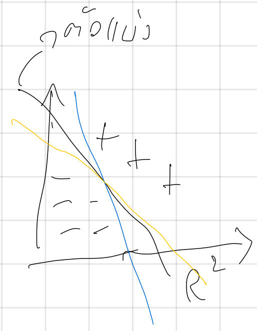
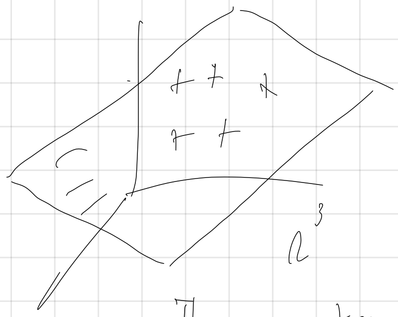
The Perceptron

- **Intro**: - The first consider learning algorithm
  - Assumption:
    - ข้อมูลต้อง Binary classification
      $y_i \in \{-1, +1\}$ #
    - Data is linearly separable.





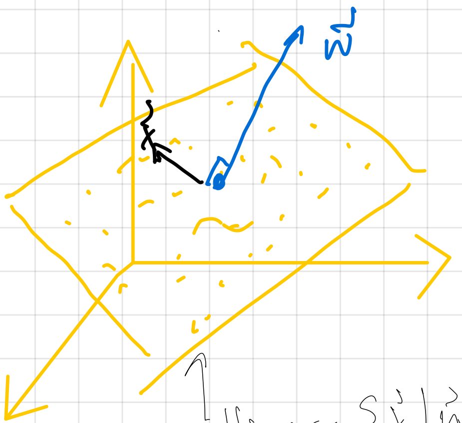
- The perceptron will try to lear a hyperplane that separate between the data

Hyper plane

- a sub space whose dimension is one less than the dimesion of ambient ~~space~~
  space ที่ใหญ่กว่า

$X = \mathbb{R}^d \quad \text{Hyper} \subseteq \mathbb{R}^{d-1}$

※ for high dimensional data จะทำงานได้ดี, เพราะ data point มีแยกห่างจน สามารถตั้ง hyper plane ได้

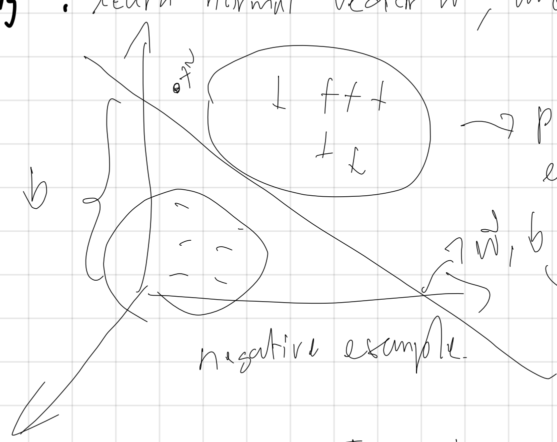


$\vec{w}^T \vec{x}_{th} = 0$

$$\begin{bmatrix} w^1 \\ w^2 \\ w^3 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} w^1x^1 + x^2x^2 + w^3x^3 \end{bmatrix} = 0$$

$\text{Hyper} = \{\vec{x} \mid \vec{w}^T \vec{x}_{th} = 0\}$ (bias) learn $\vec{w}$ (normal vector to hyperplane)

Traning: learn normal vector $\vec{w}$, and bias term b form date



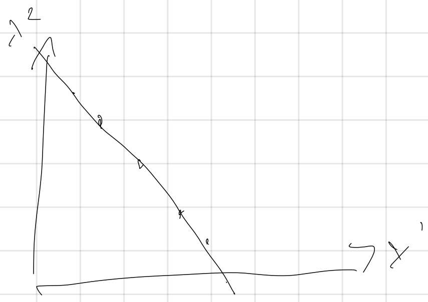
→ positive example

$\vec{w}, b$ → ค่าปรับระดับของ plane

testing: $h(x) = \text{sign}(\vec{w}^T \vec{x} + b)$

ดูสัญลักษณ์: ถ้า $= 0$ อยู่บน hyperplane $(-1, +1)$
> 0 positive
< 0 Negative

Point of hyperplane



$x^2 = -2x^1 + 10$

$y = -2x + 10$

$\underbrace{\vec{w}^T \vec{x}}_{\text{inner product}} + b = 0$

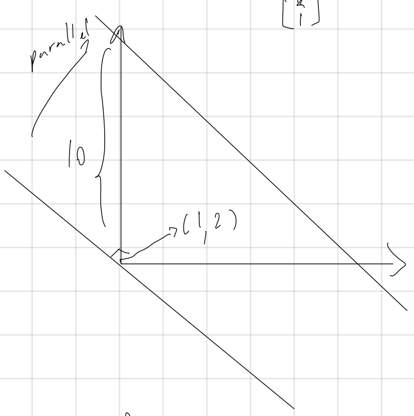
$[w_1\ w_2] \begin{bmatrix} x^1 \\ x^2 \end{bmatrix} + b = 0$

$w_1 x^1 + w_2 x^2 + b = 0 \quad = x^2 = -2x^1 + 10$

$= x^2 + 2x^1 - 10$

$w_1 = 2 \quad w_2 = 1 \quad b = -10$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$



2) $\vec{x}_{test} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$; $h(\vec{x}_{test}) = ?$

$h(\vec{x}_{test}) = \text{sign}(\vec{w}^T \vec{x}_{test} + b)$

$= \text{sign}\left([2\ 1]\begin{bmatrix} 2 \\ 3 \end{bmatrix} + (-10)\right)$

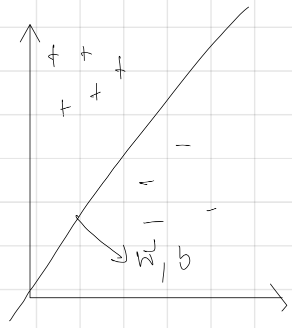
$= \text{sign}(\cancel{10} + (-10)) = 0$ = Negative

เหลือแต่เรา

$\text{sign}(4 + 3 + (-10))$ = neg..

Perceptron

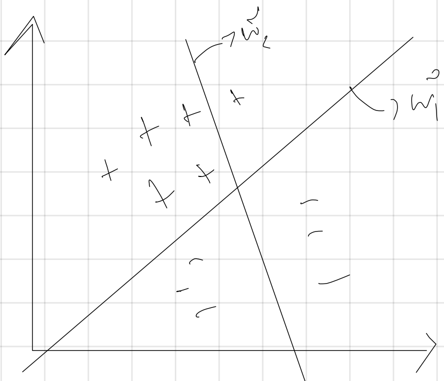
- $y_i \in \{-1, +1\}$
- D must be linearly separable
- Testing : $\text{sign}(\vec{w}^T\vec{x} + b)$
- Training: learn to find $\vec{w}, b$ from D



How to learn $\vec{w}, b$?

$\vec{w} = \begin{bmatrix} w^1 \\ w^2 \\ \vdots \\ w^d \end{bmatrix}$; $\vec{x} = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^n \end{bmatrix}$ $= \vec{w}^T\vec{x} + b = \underbrace{w^1x^1 + w^2x^2 \dots w^dx^d + b(1)}_{\vec{w}' = \begin{bmatrix} w^1 \\ w^2 \\ \vdots \\ w^d \\ b \end{bmatrix} \quad \vec{x}' = \begin{bmatrix} x^1 \\ \vdots \\ x^d \\ 1 \end{bmatrix}}$

* $\vec{w}'\vec{x}'$



- $\vec{w}_1$ is the right normal vector but $\vec{w}_2$ isn't

$\vec{w}^T\vec{x} \to > 0$ ⊕ predict
$\to \leq 0$ ⊖ predict

$y(\vec{w}^T\vec{x})$ สมมุติ prediction ⊕, y⊕ > 0
true label — predict
⊕, y⊖ ≤ 0 ✗ เอา ⊖ * ⊕ ; misclassification
⊖, y⊖ > 0

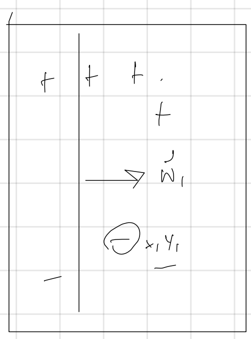
$\forall (x_i, y_i) \in D$, $y_i \underbrace{(\vec{w}_1^T\vec{x}_i)}_{\text{classify correctly for any data set in D}} > 0$ → $\vec{w}_1^T\vec{x}_i$ มาจาก $(\vec{w}_1^T\vec{x}_i + b)$ ย่อจากด้านบน
นับทุก $x_i\, y_i$ จาก D
สมการ train

Misclassification : $\exists (x_i, y_i) \in D,\ y_i(\vec{w}^T\vec{x}_i) \leq 0$

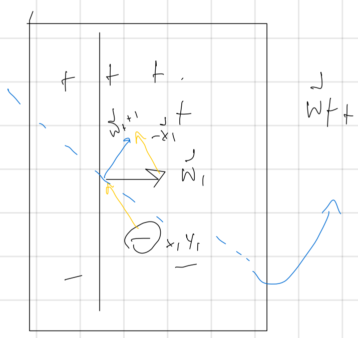
$\Rightarrow \vec{w}$ is not the right normal vector

Perceptron algorithm:

```
w ← 0          ; |w| = d+1
while true:
    m ← 0     // counter to count # of missclassif..
    for (xi, yi) ∈ D:
        if yi (w^T xi) ≤ 0:  // if miss
            m ← m+1       // update counter m
            [w ← w + yi·xi]   // update w
    if m = 0:
        break
```



$\underbrace{y_1}_{<0}\underbrace{(\vec{w}_1^T x_1)}_{>0} \leq 0$; miss



Update (Rotate)

$\vec{w}_{t+1} \leftarrow \underbrace{\vec{w}_t + (-1)(\vec{x}_1)}_{\vec{w}_t - \vec{x}_1}$

โกดที่ไปโดนสายไฟเหรอ?

OK

ชูสองนิ้ว = ไม่โกรธ

สรุป:

- Assume D is linearly separable
- The perceptron algorithm will find a hyperplane that separates the data
- $y \Rightarrow$ binary classification

$(\vec{x}_i, y_i) \in D \longrightarrow \boxed{\text{learn}} \longrightarrow (\vec{w}, b)$ $\quad \vec{w} \in \mathbb{R}^{d+1}$

$\vec{x}_i \in \mathbb{R}^{d+1}$

bias ใส่ที่ได้ $\rightarrow \vec{w}$

$$\vec{x}_i \text{ ใหม่} = \begin{bmatrix} x_i^1 \\ x_i^2 \\ \vdots \\ x_i^d \\ 1 \end{bmatrix}$$

Algorithm: try to adjust the vector $\vec{w}$ to satisfy

$$\boxed{\forall (x_i, y_i) \in D,\ y_i(\vec{w}^T \vec{x}_i) > 0}$$

- update (when misclassifying): $y_i(\vec{w}^T \vec{x}_i) \le 0$
  - $\vec{w} \leftarrow \vec{w} + y_i \vec{x}_i$

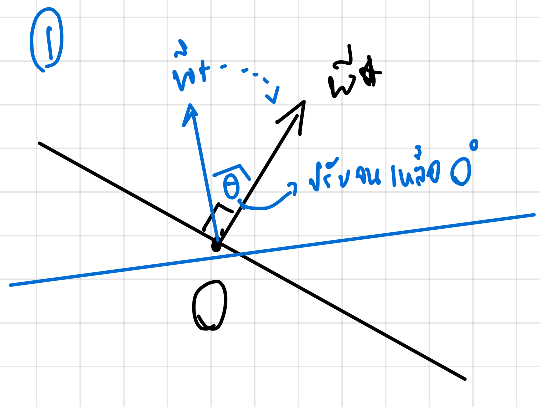
Q: when the algorithm terminates how can we be sure that $\vec{w}$ defines the right hyperplane?

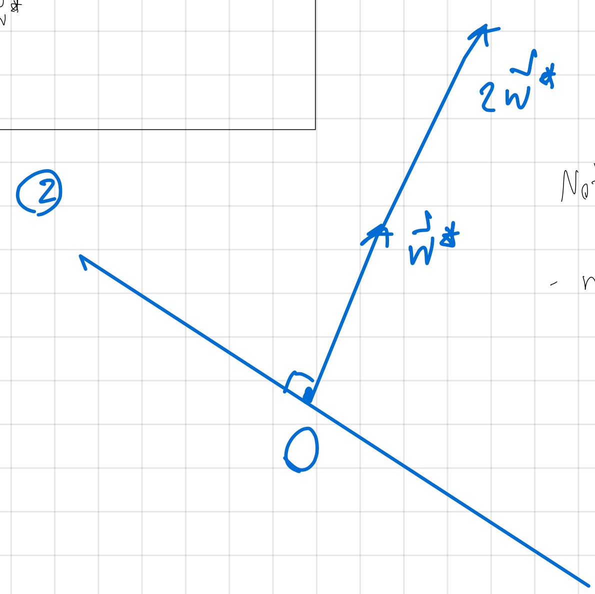
Perceptron's covergence Proof:

- provide strong formal guarantee of the separating hyperplane by the algorithm
- setup: - Assumption: D is linearly separable

- Equalvanlently, there exists $\vec{w}^*$ such that
  - define right hyperplane (เราอยากได้)
  - $\vec{w}$ vector ใน algorithms จะ converge เข้าใกล้ $\vec{w}^*$

$\forall (x_i, y_i) \in D,\ y_i(\vec{w}^{*T} \vec{x}_i) > 0$

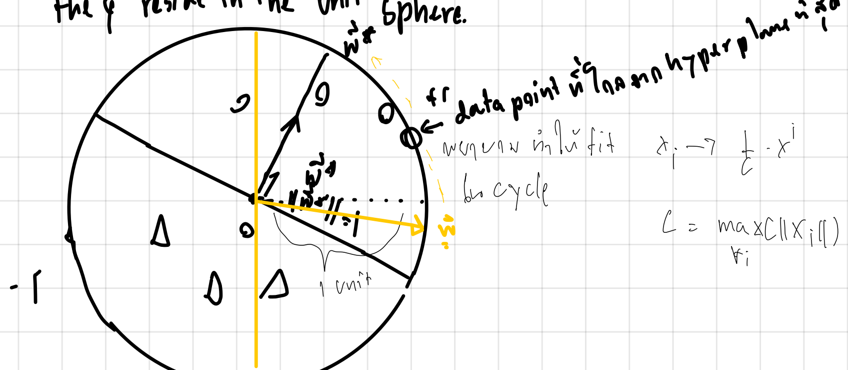




Note: there are infinitely many such $\vec{w}^*$

- we will focus on $\vec{w}^*$ with $\|\vec{w}^*\| = 1$ (unique)

- Futhermore, we consider the rescaling of all the data points in D so that they reside in the unit sphere.



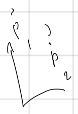
$x_i \rightarrow \frac{1}{C} \cdot x^i$

$C = \max_{\forall i}(\|x_i\|)$

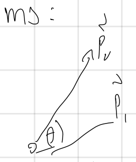
Effect caused by an update:

- Each update is made in hope of tuning $\vec{w}$ to ward $\vec{w}^*$
- to measure how much the vector $\vec{w}$ chaning to wards $\vec{w}^*$,

we consider two terms:



① $\vec{w}^T \vec{w}^*$

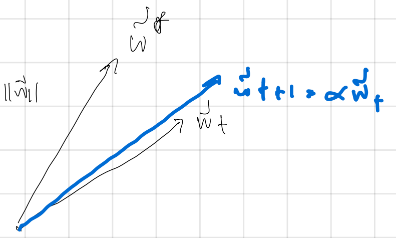


$\vec{p}_1^T p_2$ = มากสุด วัดความใกล้

(closeness between $\vec{w}$ and $\vec{w}^*$, when $\vec{w} = \vec{w}^*$
$\Rightarrow \vec{w}^T \vec{w}^*$ is at maximum)

② $\vec{w}^T \vec{w}$

$\alpha \sqrt{\vec{w}^T \vec{w}} = \|\vec{w}\|$



① เพิ่มเยอะ ② เพิ่มน้อย $\Rightarrow$ เปลี่ยน Direction ใกล้ $\vec{w}^*$

analyze ① $\vec{w}^T \vec{w}^*$ $\quad \vec{w} \leftarrow \vec{w} + y_i x_i$

$$\vec{w}^T \vec{w}^* \rightarrow (\vec{w} + y\vec{x})^T \vec{w}^*$$

$$\vec{w}^T \vec{w}^* \rightarrow \vec{w}^T \vec{w}^* + \underbrace{y\vec{x}^T \vec{w}^*}_{>0}$$

หนึ่งครั้งในการอัพเดท เปลี่ยนได้ปริมาณเท่า

$$\vec{w}^T \vec{w}^* = \vec{w}^T \vec{w}^* + \underbrace{y\vec{x}\vec{w}^*}_{\geq \gamma} \Rightarrow \vec{w}^T \vec{w}^* \geq \vec{w}^T \vec{w}^* + \gamma$$

margin:

$\gamma = \min_{\forall i} |x_i^T \vec{w}^*|$

ระยะทางจาก Vector $\vec{w}^*$ กับ x_i ที่น้อยสุด

② $\vec{w}^T \vec{w} \rightarrow (\vec{w} + y\vec{x})^T (\vec{w} + y\vec{x})$

$$\vec{w}^T \vec{w} \rightarrow \vec{w}^T \vec{w} + \underbrace{\boxed{2y\vec{x}^T \vec{w}}}_{\leq 0} + \underbrace{y^2 \vec{x}^T \vec{x}}_{1 \cdot \boxed{}\;\; 1 \cdot \leq 1}$$

$y \in \{-1, 1\}$

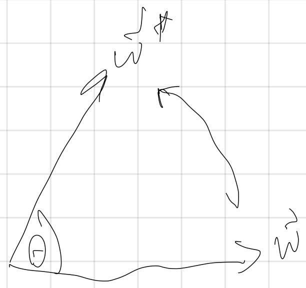
$\|x\| \leq 1$

$\|x\| = \sqrt{x^T x} \leq 1$

(w must misclassify on x)

$$\Rightarrow \vec{w}^T \vec{w} \leq \vec{w}^T \vec{w} + 1$$

for each update:



① $\vec{w}^T \vec{w}^* \geq \vec{w}^T \vec{w}^* + \gamma$

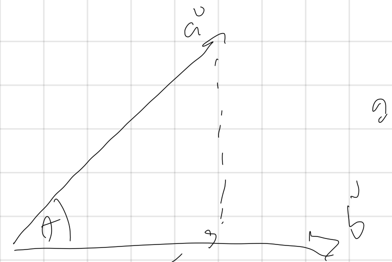
$\vec{w}^T \vec{w} \leq \vec{w}^T \vec{w} + 1$

Inner product / dot product

Inner: $\vec{a}, \vec{b} \in \mathbb{R}^d$

$$\vec{a} \circ \vec{b} \;/\; \vec{a}^T \vec{b} = \sum_{i=1}^{d} a_i b_i \quad \to \text{scalar}$$

Geometrically the inner product $\vec{a}^T\vec{b}$ can be used to measure how close the vector $\vec{a}$ to vector $\vec{b}$, and vice versa



คูณตรงๆ ไม่ได้
(อยู่คนละทิศ)

$\vec{a}_b = \|a\| \cos\theta$

θ is the angle between $\vec{a}, \vec{b}$

$\vec{a}^T \vec{b} = (\|\vec{a}\| \cos\theta)\vec{b}$

inner product $= (\|\vec{a}\|)(\|\vec{b}\|)\cos\theta$

$$\boxed{\vec{a}^T\vec{b} = \underbrace{\|\vec{a}\|\|\vec{b}\|}_{\text{constant}} \cos\theta}$$

$\theta = 0$

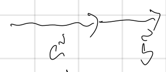
$\theta = 180^\circ (\pi)$

$-1 \le \cos\theta \le 1$

when $\vec{a}^T\vec{b}$ is at max

$\Rightarrow \cos\theta = 1 \Rightarrow \theta = 0$

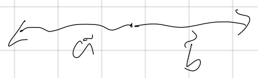
$\vec{a}, \vec{b}$ are in the same direction.



- when $\vec{a}^T\vec{b}$ is at min

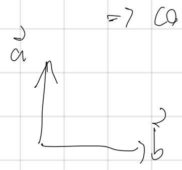
$\to \cos\theta = 0 \Rightarrow \theta = \pi$

$\vec{a}, \vec{b}$ are in opposite direction



- when $\vec{a}^T\vec{b}$ is zero

$\Rightarrow \cos\theta = 0 \to \theta = \frac{\pi}{2} \Rightarrow \vec{a}, \vec{b}$ are perpendicular / orthorgonal

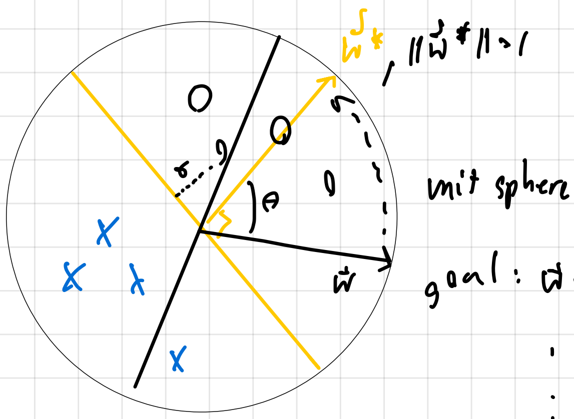


* $\theta = 0 \Rightarrow \cos\theta = 1 \Rightarrow \vec{a}^T\vec{b}$ is maximum

- Assumption : $\exists \vec{w}^*$ by $\forall (x_i, y_i) \in D$, $y_i(\vec{w}^{*T}\vec{x}_i) > 0$
  and $\|\vec{w}^*\| = 1$

- Transformation to unit sphere:
  $\forall (x_i, y_i) \in D$, $\|x_i\| \le 1$

- Define margin $\gamma = \min_{\forall i} |\vec{x}_i^T \vec{w}^*|$ (γ is the distance from the close point to the hyperplane.)



unit sphere

goal : $\vec{w} \to \vec{w}^*$

We consider 2 terms

① $\vec{w}^T\vec{w}^*$ (to be large)

② $\vec{w}^T\vec{w}$ (to be small)

Note :

$\vec{w}^T\vec{w}^* = \|\vec{w}\|\|\vec{w}^*\| \cos\theta$

$\cos\theta = \dfrac{\vec{w}^T\vec{w}^*}{\|\vec{w}\| \cancel{\|\vec{w}^*\|}_1}$

$\cos\theta = \dfrac{\vec{w}^T\vec{w}^*}{\sqrt{\vec{w}^T\vec{w}}} \approx 1 \qquad -1 \le \cos\theta \le 1$

For each update

① $\vec{w}^T\vec{w}^* \rightarrow \geq \vec{w}^T\vec{w}^* + \gamma$

(each update increase $\vec{w}^T\vec{w}^*$ by at least γ)

② $\vec{w}^T\vec{w} \rightarrow \leq \vec{w}^T\vec{w} + 1$

(each update increase $\vec{w}^T\vec{w}$ by at most 1)

Suppose after M updates, we have $\cos\theta = 1$

Then, - $\vec{w}^T\vec{w}^* \geq \gamma M$

- $\vec{w}^T\vec{w} \leq M$

$$\theta = 0 \Rightarrow \cos\theta = 1 = \frac{\vec{w}^T\vec{w}^*}{\sqrt{\vec{w}^T\vec{w}}} \geq \frac{\gamma M}{\sqrt{\vec{w}^T\vec{w}}} \geq \frac{\gamma M}{\sqrt{M}}$$

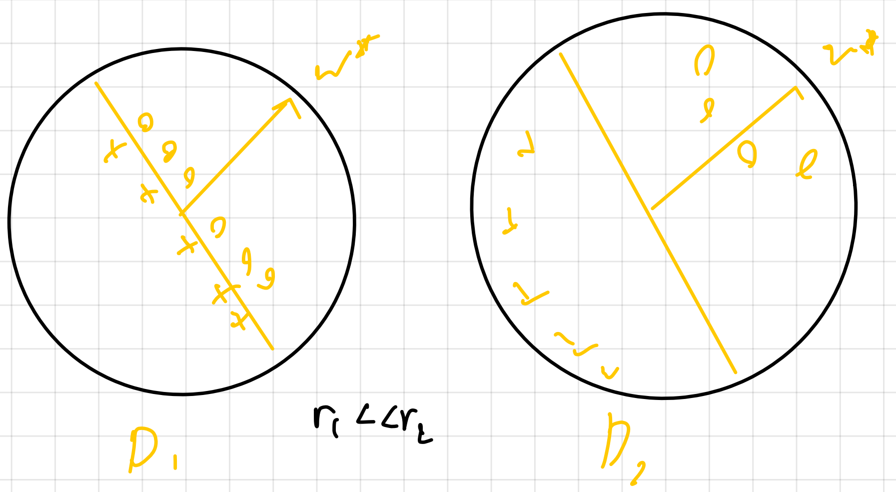
$$\Rightarrow 1 \geq \frac{\gamma M}{\sqrt{M}}$$

$$\sqrt{M} \geq \gamma M$$

$$M \geq \gamma^2 M^2$$

$$1 \geq \gamma^2 M \Rightarrow M \leq \frac{1}{\gamma^2}$$

Theorem : The perceptron will find $\vec{w}^*$ within at most $\frac{1}{\gamma^2}$ update.



$r_1 << r_2$

Q : ถ้ารัน perceptron บน D_1, D_2

แล้วเราจะหา $\vec{w}^*$ เจอใน D_1 / D_2 ก่อน

: เนื่องจาก γ ห่างจาก hyperplane จะหาได้เร็วกว่า จากสูตร $\frac{1}{\gamma^2}$

k-NN vs Perceptron

- for low-dimensional data, k-NN is very efficient.

for high-dimensional data, Perceptron is more suitable

- Limitations