# 人工智能
# ——人工神经网络 I


饶洋辉
数据科学与计算机学院,
中山大学
raoyangh@mail.sysu.edu.cn

# Perceptron Learning Algorithm

- Perceptron Learning Algorithm (感知机学习算法)：单层前馈神经网络
- Dealing with all attributes jointly which are continuous variables

# Perceptron Learning Algorithm

- Perceptron Learning Algorithm (感知机学习算法)：单层前馈神经网络
- Dealing with all attributes jointly which are continuous variables
- For $\mathbf{x}=(x_1,x_2,\ldots,x_d)$ with $d$ features, compute a weighted 'score' and
  predict +1(good) if $\sum_{k=1}^{d} w_k x_k > threshold$
  predict -1(bad) if $\sum_{k=1}^{d} w_k x_k < threshold$
- $\mathbf{y}$={+1(good), -1(bad)}

$$h(\mathbf{x}) = sign\left(\left(\sum_{k=1}^{d} w_k x_k\right) - threshold\right)$$

# Perceptron Learning Algorithm

$$h(\mathbf{x}) = sign\left(\left(\sum_{k=1}^{d} w_k x_k\right) - threshold\right)$$

$$= sign\left(\left(\sum_{k=1}^{d} w_k x_k\right) + \underbrace{(-threshold)}_{w_0} \cdot \underbrace{(+1)}_{x_0}\right)$$

$$= sign\left(\sum_{j=0}^{d} w_j x_j\right)$$

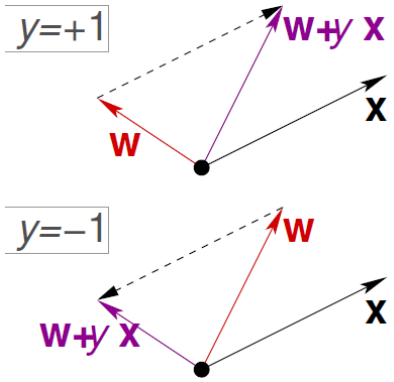
$$= sign\left(\tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}\right)$$

# Perceptron Learning Algorithm

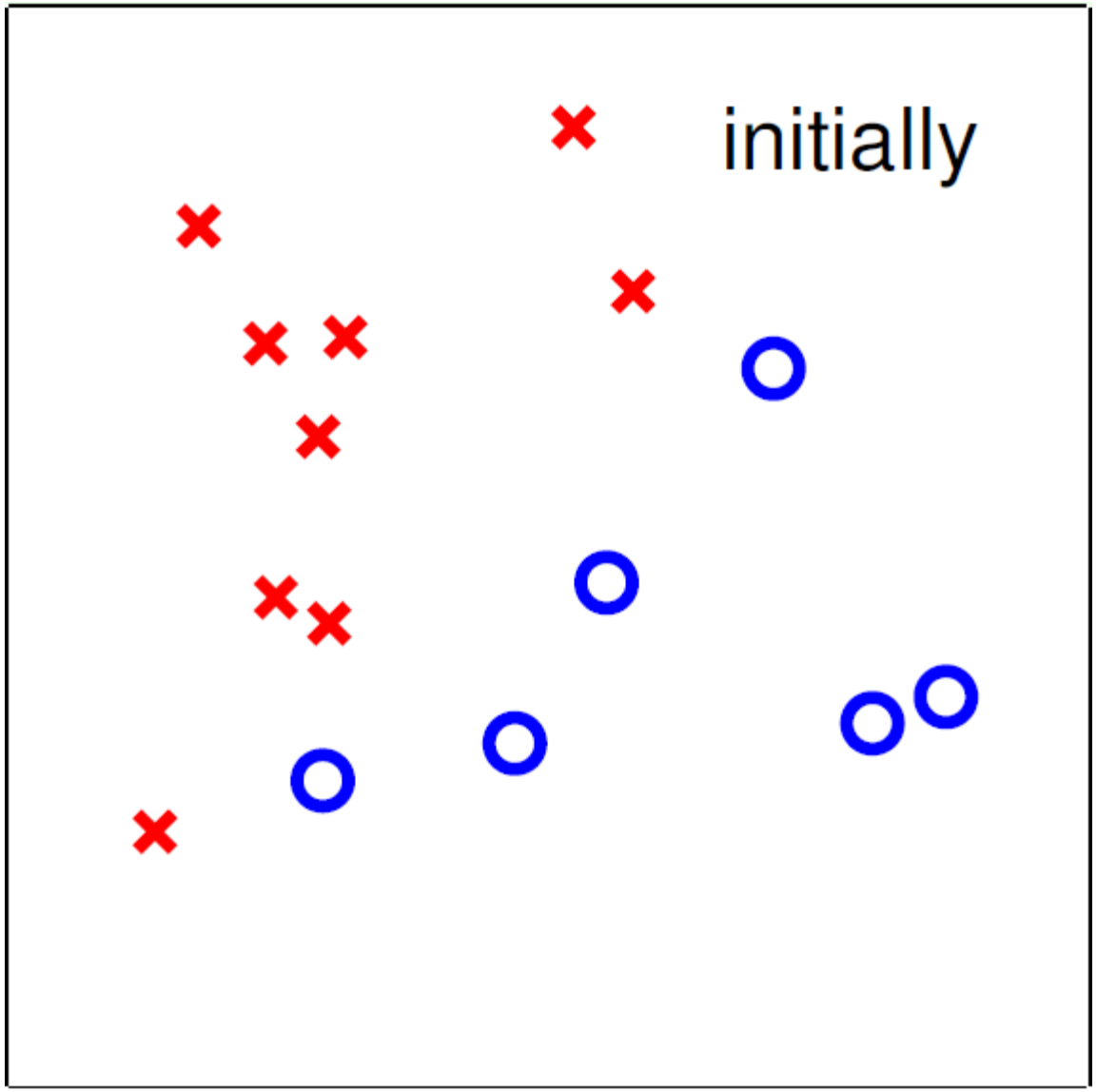
- Difficult: the set of $h(\mathbf{x})$ is of infinite size
- Idea: start from some initial weight vector $\mathbf{w}_{(0)}$, and "correct" its mistakes on $D$
- For t = 0, 1,…
  - find a mistake of $\mathbf{w}_{(t)}$ called $(\mathbf{x}_{i(t)}, y_{i(t)})$

    $sign\left(\tilde{\mathbf{w}}_{(t)}^{\mathrm{T}}\tilde{\mathbf{x}}_{i(t)}\right) \neq y_{i(t)}$
  - (try to) correct the mistake by

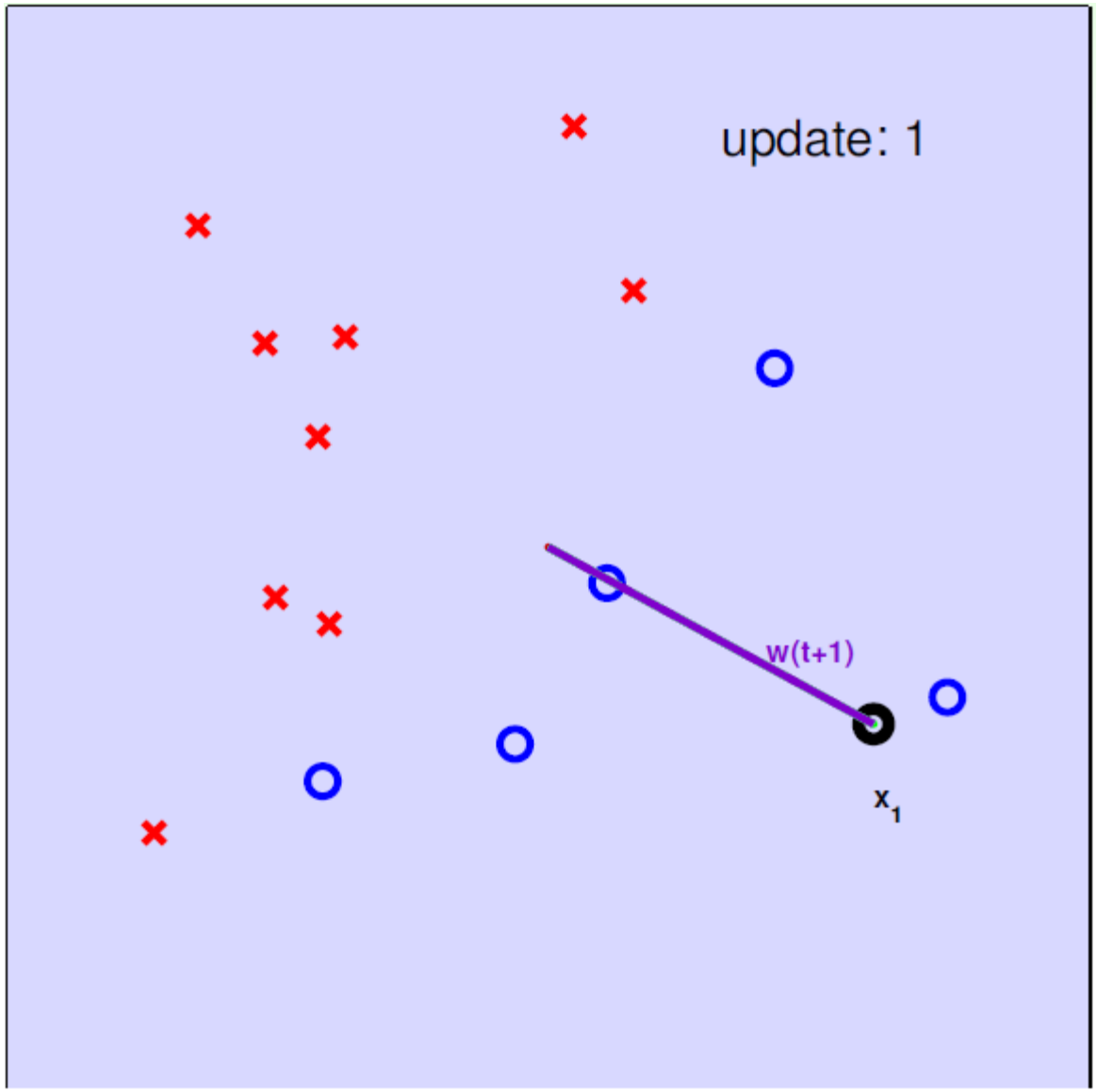
    $\tilde{\mathbf{w}}_{(t+1)} \leftarrow \tilde{\mathbf{w}}_{(t)} + y_{i(t)}\tilde{\mathbf{x}}_{i(t)}$
  - until no more mistakes
- Return last $\mathbf{W}$ (called $\mathbf{W}_{\mathrm{PLA}}$)



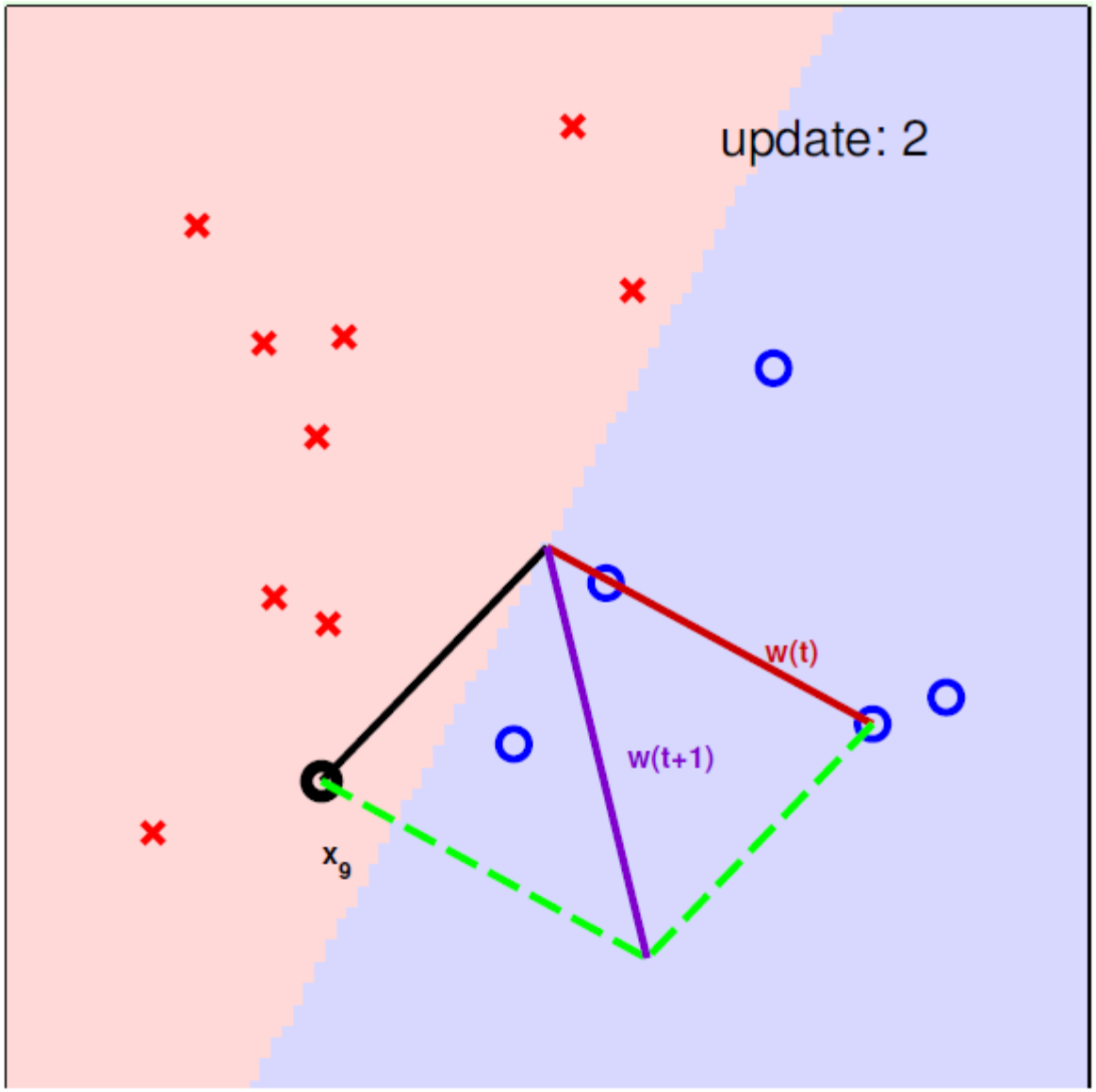
# Perceptron Learning Algorithm



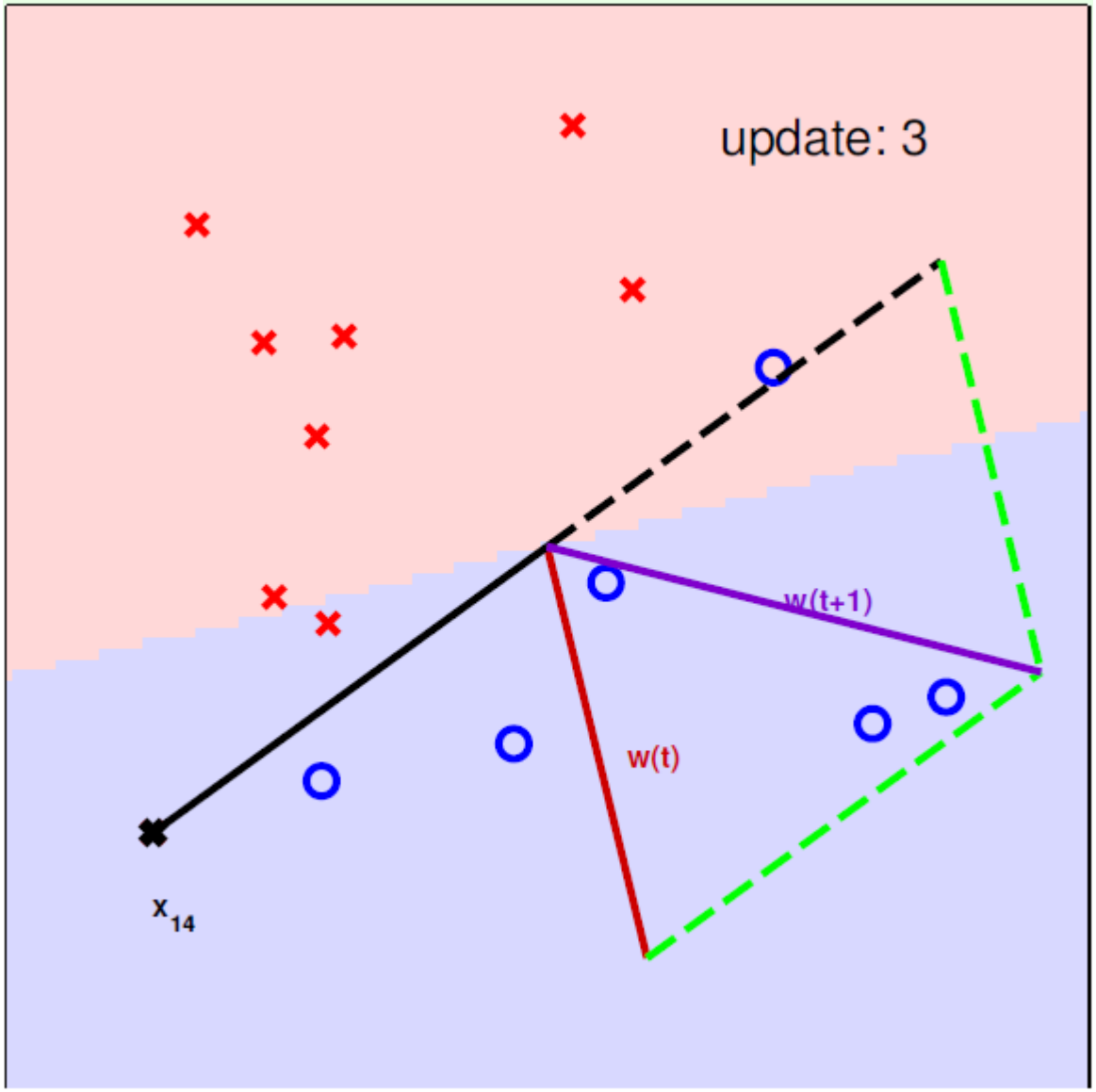
# Perceptron Learning Algorithm



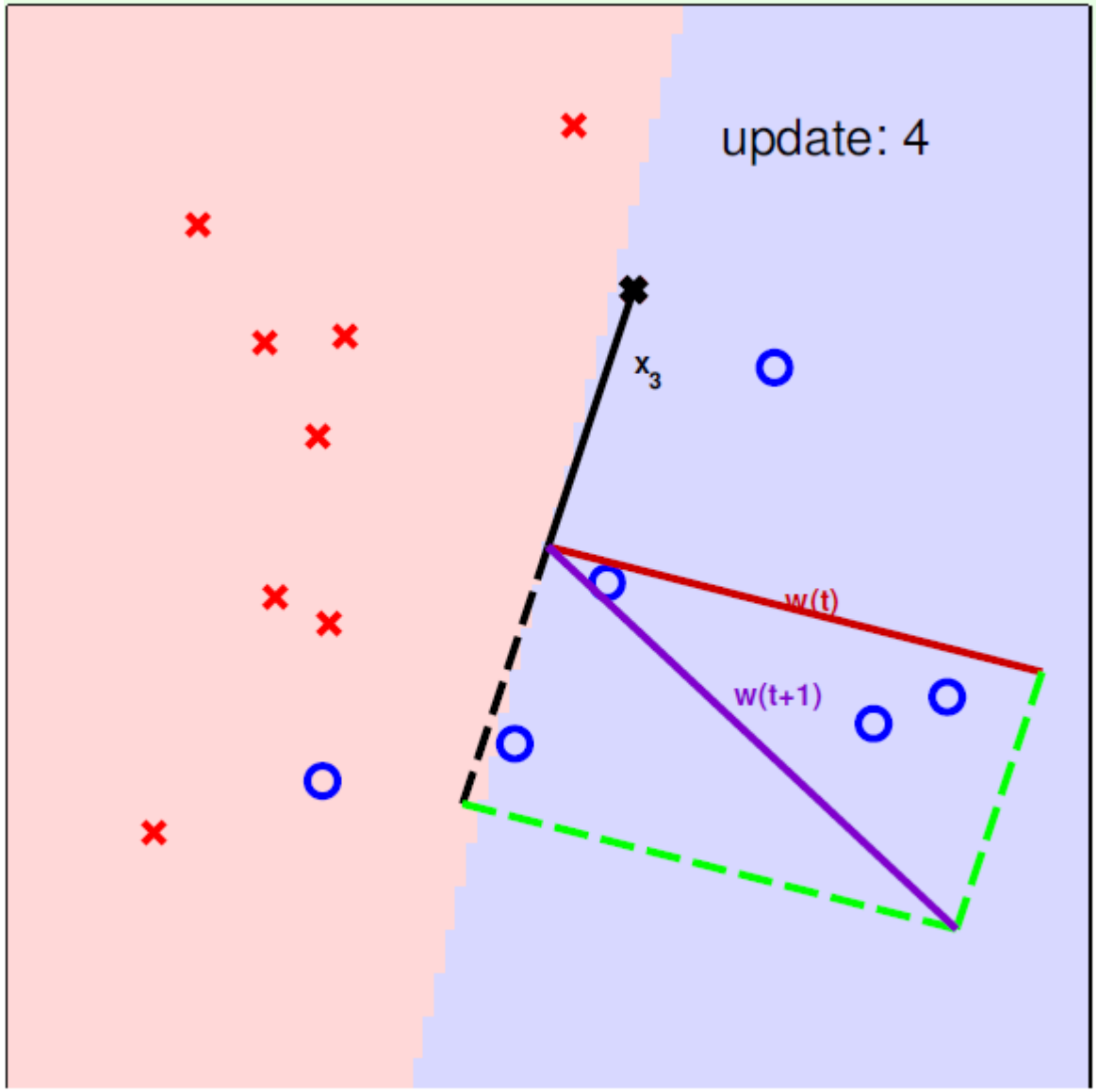
# Perceptron Learning Algorithm



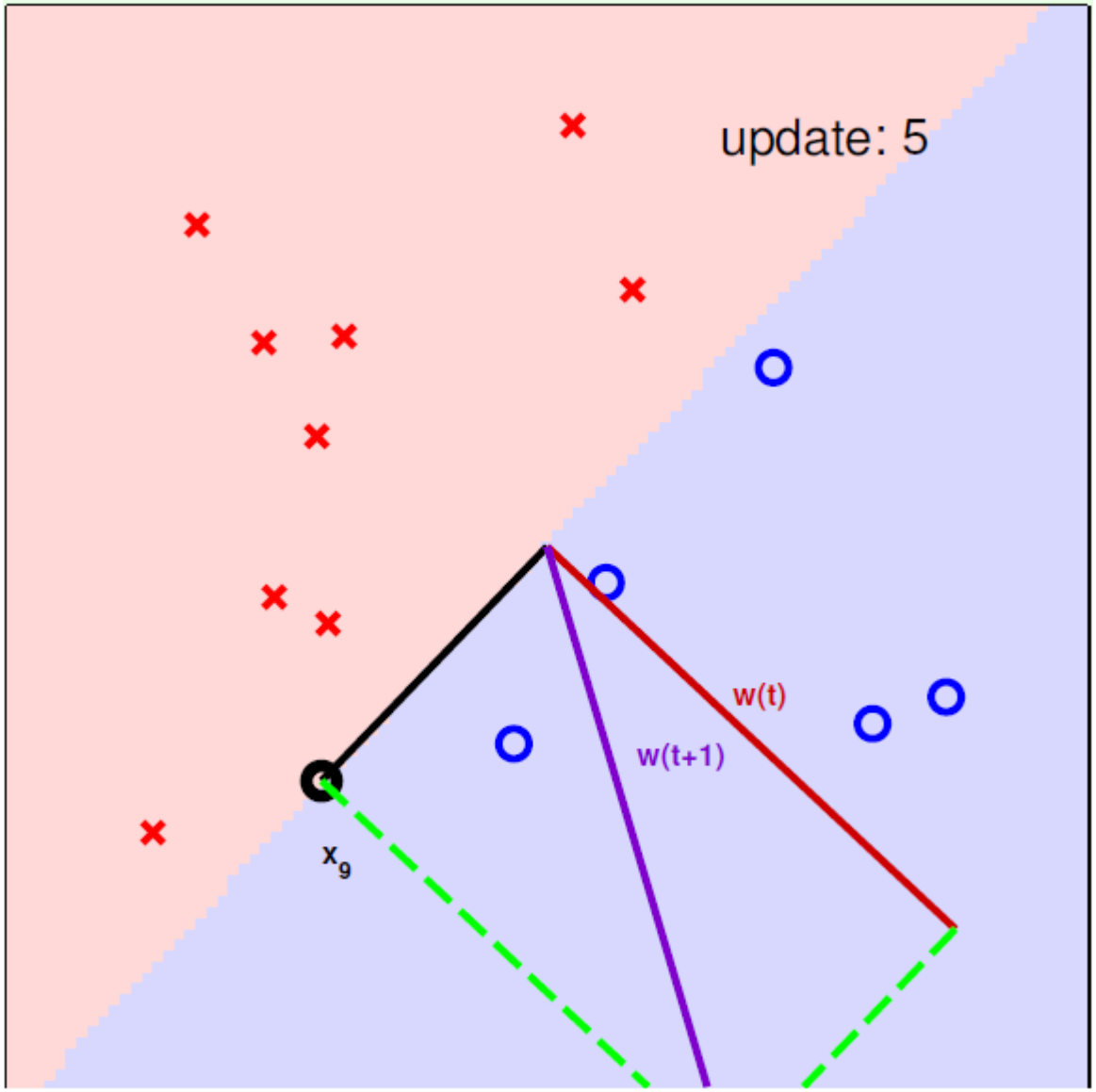
# Perceptron Learning Algorithm



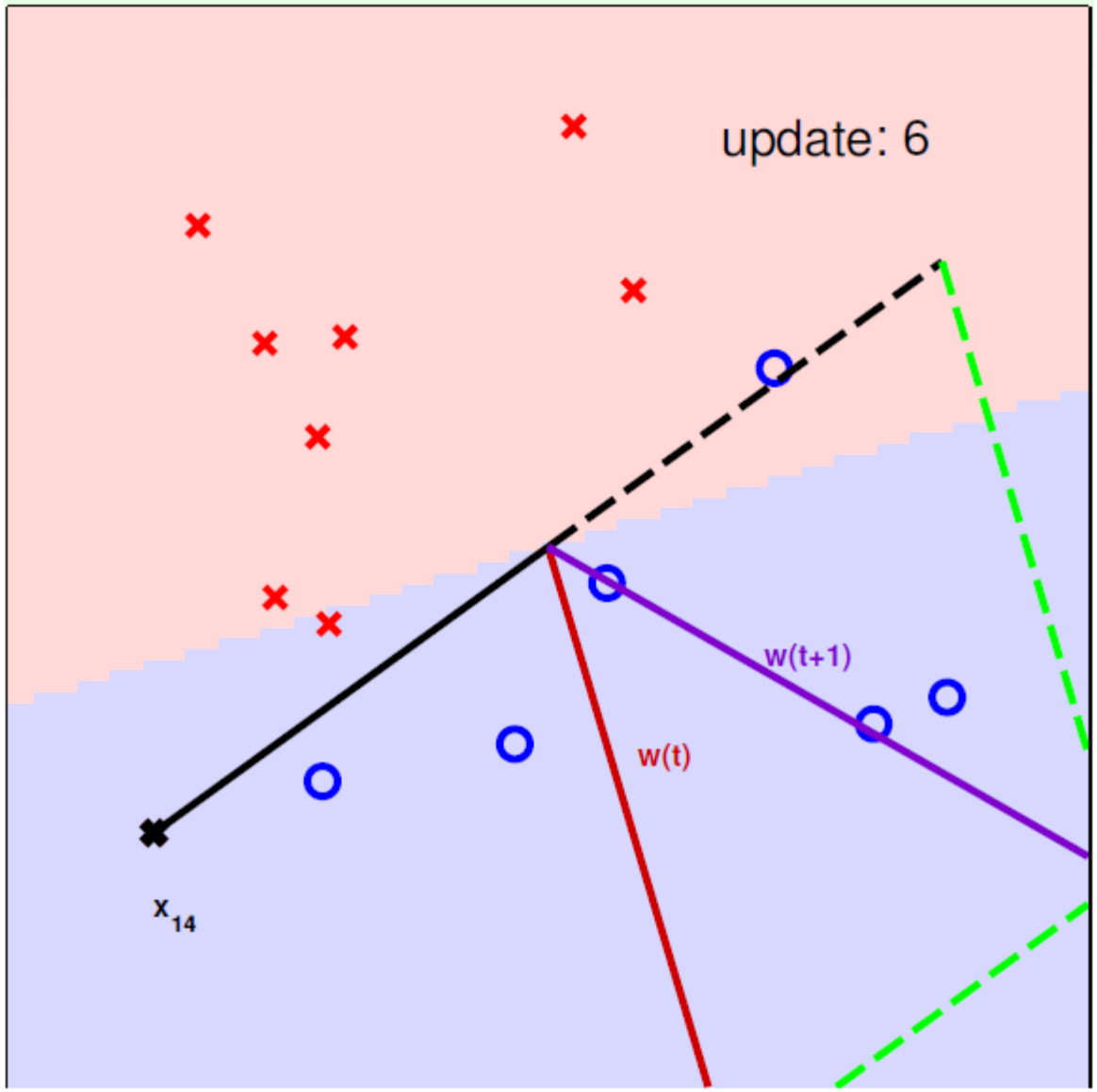
# Perceptron Learning Algorithm



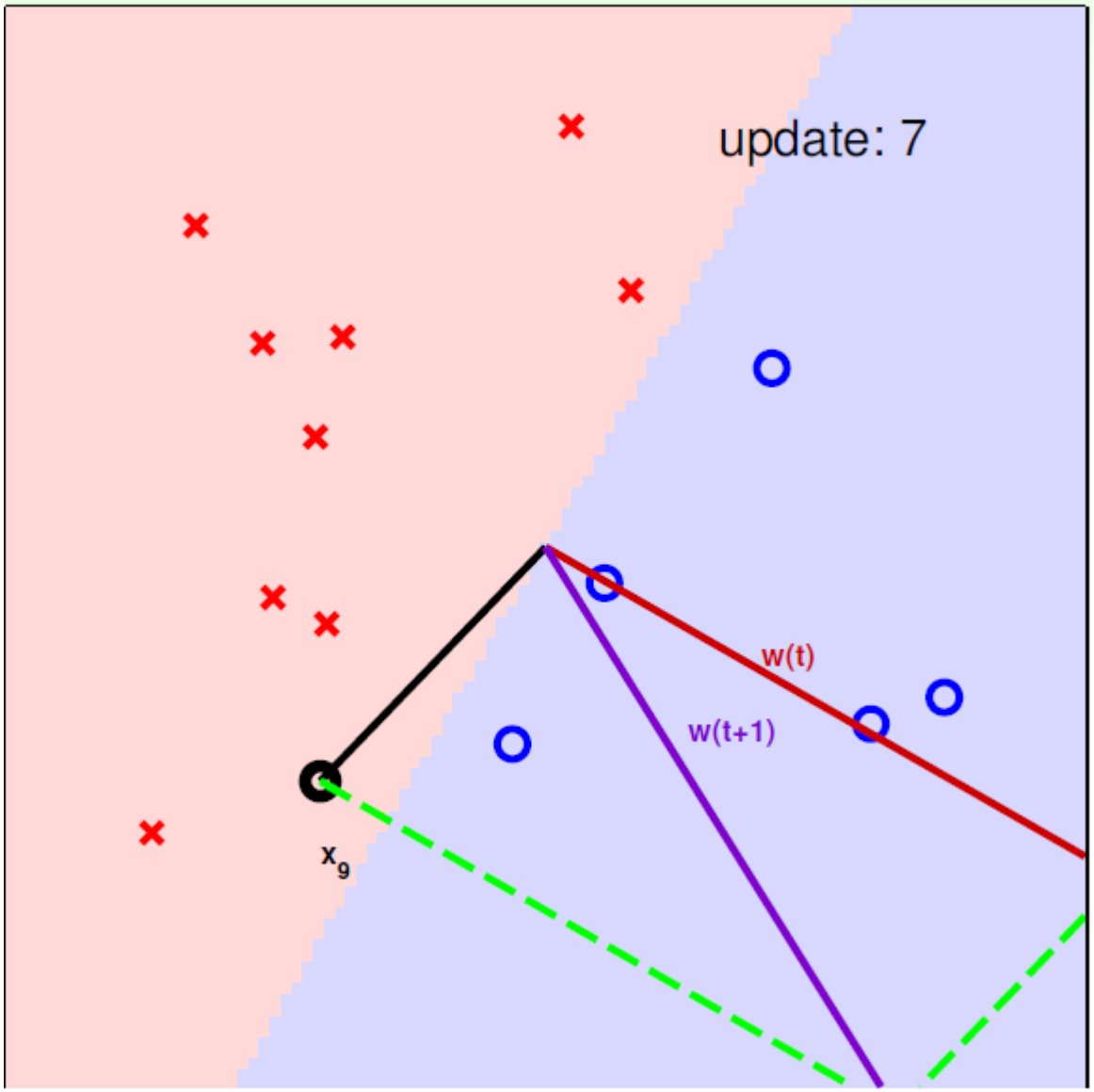
# Perceptron Learning Algorithm



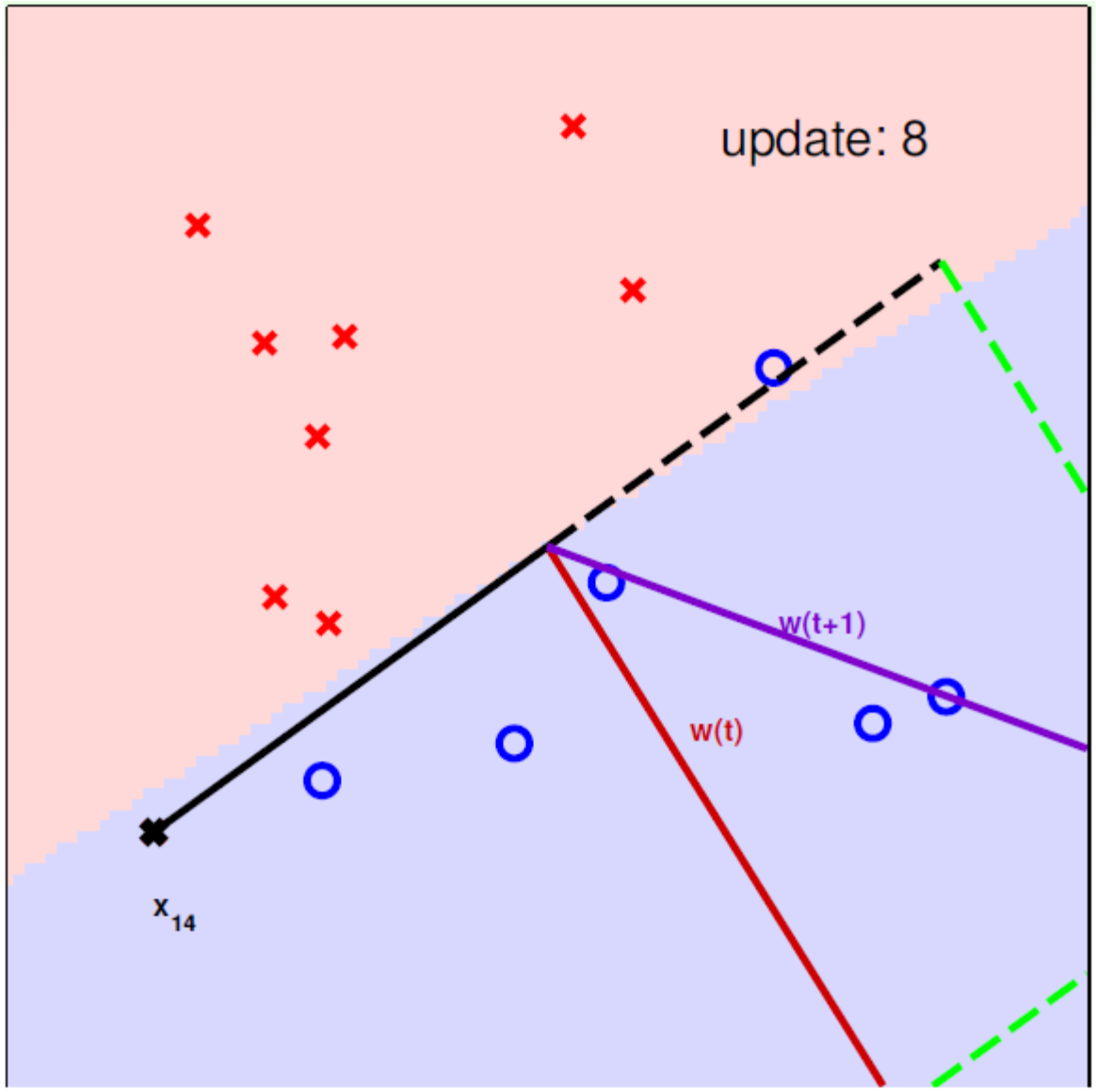
# Perceptron Learning Algorithm



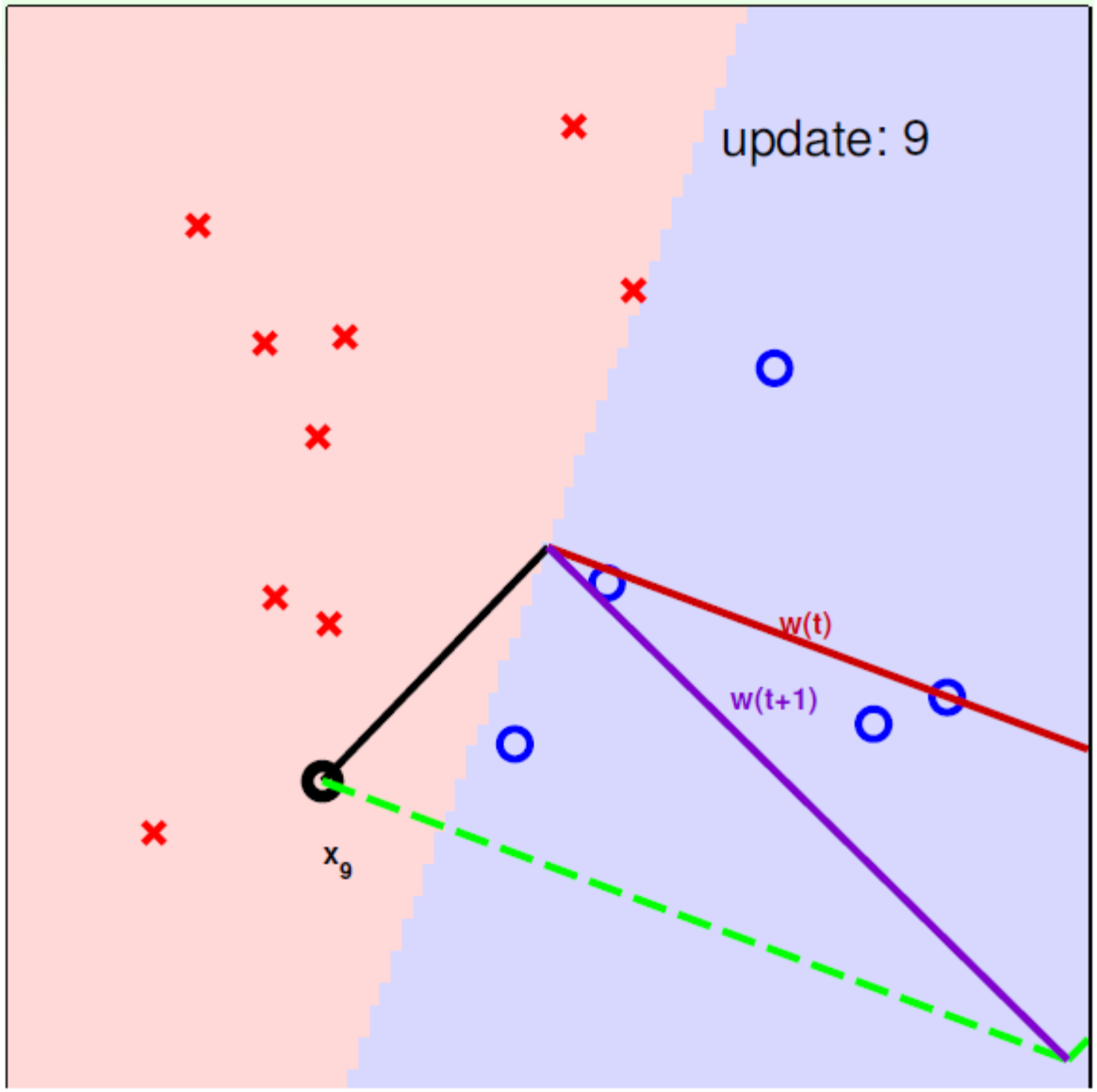
# Perceptron Learning Algorithm



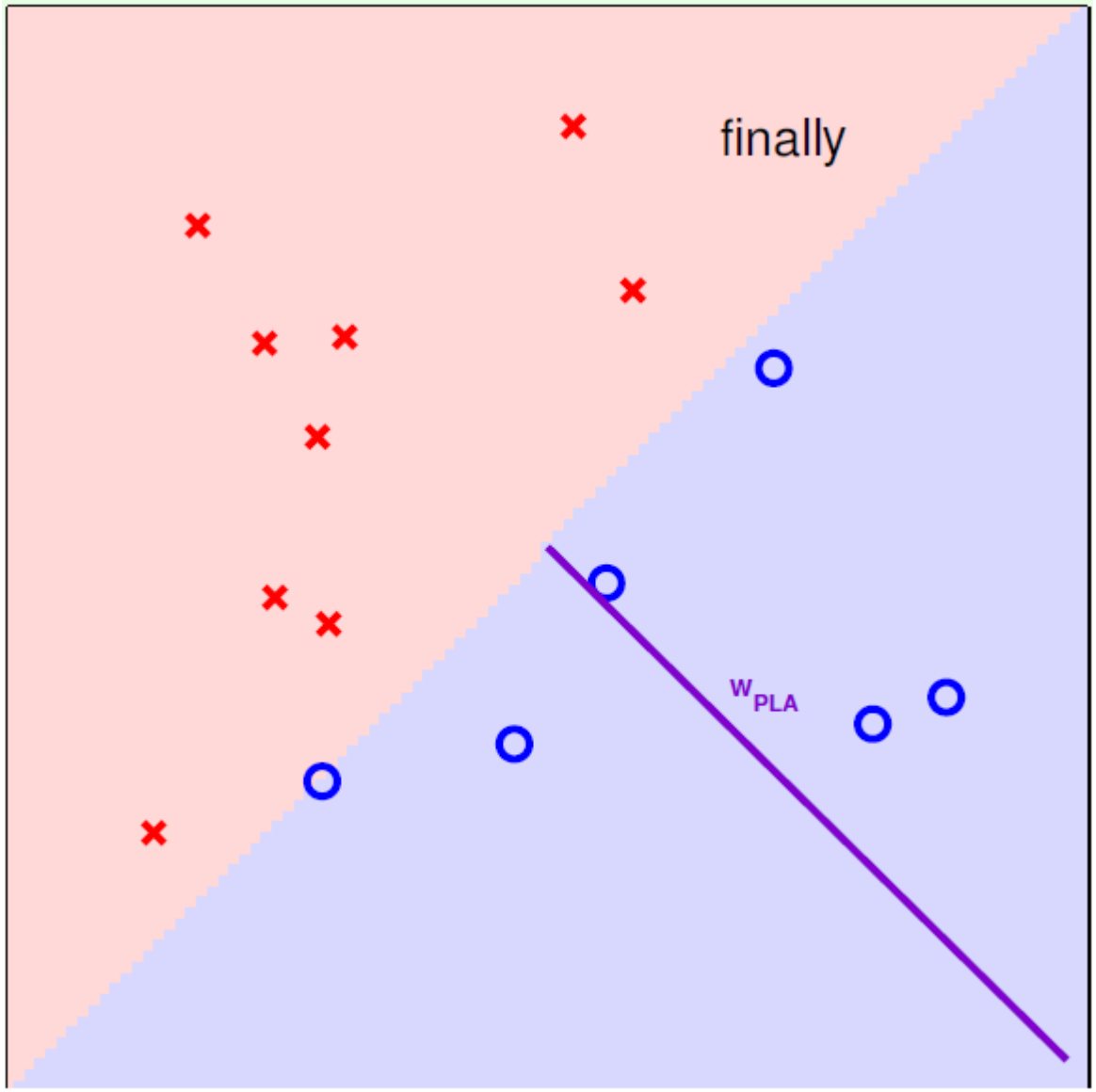
# Perceptron Learning Algorithm



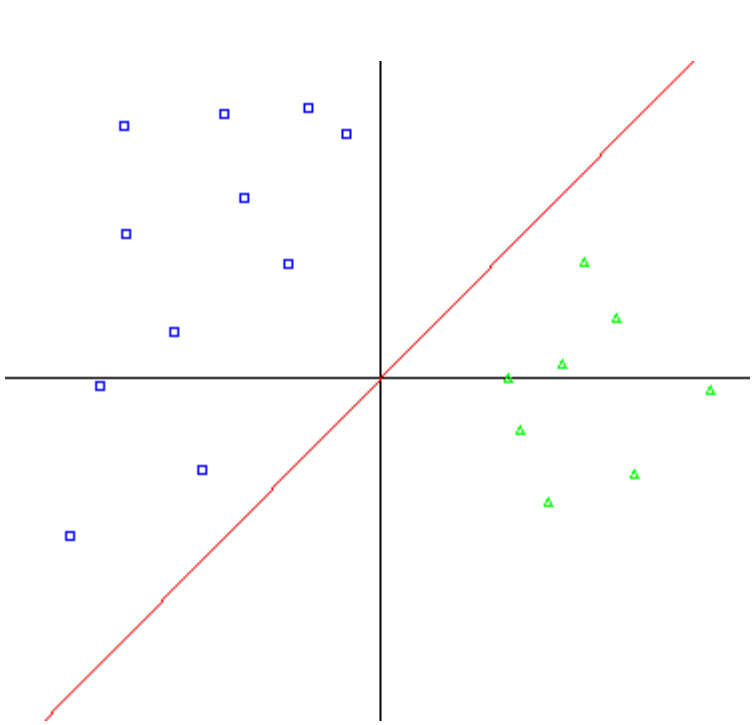
# Perceptron Learning Algorithm

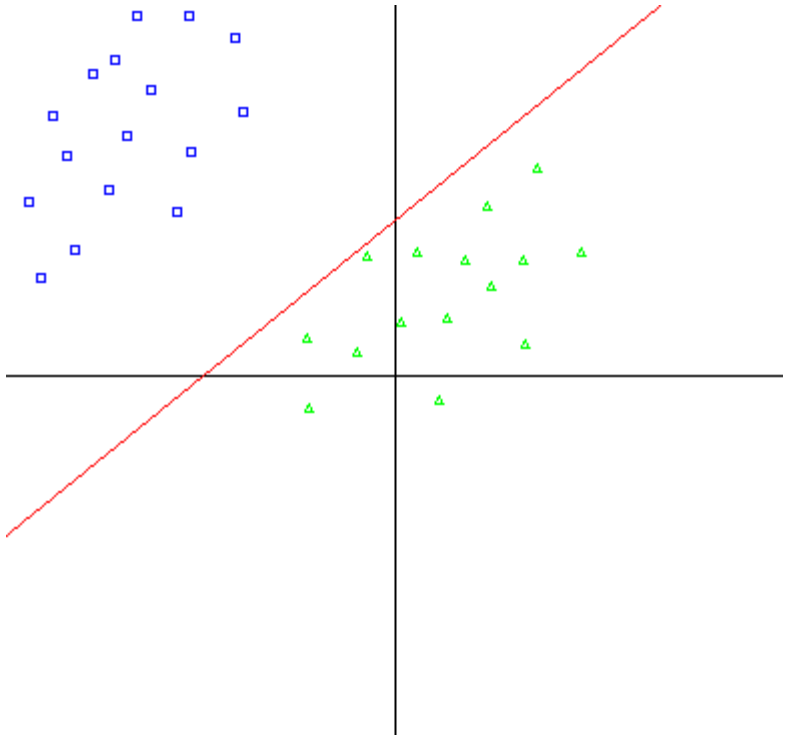


# Perceptron Learning Algorithm



# Perceptron Learning Algorithm

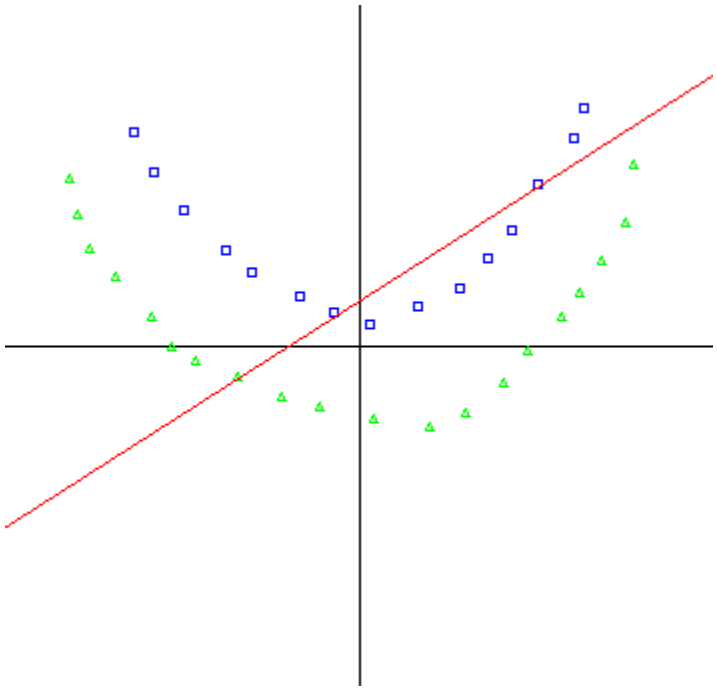


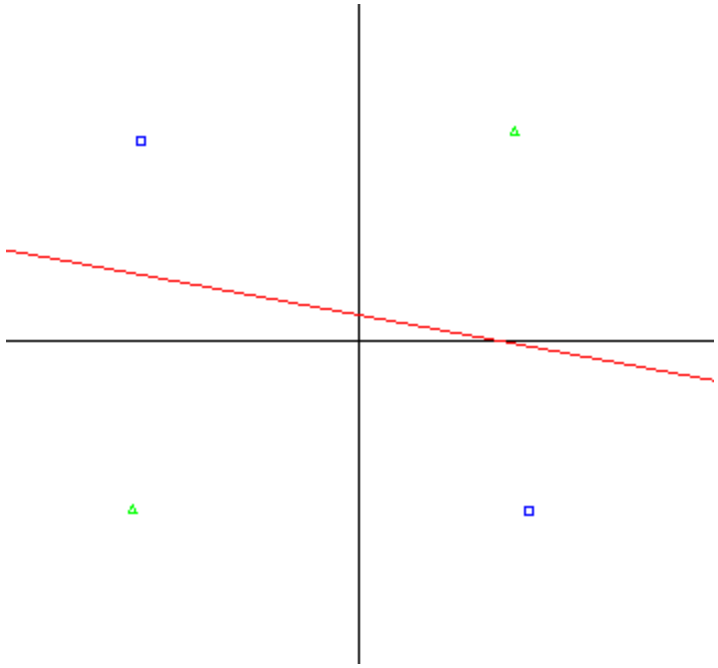


# Perceptron Learning Algorithm

- Only if there exists an hyperplane that correctly classifies the data, the Perceptron procedure is guaranteed to converge; furthermore, the algorithm may give different results depending on the order in which the elements are processed, indeed several different solutions exist.

# Perceptron Learning Algorithm





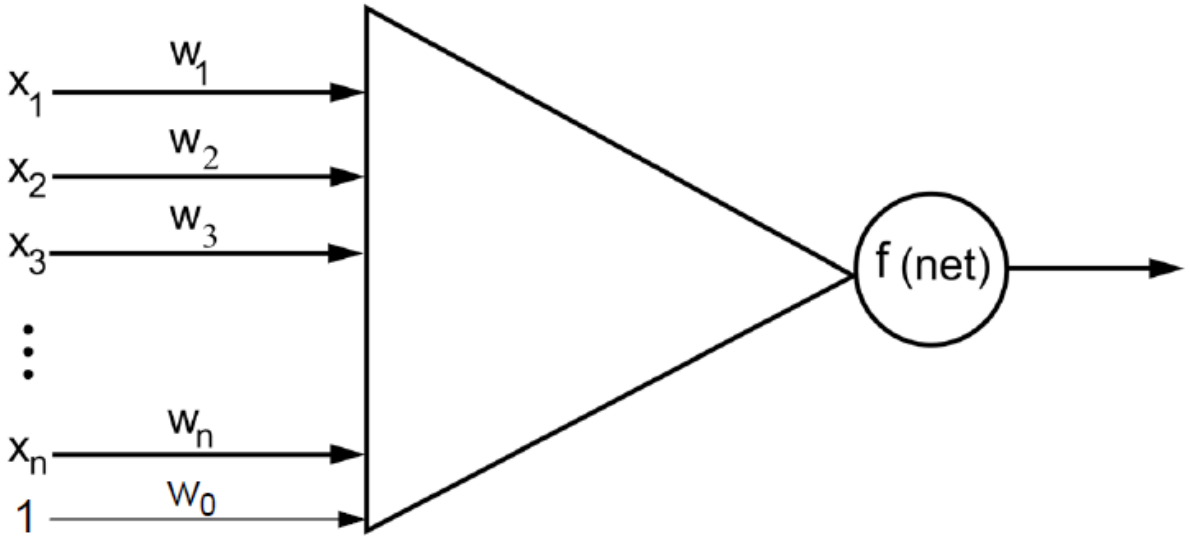
# Artificial Neuron

- The unit of computation in neural networks is the artificial neuron.
- An artificial neuron consists of
  - Input signals $x_i$. These signals represent data from the environment or activation of other neurons.
  - A set of real-valued weights $w_i$. The values of these weights represent connection strengths.
  - An activation level $\sum_i w_i x_i$ . The neuron's activation level is determined by the sum of the weighted inputs.
  - A threshold function $f$. This function computes the final output by determining if the activation is below or above a threshold.

# Artificial Neuron

- Given the activation value $net = \sum_i w_i x_i$ , the output of the neuron is given by

$$f(net) = \begin{cases} +1 & if \sum_i w_i x_i \geq 0 \\ -1 & if \sum_i w_i x_i < 0 \end{cases}$$



# Example

- An artificial neuron can be used to compute the logic AND function.
  - The neuron has three inputs
  - $x_1$ and $x_2$ are the original inputs.
  - The third is the bias input which has a constant value of +1.
  - The input data and bias have weights of +1, +1, and –2 respectively.
- What about the logic OR function?

# Artificial Neuron

- The perceptron learning algorithm (PLA) can be used to adjust the weights of an artificial neuron.
- The weights are adjusted until the outputs of the neuron become consistent with the true outputs of training examples.
- The following rule is used

$$\mathbf{w}_{(t+1)} \leftarrow \mathbf{w}_{(t)} + y_{n(t)}\mathbf{x}_{n(t)}$$

# Artificial Neuron

- Perceptron learning algorithm can not solve those problems where the patterns are not linearly separable.
- An example of this is the exclusive-OR problem.
- Multilayer networks are required for solving such kinds of problems.

# Example

| $x_1$ | $x_2$ | Output |
|---|---|---|
| 1 | 1 | -1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | -1 |