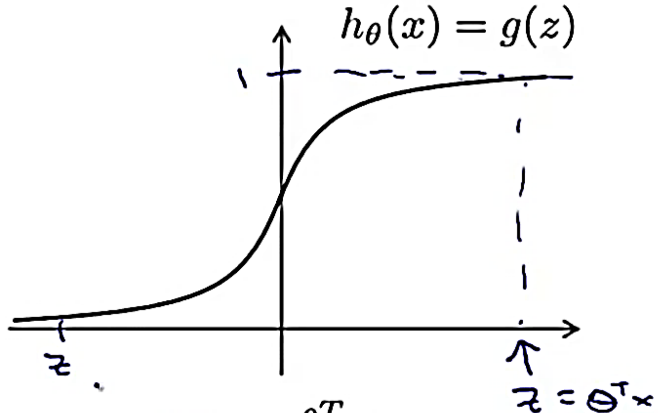
12. Support Vector Machines – Supervised Leaning Algorithm

(Like linear/logistic regression and neural networks)

Alternative view of logistic regression

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



$\rightarrow z = \theta^T x$

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

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

Alternative view of logistic regression

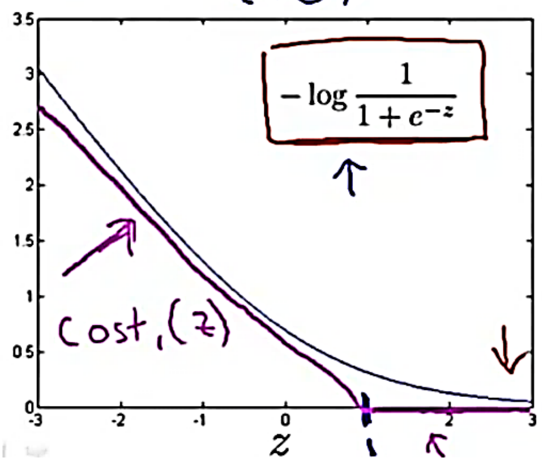
(x, y)

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

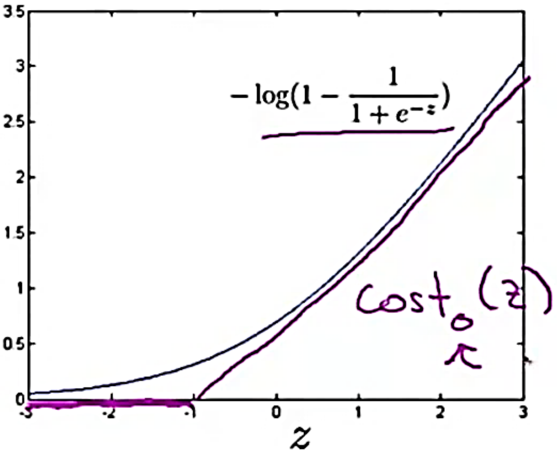
$$= \boxed{-y \log \frac{1}{1+e^{-\theta^T x}}} - \boxed{(1-y)\log\left(1-\frac{1}{1+e^{-\theta^T x}}\right)}$$

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

$z = \theta^T x$



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



These lines in magenta are the cost fxns modified such that they are just in the form of straight lines… which are very close to the Cost function curves

Support vector machine

Logistic regression:

$$\min_\theta \underbrace{\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \underbrace{\left(-\log h_\theta(x^{(i)})\right)}_{cost_1(\theta^T x^{(i)})} + (1-y^{(i)}) \underbrace{\left((-\log(1-h_\theta(x^{(i)})))\right)}_{cost_0(\theta^T x^{(i)})}\right]}_{A} + \frac{\lambda}{2m}\sum_{j}$$

Support vector machine:

It can be written as:

Support vector machine:

$$\min_\theta \cancel{\frac{1}{m}} \sum_{i=1}^{m} y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)}) + \frac{\lambda}{2\cancel{m}} \sum_{j=0}^{n} \theta_j^2$$

We can get rid of **1/m** term as it **doesn’t affect** the value of minimum Θ, which gives us

Cost Function:

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

Hypothesis:

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

So, SVM hypothesis outputs **1** directly if **z >= 0** and **0 if z < 0**

Note:

$$A + \lambda B$$

$$C A + B$$

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

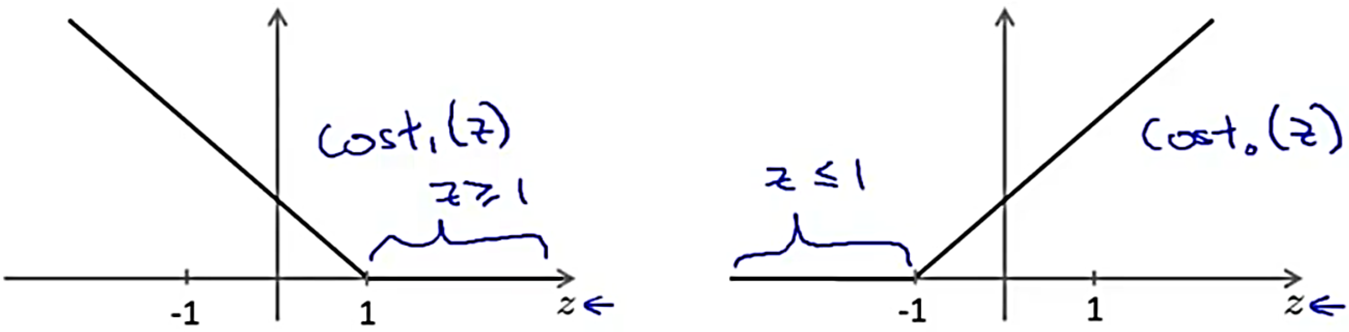
In cost function, we use λ as the parameter to minimize the values of Θ, but it can also be written as C.A + B… this just makes the same effect if C=1/λ …

We use an acceptably large value of λ .. so we can use an equivalrent small value of C to make the same effect and obtain the same Θ

LARGE MARGIN INTUITION:

Support Vector Machine

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



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

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

Here we want **Θ^{T}x** to be **>=1** for **positive** examples as safety margins

Similar for negative examples

Now, if we set C to be a very large value, say 100,000:

Then our minimization algo will technically vanish the term multiplied with C

SVM Decision Boundary

$$\min_{\theta} C \underbrace{\sum_{i=1}^{m} \left[y^{(i)} cost_1(\theta^T x^{(i)}) + (1 - y^{(i)}) cost_0(\theta^T x^{(i)}) \right]}_{=0} + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

So, we minimize regularization term subject to:

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

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

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

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

$$\min \cancel{C \times 0} + \frac{1}{2} \sum_{i=1}^{n} \theta_j^2$$

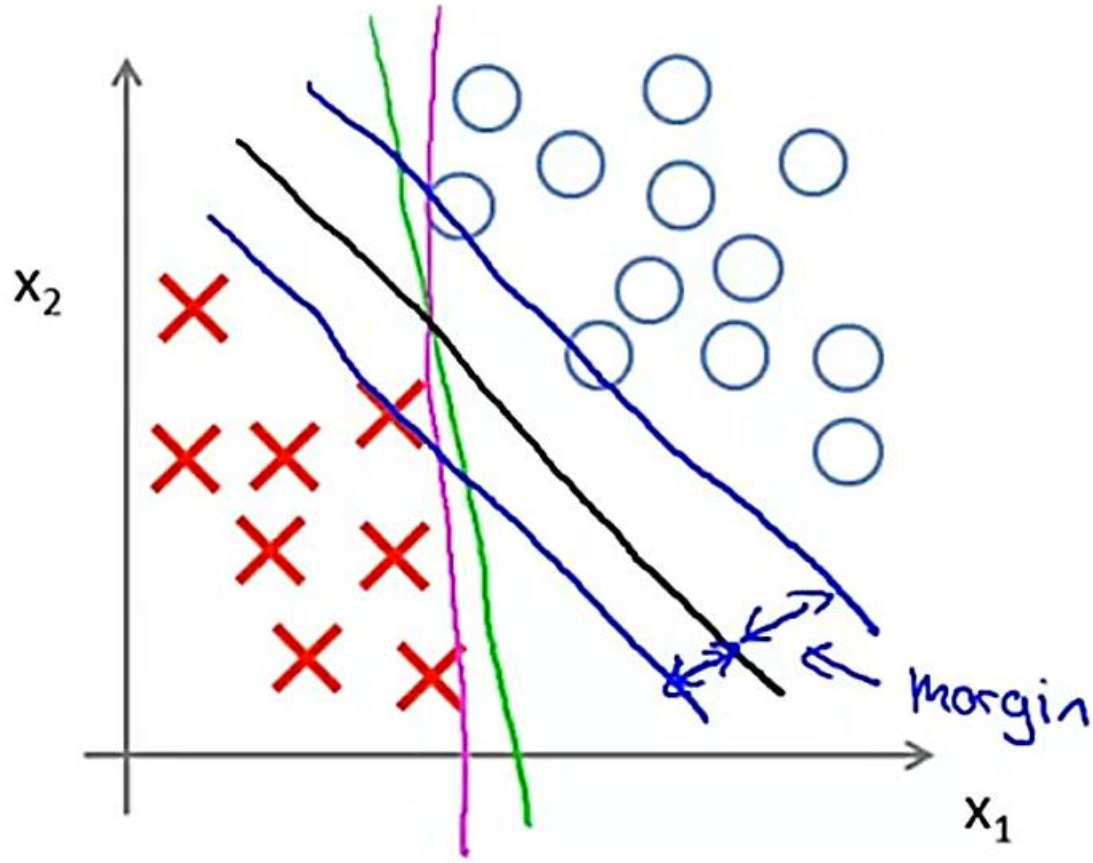
$$\text{s.t.} \quad \theta^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$$

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

This bring out a very interesting decision boundary: SVM DECISION BOUNDARY

SVM Decision Boundary: Linearly separable case

SVM give the best decision boundary (black one), which is at a minimum distance from both positive and negative examples



Large margin classifier

LARGE MARGIN OCCURS WHEN WE CHOOSE "C" – A VERY LARGE VALUE

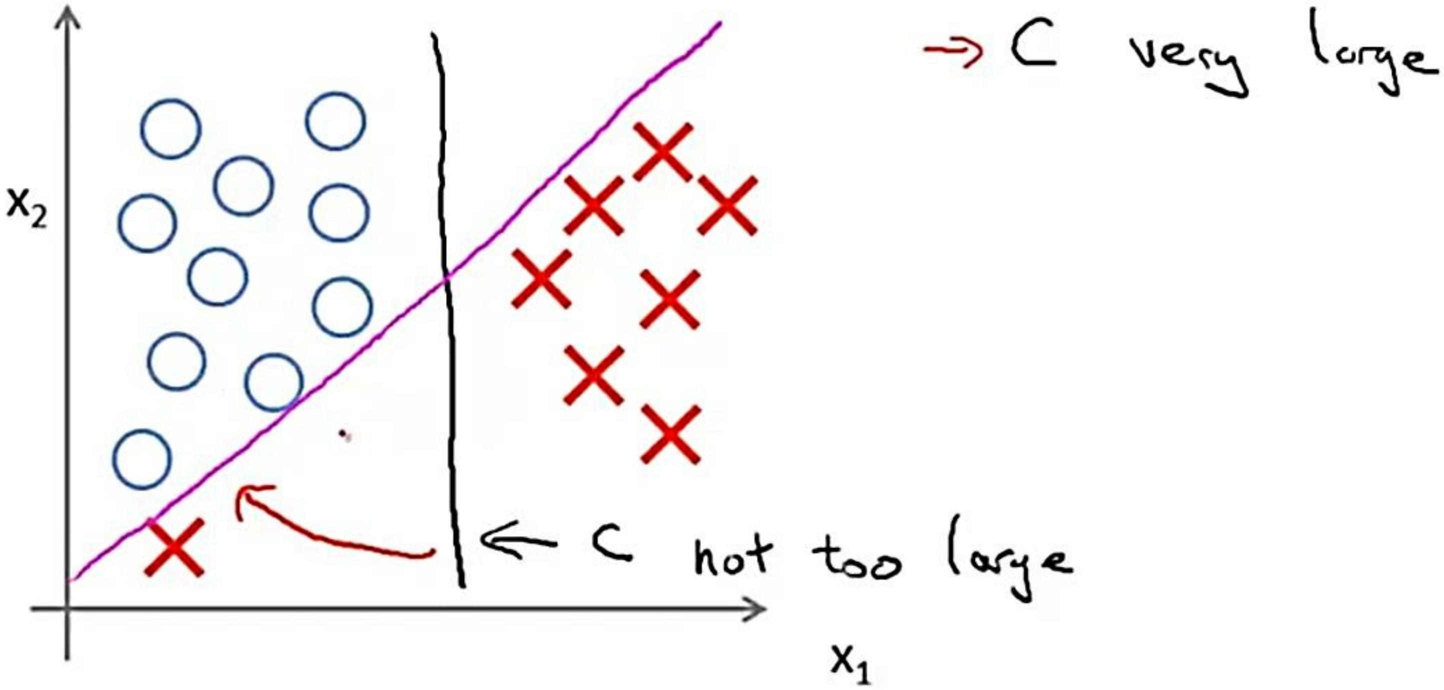
Large margin classifier in presence of outliers

If C is very large, SVM will not act as a large margin classifier:

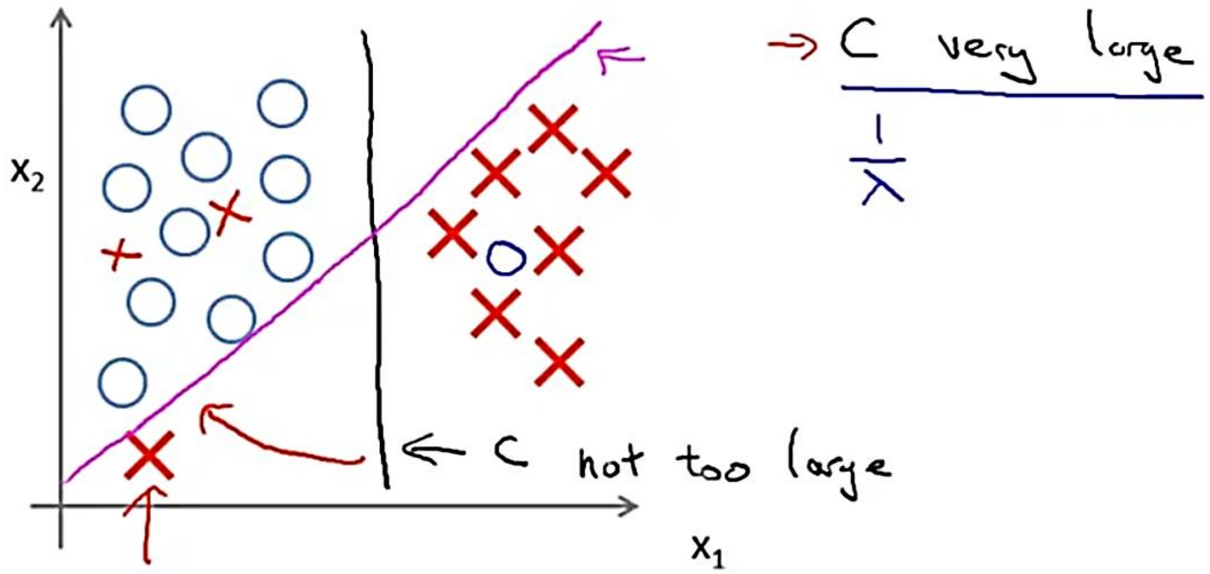
Instead it will give a closer margin.

In case of a few outliers, if C is very large: we will get magenta line

While if C is (large but) not too large: we will still get black line



In case, if the data is not linearly separable *<u>OR</u>* there are more outliers: Choosing a value of C (large but) not too large, SVM will still do the right thing, i.e., it will give the black line



MATH BEHIND LARGE MARGIN CLASSIFICATION:

Vector Inner Product:

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

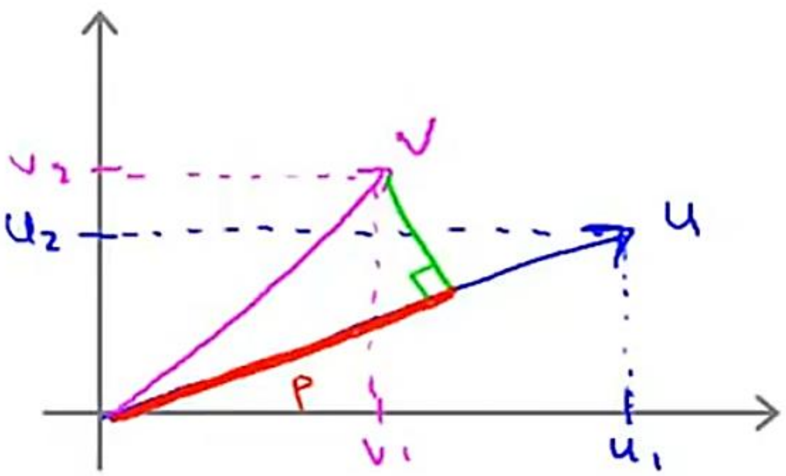
So, **we need to find:**

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

We have:

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

Projections:



Here:

p = length of projection of v onto u.

➔ P is signed, it can be **+ve or -ve**

So, we get:

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

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

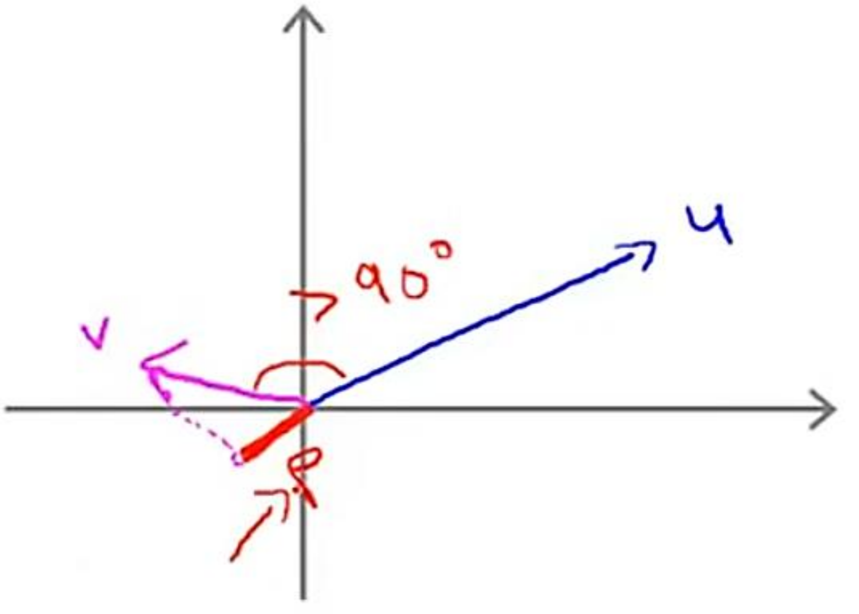
Matrix representation:

$u^T v$ =

$$\begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Since p is signed:

If $p < 0$:



Optimization objective of SVM: Math behind it:

Why SVM gives large margin classification?

For simplification we take $\Theta_0 = 0$ and $n = 2$:

$\Theta_0 = 0$ → so that Θ vector passes through origin.

N=2 → only two features in the data

SVM Decision Boundary

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

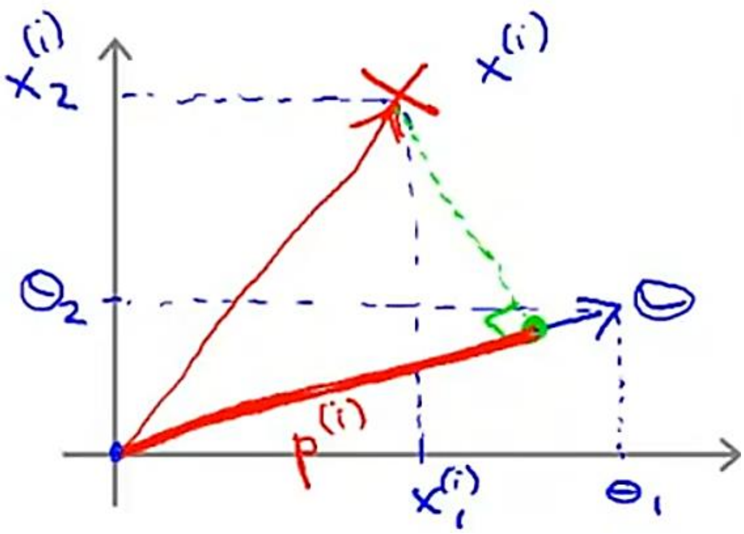
s.t. $\boxed{\theta^T x^{(i)} \geq 1}$ if $y^{(i)} = 1$

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

Simplication: $\theta_0 = 0$. $n=2$

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

So, to find $\Theta^T x$:



Therefore:

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

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

This gives us:

SVM Decision Boundary

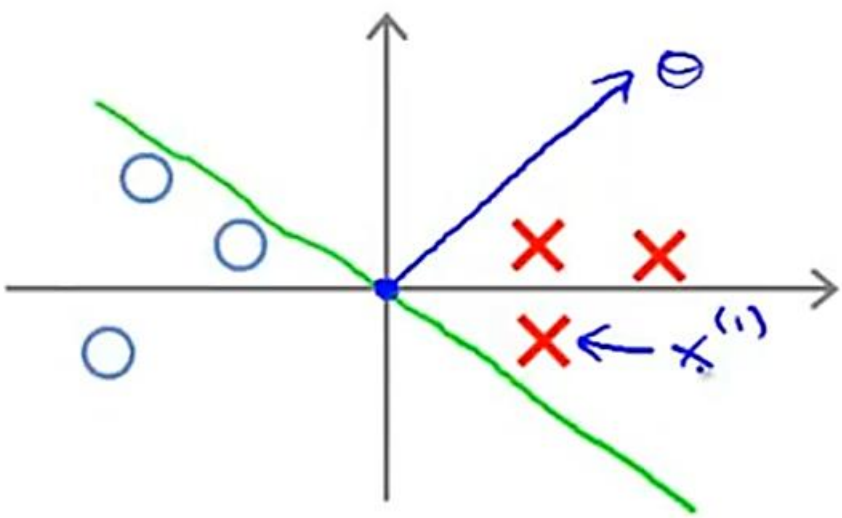
$$\min_{\theta} \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 = \frac{1}{2}\|\theta\|^2$$

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

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

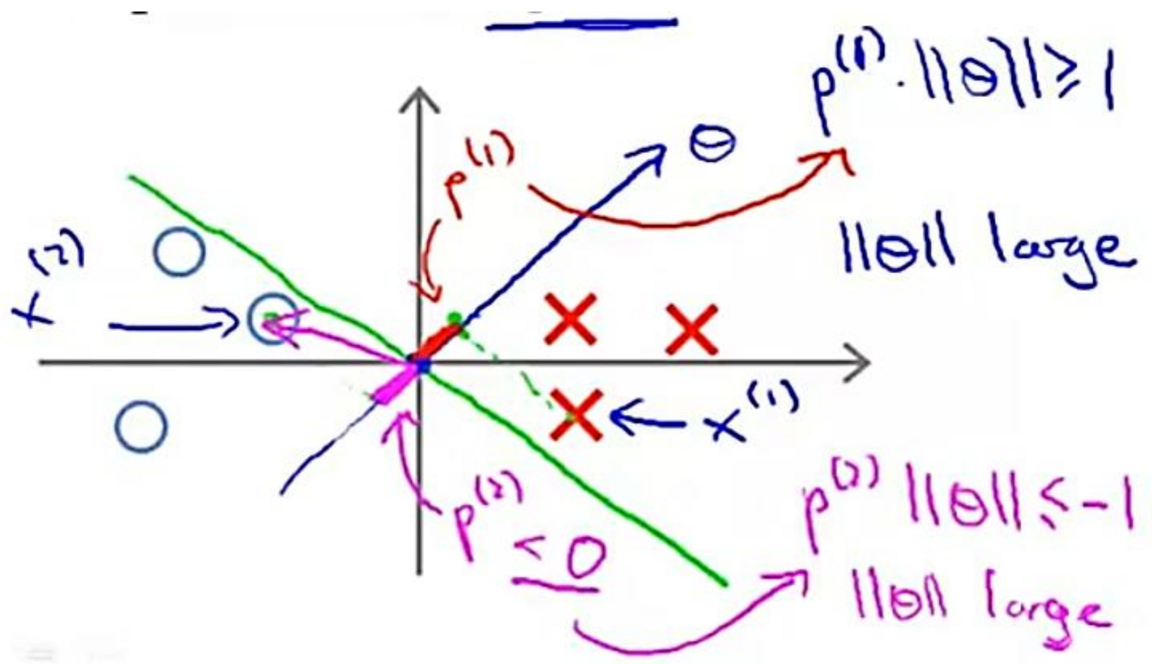
C very large

Let's consider the case of **small margin decision boundary**: it's not a very good choice though:



- Θ vector will be perpendicular to decision boundary as we can recall that **decision boundary does not depend on parameters or hypothesis:** it **only depends on features.**

Now lets plot the examples on this decision boundary:

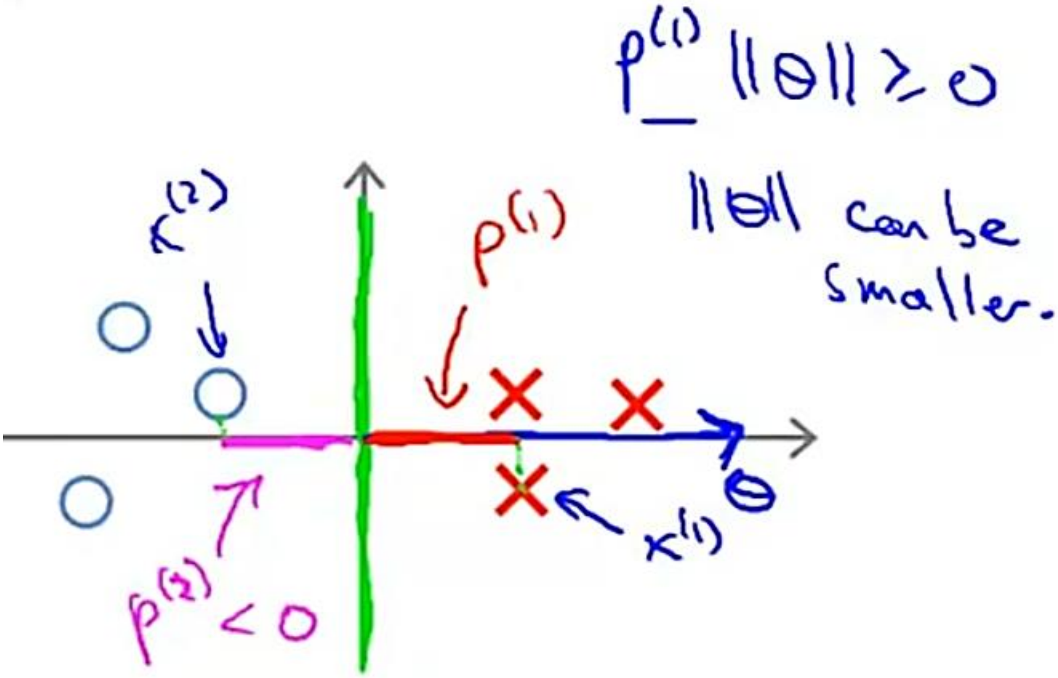


This means that, since **$p^{(1)}$ and $p^{(2)}$ are small,** for **p(1)||Θ|| to be greater than 1:**

||Θ|| will have to be large.

But this contradicts our cost minimization efforts, So, this decision boundary is not the chosen

Now suppose a large margin decision boundary is chosen:



Since $p^{(1)}$ is larger, Θ can be smaller, which supports are norm of minimizing Θ in cost function's regularization part.

- ➢ The values of margin is equal to the values of p for given example

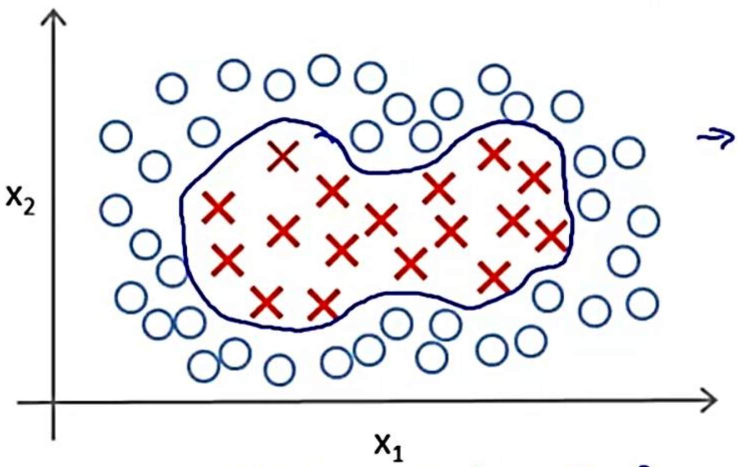
---

KERNELS:

To write complex **non-linear classifiers**

Usually what we do is:

Non-linear Decision Boundary



Predict $y = 1$ if

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$$

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

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

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

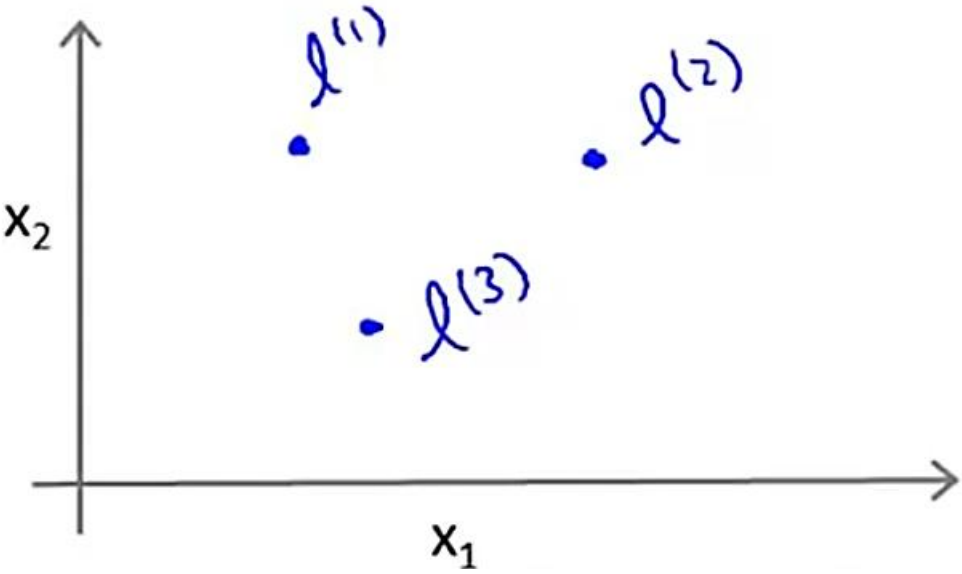
This is how we define features for our hypothesis.

But:

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

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

Kernel



Landmarks are some vectors on the feature vs feature graph

x_1

Given x:

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

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

$$f_3 = \text{similarity}(x, \ell^{(3)}) = \exp(\ldots)$$

Kernel (Gaussian Kernels) $\quad k(x, \ell^{(i)})$

Similarity can also be written as:

$$k(x, \ell^{(i)})$$

- $\|x - \ell^{(1)}\|$ is the component wise difference bw vector x and vector l.

What are kernels:

Kernels and Similarity

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

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

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

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

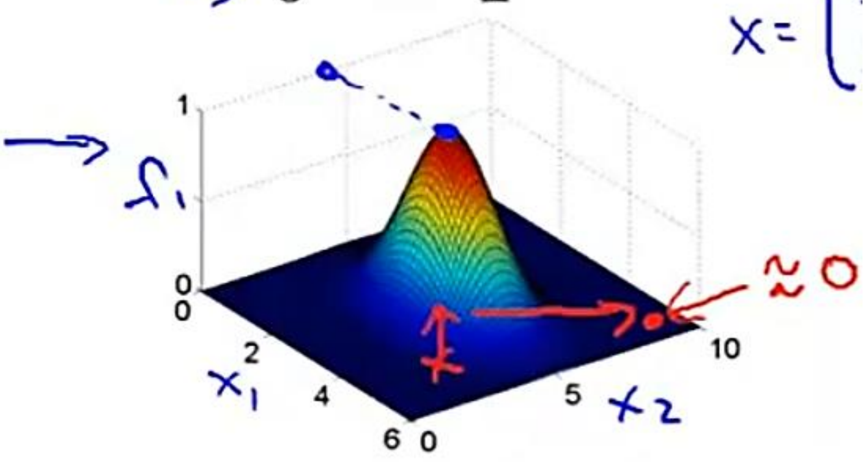
$$f_1 = \exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \approx 0.$$

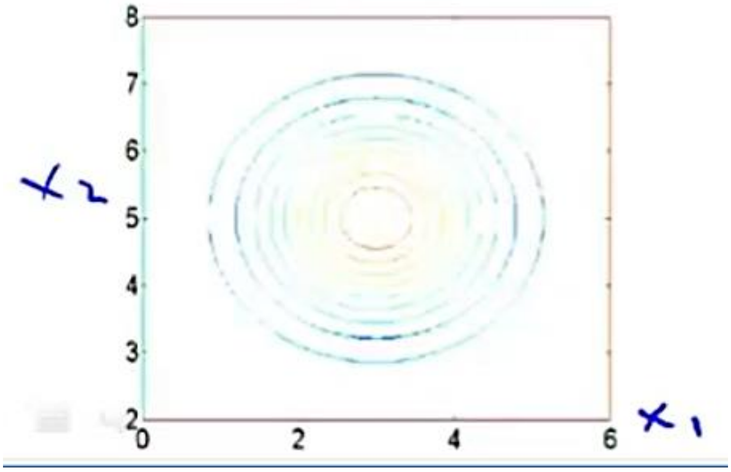
Example:

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

Lets try diff values of σ :

$\sigma^2 = 1$

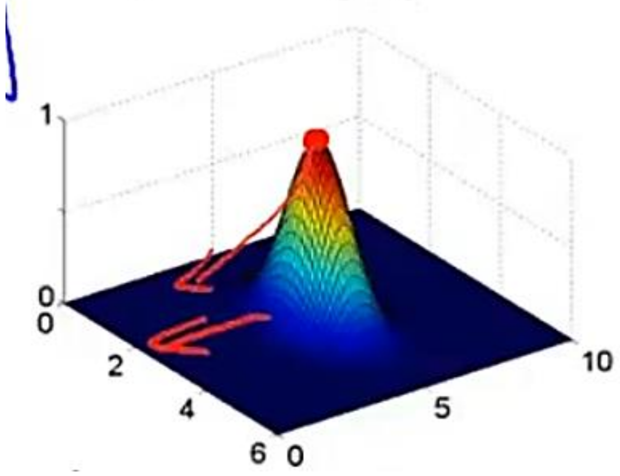


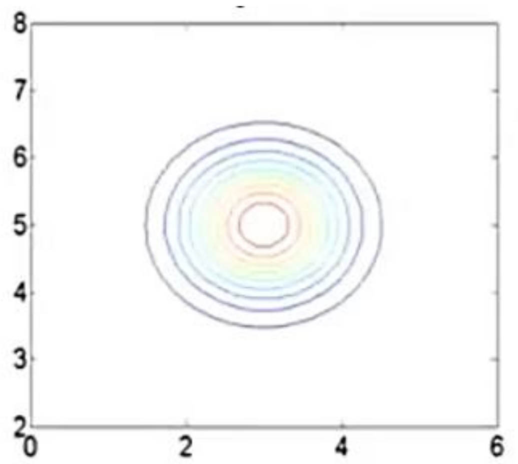


If we decr σ:

More values of x will be close to 0.

$$\sigma^2 = 0.5$$

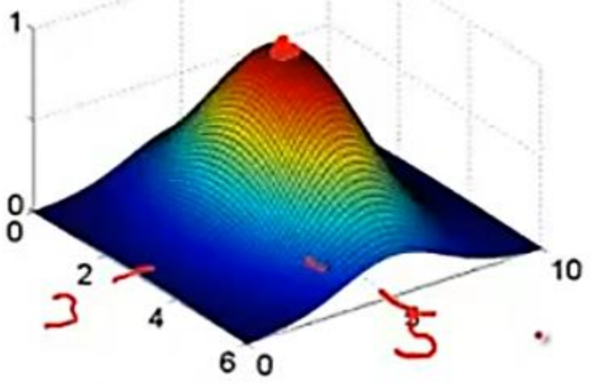


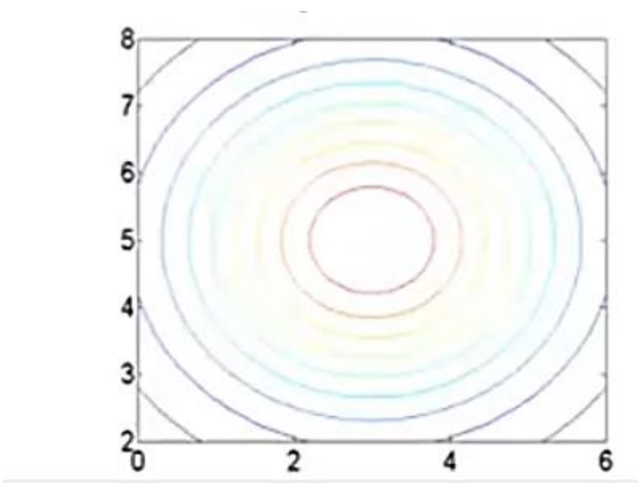


If we incr σ :

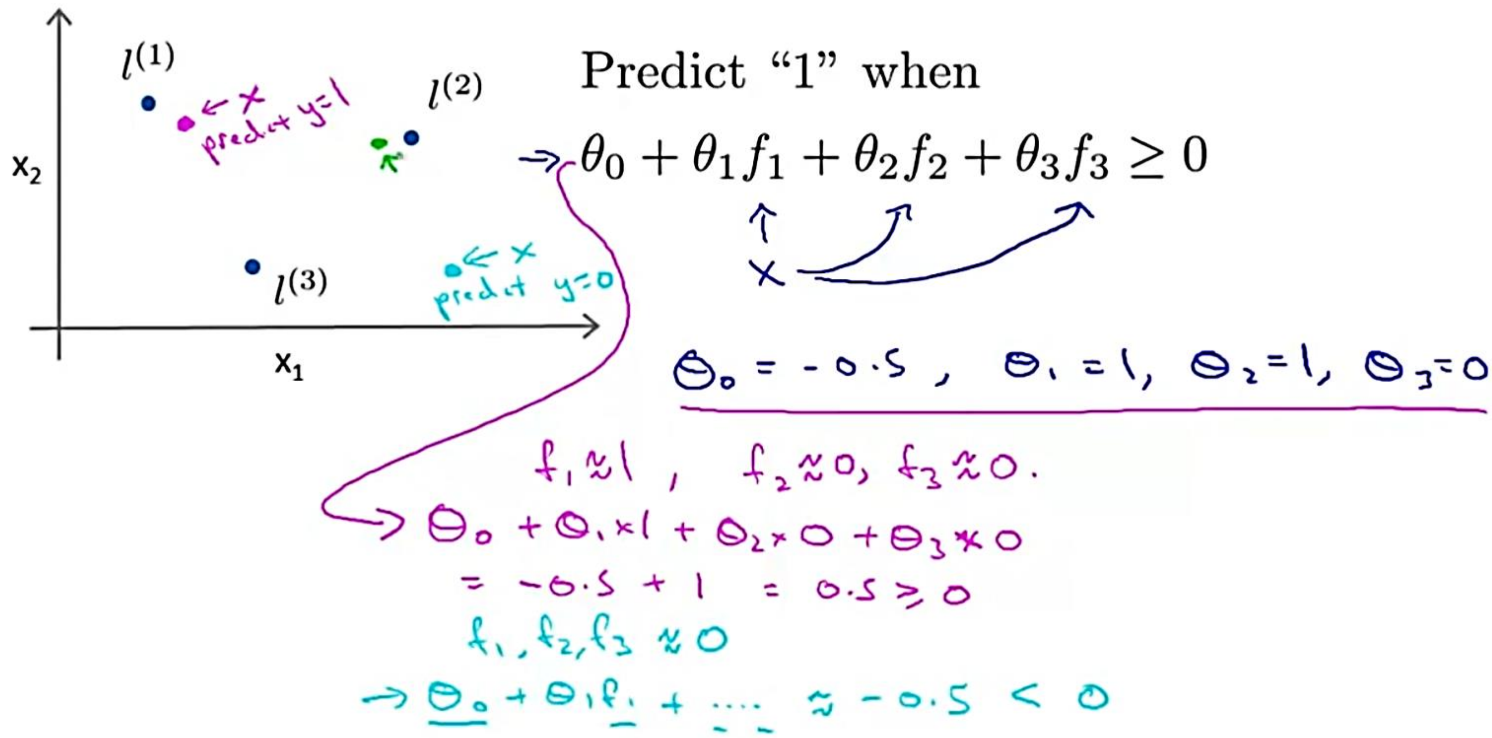
More values of x will be higher than 0.

$$\sigma^2 = 3$$





Example:



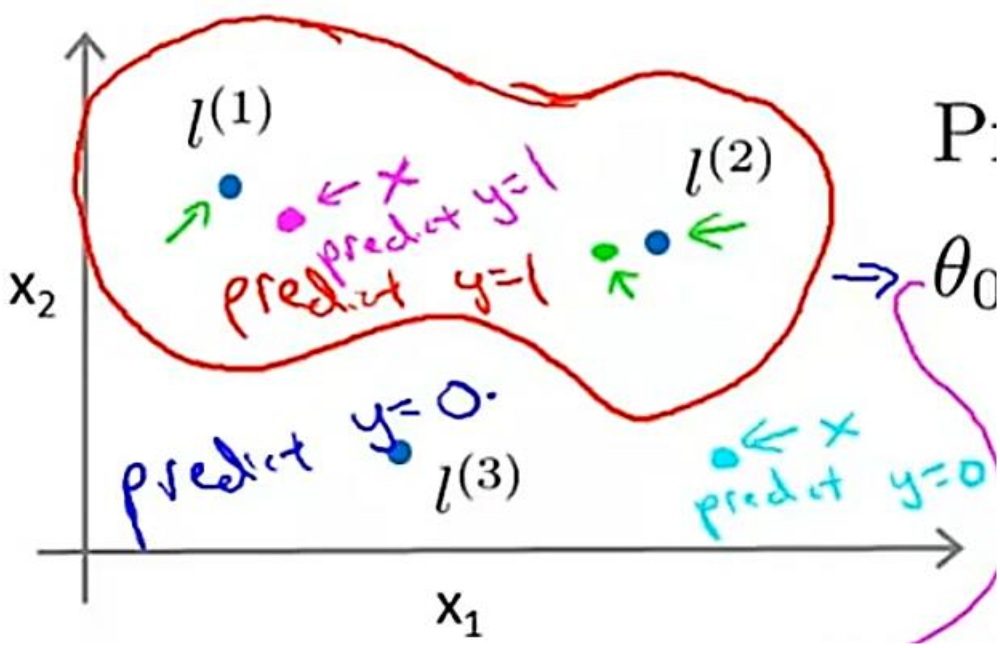
Here, pink point is the example close to landmark 1:

⇨**So the hypothesis o/p is greater than 0** →hence prediction is 1

Blue point is far from all landmarks

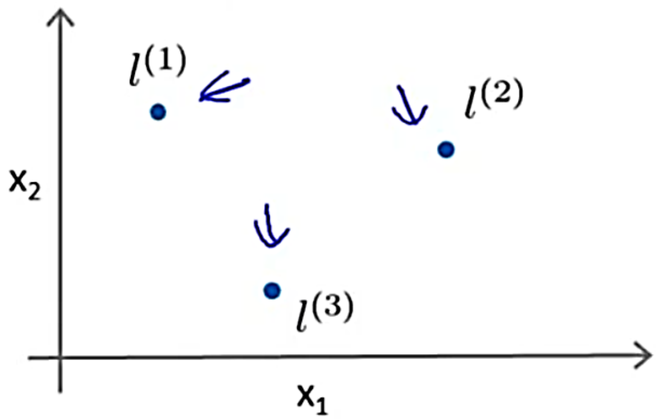
⇨**So the hypot. o/p is less than 0** → hence the prediction is 0

So this gives us a non-linear decision boundary:



HOW TO CHOOSE LANDMARKS:

Choosing the landmarks



Given x:

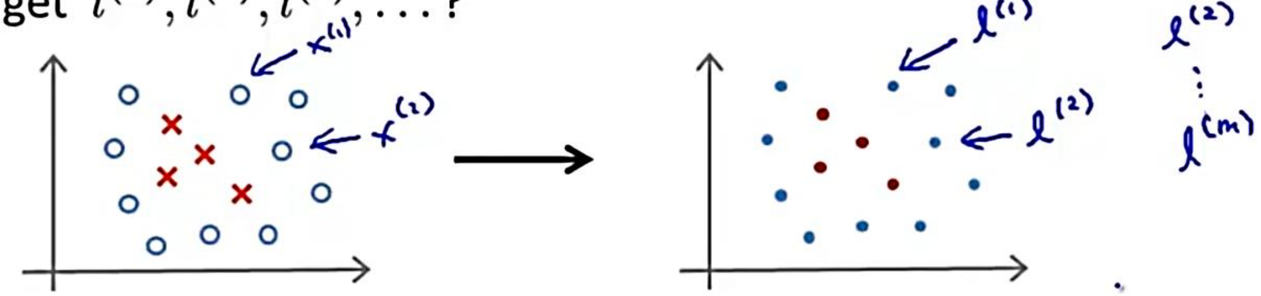
$$\rightarrow f_i = \text{similarity}(x, l^{(i)})$$
$$= \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right) \leftarrow$$

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

We choose landmarks at exactly the same points as our examples:

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

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



→→

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

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

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

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

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

Andrew

We will obtain a feature vector f from x.

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

→ Predict "y=1" if $\theta^T f \geq 0$ $\qquad \theta_0 f_0 + \theta_1 f_1 + \dots + \theta_m f_m$

$n = m$

Training:

$$\rightarrow \min_\theta C \sum_{i=1}^{m} y^{(i)} cost_1(\theta^T f^{(i)}) + (1-y^{(i)}) cost_0(\theta^T f^{(i)}) + \frac{1}{2}\sum_{j=1}^{\cancel{n} = m} \theta_j^2$$

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

$$\sum_j \theta_j^2 = \theta^T \theta \leftarrow \theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_m \end{bmatrix} \quad (\text{ignore } \theta_0)$$

$\|\theta\|^2$

$\theta^T M \theta$

For implementation purpose, a **scaling factor M** is used while calculating **Θ²,** to counter the expense of large training sets

---

HOW TO CHOOSE PARAMETER "C":

SVM parameters:

C ($= \frac{1}{\lambda}$). → Large C: Lower bias, high variance. (small λ)

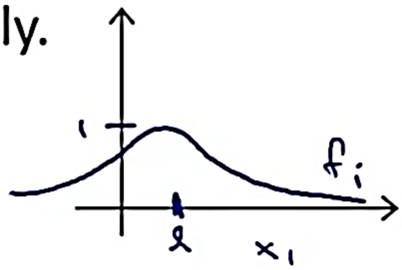
→ Small C: Higher bias, low variance. (large λ)

HOW TO CHOOSE σ^2 :

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

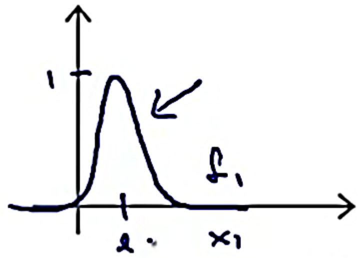
→ Higher bias, lower variance.

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



Small σ^2: Features f_i vary less smoothly.

Lower bias, higher variance.



Andrew N

HOW TO USE SUPPORT VECTOR MACHINES:

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

$x \in \mathbb{R}^{n+1}$

OR:

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.

$x \in \mathbb{R}^n$, n small

and/or m large

- In case of Gaussian Kernel:

Kernel (similarity) functions: $x^{(i)}$ $l^{(j)} = x^{(j)}$

f_i

```
function f = kernel(x1, x2)
```

$$f = \exp\left(-\frac{\|\mathbf{x1} - \mathbf{x2}\|^2}{2\sigma^2}\right)$$

```
return
```

$x \rightarrow f_1, f_2, \ldots, f_m$

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

$x \in \mathbb{R}^n$

$\boxed{\|x - l\|^2}$ $\quad v = x - l$

$\|v\|^2 = v_1^2 + v_2^2 + \cdots + v_n^2$

$= \underbrace{(x_1 - l_1)^2}_{1000 \text{ feet}^2} + \underbrace{(x_2 - l_2)^2}_{1\text{-}5 \text{ bedrooms}} + \cdots + (x_n - l_n)^2$.

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: $k(x, l) = (x^T l)^2 + 0$ $\quad (x^T l + \text{constant})^{\text{degree}}$

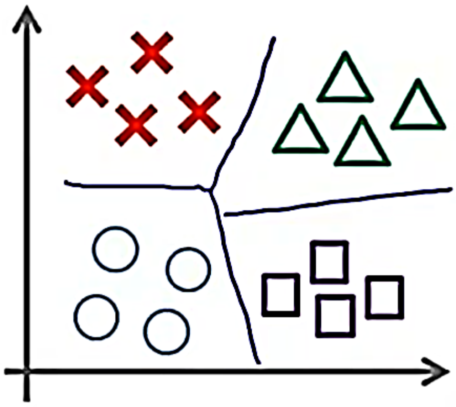
  $(x^T l)^3, \quad (x^T l + 1)^3, \quad (x^T l + 5)^4$

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

  $\text{sim}(x, l)$

➔ **Polynomial kernels** are usually (although very rarely) used when x and l are positive

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

$y=1$ $\quad$ $y=2$ $\quad \ldots \quad$ $\theta = K$

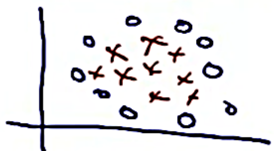
WHEN TO USE LOGISTIC REGRESSION vs SVM ?

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. $n \geq m$, $n = 10{,}000$, $m = 10 \ldots 1000$)
- 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.
