

Machine Learning

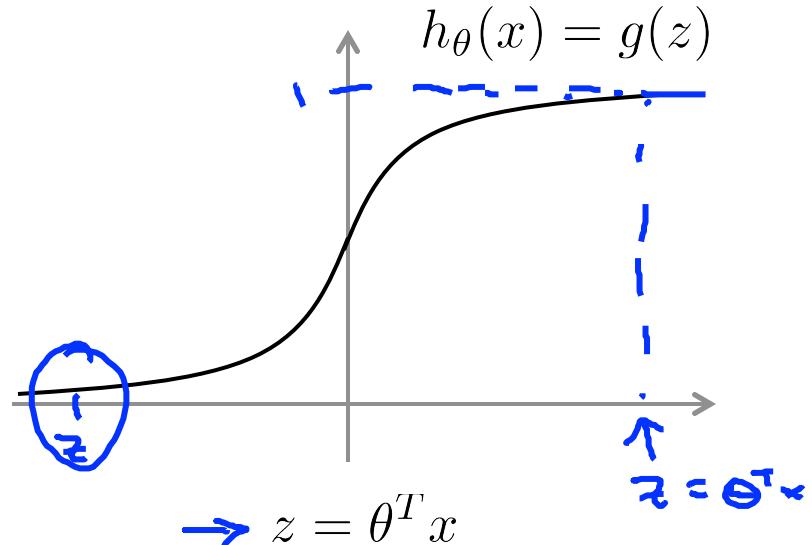
# Support Vector Machines

---

## Optimization objective

# Alternative view of logistic regression

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



If  $y = 1$ , we want  $h_{\theta}(x) \approx 1$      $\downarrow$      $\theta^T x \gg 0$

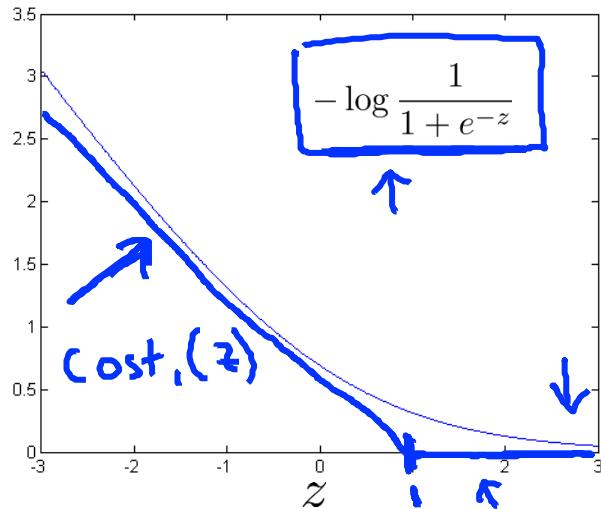
If  $y = 0$ , we want  $h_{\theta}(x) \approx 0$      $\downarrow$      $\theta^T x \ll 0$

## Alternative view of logistic regression

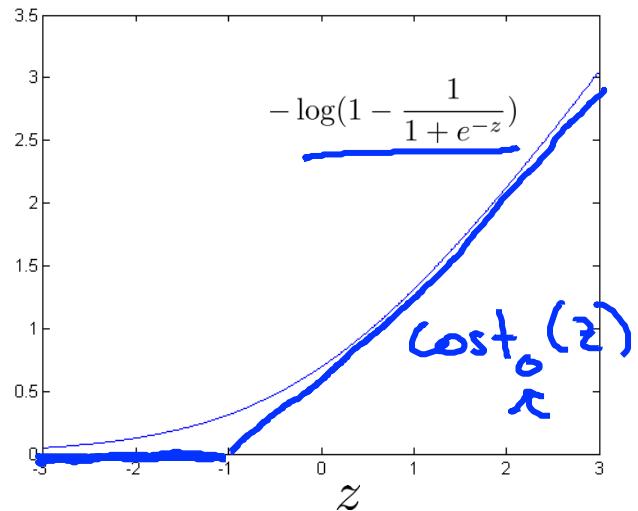
$$\text{Cost of example: } -(y \log h_{\theta}(x) + (1 - y) \log(1 - h_{\theta}(x))) \leftarrow$$

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

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



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



# Support vector machine

Logistic regression:

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

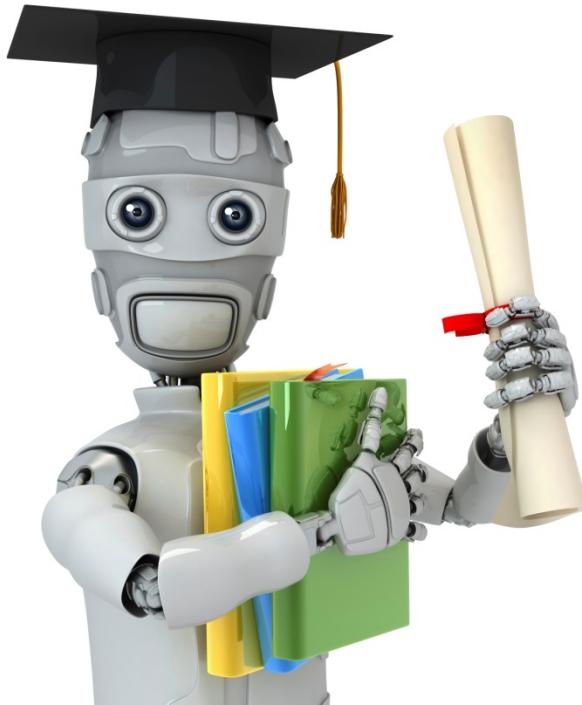
Support vector machine:

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

## SVM hypothesis

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

Hypothesis:



Machine Learning

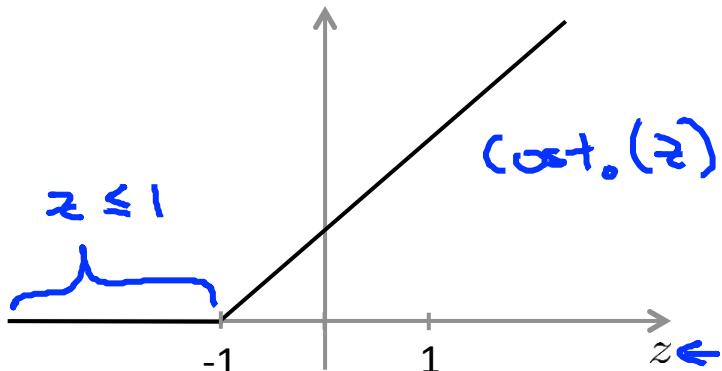
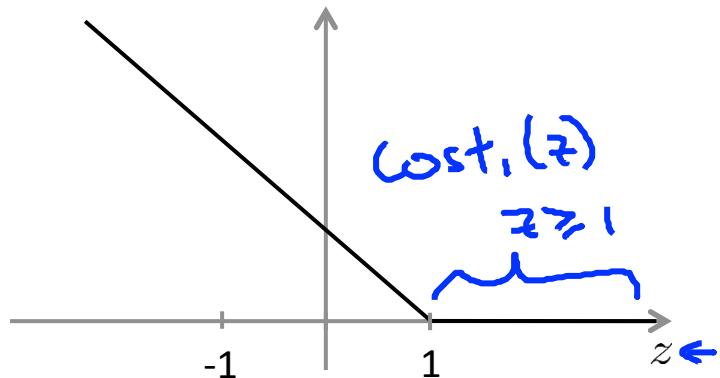
# Support Vector Machines

---

## Large Margin Intuition

# Support Vector Machine

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



→ If  $y = 1$ , we want  $\underline{\theta^T x \geq 1}$  (not just  $\geq 0$ )

$$\Theta^T x \geq \alpha \ 1$$

→ If  $y = 0$ , we want  $\underline{\theta^T x \leq -1}$  (not just  $< 0$ )

$$\Theta^T x \leq \alpha \ -1$$

$$C = 100,000$$

## SVM Decision Boundary

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

Whenever  $y^{(i)} = 1$ :

$$\theta^T x^{(i)} \geq 1$$

Whenever  $y^{(i)} = 0$ :

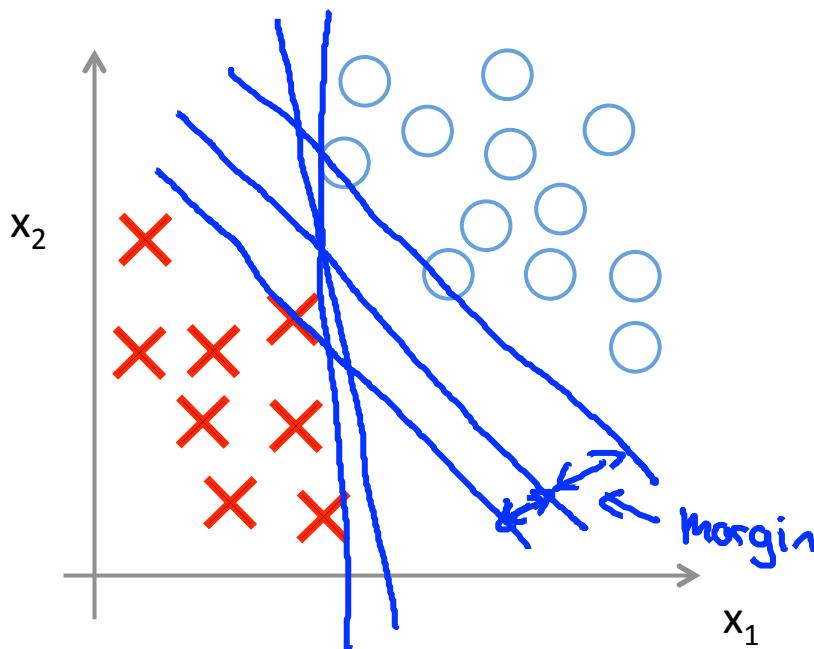
$$\theta^T x^{(i)} \leq -1$$

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

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

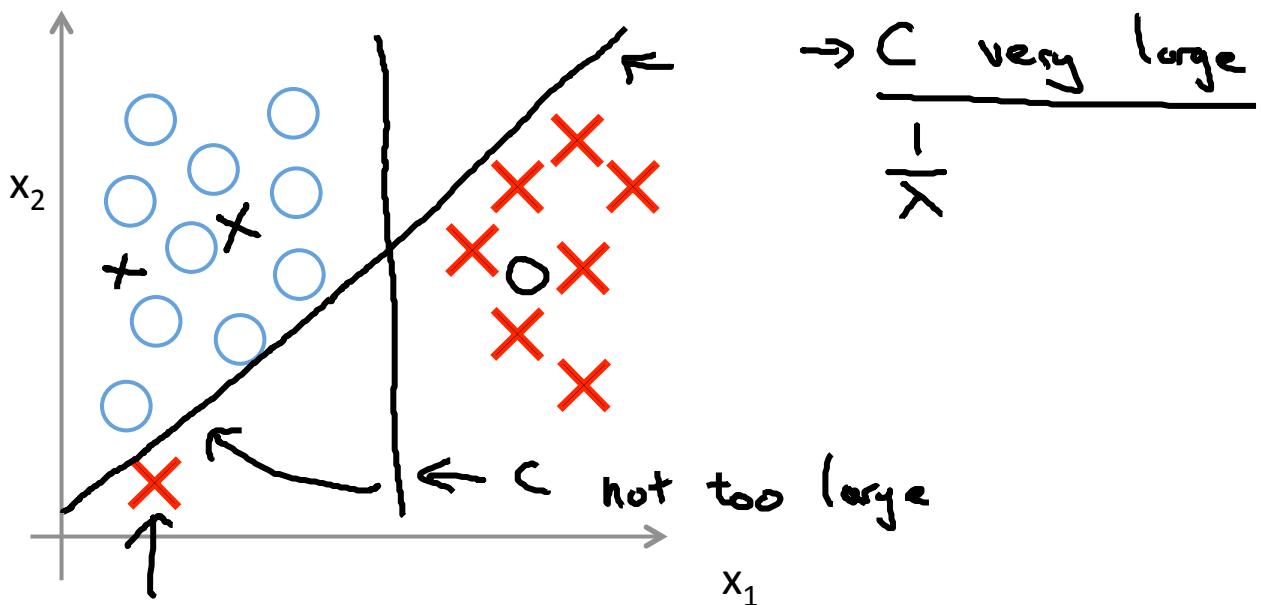
$$\theta^T x^{(i)} \leq -1 \quad \text{if } y^{(i)} = 0$$

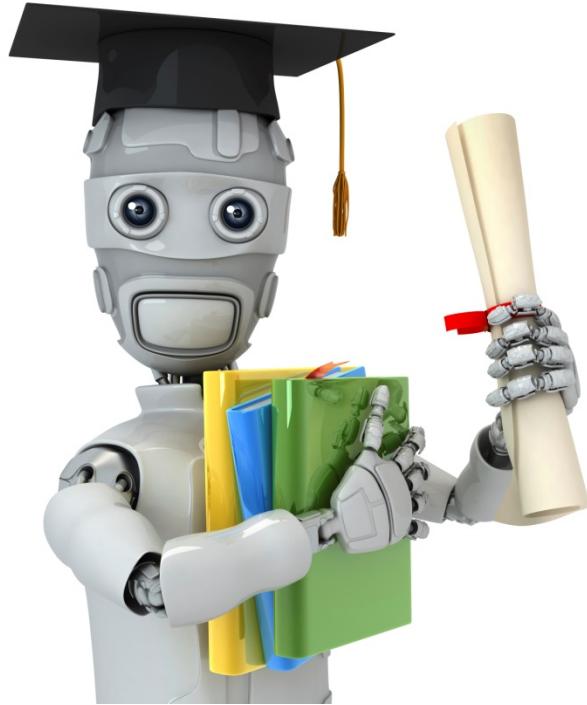
## SVM Decision Boundary: Linearly separable case



Large margin classifier

# Large margin classifier in presence of outliers





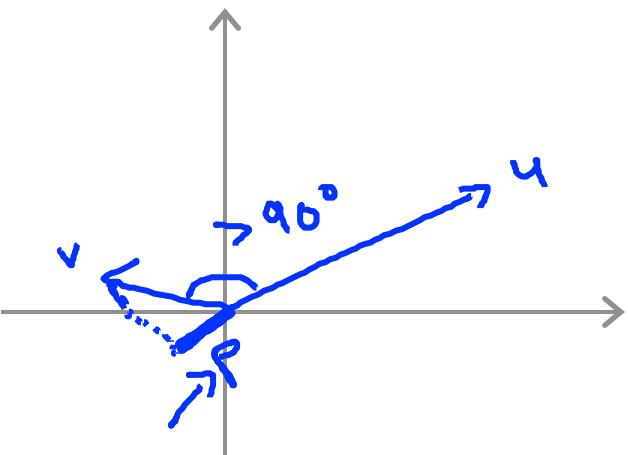
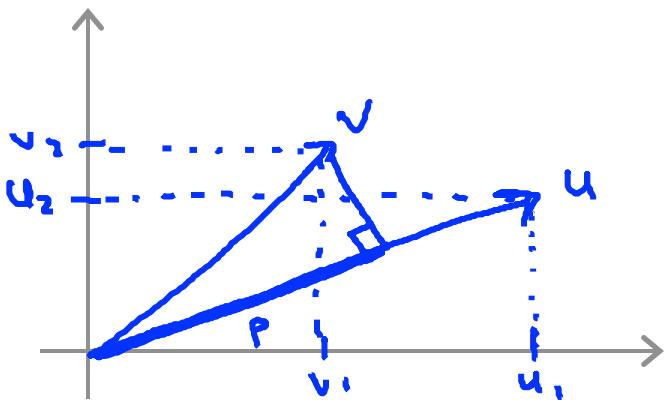
Machine Learning

# Support Vector Machines

---

The mathematics  
behind large margin  
classification (optional)

## Vector Inner Product



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

$$u^T v = ? \quad [u_1 \ u_2] \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$\|u\| = \text{length of vector } u \\ = \sqrt{u_1^2 + u_2^2} \in \mathbb{R}$$

$p = \text{length of projection of } v \text{ onto } u.$

$$u^T v = \frac{p \cdot \|u\|}{\|u\|} \leftarrow = v^T u$$

Signed

$$= u_1 v_1 + u_2 v_2 \leftarrow p \in \mathbb{R}$$

$$u^T v = p \cdot \|u\|$$

$$p < 0$$

$$\omega = (\sqrt{\omega})^2$$

## SVM Decision Boundary

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

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

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

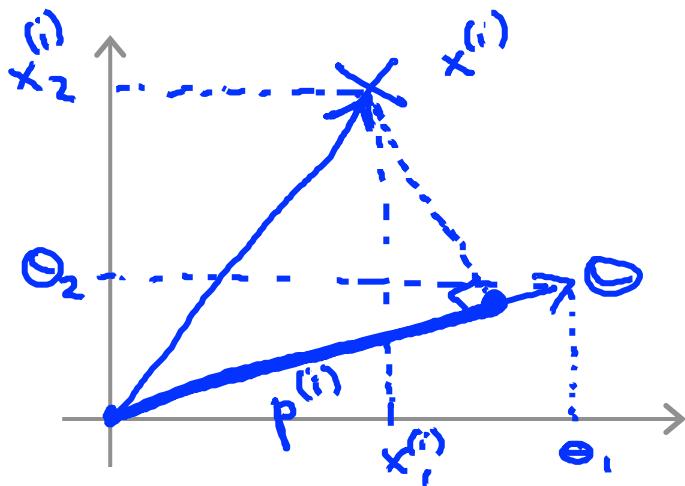
Simplification:  $\underline{\theta_0 = 0}$ .  $\underline{n=2}$

$$= \|\theta\|$$

$$\begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}, \theta_0 = 0$$

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

$$\begin{array}{c} \uparrow \\ \theta^T \\ \uparrow \\ u^T v \end{array}$$



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

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

## SVM Decision Boundary

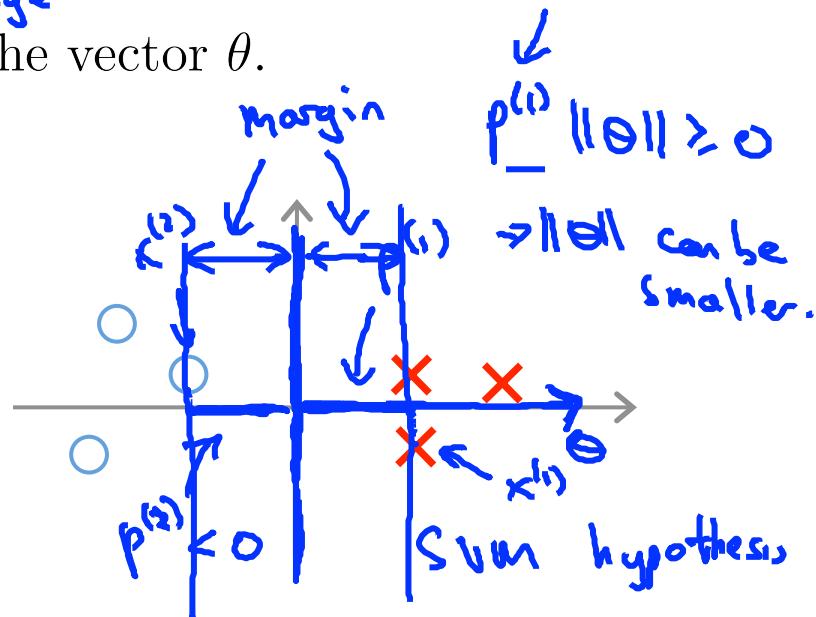
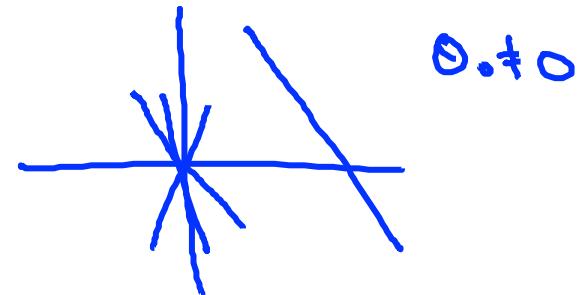
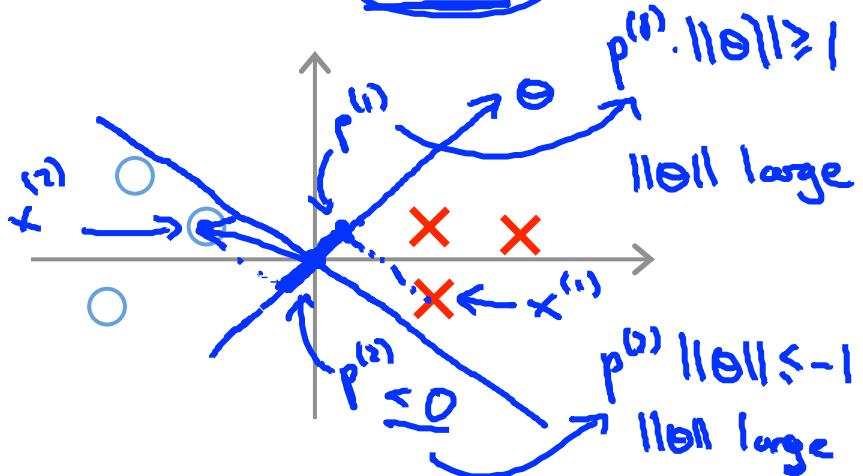
$$\rightarrow \min_{\theta} \frac{1}{2} \sum_{j=1}^n \theta_j^2 = \frac{1}{2} \|\theta\|^2 \leftarrow$$

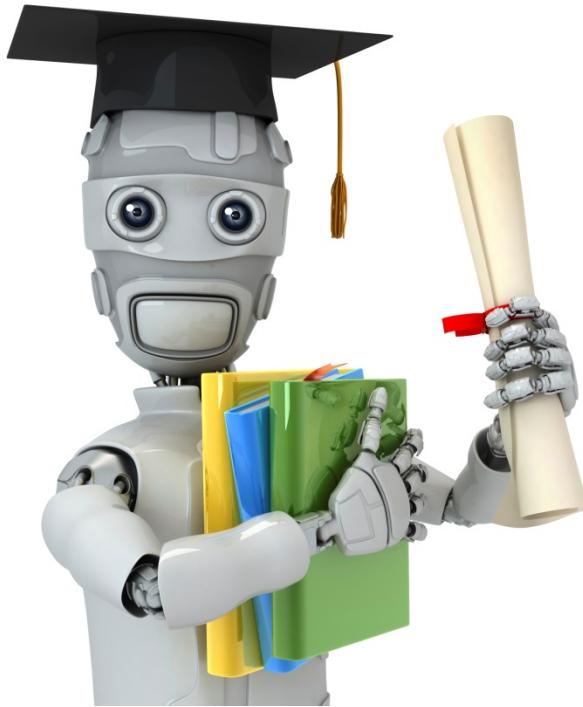
s.t.  $\boxed{p^{(i)} \cdot \|\theta\| \geq 1}$  if  $y^{(i)} = 1$

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

where  $p^{(i)}$  is the projection of  $x^{(i)}$  onto the vector  $\theta$ .

Simplification:  $\theta_0 = 0$





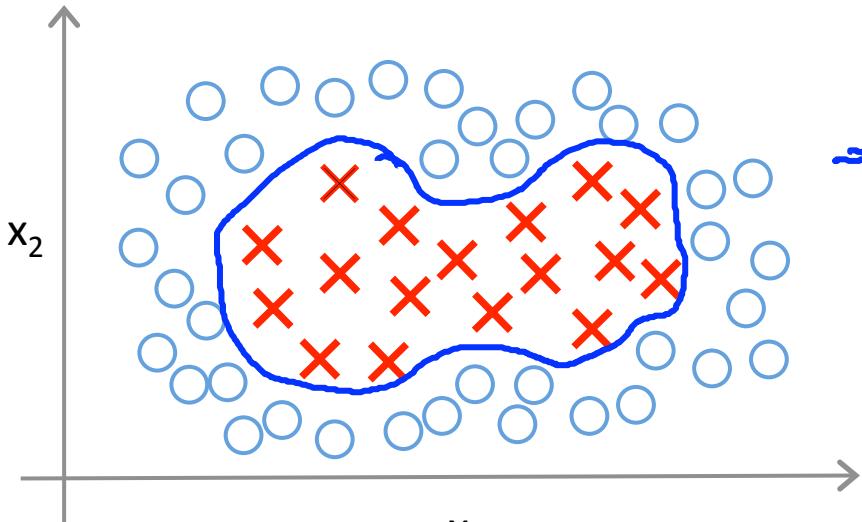
Machine Learning

# Support Vector Machines

---

## Kernels I

## Non-linear Decision Boundary



Predict  $y = 1$  if

$$\theta_0 + \theta_1 \underline{x_1} + \theta_2 \underline{x_2} + \theta_3 \underline{x_1 x_2} + \theta_4 \underline{x_1^2} + \theta_5 \underline{x_2^2} + \dots \geq 0$$

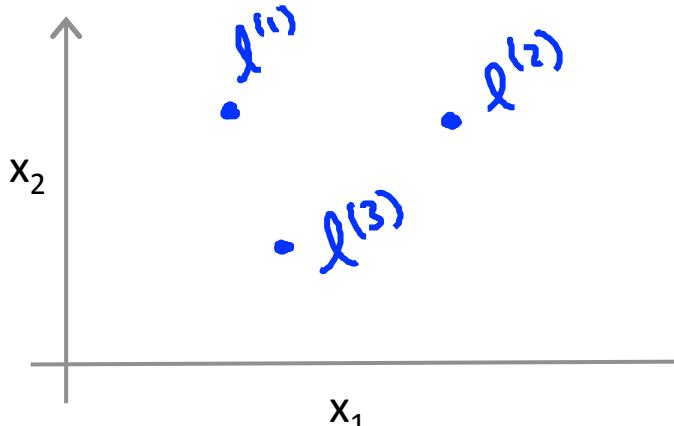
$$h_{\theta}(x) = \begin{cases} 1 & \text{if } \theta_0 + \theta_1 x_1 + \dots \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

$$\rightarrow \theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \dots$$

$$f_1 = x_1, \quad f_2 = x_2, \quad f_3 = x_1 x_2, \quad f_4 = x_1^2, \quad f_5 = x_2^2, \dots$$

Is there a different / better choice of the features  $f_1, f_2, f_3, \dots$ ?

## Kernel



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

Given  $x$ :

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

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

$$f_3 = \text{similarity}(x, l^{(3)}) = \exp(\dots)$$

↑ Kernel (Gaussian kernels)

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

## Kernels and Similarity

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

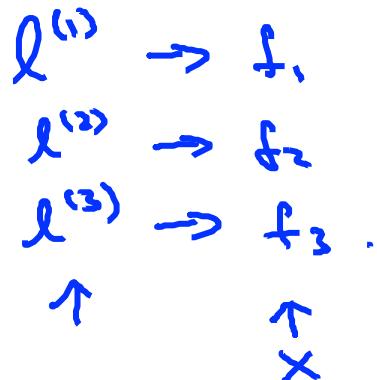


If  $x \approx l^{(1)}$  :

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

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

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



## Example:

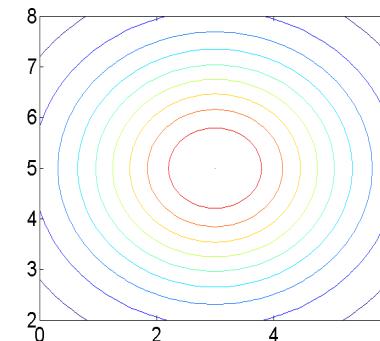
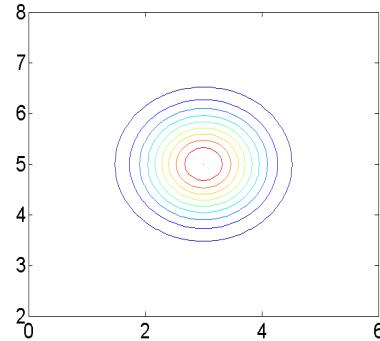
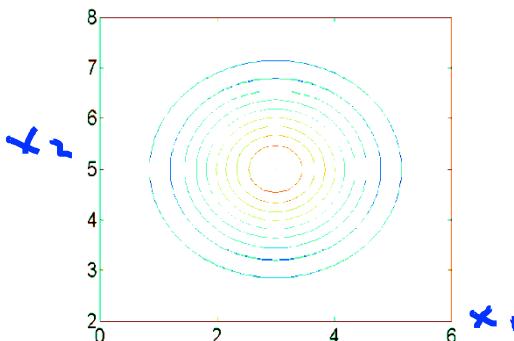
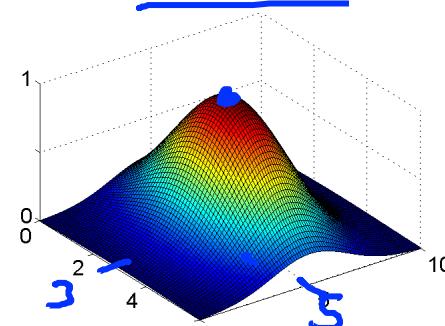
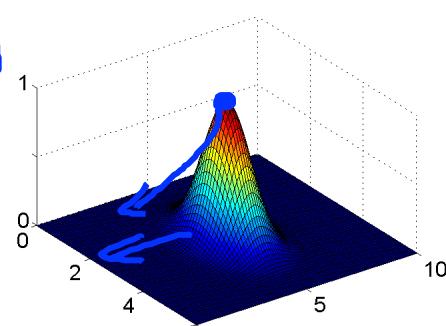
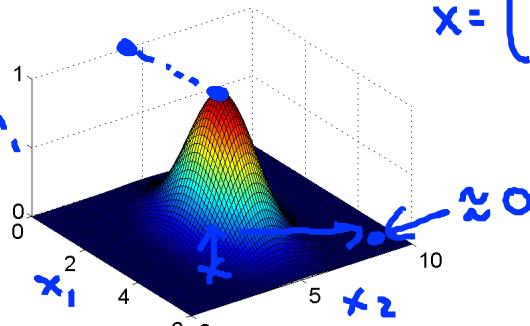
$$\rightarrow l^{(1)} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \quad f_1 = \exp\left(-\frac{\|x - l^{(1)}\|^2}{2\sigma^2}\right)$$

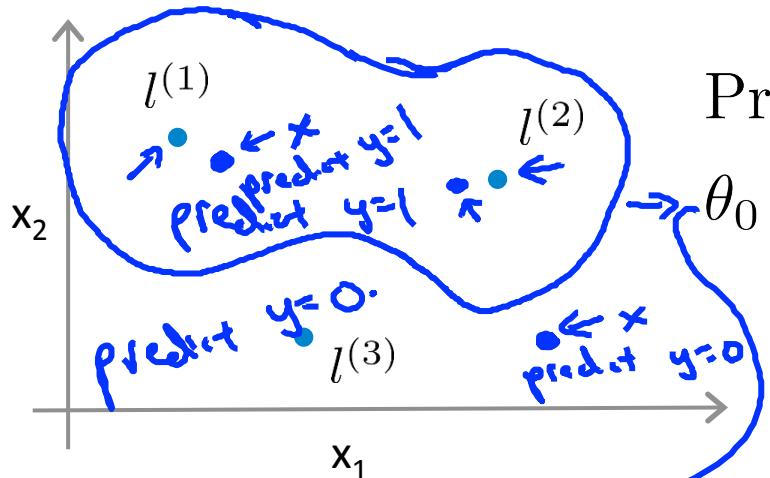
$$\rightarrow \sigma^2 = 1$$

$$x = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

$$\sigma^2 = 0.5$$

$$\sigma^2 = 3$$





Predict "1" when

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



---

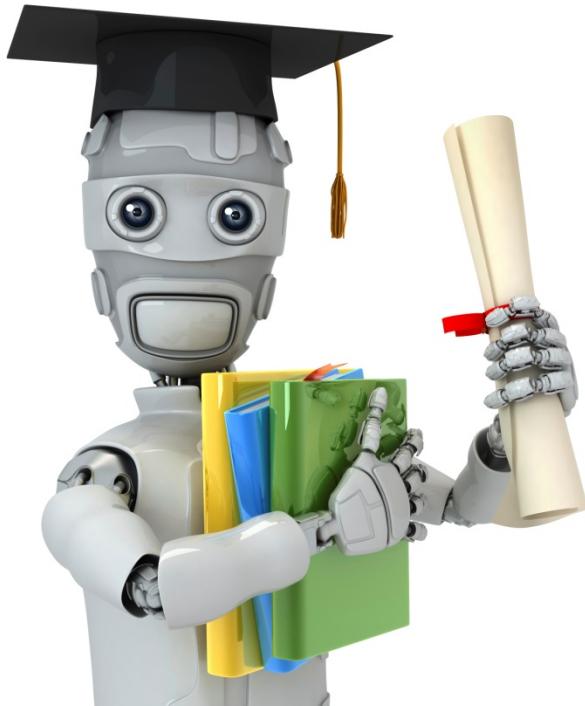

$$\underline{\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0}$$

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

$$\begin{aligned} \rightarrow \theta_0 + \theta_1 \cdot 1 + \theta_2 \cdot 0 + \theta_3 \cdot 0 \\ = -0.5 + 1 = 0.5 \geq 0 \end{aligned}$$

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

$$\rightarrow \underline{\theta_0 + \theta_1 f_1 + \dots} \approx -0.5 < 0$$



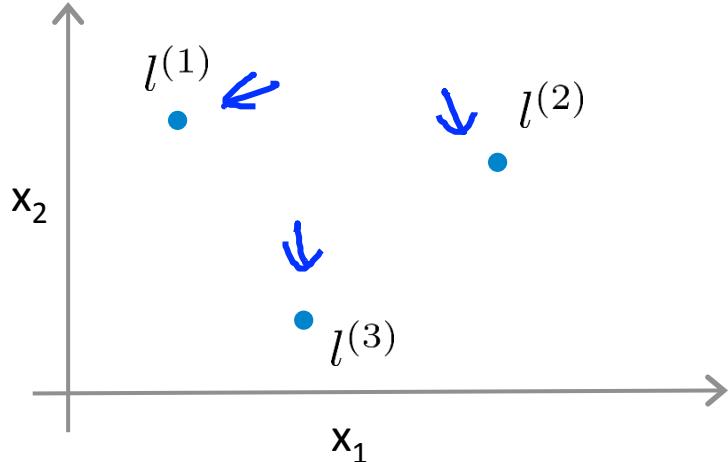
Machine Learning

# Support Vector Machines

---

## Kernels II

## Choosing the landmarks

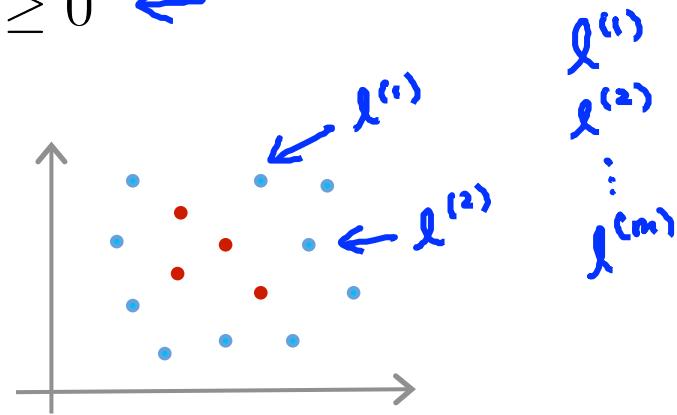
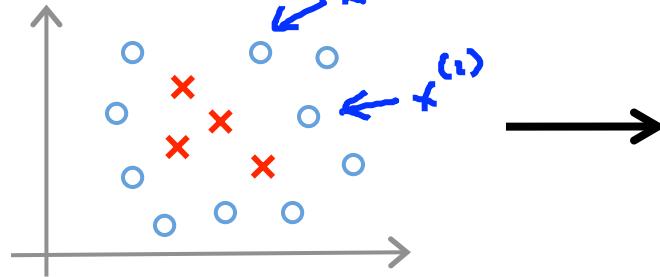


Given  $x$ :

$$\begin{aligned} \rightarrow f_i &= \text{similarity}(x, l^{(i)}) \\ &= \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right) \end{aligned}$$

Predict  $y = 1$  if  $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

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



## SVM with Kernels

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

Given example  $\underline{x}$ :

$$\begin{aligned} \rightarrow f_1 &= \text{similarity}(x, l^{(1)}) && \downarrow x^{(1)} \\ \rightarrow f_2 &= \text{similarity}(x, l^{(2)}) \\ &\vdots \\ \rightarrow f_m &= \text{similarity}(x, l^{(m)}) \end{aligned}$$

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

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

$$\begin{aligned} \underline{x^{(i)}} \rightarrow f_1^{(i)} &= \overline{\text{sim}}(x^{(i)}, l^{(1)}) && \downarrow x^{(i)} \\ f_2^{(i)} &= \overline{\text{sim}}(x^{(i)}, l^{(2)}) \\ &\vdots \\ f_m^{(i)} &= \overline{\text{sim}}(x^{(i)}, l^{(m)}) = \exp(-\frac{\alpha}{\gamma_{\text{sim}}}) = 1 \end{aligned}$$

$$\begin{aligned} \underline{x^{(i)}} \in \mathbb{R}^{n+1} & \quad (\text{or } \mathbb{R}^n) \\ f^{(i)} = & \begin{bmatrix} f_0^{(i)} \\ f_1^{(i)} \\ f_2^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix} \\ f_0^{(i)} &= 1 \end{aligned}$$

## SVM with Kernels

Hypothesis: Given  $\underline{x}$ , compute features  $\underline{f} \in \mathbb{R}^{m+1}$

→ Predict "y=1" if  $\theta^T \underline{f} \geq 0$

$$\sqrt{\theta_0 + \theta_1 + \dots + \theta_m}$$

$$\theta \in \mathbb{R}^{m+1}$$

$$\theta_0 f_0 + \theta_1 f_1 + \dots + \theta_m f_m$$

Training:

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

$$\begin{array}{c} n = m \\ \cancel{\theta_0} = m \\ \frac{1}{2} \sum_{j=1}^m \theta_j^2 \\ \rightarrow \theta_0 \end{array}$$

$$\cancel{\theta_0} \quad \theta^T f^{(i)}$$

$$\begin{bmatrix} - & \sum_j \theta_j^2 \\ - & \end{bmatrix} = \theta^T \theta \quad \leftarrow \theta = \begin{bmatrix} \theta_0 \\ \vdots \\ \theta_m \end{bmatrix}$$

$$\rightarrow \theta^T M \theta \quad \leftarrow \| \theta \|^2$$

(ignoring  $\theta_0$ )  
 $M = 10,000$

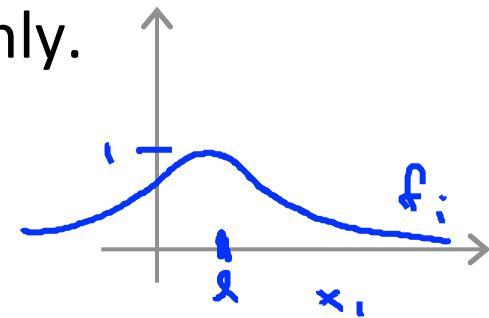
## SVM parameters:

$C \left( = \frac{1}{\lambda} \right)$ .  $\rightarrow$  Large C: Lower bias, high variance. (small  $\lambda$ )  
 $\rightarrow$  Small C: Higher bias, low variance. (large  $\lambda$ )

$\sigma^2$  Large  $\sigma^2$ : Features  $f_i$  vary more smoothly.

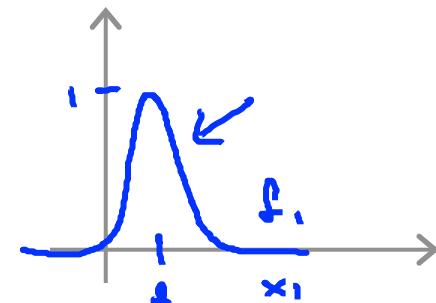
$\rightarrow$  Higher bias, lower variance.

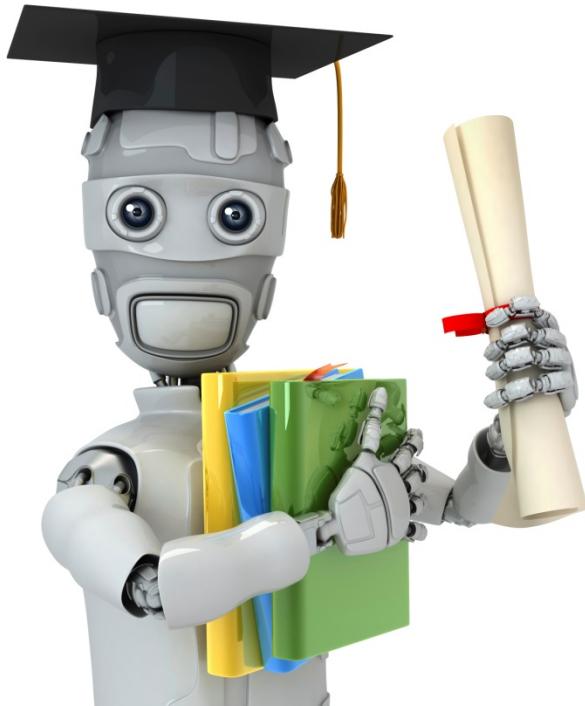
$$\exp \left( - \frac{\|x - f_i\|^2}{2\sigma^2} \right)$$



Small  $\sigma^2$ : Features  $f_i$  vary less smoothly.

Lower bias, higher variance.





Machine Learning

# Support Vector Machines

---

## Using an SVM

Use SVM software package (e.g. liblinear, libsvm, ...) to solve for parameters  $\theta$ .



Need to specify:

→ Choice of parameter C.

Choice of kernel (similarity function):

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

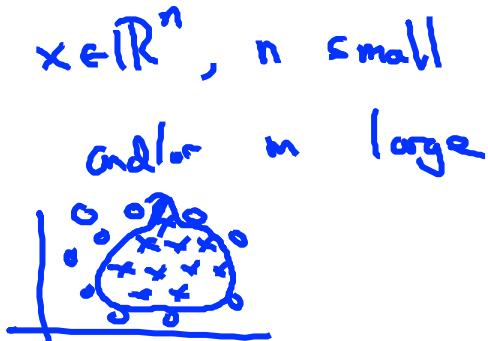
Predict "y = 1" if  $\underline{\theta^T x} \geq 0$

$$\theta_0 + \theta_1 x_1 + \dots + \theta_n x_n \geq 0 \quad \rightarrow \underline{n} \text{ large, } \underline{m} \text{ small} \quad \underline{x \in \mathbb{R}^{n+1}}$$

→ Gaussian kernel:

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

Need to choose  $\underline{\sigma^2}$ .



## Kernel (similarity) functions:

function  $f = \text{kernel}(x_1, x_2)$

$$f = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)$$

return

$$\begin{aligned} x &\rightarrow \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{aligned}$$

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

$$\begin{aligned} &\Rightarrow \|x - l\|^2 \quad x \in \mathbb{R}^{n+1} \\ &V = x - l \\ &\|v\|^2 = v_1^2 + v_2^2 + \dots + v_n^2 \\ &= (x_1 - l_1)^2 + (x_2 - l_2)^2 + \dots + (x_n - l_n)^2 \\ &\quad \underbrace{\quad}_{1000 \text{ feet}^2} \quad \underbrace{\quad}_{1-5 \text{ bedrooms}} \end{aligned}$$

## 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).

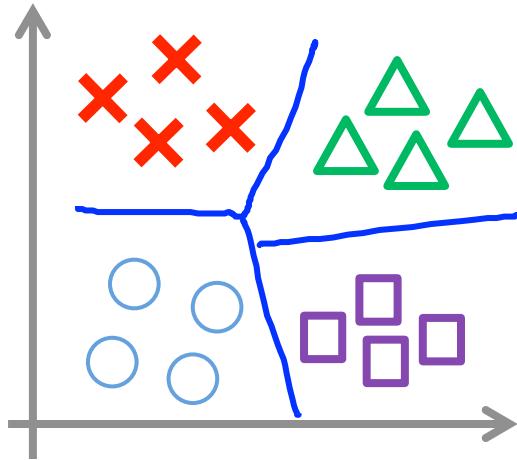
Many off-the-shelf kernels available:

- Polynomial kernel:  $k(x, l) =$

$$(x^T l)^2, \quad (x^T l + \text{constant})^{\text{degree}}$$
$$(x^T l)^3, \quad (x^T l + 1)^3, \quad (x^T l + 5)^4$$

- More esoteric: String kernel, chi-square kernel, histogram intersection kernel, ...  
 $\text{sim}(x, l)$

## Multi-class classification



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

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  $i = 1, 2, \dots, K$ ), get  $\theta^{(1)}, \theta^{(2)}, \dots, \underline{\theta^{(K)}}$
- Pick class  $i$  with largest  $(\theta^{(i)})^T x$
- $\underline{y=1} \quad \underline{y=2} \quad \dots \quad \underline{\theta^{(K)}}$

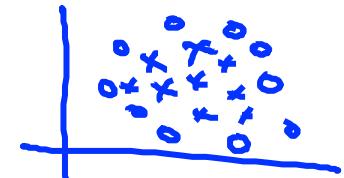
## Logistic regression vs. SVMs

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

- If  $n$  is large (relative to  $m$ ): (e.g.  $n \geq m$ ,  $n = \underline{10,000}$ ,  $m = \underline{10} \dots \underline{1000}$ )
- Use logistic regression, or SVM without a kernel ("linear kernel")

- If  $n$  is small,  $m$  is intermediate: ( $n = \underline{1-1000}$ ,  $m = \underline{10 - 10,000}$ )
  - Use SVM with Gaussian kernel

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



- 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.