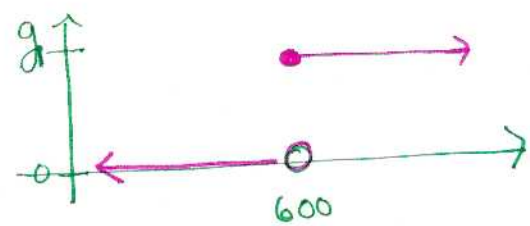
Math 342 (650, 4) || Theory Lecture # 3 || Feb. 8, 2021

$$\mathcal{H} = \{ \mathbb{1}_{x \geq \theta} : \theta \in \Theta \}$$

model parameter (θ); parameter space (Θ)

ex.



prediction: $\hat{y} = g(\vec{x})$

$$y = g(\vec{x}) + e = \hat{y} + e = \hat{y} + \underbrace{(y - \hat{y})}_{=e}$$

The algorithm $\mathcal{A}$ produces g.
Since g is fully specified by theta, the algorithm selects/estimates/optimizes/fits a theta.

Let's create an algorithm.

A bad algorithm ~~with~~ will have high estimation error.

y \ $\hat{y}$	0	1
0	0	-1
1	+1	0

e

Let's define an overall error fth/ objective fth called "misclassification error" (ME)

$$\left[ME = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{g(\vec{x}_i) \neq y_i} = \frac{1}{n} \sum_{i=1}^{n} |e_i| \right]$$

or accuracy (ACC) as

$$\left[ACC = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{g(\vec{x}_i) = y_i} = 1 - ME \right]$$

Now, goal of the algorithm is to minimize ME (or maximize ACC).

To do so, we check every possible $\theta \in \Theta$ and keep track of the ME(theta) and then return the model with the lowest ME.

How to define parameter space?

It must be finite because we need to check (i.e. compute ME) each element.

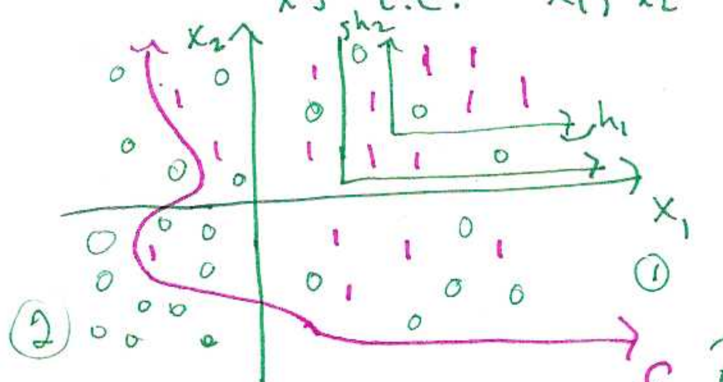
Gabriel says grid up [300, 850].

That's fine, but it's more convenient to only check the unique values of x.

A produces $g(x) = \mathbb{1}_{x \geq \underset{\theta \in \text{unique}(\vec{x})}{\text{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\mathbb{1}_{x_i \geq \theta} \neq y_i} \right\}}$

a for loop

Let's make a loan model w/ two continuous x's i.e. x_1, x_2 $(p=2)$



$\dim[\Theta] = 2 = p$

A two-dim threshold model extending what we have before has candidate set:

f $\mathcal{H} = \left\{ \mathbb{1}_{x_1 \geq \theta_1} \mathbb{1}_{x_2 \geq \theta_2} : \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} \in \Theta \right\}$

This candidate set of "angle bracket"-looking things is very restrictive!
This means we will (probably) have high model misspecification error

· So, let's use another hypothesis set:
all lines.

$$\mathcal{H} = \{ \mathbb{1}_{x_2 \geq a + b x_1} : a \in \mathbb{R}, b \in \mathbb{R} \}$$

interupt slope

The slope and intercept provide you w/ enough "degrees of freedom" to specify any separating line.

We need an algorithm to find g
i.e. specify a and b.

[This is a hard problem, so we will study it w/ different conditions.]

FIRST we will reparameterize the hypothesis space to be:

$$\mathcal{H} = \{ \mathbb{1}_{w_0 + w_1 x_1 + w_2 x_2 \geq 0} : w_0 \in \mathbb{R}, w_1 \in \mathbb{R}, w_2 \in \mathbb{R} \}$$

$$\text{"} \mathbb{1}_{\vec{w} \cdot \vec{x} \geq 0}$$

Intercept term or bias

weight of: first feature, second feature

Notes: [In order to fit this model, we "add" a dummy value of 1 to each dat record:

$$\vec{x} = [750 \quad \$58000] \longrightarrow \vec{x} = [1 \quad 750 \quad \$58000]$$

· So, we append the $\vec{1}$, the n-dim column vector to X, the matrix of features in $\mathbb{D}$.

· We only need 2 parameters (a, b) but we have three (w_0, w_1, w_2) ⇒ hence we are "over-parameterized"
⇒ we have infinite sols.

$$\left[\; \mathbb{1}_{\vec{w}\cdot\vec{x}\geq 0} = \mathbb{1}_{c\vec{w}\cdot\vec{x}\geq 0} \quad \forall c \neq 0 \qquad x_1 + x_2 = 7 \;\right]$$

$\uparrow$ pin

A: Find w_0, w_1, w_2 to minimize ME

i.e. $$\vec{w}_* := \underset{\vec{w}\in\mathbb{R}^3}{\text{argmin}} \left\{ \sum_{i=1}^{n} \mathbb{1}_{\mathbb{1}_{\vec{w}\cdot\vec{x}_i \geq 0} = y_i} \right\}$$

$$= \text{argmin}\{ME\}$$

We have a problem here:

There is no analytic solution (of the indicator ftn).

We <u>need</u> a way to search over all possible lines.

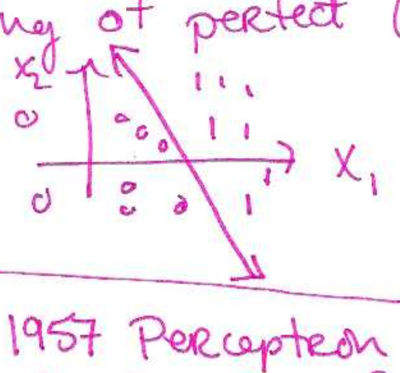
So (1) we need to reduce the # of lines

(2) use an iterative algorithm to find a local solution (not best but hopefully pretty good)

(3) Change our objective ftn.

In the setting of perfect linear separability:

e.g.



where ME of that linear discrimination model is zero (i.e. no errors)

Consider the 1957 Perceptron iterative algorithm for p features:

Note: t is the iteration value.

$\boxed{\text{STEP 1}}$: Initialize

$$\vec{W}^{t=0} = \vec{O}_{p+1}$$ OR to a random vector value.

$\boxed{\text{STEP 2}}$: Compute $\hat{y}_i = \mathbb{1}_{\vec{w}^{t=0} \cdot \vec{x}_i \geq 0}$

$\boxed{\text{STEP 3}}$: For $j = 0, 1, \ldots, p$

set
$$w_0^{t=1} = w_0^{t=0} + \underbrace{(y_i - \hat{y}_i)}_{e_i}(1)$$
$$w_1^{t=1} = w_1^{t=1} + (y_i - \hat{y}_i)(x_{i,1})$$
$$\vdots$$
$$w_p^{t=1} = w_p^{t=0} + (y_i - \hat{y}_i)(x_{i,p})$$

$\boxed{\text{STEP 4}}$: Repeat Steps 2 and 3 for $i = 1, \ldots, n$ (all the observations)

$\boxed{\text{STEP 5}}$: Repeat steps 2,3 and 4 until

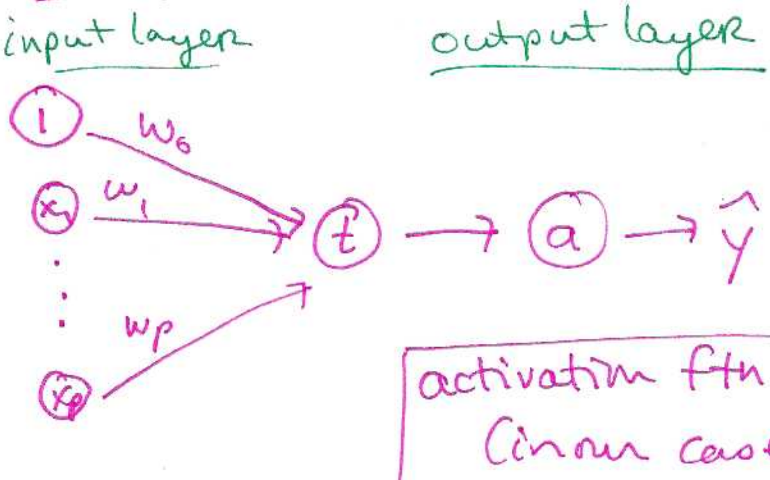
$ME = 0$ i.e.

e_i's $= 0$ or

until a ~~prescribed~~ prespecified (large) number of iterations.

The perceptron is proved to converge for linearly separable datasets, but for non-linearly separable datasets, anything can happen. Uh-Oh! So, it may fail.

Diagram : Perceptron

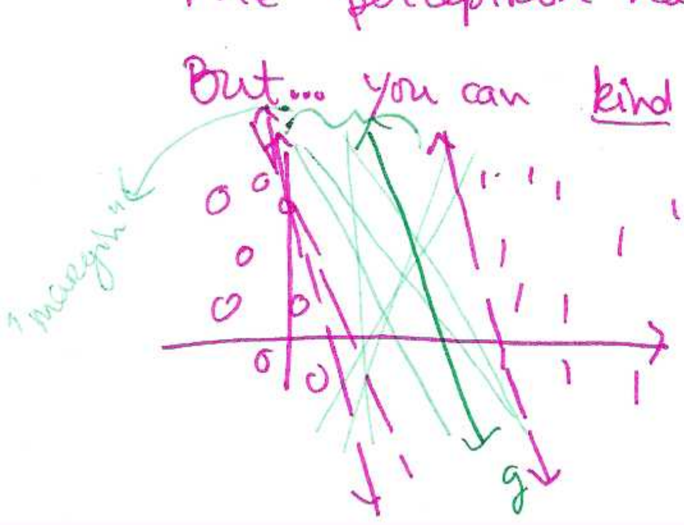


activation ftn
(in our case the
Heavyside indicator
ftn)

The perceptron is a type of "neural network" model.
So, ~~they~~ are deep learning models.
They're called neurons since the kind of act like neurons (in biology):

[Insert ~~...~~ nerve picture here]
aka: see prof's notes (:

The perceptron has <u>infinitely many</u> solutions.
But... you can <u>kind</u> of see there is a "best" model (g).



This "best model" is called the "maximum margin hyperplane" and it was proven in 1998 to be optimal linear classifier.