1) perceptron dates back to early 50s. biologically inspired (and can be viewed as a "single neuron" network)

L7

Mar. 31st (Midterm)

HW4 due Apr. 2nd

2) support vector machines: funny name (and not intuitive). as we discussed the reason for "support vectors" comes by examining the dual. in some sense, the optimal model (w) is "supported" on a subset of the data points

Recap:

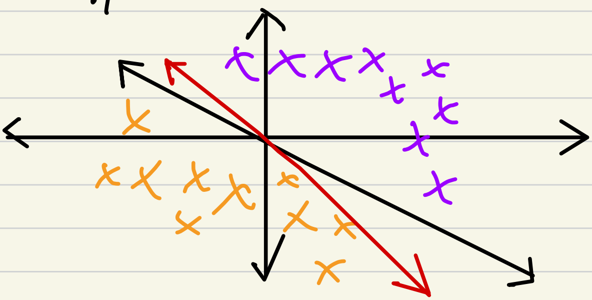
- K-NN
- The classification Algorithm

---

suppore vector machines (SVMs)

kernel methods

Suppore Vecetor Machines (SVMs)



there could be multiple lines (separators)

Infinite!

Q: What is the "best" separator b/w the classes?

Revisiting the Perception

Dataset $\{(x_1, y_1), (x_2, y_2), (x_3, y_3) \ldots\}$

Find w s.t. $\text{sign} \langle w, x_i \rangle = y_i$

such that

Rewriting this :

同号 why not ≥ 0

$$l_i(w) = \begin{cases} 0, & \text{if } \operatorname{sign}\langle w, x_i\rangle = y_i \Rightarrow y_i \cdot \operatorname{sign}\langle w, x_i\rangle \geq 1 \Rightarrow y_i\langle w, x_i\rangle \geq 1 \\ 1, & \text{otherwise} \Rightarrow 1 - \underbrace{y_i\langle w, x_i\rangle}_{<1} \end{cases}$$

Intuition: 1) if y_i & $\langle w, x_i\rangle$ are of same sign AND $\langle w, x_i\rangle$ is large then no loss — 0

2) if y_i & $\langle w, x_i\rangle$ are of opposite signs, then loss ≥ 1 — 1

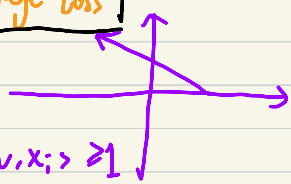
3) everything else in between 0 & 1

$$L(w) = \sum_{i=1}^{n} l_i(w)$$

$$= \sum_{i=1}^{n} \max(0, 1 - y_i\langle w, x_i\rangle)$$

$\boxed{\text{Hinge Loss}}$

$f(z) = \max(0, 1-z)$



$$l_i(w) = \begin{cases} 0, & y\langle w, x_i\rangle \geq 1 \\ 1 - y_i\langle w, x_i\rangle & \text{otherwise} \end{cases}$$

$$\partial l_i(w) = \begin{cases} 0, & y\langle w, x_i\rangle \geq 1 \\ -y_i x_i, & \text{otherwise} \end{cases}$$

(Sub) gradient descent :

不是严格意义的求导/GD...

$$w_{k+1} \leftarrow w_k - \eta_k \, \partial L(w_k)$$

$$= \begin{cases} w_k & , y_i\langle w, x_i\rangle \geq 1 \\ w_k + y_i x_i & , \text{otherwise} \end{cases}$$

SVM

$$L_{svm}(w) = L_{hinge}(w) + \underbrace{\frac{\lambda}{2}\|w\|_2^2}_{\text{support vector}}$$

$$= \sum_{i=1}^{n} \max(0, 1 - y_i \langle w, x_i \rangle) + \frac{\lambda}{2}\|w\|_2^2$$

"primal SVM"

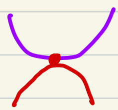
Hinge loss

regularizer

SVM

(Primal) $\min_{w} L_{svm}(w)$

$\boxed{\text{solution}: w^*}$



(Dual) $\max_{\alpha} -\sum_{i=1}^{n} \alpha_i - \sum y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle$

such that $0 \le \alpha_i \le \frac{1}{\lambda}$

Solution: α^*

$$\boxed{w^* = \sum_{i=1}^{n} \alpha_i y_i x_i}$$

Intuition: SVM model is a linear combination of data points

wherever $\alpha_i^* = 0$, solution w^* doesn't depend on x_i

∴ The data points x_i for are called "support vectors"

性能:

Separability:

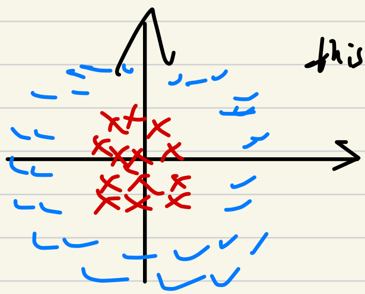
有时可能无法正确完成工作

separable

(could be small mistake)

(could be no-100% ✓)

this is not linear separable.



Ex 1

Ex 2

Ex1:

$(x_1, x_2) \to (x_1, x_2, \sqrt{x_1^2 + x_2^2})$

blue is outer and higher with third dimension

Ex2: do we need 3rd dimension? No!

look at graph. red in I. III phase

blue in II IV phase

$(x_1, x_2) \longrightarrow x_1 x_2$

Embed the data into a different feature space in which

kernel methods :

$x \to \phi(x)$

$\phi(\cdot)$: a transition of the data called the kernel Mapping

Typical feature transformations :

* original feature
* 二次的 Quadratic features

  $(x_1, x_2) \to (x_1^2, x_2^2, x_1 x_2)$
* Higher order polynomials

  $(x_1, x_2) \to (x_1^2, x_2^2, x_1 x_2, x_1^3, x_2^3, x_1^2 x_2, x_1 x_2^2, x_1 x_2),$

  etc.

$$(x_1, x_2, \cdots, x_d) \xrightarrow{\text{Quadratic}}$$

$$(x_1^2, x_2^2, \cdots, x_d^2, x_1x_2, x_1x_3, \cdots, x_1x_d, \cdots \cdots x_{d-1}x_d)$$

approximately d^2 seems unpractical

Going back to circle example

$(x_1, x_2) \to (x_1, x_2, x_1^2, x_2^2, x_1 x_2)$

Circle $\quad w \quad (0, 0, 1, 1, 0)$

$\langle w, \phi(x) \rangle = x_1^2 + x_2^2$

$f(x) = \text{sign}(\langle w, \phi(x)\rangle - \text{radius})$

XOR $w \in 0, 0, 0, 0, 1)$

$f(x) = \text{sgn}(\langle w, \phi(x)\rangle)$

What is important is not the feature mapping, but the ability to take dot products with the feature mapping ☆

→ "kernel trick"

implicitly define feature mapping via the dot product

→ kernel dot product

or kernel inner product

Example of kernel dot products

☆ Regular dot product

$(x_1, x_2) \to (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2)$

☆ quadratic dot product

lift dimension, but make dot product more simple important

$\langle\phi(x), \phi(y)\rangle = k(x,y) \to (1 + \langle x, y\rangle)^2$

$(y_1, y_2) \to (1, \sqrt{2}y_1, \sqrt{2}y_2, \sqrt{2}y_1y_2, y_1^2, y_2^2)$

Dot product $= 1 + 2x_1y_1 + 2x_2y_2 + 2x_1y_1x_2y_2 + x_1^2y_1^2 + x_2^2y_2^2$

$= (1 + x_1y_1 + x_2y_2)^2 = (1 + \langle [x_1, x_2], [y_1, y_2]\rangle)^2$

☆ Cubic dot product

$k(x,y) \to (1 + \langle x, y\rangle)^3$

☆ Exponential

$k(x,y) = \exp(-\|x-y\|^2)$ "Gaussian kernel" "Radial basis function" / RBF