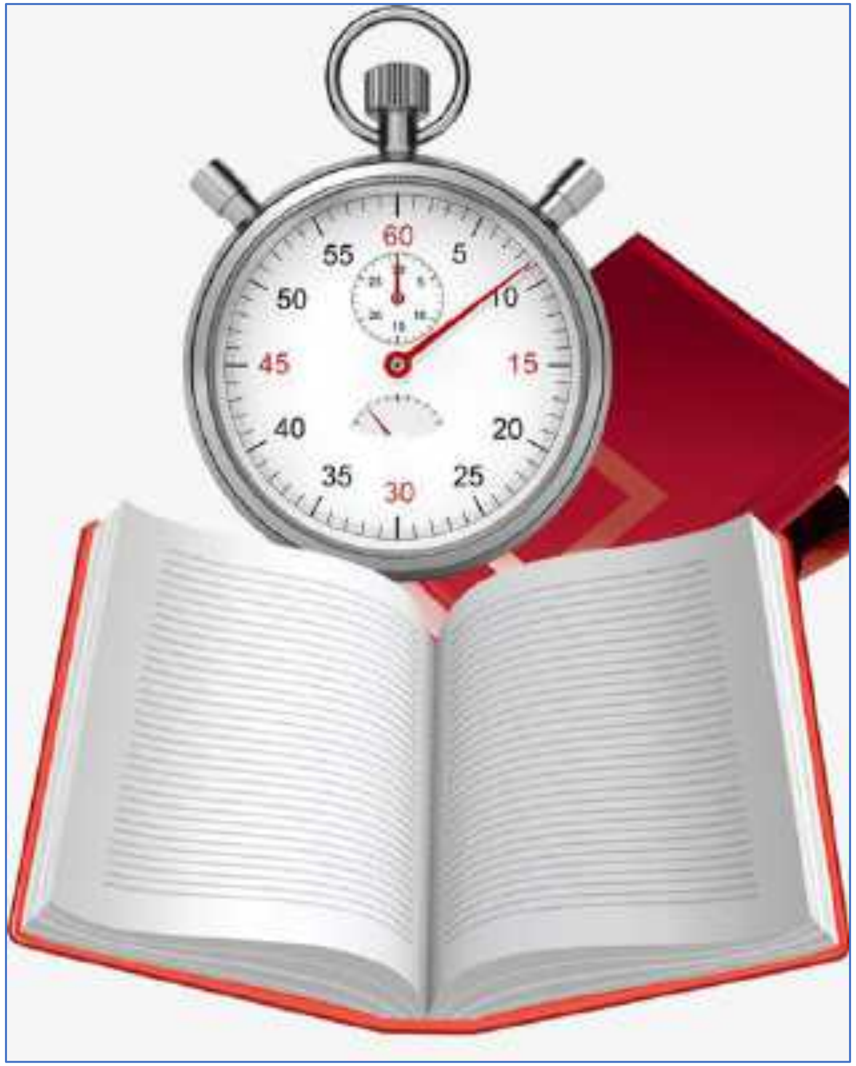
CS 103 -05

Perceptron and AI Early Day Algorithms

Jimmy Liu 刘江

2020-10-16

Lecture 04 Review



Early AI Algorithms

3 3

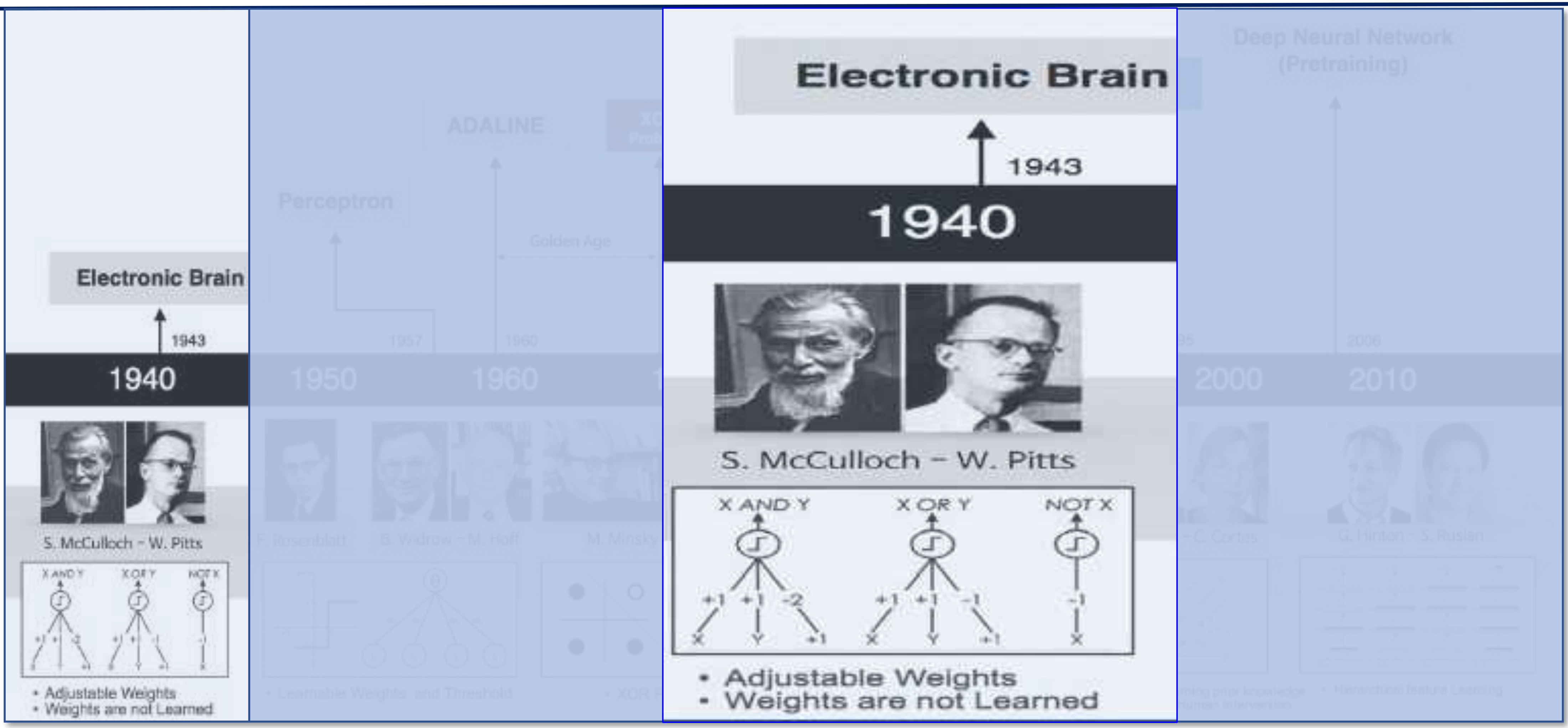
1 Early-AI Algorithms

2 Electronic Brain

3 Turing Test

4 Chinese Card Room

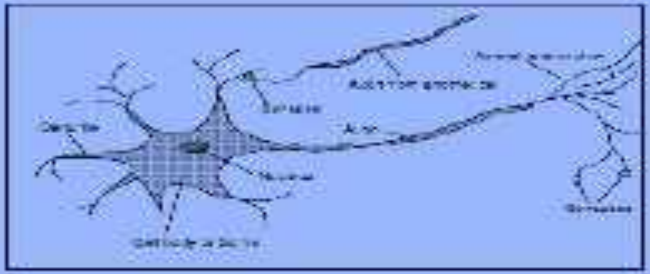
AI algorithm Developments - A Closer Look

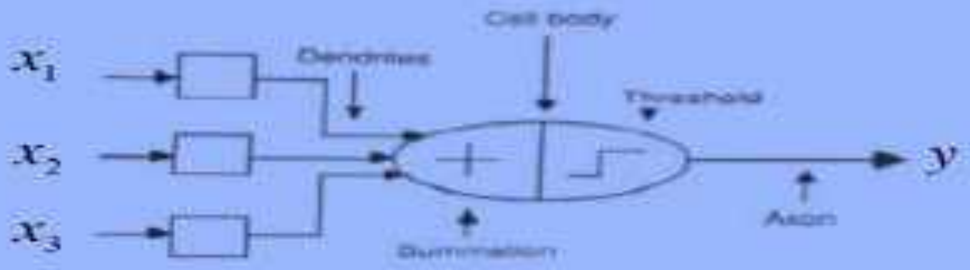


Electronic Brain from McCulloch and Pitts

Early Artificial Neurons (1943-1969)

- McCulloch- Pitts neuron

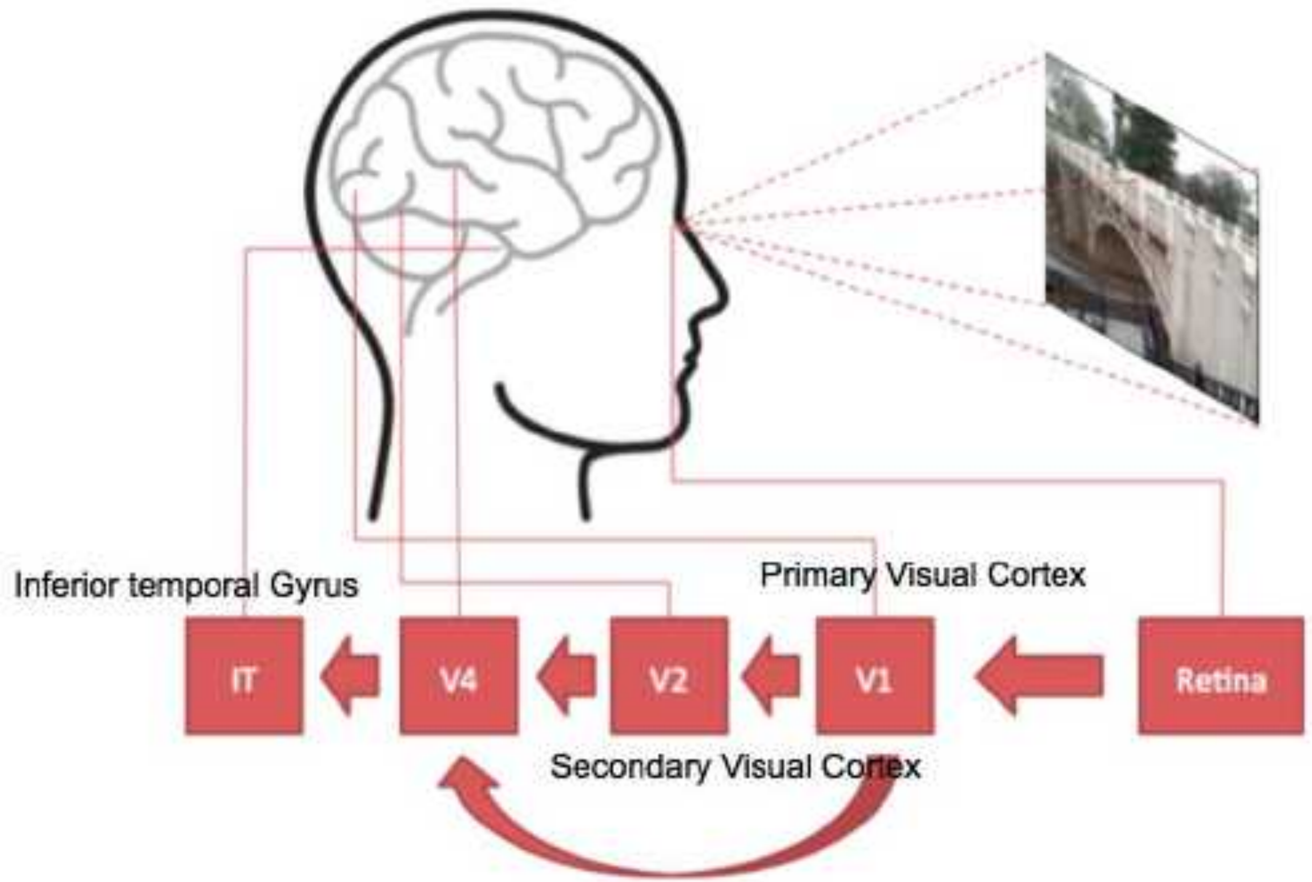




$$y = f\left(\sum_m w_m x_m - b\right) \quad f(x) = \begin{cases} 1 & \text{if } x \geq 0; \\ 0 & \textit{otherwise} \end{cases}$$

- McCulloch and Pitts [1943] proposed a simple model of a neuron as computing machine
- The artificial neuron computes a weighted sum of its inputs from other neurons, and outputs a one or a zero according to whether the sum is above or below a certain threshold

Visual Input to Brain: Brain Computing



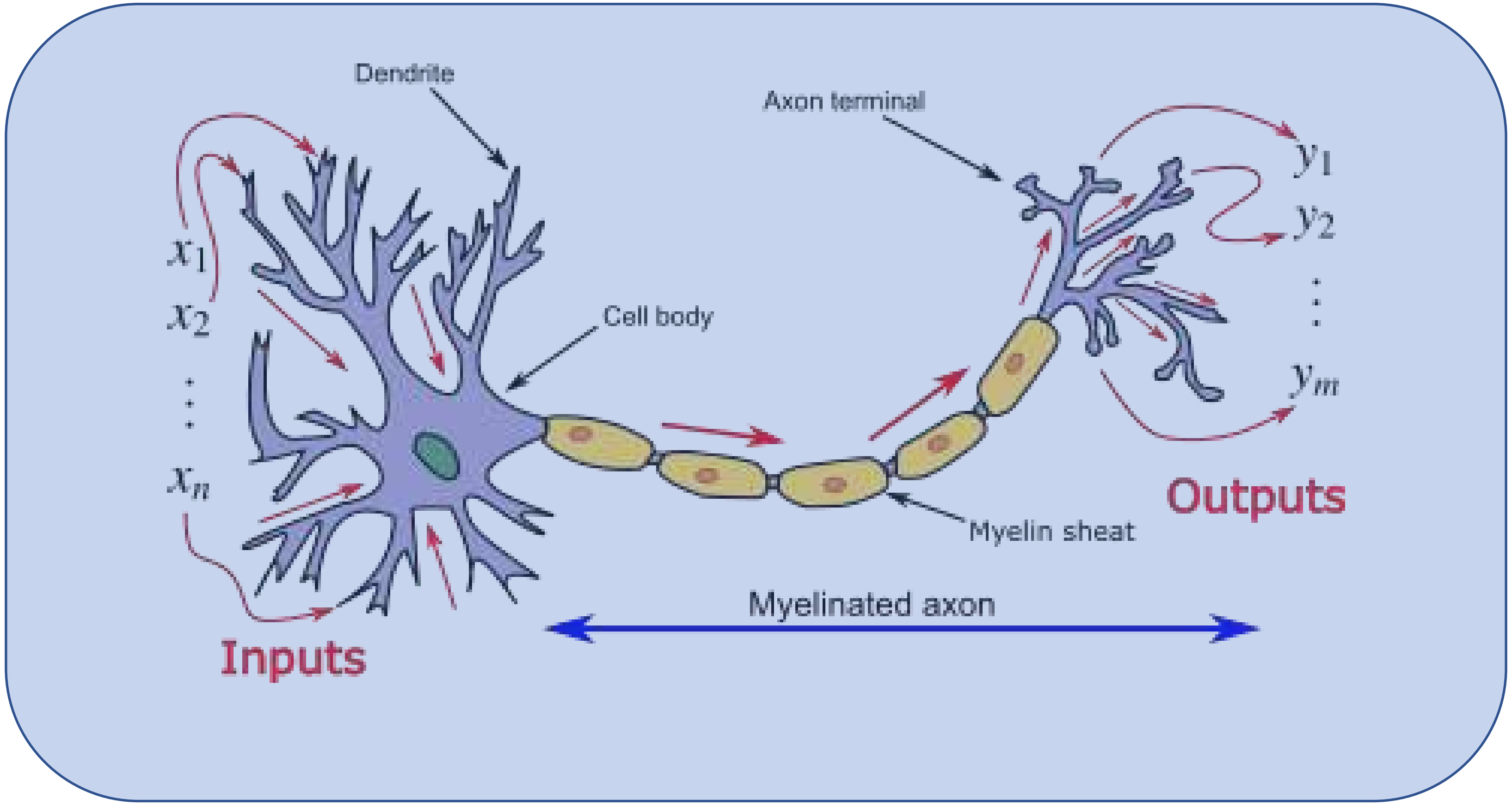
V1: Edge detection, etc.

V2: Extract simple visual properties (orientation, spatial frequency, color, etc)

V4: Detect object features of intermediate complexity

TI: Object recognition.

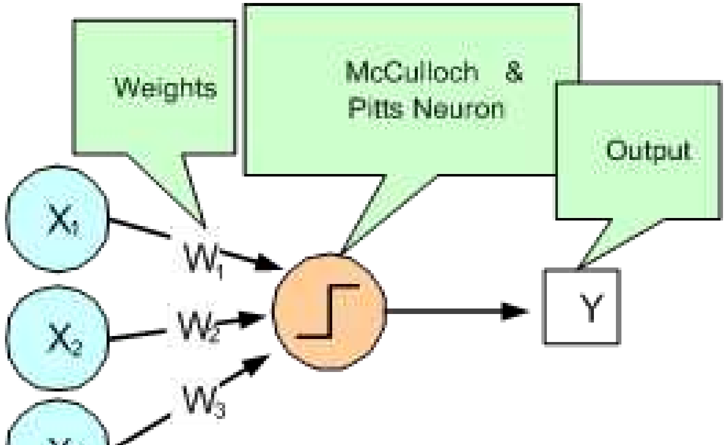
Schematic of a Biological Neuron



MCP (McCulloch and Pitts) Neuron – Weights Are Adjusted But Not Learnt

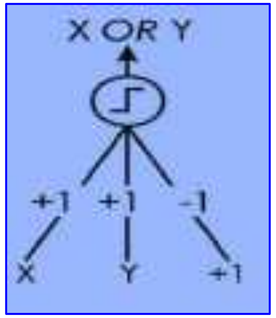
- Tried to understand how the brain could produce highly complex patterns by using many basic cells that are connected together.

- These basic brain cells are called neurons, and McCulloch and Pitts gave a highly simplified model of a MCP neuron in their paper.



Warren S. McCulloch, a neuroscientist, and Walter Pitts, a logician, "A logical calculus of the ideas immanent in nervous activity", the Bulletin of Mathematical Biophysics 5:115-133. 1943

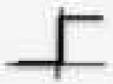
Prove “OR” MCP (McCulloch and Pitts) Neuron



A	B	Bias	W1	W2	W3	Transfer
0	0	1	1	1	-1	G(z)
0	1	1	1	1	-1	G(z)
1	0	1	1	1	-1	G(z)
1	1	1	1	1	-1	G(z)

$Z = \sum$ = A*W1+B*W2+ Bias*W3	Output F=G(Z) (Z>=0)	A OR B
0*1 + 0*1 + 1*(-1) = (-1)	0	0
0*1 + 1*1 + 1*(-1) = 0	1	1
1*1 + 0*1 + 1*(-1) = 0	1	1
1*1 + 1*1 + 1*(-1) = 1	1	1

Transfer Function is G(z)



$$g(z) = \begin{cases} 1 \text{ if } z \geq 0 \\ 0 \text{ otherwise.} \end{cases}$$

Turing Test – Operational Definition

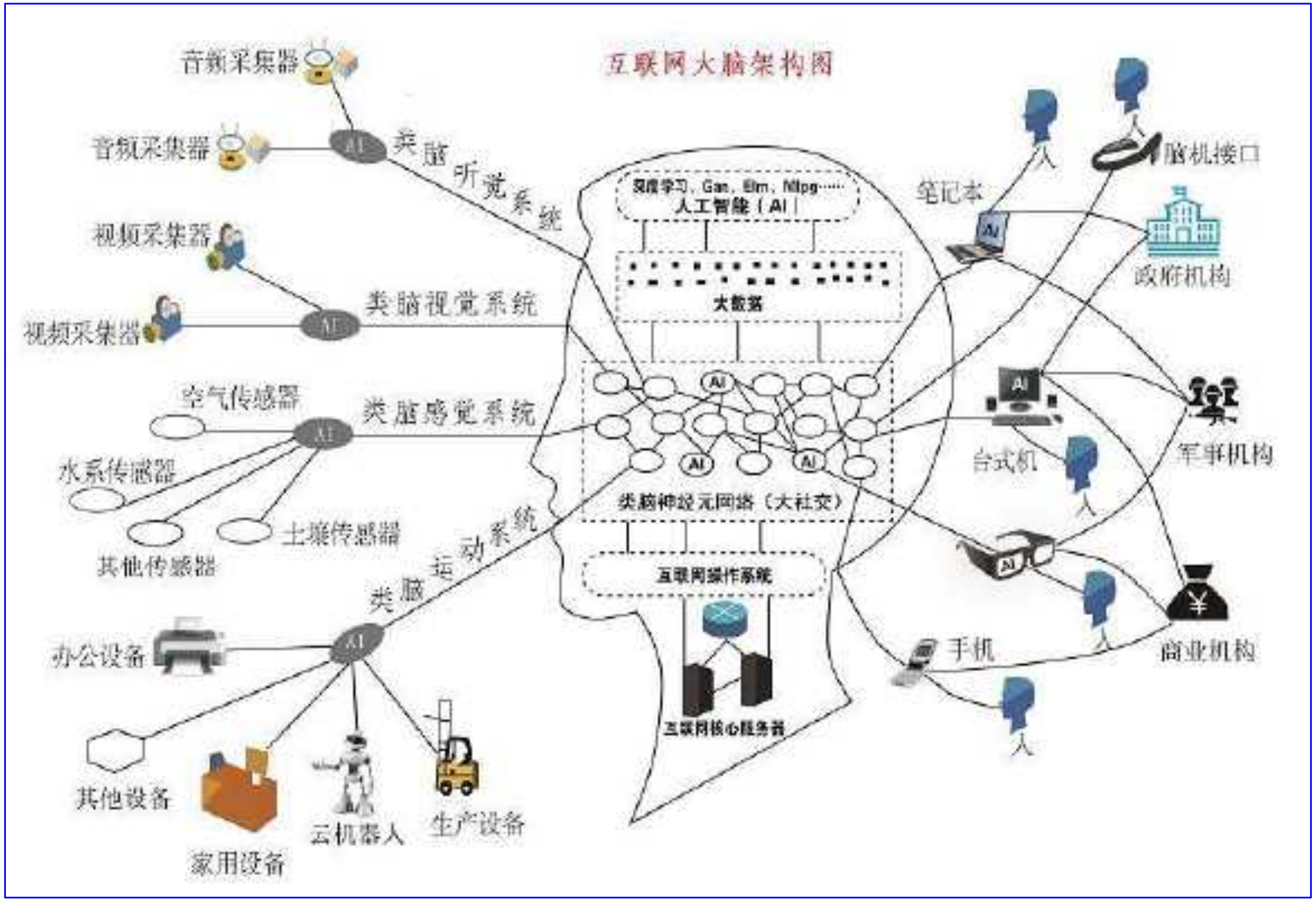
1. If the Turing Test was passed, Turing would conclude that the machine was intelligent.
2. Suggested as a way of saying when we could consider machines to be intelligent, or at least act intelligently
3. A satisfactory operational definition of intelligence

Chinese Room and Turing Test

The Chinese Room thought experiment is an analogy to artificial intelligence. A person who can't speak Chinese is sitting in a room text chatting in Chinese. They have a book that gives them an appropriate response to each series of symbols that appear in the chat. The person on the other side of the chat can't tell that they are speaking to someone who can't speak Chinese. The person in the room doesn't understand anything about the conversation and is simply looking up symbols in a book.

The Turing test, a common way to define and test artificial intelligence, involves a computer imitating a human on a chat. According to the test, if a machine can convince people that it's human, it's intelligent. The Chinese Room analogy shows that by this definition of machine intelligence that computers need not understand the conversation to pass.

类脑（Brain-Inspired Intelligence）



人工智能导论项目分组汇总

1. **AI+斗地主**：孙永康、李怀武、胡鸿飞、吴一凡、杨光、张习之、**金肇轩（组长）**、于佳宁
2. **AI+五子棋**：周贤玮、**韩梓辰（组长）**、金冬阳、赵云龙、张坤龙、陈茜、夏星晨
3. **High Score Gamer**：易辰朗、许天淇、**黄北辰（组长）**、赵思源、朱佳伟、宛清源（未选择小组）
4. **AI application on diabetes**：周钰奇、**李仪轩（组长）**、董叔文、湛掌、胡钧淇、裴鸿婧
5. **AI in lung cancer**：夏瑞浩、李悦明、龚颖璇、**吴云潇潇（组长）**、姜欣瑜、王英豪
6. **基于MRI图像的阿尔茨海默症分类**：董廷臻、**郑英炜（组长）**、李博翱、朱嘉楠、李杨燊
7. **AI Applications in Breast Cancer Imaging**：林文心、**翟靖蕾（组长）**、孙瀛、林宝月、陈帅名、冀鹏宇
8. **Applications of artificial intelligence in covid-19 patients**：**罗岁岁（组长）**、周雅雯、肖雨馨、程旸、尹子宜
9. **基于OCT图像的眼部多种疾病诊断和分析的调研**：何忱、郭煜煊、朱寒旭、**赵子璇（组长）**、王子杰、张晓新
10. **人工智能对白内障分级的算法综述**：赵宇航、徐格蕾、陈星宇、祖博瀛、**黄弋骞（组长）**
11. **句子图片的文本情感分析**：唐云龙、刘叶充、刘旭坤、马卓远、**陈子蔚（组长）**、江欣乐、陈浩然（未选择小组）
12. **gesture recognition**：车文心、张静远、**张骥霄（组长）**、杜鹏辉
13. **AI in Lab**：孙含曦、于松琦、**罗西（组长）**、唐家豪、陈松斌、孙杰欣（未选择小组）
14. **人脸识别算法的发展与应用**：**易翔（组长）**、陈俊滔、罗景南、胡泰玮、文颖潼、吴杰翰
15. **人工智能在无障碍设施领域中的使用调查**：**马子晗（组长）**、陈沐尧、林小璐、任艺伟、王增义
16. **identification of handwriting elements**：刘通、谈思序、赵伯航、张皓淇
17. **AI虚拟主播制作计划**：王标、**张倚凡（组长）**、李康欣、何泽安、曾宇祺、**Zhang Kenneth（未选择小组）**
18. **人工智能技术在个性化推荐系统上的应用与研究**：谭雅静、刘思岑、Ooi Yee Jing、孟宇阳、**杨锦涛（组长）**
19. **校园巴士路线优化**：王祥辰、何鸿杰、吴子彧、**樊青远（组长）**、方琪涵、袁通
20. **给线稿上色的强大AI的算法研究（人工智能应用于病理分析的前景与挑战）**：**韩晗（组长）**、刘思语、赵晓蕾、刘宇欣、**李修治（组长）**、沈睿琦
21. **深度学习在自动驾驶中的应用**：**王晓轩**

人工智能导论项目小组组长

1. AI+斗地主：**金肇轩**
2. AI+五子棋：**韩梓辰**
3. High Score Gamer：**黄北辰**
4. AI application on diabetes：**李仪轩**
5. AI in lung cancer：**吴云潇潇**
6. 基于MRI图像的阿尔茨海默症分类：**郑英炜**
7. AI Applications in Breast Cancer Imaging：**翟靖蕾**
8. Applications of artificial intelligence in covid-19 patients：**罗岁岁**
9. 基于OCT图像的眼部多种疾病诊断和分析的调研：**赵子璇**
10. 人工智能对白内障分级的算法综述：**黄弋骞**
11. 句子图片的文本情感分析：**陈子蔚**
12. gesture recognition：**张骥霄**
13. AI in Lab：**罗西**
14. 人脸识别算法的发展与应用：**易翔**
15. 人工智能在无障碍设施领域中的使用调查：**马子晗**
16. identification of handwriting elements：没有确定
17. AI虚拟主播制作计划：**张倚凡**
18. 人工智能技术在个性化推荐系统上的应用与研究：**杨锦涛**
19. 校园巴士路线优化：**樊青远**
20. 给线稿上色的强大AI的算法研究（人工智能应用于病理分析的前景与挑战）：**韩晗、李修治**
21. 深度学习在自动驾驶中的应用：**王晓轩**

Wrong Answer 1

Prove “NOT X” Artificial Neuron

X	W1	Transfer	Z=-w1-x	G(z)	Not X
0	-1	G(z)	1	1	1
1	0	G(z)	-1	0	0

Wrong Answer 2

A	W^1	Transfer	$z = A * W^1$	Output = G(z)
0	-1	G(z)	0 * 1 = 0	1
1	-1	G(z)	1 * (-1) = 1	0

Wrong Answer 3

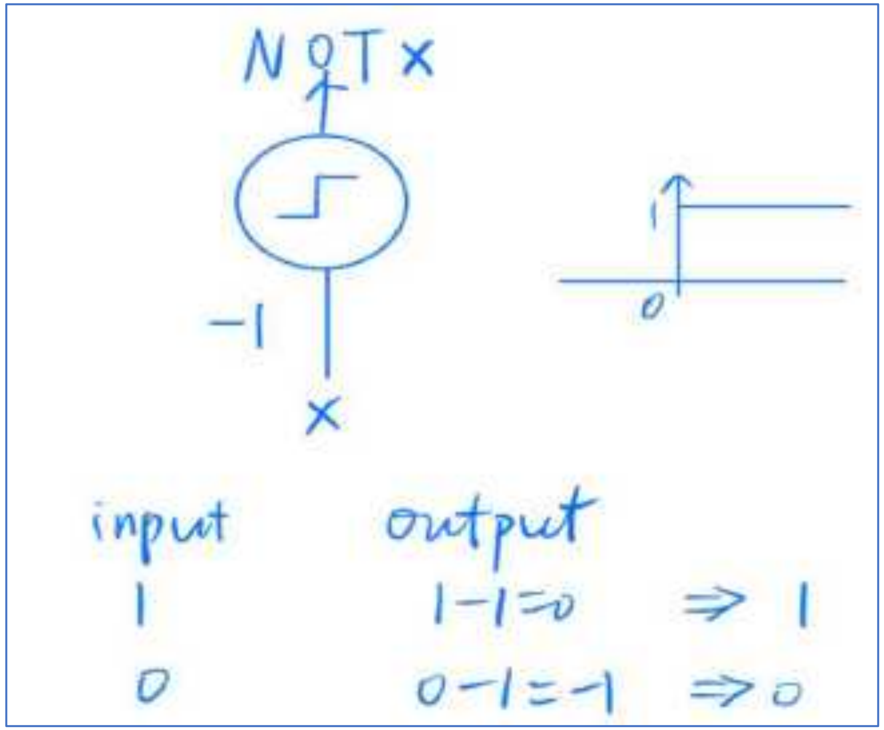
Prove not X Artifical Neuron.

A	Bias	W_1	W_2	Transfer
1	1	-1	1	$G(z)$
0	1	-1	1	$G(z)$

$Z = \Sigma = A^* W_1 + Bias^* W_2$	Output $f = G(z)$	A NOT B
$1 \times (-1) + 1 \times 1 = 0$	0	0
$0 \times (-1) + 1 \times 1 = 1$	1	1

Transfer function $G(z)$ is $g(z) = \begin{cases} 0, & \text{if } z \le 0 \\ 1, & z \text{ else} \end{cases}$

Wrong Answer 4



Wrong Answer 5

A	W_1	Transfer	Output(=g(z))	NOTX
0	0	$g(z)$	1	1
1	-1	$g(z)$	0	0

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Correct Answers

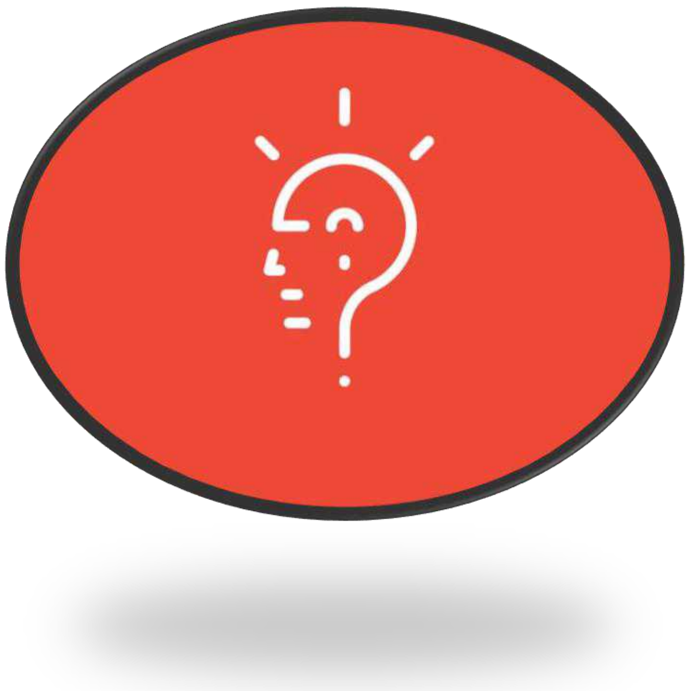
x	w	z = x*w	g(z)	NOT X
0	-1	0	1	1
1	-1	-1	0	0

A	Bias	W1	transger	Z = W1*A	Output F =G(z)	Not A
0	0	-1	G(z)	0	1	1
1	0	-1	G(z)	-1	0	0

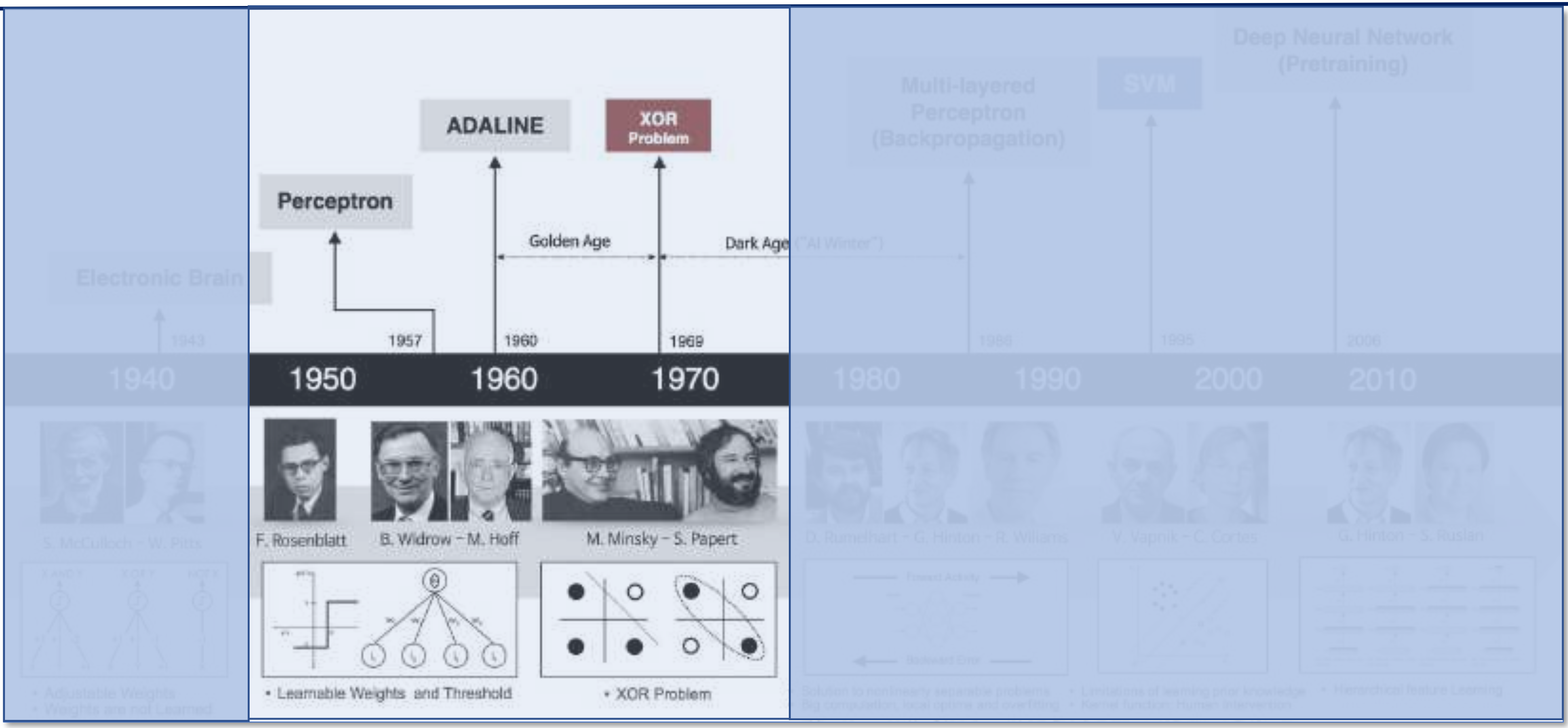
x	w	Transfer	Z = x*w	Output y=g（z）	Not x
1	-1	g（z）	-1	0	0
0	-1	g（z）	0	1	1

X	W	Transfer	Z=Σ=X*W	Output F=G(Z)	NOT X
0	-1	G(Z)	0*(-1)=0	1	1
1	-1	G(Z)	1*(-1)=-1	0	0

Any Question?



AI algorithm Developments - A Closer Look



Perceptron

3 4

1 Perceptron

2 Perceptron Learning

3 ADALINE

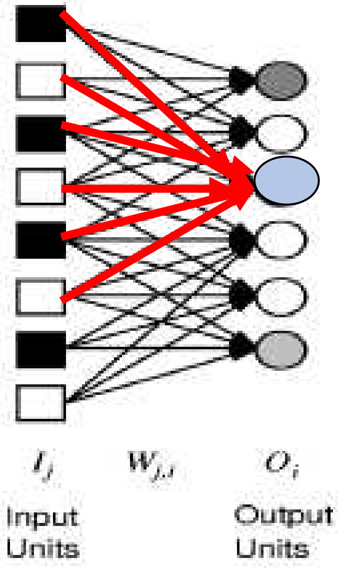
4 Limitation of Perceptron

Perceptrons

•Single-layer feed forward neural network (perceptron network)

A network with all the inputs connected directly to the outputs

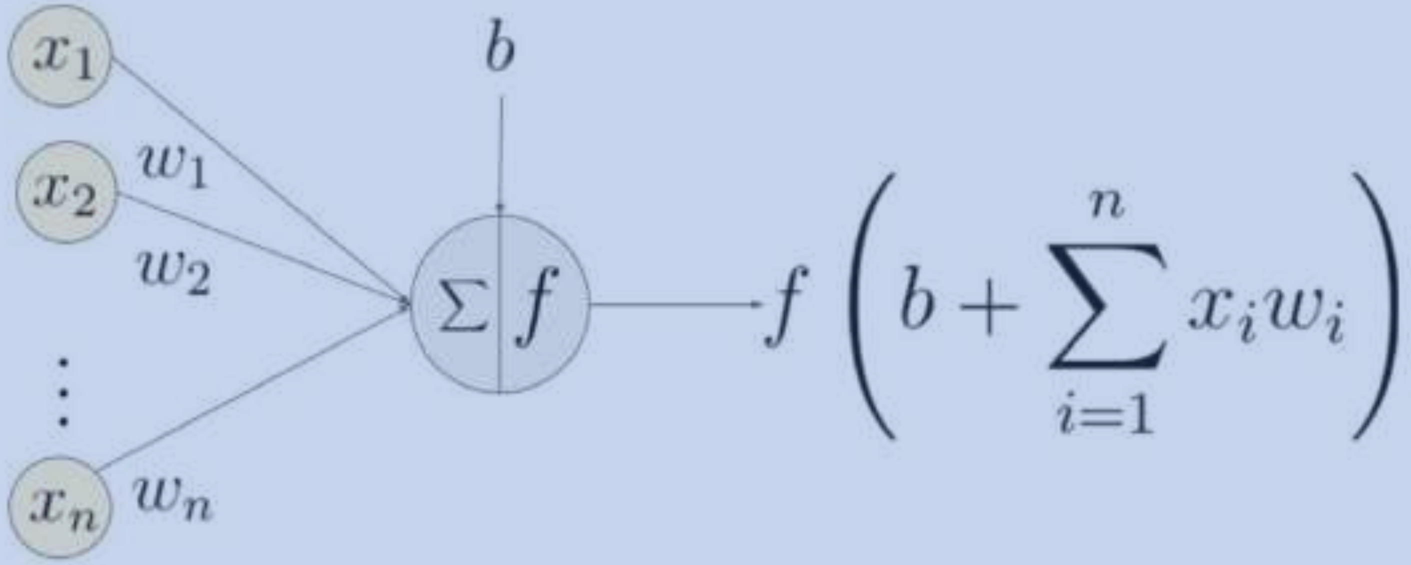
–Output units all operate separately: no shared weights



Perceptron Network

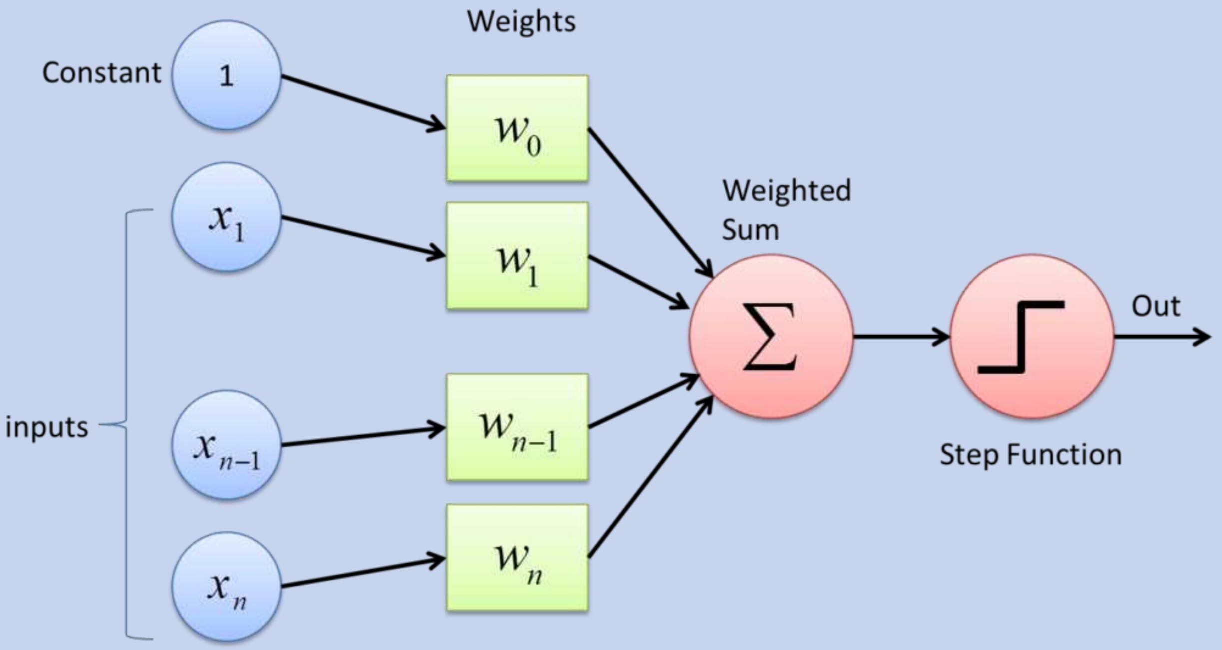
Since each output unit is independent of the others, we can limit our study to single output perceptrons.

“Artificial” Neuron



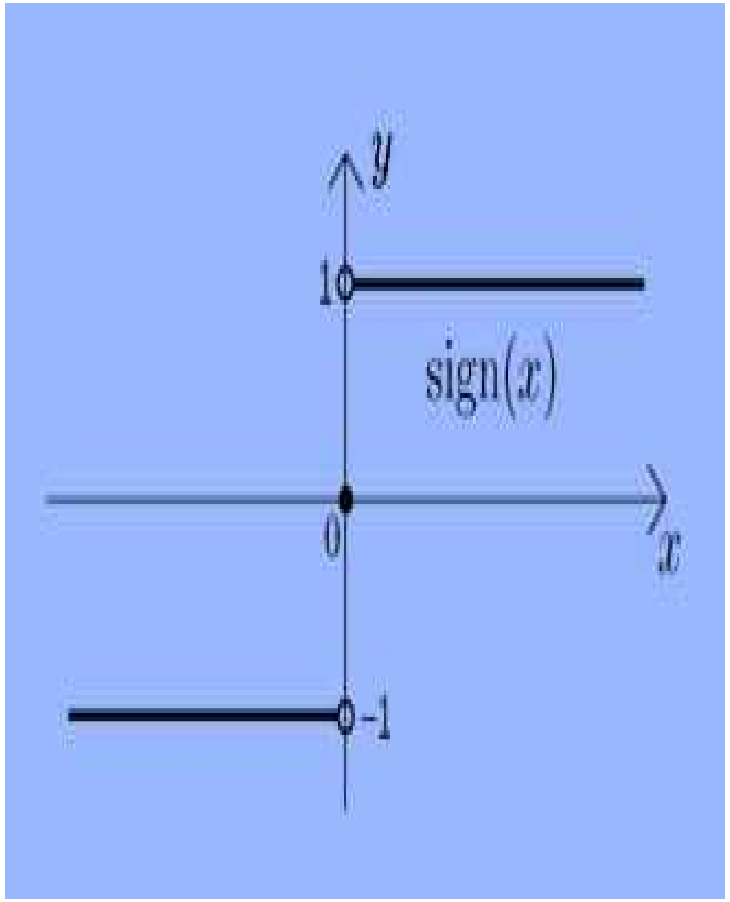
An example of a neuron showing the input (x_1 - x_n), their corresponding weights (w_1 - w_n), a bias (b) and the activation function f applied to the weighted sum of the inputs.

Traditional Perceptron



$$f(x) = sign(w \cdot x + b)$$

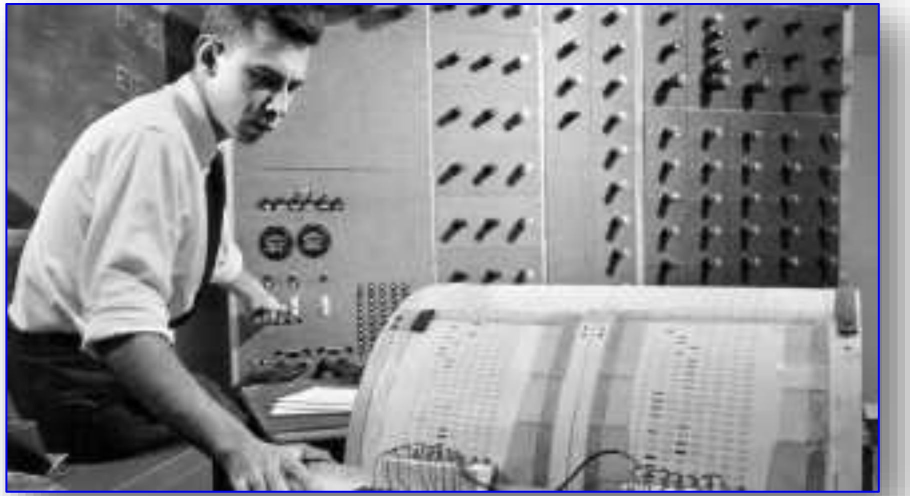
$$sign(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases}$$



Perceptron Weights are Learned Y= F (W∗X)

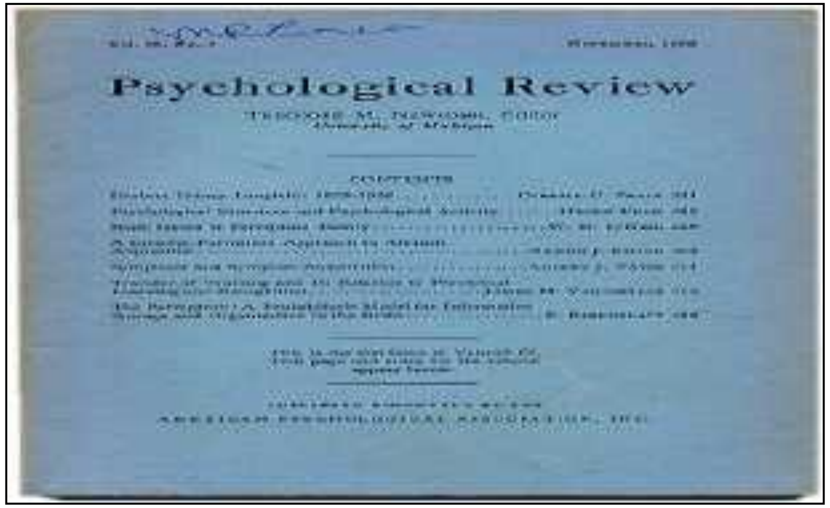
Initialize $\vec{w} = \vec{0}$	// Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.
while TRUE **do**	// Keep looping
$m = 0$	// Count the number of misclassifications, m
for $(x_i, y_i) \in D$ **do**	// Loop over each (data, label) pair in the dataset, D
if $y_i(\vec{w}^T \cdot \vec{x}_i) \leq 0$ **then**	// If the pair $(\vec{x}_i, y_i)$ is misclassified
$\vec{w} \leftarrow \vec{w} + y\vec{x}$	// Update the weight vector $\vec{w}$
$m \leftarrow m + 1$	// Counter the number of misclassification
end if	
end for	
if $m = 0$ **then**	// If the most recent $\vec{w}$ gave 0 misclassifications
break	// Break out of the while-loop
end if	
end while	// Otherwise, keep looping!

Perceptron and Rosenblatt



- In July 1958, the U.S. Office of Naval Research unveiled an invention.
- An IBM 704 – a 5-ton computer the size of a room – was fed a series of punch cards. After 50 trials, the computer taught itself to distinguish cards marked on the left from cards marked on the right.
- Frank Rosenblatt Cornell Ph.D. works on the “perceptron” – what he described as the first machine “capable of having an original idea.”

Perceptron


Psychological Review

ROSENBLATT, Frank. (Cornell Aeronautical Laboratory at Cornell University)

The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain.

In, Psychological Review, Vol. 65, No. 6, pp. 386-408, November, 1958.

“ Introducing the Perceptron – A machine which

senses,
recognizes,
remembers,
responses

like the human minds”

Perceptron From New York Times

NEW NAVY DEVICE LEARNS BY DOING

Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

…, July 7 (UPI)—… ealed the em… onic computer today … expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The embryo—the Weather Bureau's $2,000,000 "704" computer—learned to differentiate between right and left after fifty attempts in the Navy's demonstration for newsmen.

The service said it would use this principle to build the first of its Perceptron thinking machines that will be able to read and write. It is expected to be finished in about a year at a cost of $100,000.

Dr. Frank Rosenblatt, designer of the Perceptron, conducted the demonstration. He said the machine would be the first device to think as the human brain. As do human beings, Perceptron will make mistakes at first, but will grow wiser as it gains experience, he said.

Dr. Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers.

Without Human Controls

The Navy said the perceptron would be the first non-living mechanism "capable of receiving, recognizing and identifying its surroundings without any human training or control."

The "brain" is designed to remember images and information it has perceived itself. Ordinary computers remember only what is fed into them on punch cards or magnetic tape.

Later Perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech or writing in another language, it was predicted.

Mr. Rosenblatt said in principle it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

In today's demonstration, the "704" was fed two cards, one with squares marked on the left side and the other with squares on the right side.

Learns by Doing

In the first fifty trials, the machine made no distinction between them. It then started registering a "Q" for the left squares and "O" for the right squares.

Dr. Rosenblatt said he could explain why the machine learned only in highly technical …

falo, said Perceptrons might be fired to the planets as mechanical space explorers.

able to walk, talk, see, write, reproduce itself and be conscious of its existence.

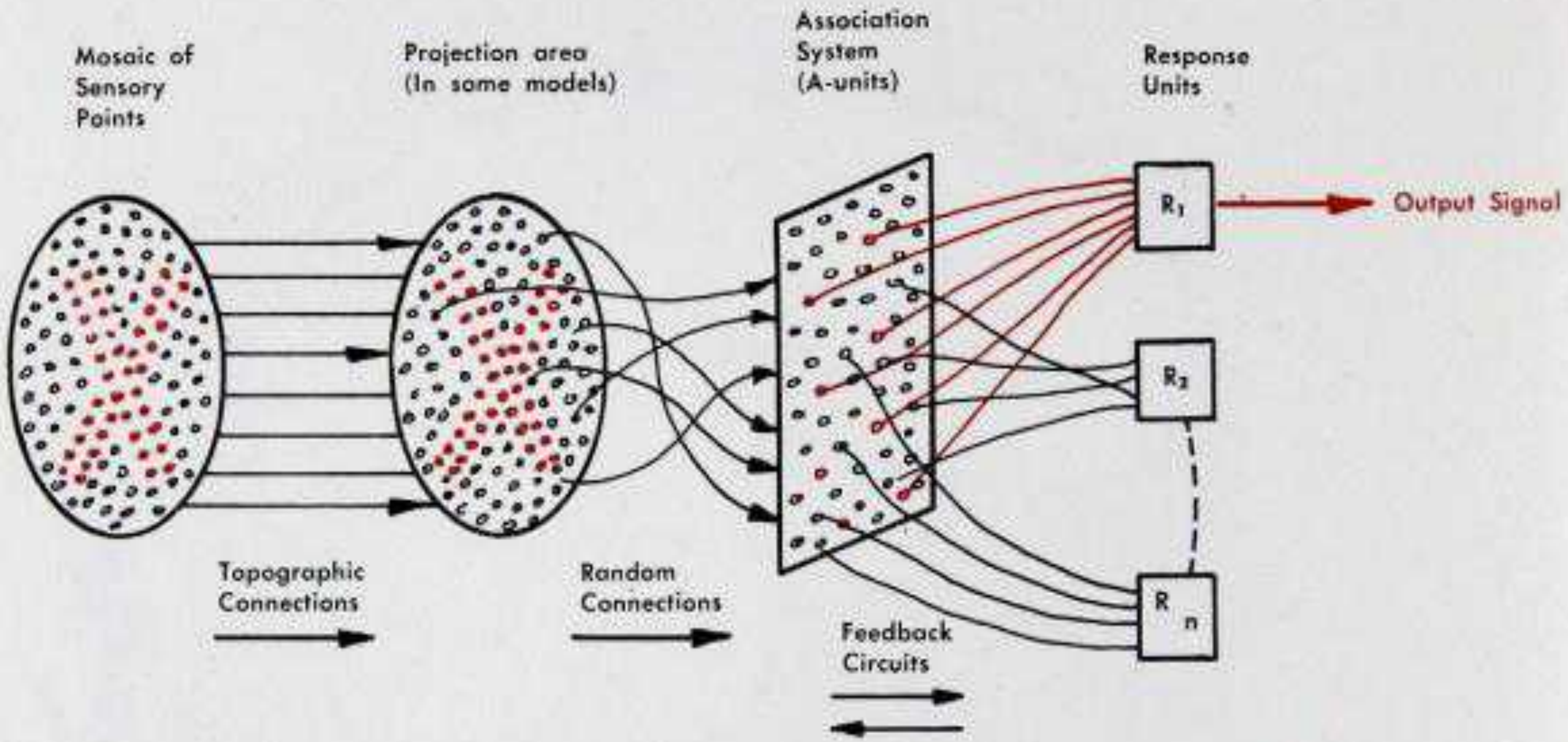
ciple it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

Perceptron and Rosenblatt

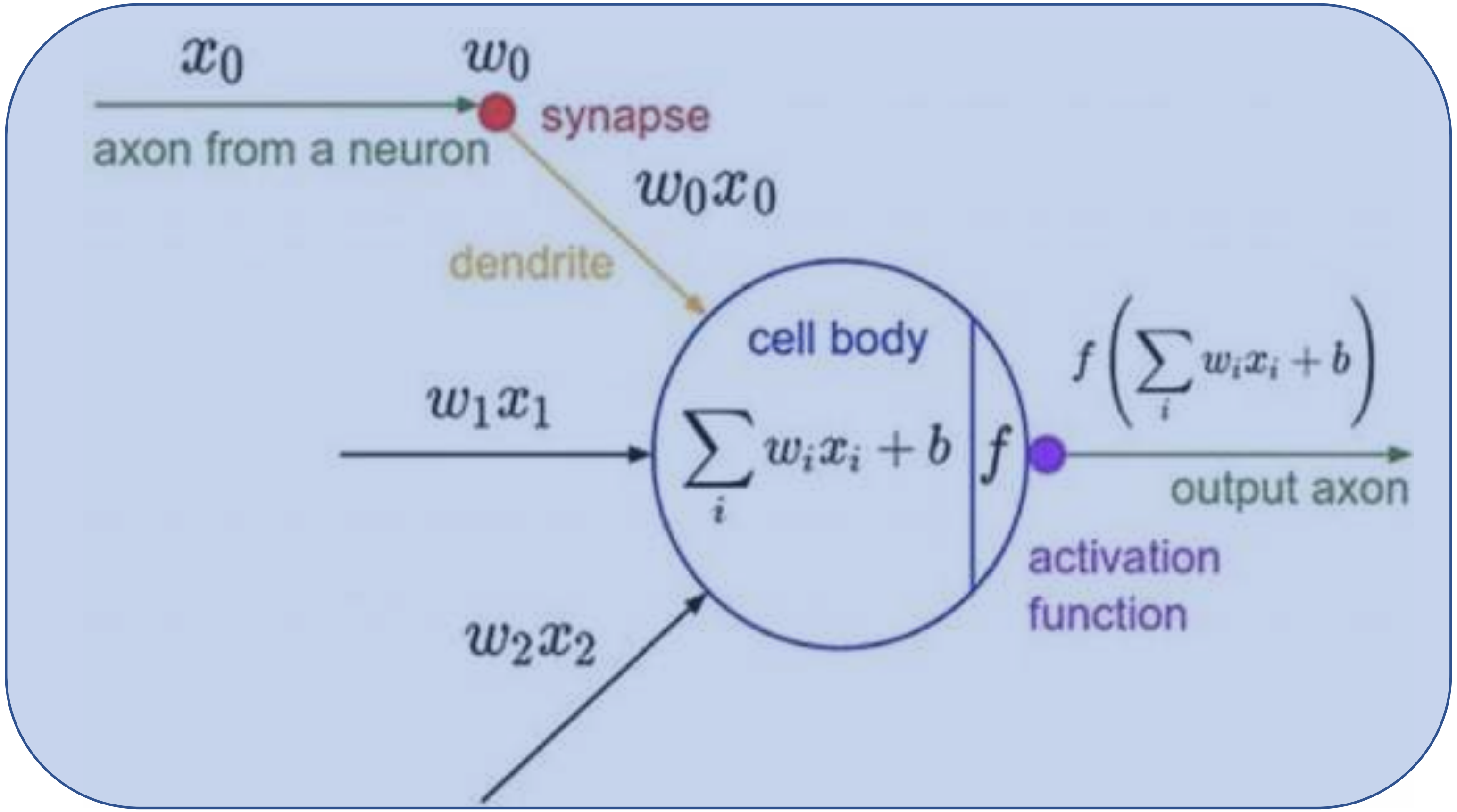
1957:	Rosenblatt's perceptrons were initially simulated on an IBM 704 computer at Cornell Aeronautical Laboratory in 1957.
1958:	The perceptron written by Frank Rosenblatt (his father was a medical doctor and he specilzed in psychological research) was first published in1958.
1959:	In 1959 he went to Cornell's Ithaca campus as director of the Cognitive Systems Research Program and also as a lecturer in the Psychology Department.
1962:	Principles of neurodynamics written by Frank Rosenblatt was first published in 1962.
1966:	In 1966 he joined the Section of Neurobiology and Behavior within the newly formed Division of Biological Sciences, as an associate professor.
1971:	Frank Rosenblatt died in July 1971 on his 43rd birthday, in a boating accident in Chesapeake Bay.

Perceptron and Human Brain

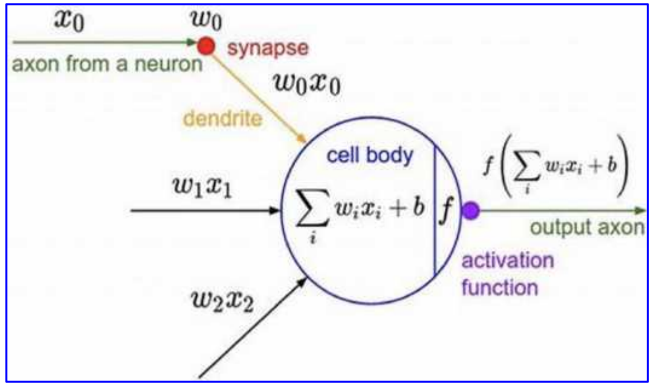
FIG. 1 — Organization of a biological brain. (Red areas indicate active cells, responding to the letter X.)



Perceptron Unit Mimics the Neuron

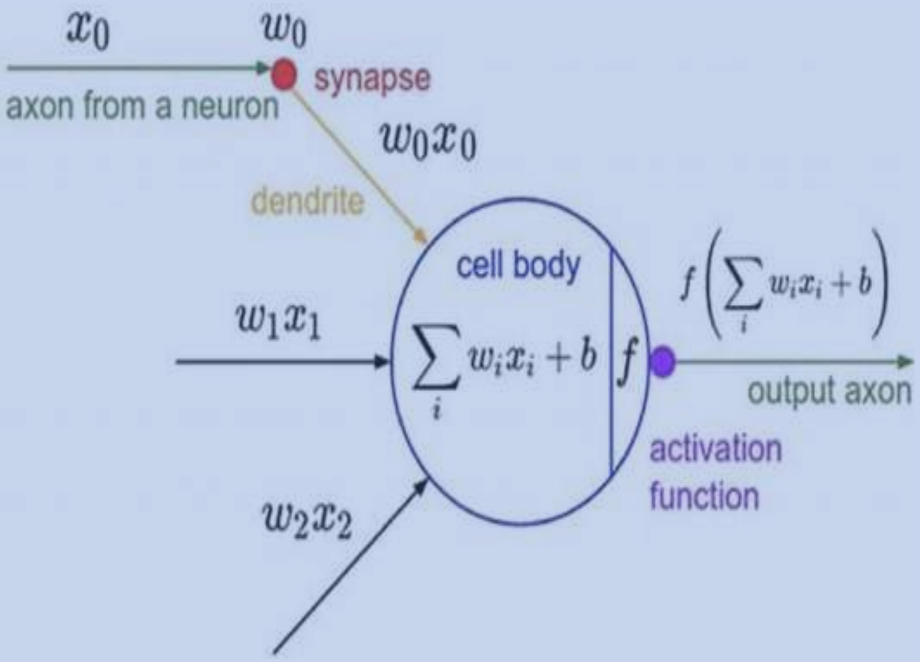


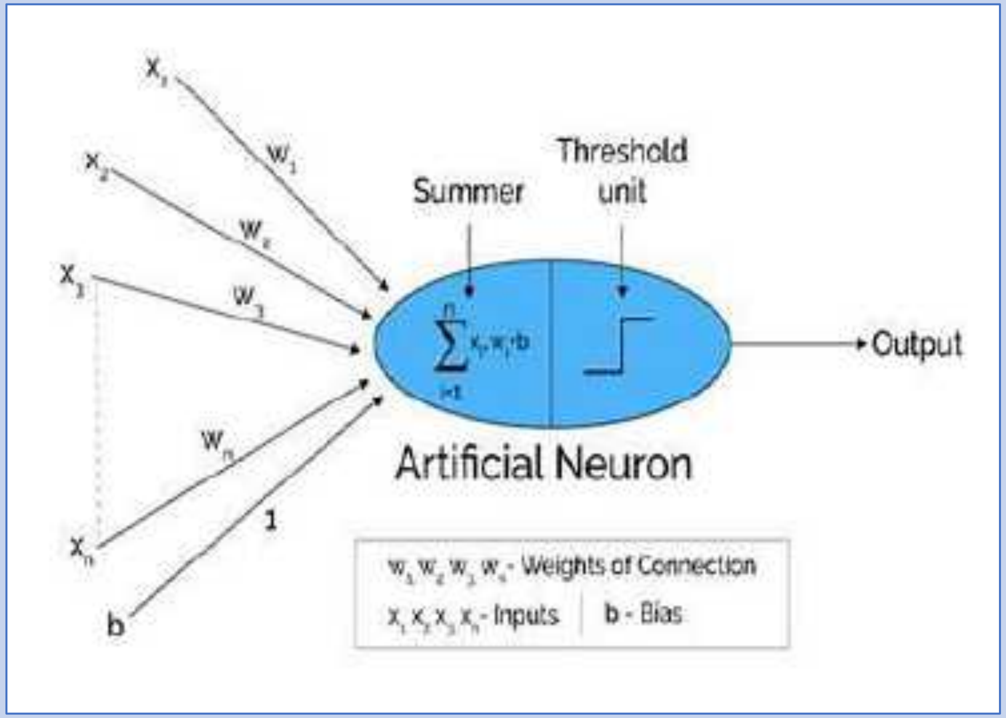
Perceptron Unit Mimics the Neuron



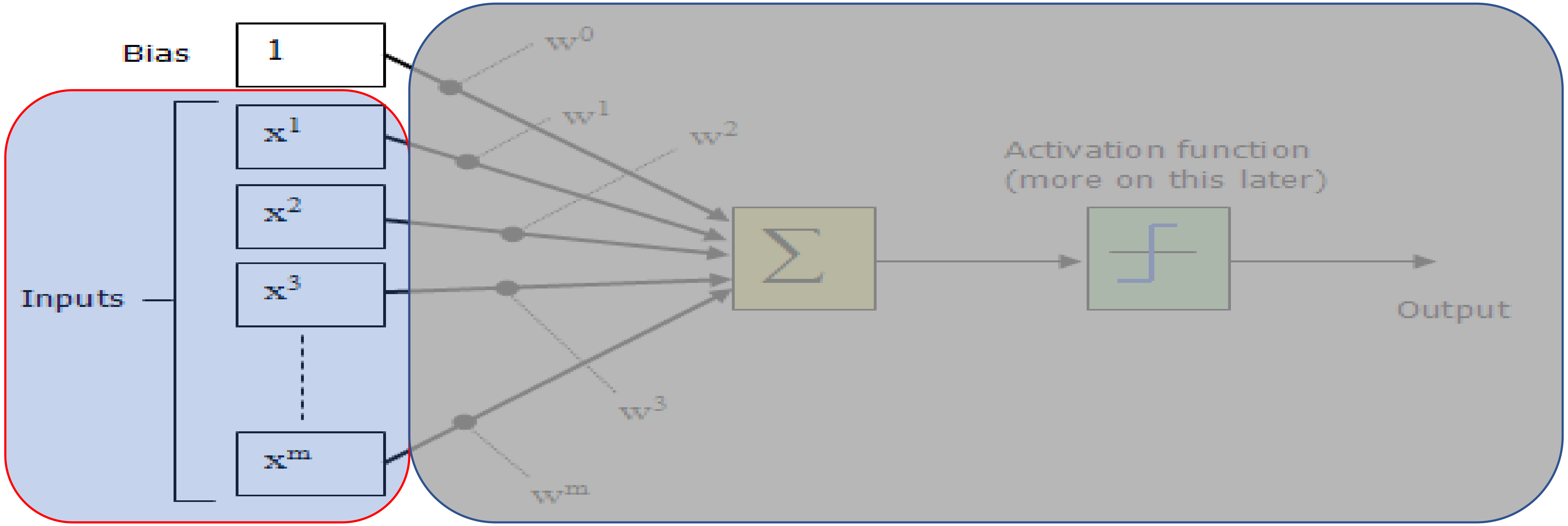
Inspired by the way neurons work together in the brain, the perceptron is a single-layer neural network – an algorithm that classifies input into two possible categories.
The neural network makes a prediction – say, right or left; or dog or cat – and if it's wrong, tweaks itself to make a more informed prediction next time. It becomes more accurate over thousands or millions of iterations.

Perceptron Unit Mimics the Neuron



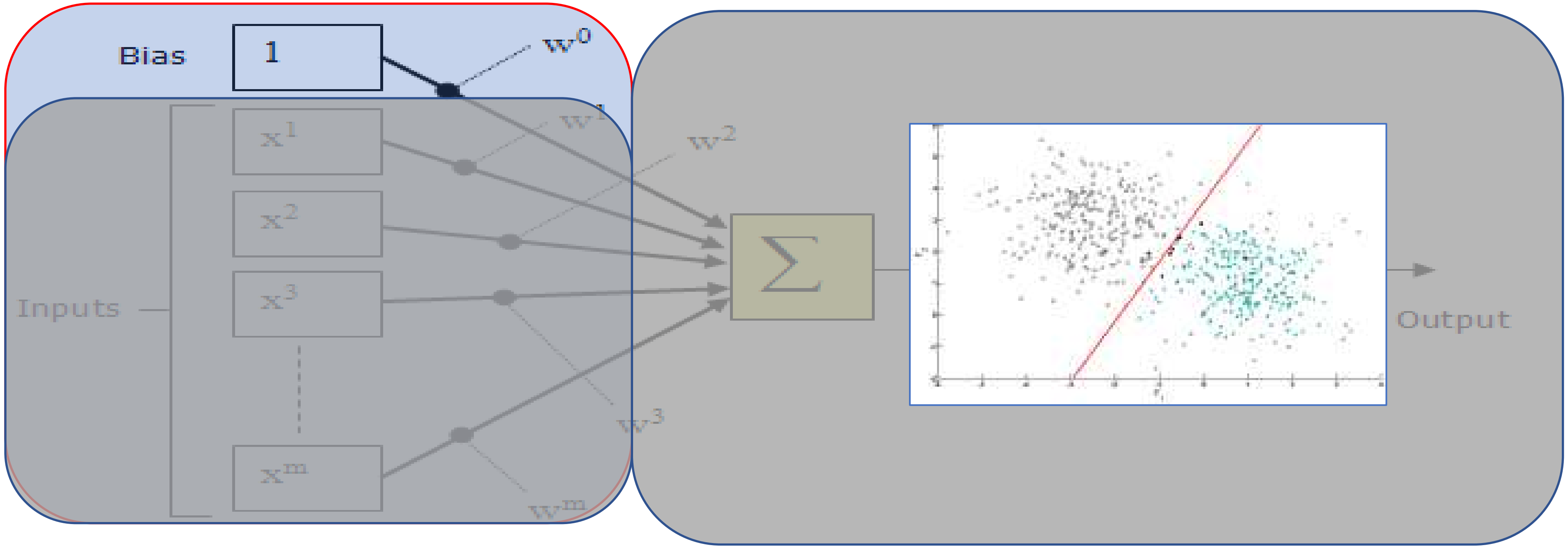


Perceptron 6 Components 1 - Input



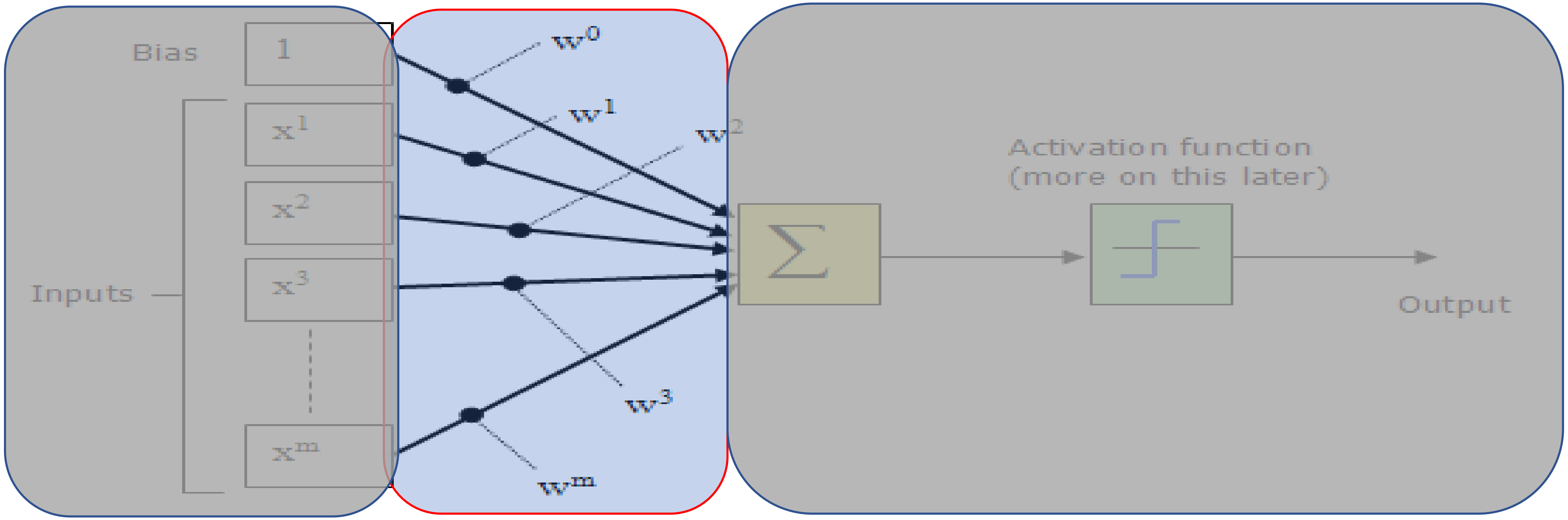
All the feature becomes the input for a perceptron. We denote the input of a perceptron by [x1, x2, x3, ..,xn], here x represent the feature value and n represent the total number of features. We also have special kind of input called the BIAS

Perceptron 6 Components 2 - Bias



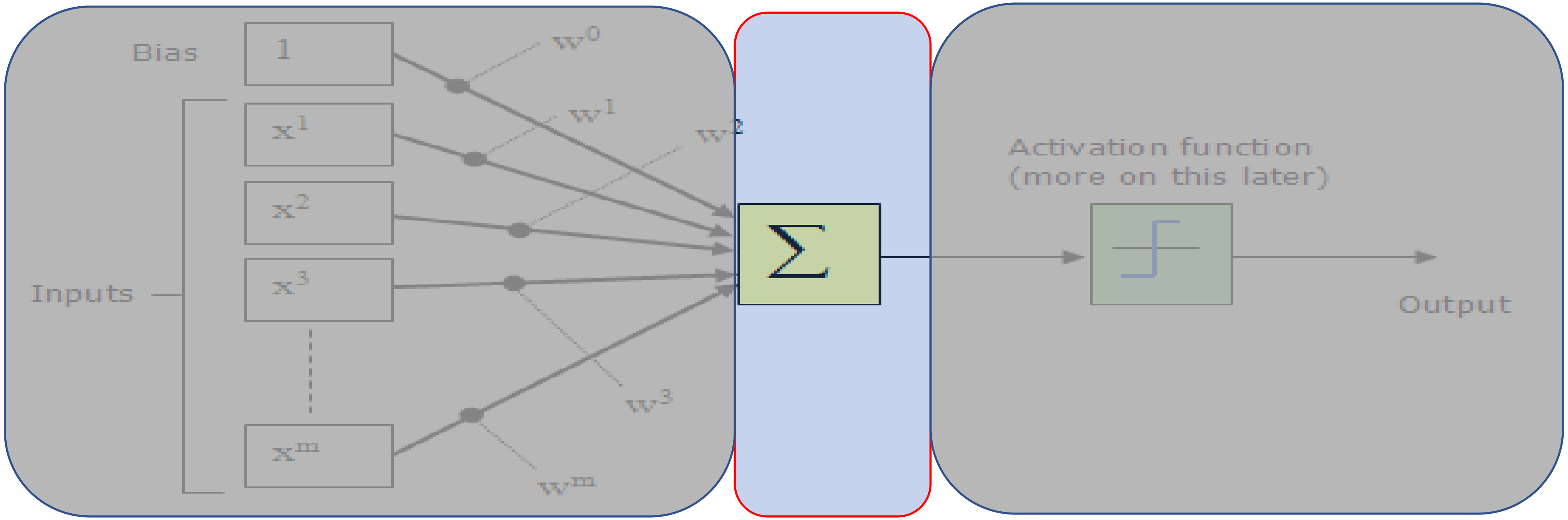
A bias neuron allows a classifier to shift the decision boundary left or right. In an algebraic term, the bias neuron allows a classifier to translate its decision boundary. To translation is to "move every point a constant distance in a specified direction". BIAS helps to training the model faster and with better quality.

Perceptron 6 Components 3 - Weight



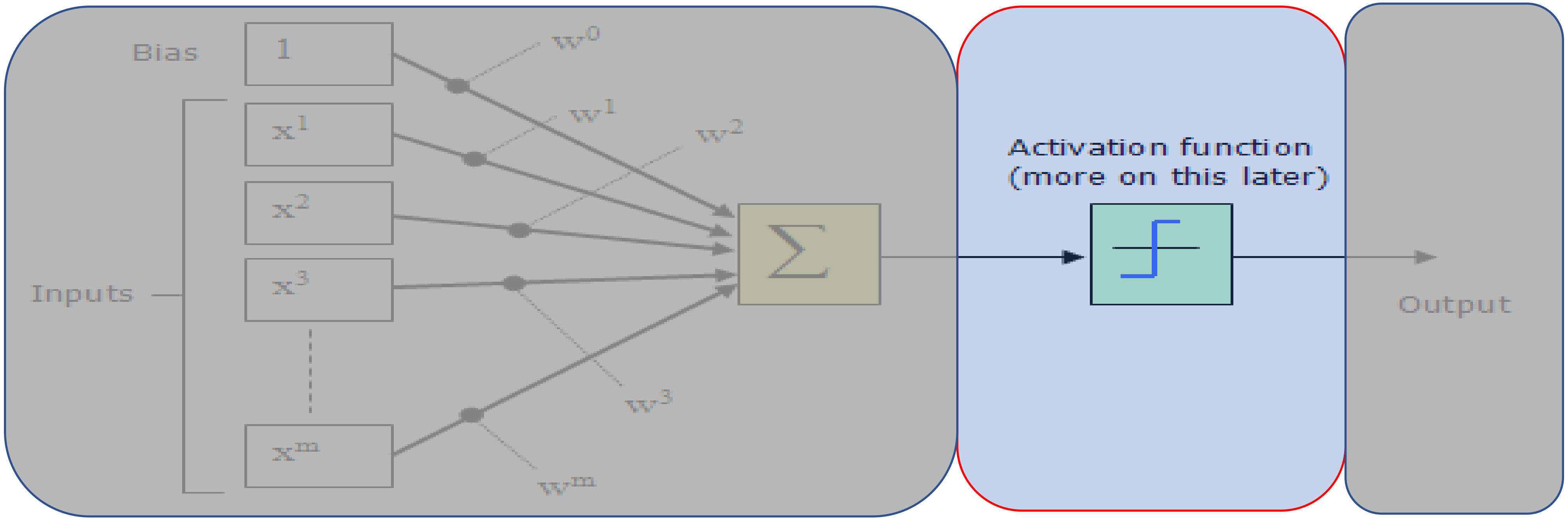
The weights offer an preliminary value in the very beginning of algorithm learning. With the occurrence of every training inaccuracy, the weights values are updated. These are mainly signified as w1, w2, w3, w4 and so on.

Perceptron 6 Components 4 – Weight Summation



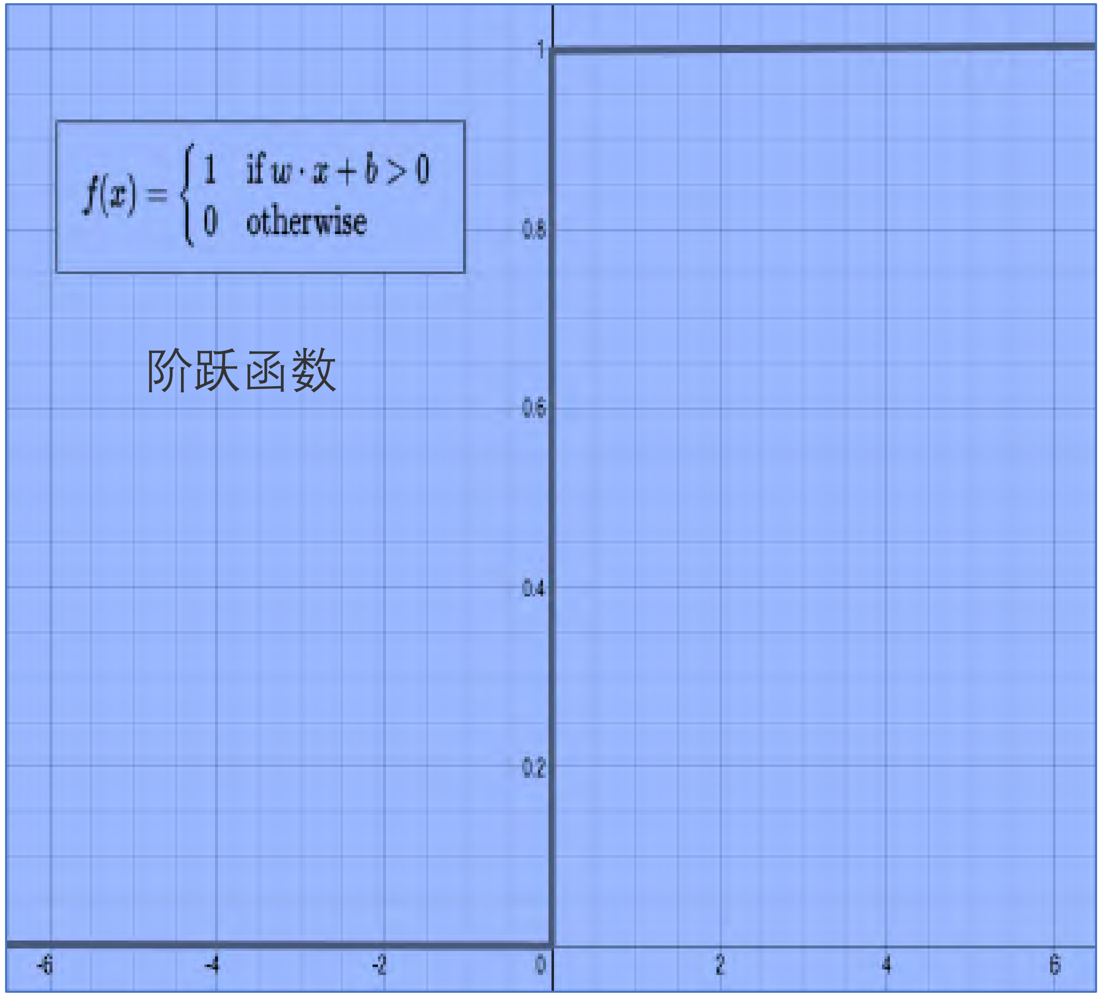
Weighted Summation is the sum of value that we get after the multiplication of each weight [wn] associated the each feature value[xn]. We represent the weighted Summation by $\sum w_i x_i$ for all **i -> [1 to n]**

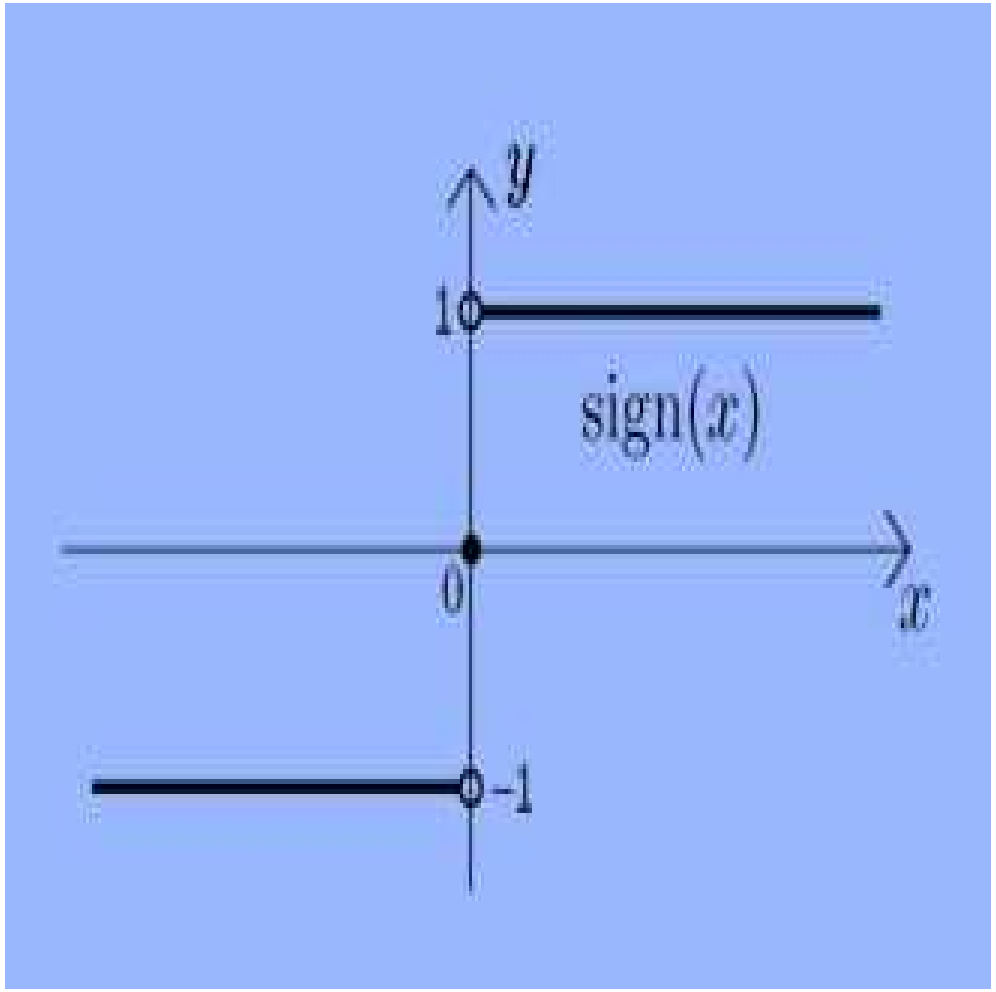
Perceptron 6 Components 5 – Transfer Function



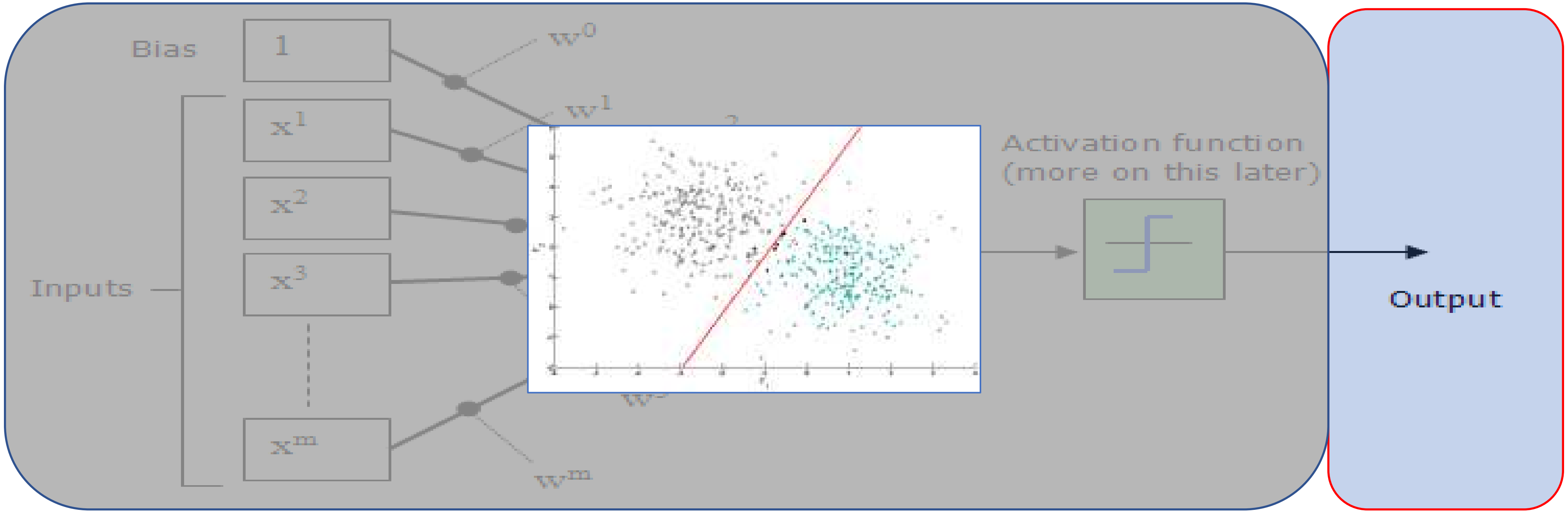
Transfer/Step/Activation Function:- the role of activation functions is make neural networks linear/non-linear. For Perceptron linearly classification of example, it typically uses Heaviside step function to make the perceptron as linear as possible.

Typical Perceptron Unit Step Transfer Function



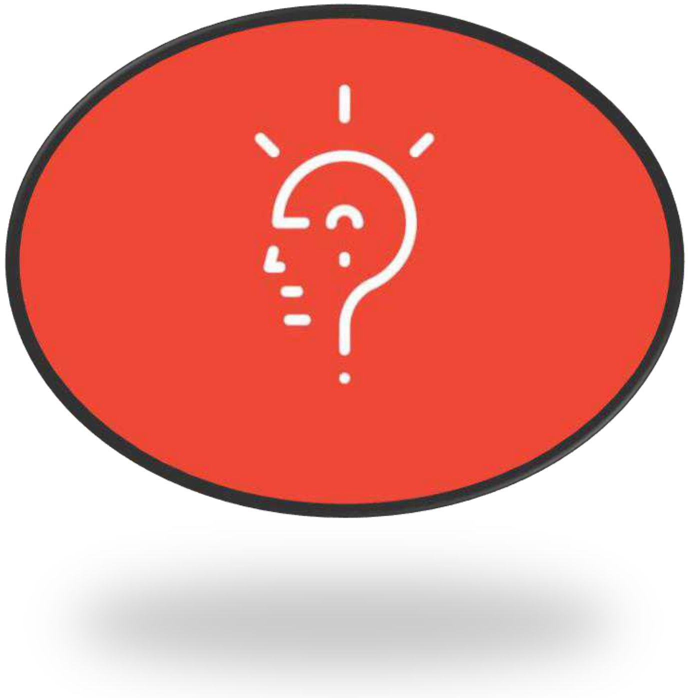


Perceptron 6 Components 6 – Output



Output:- The weighted Summation is passed to the step/activation function and whatever value we get after computation is our predicted output. (different classes, -1/1, 0/1, face/none-face, disease/non-disease, etc..)

Any Question?



Perceptron

3 4

1 Perceptron

2 Perceptron Learning

3 ADALINE

4 Limitation of Perceptron

Med

Recall: Perceptron Weights are Learned

Initialize $\vec{w} = \vec{0}$	// Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.
while TRUE **do**	// Keep looping
$m = 0$	// Count the number of misclassifications, m
for $(x_i, y_i) \in D$ **do**	// Loop over each (data, label) pair in the dataset, D
if $y_i(\vec{w}^T \cdot \vec{x}_i) \leq 0$ **then**	// If the pair $(\vec{x}_i, y_i)$ is misclassified
$\vec{w} \leftarrow \vec{w} + y\vec{x}$	// Update the weight vector $\vec{w}$
$m \leftarrow m + 1$	// Counter the number of misclassification
end if	
end for	
if $m = 0$ **then**	// If the most recent $\vec{w}$ gave 0 misclassifications
break	// Break out of the while-loop
end if	
end while	// Otherwise, keep looping!

Perceptron Learning Rule 1: PLR

1. Randomly choose the weights in the range 0 and 1.

2. Training examples are presented to perceptron one by one from the beginning, and its output is observed for each training example.

3. If the output is correct then the next training example is presented to perceptron.

4. If the output is incorrect then the weights are modified as per the following Perceptron Learning Rule (PLR).

$$\textbf{New Wi = Wi + } (\eta * \textbf{Xi} * \textbf{E}).$$

Change in Weight i = Learning Rate × Current Value of Input i × E (Expected Output, Current Output).

5. A simple form of E = (Expected Output - Current Output) or

SIGN (Expected Output - Current Output).

6. In PLR, output is 1/0 (or -1), and the transfer is Threshold Step Function

Typical Perceptron Weight Updates

- Weights modified for each example
- Update Rule:

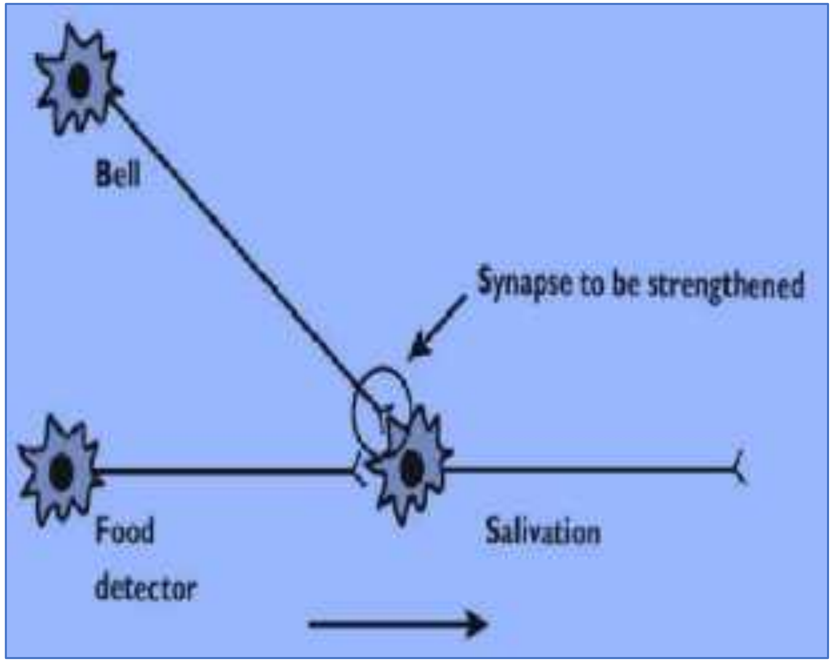
$$w_i \leftarrow w_i + \Delta w_i$$

where

$$\Delta w_i = \eta(t - o)x_i$$

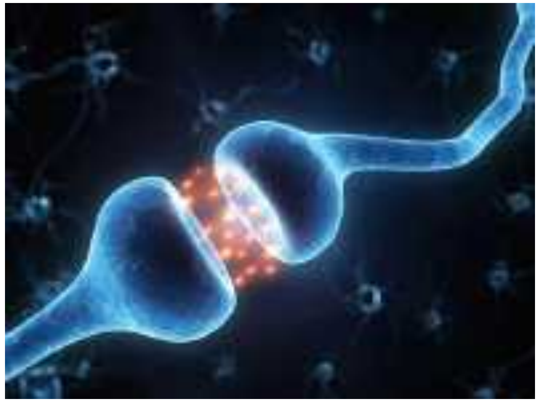
learning rate | target value | perceptron output | input value

PLR Foundation - Hebb’s Law



When an axon of Neuron A is near enough to excite a Neuron B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both Neurons such that A's efficiency, as one of the Neurons firing B, is increased

Hebb



- Donald Olding Hebb (1904-1985) was a Canadian psychologist pioneer of neuropsychology (of the study of the relationship between psychology and neuroscience).
- His book 'The Organization of Behavior'(1949) gives us a theory about behaviour, based on the physiology of the nervous system. It makes an important attempt to find the common between neurological and psychological conceptions.

Hebb Rule to Update Weights

Hebb Rule

$$w_{ij}^{new} = w_{ij}^{old} + \alpha\, f_i(a_{iq}) g_j(p_{jq})$$

Presynaptic Signal

Postsynaptic Signal

Simplified Form:

actual output

$$w_{ij}^{new} = w_{ij}^{old} + \alpha a_{iq} p_{jq}$$

input pattern

Supervised Form:

$$w_{ij}^{new} = w_{ij}^{old} + t_{iq} p_{jq}$$

Matrix Form:

desired output

$$\mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{t}_q \mathbf{p}_q^T$$

From Unsupervised Hebb Rule to PLR
- From "output" to "output difference"

- Feedforward unsupervised learning also known as "coincidence learning"
- The learning signal is equal to the neuron's output
- To update the weights of a neuron i, the inputs to neuron i come from a preceding neuron j (x_j)

$$w_{i,j} = w_{i,j} + \Delta w_{i,j}$$

$$\Delta w_{i,j} = \alpha * \text{output}_i * \text{input}_j$$

$$\Delta w_{i,j} = \alpha\, o_i x_j$$

- It is clear that Hebbian learning is not going to get our Perceptron to learn a set of training data, since weight changes depend only on the actual outputs and we don't have desired outputs to compare to.
- ∴ Hebb rule can be used for pattern association, pattern categorization, pattern classification and over a range of other areas

Rule 2: Perceptron Converge Theorem

The Perceptron convergence theorem states that for any data set which is linearly separable the Perceptron learning rule is guaranteed to find a solution in a finite number of steps.

In other words, the Perceptron learning rule is guaranteed to converge to a weight vector that correctly classifies the examples provided the training examples are linearly separable.

Rule 2: Perceptron Converge Theorem

Theorem 3 (Perceptron convergence). *The Perceptron Learning Algorithm makes at most $\frac{R^2}{\gamma^2}$ updates (after which it returns a separating hyperplane).*

Proof. It is immediate from the code that should the algorithm terminate and return a weight vector, then the weight vector must separate the $+$ points from the $-$ points. Thus, it suffices to show that the algorithm terminates after at most $\frac{R^2}{\gamma^2}$ updates. In other words, we need to show that k is upper-bounded by $\frac{R^2}{\gamma^2}$. Our strategy to do so is to derive both lower and upper and bounds on the length of $\mathbf{w}^{k+1}$ in terms of k, and to relate them.

Note that $\mathbf{w}^1 = \mathbf{0}$, and for $k \geqslant 1$, note that if $\mathbf{x}^j$ is the misclassified point during iteration k, we have

$$\begin{aligned} \mathbf{w}^{k+1} \cdot \mathbf{w}^* &= (\mathbf{w}^k + y^j \mathbf{x}^j) \cdot \mathbf{w}^* \\ &= \mathbf{w}^k \cdot \mathbf{w}^* + y^j (\mathbf{x}^j \cdot \mathbf{w}^*) \\ &> \mathbf{w}^k \cdot \mathbf{w}^* + \gamma. \end{aligned}$$

It follows by induction that $\mathbf{w}^{k+1} \cdot \mathbf{w}^* > k\gamma$. Since $\mathbf{w}^{k+1} \cdot \mathbf{w}^* \leqslant \|\mathbf{w}^{k+1}\| \|\mathbf{w}^*\| = \|\mathbf{w}^{k+1}\|$, we get

$$\|\mathbf{w}^{k+1}\| > k\gamma. \tag{1}$$

To obtain an upper bound, we argue that

$$\begin{aligned} \|\mathbf{w}^{k+1}\|^2 &= \|\mathbf{w}^k + y^j \mathbf{x}^j\|^2 \\ &= \|\mathbf{w}^k\|^2 + \|y^j \mathbf{x}^j\|^2 + 2(\mathbf{w}^k \cdot \mathbf{x}^j) y^j \\ &= \|\mathbf{w}^k\|^2 + \|\mathbf{x}^j\|^2 + 2(\mathbf{w}^k \cdot \mathbf{x}^j) y^j \\ &\leqslant \|\mathbf{w}^k\|^2 + \|\mathbf{x}^j\|^2 \\ &\leqslant \|\mathbf{w}^k\|^2 + R^2, \end{aligned}$$

from which it follows by induction that

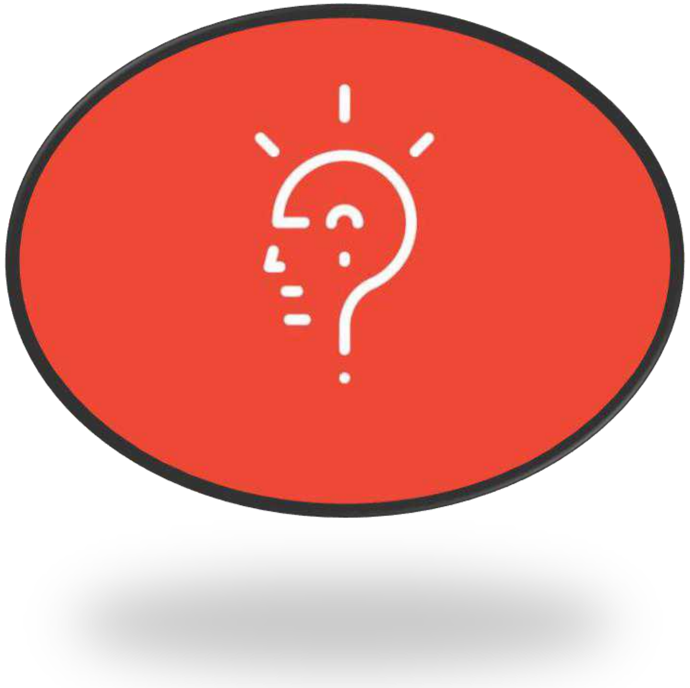
$$\|\mathbf{w}^{k+1}\|^2 \leqslant kR^2. \tag{2}$$

Together, (1) and (2) yield

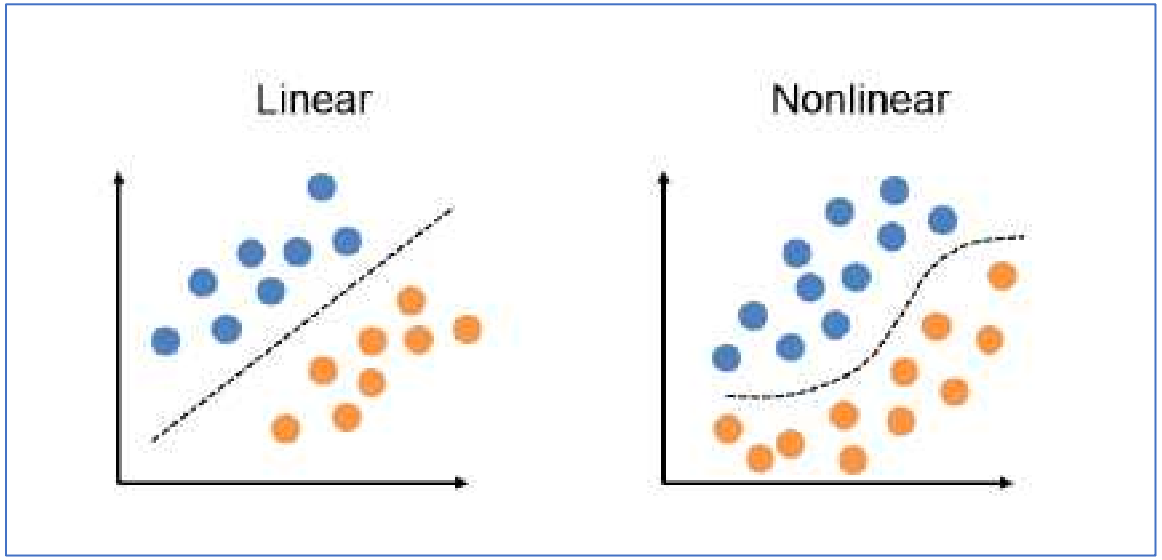
$$k^2\gamma^2 < \|\mathbf{w}^{k+1}\|^2 \leqslant kR^2,$$

which implies $k < \frac{R^2}{\gamma^2}$. Our proof is done. □

Any Question?



Linearly Separable



A function is said to be linearly separable when its outputs can be discriminated by a function which is a linear combination of features, that is we can discriminate its outputs by a line or a hyperplane.

Homework 05

1. Start to Write the Introduction of Your Group Project
2. Review All the Previous Lectures and Prepare for the Mid-term Exam

CS 103 -05

Perceptron and AI Early Day Algorithms

Jimmy Liu 刘江

2020-10-16