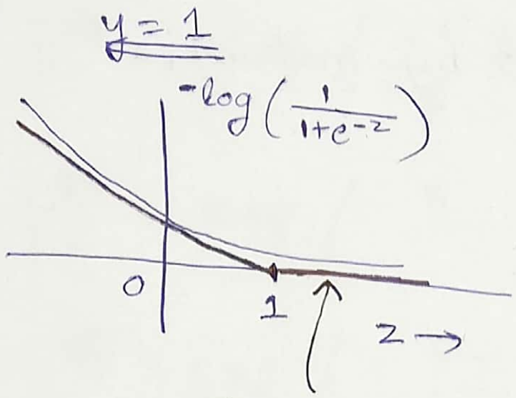
①

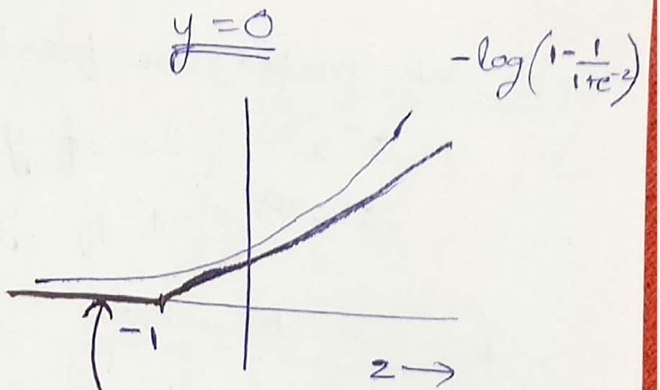
Support Vector Machines (Week 7)

The cost fn. of logistic regression is transformed:

$$-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))$$



Transformation
$\text{Cost}_1(z)$



Transformation
$\text{Cost}_0(z)$

∴ Cost fn:

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

Transformed to $CA+B$ rather than $A+\lambda B$.

∴ Transformed cost fn:

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

Hypothesis:

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

SVM Decision boundary

$$\min_{\theta} C \sum_{i=1}^{m} y^{(i)} cost_1(\theta^T x^{(i)}) + (1-y^{(i)}) cost_0(\theta^T x^{(i)})$$

$$+ \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

We make the first term 0 by making,

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

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

∴ Decision boundary:

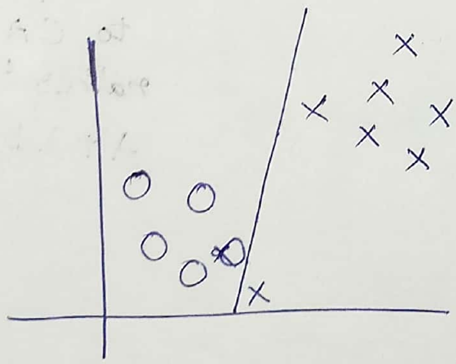
$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

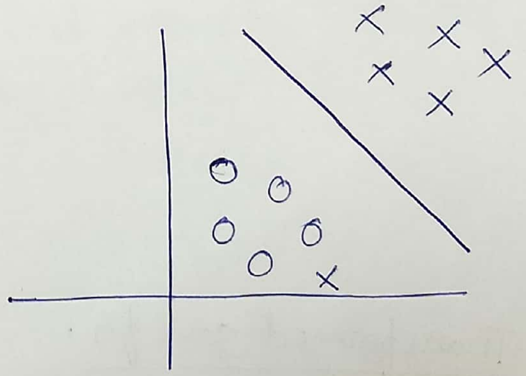
$\theta^T x^{(i)} \geq 1$, if $y^{(i)} = 1$

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

* This is a large margin classifier problem

*





C is large
→ Outliers are also considered.

C is not very large
→ outliers are ignored

* Direcn of Theta is ⊥ to the decision boundary.

Scanned by CamScanner

②

Kernels

iven a set of features x, we find features f_1, f_2, f_3 corresponding to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$.

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

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

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

f_i = similarity $(x, l^{(i)})$

GAUSSIAN KERNEL

where the new hypothesis will be,

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

$$h_\theta(x) = 1 \quad , \text{ if } \theta_0 + \theta_1 f_1 + \ldots \geq 0$$

$$= 0 \quad , \text{ if } \theta_0 + \theta_1 f_1 + \ldots \leq 0$$

$$\boxed{\begin{aligned} f_1 &\approx 1 \text{ if } x \text{ is close to } l^{(1)} \\ &\approx 0 \text{ if } x \text{ is far from } l^{(1)} \end{aligned}}$$

Choosing the landmarks:

$l^{(1)} = x^{(1)}$

$l^{(2)} = x^{(2)}$

$\vdots$

$l^{(m)} = x^{(m)}$

They are chosen to be exactly the pts in the training set.

$$f_p^{(i)} = \exp\left(-\frac{\|x^{(i)} - l^{(i)}\|^2}{2\sigma^2}\right), \quad p \in 1, \ldots, m$$

$$f_0^{(i)} = 1$$

$$\therefore f^{(i)} \in \mathbb{R}^{m+1}.$$

Scanned by CamScanner

SVM with Kernels

Training:

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

$$+ \frac{1}{2} \sum_{j=1}^{m} \theta_j^2$$

Note: $\sum_{j=1}^{m} \theta_j^2 = \theta^T \theta$ where θ doesn't contain θ_0.

* For advanced optimization we minimise $\theta^T M \theta$ instead of $\theta^T \theta$.

Prediction:

Predict $y = 1$ if $\theta^T f \geq 0$.
$y = 0$ otherwise.

SVM parameters:

$C = \uparrow$ high variance
$\downarrow$ ~~hi~~ low variance

$\sigma^2 = \uparrow$ low variance
$\downarrow$ high variance.

Using an SVM

Use SVM software package (eg. liblinear, libsvm, ...) to solve for parameters θ.

We need to choose:

1) Parameter C.
2) Kernel (similarity func^n)

→ No kernel is also called "linear kernel"

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

* Use this when n is large, m is small.

→ Gaussian kernel

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

Need to choose σ^2.

* FEATURE SCALING should be done for gaussian kernel.

→ Other kernels are also available like, OFF-THE-SHELF KERNELS

- Polynomial kernel $k(x, l) = (x^T l + \text{const})^{\text{deg}}$

ESOTERIC KERNELS:

- String kernel, chi-squared kernel, histogram intersect^n kernel, ...

Note: All kernels must satisfy Mercer's Thm. to be used by the advanced optimiz^n algos.

When to use which algorithm?

→ If $n \geq m$, $n = 10,000$ | $m = 10 - 1000$
Use logistic regression or
SVM with linear kernel

→ If n is small, m is intermediate: ($n = 1-1000$, $m = 10-10000$)
Use ~~Ga~~ 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 with linear kernel.

Note: Neural networks work well for most of these settings but maybe slower to train than an SVM...