Week 7

- ## Large Margin Classification:

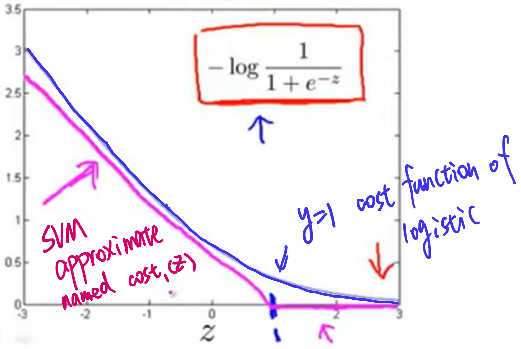
a. Optimization Objective:

Alternative view of logistic regression

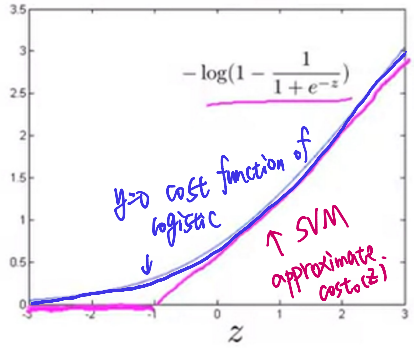
Cost of example: $-(y \log h_\theta(x) + (1-y)\log(1-h_\theta(x)))$

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

If $y = 1$ (want $\theta^T x \gg 0$):



If $y = 0$ (want $\theta^T x \ll 0$):



Support vector machine

Logistic regression:

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

$\text{cost}_1(\theta^T x^{(i)})$ $\quad$ $\text{cost}_0(\theta^T x^{(i)})$

Support vector machine:

$$\min_\theta \sum_{i=1}^{m} y^{(i)}\,\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\,\text{cost}_0(\theta^T x^{(i)}) + \frac{\lambda}{2}\sum_{i=0}^{n}\theta_j^2$$

(A) (B)

$\min_u (u-5)^2 + 1 \rightarrow u = 5$

$\min_u 10(u-5)^2 + 10 \rightarrow u = 5$

$A + \lambda B \rightarrow C A + B$

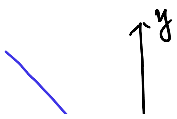
λ and C just decide how much we care about previous term and last term.

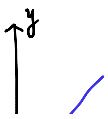
SVM hypothesis:

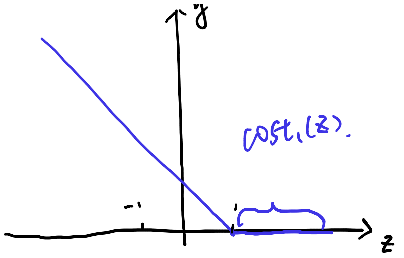
$$\min_\theta C\sum_{i=1}^{m}\left[y^{(i)}\text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)})\text{cost}_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

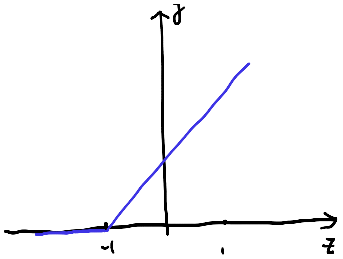
$$h_\theta(x) = \begin{cases} 1, & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

b. Large Margin Intuition:









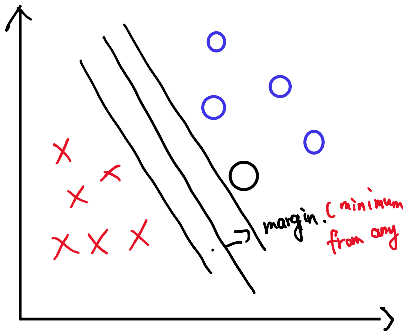
if $y=1$, we want $\theta^T x \geq 1$ if $y=0$, we want $\theta^T x \leq -1$.

suppose we set C as a very large value.

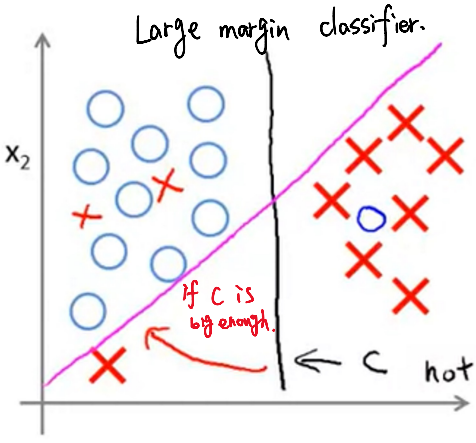
$$\min_\theta C \sum_{i=1}^{m} [y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1-y^{(i)}) \text{cost}_0(\theta^T x^{(i)})] + \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$$

$\Rightarrow \min C \cdot 0 + \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$

$$\text{s.t} \begin{cases} \theta^T x^{(i)} \geq 1 & \text{if } y^{(i)}=1 \\ \theta^T x^{(i)} \leq -1 & \text{if } y^{(i)}=0. \end{cases}$$



Large margin classifier.

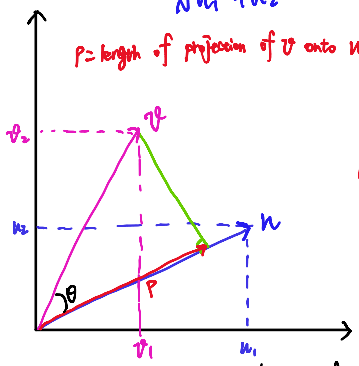


→ C very large

$\frac{1}{\lambda}$

c. Mathematics Behind Large Margin Classification:

$\|u\|$ = length of vector $= \sqrt{u_1^2 + u_2^2} \in \mathbb{R}$ $\quad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ $\quad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$



$u^T v = p \cdot \|u\| = u_1 v_1 + u_2 v_2$.

$p = \cos\theta \|v\|$.

SVM Decision Boundry:

$$\min \frac{1}{2}\sum_{j=1}^{n} \theta_j^2 \quad \text{(simplication: } \theta_0 = 0\text{)}$$

$$= \frac{1}{2}(\theta_1^2 + \theta_2^2) = \frac{1}{2}\left(\sqrt{\theta_1^2+\theta_2^2}\right)^2 = \|\theta\|.$$

$$\text{s.t.} \begin{cases} \theta^T x \geq 1 & \text{if } y^{(i)}=1 \\ \theta^T x \leq -1 & \text{if } y^{(i)}=0. \end{cases}$$

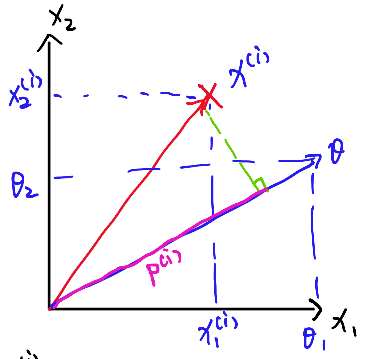
↓ what does it do?

$= \frac{1}{2}\|\theta\|^2$.

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

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

so. $\min \frac{1}{2}\sum_{j=1}^{n} \theta_j^2 = \frac{1}{2}\|\theta\|^2$.

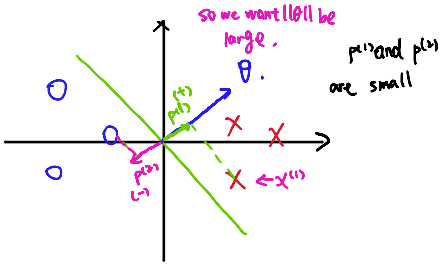


$$\text{s.t.} \begin{cases} p^{(i)}\|\theta\| \geq 1 & \text{if } y^{(i)} = 1 \\ p^{(i)}\|\theta\| \leq -1 & \text{if } y^{(i)} = 0 \end{cases}$$

① choice I

$p^{(i)}\|\theta\| \geq 1$ and $p^{(i)}$ is small

so we want $\|\theta\|$ be large.

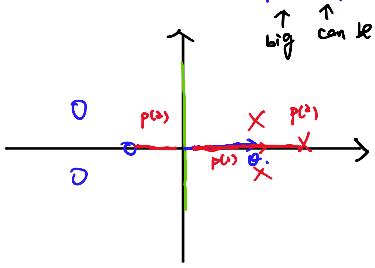


$p^{(1)}$ and $p^{(2)}$ are small

② choice II

$p^{(1)}\|\theta\| \geq 1$

big ↑ ↑ can be small.



- **Kernels:**

  a. Kernels I:

predict $y=1$ if $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$

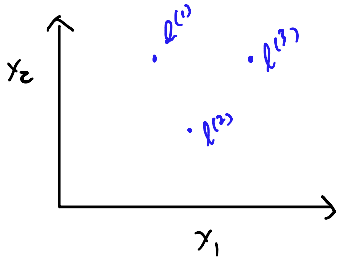
$$h_\theta(x) = \begin{cases} 1 & \theta_0 + \theta_1 + \cdots \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

another form. $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_4 + \cdots$

$f_1 = x_1$, $f_2 = x_2$, $f_3 = x_1 x_2$, $f_4 = x_1^2$, $f_5 = x_2^2$, $\cdots$.

Is there a different / better choice of the features $f_1, f_2, \cdots$?

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



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

$f_2 = \text{similarity}(x, l^{(2)}) = \exp\left(-\frac{\|x - l^{(2)}\|}{2\sigma^2}\right)$

$f_3 = \text{similarity}(x, l^{(3)}) = \exp\left(-\frac{\|x - l^{(3)}\|}{2\sigma^2}\right)$.

↑ kernel function. (Gaussian kernel).

$k(x, l^{(1)})$.

What these kernels actually do?

Kernels and similarity:

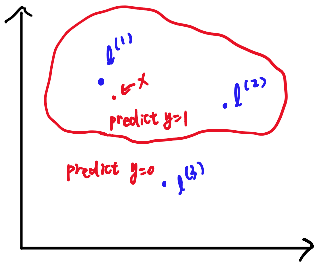
$f_1 = \text{similarity}(x, l^{(1)}) = \exp\left(-\frac{\|x - l^{(1)}\|}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{j=1}^{n}(x_j - l_j^{(1)})^2}{2\sigma^2}\right)$.

if $x \simeq l^{(1)}$ $\|x - l^{(1)}\| \approx 0$

$f_1 \approx \exp\left(-\frac{0}{2\sigma^2}\right) \approx 1$.

if x if far from $l^{(1)}$

$f_1 = \exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \approx 0$



predict "1" when

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

compute f_1, f_2, f_3 from x

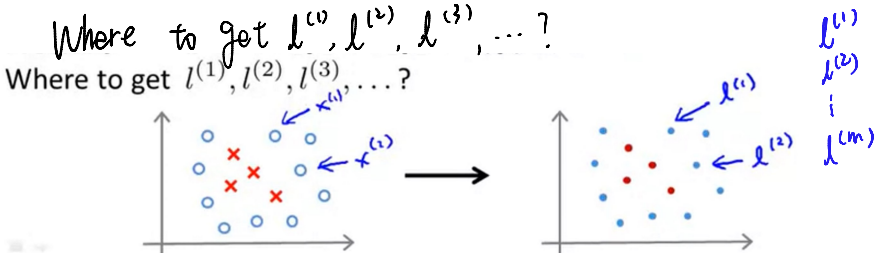
$\theta_0 = -0.5 \quad \theta_1 = 1 \quad \theta_2 = 1 \quad \theta_3 = 0$.

$f_1 \approx 1, f_2 \approx 0, f_3 \approx 0$.

$\theta_0 + \theta_1 \times 1 = 0.5 \geq 0$.

b. Kernels II:

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \ldots$?



SVM with Kernels

- Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$,
- choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, \ldots, l^{(m)} = x^{(m)}$.

Given example x:

- $f_1 = \text{similarity}(x, l^{(1)})$ ($l^{(1)} = x^{(1)}$)
- $f_2 = \text{similarity}(x, l^{(2)})$
- $\ldots$

$f = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$ $f_0 = 1$

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

$x^{(i)} \rightarrow \begin{bmatrix} f_1^{(i)} = \text{sim}(x^{(i)}, l^{(1)}) \\ f_2^{(i)} = \text{sim}(x^{(i)}, l^{(2)}) \\ \vdots \\ f_i^{(i)} = \text{sim}(x^{(i)}, l^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1 \\ \vdots \\ f_m^{(i)} = \text{sim}(x^{(i)}, l^{(m)}) \end{bmatrix}$ ($l^{(i)} = x^{(i)}$)

$x^{(i)} \in \mathbb{R}^{n+1}$ (or $\mathbb{R}^n$)

$f^{(i)} = \begin{bmatrix} f_0^{(i)} \\ f_1^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix}$ $f_0^{(i)} = 1$

Training

$$\min C\sum_{i=1}^{m} y^{(i)}\,\text{cost}_1(\theta^T f^{(i)}) + (1-y^{(i)})\,\text{cost}_0(\theta^T f^{(i)}) + \frac{1}{2}\sum_{j=1}^{n=m}\theta_j^2 = \theta^T\theta \quad \text{do not regularize } \theta_0.$$

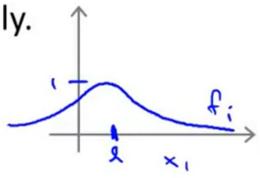
$C = \frac{1}{\lambda}$ Large C: Lower bias, high variance.

Small C: Higher bias, low variance.

σ^2 Large σ^2: Features f_i vary more smoothly.

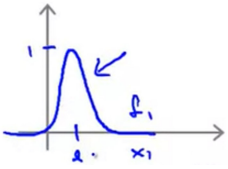
→ Higher bias, lower variance.

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



Small σ^2: Features f_i vary less smoothly.

Lower bias, higher variance.



- SVMs in Practice:
  a. Using An SVM:

  Use SVM software package(e.g. liblinear, libsvm,…)to solve for parameters θ

Need to specify:

→ Choice of parameter C.

Choice of kernel (similarity function):

E.g. No kernel ("linear kernel")

Predict "y = 1" if $\theta^T x \geq 0$

$\theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \geq 0$

n large, m small $x \in \mathbb{R}^{n+1}$

many features, few data, so we want a linear boundary.

Gaussian kernel:

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

Need to choose σ^2.

When to choose Gaussian kernel.

$x \in \mathbb{R}^n$, n small and/or m large

if you have features of very different scales, perform feature scaling.

Other choices of kernel

Note: Not all similarity functions $\text{similarity}(x, l)$ make valid kernels.

→ (Need to satisfy technical condition called "Mercer's Theorem" to make sure SVM packages' optimizations run correctly, and do not diverge).

Many off-the-shelf kernels available:

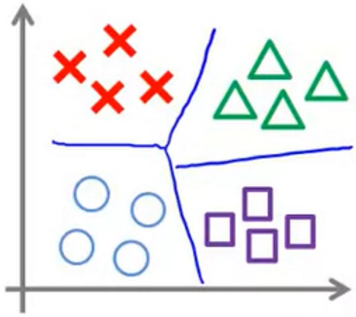
- Polynomial kernel: $k(x, l) = (x^T l)^3$, $(x^T l)^2$ to $(x^T l + 1)^3$, $(x^T l + 5)^4$

$x^T l$ bigger indicate they are more similar.

$(x^T l + \text{constant})^{\text{degree}}$

- More esoteric: String kernel, chi-square kernel, histogram intersection kernel, ... text classification.

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

$y=1$ $y=2$ .... $\theta = K$

Logistic regression vs. SVMs

n = number of features ($x \in \mathbb{R}^{n+1}$), m = number of training examples

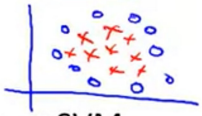
→ If n is large (relative to m): (e.g. $n \geq m$, $n = 10{,}000$, $m = 10 \cdots 1000$)

→ Use logistic regression, or SVM without a kernel ("linear kernel")

→ If n is small, m is intermediate: ($n = 1$-1000, $m = 10$-$10{,}000$)

→ Use SVM with Gaussian kernel

If n is small, m is large: ($n = 1$-1000, $m = 50{,}000+$)



→ Create/add more features, then use logistic regression or SVM without a kernel

→ Neural network likely to work well for most of these settings, but may be slower to train.