Machine Learning Notes

#1. 多元线性回归. (以下默认m个样本，n个维度/样本)

Prediction func:

即 $X_{n\times m}$.

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T X \quad (x_0 = 1)$$

cost func: (MSE)

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

对损失函数 θ_j 求偏导:

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

⇒ Gradient descent.

$$\text{repeat}\left\{\theta_j = \theta_j - \alpha\cdot\frac{\partial}{\partial\theta_j}\cdot J(\theta)\right\}$$

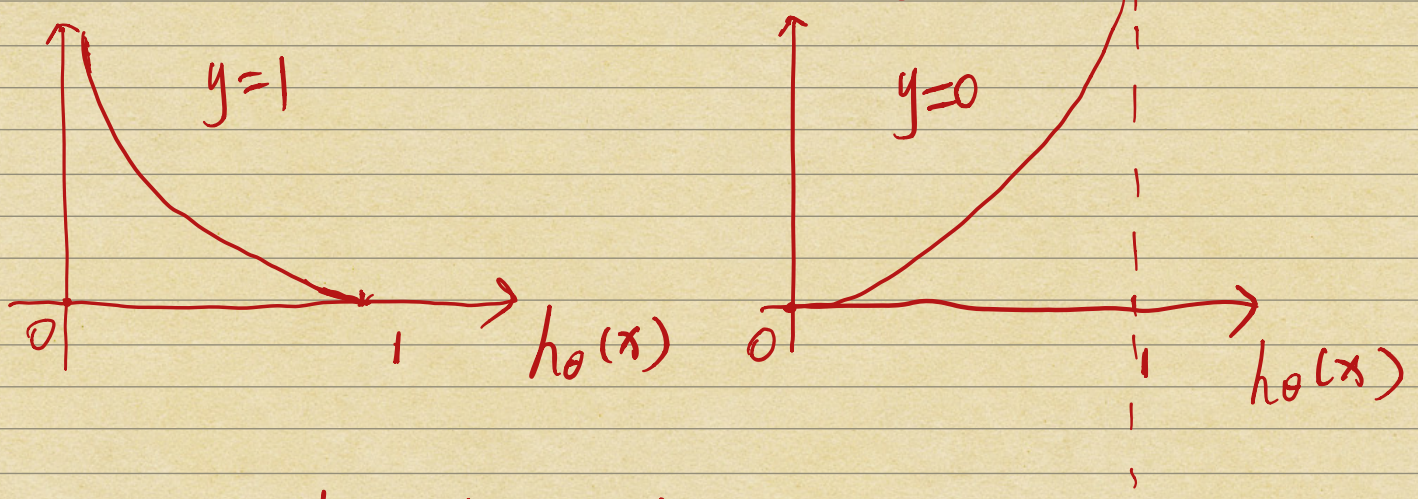
#2. Logistic Regression:

Prediction func: $h_\theta(x) = \dfrac{1}{1+e^{-\theta^T x}}$

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

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

$$cost\left(h_\theta(x^{(i)}), y^{(i)}\right) = \begin{cases} -\log h_\theta(x), & y=1 \\ -\log\left(1-h_\theta(x)\right), & y=0 \end{cases}$$



合并 $cost\left(h_\theta(x^{(i)}), y^{(i)}\right)$

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

To fit parameters θ:

$$\min J(\theta) \xrightarrow{\text{get}} \theta$$

Gradient descent:

$$\text{repeat } \{ \ \theta_j = \theta_j - \alpha \cdot \frac{\partial}{\partial \theta_j} J(\theta) \ \}$$

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

$$
\begin{aligned}
\frac{\delta}{\delta_{\theta_j}} J(\theta) &= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) - (1-y_i) \frac{1}{1-h_\theta(x_i)} \frac{\delta}{\delta_{\theta_j}} h_\theta(x_i) \right) \\
&= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{g(\theta^T x_i)} - (1-y_i) \frac{1}{1-g(\theta^T x_i)} \right) \frac{\delta}{\delta_{\theta_j}} g(\theta^T x_i) \\
&= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i \frac{1}{g(\theta^T x_i)} - (1-y_i) \frac{1}{1-g(\theta^T x_i)} \right) g(\theta^T x_i)(1-g(\theta^T x_i)) \frac{\delta}{\delta_{\theta_j}} \theta^T x_i \\
&= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i (1-g(\theta^T x_i)) - (1-y_i) g(\theta^T x_i) \right) x_i^j \\
&= -\frac{1}{m} \sum_{i=1}^{m} \left(y_i - g(\theta^T x_i) \right) x_i^j \\
&= \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i \right) x_i^j
\end{aligned}
$$

对theta求偏导

此项为sigmoid函数求导公式，推导如下：

$g(x) = \frac{1}{1+e^{-x}}$ ， $f(x) = 1+e^{-x}$

$\Rightarrow f(x) = \frac{1}{g(x)}$ ， $f'(x) = -\frac{g'(x)}{g^2(x)}$ ①

又 $f'(x) = (1+e^{-x})' = -e^{-x} = 1-f(x)$

$= 1 - \frac{1}{g(x)}$ ②

①=②，即 $g'(x) = g(x) \cdot (1-g(x))$

#补充1

正规方程推导：（m个样本，n个维度）→ $X_{m\times n}$

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

对于 $h_\theta(x)$，输入为 $X_{m\times n}$，输出为 $y_{m\times 1}$，权重为 $\theta_{n\times 1}$.

即 $y=X\cdot\theta$.

根据范数定义有：

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

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

令 $\frac{\partial}{\partial\theta_i}J(\theta)=0$ 有：

$$\frac{1}{2m}\left(\frac{\partial\theta^TX^TX\theta}{\partial\theta}-\frac{\partial y^TX\theta}{\partial\theta}-\frac{\partial\theta^TX^Ty}{\partial\theta}+\frac{\partial y^Ty}{\partial\theta}\right)=0$$

① 其中，$\frac{\partial y^Ty}{\partial\theta}=0$，对于 $\frac{\partial\theta^TX^TX\theta}{\partial\theta}=\frac{\partial X^TX\theta^T\theta}{\partial\theta}$ 成立？（θ^T 标量，X^TX 矩阵）

$$X^TX\theta^T\theta=X^TX(\theta_1^2+\theta_2^2+\cdots+\theta_n^2)$$

故 $$\frac{\partial X^TX\theta^T\theta}{\partial\theta}=\begin{pmatrix}\frac{\partial X^TX\theta^T\theta}{\partial\theta_1}\\ \vdots\\ \frac{\partial X^TX\theta^T\theta}{\partial\theta_n}\end{pmatrix}_{n\times 1}=2X^TX\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_n\end{pmatrix}=2X^TX\theta$$

② 对于 $y^T X\theta = (y_1, y_2, \cdots, y_m)\begin{pmatrix} x_1^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(m)} & \cdots & x_n^{(m)} \end{pmatrix}\begin{pmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{pmatrix}$

$= (x_1^{(1)} y_1 + x_1^{(2)} y_2 + \cdots + x_1^{(m)} y_m)\theta_1 + (x_2^{(1)} y_1 + \cdots + x_2^{(m)} y_m)\theta_2$

$+ \cdots + (x_n^{(1)} y_1 + \cdots + x_n^{(m)} y_m)\theta_n$

故 $\dfrac{\partial y^T X\theta}{\partial \theta} = \begin{pmatrix} \dfrac{\partial y^T X\theta}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial y^T X\theta}{\partial \theta_n} \end{pmatrix} = X^T y$.

③ 对于 $\theta^T X^T y = (\theta_1, \theta_2, \cdots, \theta_n)\begin{pmatrix} x_1^{(1)} & \cdots & x_1^{(m)} \\ \vdots & \ddots & \vdots \\ x_n^{(1)} & \cdots & x_n^{(m)} \end{pmatrix}\cdot\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$

$= (x_1^{(1)}\theta_1 + x_2^{(1)}\theta_2 + \cdots + x_n^{(1)}\theta_n) y_1 + (x_1^{(2)}\theta_1 + \cdots + x_n^{(2)}\theta_n) y_2$

$+ \cdots + (x_1^{(m)}\theta_1 + \cdots + x_n^{(m)}) y_m$

故 $\dfrac{\partial \theta^T X^T y}{\partial \theta} = \begin{pmatrix} \dfrac{\partial \theta^T X^T y}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial \theta^T X^T y}{\partial \theta_n} \end{pmatrix} = X^T y$.

从而 $\frac{\partial}{\partial\theta}J(\theta)=0$ 即 $\frac{1}{2m}\left(2X^TX\theta-2X^Ty\right)=0$

$\Rightarrow X^TX\theta=X^Ty$，即 $\theta=(X^TX)^{-1}X^Ty$.

梯度下降与正规方程的比较：

梯度下降	正规方程
需要选择学习率 α	不需要
需要多次迭代	一次运算得出
当特征数量 n 大时也能较好适用	需要计算$(X^TX)^{-1}$ 如果特征数量 n 较大则运算代价大，因为矩阵逆的计算时间复杂度为 $O(n^3)$，通常来说当 n 小于 10000 时还是可以接受的
适用于各种类型的模型	只适用于线性模型，不适合逻辑回归模型等其他模型

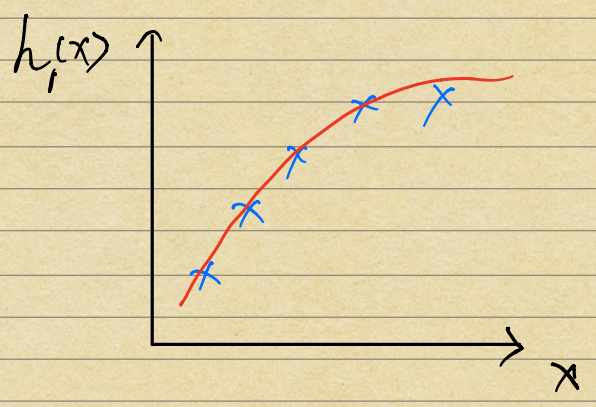
总结一下，只要特征变量的数目并不大，标准方程是一个很好的计算参数 θ 的替代方法。具体地说，只要特征变量数量小于一万，我通常使用标准方程法，而不使用梯度下降法。

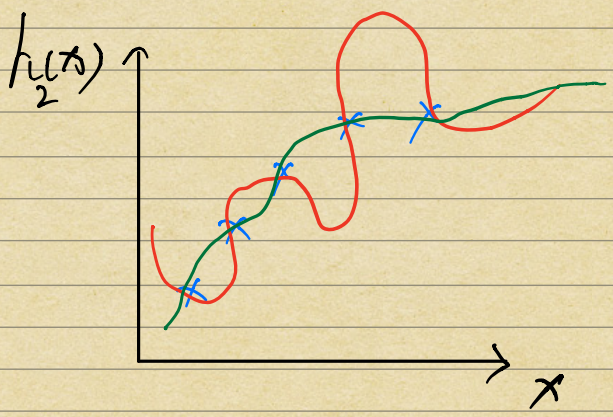
#3. 正则化

1). 对于多元线性回归

Intuition (直觉)

Over-fitting





$h_1(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

$h_2(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Penalize and make θ_3、θ_4 very small.

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

将 θ_3、θ_4 带上一个较大的系数加至 min 函数，

为了得到最小值，$\theta_3 \approx 0$，$\theta_4 \approx 0$，得到

一个新的函数，绿色曲线以降低过拟合。

cost function:

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

对 θ_j 求偏导:

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

Gradient descent {

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

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

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

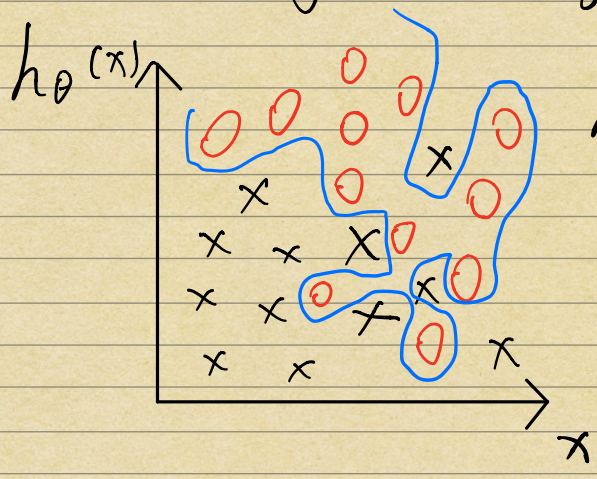
$(j = 1, 2, 3, \cdots, n)$

}

α 很小, m 很大, $\left(1-\alpha\cdot\frac{\lambda}{m}\right) \sim 0.99 < 1$

和正则化前保持一致

2). 对于 Logistic Regression



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \cdots)$$

思路和多元线性回归正则化相同.

cost function:

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

权重衰减 $\theta^T\theta$

对 θ_j 求偏导:

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

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

Gradient descent {

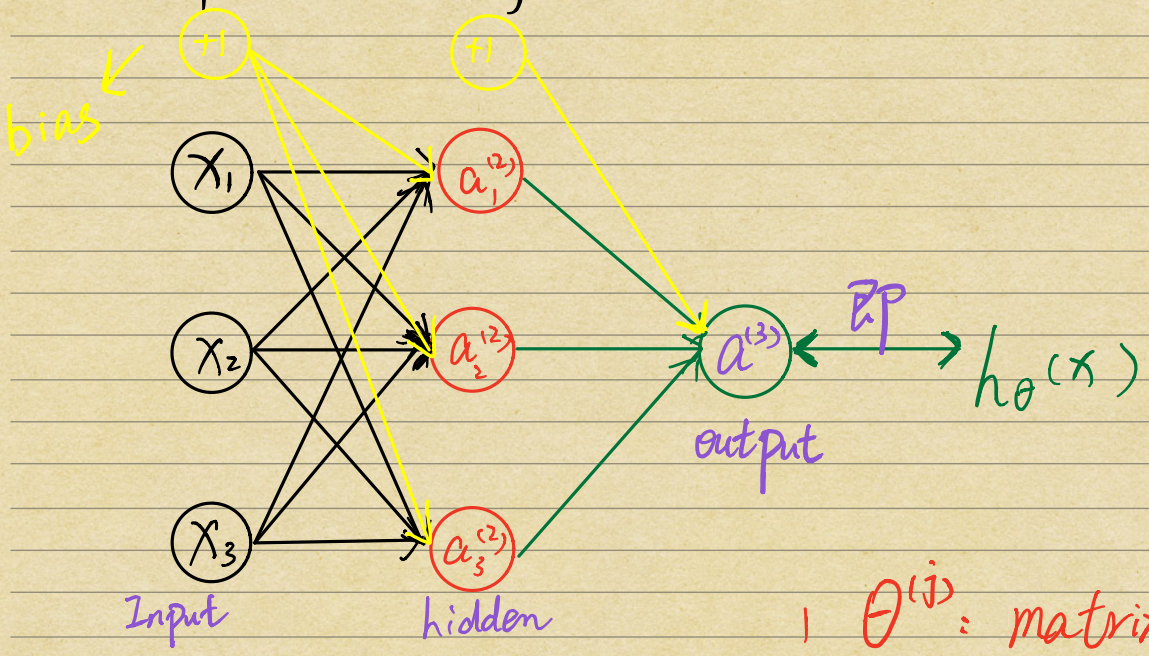
$$\theta_0 = \theta_0 - \alpha \cdot \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)}$$

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

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

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

#4. 神经网络前向传播



$a_i^{(j)}$: "activation" of unit i in layer j.

$\Theta^{(j)}$: matrix of weights controlling function mapping from layer j to layer j+1.

$$\begin{cases} a_1^{(2)} = g(\theta_{10}^{(1)}x_0 + \theta_{11}^{(1)}x_1 + \theta_{12}^{(1)}x_2 + \theta_{13}^{(1)}x_3) = z_1^{(2)} \\ a_2^{(2)} = g(\theta_{20}^{(1)}x_0 + \theta_{21}^{(1)}x_1 + \theta_{22}^{(1)}x_2 + \theta_{23}^{(1)}x_3) = z_2^{(2)} \\ a_3^{(2)} = g(\theta_{30}^{(1)}x_0 + \theta_{31}^{(1)}x_1 + \theta_{32}^{(1)}x_2 + \theta_{33}^{(1)}x_3) = z_3^{(2)} \end{cases}$$

注：x_0、a_0 均为 bias 项，默认为 1.

$$\Rightarrow h_\theta(x) = g(\theta_{10}^{(2)}a_0^{(2)} + \theta_{11}^{(2)}a_1^{(2)} + \theta_{12}^{(2)}a_2^{(2)} + \theta_{13}^{(2)}a_3^{(2)})$$

> 向量化 $h_\theta(x)$.

$$X = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad Z = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}$$

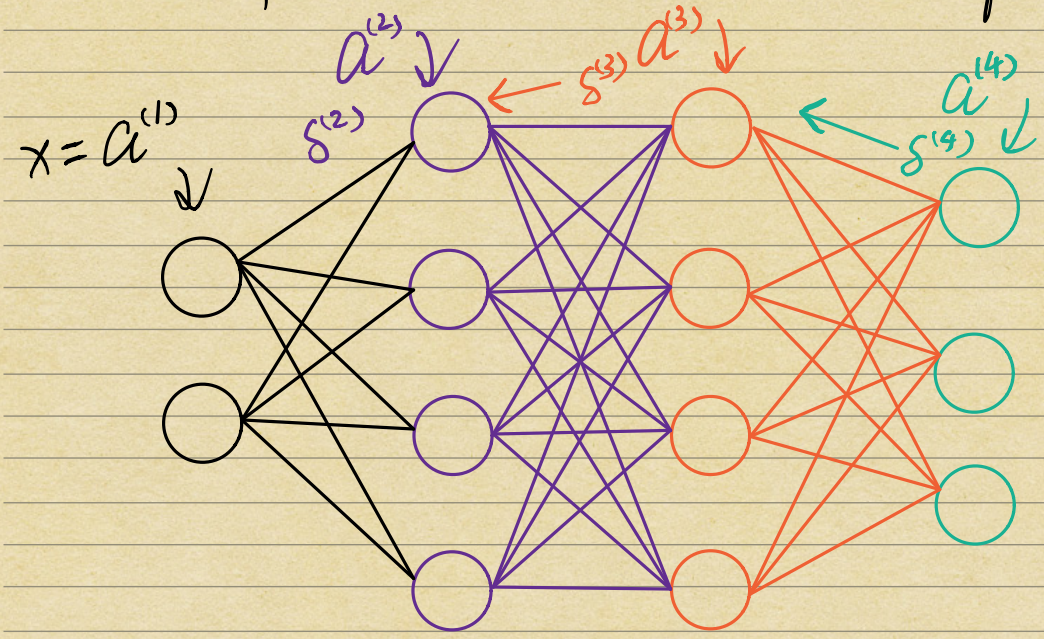
即 $a^{(1)}$

$$z^{(2)} = \theta^{(1)} \cdot x, \quad a^{(2)} = g(z^{(2)})$$

故 $z^{(3)} = \theta^{(2)} \cdot a^{(2)}$

从而 $h_\theta(x) = a^{(3)} = g(z^{(3)})$

#5. 神经网络反向传播 (Back propogation)



K classes
⇕
K output units
⇕
$y = h_\Theta(x) \in R^K$

L: total no. of layers in neural network.

S_l: no. of units (except bias unit) in layer L.

cost function:

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

To $\min J(\Theta)$ → compute $J(\Theta)$, $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta)$

Gradient compute: forward propogation

$a^{(1)} = x$, $z^{(2)} = \Theta^{(1)} a^{(1)}$ $z^{(3)} = \Theta^{(2)} a^{(2)}$ $z^{(4)} = \Theta^{(3)} a^{(3)}$

$a^{(2)} = g(z^{(2)})$, $a^{(3)} = g(z^{(3)})$, $a^{(4)} = g(z^{(4)})$

Gradient compute : back propogation

$$\delta^{(4)} = a^{(4)} - y,$$

$$\delta^{(3)} = \Theta^{(3)T}\delta^{(4)} \cdot g'(z^{(3)}) \quad \rightarrow \quad g'(z^{(l)}) = a^{(l)} \cdot (1 - a^{(l)})$$

$$\delta^{(2)} = \Theta^{(2)T}\delta^{(3)} \cdot g'(z^{(2)})$$

$$\Rightarrow \frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = a_j^{(l)} \delta_i^{(l+1)}, \quad \text{if } \lambda = 0.$$

Backprop Algorithm :

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

Set $\Delta_{ij}^{(l)} = 0$ (for all i, j, l)

For $i = 1$ to m :

Set $a^{(1)} = x$

Perform forwardprop to compute $a^{(l)}$, $l = 2, 3, \cdots, L$

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

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \cdots, \delta^{(2)}$

$$\Delta_{ij}^{(l)} = \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$$

$$\Rightarrow D_{ij}^{(l)} = \begin{cases} \frac{1}{m}\Delta_{ij}^{(l)} + \lambda \Theta_{ij}^{(l)}, & j \neq 0 \\ \frac{1}{m}\Delta_{ij}^{(l)}, & j = 0 \end{cases}$$

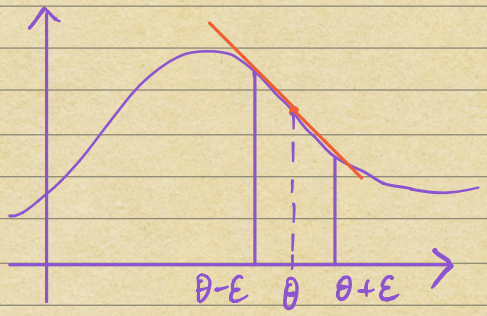
即 $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)}$

Additionally,

e.g. $\varepsilon = 10^{-4}$

numerical estimate

① Gradient check



构造 $\frac{J(\theta+\varepsilon) - J(\theta-\varepsilon)}{2\varepsilon} = gradApprox$

验证 $\frac{\partial}{\partial \theta} J(\theta) \approx gradApprox$.

② Random initialization

若 $\Theta_{ij}^{(l)} = 0$ (for all i,j,l)

则每层的unit中激活 a 相同，偏差 δ 相同，偏导 $\frac{\partial}{\partial \Theta} J(\Theta)$ 相同.

故一般令 $\Theta_{ij}^{(l)}$ 初始化在 $[-\varepsilon, \varepsilon]$，打破这种对称性.

训练神经网络的一般步骤：

① Randomly initialize weights

② Implement forwardprop to compute $h_\Theta(x^{(i)})$ for any $x^{(i)}$.

③ Implement code to compute $J(\Theta)$

④ Implement backprop to compute partial derivatives $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta)$

⑤ Use gradient checking $\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta)$ v.s. numerical estimate, then disable checking code.

⑥ Use gradient descent or advanced method to min $J(\theta)$.

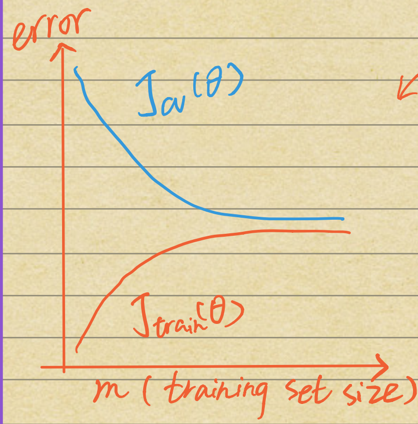
#补充2 (应用机器学习的建议)

①

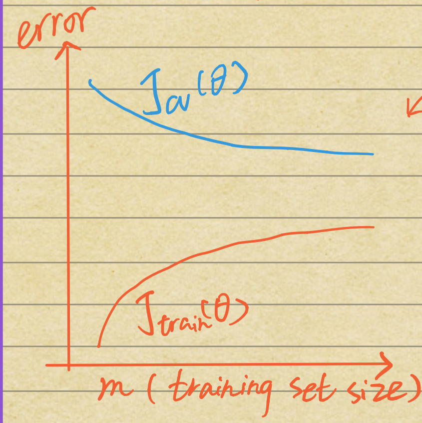
J(θ) 关于正则参数的走势图

如正则项 $\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$

J(θ) 关于样本m的学习曲线



↙ high bias: 增大样本手收益无益



↙ high variance. 增大样本可一定程度提升性能.

采取措施改善model前，先衡量偏差or方差.

a. Getting more training examples → fix high variance

b. try smaller sets of features → fix high variance

c. try additional features → fix high bias

d. try add polynomial features (x_1^2, x_2^2, x_1x_2, etc) → fix high bias

e. try decreasing λ → fix high bias

f. try increasing λ → fix high variance

②

Error metrics for skewed classes

Cancer classification example:

Training a logistic regression model ($y = \begin{cases} 1, \text{if cancer} \\ 0, \text{otherwise} \end{cases}$), find that you got 1% error on test set.

And only 0.5% patients have cancer, if you ignore x, let y always equals 0, this algorithm gets only 0.5% error rate. —lower error→ But not a good algorithm.

So, to measure a algorithm's reliability, let's introduce precision and recall:

cancer = 1, otherwise = 0.

Actual / Predict	1	0
1	True Positive	False Positive
0	False Negative	True Negative

(查准率)

$$\text{Precision} = \frac{TP}{TP + FP}$$

预测正确患癌 / 全部预测患癌

$$\text{Recall} = \frac{TP}{TP + FN}$$

(召回率)

预测正确患癌 / 全部实际患癌

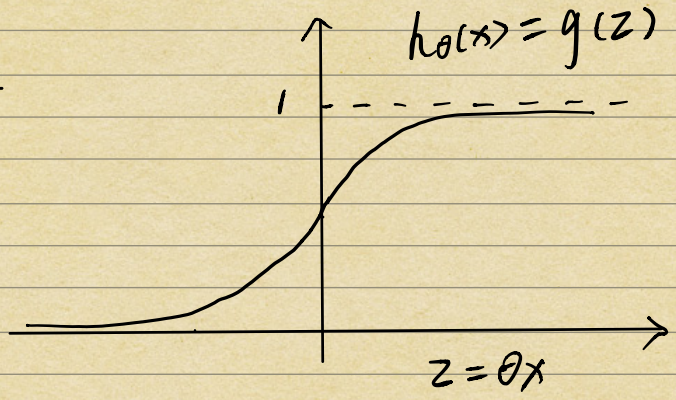
Conclusion: 对于偏斜类问题可观测查准率及召回率.

#6. 支持向量机 (SVM)

① Optimization objective

Alternative view of logistic regression.

$$\begin{cases} h_\theta(x) = \dfrac{1}{1+e^{-\theta x}} \\ g(z) = \dfrac{1}{1+e^{-z}} \end{cases}$$



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

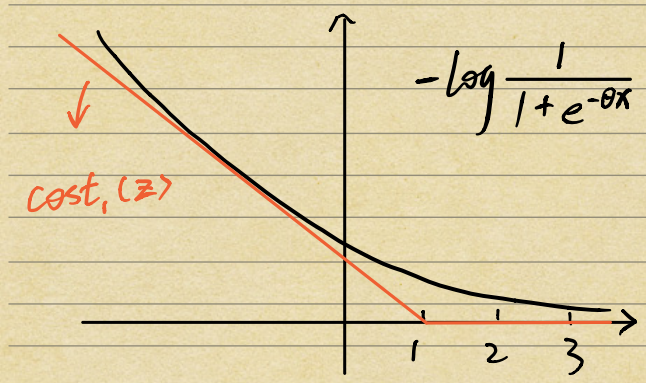
if $y=0$, want $h_\theta(x) \approx 0$, need $\theta x \ll 0$

Considering one sample's $J(\theta)$ of logistic regression.

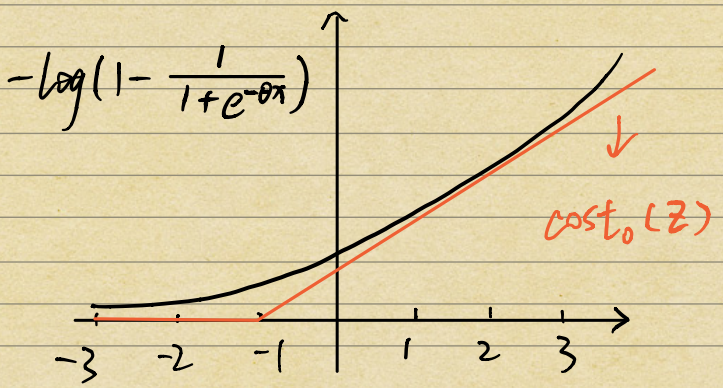
$$J(\theta) = -y \log \frac{1}{1+e^{-\theta x}} - (1-y) \log\left(1 - \frac{1}{1+e^{-\theta x}}\right)$$

$h_\theta(x) \approx$ $y=1$ (want $\theta x \gg 0$)

$h_\theta(x) =$ $y=0$ (want $\theta x \ll 0$)



以 $\text{cost}_1(z)$ 替换



以 $\text{cost}_0(z)$ 替换

② SVM — 大间距分类的直观

在逻辑回归中，我们想要

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

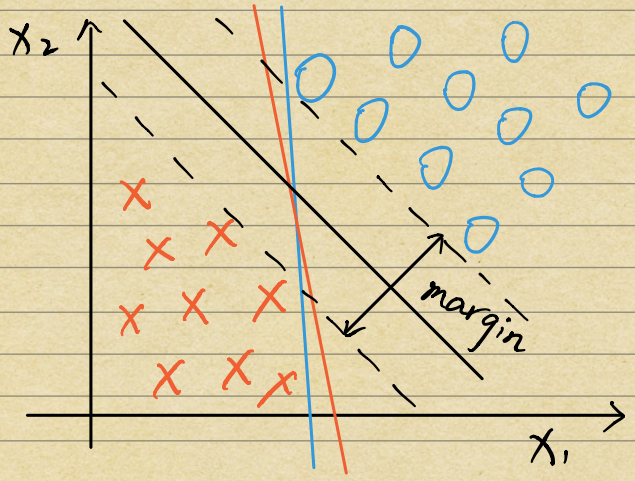
（两个 $-\log$ 项：替换项）

拿掉 $1/m$，对求最优解无影响。同时，拿掉正则项参数 λ，以常数 C 来惩罚经验误差项，最后以 $cost_1(z)$、$cost_0(z)$ 替换，即：

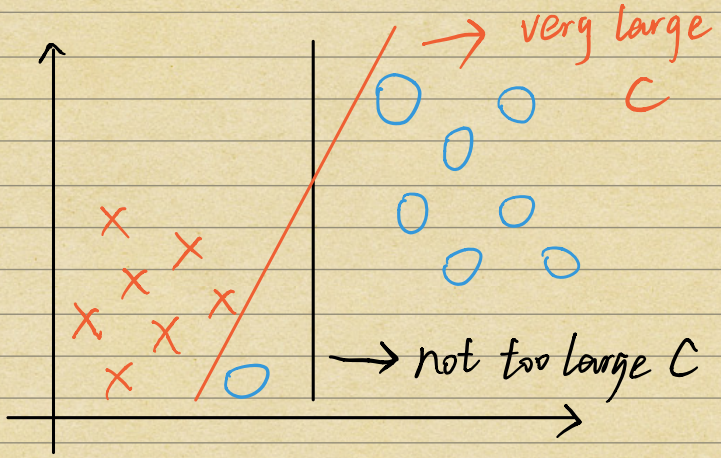
$$\min_\theta C\cdot\sum_{i=1}^{m}\left[y^{(i)}\cdot cost_1(\theta^T x^{(i)})+(1-y^{(i)})\cdot cost_0(\theta^T x^{(i)})\right]+\frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

构造上式后，有 $y^{(i)}=\begin{cases}1, & \theta^T x\geq 1\\ 0, & \theta^T x\leq -1\end{cases}$　相当于构造了一个安全因子

SVM Decision Boundary: Linearly separable case.



SVM会找到一个大间距的决策边界.



通过调节C的值，SVM决策边界可忽略一些异常点.

把C看成 $\frac{1}{\lambda}$ 的话，C很大时 $\rightarrow\lambda$ 很小，则趋向于画出过拟合的线，C不是很大时 $\rightarrow\lambda$ 有值，即加入了正则项，泛化能力增强，决策边界不会因为一两个异常值改变。

③ 大间距分类背后的数学原理

SVM损失函数如下：

$$J(\theta) = C\cdot\sum_{i=1}^{m}\left[y^{(i)}\cdot cost_1(\theta^T x^{(i)}) + (1-y^{(i)})\cdot cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

先考虑C很大的情况，$\min_\theta J(\theta)$会使得$\theta^T x^{(i)} \geq 1$或$\theta^T x^{(i)} \leq -1$，即经验误差项为0，从而 $\min_\theta J(\theta) = \min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$

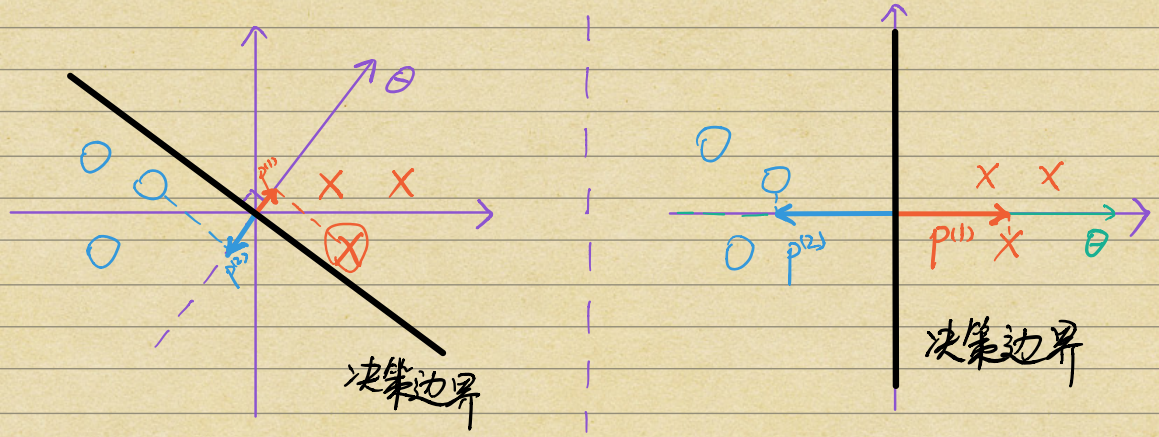
又 $\min_\theta \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \min_\theta \frac{1}{2}\left(\sqrt{\theta_1^2+\theta_2^2+\cdots+\theta_j^2}\right)^2 = \min_\theta \frac{1}{2}\|\theta\|^2$

$\because \theta^T x^{(i)} = p^{(i)}\|\theta\|$，其中$p^{(i)}$为$x^{(i)}$在$\theta^T$上的投影.

于是有 $\begin{cases} p^{(i)}\|\theta\| \geq 1, & \text{if } y^{(i)}=1 \\ p^{(i)}\|\theta\| \leq -1, & \text{if } y^{(i)}=0 \end{cases}$

其中$\vec{\theta}$一定垂直于决策边界，只有如此，才能保证$y=1$时，$p^{(i)}\|\theta\|$取正号，$y=0$时，取负号.

考虑$\theta_0 = 0$，即决策边界过原点.



要$p^{(1)}\|\theta\| \geq 1$，因$p^{(1)}$太小，$\|\theta\|$取大值

要$p^{(2)}\|\theta\| \leq -1$，因$p^{(2)}$太小，$\|\theta\|$取大值

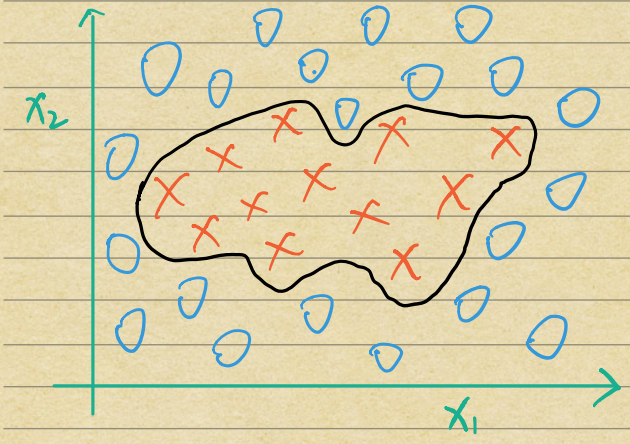
要$p^{(1)}\|\theta\| \geq 1$，因$p^{(1)}$较大，$\|\theta\|$取小值

要$p^{(2)}\|\theta\| \leq -1$，因$p^{(2)}$较大，$\|\theta\|$取小值

因为优化目标为$\min_\theta \frac{1}{2}\|\theta\|^2$，显然，SVM会选择右边大间距的决策边界.

④ 核函数

SVM决策边界：非线性情况



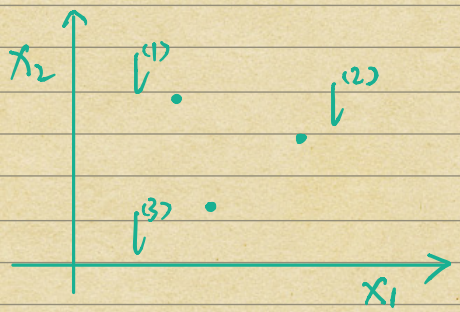
Predict $y=1$, if

$$\theta_0+\theta_1x_1+\theta_2x_2+\theta_3x_1x_2+\theta_4x_1^2+\theta_5x_2^2+\cdots+\geq 0.$$

记 $f_1=x_1$, $f_2=x_2$, $f_3=x_1x_2$, $f_4=x_1^2$, $f_5=x_2^2$, $\cdots$, 则有：

$$\theta_0+\theta_1f_1+\theta_2f_2+\theta_3f_3+\cdots\geq 0, \text{ if } y=1.$$

那么，是否存在一组更好的特征呢？（对比 $f_1, f_2, f_3, \cdots$）



$$\begin{cases} f_1=\text{similarity}(x, l^{(1)}) \\ f_2=\text{similarity}(x, l^{(2)}) \\ f_3=\text{similarity}(x, l^{(3)}) \end{cases}$$

也记作 $\text{kernel}(x, l^{(i)})$

Given x, compute new features depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$.

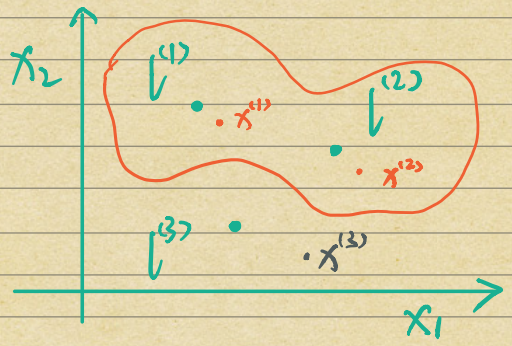
e.g. $f_1=\text{similarity}(x, l^{(1)})=\exp\left(-\frac{\|x-l^{(1)}\|^2}{2\sigma^2}\right)$ Gaussian Kernel

if $x\approx l^{(1)}$, $f_1=\exp\left(-\frac{0}{2\sigma^2}\right)\approx 1$

if x is far from $l^{(1)}$, $f_1=\exp\left(-\frac{(\text{large num})^2}{2\sigma^2}\right)\approx 0$

Predict "1" when

$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$



若 $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$，则对 $x^{(1)}$ 有

$-0.5 + 1\times 1 + 1\times 0 + 0\times 0 = 0.5 \geq 0 \Rightarrow \hat{y} = 1$.

而对 $x^{(2)}$，$\hat{y} = 1$，$x^{(3)}$，$\hat{y} = 0$.

由上，可大致画出这组 θ 和 l 下的决策边界.（红线框）

SVM 决策边界：如何选择 landmarks

Given $(x^{(1)}, y^{(1)})$、$(x^{(2)}, y^{(2)})$、…、$(x^{(m)}, y^{(m)})$

choose $l^{(1)} = x^{(1)}$、$l^{(2)} = x^{(2)}$、… $l^{(m)} = x^{(m)}$.

For training example $(x^{(i)}, y^{(i)})$:

$x^{(i)} \rightarrow$

$f_1^{(i)} = \text{similarity}(x^{(i)}, l^{(1)})$

$f_2^{(i)} = \text{similarity}(x^{(i)}, l^{(2)})$

⋮

$f_i^{(i)} = \text{similarity}(x^{(i)}, l^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1$

⋮

$f_m^{(i)} = \text{similarity}(x^{(i)}, l^{(m)})$

$\forall x^{(i)} \in R^{m+1}$，有 $f^{(i)} \in R^{m+1}$，其中 $x_0^{(i)} = 1$，$f_0^{(i)} = 1$.

$f^{(i)} = (f_0^{(i)}, f_1^{(i)}, f_2^{(i)}, \cdots, f_m^{(i)})^T$.

综之：Given x, compute features $f \in R^{m+1}$，$\begin{cases} \text{if } \theta^T f \geq 0 \\ \hat{y} = 1 \end{cases}$.

Training:

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

→ $n = m$

SVM parameters:

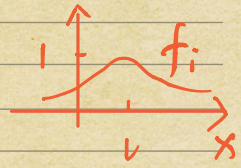
$C (= \frac{1}{\lambda})$, large C: lower bias, higher variance

small C: higher bias, lower variance

σ^2,

large σ^2: 相当于缩小了分子变化率,

f_i更平滑. 高偏差, 低方差.



$f_i = \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right)$

small σ^2: 相当于放大了分子变化率,

f_i更陡峭. 低偏差, 高方差.