Linear Binary Classification

Thursday, January 16, 2020 12:38

Binary Linear Classification
(perceptron)

- Draw a linear geometry (line/plane...) between 2 categories

$\mathcal{H}$ is a set of lines/hyperplanes.

Model Setup

$$\mathcal{D} = \{(\underline{x}_1, y_1), (\underline{x}_2, y_2) \dots (\underline{x}_N, y_N)\}$$

in out

$$\underline{x}_i = [\underbrace{x_{i0} = 1}_{\text{bias}}, \underbrace{x_{i1}, x_{i2} \dots x_{id}}_{d\text{-values.}}]^T \in \mathbb{R}^{d+1}$$

$y_i = -1$ or $+1$

$y_i \in \{-1, +1\}$

parameter/weights $\quad W = [w_0, w_1 \dots w_d] \in \mathbb{R}^{d+1}$

CLASSIFICATION RULE:

function $h(\underline{x}_n) = \text{sign}(W^T \cdot \underline{x}_n) = \hat{y}_n$ (estimate)

? how do we find values for W

we need a way to see how good/bad our weights are.

⇒ Loss Function:

- one type is Mean Square Error

  but in this classification case, we can just simply count the number of errors.

- indicator function:

$$f(\text{event}) = 1 \text{ if } \{\hat{y}_n \neq y_n\}$$

event is true

$$\Downarrow$$

N

event is true

$$E_{in}(\underline{w}) = \frac{1}{N}\sum_{n=1}^{N} 1\{\hat{y}_n \neq y_n\}$$

in-sample error (for training data)

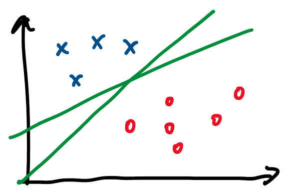
Training involves minimizing $E(\underline{w})$ for the available training data ($\mathcal{D}$)

Perceptron Learning Algorithm (PLA)

INPUT: training set $\mathcal{D}$ is linearly separable

we can put a line to separate data

ex. this is linearly separable



It can be linearly separated using non-unique solutions

OUTPUT: PLA finds $\underline{w} \in \mathbb{R}^{d+1}$ such that $E_{in}(\underline{w}) = 0$

ALGORITHM

d+1 dimensions

- Initialize $\underline{w}(0) = [0, 0, 0 \dots]$

  set $t = 0$

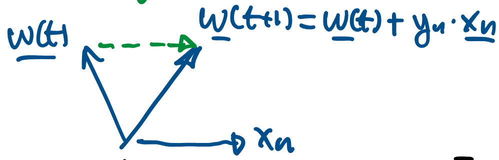
- While $E_{in}(\underline{w}(t)) \neq 0$ then
  - pick any misclassified $(\underline{x}_n, y_n)$
  - update $\underline{w}(t+1) = \underline{w}(t) + y_n \underline{x}_n$
  - $t{+}{+}$

look at it in more detail

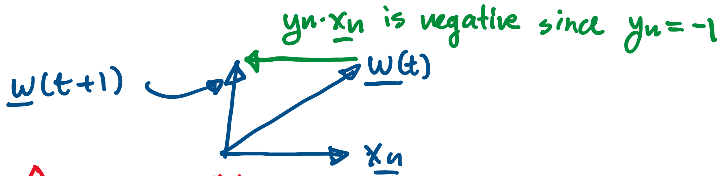
$$\underline{w}(t+1) = \underline{w}(t) + \underbrace{y_n \underline{x}_n}_{\text{misclassified}}$$

case ① $y_n = +1$ $\hat{y}_n = -1$ $\iff$ $\underbrace{\underline{w}(t)^T \cdot \underline{x}_n < 0}_{\text{dot product negative}}$

the idea is to add vector so that we get positive (correct classification)



case ② $\quad y_n = -1 \quad \hat{y}_n = +1 \quad \Longleftrightarrow \quad W(t)^T \cdot x_n > 0$



⚠ notice in this case the "fix" doesn't fully succeed since the product is still ⊕

In summary

tell that a set is misidentified.

y_n	$W(t)^T x_n$	$y_n W(t)^T x_n$	classification
+1	>0	>0	✓
+1	<0	<0	⊗
-1	>0	<0	⊗
-1	<0	>0	✓

PLA: $y_n W^T(t+1) x_n > y_n W^T(t)\, x_n$

⚠ what if training data is not linearly separable?

- → the training will not converge and terminate (and run forever ∞)
- → we can modify algorithm so just minimize E_{in} (keep the best, and continue look for better solution)
- → change objective E_{in}

Pocket algorithm: keep the best vector W until it finds another better set. Terminates after some iterations.