Lecture 16: Kernel perceptron, Logistic Regression

Instructor: Prof. Ganesh Ramakrishnan

Classification: $f(\phi(x)) \rightarrow \{C_1, C_2 \ldots C_K\}$

often approximated as

$f_1(\phi(x)) \rightarrow \mathbb{R}$ often $[0,1]$

$f_2(\phi(x)) \rightarrow \mathbb{R}$

$f_K(\phi(x)) \rightarrow \mathbb{R}$

Relevance of example $\phi(x)$ to some C_K.

For $K=2$, $\{+1,-1\}$

$\{0,1\}$

are also used

Binary Classification using Perceptron

$\{-1, +1\}$ as 2 classes

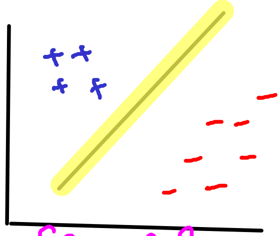
Perceptron Classifier

- Consider a binary classification problem: $f(x) \in \{-1, +1\}$ (If $f(x) \in \{C_1 .. C_K\}$ ⇒ Multiclass problem)
- Assuming linearly separability, is there a learning rule that converges in finite time?

linear in the space of ϕ

$$f(x) = \text{sign}\left(w^T \phi(x) + b\right)$$

Assume $\text{sign}(0) = +1$

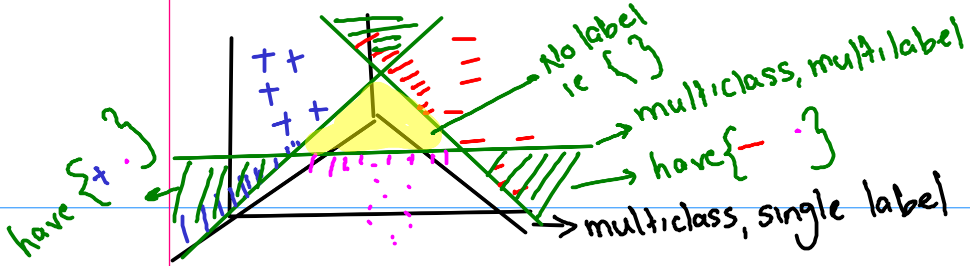


For case of multiple classes $\{C_1 . . C_K\}$

1. Learn $f_K(x)$ for each class C_K: $\underbrace{f_K(x) \in \{C_K, \neg C_K\}}$
   A common approach to mult-class multi-label classification

2. $\begin{bmatrix} 1 \\ \\ 1 \end{bmatrix}$ } K bit vector → Single | multiple bits turned on

   1 at k^{th} position means class C_K is correct for x



Identifying metals in an equipment: Multiclass, multilabel problem

Copper vs Brass: Binary classification

Copper vs Brass vs Bronze: Multiclass classification (single label)

Identify Uranium : Positive unlabeled learning (PU)

(or rare disease identification)

Perceptron Classifier

- Consider a binary classification problem: $f(\mathbf{x}) \in \{-1, +1\}$
- Assuming linearly separability, is there a learning rule that converges in finite time?
- Naive Idea: Perform linear regression by constraining $y \in \{+1, -1\}$.

$$w_{ridge} = (\phi^T\phi + \lambda I)^{-1}\phi^T y$$

$$4 \quad f(x) = \text{sign}(w_{ridge}^T \phi(x))$$

Problems?

1. Training y is pretended to be real valued but for inference $\text{sign}(.)$ is used to predict y

2. $y_4 = 500 \quad y_3 = 1$

   $y_2 = -1 \quad y_1 = -500$

Training: $\text{dist}(y_4, y_3) >> \text{dist}(y_3, y_2) \Longrightarrow$ Result in large # errors!

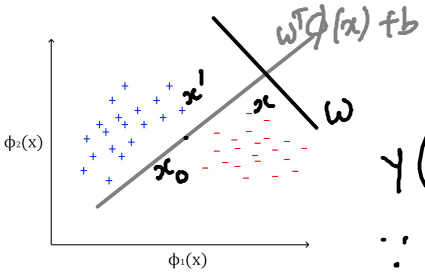
Test: $\text{dist}(y_4, y_3) < \text{dist}(y_3, y_2)$ [dist is $\text{sign}(.)$]

Perceptron Classifier

- Consider a binary classification problem: $f(\mathbf{x}) \in \{-1, +1\}$ $y \in \{-1, +1\}$
- Assuming linearly separability, is there a learning rule that converges in finite time?
- Naive Idea: Perform linear regression by constraining $y \in \{+1, -1\}$.
- Can we do better? What is ideal?
- Desirable: Any new (unseen) input pattern similar to a seen pattern is **classified** correctly

ie in same half-space

(what is quantitative measure of half space?)



$$y\left(w^T\left(\phi(x) - \phi(x_0)\right)\right) = y\left(w^T\phi(x) + b\right)$$

$$\therefore\ w^T\phi(x_0) + b = 0 \Rightarrow w^T\phi(x_0) = -b$$

Perceptron Classifier

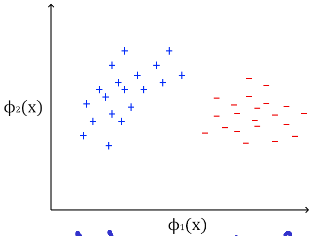
- Consider a binary classification problem: $f(\mathbf{x}) \in \{-1, +1\}$
- Assuming linearly separability, is there a learning rule that converges in finite time?
- Naive Idea: Perform linear regression by constraining $y \in \{+1, -1\}$.
- Can we do better? What is ideal?
- Desirable: Any new (unseen) input pattern similar to a seen pattern is **classified** correctly

Called signed distance

$$\forall\ y=+1,\quad w^T\phi(x)+b \geq 0$$

$$y=-1,\quad w^T\phi(x)+b \leq 0$$

Combining



Linear Classification?

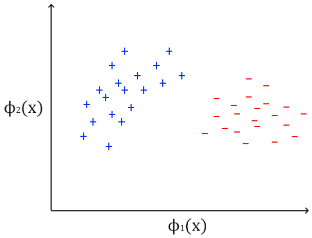
$$y\,(w^T\phi(x)+b) \geq 0$$

Supposed to be absolute (+ve) distance & called UNSIGNED distance

Perceptron Classifier

- Consider a binary classification problem: $f(\mathbf{x}) \in \{-1, +1\}$
- Assuming linearly separability, is there a learning rule that converges in finite time?
- Naive Idea: Perform linear regression by constraining $y \in \{+1, -1\}$.
- Can we do better? What is ideal?
- Desirable: Any new (unseen) input pattern similar to a seen pattern is **classified** correctly

Any point with negative value of unsigned distance $y(w^T\phi(x)+b)$ must be MISCLASSIFIED



Linear Classification?

$\mathbf{w}^\top \phi(\mathbf{x}) + b \geq 0$ *for +ve points* ($y = +1$)

$\mathbf{w}^\top \phi(\mathbf{x}) + b < 0$ *for -ve points* ($y = -1$)

$\mathbf{w}, \phi \in \mathbb{R}^m$

Perceptron Classifier: Setting up Notation

- Often, b is indirectly captured by including it in $\mathbf{w}$, and using a ϕ as: $\phi_{aug} = [\phi, 1]$
- Thus, $\mathbf{w}^\top \phi(\mathbf{w})$

$$= \begin{bmatrix} w_1 & w_2 & w_3 & \dots & w_m & b \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_m \\ 1 \end{bmatrix}$$

- $\mathbf{w}^\top \phi(\mathbf{x}) = 0$ is the separating hyperplane.

($y w^T \phi(x)$ is unsigned distance & perceptron algo tries to make each $y w^T \phi(x)$ increasingly positive)

Perceptron Intuition

Verify that for this $\phi(x)$, perceptron update makes unsigned distance increasingly positive:

1. Go over all the existing examples, whose class is known, and check their classification with the current weight vector
2. If correct, continue
3. If not, marginally correct the weights
   - By adding to the weights a quantity that is proportional to the product of the input pattern with the desired output $y = \pm 1$

ie $y(w^T\phi(x)) < 0$ $\quad w^{(k+1)} = w^{(k)} + \eta y \phi(x) \quad (\eta > 0)$

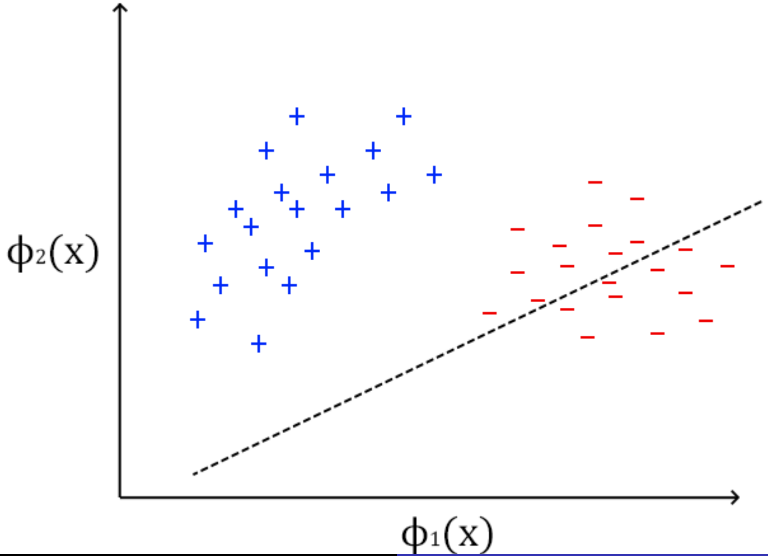
$$y\left(w^{(k+1)^T}\phi(x)\right) = y\left(w^{(k)^T}\phi(x) + \eta y \phi^T(x)\phi(x)\right) = y w^{(k)^T}\phi(x) + \underbrace{y^2 \eta \phi^T(x)\phi(x)}_{\eta\|\phi(x)\|^2 \geq 0}$$

Perceptron Update Rule

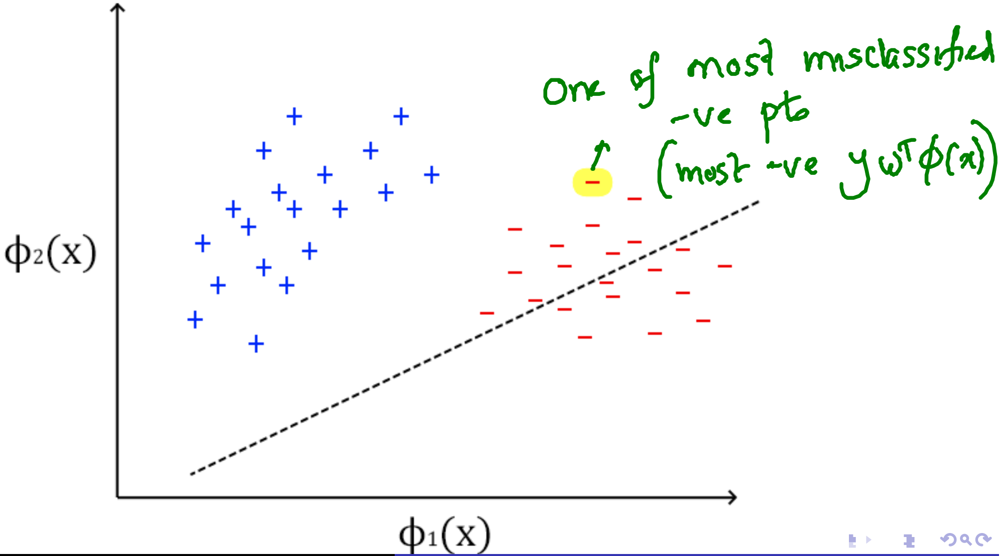
- Start with some weight vector $\mathbf{w}^{(0)}$, and for $k = 0, 1, 2, 3, \ldots, n$ (for every example), do:
  $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + y'\phi(x')$
- where $\mathbf{x}'$ s.t. $\mathbf{x}'$ is misclassified by $(\mathbf{w}^{(k)})^\top \phi(\mathbf{x})$
  i.e. $y'(\mathbf{w}^{(k)})^\top \phi(\mathbf{x}') < 0$

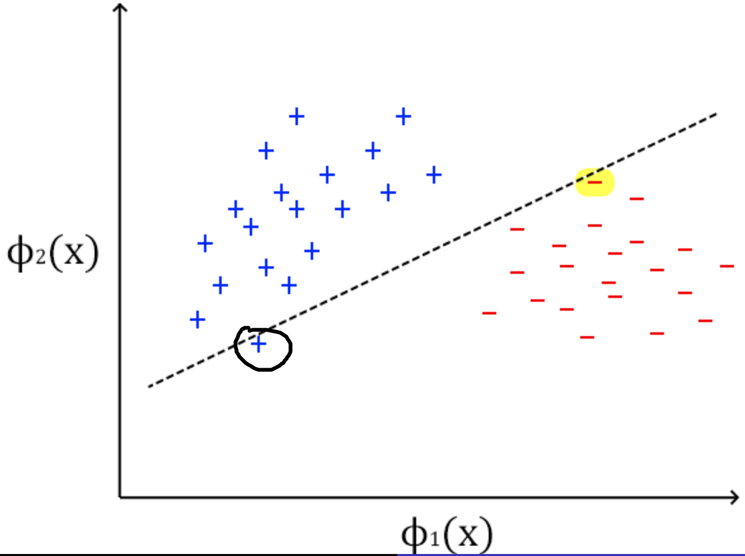
Assured $\quad y'(w^{(k+1)})^T \phi(x') \geq y'(w^{(k)})^T \phi(x')$

Q: Is it assured of convergence? Will it keep oscillating?

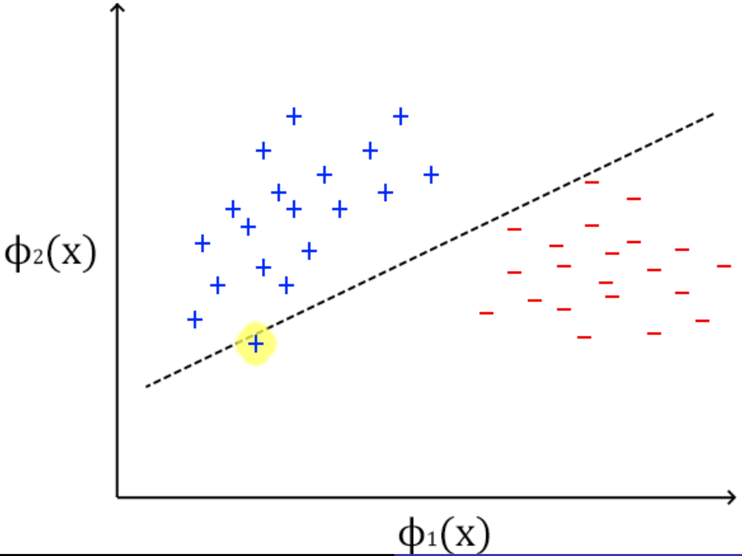


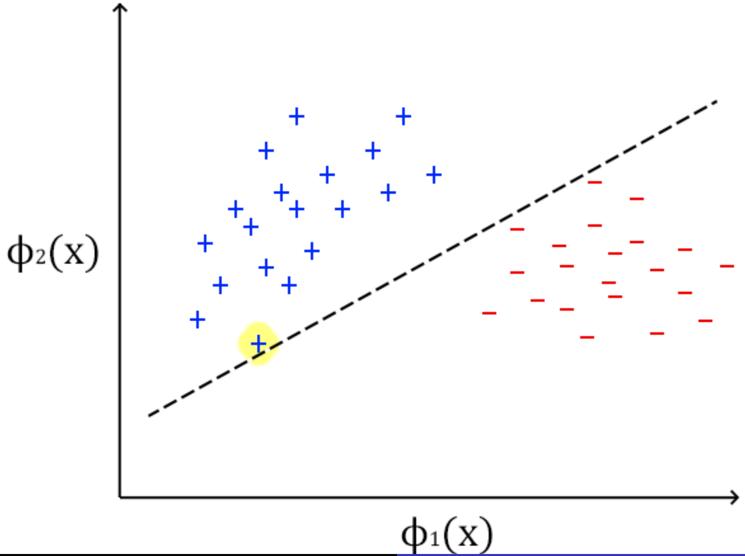
$\phi_2(x)$
$\phi_1(x)$


One of most misclassified -ve pts
(most -ve $y w^T \phi(x)$)
$\phi_2(x)$
$\phi_1(x)$



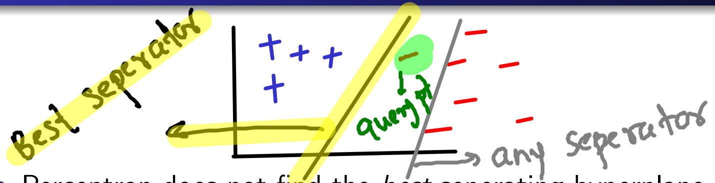
$\phi_2(x)$
$\phi_1(x)$


$\phi_2(x)$
$\phi_1(x)$



$\phi_2(x)$
$\phi_1(x)$

Some Notes about the Perceptron Algorithm



- Perceptron does not find the *best* seperating hyperplane, it finds *any* seperating hyperplane.
- In case the initial **w** does not classify all the examples, the seperating hyperplane corresponding to the final $\mathbf{w}^*$ will often *pass through an example*.
- The seperating hyperplane does not provide enough breathing space – this is what SVMs address!

Margin leads to good generalization

Perceptron Update Rule: Basic Idea

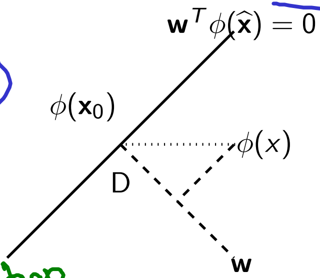
- Perceptron works for two classes ($y = \pm 1$). A point is misclassified if $y\mathbf{w}^T(\phi(\mathbf{x})) < 0$
- Perceptron Algorithm:
  - INITIALIZE: w=ones()
  - REPEAT: for each $< \mathbf{x}, y >$
    - If $y\mathbf{w}^T\Phi(\mathbf{x}) < 0$
    - then, $\mathbf{w} = \mathbf{w} + \eta\phi(\mathbf{x}).y$
    - endif
- **Intuition:**

$$\begin{aligned} y(\mathbf{w}^{(k+1)})^T\phi(\mathbf{x}) &= y\left(\mathbf{w}^k + \eta y\phi^T(\mathbf{x})\right)\phi(\mathbf{x}) \\ &= y(\mathbf{w}^k)^T\phi(\mathbf{x}) + \eta y^2\|\phi(\mathbf{w})\|^2 \\ &> y(\mathbf{w}^k)^T\phi(\mathbf{x}) \end{aligned}$$

Since $y(\mathbf{w}^k)^T\phi(\mathbf{x}) \leq 0$, we have $y(\mathbf{w}^{(k+1)})^T\phi(\mathbf{x}) \;>\; y(\mathbf{w}^k)^T\phi(\mathbf{x}) \Rightarrow$ more hope that this point is classified correctly now.

Perceptron Update Rule: Error Perspective

- Explicitly account for signed distance of (misclassified) points from the hyperplane $\mathbf{w}^T\phi(\widehat{\mathbf{x}}) = 0$. Consider point $\mathbf{x}_0$ such that $\mathbf{w}^T(\phi(\mathbf{x}_0)) = 0$
- (Signed) Distance from hyperplane is: $\mathbf{w}^T(\phi(\mathbf{x}) - \phi(\mathbf{x}_0)) = \mathbf{w}^T(\phi(\mathbf{x}))$
- Unsigned distance from hyperplane is: $y\mathbf{w}^T(\phi(\mathbf{x}))$ (assumes correct classification)



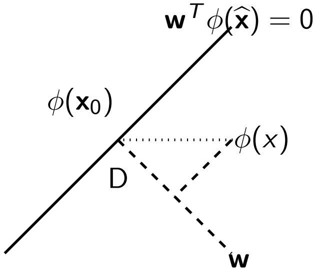
$-yw^T\phi(x)$ is error

$$\text{Error}(w^k) = \sum_{x \in M} -y(w^k)^T\phi(x)$$

Set of misclassified pts as index set in summation makes problem combinatorial (hard)

Perceptron Update Rule: Error Perspective

- Explicitly account for signed distance of (misclassified) points from the hyperplane $\mathbf{w}^T\phi(\widehat{\mathbf{x}}) = 0$. Consider point $\mathbf{x}_0$ such that $\mathbf{w}^T(\phi(\mathbf{x}_0)) = 0$
- (Signed) Distance from hyperplane is: $\mathbf{w}^T(\phi(\mathbf{x}) - \phi(\mathbf{x}_0)) = \mathbf{w}^T(\phi(\mathbf{x}))$
- Unsigned distance from hyperplane is: $y\mathbf{w}^T(\phi(\mathbf{x}))$ (assumes correct classification)



- If $\mathbf{x}$ is misclassified, the $\underbrace{\text{misclassification cost}}_{\text{Error}}$ for $\mathbf{x}$ is $-y\mathbf{w}^T(\phi(\mathbf{x}))$

Perceptron Update Rule: Error Minimization

- Perceptron update tries to minimize the error function E = negative of sum of unsigned distances over misclassified examples = sum of misclassification costs

$$E = -\sum_{(\mathbf{x},y)\in\mathcal{M}} y\mathbf{w}^T\phi(\mathbf{x})$$

where $\mathcal{M} \subseteq \mathcal{D}$ is the set of misclassified examples.

- **Gradient Descent (Batch Perceptron) Algorithm**

$$w = w - \eta \nabla E(w) = w - \left(-\eta \sum_{(x,y)\in M} y\phi(x)\right)$$

$$w^{(k+1)} = w^{(k)} + \eta \sum_{(x,y)\in M} y\phi(x)$$

Perceptron update picks only one of them

Perceptron Update Rule: Error Minimization

- Perceptron update tries to minimize the error function E = negative of sum of unsigned distances over misclassified examples = sum of misclassification costs

$$E = -\sum_{(\mathbf{x},y)\in\mathcal{M}} y\mathbf{w}^T\phi(\mathbf{x})$$

where $\mathcal{M} \subseteq \mathcal{D}$ is the set of misclassified examples.

- **Gradient Descent (Batch Perceptron) Algorithm** $\nabla_{\mathbf{w}} E = -\sum_{(\mathbf{x},y)\in M} y\phi(\mathbf{x})$

Batch Stochastic

$$\mathbf{w}^{(k+1)} = \mathbf{w}^k - \eta\nabla_{\mathbf{w}} E$$
$$= \mathbf{w}^k + \eta \sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x})$$

Picking a single eg is stochastic gradient descent

Perceptron Update Rule: Error Minimization

- Batch update considers all misclassified points simultaneously

$$\begin{aligned}\mathbf{w}^{(k+1)} &= \mathbf{w}^k - \eta \nabla_{\mathbf{w}} E \\ &= \mathbf{w}^k + \eta \sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x})\end{aligned}$$

- Perceptron update $\Rightarrow$ *Stochastic Gradient Descent*:

$$\nabla_{\mathbf{w}} E = -\sum_{(\mathbf{x},y)\in\mathcal{M}} y\phi(\mathbf{x}) = -\sum_{(\mathbf{x},y)\in\mathcal{M}} \nabla_{\mathbf{w}} E(\mathbf{x}) \text{ s.t. } E(\mathbf{x}) = -y\mathbf{w}^T\phi(\mathbf{x})$$

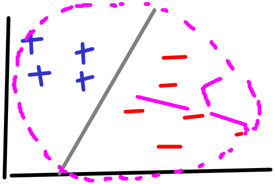
$$\begin{aligned}\mathbf{w}^{(k+1)} &= \mathbf{w}^k - \eta \nabla_w E(\mathbf{x}) \qquad\qquad (\text{for any } (\mathbf{x}, y) \in \mathcal{M}) \\ &= \mathbf{w}^k + \eta y\phi(\mathbf{x})\end{aligned}$$

Stochastic analysis & regret bounds. (Extra optional reading)

Perceptron Update Rule: Further analysis

- **Formally,**:- If $\exists$ an optimal separating hyperplane with parameters $\mathbf{w}^*$ such that,

$$\forall\ (\mathbf{x}, y),\ \ y\phi^T(\mathbf{x})\mathbf{w}^* \geq 0$$



then the perceptron algorithm converges.
Proof:- We want to show that

$$\lim_{k\to\infty} \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = 0 \qquad (1)$$

(If this happens for some constant ρ, we are fine.)

ie as $k \to \infty$ direction of $w^{(k+1)}$ tends to direction of w^*

ie ρ is independent of k

① $w^{(k+1)} = w^{(k)} + \eta y \phi(x)$ ② $y(w^{(k)})^T\phi(x) < 0$ ③ $\forall x\ \|\phi(x)\|^2 \leq r^2$

Perceptron Update Rule: Further analysis

- **Formally,**:- If $\exists$ an optimal separating hyperplane with parameters $\mathbf{w}^*$ such that,

$$\forall\ (\mathbf{x}, y),\ \ y\phi^T(\mathbf{x})\mathbf{w}^* \geq 0$$

then the perceptron algorithm converges.
Proof:- We want to show that

$$\lim_{k\to\infty} \|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = 0 \tag{1}$$

(If this happens for some constant ρ, we are fine.)

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 = \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 + \|y\phi(\mathbf{x})\|^2 + 2y(\mathbf{w}^k - \rho\mathbf{w}^*)^T\phi(\mathbf{x}) \tag{2}$$

- For convergence of perceptron, we need L.H.S. to be less than R.H.S. at every step, although by some small but non-zero value (with $\theta \neq 0$)

$$\|\mathbf{w}^{(k+1)} - \rho\mathbf{w}^*\|^2 \leq \|\mathbf{w}^k - \rho\mathbf{w}^*\|^2 - \theta^2 \tag{3}$$