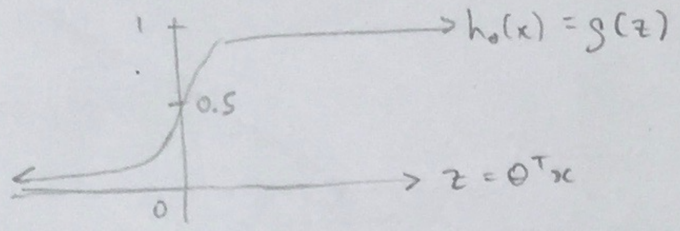
ML COURSERA WEEK 7:

LEO ROBINOVITCH

Support Vector Machines: powerful supervised learning algorithm

⇒ recall logistic regression: sigmoid activation function

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

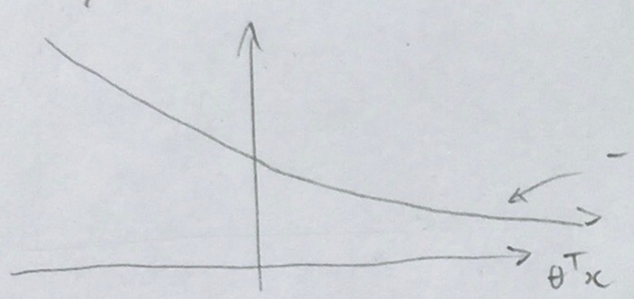


$\left(g = \frac{1}{1+e^{-z}}\right)$

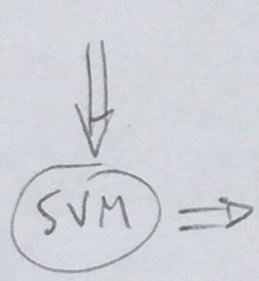
↳ need $\theta^T x \gg 0$ for $h_\theta(x) \simeq 1$ and vice versa for $h_\theta(x) = 0$

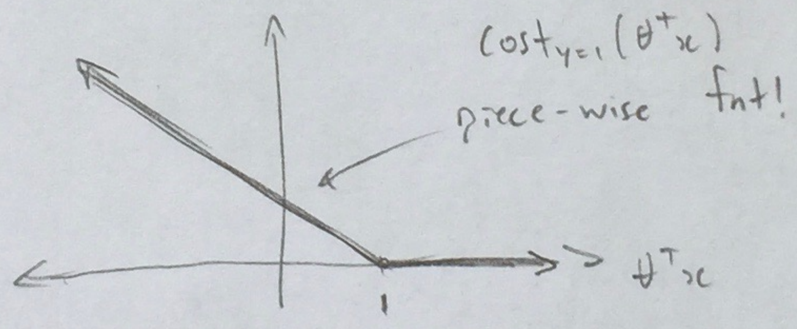
cost of single example $J_i = -\underbrace{[y \log(h_\theta(x))}_{①} + \underbrace{(1-y)\log(1-h_\theta(x))]}_{②}$ for $(x^{(i)}, y^{(i)})$

↳ if $y = 1$; only term ①:

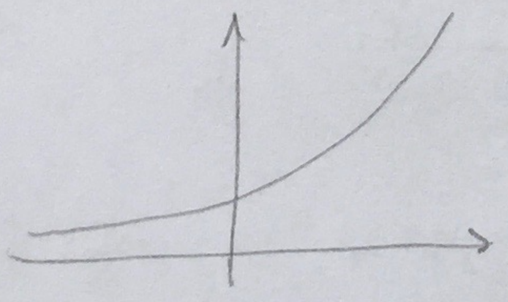


$-\log\left(\frac{1}{1+e^{-\theta^T x}}\right)$ ⇒ $\theta^T x \gg 0$ → contributes little to cost!

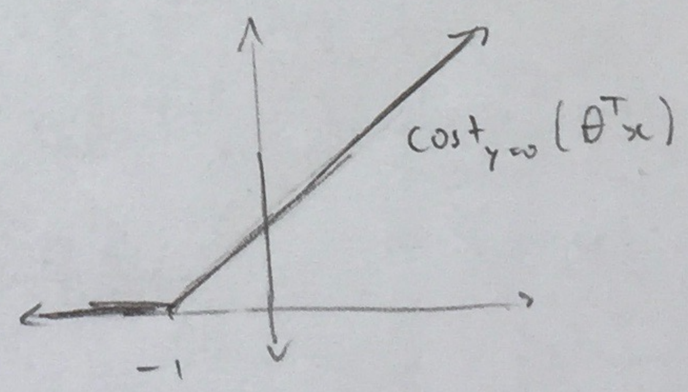




→ if $y = 0$: $-\log\left(1 - \frac{1}{1+e^{-\theta^T x}}\right)$



⇒ SVM ⇒

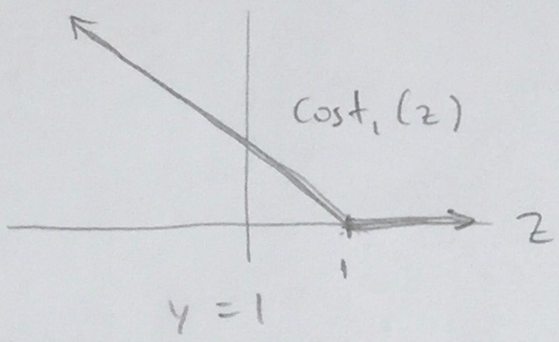


So: $J_{log.reg.} = \min_\theta \frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}(-\log(h_\theta(x^{(i)})) + (1-y^{(i)})(-\log(1-h_\theta(x^{(i)})))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$

⇒ $J_{SVM} = \min_\theta C\sum_{i=1}^{m}\left[y^{(i)} cost_{y=1}(\theta^T x^{(i)}) + (1-y^{(i)}) cost_{y=0}(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=0}^{n}\theta_j^2$

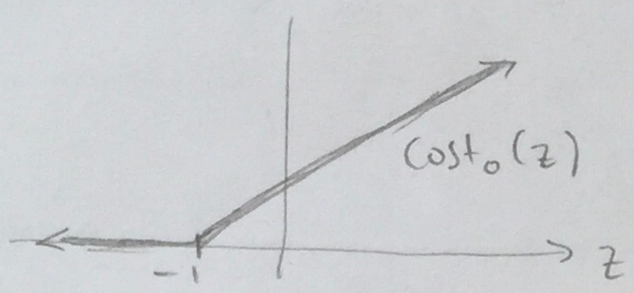
=> Note: rather than $A + \lambda B \longrightarrow CA + B$ $\left(C \propto \frac{1}{\lambda} \right)$

SVM hypothesis $\begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$



$y = 1$

want $z = \theta^T x \geq 1$ (not 0)



$y = 0$

want $z = \theta^T x \leq -1$ (not 0)

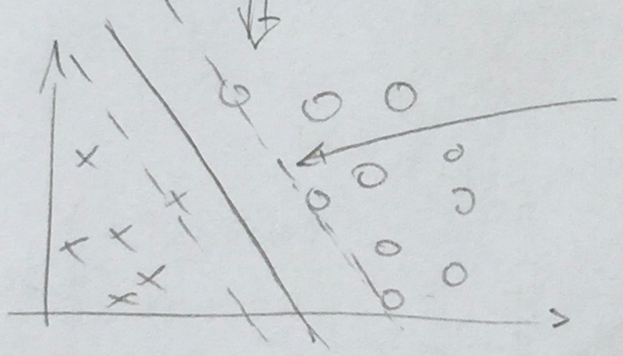
↳ SVM's give some "safety factor" margin ($|1|$ rather than 0)

↳ $\min_\theta \cancel{C \times 0} + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$ (C large) subject to $\theta^T x \geq 1$ if $y^{(i)} = 1$

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

SVM

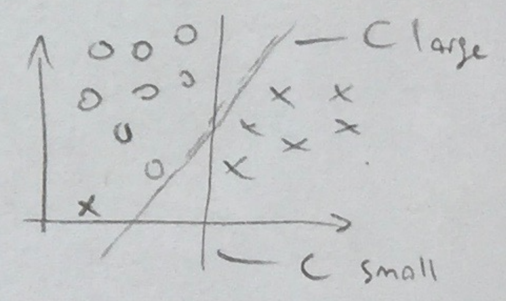
(Linearly Seperable)



SVM's ensure maximization of margins (- - -) "LARGE MARGIN CLASSIFIER"

(given C large)

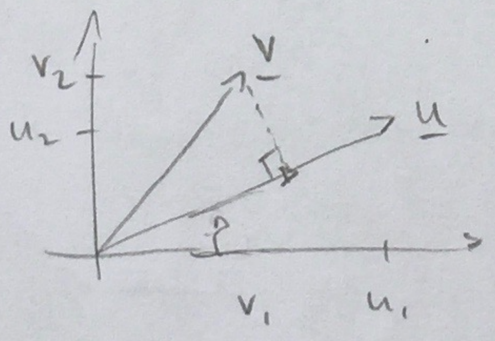
=> if C large, sensitive to outliers →



Mathematics of Large Margin Classifier:

=> vector inner product:

$u^T v$ $(u \cdot v)$



$\|\underline{u}\| = \sqrt{u_1^2 + u_2^2} \in \mathbb{R}^1$

p = length of projection of v onto u (positive or negative)

$\underline{u}^T \underline{v} = p \cdot \|\underline{u}\| = \underline{v}^T \underline{u} = u_1 v_1 + u_2 v_2$

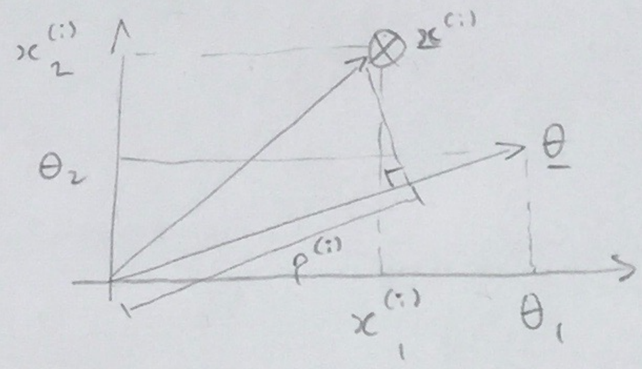
=> SVM Decision Boundary → ignore θ_0, $n=2$

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

↑ MINIMIZE SQUARED NORM OF θ

such that $\theta^T x^{(i)} \geq 1$ for $y^{(i)}=1$

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



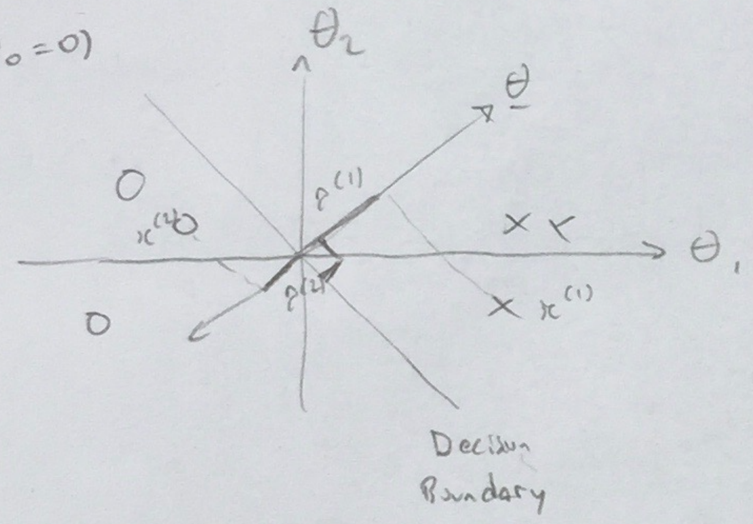
$$\Rightarrow \theta^T x^{(i)} = p^{(i)}\|\theta\| = \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)}$$

WHEN C LARGE:

$$\Rightarrow \min \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}\|\underline{\theta}\|^2 \quad \text{such that} \quad p^{(i)}\|\underline{\theta}\| \geq 1 \text{ if } y^{(i)}=1$$

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

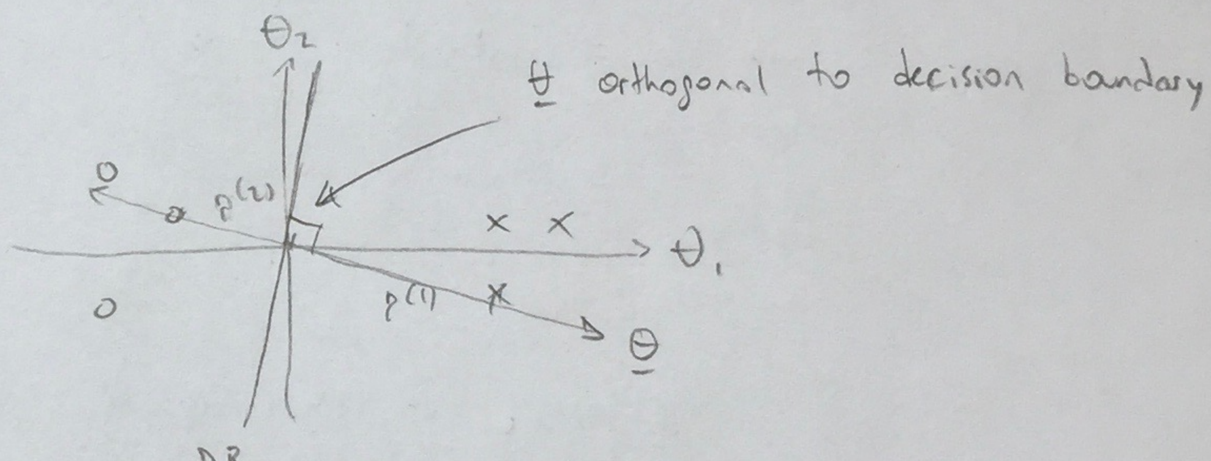
($\theta_0=0$)



need $p^{(2)}\|\theta\| \leq -1$, but $p^{(2)}$ small!

need $p^{(1)}\|\theta\| \geq 1$, but $p^{(1)}$ small!

BAD DECISION BOUNDARY

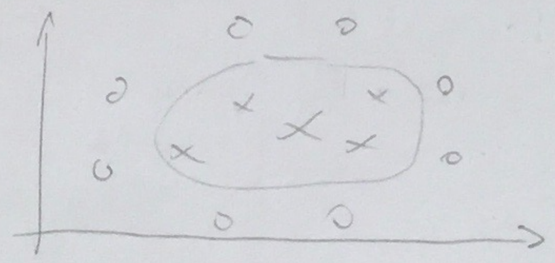


need $p^{(2)}\|\theta\| \leq -1$ → $p^{(2)}$ larger, so $\|\theta\|$ can be smaller! (same for $p^{(1)}$)

=> recall: goal is minimize $\|\theta\|$

↳ BETTER DECISION BOUNDARY

Non-Linear Decision Boundaries w/ SVM's: KERNELS



predict $y=1$ if

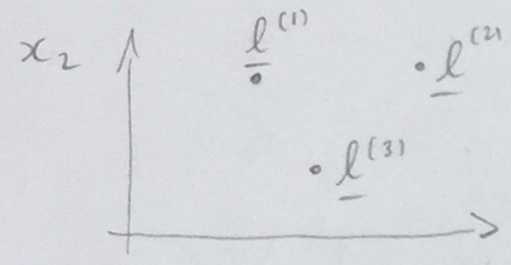
$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \dots + \theta_5 x_2^2 + \dots \geq 0$

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

f_j = feature:

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

Kernel: given x, compute new feature depending on proximity to landmarks.



ℓ = landmarks

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

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

↑ euclidean distance w

↓ GAUSSIAN KERNEL

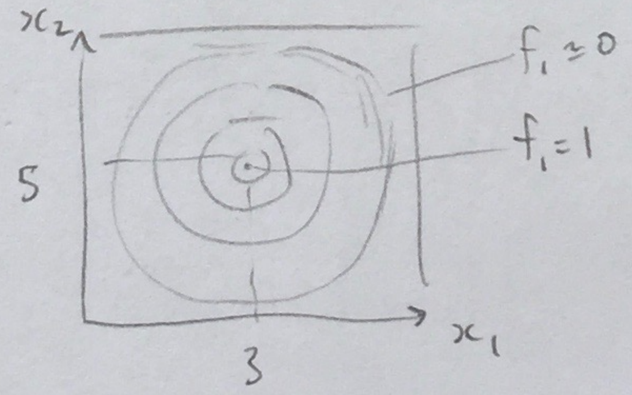
KERNEL = "SIMILARITY FUNCTION"

* note $\|\underline{x} - \ell^{(1)}\|^2 = \sum_{j=1}^{n} (x_j - \ell_j^{(1)})^2$

$\Rightarrow$ so if $x \simeq \underline{\ell}^{(1)} \rightarrow f_1 \simeq 1$, or if $x \neq \underline{\ell}^{(1)}$ at all, $f_1 \simeq 0$

↳ given one $\underline{x}$ → 3 features (one from each landmark)

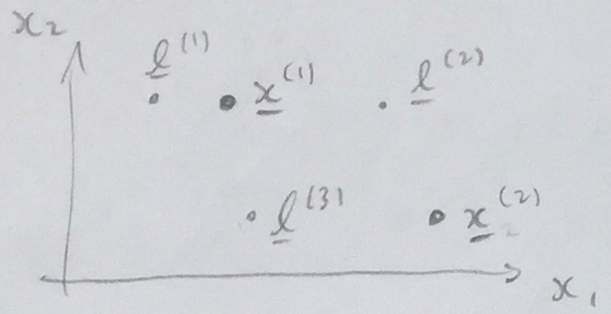
ex. $\underline{x} = \begin{bmatrix} 3 \\ 5 \end{bmatrix} \rightarrow$



$\Rightarrow \sigma^2$ controls width of "hump"

($\downarrow \sigma^2$, hump sharper
$\uparrow \sigma^2$, hump wider)

predict 1 when $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

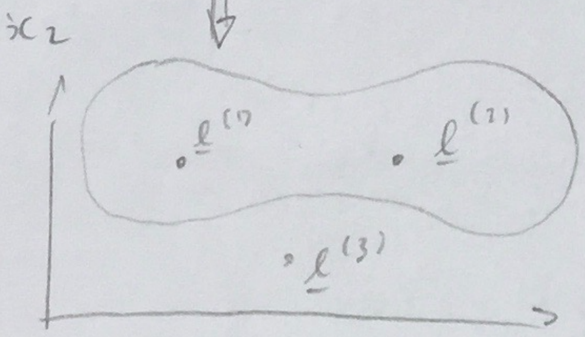


have $\theta_0 = -0.5$, $\theta_1 = 1$, $\theta_2 = 1$, $\theta_3 = 0$

given $\underline{x}^{(1)} \rightarrow f_1 \simeq 1$, $f_2 \simeq 0$, $f_3 \simeq 0$

$\Rightarrow -0.5 + 1 = 0.5 \geq 0 \rightarrow \underline{h_\theta(\underline{x}^{(1)}) = 1}$

given $\underline{x}^{(2)} \rightarrow -0.5 \leq 0 \rightarrow \underline{h_\theta(\underline{x}^{(2)}) = 0}$



$\rightarrow$ decision boundary result! ($\theta_3 = 0$)

$\Rightarrow$ where to get landmarks? How many?

↳ put landmarks @ locations of training examples

("m" landmarks)

$\Rightarrow$ measures how close a new point is to data in training set

$\Rightarrow$ given example $\underline{x}$: $f_1 = \text{similarity}(\underline{x}, \underline{\ell}^{(1)})$ where $\underline{\ell}^{(1)} = \underline{x}^{(1)}$

$f_2 = \text{similarity}(\underline{x}, \underline{\ell}^{(2)})$ where $\underline{\ell}^{(2)} = \underline{x}^{(2)}$

etc.

$\Rightarrow \underline{f} = \begin{bmatrix} f_0 = 1 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$ $[(m+1) \times 1]$

$\Rightarrow$ for $\underline{x}^{(i)} \rightarrow f_i = \text{sim}(\underline{x}^{(i)}, \underline{\ell}^{(i)}) \Rightarrow \boxed{\underline{x}^{(i)} \in \mathbb{R}^{n+1} \rightarrow \underline{f}^{(i)} \in \mathbb{R}^{m+1}}$

etc.

(one will be $\text{sim}(\underline{x}^{(i)}, \underline{\ell}^{(i)}) = 1$)

$\Rightarrow$ predict $y = 1$ if $\underline{\theta}^T \underline{f} \geq 0$ where $\underline{\theta} \in \mathbb{R}^{m+1}$

↳ get $\underline{\theta}$ using SVM!

=> SVM for $\underline{\theta}$:

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

($n=m$!; $\theta_0 = 1$ unregularized)

↳ in reality, $\sum_{j=1}^{n=m}\underline{\theta}_j^{\,2} = \underline{\theta}^T\underline{\theta}$ replaced w/ $\underline{\theta}^T\underline{\underline{M}}\,\underline{\theta}$

↳ depends on kernel used, runs more efficiently

* recommends using SVM software for minimization optimization

BUT: must choose C! → $\uparrow C \propto \downarrow \lambda$

↳ large C → lower bias, high variance

small C → higher bias, low variance

& must choose σ^2! → large σ^2, f_i varies smoothly

(higher bias, lower variance)

→ small σ^2, f_i varies sharply

(lower bias, high variance)

Practical SVM Use:

=> liblinear, libsvm packages to solve for $\underline{\theta}$

↳ specify C, kernel choice

↳ eg. "no kernel" / linear kernel → $y=1$ if $\underline{\theta}^T\underline{x} \geq 0$, $\underline{x} \in \mathbb{R}^{n+1}$

(n large, m small)

↳ e.g. gaussian kernel → choose σ^2 $\quad f_i = \exp\left(-\frac{\|\underline{x} - \underline{\ell}^{(i)}\|^2}{2\sigma^2}\right)$

where $\underline{\ell}^{(i)} = \underline{x}^{(i)}$

↳ must write fxn for software

(feature scale before using Gaussian!)

↳ ensures equal weight for each feature

⇒ Not all kernels work → must satisfy Mercer's theorem (don't diverge)

↳ polynomial kernel $k(x, \ell) = (\underline{x}^T \underline{\ell})^2$, $(\underline{x}^T \underline{\ell})^3$, $(\underline{x}^T \underline{\ell} + 1)^3$

(worse than Gaussian) ↳ choose degree, constant

↳ string, chi-square, histogram intersection, etc.

⇒ Multiclass Classification:

⇒ If not built in to software, use one-vs.-all → get $\underline{\theta}^{(1)}, \underline{\theta}^{(2)} \ldots \underline{\theta}^{(k)}$,
choose $\underline{\theta}^{(i)T} \underline{x}$ that's largest
(i: class)

Logistic Regression vs. SVM's:

⇒ If n too large (relative to m), use log. reg. or SVM w/ linear kernel

⇒ If n small, m intermediate → SVM w/ Gaussian
(1-1000) (10-10,000)

⇒ If n small, m large → add more features, use log. reg. or SVM w/ lin. kernel
(1-1000) (50,000+)

⇒ Neural Networks work well for most of these ↑, but slower to train

↳ SVM's convex, so global mins always found (unlike N.N.'s)