

12. SUPPORT VECTOR MACHINES

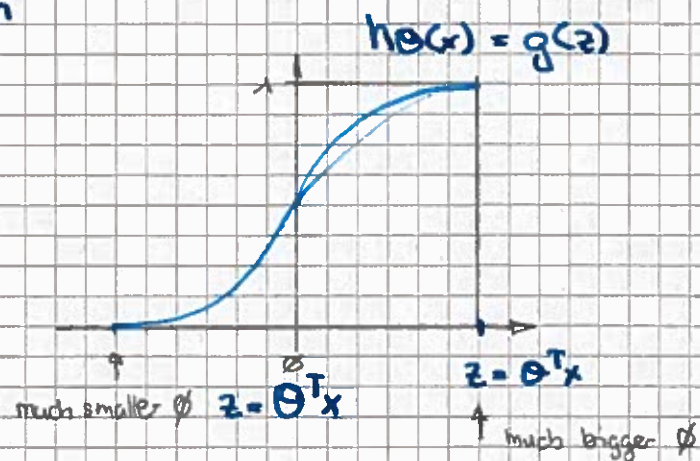
Optimization Objective:

Alternative view of logistic regression

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

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

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

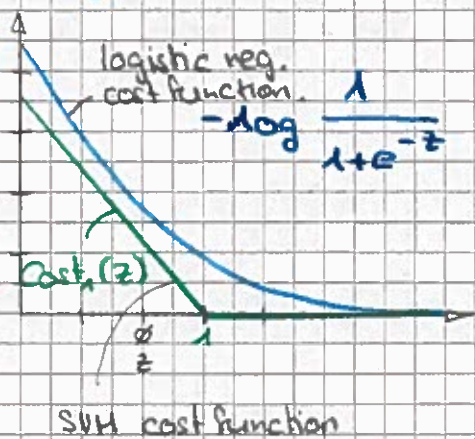


Cost Function: $-(y \log h_{\theta}(x) + (1-y) \log (1 - h_{\theta}(x)))$

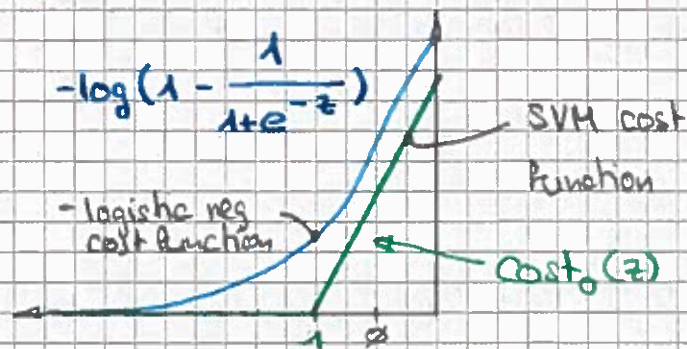
$$= -y \log \frac{1}{1 + e^{-\theta^T x}} - (1-y) \log \left(1 - \frac{1}{1 + e^{-\theta^T x}} \right)$$

for single example

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

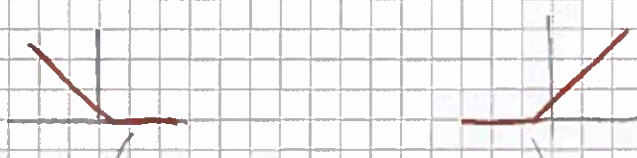


If $y=0$ (want $\theta^T x \ll 0$)



Support Vector Machine:

Logistic Regression:



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

Support Vector Machine:

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

$$\Rightarrow \begin{aligned} &A + \lambda B \\ &CA + B \\ &C = \frac{1}{\lambda} \end{aligned}$$

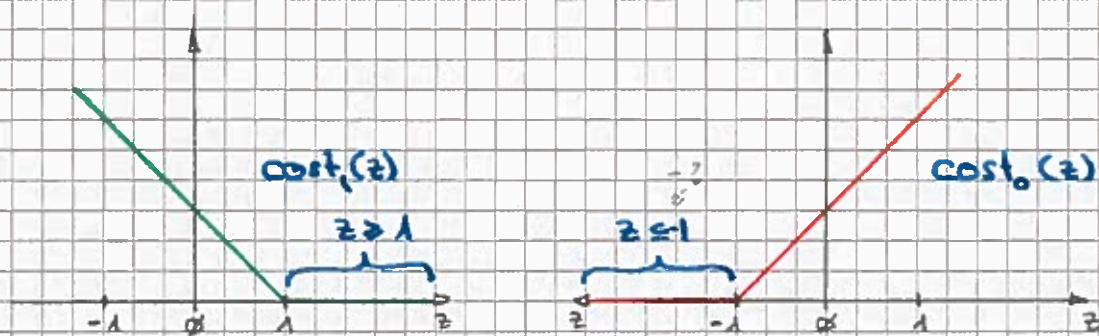
\Rightarrow is basically the same if you want to control the regularization Param. Either give λ more weight or reduce the size of C to give the regularization more weight.

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

SVH Hypothesis:

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

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



if $y = 1 \rightarrow \Theta^T x \geq 1$ (not just ≥ 0)

if $y = 0 \rightarrow \Theta^T x \leq -1$ (not just ≤ 0)

SVM decision Boundary

assume to be $C = 100,000$

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

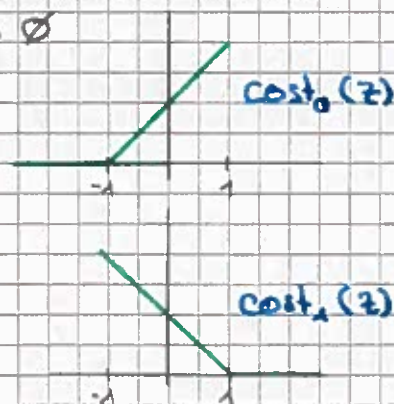
try to get this to 0

whenever $y^{(i)} = 1$

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

whenever $y^{(i)} = 0$

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

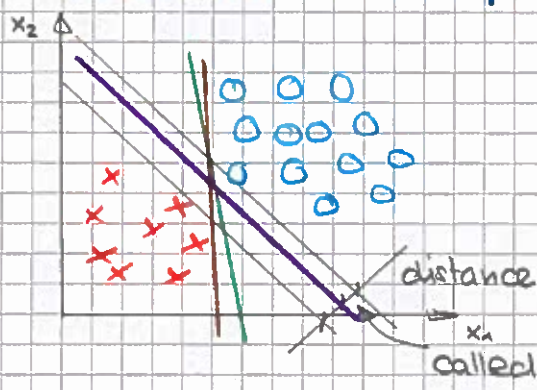


$$\min_{\Theta} C = 0 + \frac{1}{2} \sum_{j=1}^n \Theta_j^2$$

subject to $\Theta^T x^{(i)} \geq 1$ if $y^{(i)} = 1$

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

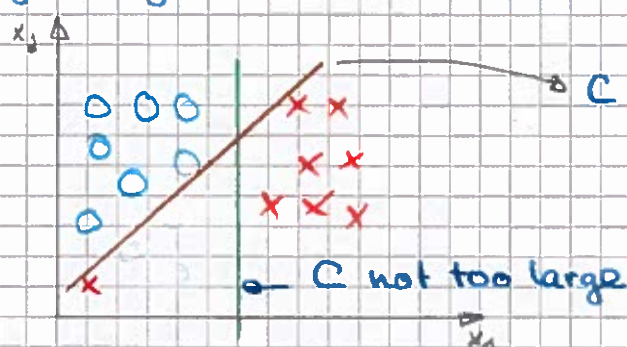
SVM Decision Boundary: linearly separated case



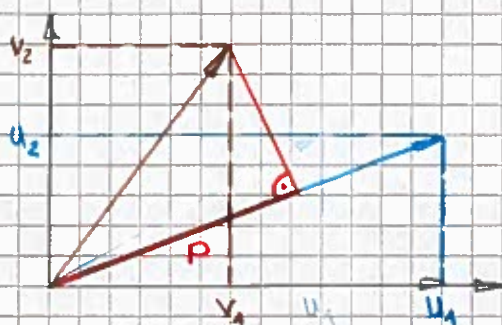
\Rightarrow gives the SVM a robustness

large margin classifier

large margin classifier in presence of outliers



C very large which is not so good
 \hookrightarrow equivalent to $\frac{1}{\lambda}$ $C = \frac{1}{\lambda}$

Vector inner Product:

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

$$u^T v = ?$$

$\|u\|$ = length of vector u

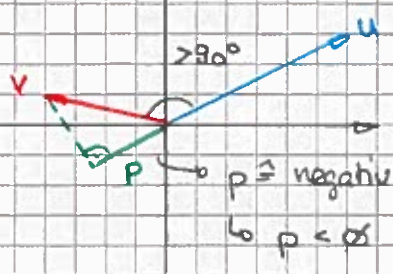
$$= \sqrt{u_1^2 + u_2^2} \in \mathbb{R}$$

p = length of projection of v onto u

signed

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

$$\Rightarrow \begin{bmatrix} u_1 & u_2 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = u_1 v_1 + u_2 v_2 \quad p \in \mathbb{R}$$



$p \hat{=}$ negativ
 $\hookrightarrow p < 0$

SVM Decision Boundary:

$$\min_{\Theta} \frac{1}{2} \sum_{i=1}^n \Theta_i^2 \Rightarrow \frac{1}{2} (\Theta_1^2 + \Theta_2^2) = \frac{1}{2} \left(\sqrt{\Theta_1^2 + \Theta_2^2} \right)^2 = \frac{1}{2} \|\Theta\|^2$$

$$\hookrightarrow \Theta^T x^{(i)} \geq 1 \quad \text{if } y^{(i)} = 1$$

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

for simplification: $\Theta_0 = 0 \quad n=2$

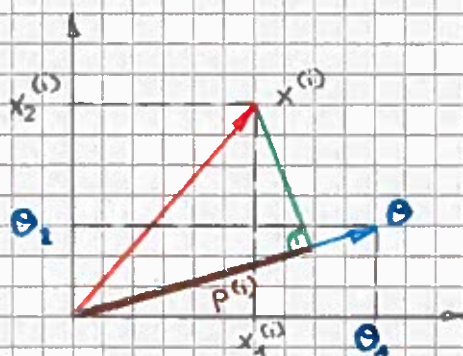
$\|\Theta\|$ = vector u above

$$\begin{bmatrix} \Theta_1 \\ \Theta_2 \end{bmatrix}$$

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

$$\uparrow \quad \uparrow$$

$$u^T v$$



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

$$= \Theta_1 x_1^{(i)} + \Theta_2 x_2^{(i)}$$

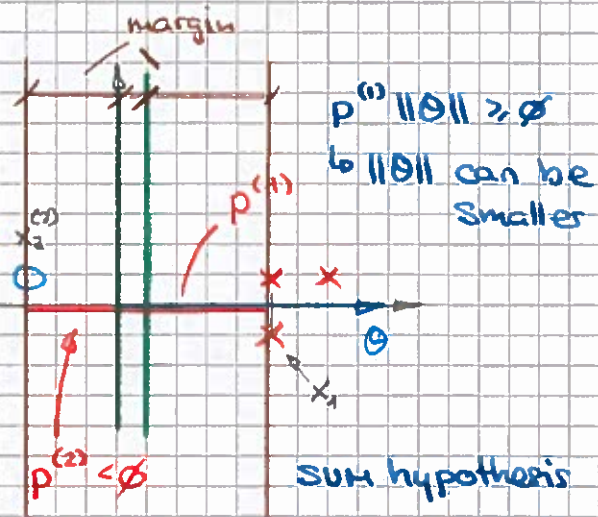
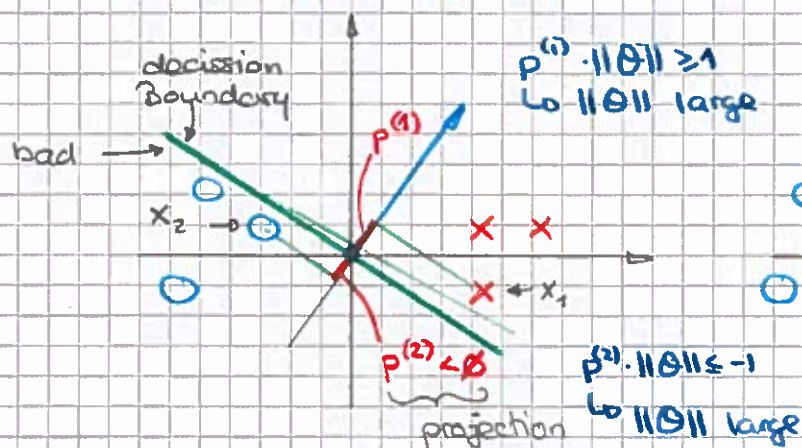
$$\min_{\theta} \frac{1}{2} \sum_{j=1}^n \theta_j^2 = \frac{1}{2} \|\theta\|^2$$

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

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

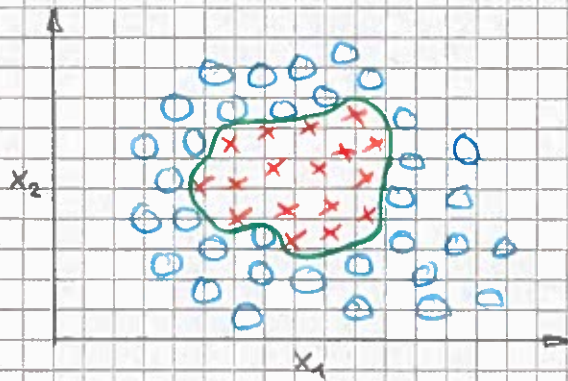
where $p^{(i)}$ is the projection of $x^{(i)}$ onto the vector θ .

Simplification: $\theta_0 = 0$



\Rightarrow need to find a value for θ close to 0

Non Linear Decision Boundaries:

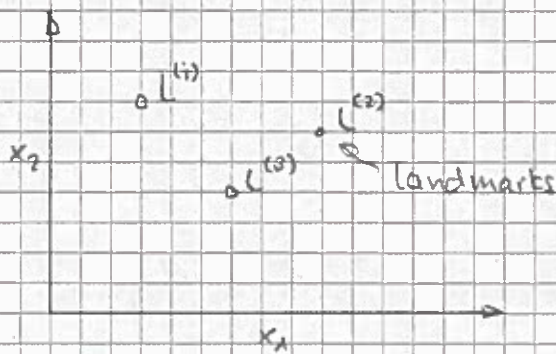
Predict $y=1$ if

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

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

$$\theta_0 + \theta_1 F_1 + \theta_2 F_2 + \theta_3 F_3 + \dots$$

$$F_1 = x_1, F_2 = x_2, F_3 = x_1 x_2 \dots$$

Is there a different / better way of features $F_1, F_2, F_3 \dots$?

Given new features

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

Given x :

$$F_1 = \text{similarity}(x, L^{(1)}) = \exp\left(-\frac{\|x - L^{(1)}\|^2}{2\sigma^2}\right)$$

$$F_2 = \text{similarity}(x, L^{(2)}) = \exp\left(-\frac{\|x - L^{(2)}\|^2}{2\sigma^2}\right)$$

$$F_3 = \dots$$

Euclidian distance

Gaussian kernel (kernel) $= \phi k(x, L^{(i)})$

Kernels and Similarity:

$$F_1 = \text{similarity}(x, L^{(1)}) = \exp\left(-\frac{\|x - L^{(1)}\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{i=1}^n (x_i - L_i^{(1)})^2}{2\sigma^2}\right)$$

If $x \approx L^{(1)}$

$$F_1 \approx \exp\left(-\frac{\sigma^2}{2\sigma^2}\right) \approx 1 \Rightarrow e^{-0.5}$$

If x is far from $L^{(1)}$:

$$F_1 = \exp\left(-\frac{(\text{large number})^2}{2\sigma^2}\right) \approx 0 \Rightarrow e^{-x}$$

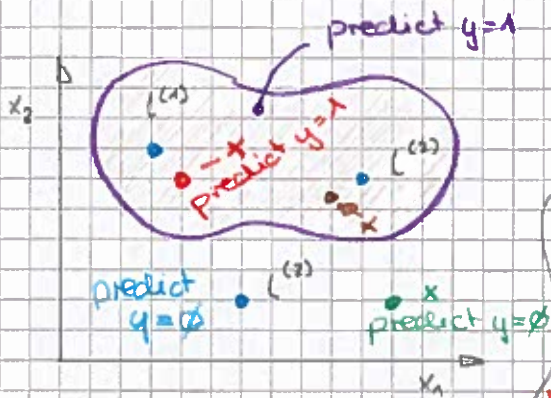
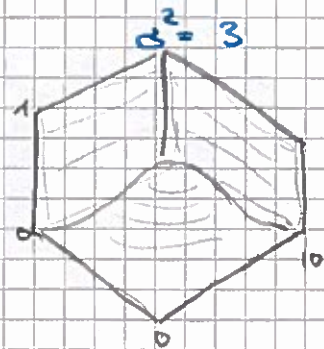
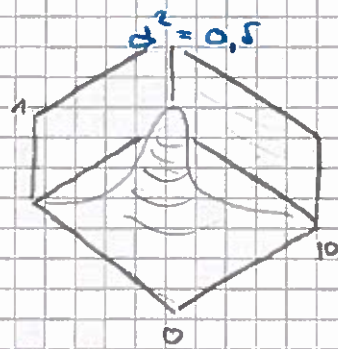
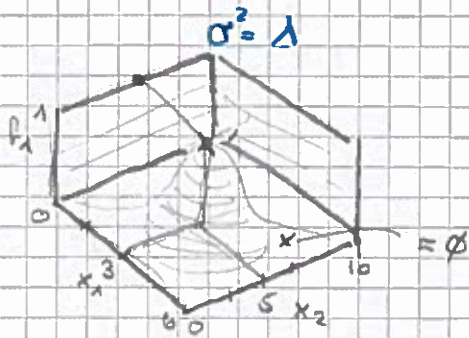
$$\begin{aligned} L^{(1)} &\rightarrow F_1 \\ L^{(2)} &\rightarrow F_2 \\ L^{(3)} &\rightarrow F_3 \end{aligned}$$

each landmark creates new feature

σ = standard deviation
 σ^2 = variance

Example:

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



Predict "1" when

$$\Theta_0 + \Theta_1 f_1 + \Theta_2 f_2 + \Theta_3 f_3 \geq \phi$$

$$\Theta_0 = -0.5, \Theta_1 = 1, \Theta_2 = 1, \Theta_3 = \phi$$

if x is close to $L^{(1)}$

$$f_1 \approx 1, f_2 \approx \phi, f_3 \approx \phi$$

$$\Rightarrow \Theta_0 + \Theta_1 \cdot 1 + \Theta_2 \cdot \phi + \Theta_3 \cdot \phi$$

$$\Leftrightarrow -0.5 + 1 = 0.5 \geq \phi$$

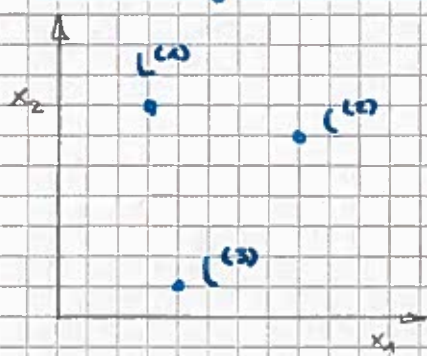
if x is far away from $L^{(1)}$

$$f_1, f_2, f_3 \approx \phi$$

$$\Theta_0 + \Theta_1 f_1 + \Theta_2 f_2 + \Theta_3 f_3 \approx -0.5 < \phi$$

$$-0.5 + \phi + \phi + \phi = -0.5$$

Choosing the landmarks:

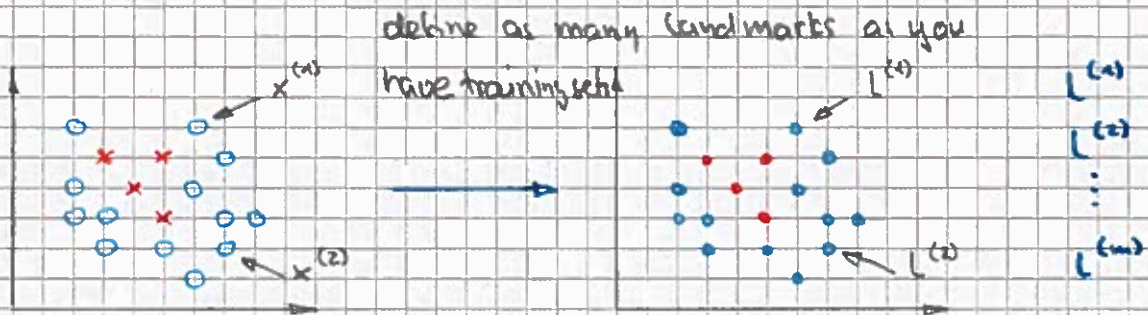


Given x :

$$P_i = \text{similarity}(x, L^{(i)})$$

$$= \exp\left(-\frac{\|x - L^{(i)}\|^2}{2\sigma^2}\right) \quad \left. \begin{array}{l} \text{Gaussian} \\ \text{kernel} \end{array} \right\}$$

Predict $y=1$ if $\Theta_0 + \Theta_1 P_1 + \Theta_2 P_2 + \Theta_3 P_3 \geq 0$
where to get $L^{(1)}, L^{(2)}, L^{(3)}$



SUM 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 x : can be training, cv or test set

$$\begin{aligned} P_1 &= \text{similarity}(x, L^{(1)}) \\ P_2 &= \text{similarity}(x, L^{(2)}) \\ &\vdots \end{aligned}$$

$$P = \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ \vdots \\ P_m \end{bmatrix} \quad P_0 = 1$$

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

$$\begin{aligned} P_1^{(i)} &= \text{sim}(x^{(i)}, L^{(1)}) \\ P_2^{(i)} &= \text{sim}(x^{(i)}, L^{(2)}) \\ &\vdots \\ P_i^{(i)} &= \text{sim}(x^{(i)}, L^{(i)}) = \exp\left(-\frac{0}{2\sigma^2}\right) = 1 \\ &\vdots \\ P_m^{(i)} &= \text{sim}(x^{(i)}, L^{(m)}) \end{aligned}$$

$$x^{(i)} \in \mathbb{R}^{nm} \rightarrow P^{(i)} = \begin{bmatrix} P_0^{(i)} \\ P_1^{(i)} \\ \vdots \\ P_m^{(i)} \end{bmatrix} \quad P_0^{(i)} = 1$$

SVM with kernels:

m training set \rightarrow L (width)

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

Predict "y=1" if $\underbrace{\Theta^T F}_{=0} \geq 0 \Rightarrow \Theta_0 F_0 + \Theta_1 F_1 + \dots + \Theta_m F_m \quad \Theta \in \mathbb{R}^{m+1}$

Training:

$$\min_{\Theta} C \sum_{i=1}^m y^{(i)} \text{cost}_1(\underbrace{\Theta^T F^{(i)}}_{\text{replacement of } \Theta^T x^{(i)}}) + (1-y^{(i)}) \text{cost}_0(\underbrace{\Theta^T F^{(i)}}_{\text{not regularized}}) + \frac{1}{2} \sum_{i=1}^m \Theta_i^2 \quad \Rightarrow n=m$$

$$-\sum_i \Theta_i^2 = \underbrace{\Theta^T \Theta}_{\|\Theta\|^2} \quad \Theta = \begin{bmatrix} \Theta_1 \\ \vdots \\ \Theta_m \end{bmatrix} \quad (\text{ignore } \Theta_0)$$

$\Theta^T H \Theta = 0$ used by some SVM implementation allows to much bigger Training sets

SVM parameters:

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

Large C: lower bias, high variance

Small C: Higher bias, low variance

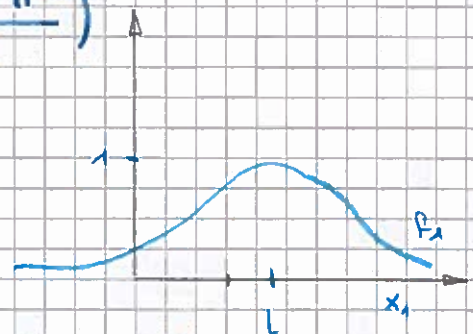
higher overfit
not much regularization
(small λ) - overfit
(large λ) - underfit
to higher bias

σ^2

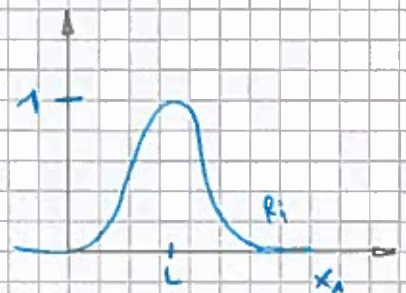
Large σ^2 : Features F_i vary more smoothly
Higher bias, lower variance

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

overfit \rightarrow increase σ^2



Small σ^2 : Features F_i vary less smoothly
Lower bias, higher variance



Using an SVM

11

Use SVM software packages (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)

\Rightarrow Standard linear classifier

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

$$\Rightarrow \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n \geq 0$$

n large, m small

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

use no kernel if above applies

\hookrightarrow risk of overfitting when using Gaussian kernel

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 n large

Kernel (similarity) function:

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

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

return

}

$$\Rightarrow x = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{pmatrix}$$

Note: Do perform feature scaling before using Gaussian kernel

$$\|x - l\|^2 \Rightarrow \overset{\text{vector}}{v} = x - l$$

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

$$= (x_1 - l_1)^2 + (x_2 - l_2)^2 + \dots + (x_n - l_n)^2$$

examp! 1000ft 1-5 bedrooms

req. feature scaling

Other choices of kernel:

Note: Not all similarity functions $\text{similarity}(x, l)$ make valid kernels. (Need to satisfy technical conditions called "Mercer's Theorem" to make sure SVM packages optimizations run correctly and do not diverge.)

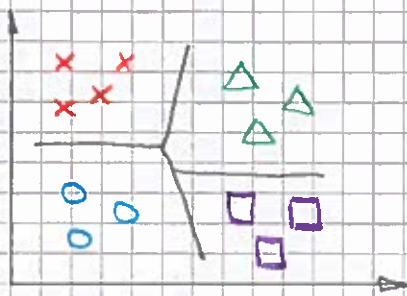
Many off-the-shelf kernels: / performs worse than Gaussian kernel

- Polynomial Kernel $\Rightarrow k(x, l) = (x^T l) = (x^T l + \text{constant})^{\text{degree}}$

$$\Rightarrow (x^T l)^2, (x^T l + 1)^3, (x^T l + 5)^4$$

- More esoteric:

String Kernel, Chi-square Kernel, histogram intersection Kernel, ...

Multi class classification:

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

Many SVM packages already have builtin multi class classification functionality.

Otherwise, use one vs. all method. (Train K SVM's, one to distinguish $y = i$ from the rest, for $i = 1, 2, \dots, k$), get $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(k)}$
Pick class i with largest $(\theta^{(i)})^T x$.

Logistic Regression vs. SVM's

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

if n is large (relative to m): (eg. $n \gg m$ $n=10'000$, $m=10 \dots 1000$)
spam mails

use logistic regression, or SVM without a kernel (linear kernel)

if n is small and m is intermediate ($n=1-1000$, $m=10-10'000$)

use SVM with a 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

logistic Regression and SVM without a kernel are about the same!

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

for SVM not to care about global optimum \Rightarrow usually always finds it
Power of SVM

-