Learning Processes

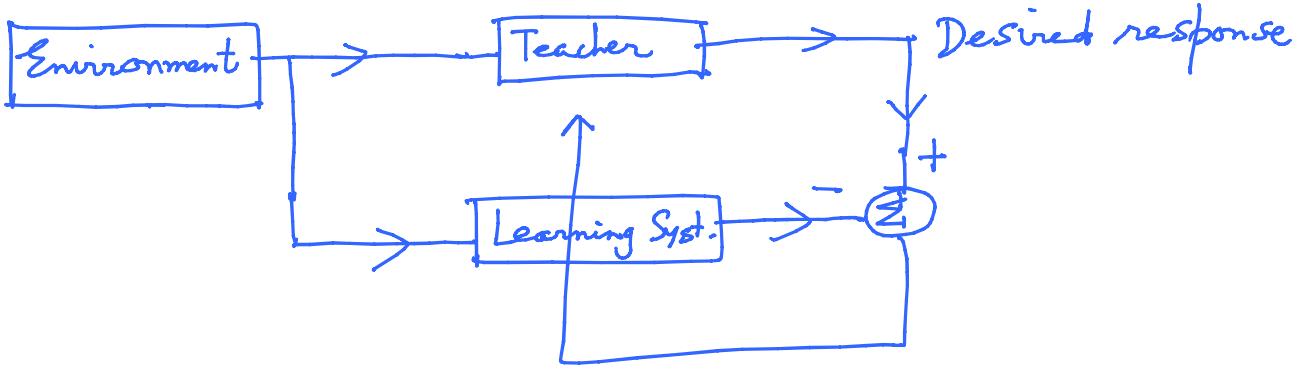
(a) Learning with a teacher

(b) Learning without a teacher

(i) Unsupervised learning

(ii) Reinforcement learning

(a)



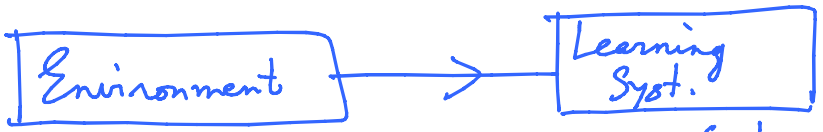
Knowledge of the expert / teacher about the environment is directly transferred to the neural n/w via training

This is also a basis for 'error correction learning'

(b) Learning w/o a teacher

Ex: Social etiquettes.

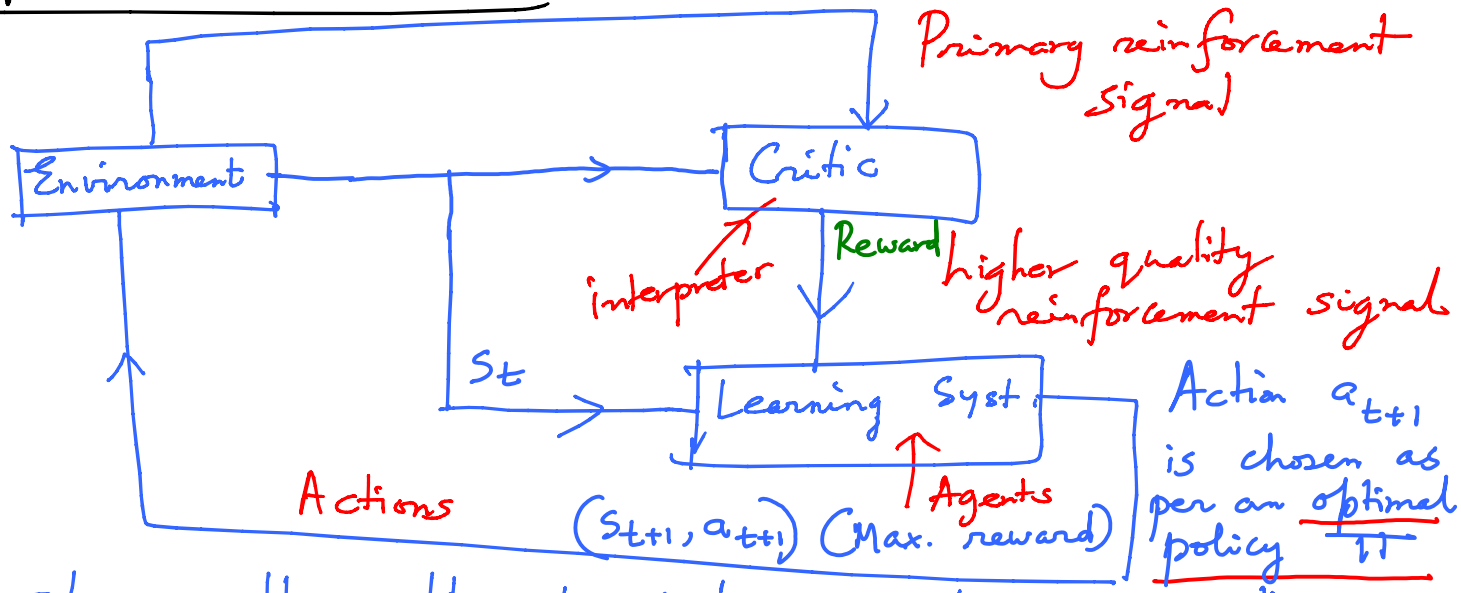
(i) Unsupervised learning



(a) Clustering algos (K-means)

(b) Self Organizing Map (competitive learning)

(ii) Reinforcement learning



Learn I/O mapping through continued interactions with the environment.

Learning Tasks

1) Pattern association
2) Pattern recognition
3) Functional Approximation
4) Control

⋮

Pattern Association

An associative memory is a brain like distributed memory

Learn through associations

(a) Auto association : Unsupervised

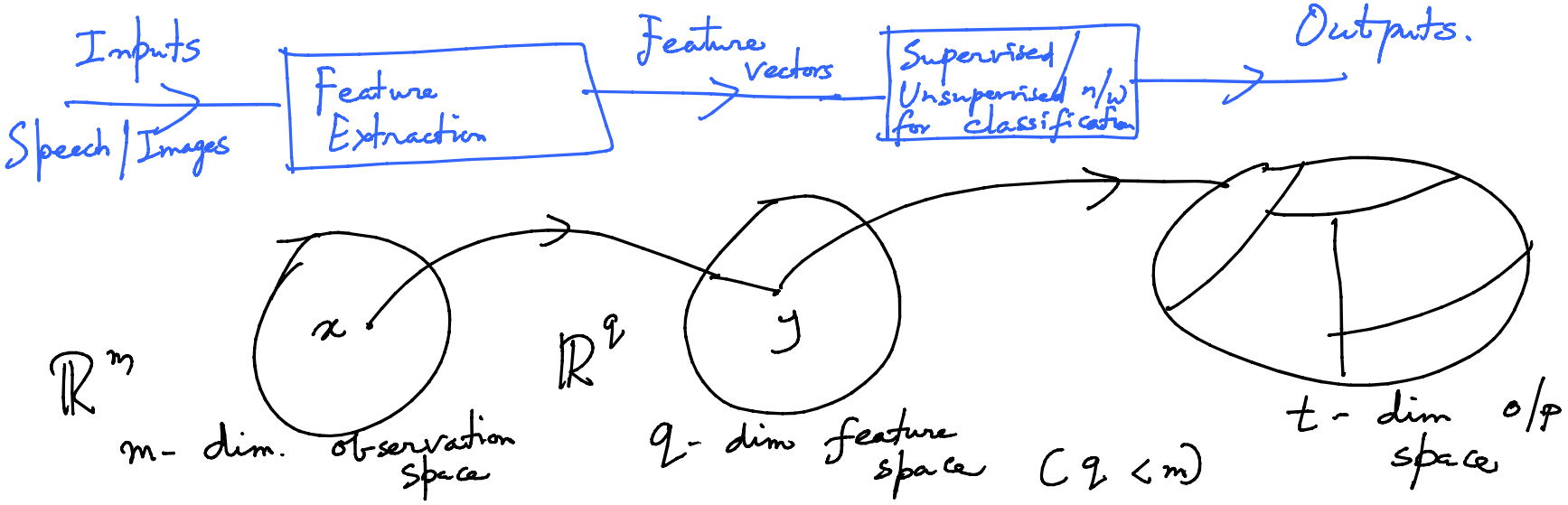
(b) Hetero association : Supervised

Storage Phase : $f : \underline{x}_k \longrightarrow \underline{y}_k$ ($\underline{x}_k$ stimulus, $\underline{y}_k$ memorized pattern)

Recall Phase : $\arg\max_{\underline{y}_k} \phi(\underline{x}, N)$ ($\underline{x}$ ↑ noisy stimulus, N ↖ network; N has $\underline{x}_k$'s embedded within)

Pattern Recognition

It is a process whereby a received pattern / signal is assigned to one of the prescribed # of classes.



Functional Approximation

$\underline{d} = f(\underline{x})$ — $f(\cdot)$ is unknown

$\underline{x}$: input

$\underline{d}$: o/p

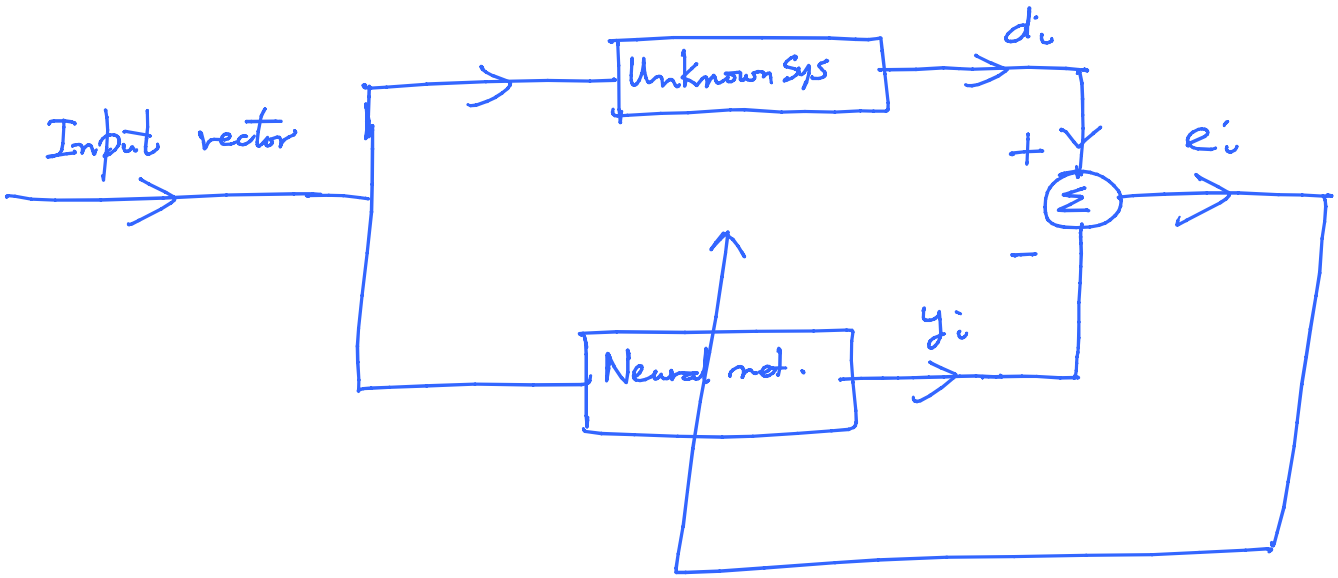
Goal: Design a neural net. / $F(\underline{x})$ / it creates a mapping

$\| F(\underline{x}) - f(\underline{x}) \| < \varepsilon \quad \forall \underline{x}$

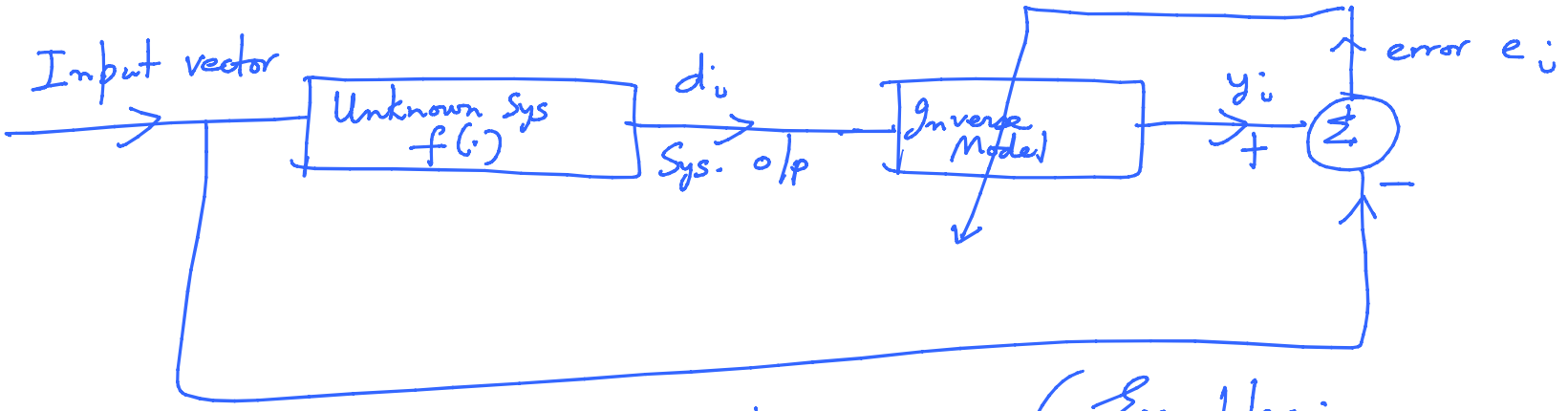
Examples:

(a) System Identification

(b) Inverse Modeling



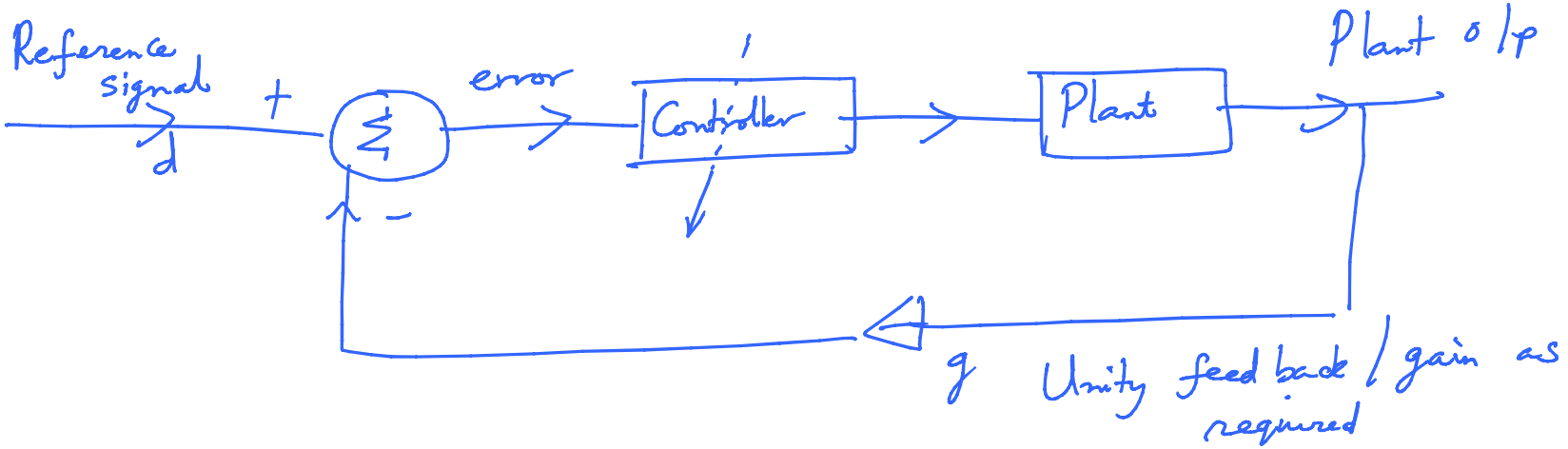
System Identification Block Diagram Schematic



Inverse Modeling

(Examples:
Channel equalization)

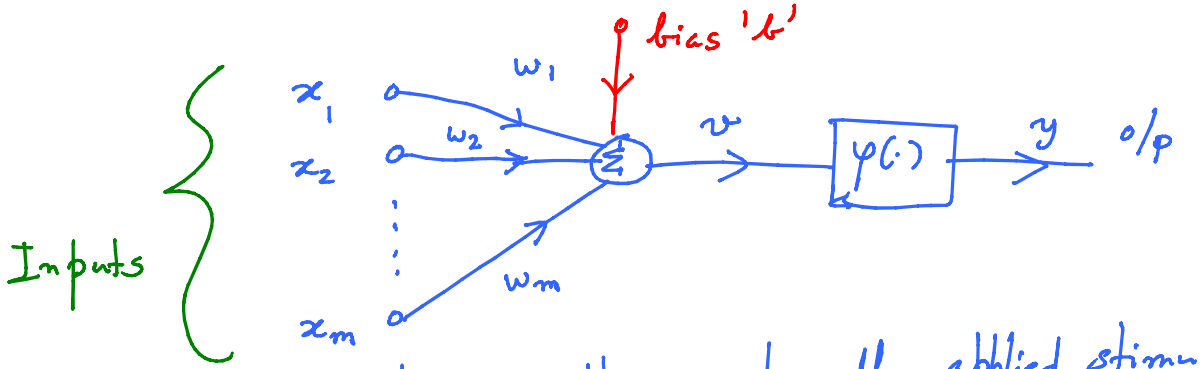
Control of a plant



PERCEPTRON

History: It is the first 'algorithmically' described neural n/w.

Rosenblatt (1943)



$$v = \sum_{i=1}^{m} w_i x_i + b$$

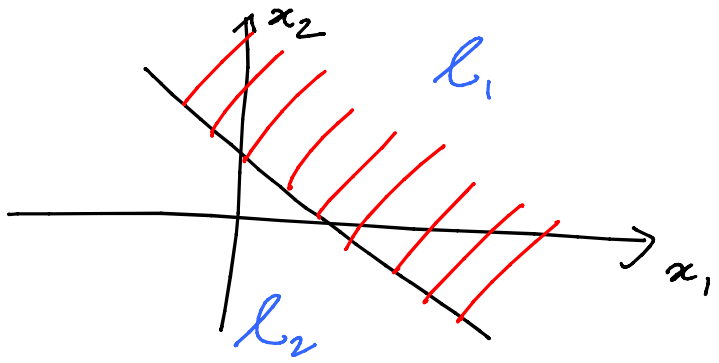
GOAL: Classify the externally applied stimulus into 2 classes C_1 and C_2 (2 class problem)

Decision Rule:

$$y = \begin{cases} +1 & \underline{x} \in \ell_1 \\ -1 & \underline{x} \in \ell_2 \end{cases}$$

Now, $\sum_{i=1}^{m} w_i x_i + b = 0$ is the equation of a hyperplane

Illustration in $\mathbb{R}^2$



We need to obtain the equation of the hyperplane

$$\underline{w}^T \underline{x} + b = 0$$

Perceptron

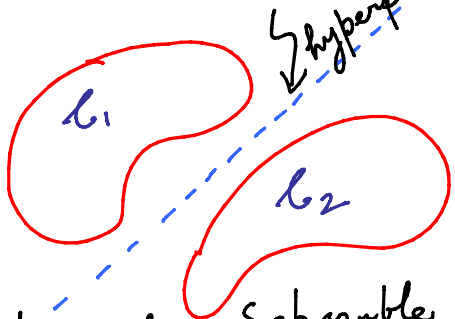
We form $\underline{x}(n) = [+1 \quad x_1(n) \quad x_2(n) \quad \dots \quad x_m(n)]^T$

$\underline{w}(n) = [b \quad w_1(n) \quad w_2(n) \quad \dots \quad w_m(n)]^T$

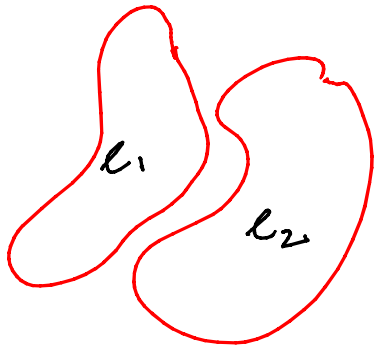
These are $(m+1) \times 1$ vectors

$v(n) = \underline{w}^T(n)\,\underline{x}(n)$

(Classes do not overlap)



(a) Linearly Separable



(b) Not linearly separable

Suppose we consider inputs that come from a linearly separable class,

$$\underline{w}_{opt}^T \underline{x} > 0 \implies \underline{x} \in C_1 \quad \forall \underline{x}$$

$$\underline{w}_{opt}^T \underline{x} < 0 \implies \underline{x} \in C_2 \quad \forall \underline{x}$$

Algorithm towards getting a solution to $\underline{w}$.

1) If the n^{th} data point from the training set, i.e., $\underline{x}(n)$ is correctly classified by the weight $\underline{w}(n)$ then no correction is made to the w.t. vector in accordance to the following rule

$$\begin{cases} \underline{w}(n+1) = \underline{w}(n) & \text{if } \underline{w}^T(n)\,\underline{x}(n) > 0 \text{ and } \underline{x}(n) \in \mathcal{C}_1 \\ \underline{w}(n+1) = \underline{w}(n) & \text{if } \underline{w}^T(n)\,\underline{x}(n) \leq 0 \text{ and } \underline{x}(n) \in \mathcal{C}_2 \end{cases}$$

ELSE

2) Update the weight vector as

$$\begin{cases} \underline{w}(n+1) = \underline{w}(n) - \eta(n)\,\underline{x}(n) & \text{if } \underline{w}^T(n)\,\underline{x}(n) > 0 \text{ & } \underline{x}(n) \in C_2 \\ \underline{w}(n+1) = \underline{w}(n) + \eta(n)\,\underline{x}(n) & \text{if } \underline{w}^T(n)\,\underline{x}(n) \leq 0 \text{ & } \underline{x}(n) \in C_1 \end{cases}$$

Here $\eta(n)$ is a learning parameter

Sometimes / Most times η is a fixed constant typically small.

$\eta = 1$ also works

Perceptron Convergence Theorem

Let us start with the initial condition $\underline{w}(0) = \underline{0}$

Suppose $\underline{w}^T(n)\,\underline{x}(n) \leq 0$ for $n = 1, 2, \ldots$

and $\underline{x}(n) \in \mathcal{C}_1$ i.e., it is incorrectly classified

With $\eta = 1$

$$\underline{w}(n+1) = \underline{w}(n) + \underline{x}(n)$$

Let us recursively unfold $\underline{w}(n+1)$

$$\underline{w}(n+1) = \underline{x}(n) + \underline{x}(n-1) + \cdots + \underline{x}(1) + \underline{x}(0) \qquad (1)$$

Since the classes are linearly separable, $\underline{w}_{opt}^T(n)\,\underline{x}(n) > 0$

$\exists$ a solution $\underline{w}_{opt}$

for all inputs $\{\underline{x}\} \in \mathcal{C}_1$

For a fixed $\underline{w}_0$, let us define

$$\alpha = \min_{\underline{x}(n) \in \mathcal{C}_1} \underline{w}_0^T\,\underline{x}(n) \qquad \text{(2)}$$

Multiply (1) b.s. by $\underline{w}_0^T$

$$\underline{w}_0^T\,\underline{w}(n+1) = \underbrace{\underline{w}_0^T\,\underline{x}(0) + \underline{w}_0^T\,\underline{x}(1) + \cdots + \underline{w}_0^T\,\underline{x}(n)}_{\text{Totally } (n+1) \text{ terms}}$$

$$\underline{w}_0^T\,\underline{w}(n+1) \geq (n+1)\alpha \qquad \text{(3)}$$

Let us apply Cauchy Schwartz inequality to the L.H.S. of (3)

$$\|\underline{w}_0\|^2 \, \|\underline{w}(n+1)\|^2 \geq \left(\underline{w}_0^T \, \underline{w}(n+1)\right)^2 \quad \text{---} \quad (4)$$

From (4) and (3)

$$\|\underline{w}_0\|^2 \, \|\underline{w}(n+1)\|^2 \geq (n+1)^2 \alpha^2$$

$$\|\underline{w}(n+1)\|^2 \geq \frac{(n+1)^2 \alpha^2}{\|\underline{w}_0\|^2} \quad \text{---} \quad (5)$$

Let us deviate slightly,

$$\underline{w}(k+1) = \underline{w}(k) + \underline{x}(k) \quad ; \quad k = 1, \ldots, n$$

$\underline{x}(k) \in \ell_1$

Taking the Euclidean norm on both sides of the above eqn

$$\|\underline{w}(k+1)\|^2 = \|\underline{w}(k)\|^2 + \|\underline{x}(k)\|^2 + 2\,\underline{w}^T(k)\,\underline{x}(k)$$

But $\underline{w}^T(k)\,\underline{x}(k) \leq 0$ by assumption

$$\therefore \quad \|\underline{w}(k+1)\|^2 \leq \|\underline{w}(k)\|^2 + \|\underline{x}(k)\|^2$$

$$\|\underline{w}(k+1)\|^2 - \|\underline{w}(k)\|^2 \leq \|\underline{x}(k)\|^2$$

$k = 1, \ldots, n$

$$\| \underline{w}(n+1) \|^2 \leq \sum_{k=0}^{n} \| \underline{x}(k) \|^2 \qquad (7)$$

Let $\beta = \max_{\underline{x}(k) \in \ell_1} \| \underline{x}(k) \|^2$

$$\| \underline{w}(n+1) \|^2 \leq (n+1)\beta \qquad (8)$$

From eqns. (8) and (5), there must ∃ a n_{max} ∋ it satisfies (8) and (5) with equality

$$\frac{(1+n_{max})^2 \alpha^2}{\|\underline{w}_0\|^2} = (1+n_{max})\beta$$

$$n_{max} = \left\lceil \frac{\beta}{\alpha^2} \|\underline{w}_0\|^2 \right\rceil - 1$$

$\Rightarrow$ A solution $\underline{w}_0$ exists and the perceptron algo. Converges in n_{max} iterations.

$\blacksquare$

Notes

1) One can start with any $\underline{w}(0)$ instead of $\underline{w}(0) = \underline{0}$

2) One can choose η to be a fixed small +ve quantity

Aside (Implementation)

You can also normalize the updates & the data so that the update rule is bounded.

Batch Perceptron Algorithm

$$J(\underline{w}) = \sum_{\underline{x} \in \mathcal{H}} (-\underline{w}^T \underline{x}) \, d_{\underline{x}}$$

$\mathcal{H}$ is the set of misclassified points

If all the points were correctly classified $\mathcal{H} = \phi$; $J(\underline{w}) = 0$

$$\nabla_{\underline{w}} J = \sum_{\underline{x} \in \mathcal{H}} -\underline{x} \, d_{\underline{x}}$$

$$\underline{w}(n+1) = \underline{w}(n) - \eta(n) \nabla_{\underline{w}} J(\underline{w})$$

$$= \underline{w}(n) + \eta(n) \sum_{\underline{x} \in \mathcal{H}} \underline{x} \, d_{\underline{x}}$$

Proof of the batch perceptron algo

We assume that η remains the same for all 'n'.

Let $\eta = 1$. Let the initial weight vector be $\underline{w}(0) = \underline{0}$

Consider $\underline{w}^T \underline{x} \leq 0$ and $\underline{x} \in \mathcal{H}$

$$\underline{w}(n+1) = \underline{w}(n) + \sum_{\underline{x} \in \mathcal{H}_n} \underline{x}$$

$$= \sum_{\underline{x} \in \mathcal{H}_0} \underline{x} + \cdots + \sum_{\underline{x} \in \mathcal{H}_n} \underline{x} \quad \text{——— (A)}$$

Let us consider $\underline{w}_0$ / $\underline{w}_0^T \underline{x} > 0 \quad \forall \underline{x} \in \mathcal{C}_1$

Let us premultiply (A) by $\underline{w}_0^T$

$$\underline{w}_0^T \underline{w}(n+1) = \sum_{\underline{x} \in \mathcal{H}_0} \underline{w}_0^T \underline{x} + \cdots + \sum_{\underline{x} \in \mathcal{H}_n} \underline{w}_0^T \underline{x} \qquad \text{(B)}$$

Let $\alpha = \min_i \sum_{\underline{x} \in \mathcal{H}_i} \underline{w}_0^T \underline{x}$

Now by Cauchy Schwartz inequality applied to LHS of (B)

$$\|\underline{w}_0\|^2 \|\underline{w}(n+1)\|^2 \geq (n+1)^2 \alpha^2 \qquad \text{(C)}$$

$$\Rightarrow \|\underline{w}(n+1)\|^2 \geq \frac{(n+1)^2 \alpha^2}{\|\underline{w}_0\|^2}$$

$$\underline{w}(n+1) = \underline{w}(n) + \sum_{\underline{x} \in \mathcal{H}_n} \underline{x}$$

$$\| \underline{w}(n+1) \|^2 = \| \underline{w}(n) \|^2 + \Big\| \sum_{\underline{x} \in \mathcal{H}_n} \underline{x} \Big\|^2 + \underbrace{2\, \underline{w}^T(n) \sum_{\underline{x} \in \mathcal{H}_n} \underline{x}}_{\leq 0}$$

$$\| \underline{w}(n+1) \|^2 \leq \| \underline{w}(n) \|^2 + \Big\| \sum_{\underline{x} \in \mathcal{H}_n} \underline{x} \Big\|^2$$

Let us choose

$$\| \underline{w}(n+1) \|^2 \leq \sum_{i=0}^{n} \Big\| \sum_{\underline{x} \in \mathcal{H}_i} \underline{x} \Big\|^2 ; \quad \beta = \max_i \Big\| \sum_{\underline{x} \in \mathcal{H}_i} \underline{x} \Big\|^2$$

$$\| \underline{w}(n+1) \|^2 \leq (n+1)\beta \quad \text{——— (D)}$$

From (C) and (D) $\exists$ n_{max} / the inequalities satisfy the equality condition

$$n_{max} = \left\lceil \frac{\beta}{\alpha^2} \| \underline{w}_0 \|^2 \right\rceil - 1$$

$\Rightarrow$ Proves that the batch perceptron algorithm Converges after n_{max} iteration ∎

Notes

In the batch perceptron algorithm 'n' is on an 'epoch' basis i.e., You group all the misclassified points & then do an update based on a gradient of the cost.

Whereas in the perceptron algo, 'n' is the online time step where data is presented to the N. Netw.

Relationship between perceptron and Bayes classifier

Consider a 2 class problem

Let p_i, $i = 1, 2$ be the apriori probability

C_{ij} be the cost of deciding ℓ_i when ℓ_j is true
$i, j \in 1, 2$

$p(\underline{x} | \ell_i)$ be the conditional prob. density of a data point $\underline{x}$ in ℓ_i

The risk

$$R = c_{11}\, p_1 \int_{\mathcal{R}_1\ (\text{Correct})} p(\underline{x}\,|\,\ell_1)\, d\underline{x} + c_{21}\, p_1 \int_{\mathcal{R}_2\ (\text{incorrect})} p(\underline{x}\,|\,\ell_1)\, d\underline{x}$$

$$+ c_{12}\, p_2 \int_{\mathcal{R}_1\ (\text{incorrect})} p(\underline{x}\,|\,\ell_2)\, d\underline{x} + c_{22}\, p_2 \int_{\mathcal{R}_2\ (\text{Correct})} p(\underline{x}\,|\,\ell_2)\, d\underline{x}$$

Any data point $\in$ either $\mathcal{R}_1$ or $\mathcal{R}_2$

The following quantities are obvious:

$$\int_{\mathcal{X}} p(\underline{x}|\ell_1)\, d\underline{x} = \int_{\mathcal{X}} p(x|\ell_2)\, dx = 1$$

$$\left.\begin{aligned} c_{11} &< c_{21} \\ c_{22} &< c_{12} \end{aligned}\right\}$$

$$R = c_{11}\, p_1 \int_{\mathcal{X}_1} p(\underline{x}|\ell_1)\, d\underline{x} + c_{22}\, p_2 \int_{\mathcal{X}-\mathcal{X}_1} p(\underline{x}|\ell_2)\, d\underline{x}$$

$\mathcal{X}_1$: corresponds to ℓ_1 $\qquad$ $\mathcal{X}-\mathcal{X}_1$: corresponds to ℓ_2

$$+ c_{21}\, p_1 \int_{\mathcal{X}-\mathcal{X}_1} p(\underline{x}|\ell_1)\, d\underline{x} + c_{12}\, p_2 \int_{\mathcal{X}_1} p(\underline{x}|\ell_2)\, d\underline{x}$$

Idea: Keep $c_{21}\, p_1 + c_{22}\, p_2$ fixed

$$R = c_{21} p_1 + c_{22} p_2 - \int_{\mathcal{X}_1} p_1 c_{21} p(\underline{x}|\ell_1)\, d\underline{x}$$

$$- \int_{\mathcal{X}_1} p_2 c_{22} p(\underline{x}|\ell_2)\, dx + \int_{\mathcal{X}_1} p_2 c_{12} p(\underline{x}|\ell_2)\, d\underline{x}$$

$$+ \int_{\mathcal{X}_1} c_{11} p_1 p(x|\ell_1)\, dx$$

Suppose $c_{21} p_1 + c_{22} p_2$ is a fixed cost.

$$R = \int_{\mathcal{X}_1} p_2 (c_{12} - c_{22}) p(\underline{x} | \ell_2) \, d\underline{x} + \int_{\mathcal{X}_1} p_1 (c_{21} - c_{11}) p(\underline{x} | \ell_1) \, d\underline{x}$$

We want to minimize R

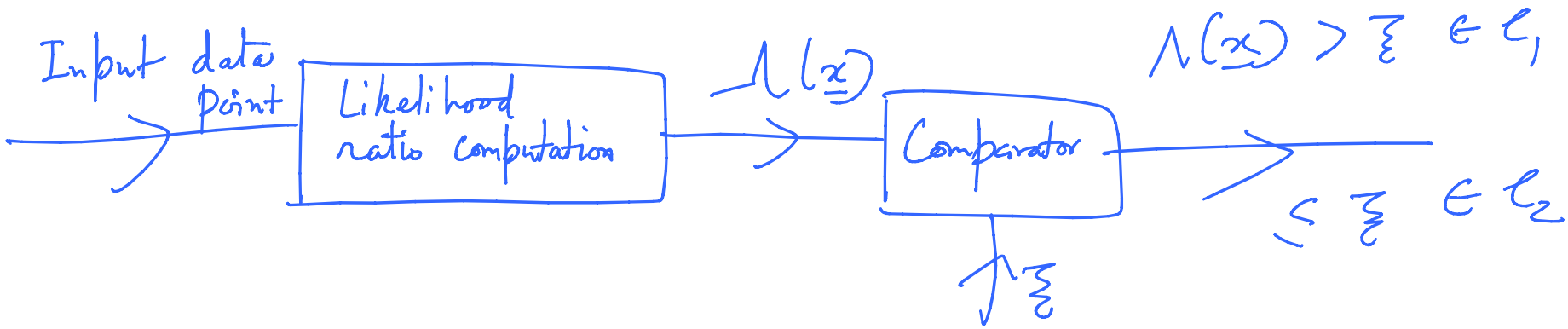
The integral must vanish for a threshold

$$p_1 (c_{21} - c_{11})\, p(\underline{x} \mid \ell_1) \underset{\underline{x} \in \ell_2}{\overset{\underline{x} \in \ell_1}{\gtrless}} p_2 (c_{12} - c_{22})\, p(\underline{x} \mid \ell_2)$$

Let $\Lambda(\underline{x}) \triangleq \dfrac{p(\underline{x} \mid \ell_1)}{p(\underline{x} \mid \ell_2)}$ and

$$\xi = \frac{p_2 (c_{12} - c_{22})}{p_1 (c_{21} - c_{11})}$$

(threshold)


Input data point
Likelihood ratio computation
$\Lambda(\underline{x})$
Comparator
ξ
$\Lambda(\underline{x}) > \xi \in \ell_1$
$\leq \xi \in \ell_2$

Example : Gaussian Distributions over 2 classes

Let us consider a multivariate Gaussian distribution

Class ℓ_1 : $E(\underline{x}) = \underline{\mu}_1$

$$E\left((\underline{x} - \underline{\mu}_1)(\underline{x} - \underline{\mu}_1)^T\right) = C$$

Class ℓ_2 : $E(\underline{x}) = \underline{\mu}_2$

$$E\left((\underline{x} - \underline{\mu}_2)(\underline{x} - \underline{\mu}_2)^T\right) = C$$

C is non-diagonal & we assume it is non-singular

$$p(\underline{x} \mid \ell_i) = \frac{1}{(2\pi)^{m/2} (\det(C))^{\frac{1}{2}}} e^{-\frac{1}{2}(\underline{x}-\underline{\mu}_i)^T C^{-1} (\underline{x}-\underline{\mu}_i)}$$

$i = 1, 2$

Let us suppose

(a) $P_1 = P_2$

(b) Misclassifications have the same cost

$C_{21} = C_{12}$ & $C_{11} = C_{22}$

$$\log \Lambda(\underline{x}) = \log\left[\frac{p(\underline{x}|c_1)}{p(\underline{x}|c_2)}\right]$$

$$(C^{-1})^T = C^{-1}$$

$$= -\frac{1}{2}(\underline{x} - \underline{\mu}_1)^T C^{-1}(\underline{x} - \underline{\mu}_1) + \frac{1}{2}(\underline{x} - \underline{\mu}_2)^T C^{-1}(\underline{x} - \underline{\mu}_2)$$

$$= \underbrace{(\underline{\mu}_1 - \underline{\mu}_2)^T C^{-1}}_{W^T}\underline{x} + \underbrace{\frac{1}{2}\left(\underline{\mu}_2^T C^{-1}\underline{\mu}_2 - \underline{\mu}_1^T C^{-1}\underline{\mu}_1\right)}_{b}$$

Let $\underline{w} = C^{-1}(\underline{\mu}_1 - \underline{\mu}_2)$

We have an equation of the form

$$y = \underline{w}^T \underline{x} + b$$

This is a linear classifier

Bayes classifier for Gaussian distributions $\equiv$ Perceptron

(Similar)

Subtle Differences b/n Perceptron & Bayes' classifier

1) Perceptron assumes inherently that the patterns/data points are linearly separable.
   This is not the case in Bayes' classifier.
2) Bayes' classifier minimizes the prob. of classification error
3) Perceptron is non-parametric ⇒ No assumptions on underlying distribution
4) Bayes' classifier is 'parametric'
   Perceptron update & convergence is simple & adaptive; Bayes' is not