, \Theta_2, \cdots \Theta_n-\varepsilon)}{2\varepsilon}$$

9.6 Random Initialization.

~~跟~~ 逻辑回归 Regression 用 0 初始化可以，但 neural network 不能用 0 初始化，否则第二层的所有单元值相同。

故用随机数初始化，between $-\varepsilon \sim \varepsilon$, $-\varepsilon \leq \Theta_{ij}^{(l)} \leq \varepsilon$

9.7 Put it together.

Traing a network:

① Randomly initialize the weights

② Implement forward propa. to get $h_\Theta(x^{(i)})$ for any $x^{(i)}$

③ Implement the cost func.

④ Implement backpropa. to compute partial derivatives

⑤ Use Gradient checking to confirm that your backpropa. works, then disable gradient checking

⑥ Use Gradient descient or built-in optimization func. to

week 6

10 Advice for Applying Machine Learning

10.1 模型误差大，如何改：

获得更多样本，但代价大，不推荐

① 减少特征的数量

② 获得更多的特征

③ 增加多项式特征

④ 减小正则化系数 λ

⑤ ↑

10.2 Evaluating a Hypothesis

分数据为 training set (70%), 和 Testy set (30%).

前面要 shuffle.

对线性回归模型，用 Testy set 算 J.

对 logistic regression，用 Testy set 算 J

还能算：

Misclassification error:

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

$$\text{Test error} = \frac{1}{M_{test}} \sum_{i=1}^{M_{test}} err(h_\theta(x_{test}^{(i)}), y^{(i)})$$

10.3 Model Selection and Train-Validation-Test sets

60% 做 training set, 20 做 validation set, 20% 做 Test set

① 用 training set 得出 10 个 model

② 用 10 个 model 对 validation set 算 CV error, (cost function)

③ 出较 CV error 最小的模型

④ 用 ③ 中选出的 model 对 Test set 算推广误差 (cost function)

Training error:

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Cross validation error:

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

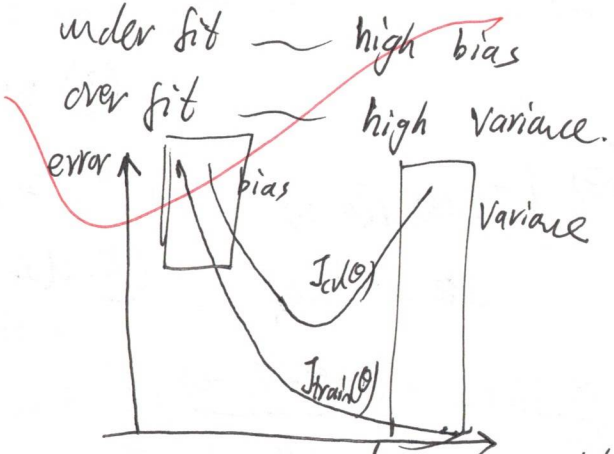
Test error:

$$J_{test}(\theta) = \frac{1}{2M_{test}} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

10.4 Diagosing Bias Vs. Variance.

under fit ~ high bias

over fit ~ high variance.



Bias (under fit)

$J_{train}(\theta)$ high,

$J_{train}(\theta) \approx J_{cv}(\theta)$

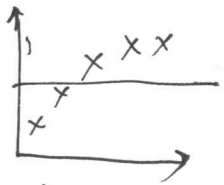
Variance (over fit)

$J_{train}(\theta)$ low

$J_{train}(\theta) \ll J_{cv}(\theta)$

10.5 Regularization and Bias_variance.

需选择合适的 λ.

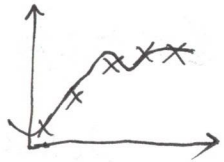


large λ

High bias (underfit)

Intermediate λ

Just right



small λ

High variance (overfit)

① 用不同的 λ 训练出 12 个 model

② 12 个 model 的 $J_{cv}(\theta)$

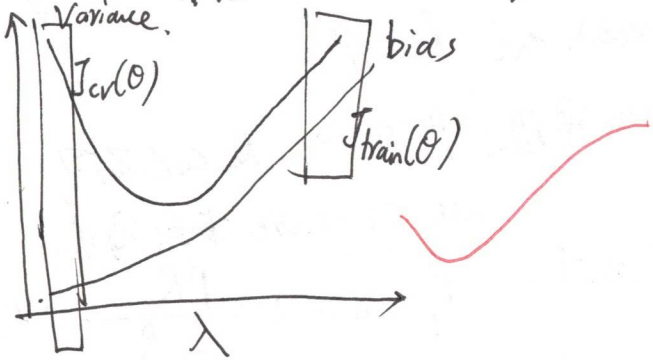
③ 选 $J_{cv}(\theta)$ 最小的 model

④ 用③中的 model 计算 generalized error $J_{test}(\theta)$.

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

$$J_{train}(\theta)=\frac{1}{2M}\sum_{i=1}^{M}(h_\theta(x^{(i)})-y^{(i)})^2$$

$$J_{cv}(\theta)=\frac{1}{2M_{cv}}\sum_{i=1}^{M}(h_\theta(x_{cv}^{(i)})-y^{(i)})^2$$

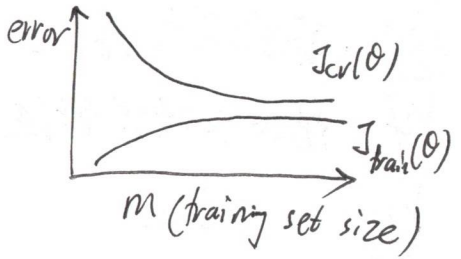


10.6 Learning Curves

learning curve 是很好的合理性检查 (sanity check).

$$J_{train}(\theta)=\frac{1}{2M}\sum_{i=1}^{M}(h_\theta(x^{(i)})-y^{(i)})^2$$

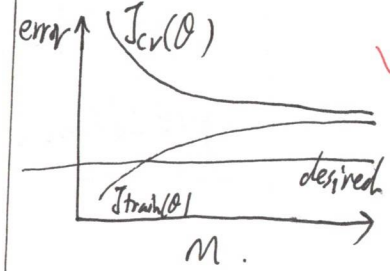
$$J_{cv}(\theta)=\frac{1}{2M_{cv}}\sum_{i=1}^{M}(h_\theta(x_{cv}^{(i)})-y_{cv}^{(i)})^2$$



在高 high bias (underfit) 的情况下, ↑ 训练集 也无帮助.

在 High variance (overfit) 一, ↑ 一 有帮助.

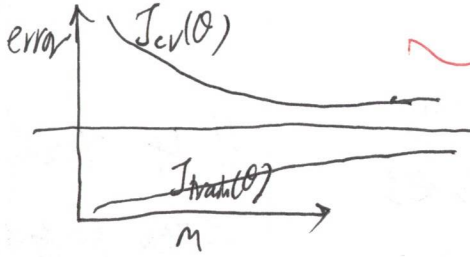
High bias.



· Low train set size: $J_{train}(\theta)$ low, $J_{cv}(\theta)$ high

large 一: $J_{train}(\theta)$, $J_{cv}(\theta)$ both are high

High variance.



low train set size: $J_{train}(\theta)$ low, $J_{cv}(\theta)$ high

large 一: $J_{train}(\theta)$ ↑ as m ↑

$J_{cv}(\theta)$ ↓.

10.7. Deciding what to do next next

① 更多训练实例, ↑m — 解决 high variance
② 减少的特征数量 — variance
③ 获得更多特征 — bias
④ 增加多项式特征 — bias
⑤ ↓λ — bias
⑥ ↑λ — variance.

小的 neural network: high bias, (underfit)
大的 neural network: high variance (overfit).

11. Machine Learning System Design.

11.1 prioritizing what to work on

11.2 Error Analysis.

①先用从简单的能快速实现的算法开始, 实现之并用 CV, ②画 learning curve, 决定是否加特征, 更多数据 或别的什么.

③进行 error Analysis: 检查 CV set 中产生误差的实例, 以改进

11.3 Error Metrics for Skewed classes

skew classes: training set 中有非常多的同一类, 有很少 or 没有其他类的实例

此时误差大小不能用来判断 model 好坏.

		Actual 1	Actual 0
predict	1	True positive (TP)	False positive (FP)
	0	False negative (FN)	True negative (TN)

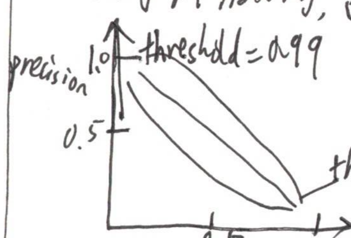
$$accuracy = \frac{TP + TN}{total\ examples}$$

precision: $\frac{TP}{TP+FP}$ of all patients we predict $y=1$, what fraction actually has cancer?

Recall: $\frac{TP}{TP+FN}$ of all patients that actually have cancer, what fraction did we detect as having cancer?

11.4 Trading off Precision and Recall

以预测肿瘤为例, 选不同的阈值, precision, Recall 不同,



use F1-score 来选阈值

F_1 score $= 2 \frac{PR}{[illegible]}$ 选大的

11.5 Data for Machine Learning

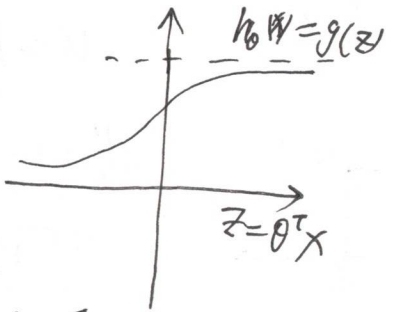
week 7

12. Support Vector Machines.

12.1 Optimization Objective.

Logistic Regression.

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



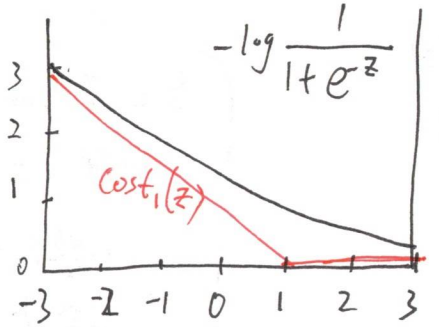
if $y=1$, we want $h_\theta(x) \approx 1$, $\theta^T x \gg 0$

$y=0$ $\quad h_\theta(x) \approx 0$, $\theta^T x \ll 0$

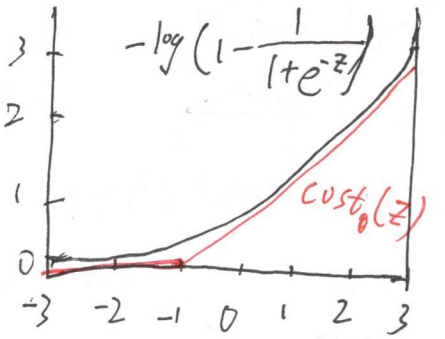
cost: $-(y \log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$

$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1-y)\log\left(1 - \frac{1}{1+e^{-\theta^T x}}\right)$

if $y=1$, $(\theta^T x \gg 0)$



if $y=0$, $(\theta^T x \ll 0)$



用这个代替,

logistic regression:

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

忽略掉常数 m.

support vector machine:

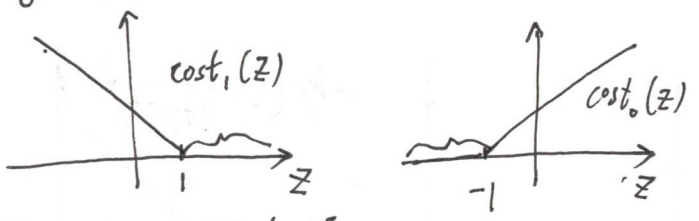
$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

$$h_\theta(x) \begin{cases} 1, & \text{if } \theta^T x \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

12.2 Large Margin Intuition.

$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\text{cost}_1(\theta^Tx^{(i)})+(1-y^{(i)})\text{cost}_0(\theta^Tx^{(i)})\right]+\frac{1}{2}\sum_{j=1}^{n}\theta_j^2$



if $y=1$, we want $\theta^Tx\geq 1$

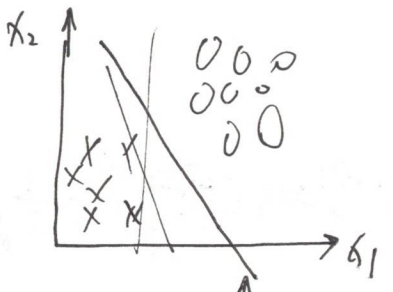
$y=0$, $\theta^Tx\leq -1$

若 C 很大(如 C=10^5), 在 min J(θ) 时, 尝试使第一项为0.

$y=1$ 时, 找一个 θ 使 $\theta^Tx\geq 1$

$y=0$ 时, ——〜 $\theta^Tx\leq -1$

这样会有边界:



这条边界最好, 因为它有最大的距离, 这个距离叫间距(margin)

这个距离叫做 SVM 的间距, 也是 SVM 具有 robust 的原因, 因其用最大间距分离样本.

故 SVM 有时称为 ~~最大间距~~ 大间距分类器 (large margin classifier)

最小化问题变成:

$\min \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$ · $\begin{cases}\theta^Tx^{(i)}\geq 1 & \text{if } y^{(i)}=1\\ \theta^Tx^{(i)}\leq -1 & \text{if } y^{(i)}=0\end{cases}$

当 C 不是非常大时, 它可忽略掉一些异常点, 得到更好的边界

回顾 $C=\frac{1}{\lambda}$,

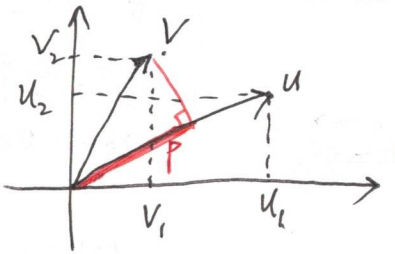
C 较大时, λ 较小, 可能会过拟合, 高方差;

C 较小, λ 较大, 〜 欠拟合, 高 bias.

12.3. Mathematics behind Large margin classifier.

vector inner product

$u=\begin{bmatrix}u_1\\u_2\end{bmatrix}$, $v=\begin{bmatrix}v_1\\v_2\end{bmatrix}$

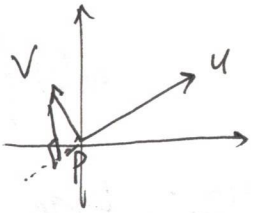


$\|u\|$ = length of vector u

$=\sqrt{u_1^2+u_2^2}$

p = length of projection of v on to u

可正可负



$u^Tv=p\cdot\|u\|$

$=u_1v_1+u_2v_2$

SVM decision bdry.

$\min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$

s.t. $\theta^T x^{(i)} \geq 1$ if $y^{(i)}=1$

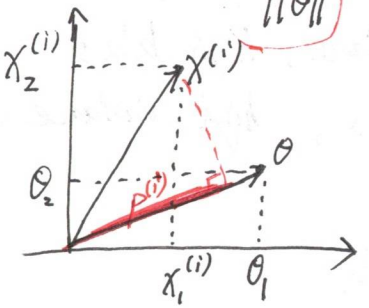
$\theta^T x^{(i)} \leq -1$ if $y^{(i)}=0$

e.g. 让 $\theta_0=0$, $n=2$

$\frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}(\theta_1^2+\theta_2^2) = \frac{1}{2}\left(\sqrt{\theta_1^2+\theta_2^2}\right)^2 = \frac{1}{2}\|\theta\|^2$

$\|\theta\|$

$\theta^T x^{(i)} = ?$



$\theta^T x^{(i)} = p^{(i)} \cdot \|\theta\|$

$= \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)}$

now, SVM decision bdry:

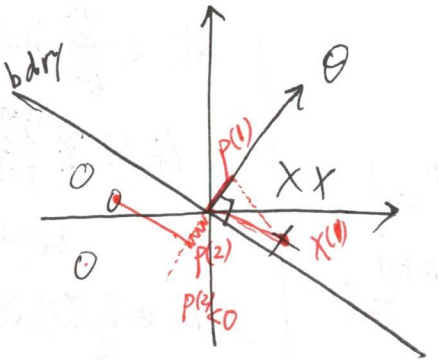
$\min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$

s.t. $p^{(i)} \cdot \|\theta\| \geq 1$ if $y^{(i)}=1$

$p^{(i)} \cdot \|\theta\| \leq -1$ if $y^{(i)}=0$

$p^{(i)}$ is the project of $x^{(i)}$ onto the vector θ.

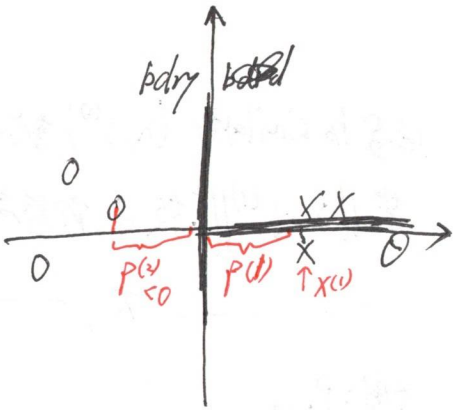
simplification: $\theta_0=0$ → $\theta_0=0$ 意味着 decision bdry 过原点



$p^{(1)} \cdot \|\theta\| \geq 1$, $p^{(1)}$ is small, so $\|\theta\|$ should be large

$p^{(2)} \cdot \|\theta\| \leq -1$, $p^{(2)}$ small, so $\|\theta\|$ large

not good, we want small θ.



$p^{(1)}$, $p^{(2)}$ larger, θ smaller

12.4 kernels 1

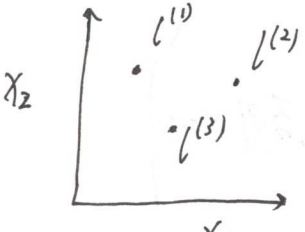
一个给出非线性模型可能是 $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \dots$

f_1 f_2 f_3 f_4 f_5

除了对原始特征组合，有无更好的方法来构造 $f_1, f_2, f_3, \dots$

可用 kernels 计算出新的特征.

给出训练集 实例 x，利用 x 和各个特征与我们预先选定的 landmarks $l^{(1)}$, $l^{(2)}$, $l^{(3)}$ 的近似程度来选取新的特征 f_1, f_2, f_3



$f_1 = \text{similarity}(x, l^{(1)}) = e\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right)$

其中 $\|x-l^{(1)}\|^2 = \sum_{j=1}^{n}(x_j - l_j^{(1)})^2$

为实例 x 计算每个特征

也是为 similarity $(x, l^{(i)})$ 就是 kernels.

若 $\|x-l^{(i)}\|^2$ 较大，即距离较远，则 $e^{-0}=1$

------ 大，~~~ 大，$e^{-(大的数)}=0$

例：P199

12.5 Kernels II.

如何选取地标：

Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots (x^{(m)}, y^{(m)})$

choose $l^{(1)}=x^{(1)}, l^{(2)}=x^{(2)}, \cdots, l^{(m)}=x^{(m)}$

$$f^{(i)}=\begin{vmatrix} f_0^{(i)}=1 \\ f_1^{(i)}=sim(x^{(i)}, l^{(1)}) \\ f_2^{(i)}=sim(x^{(i)}, l^{(2)}) \\ \vdots \\ f_i^{(i)}=sim(x^{(i)}, l^{(i)})=e^0=1 \\ \vdots \\ f_m^{(i)}=sim(x^{(i)}, l^{(m)}) \end{vmatrix}$$

修改 SVM 假设为：

给定 x，计算拟合点 f，当 $\theta^T f \geq 0$ 时，预测 $y=1$，否则 $y=0$

修改 cost func 为 $\sum_{j=1}^{n=m} \theta_j^2 = \theta^T\theta$

$$\min_\theta C\sum_{i=1}^{m}[y^{(i)}\text{cost}_1(\theta^T f^{(i)})+(1-y^{(i)})\text{cost}_0(\theta^T f^{(i)})]+\frac{1}{2}\sum_{j=1}^{n=m}\theta_j^2$$

在计算 $\sum_{j=1}^{n=m}\theta_j^2=\theta^T\theta$ 时，用 $\theta^T M\theta$ 代替 $\theta^T\theta$，M 是根据选择的 kernels 不同而不同的矩阵，为了简化计算.

SVM 也可以不使用 kernel，称为 linear kernel.

$C=\frac{1}{\lambda}$.

C 大时，λ 小，overfity，high variance

C 小时，λ 大，underfity，high bias

σ 大时，low variance，high bias

σ 小时，low bias，high variaace ？理解！

12.6 Using SVM.

Week 8.

13 Clustering

13.1 Unsupervised Learning — Introduction.

13.2 K-Means Algorithm.

假设分成 n 组

① 先 K 个随机点为 cluster centroids.

② 数据集中所有点与最近的中心点关联，聚成一类

③ 计算每一组的均值，将中心移到平均值位置.

用 $\mu^1, \mu^2, \dots \mu^k$ 表示聚类中心

用 $c^{(1)}, c^{(2)}, \dots c^{(m)}$ 存储与第 i 个实例最近的聚类中心的索引

13.3 Optimization Objective.

K-means 最小化：最小化所有点与相聚类中心距离之和最小.

K-means 的 cost func：(Distortion func)

$$J(c^{(1)}, c^{(2)}, \dots c^{(m)}, \mu_1, \dots \mu_k) = \frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - \mu_{c^{(i)}}\|^2$$

目标是找出使 J( ) 最小的 $c^{(1)}, c^{(2)}, \dots c^{(m)}, \mu^1, \mu^2, \dots \mu^k$

13.4 Random Initialization.

K < m，随机选 K 个实例作聚类中心

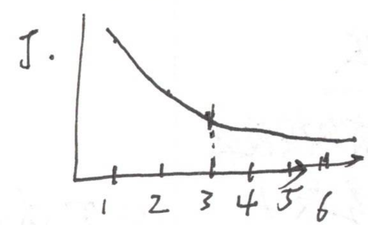
问题：可能停在 local minimum.

解决：多次运行 K-means，每次重新 random initialization.

~~K (or~~ K 小时 (2~10) 可行，大了不适用

13.5 Choosing the # of Clusters.

选择 K，用 elbow method.



参考资料 P213.

14. Dimensionality Reduction.

14.1 Motivation I — Data compression

14.2 Motivation II — Visualization.

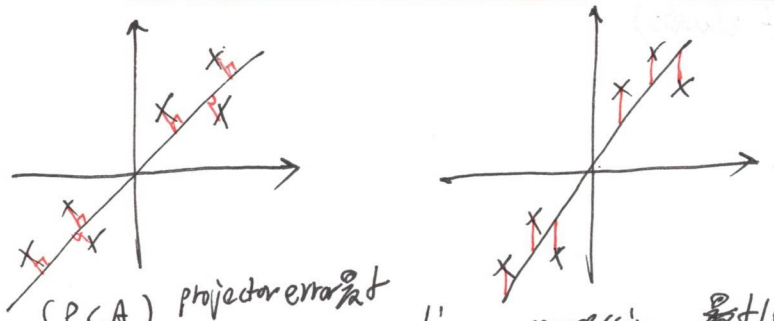
特征很多数据降维才能可视化.

降维之后的特征没有意义由自己定义.

14.3 Principle component analysis problem Formulation. (PCA)

PCA是常见降维算法.

在PCA中，要找到 vector direction，所有数据投射到该 vector direction，使平均均方误差最小.



(PCA), projector error最小

linear regression，最小化预测误差

14.4 Principal Component Analysis Algorithm

PCA 减少 n 维到 k 维：

① 均值归一化 x 算均值 μ_j，$x_j = x_j - \mu_j$，如果数据还要除以 σ^2

② covariance matrix 为：$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T$ ($n\times 1$, $1\times n$)

③ 计算 covariance matrix 的 eigenvectors

可用 SVD，$[U,S,V] = SVD(\Sigma)$ (Σ: $n\times n$ matrix)

$U = \begin{bmatrix} | & | & | & & | \\ u^{(1)} & u^{(2)} & u^{(3)} & \cdots & u^{(n)} \\ | & | & | & & | \end{bmatrix}$

从 $n-D \to k-D$，就只取 U 中的前 k 个向量，用 U_{reduce} 表示，要求的新特征向量 $z^{(i)} = U_{reduce}^T * x^{(i)}$

($z^{(i)}$: $k\times 1$, U_{reduce}^T: $k\times n$, $x^{(i)}$: $n\times 1$)

14.5 Choosing the # of principle components

PCA 是减少投射的平均均方误差 $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2$ (Ave. squared projection error)

训练集的方差 (total variation)：$\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2$

我们希望在平均均方误差与训练集方差的比例尽可能小的情况下选择尽可能小的 k 值.

choose k to be smallest value so that

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

99% of variance is retained

14.6 Reconstruction from compressed Representation.

$z = U_{reduce}^T x$

$x_{approx} = U_{reduce} \cdot z$

$x_{approx} \approx x$

14.5 (continued)

Algorithm:

① Try PCA with $k=1$

② compute $U_{reduce}, z^{(1)}, z^{(2)}, \cdots z^{(m)}$

$x_{approx}^{(1)}, x_{approx}^{(2)}, \cdots x_{approx}^{(m)}$

③ check if

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

④ 若③不满足, try $k=2$, or 3, ... repeat ②~③.

$[U, S, V] = SVD(\Sigma)$

$$S = \begin{bmatrix} S_{11} & & & 0 \\ & S_{22} & & \\ & & S_{33} & \\ 0 & & & \ddots \end{bmatrix}$$

For given k, 算 $1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \leq 0.01$? or $\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$

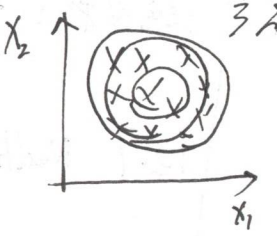
14.7 Advice for Applying PCA.

week 9.

15. Anomaly Detection.

15.1 problem Motivation.

什么是 anomaly detection: 有一组数据 $x^{(1)}, x^{(2)} \sim x^{(m)}$, 某天有一新的 x, 将其数据与已有 m 个数据比较, 是否异常?



若在圈内, 则属于该组可能性较高.

称为密度估计, $p(x) \begin{cases} < \varepsilon, & \text{anomaly} \\ \geq \varepsilon, & \text{normal} \end{cases}$

$p(x)$ 为 x 属于某一组数据的概率

15.2 Gaussian Distribution.

认为 x 服从正态分布, $x \sim N(\mu, \sigma^2)$

probability density func. $p(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

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

15.3 Algorithm.

异常检测算法：

对给定数据集 $x^{(1)}, x^{(2)}, \cdots x^{(m)}$，针对每一个特征计算 μ, σ^2.

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

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

给定新的训练实例，计算 $p(x)$

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

$p(x) < \varepsilon$ 时为异常.

15.4 Developing and Evaluating an Anomaly Detection system

↑ unsupervised

从带标记的(正常，异常)数据着手，选一部分正常数据作 training set，用剩下的正常和异常数据混合构成 cv set，Test set.

① 根据 training set，计算 μ_j, σ_j，构建 $p(x)$ 函数

② 对 cv set，尝试不同 ε 作阈值，根据 F_1 score，precision，Recall 来选择 ε

③ 选出 ε，对 Test set，计算 F_1 score，precision，Recall.

15.5 Anomaly Detection vs. Supervised Learning

见 P_{232}

15.6 Choosing what features to use

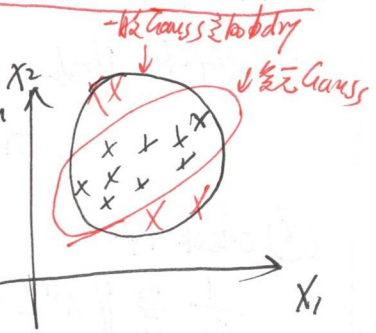
Anomaly detection assume features 符合 Gauss distribution

数据不是 Gauss dist. 最好转成 Gauss dist.

如 $x = \log(x+c)$，$x = x^c$ 等方法.

误差分析：某些异常值可能也有高 $p(x)$ 值，可以创建新特征.

15.7 Multivariate Gaussian Distribution



一般Gauss，还是 $p(x) =$ 计算各特征的 $p(x)$ 并累乘

multivariate Gauss：构建特征的 covariance matrix，用所有的特征一起算 $p(x)$.

所有特征的均值：$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$ → 是向量，每行的均值

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

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

↑ determinant of Σ

15.8 Anomaly Detection using the Multivariate Gaussian dist.

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

Fit model $p(x)$ by setting

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

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

Given a new example x, compute

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

flag an anomaly if $p(x)<\varepsilon$

16. Recommender System

16.1 problem Formulation

一个给电影打分的例子,

n_u = 用户数量

n_m = 电影数量

$r(i,j)$: 若用户 j 给电影 i 打过分, 则 $r(i,j)=1$

$y^{(i,j)}$: 用户 j 给电影 i 打的分

m_j: 用户评过的电影的总数

16.2 Content-based Recommendations

每部电影有特征, x_1 ~浪漫程度, x_2 ~action程度 / 用linear regression

$\theta^{(j)}$: 用户 j 的参数向量

$x^{(i)}$: 电影 i 的特征向量

对用户 j 和电影 i, 预测评分为 $(\theta^{(j)})^T x^{(i)}$

针对用户 j 的 cost function:

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

所有用户:

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

用 Gradient Descent 法,

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

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

16.3 Collaborative Fitering

协同过滤, 若拥有用户的参数, 可学习出电影的特征.

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

collaborative filtering 可用的梯度下降算法，AP算法.

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

求偏导：

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

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

collaborative filtering algorithm

① Initialize $x^{(1)},\sim x^{(n_m)},\theta^{(1)},\sim\theta^{(n_u)}$ to small random values.

② Minimize $J(x^{(1)},\sim x^{(n_m)},\theta^{(1)}\sim\theta^{(n_u)})$ use gradient descent

③ For user θ, movie x, predit a star ratty $\theta^T x$

16.5 Vectorization — Low Rank Matrix Factorization

5 movies, 4 users, 5行 4列.

$$\begin{bmatrix}(\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T x^{(1)} & \cdots & (\theta^{(n_u)})^T x^{(1)}\\ (\theta^{(1)})^T x^{(2)} & (\theta^{(2)})^T x^{(2)} & \cdots & (\theta^{(n_u)})^T x^{(2)}\\ \vdots & \vdots & & \\ (\theta^{(1)})^T x^{(n_m)} & (\theta^{(2)})^T x^{(n_m)} & \cdots & (\theta^{(n_u)})^T x^{(n_m)}\end{bmatrix}$$

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

How to find movie j related to movie i?

small $\|x^{(i)}-x^{(j)}\|\rightarrow$ movie i, j are similar.

16.6 Implementational Detail — Mean Normalization

首先对矩阵 mean normalization 之 ~~然后去~~ 每一用户对每一电影评分减去所有评分的均值.

用为matric 来训练 model,

做预测时，要加@均值，$(\theta^{(i)})^T x^{(i)}+\mu_i$

Week 10

17. Large scale Machine Learning

17.1 Learning with Large Datasets.

17.2 Stochastic Gradient Descent

要大规模训练集，用random stochastic Gradient Descent 代替批量梯度下降法.

in SGD: cost func:

$$cost(\theta,(x^{(i)},y^{(i)}))=\frac{1}{2}(h_\theta(x^{(i)})-y^{(i)})^2$$

SGD: 先对训练集随机"洗牌"再

Repeat (usually anywhere between 1~10)

{ for i=1:m

{ $\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$

(for j=0:n)

}

}

技巧：可能无法到最小点，在其附近徘徊。

17.3 Mini-Batch Gradient Descent

Mini-batch GD 是介于 Batch GD 与 SGD 之间的算法，每计算 b 次训练实例，更新一次 θ。 $b \in [2, 200]$

Repeat {

for i=1:m

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

(for j=0:n)

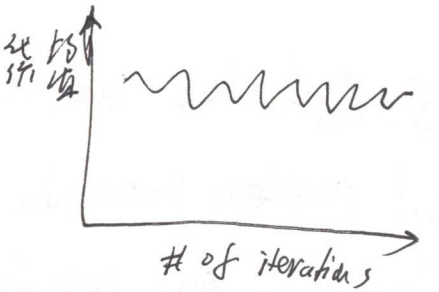
i+=10

}

}

17.4 Stochastic GD convergence

in SGD，每次对代价的值，



真实数不平且不下降：↑α 试之再说。

仍抖动且不下降：model 有问题

or 还在上升，中↓α

也可令α随 iteration ↑而↓

$$\alpha = \frac{\text{const 1}}{\text{\# of iteration} + \text{const 2}}$$

17.5 Online Learning

对网站中一个个用户，给出喜好预测 P(y=1)。

对单一实例进行学习：

Repeat forever (as long as the website is running)

{ get (x,y) corresponding to the current user

$\theta_j := \theta_j - \alpha(h_\theta(x) - y)x_j$

(for j=0:n)

}

17.6 Mapping reduce and Data parallelism

18. Application Example: photo OCR

18.1 problem description and pipeline.

image → Text Detection → character segmentation

→ character recognition.

18.2 Sliding windows. —滑动窗

18.3 Getting lots of data and artificial data.

· 获得更多数据的方法：

① 人工造

② 手动收集，标记数据

③ 众包

8.4 Ceiling Analysis _what part of the pipeline to work on Next

前一步的输入100%正确的话，下一步之后 Accuracy 是

第个，说明~~这~~前一步值得花精力探讨

复习记录：(复习notes, assignment)

2020.8.25 复习Notes.