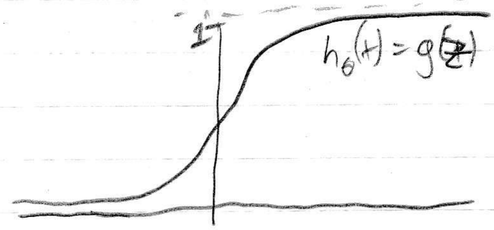
Machine Learning - Week 7 - Support Vector Machines

T1 - Large Margin Classification

L1 - Optimization Objective

Alternate view of logistic regression.

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



$z = \theta^T x$

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

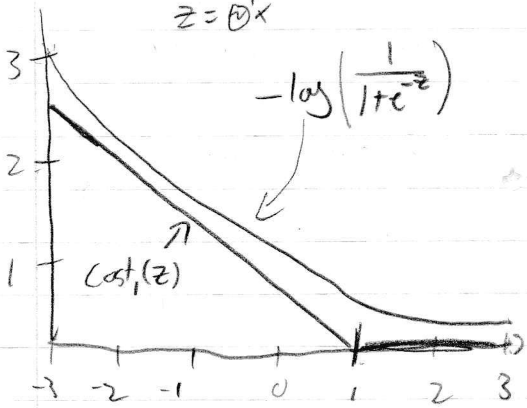
if $y=0$, we want $h_\theta(x) \approx 0$, $\theta^T x \ll 0$

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

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

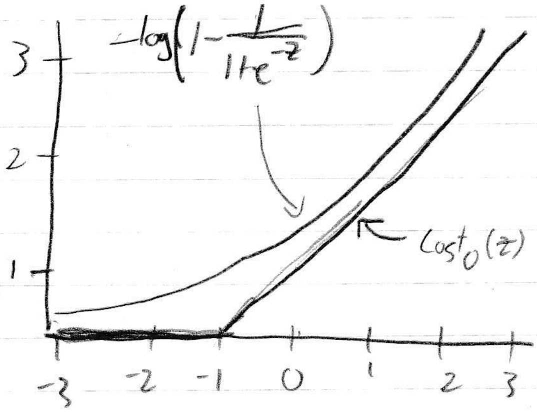
If $y=1$ (want $\theta^T x \gg 0$)

$z = \theta^T x$



z

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



z

Support Vector Machine

Logistic Regression:

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

SVM:

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

Which we rewrite as:

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

Note we remove the $\frac{1}{m}$ from both terms, and rewrote the equation from the form $A + \lambda B$ to $CA + B$ (if $C = \frac{1}{\lambda}$, you'll get the same result)

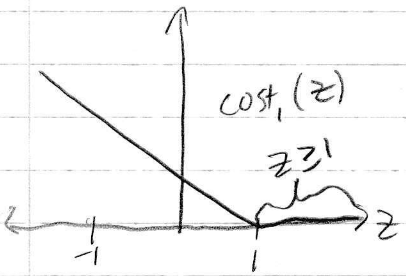
SVM Hypothesis:

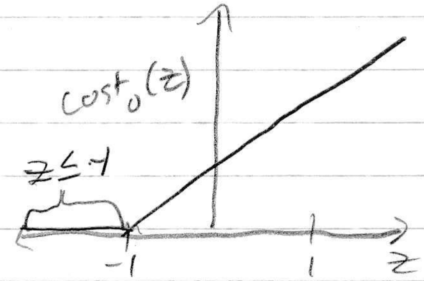
$$h_\theta(x) \begin{cases} 1 & \text{if } \theta^T x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

L2 - Large Margin Intuition

Recall:

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

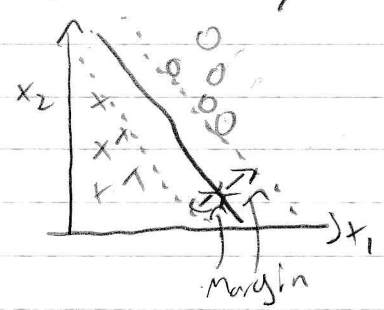




if $y=1$, we want $\theta^T x \geq 1$ (not just ≥ 0)

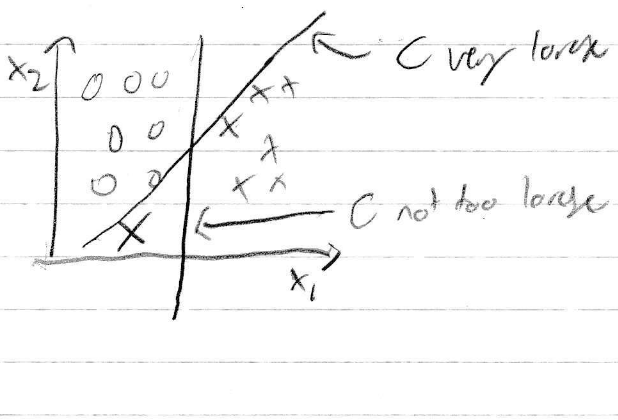
if $y=0$, we want $\theta^T x \leq -1$ (not just < 0)

SVM Decision Boundary: Linearly separable case



SVM will try to select a decision boundary that maximizes the margins (Also called the "large margin classifier".

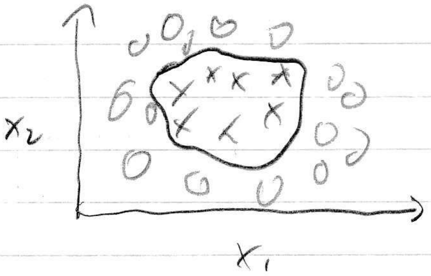
Large Margin Classifier in presence of outliers



T2 - Kernels

L1 - Kernels 1

Non-linear Decision Boundary



Predict $y=1$ if

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

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

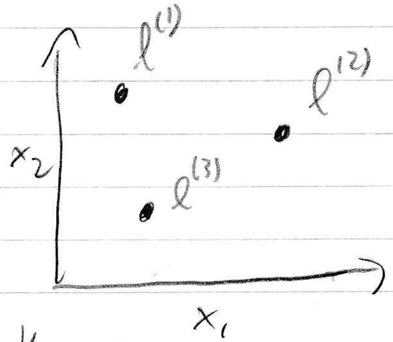
Can rewrite this as $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \dots$

where $f_1 = x_1$, $f_2 = x_2$, $f_3 = x_1 x_2$, ...

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 $l^{(1)}, l^{(2)}, l^{(3)}$.



$\|w\|$ = length of a vector
$\|x - l^{(1)}\|$ = euclidean distance between points

Given x:

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

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

Sometimes written as $k(x, l^{(i)})$

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

Kernels and Similarity.

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

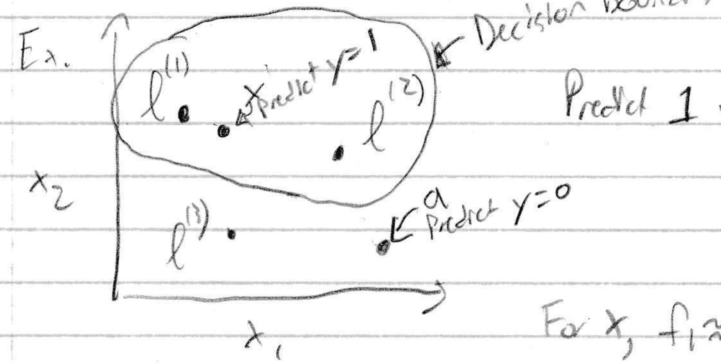
if $x \approx l^{(1)}$:

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

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

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

Ex.



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

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

For x, $f_1 \approx 1$, $f_2 \approx 0$, $f_3 \approx 0$

$\therefore \theta_0 + \theta_1(1) + \theta_2(0) + \theta_3(0)$

$= -0.5 + 1$

$\boxed{= 0.5 \geq 0}$

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

$\therefore \theta_0 + \theta_1(0) + \theta_2(0) + \theta_3(0)$

$\boxed{= -0.5 < 0}$

L3-Kernels II

Choosing the landmarks.

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:

$$f_1 = \text{similarity}(x, \underbrace{l^{(1)}}_{x^{(1)}})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$\vdots$$

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

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

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

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

$$\hookrightarrow f^{(i)} = \begin{bmatrix} f_0^{(i)} \\ f_1^{(i)} \\ \vdots \\ f_m^{(i)} \end{bmatrix} \quad f_0^{(i)} = 1$$

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

Predict "$y=1$" if $\Theta^T f \geq 0$ $\quad \Theta \in \mathbb{R}^{m+1}$

$$\Theta^T f = \Theta_0 f_0 + \Theta_1 f_1 + \dots + \Theta_m f_m$$

Training:

Note: $n = m$ here

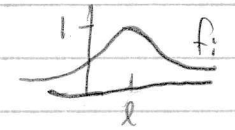
$$\min_\Theta C \sum_{i=1}^{m} y^{(i)} cost_1(\Theta^T f^{(i)}) + (1-y^{(i)}) cost_0(\Theta^T f^{(i)}) + \frac{1}{2}\sum_{j=1}^{n} \Theta_j^2$$

Θ_0

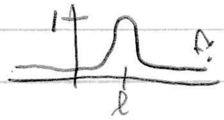
SVM Parameters

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

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



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



T3 - SVMs in Practice
L1 - Using an SVM

Use SVM software package to solve for parameters θ.
Need to specify:
- Choice of parameter C.
- Choice of Kernel (similarity function:
  - eg. No Kernel ("linear Kernel") Predict "$y=1$" if $\theta^T x \geq 0$ } n large, m small $x \in \mathbb{R}^{n+1}$
  - eg. Gausian Kernel: $f_i = \exp\left(-\frac{\|x-l^{(i)}\|^2}{2\sigma^2}\right)$, where $l^{(i)} = x^{(i)}$ Need to choose σ^2 } $x \in \mathbb{R}^n$, n small, and/or m large

Kernel (similarity) functions:
function f = Kernel (x_1, x_2)
$$f = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)$$
return

Note: Do perform feature scaling before using the Gausian Kernel.

Other choices of Kernel
Note: Not all similarity functions 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: $k(x,l) = (x^T l + \text{constant})^{\text{degree}}$
- More esoteric: String kernel, chi-square kernel, histogram intersection kernel...

Multiclass classification
- 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 $i=1 \ldots K$
  → Get $\Theta^{(1)}, \Theta^{(2)}, \ldots, \Theta^{(K)}$
  → Pick class i with largest $(\Theta^{(i)})^T x$

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): (eg $n \geq m$, $n = 10\,000$, $m = 10 \ldots 1000$)
Use logistic regression, or SVM without a kernel ("linear kernel")

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

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