Label 1 $\langle \underline{w}_1, \underline{x} \rangle + b_1$

Label 2 $\langle \underline{w}_2, \underline{x} \rangle + b_2$

Label 3 $\langle \underline{w}_3, \underline{x} \rangle + b_3$.

$\underline{x} \begin{bmatrix} - \\ - \\ - \\ - \\ - \\ - \end{bmatrix}_{w \times h}$

Linear discriminant function

prediction $= \underset{i}{\operatorname{argmax}} \langle \underline{w}_i, \underline{x} \rangle$

W = np.random.randn (3, 3072)

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \overset{3072 \longrightarrow}{\begin{bmatrix} - - - - - - - - \\ \\ \\ \end{bmatrix}} \underset{\underline{x}}{\begin{bmatrix} \\ \\ \\ \end{bmatrix}} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix}$$

32×32

1024×3
= 3072

$$\begin{bmatrix} (r,g,b) & (r,g,b) & \cdots & 32 \\ (rgb) & (rgb) & \cdots & \\ \vdots & & & \end{bmatrix}$$

Image

Flatten

$$\begin{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix} 1^{st} \text{ pixel} \\ \begin{bmatrix} r \\ g \\ b \end{bmatrix} 2^{nd} \text{ pixel} \\ \vdots \\ \vdots \end{bmatrix} 3072$$

1-D vector

flattened vector

Writing the mapping $x \longmapsto \langle \tilde{w}, \tilde{x} \rangle$ makes it a homogeneous linear function.

i.e. a homogeneous map.

What's the advantage of writing a homogeneous form?

We can scale the vector w to impose a desired minimum value

$$\left.\begin{aligned} \langle w_1, x_1 \rangle &= 10 \\ \langle w_2, x_2 \rangle &= 7 \\ \langle w_1, x_3 \rangle &= 3 \end{aligned}\right]$$

If we insist that the minimum value should be 1

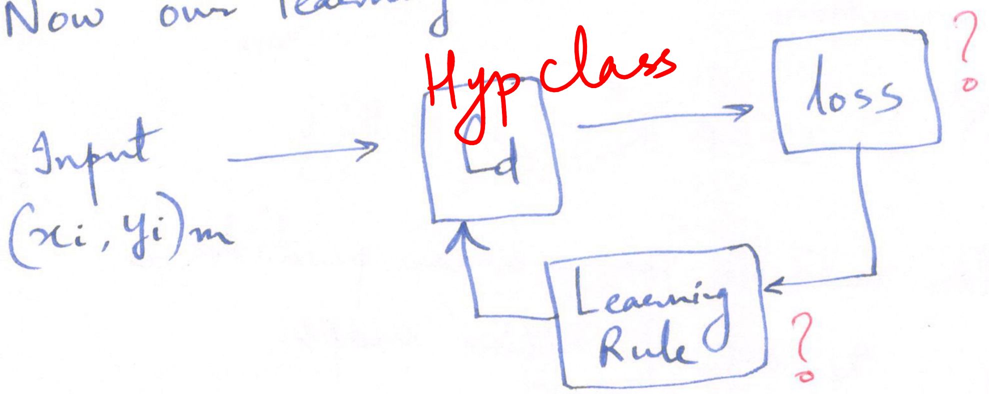
$w \leftarrow \frac{1}{3} w_1$

scaled

then
$$\begin{aligned} \langle w, x_1 \rangle &= 10/3 \\ \langle w, x_2 \rangle &= 7/3 \\ \langle w, x_3 \rangle &= 3/3 = 1. \end{aligned}$$

min value 1.

Now our learning model is



The loss function to be used depends on the final task.

Task 1 : Binary Classification 2 classes.

+1

−1.

$h_w(x) \in \mathbb{R} \longmapsto \{+1, -1\}$ by checking the sign.

$\{-, -, \}$

linear Hypothesis

i.e. if $\underline{h_w(x)} \geq 0 \xmapsto{\text{map to}} +1$

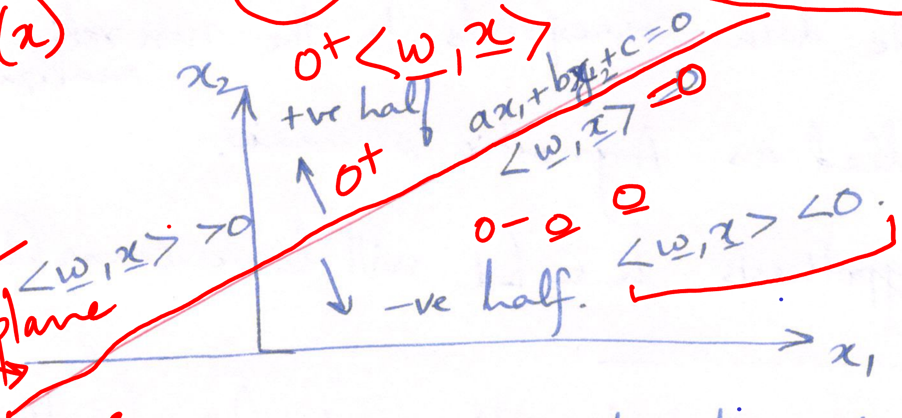
$h_w(x) < 0 \longmapsto -1$

$\Big\}\ \underline{\underline{\text{sign}}}\ \underline{(h_w(x))}$

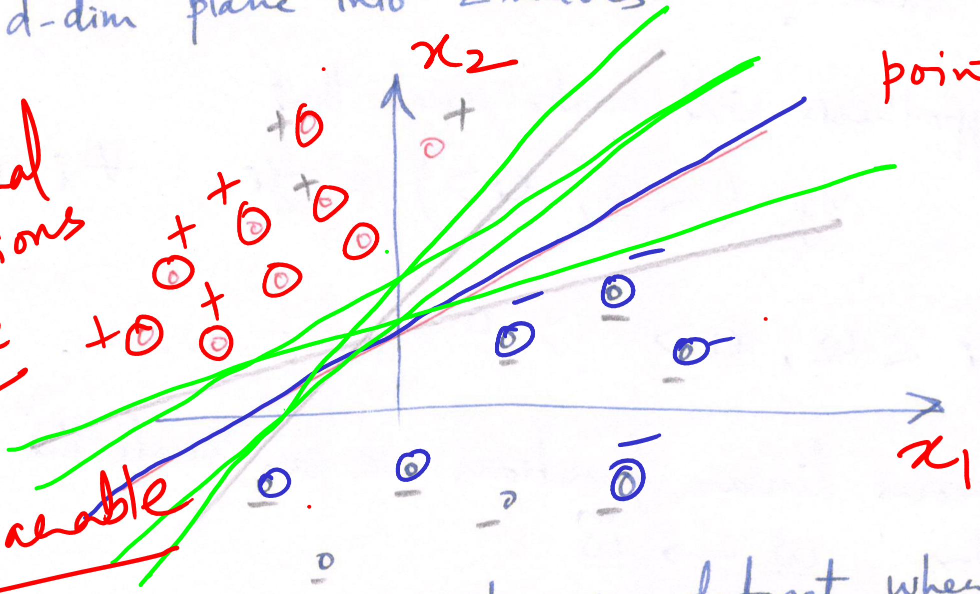
$h_{w_1}(x)$
$h_{w_2}(x)$
$h_{w_3}(x)$

i.e. $x \longmapsto \text{sign}(h_w(x))$ → d parameters, binary

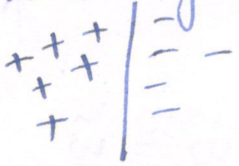
2d: $\underline{h_{w_1}(x)}$, $\underline{h_{w_2}(x)}$

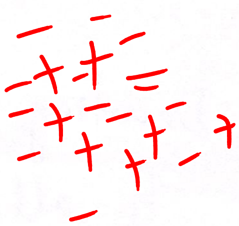


In general, a d-dim hyperplane divides the d-dim plane into 2 halves.



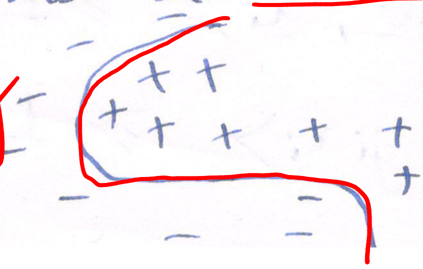
This is a simple training dataset where the dataset is separable, i.e, the positive and negative points are well separated.





Data can also be <u>non-separable</u>

Not linearly separable



This requires a non-linear separating boundary

The separable case is also called as the Realizable case

For this case, the best hypothesis $h \in \mathcal{L}_d$ has $\text{error}(h) = 0$

The non separable data corresponds to the non ~~reli~~ realizable case. Also called as Agnostic scenario.

The best hypothesis $h \in \mathcal{L}_d$ will have $\text{error}(h) > 0$

$S: \{x_i, y_i\}_m$

How to find the best hypothesis in $\mathcal{L}_d$ for the Realizable case?

$\underline{w}$ gives the best hypothesis

The best hypothesis will make sure that

100% correct classification

$$\text{sign}(\underbrace{\langle \underline{w}, \underline{x_i} \rangle}_{\text{scalar}}) = y_i \quad \underbrace{\forall i = 1..m.}_{\text{for all examples}}$$

i.e.

condition for correct classification

$$\boxed{y_i \langle \underline{w}, \underline{x_i} \rangle > 0} \quad \underbrace{\forall i = 1..m}_{\text{set of constraints}} \quad y_i \in \{+1, -1\}$$

LHS

For homogeneous linear functions we can scale the parameter vector $\underline{w} \longrightarrow \underline{\hat{w}}$

$\underline{\hat{w}}$

such that

$$\boxed{y_i \langle \underline{\hat{w}}, \underline{x_i} \rangle \geq 1} \quad \forall i$$

Denoting $\underline{w} \equiv \underline{\hat{w}}$ we require for correct classification of all data points

$$y_i \langle \underline{w}, \underline{x_i} \rangle \geq 1 \quad \forall_i$$

y_i $\quad \underline{x_i} = \begin{bmatrix} x_i^1 \\ x_i^2 \\ x_i^3 \\ \vdots \\ x_i^d \end{bmatrix} d$

$$\boxed{\langle \underline{w}, y_i \underline{x_i} \rangle \geq 1} \quad \forall_i \quad m \text{ constraints}$$

100 examples

$y_1 \langle \underline{w}, \underline{x_1} \rangle = 10$

$y_2 \langle \underline{w}, \underline{x_2} \rangle = 13$

$y_3 \langle \underline{w}, \underline{x_3} \rangle = 6$ min

$\underline{x_4} = 23.$

⋮

100

$\underline{\tilde{w}} = \frac{1}{6} \underline{w}$

scaling

$\langle \underline{\tilde{w}}, \underline{x_1} \rangle = 10/6$

$\underline{x_2} = 13/6$

$\underline{x_3} = 6/6 =$

$$\underbrace{\begin{bmatrix} y_1 x_{11} & y_1 x_{12} & y_1 x_{13} & \cdots & y_1 x_{1d} \\ y_2 x_{21} & y_2 x_{22} & \cdots & \cdots & y_2 x_{2d} \\ & & \vdots & & \end{bmatrix}}_{\underline{A}} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}}_{\underline{w}} \geq \underline{1}.$$

$A \, \underline{w} \geq \underline{v}$ where $\underline{v} \equiv \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$ vector of 1s.

LP solvers.

The standard linear programming problem is

Input $\underline{u}$, $\underline{A}$, $\underline{v}$ Output $\underline{w}$

$$\max_{\underline{w}} \left(\underline{u}^T \underline{w} \right)$$

subject to linear inequality constraints

$$\boxed{A \, \underline{w} \geq \underline{v}}$$

$\underline{u} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$ $\underline{w} \begin{bmatrix} \\ \\ \\ \end{bmatrix}$

So we design a linear program for our problem:

$$\max_{\underline{w}} \left(\text{constant } 1 \right)$$

We have nothing to maximize

subject to $\langle \underline{w}, y_i \underline{x}_i \rangle \geq \underline{1}$ $\forall_i$

solution given by the LP solver is the weight vector $\underline{w}$

Using the weight vector, any given point $\underline{x}$ can be classified using $\operatorname{sign}(\langle \underline{w}, \underline{x} \rangle)$ prediction

LP solver gives one solution

There may be many solutions

Generalization

S: training set

$$h^{*}_{\text{best}} = \arg\min_{h \in H} \text{error}_S(h)$$

ERM learning rule

M: minimization

Empirical Risk (expected loss)

training set