

Soft computing



K.N. Toosi  
University of  
Technology

محاسبات نرم



Hasan Ghasemzadeh  
<http://wp.kntu.ac.ir/ghasemzadeh>

Soft Computing

Neural Networks

شبکه عصبی



### مغز انسان

The diagram shows a lateral view of the human brain with the following color-coded regions and functions:

- Frontal Lobe (لوب پیشانی):** Problem solving, Emotional traits, Reasoning (judgment), Speaking, Voluntary motor activity.
- Parietal Lobe (لوب آهیانه‌ای):** Knowing right from left, Sensation, Reading, Body orientation.
- Occipital Lobe (لوب پس سری):** Vision, Color perception.
- Temporal Lobe (لوب گیجگاهی):** Understanding language, Behavior, Memory, Hearing.
- Cerebellum (مخچه):** Balance, Coordination and control of voluntary movement, Fine muscle control.
- Brain Stem (ساقه مغز):** Breathing, Body temperature, Digestion, Alertness/sleep, Swallowing.

Soft Computing

3

### برخی ویژگیهای مغز

- یک توده بافت عصبی (مغز انسان - حدود ۱/۴ کیلوگرم و ۱/۵ لیتر )
- هدایت کل جسم بصورت خودآگاه و ناخودآگاه
- مرکز اندوختن تجربه، یادگیری و تصمیم گیری
- حجم حافظه بالا و سرعت پردازش زیاد
- نامشخص بودن نحوه عملکرد مغز (تاکنون)

The diagram on the right provides a more detailed view of the brain's functional areas:

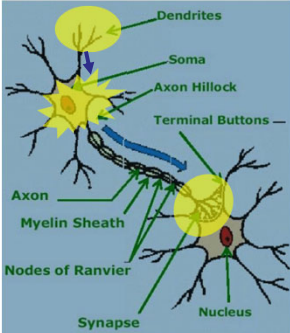
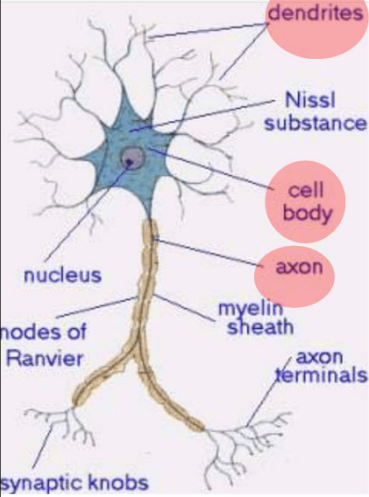
- Central sulcus:** Separates the frontal and parietal lobes.
- Primary motor cortex:** Involved with the control of voluntary muscles.
- Somatosensory cortex:** For cutaneous and proprioceptive senses.
- Frontal lobe:** Includes the motor speech area (Broca's area).
- Parietal lobe:** Includes the general interpretive area.
- Auditory area:** Located in the temporal lobe.
- Lateral sulcus:** Separates the temporal lobe from the frontal and parietal lobes.
- Occipital lobe:** Involved in combining visual images and visual recognition of objects.
- Temporal lobe:** Involved in the interpretation of sensory experiences and memory of visual and auditory patterns.
- Cerebellum:** Located at the back and bottom of the brain.
- Brain stem:** The base of the brain.

Soft Computing

4

### برخی ویژگیهای مغز

- مغز شامل ۸۶ میلیارد نورون (۱۹ درصد در قشر حدود ۳ میلیمتری مغز و ۸۰ درصد در مخچه)
- نورون‌ها از یک جسم سلولی، دندریت و آکسون تشکیل شده‌اند.
- نورون اطلاعات را از طریق سیگنال‌های الکتریکی و شیمیایی منتقل می‌کند.
- نورون‌ها انرژی خود را از طریق یک شکاف کوچک به نام سیناپس (بیش از ۱۰۰ تریلیون سیناپس) به یکدیگر منتقل می‌کنند



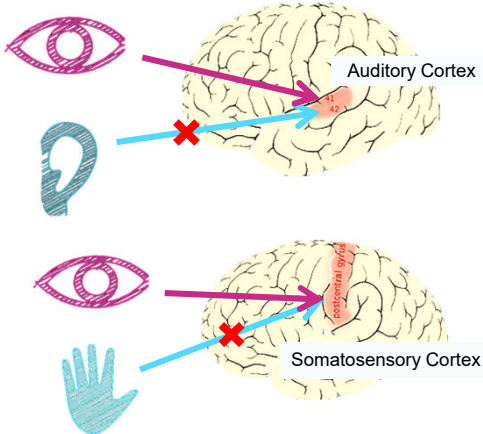
The soma, sums the incoming signals. When sufficient input is received, the cell fires; that is it transmit a signal over its axon to other cells

5

### برخی ویژگیهای مغز

#### The “one learning algorithm” hypothesis

گرچه هر قسمت مغز وظیفه مخصوص خود را انجام می‌دهد ولی در شرایط خاص محرومیت از برخی سنسورها سعی بر جابجایی وظایف خود میکند



Auditory cortex learns to see  
[Roe et al., 1992]

Somatosensory cortex learns to see  
[Metin & Frost, 1989]

Soft Computing

6

### مقایسه مغز انسان و رایانه

| Feature             | Human (Brain)                               | Computer                                     |
|---------------------|---------------------------------------------|----------------------------------------------|
| Processing Elements | ~86 billion neurons                         | 100 billion in modern chips (like Apple M2). |
| Interconnects       | ~10,000 synaptic connections per neuron     | a few interconnects per transistor.          |
| Cycles per Second   | ~200 Hz to 1000 Hz (neuron firing rate)     | Up to 5 GHz                                  |
| Memory              | Dynamic and associative (estimated ~2.5 PB) | Explicit, structured (up to terabytes).      |
| 2X Improvement      | Evolved over millions of years              | ~1.5–2 years                                 |

Neuron

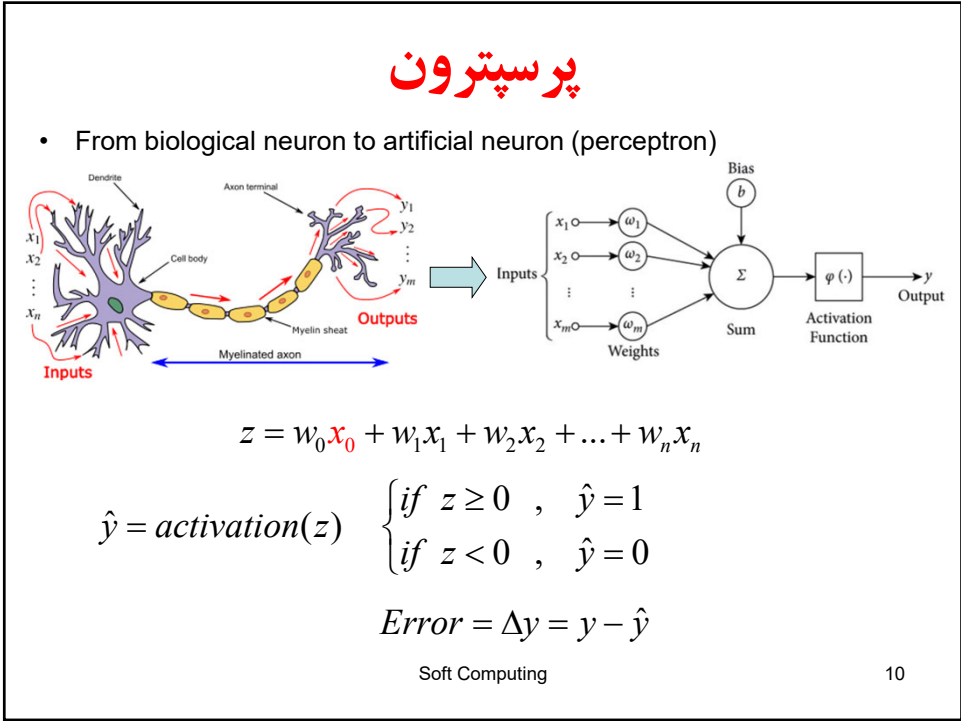
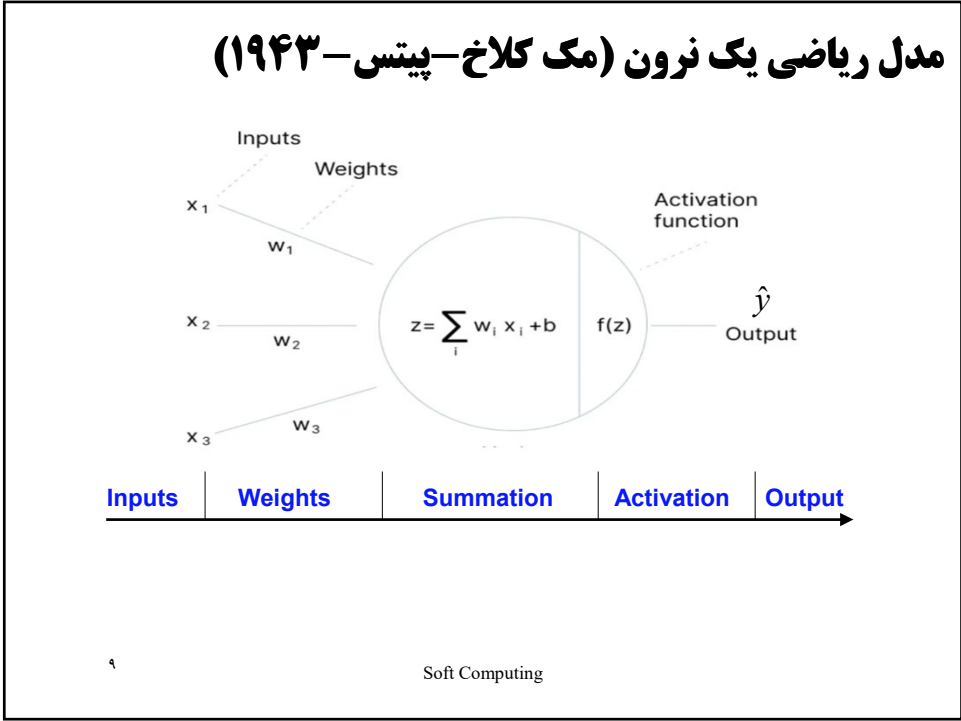
Soft Computing7

### پرسپترون

- شبکه عصبی مصنوعی از رفتار نورنهای مغز تقلید می کند.
- یک واحد منطقی مدلی از یک نورن است. Neuron model: Logistic unit
- شبکه‌های تک لایه، با توابع فعال سازی آستانه‌ای، توسط روزنبلات (۱۹۶۲) بنیان گذاری شدند که این نوع شبکه‌ها، پرسپترون نامیده شدند.

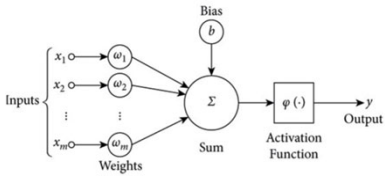
Sigmoid (logistic) activation function

Soft Computing8



پرسپترون

• Activation function



$$z = w_0x_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n$$
$$\begin{cases} \text{if } z \geq 0, & \hat{y} = 1 \\ \text{if } z < 0, & \hat{y} = 0 \end{cases}$$

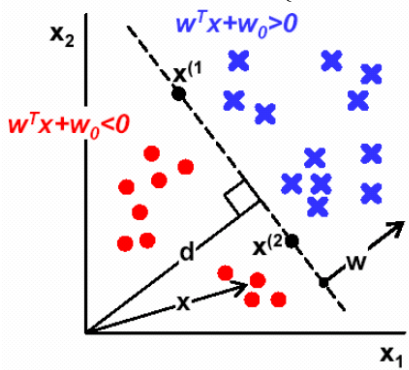
$$w_i + \eta \cdot \Delta y \cdot x_i \rightarrow w_i$$

$x_i$  Input feature value

$w_i$  Activation function

$\Delta y$  Error

$\eta$  Learning rate



Soft Computing

11

مثال پرسپترون: Logical AND

|      | x1  | x2  | T   |
|------|-----|-----|-----|
| P1 = | 0   | 1   | 0   |
| P2 = | 1   | 1   | 1   |
| Pn = | ... | ... | ... |

الگوی P2 است که باید پرسپترون یاد بگیرد:  
یعنی، اگر هر دو ورودی 1 هستند، خروجی باید 1 باشد

inputs

$x_1=1$

weights

$w_1=2$

$x_2=1$

$w_2=3$

$\Sigma$

threshold

$>8?$

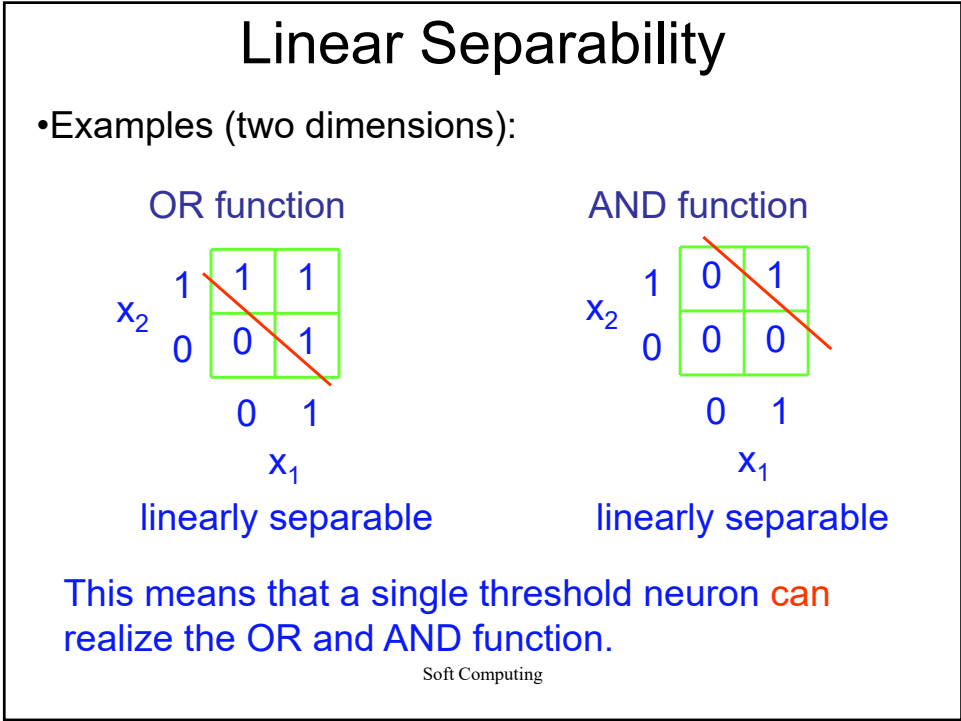
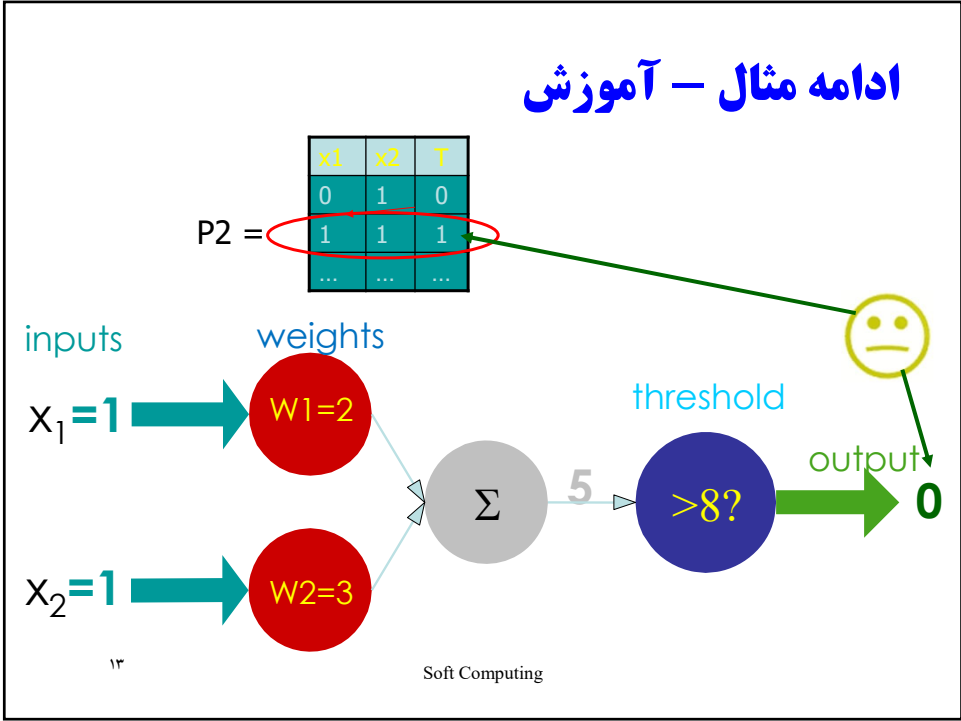
output

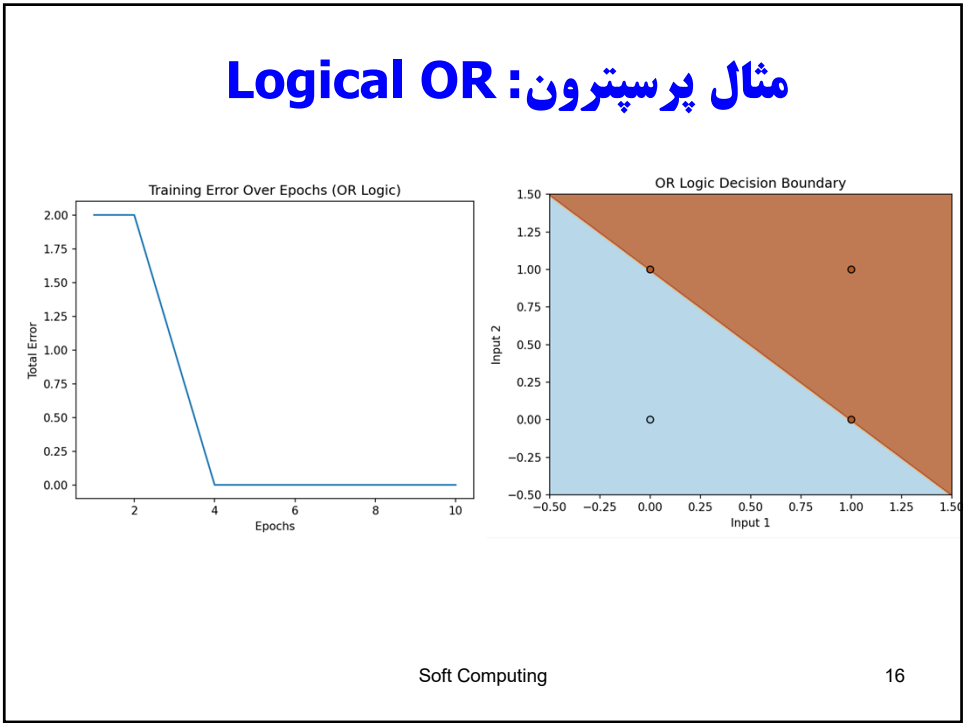
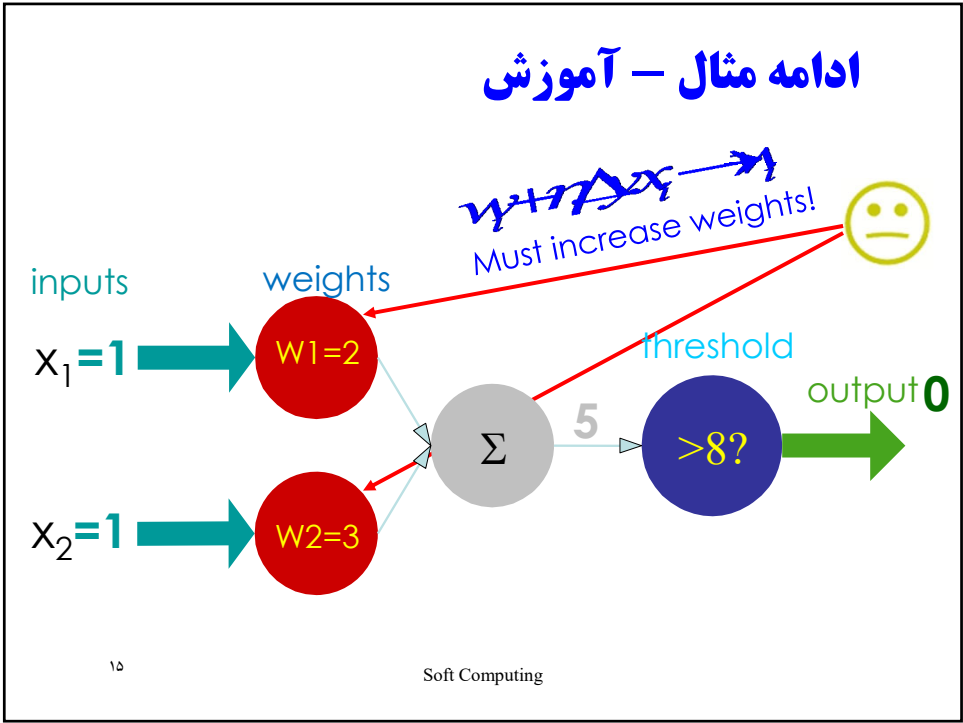
output

Soft Computing

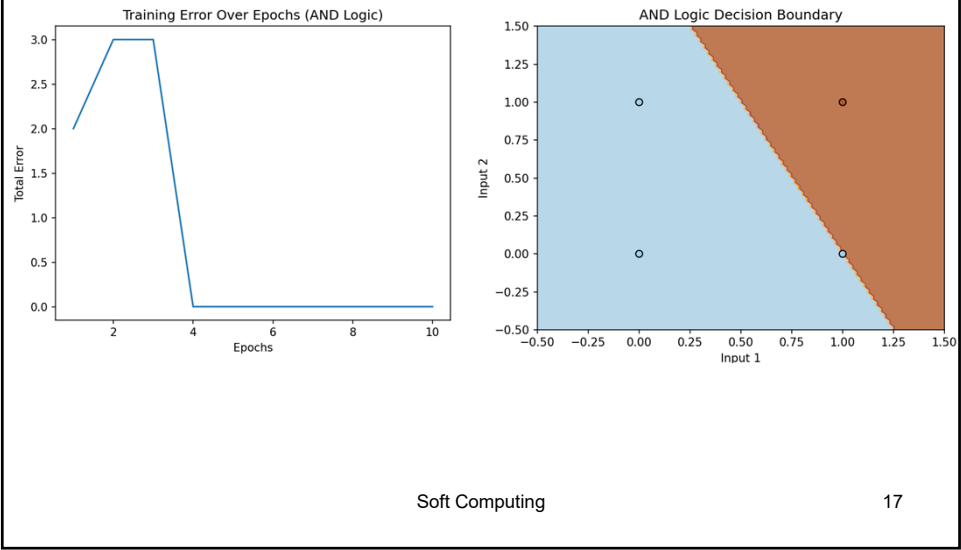
Dr. Hasan Ghasemzadeh

6





# مثال پرسپترون: Logical AND



# Linear Inseparable

•Examples (two dimensions):

XOR function

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

linearly inseparable

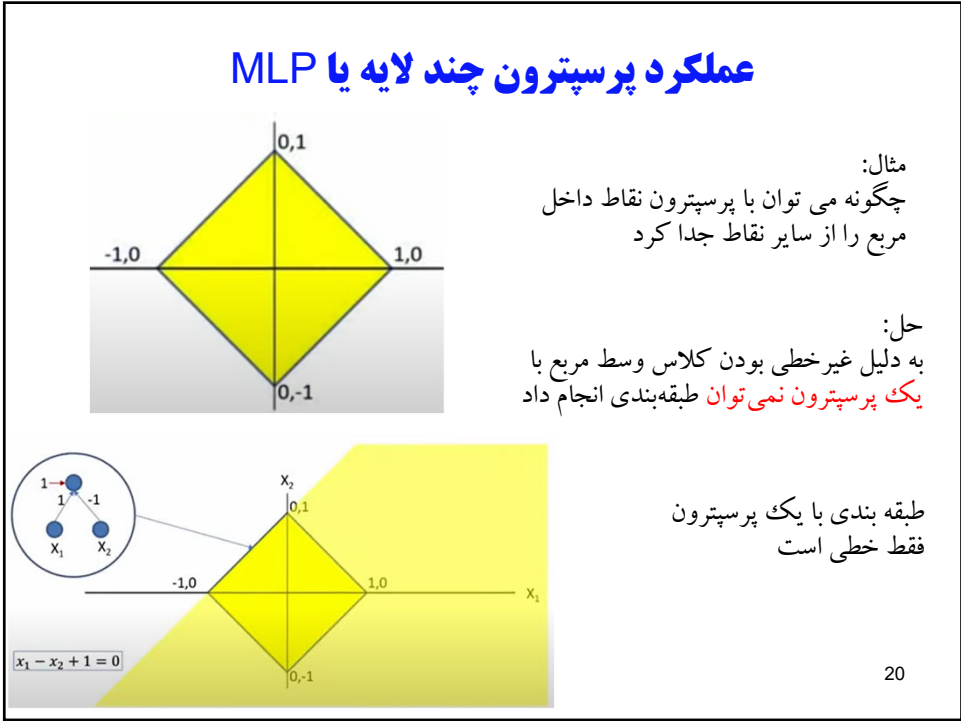
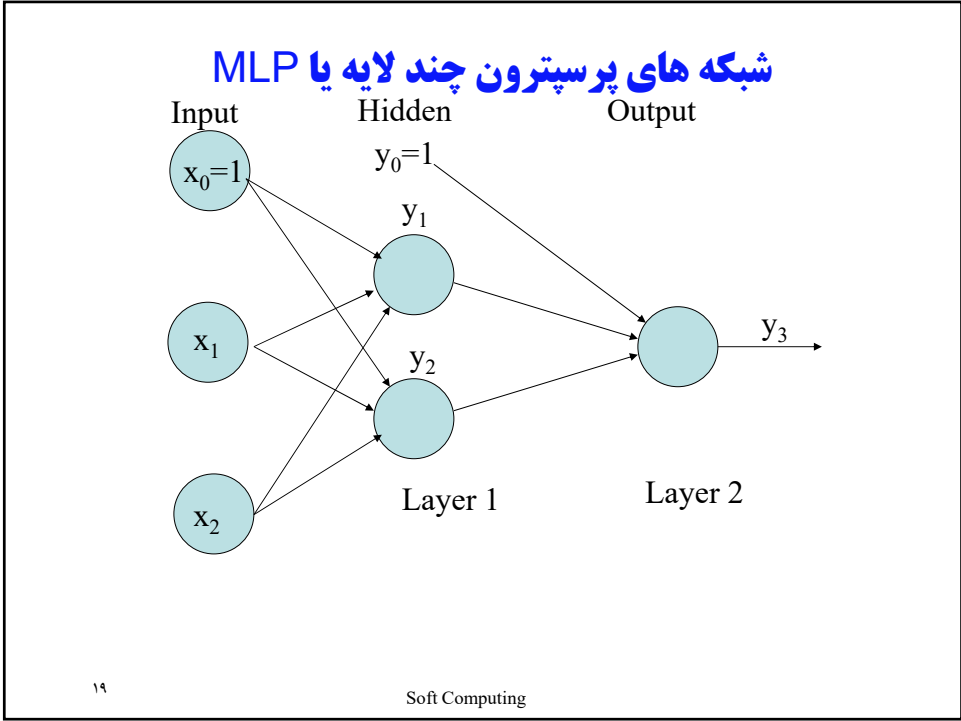
XNOR function

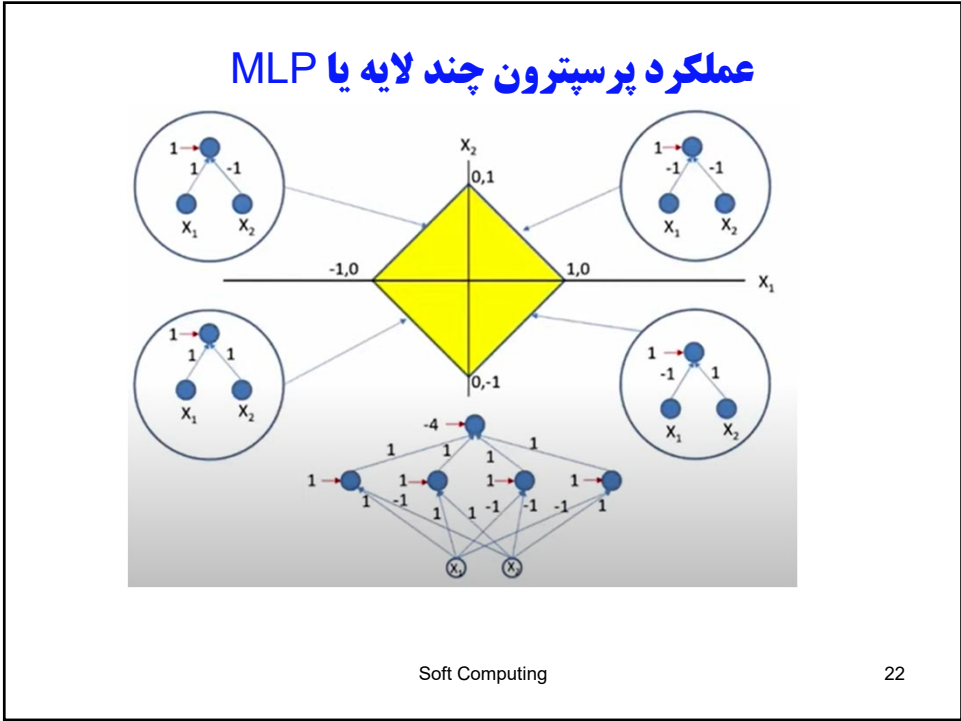
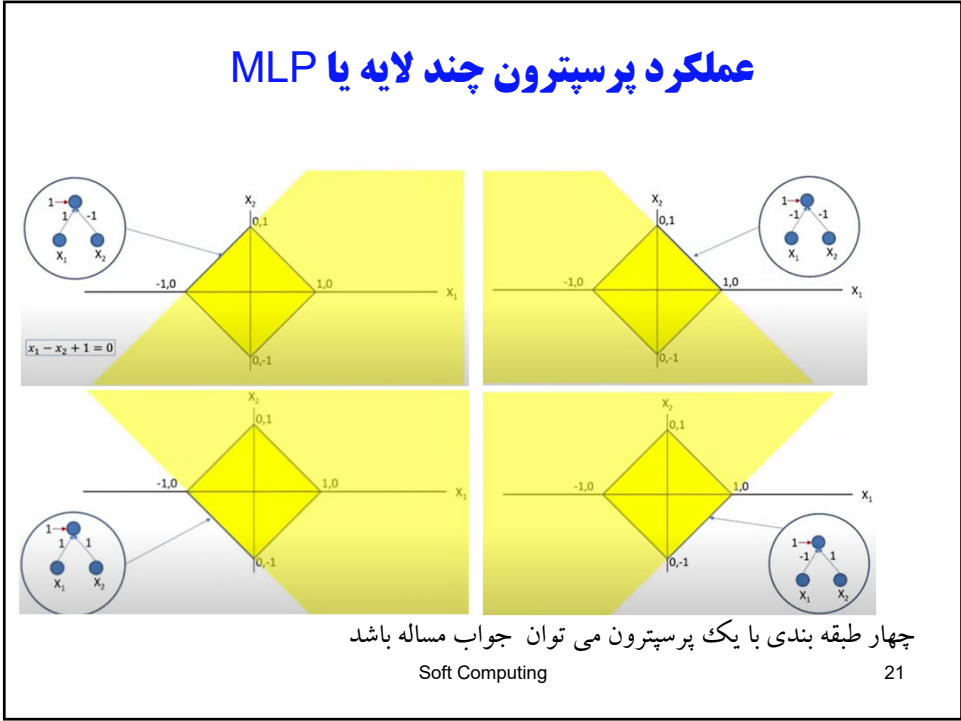
|       |   |       |   |
|-------|---|-------|---|
| $x_2$ | 1 | 0     | 1 |
|       | 0 | 1     | 0 |
|       |   | 0     | 1 |
|       |   | $x_1$ |   |

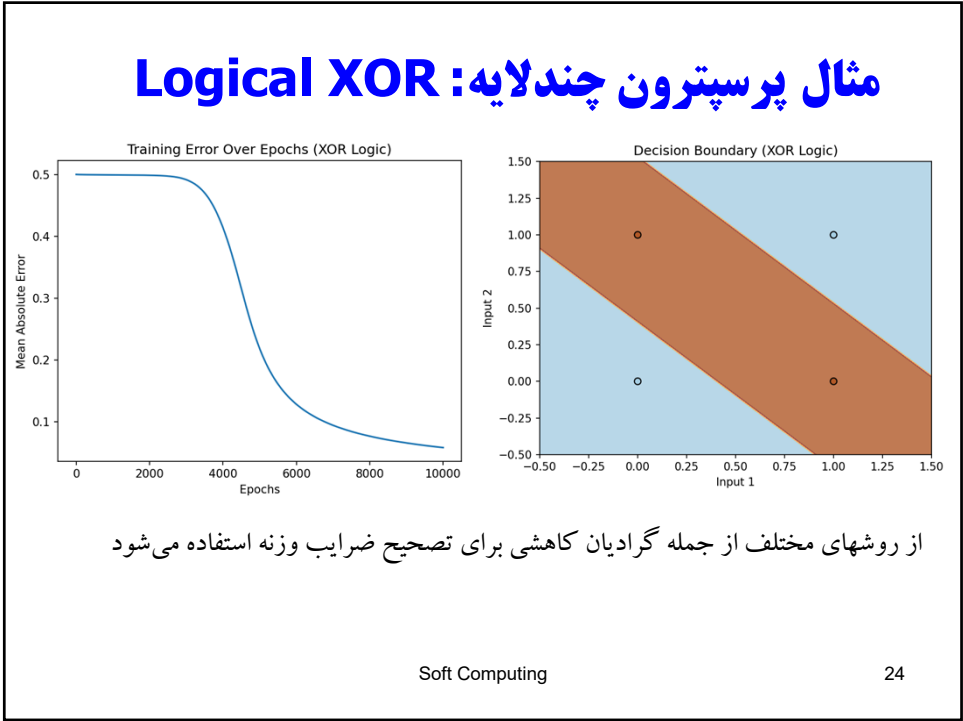
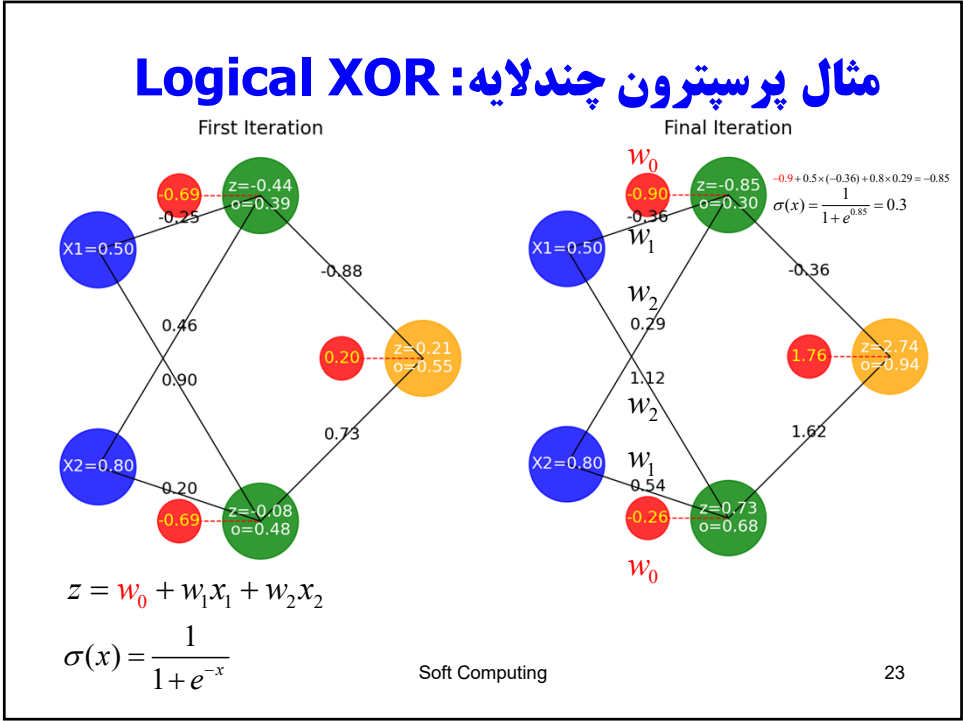
linearly inseparable

This means that a single threshold neuron cannot realize the XOR and XNOR function.

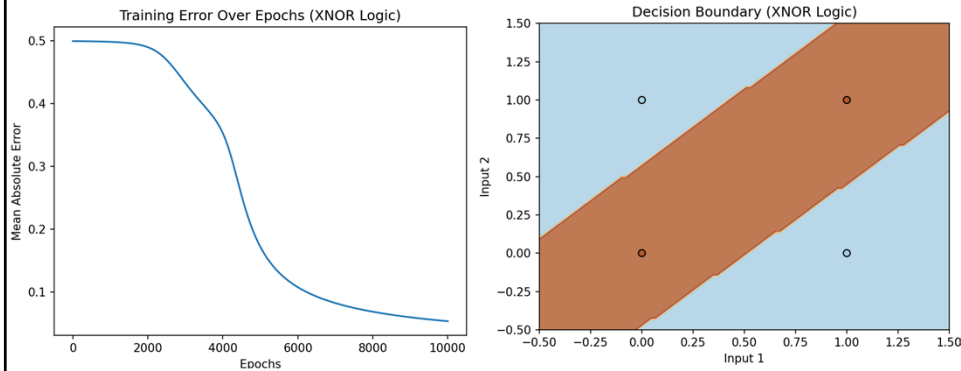
Soft Computing







## مثال پرسپترون چند لایه : Logical XNOR



پرسپترون چند لایه، شبکه عصبی یا شبکه ژرف نیز نامیده می شود

Soft Computing

25

## شبکه عصبی - ویژگی های تابع فعال سازی

### ویژگی های مورد نیاز

• پیوسته

• مشتق پذیر

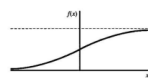
• دارا بودن کارایی محاسباتی (به راحتی قابل محاسبه باشد)  
• مشتق تابع را بتوان برحسب مقدار خود تابع نوشت

• به صورت یکنوا غیر نزولی

• قابلیت اشباع (Saturate)

• به صورت مجانبی به مقادیر بیشینه و کمینه خود نزدیک شود

• سیگموئید دودویی

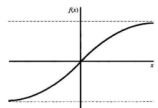


$$f_1(x) = \frac{1}{1 + \exp(-x)}$$

$$f_1'(x) = f_1(x)[1 - f_1(x)]$$

• سیگموئید دوقطبی

• شباهت به تانژانت هیپربولیک



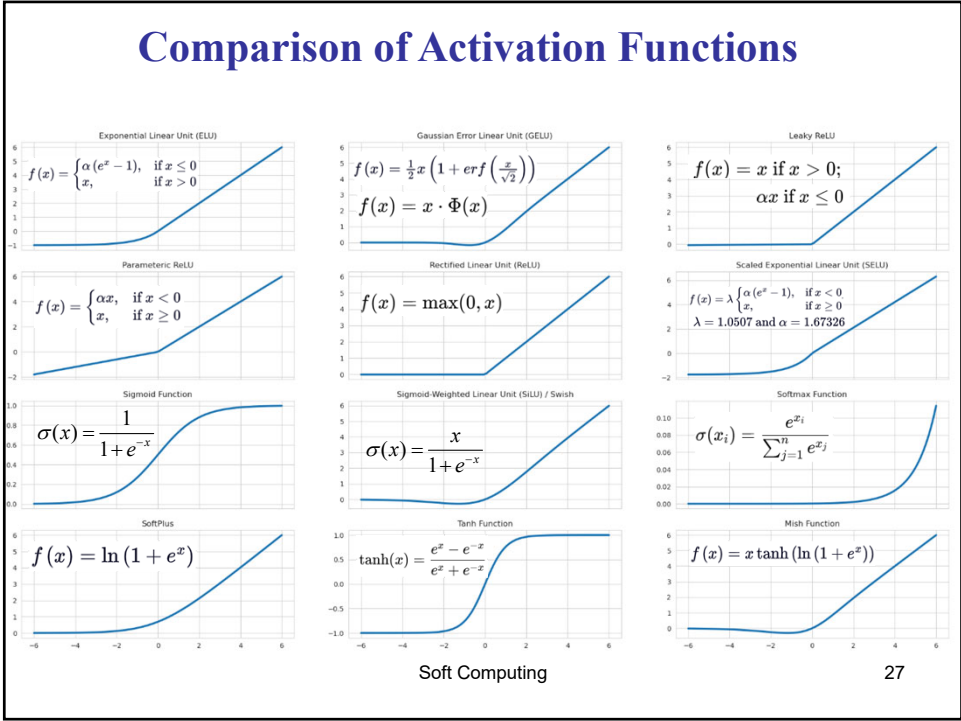
$$f_2(x) = \frac{2}{1 + \exp(-x)} - 1$$

$$f_2'(x) = \frac{1}{1 + \exp(-x)}$$

$$f_1(x) = \frac{1}{1 + \exp(-x)}$$

با کشف الگوریتم پس-انتشار توسط رملها، هینتن و ویلیامز در سال ۱۹۸۶ مطالعات جدید بر روی شبکه های عصبی مجدداً شروع شد. اهمیت ویژه این الگوریتم این بود که شبکه های عصبی چند لایه توسط آن می توانستند آموزش داده شوند.

Soft Computing

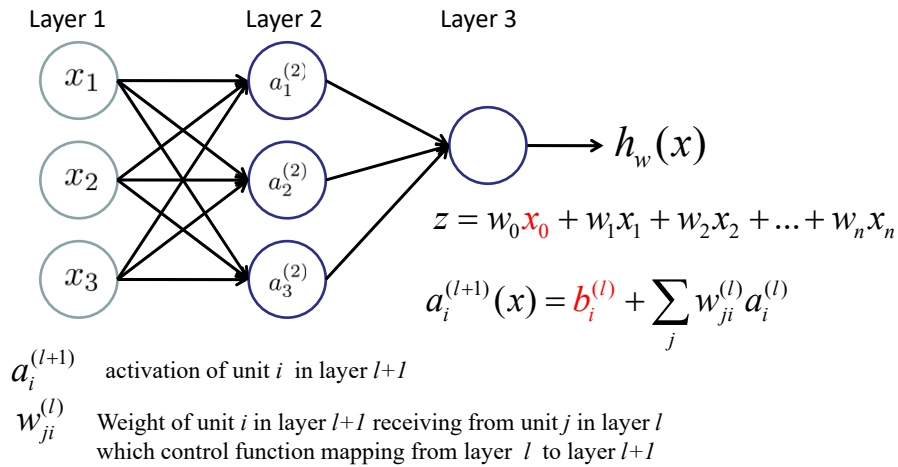


| Comparison of Activation Functions                                                                    |                     |                                 |                                      |                                           |
|-------------------------------------------------------------------------------------------------------|---------------------|---------------------------------|--------------------------------------|-------------------------------------------|
| Activation Function                                                                                   | Range               | Use Case                        | Advantages                           | Limitations                               |
| $\sigma(x) = \frac{1}{1 + e^{-x}}$ Sigmoid                                                            | (0, 1)              | Binary classification           | Smooth, probabilistic output         | Vanishing gradient, not zero-centered     |
| $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ Tanh                                                   | (-1, 1)             | Hidden layers                   | Zero-centered output                 | Vanishing gradient                        |
| $f(x) = \max(0, x)$ ReLU (Rectified Linear Unit)                                                      | $[0, \infty)$       | Hidden layers                   | Efficient, avoids vanishing gradient | Dying neurons                             |
| $f(x) = \begin{cases} x & \text{if } x > 0; \\ \alpha x & \text{if } x \leq 0 \end{cases}$ Leaky ReLU | $(-\infty, \infty)$ | Hidden layers                   | Avoids dying neurons                 | Arbitrary slope choice                    |
| $\sigma(x_i) = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$ Softmax                                          | (0, 1)              | Output for multi-class problems | Probabilistic interpretation         | Expensive for large datasets              |
| $f(x) = x \cdot \sigma(x)$ Swish                                                                      | $(-\infty, \infty)$ | Deep networks                   | Smooth and flexible                  | Slightly more computationally intensive   |
| $f(x) = x \cdot \Phi(x)$ GELU                                                                         | $(-\infty, \infty)$ | Transformer-based architectures | Smooth and efficient                 | Relatively new, computationally expensive |

Soft Computing 28

## شبکه عصبی مصنوعی

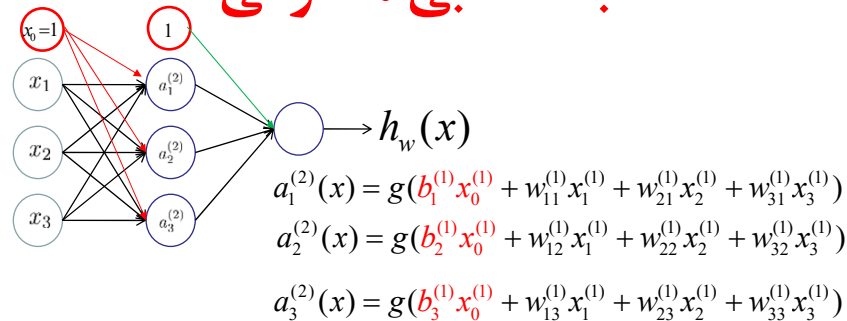
افزایش تعداد لایه‌های برای پردازش مسایل پیچیده‌تر



Soft Computing

29

## شبکه عصبی مصنوعی



$$h_w(x) = a_1^{(3)}(x) = g(b_1^{(2)} a_0^{(2)} + w_{11}^{(2)} a_1^{(2)} + w_{21}^{(2)} a_2^{(2)} + w_{31}^{(2)} a_3^{(2)})$$

Learnable parameter for  $l$  layer

$$\sum_{l=1}^{L-1} (N_L + 1)(N_{L+1}) = (N_1 + 1)N_2 + (N_2 + 1)N_3 + \dots + (N_{L-1} + 1)N_L$$

$N_l$  Number of neuron in  $l$  layer

Soft Computing

30

## شبکه عصبی مصنوعی

pip install tensorflow

pip install intel-tensorflow

Use Intel-optimized TensorFlow

```
import os
import numpy as np
import random
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, ConfusionMatrixDisplay

# Optional: Disable oneDNN custom operations to avoid floating-point warnings
os.environ["TF_ENABLE_ONEDNN_OPTS"] = "0"
os.environ["TF_CPP_MIN_LOG_LEVEL"] = '2' #Suppress TensorFlow INFO and WARNING logs

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Input

# Set random seeds for reproducibility
np.random.seed(42)
tf.random.set_seed(42)
random.seed(42)
```

13 ANN.py

Soft Computing

31

## شبکه عصبی مصنوعی

```
# Generate a 2D dataset for visualization
X, y = make_classification(
    n_samples=200, n_features=2, n_informative=2, n_redundant=0,
    n_clusters_per_class=1, random_state=42
)

# Standardize the dataset
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.3, random_state=42)

# Build the ANN model
def build_model(input_dim, output_dim):
    """Build a simple feedforward ANN model with two hidden layers.
    """
    model = Sequential([
        Input(shape=(input_dim,)), # Explicit Input layer
        Dense(4, activation='relu'), # 4 neuron First hidden layer
        Dense(2, activation='relu'), # 2 neuron Second hidden layer
        Dense(output_dim, activation='sigmoid') # Output layer for
        binary classification
    ])
    return model
```

Soft Computing

32

## شبکه عصبی مصنوعی

```
# Train, evaluate, and plot ANN
def train_evaluate_ann(epochs=50, batch_size=8):
    """Train the ANN, evaluate on test data, and visualize results.
    print("Training Artificial Neural Network...")

    # Build and compile the model
    model = build_model(input_dim=2, output_dim=1)
    model.compile(optimizer='adam', loss='binary_crossentropy',
    metrics=['accuracy'])

    # Train the model
    history = model.fit(X_train, y_train, epochs=epochs,
    batch_size=batch_size, verbose=0)

    # Evaluate on test set
    y_pred = (model.predict(X_test) > 0.5).astype(int)
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Test Accuracy: {accuracy:.2f}")

    # Plot decision boundary
    plot_ann_with_boundary(X_train, y_train, model, accuracy)

# Run the ANN training and visualization
if __name__ == "__main__":
    try:
        train_evaluate_ann(epochs=50, batch_size=8)
    except Exception as e:
        print(f"An error occurred: {e}")

# Predict a new example
predict_new_example(model, scaler, target_names)
```

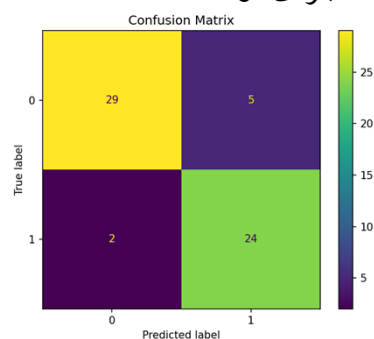
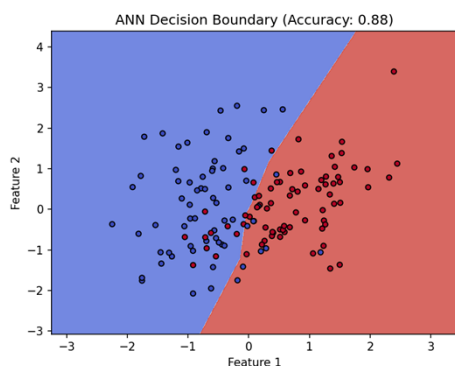
33

Soft Computing

## مثال

حل مساله به روش شبکه عصبی برای مثال دسته بندی در قسمت SVM

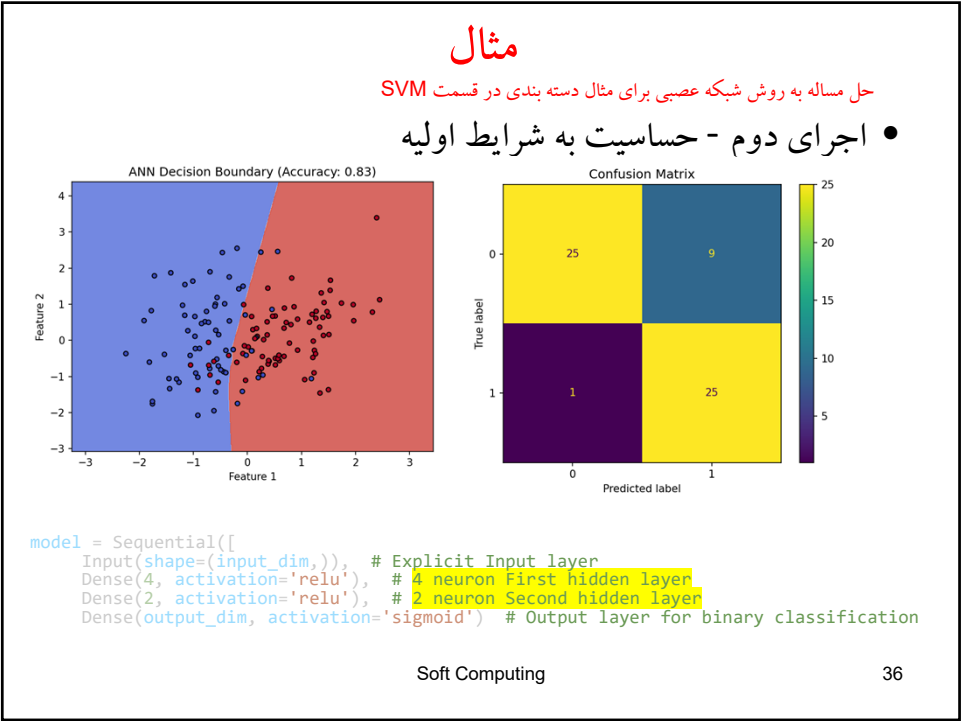
### • اجرای اول

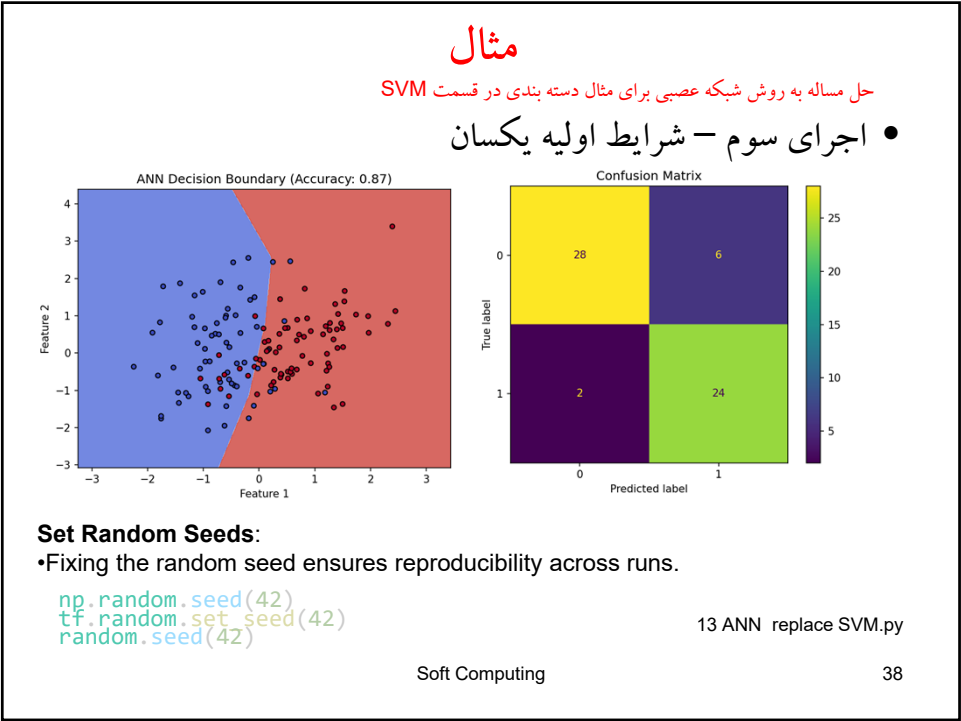
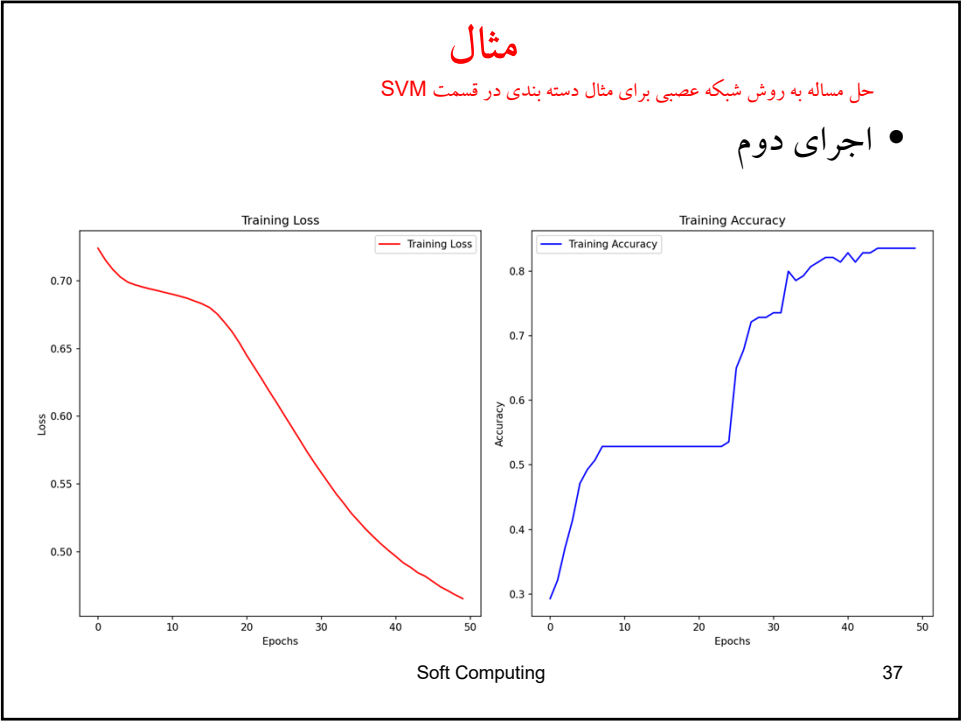


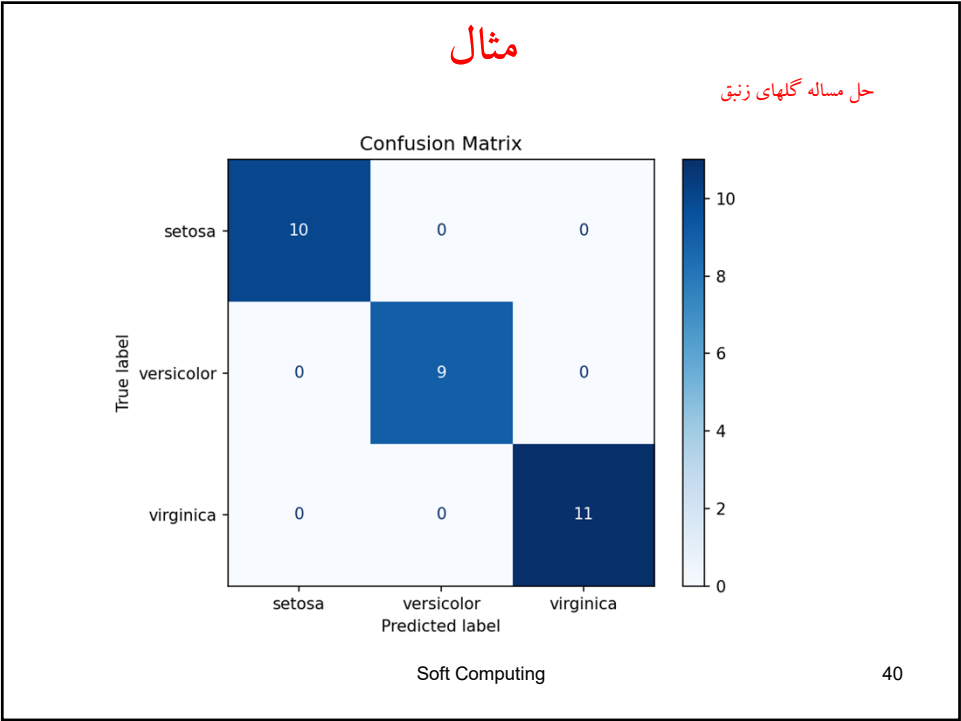
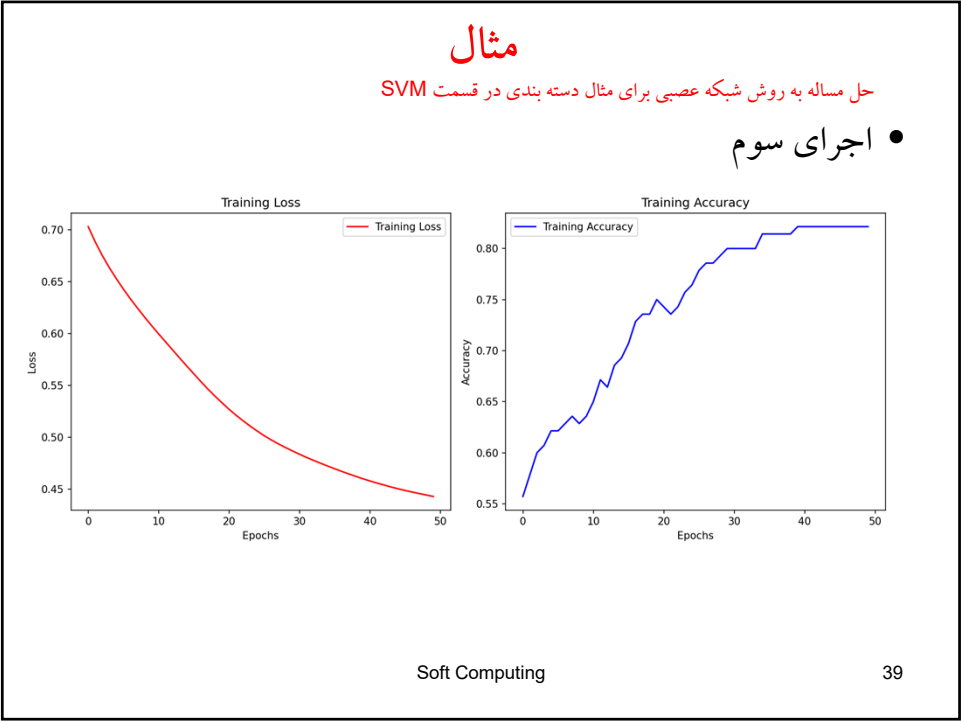
```
model = Sequential([
    Input(shape=(input_dim,)), # Explicit Input layer
    Dense(4, activation='relu'), # 4 neuron First hidden layer
    Dense(2, activation='relu'), # 2 neuron Second hidden layer
    Dense(output_dim, activation='sigmoid') # Output layer for binary classification
```

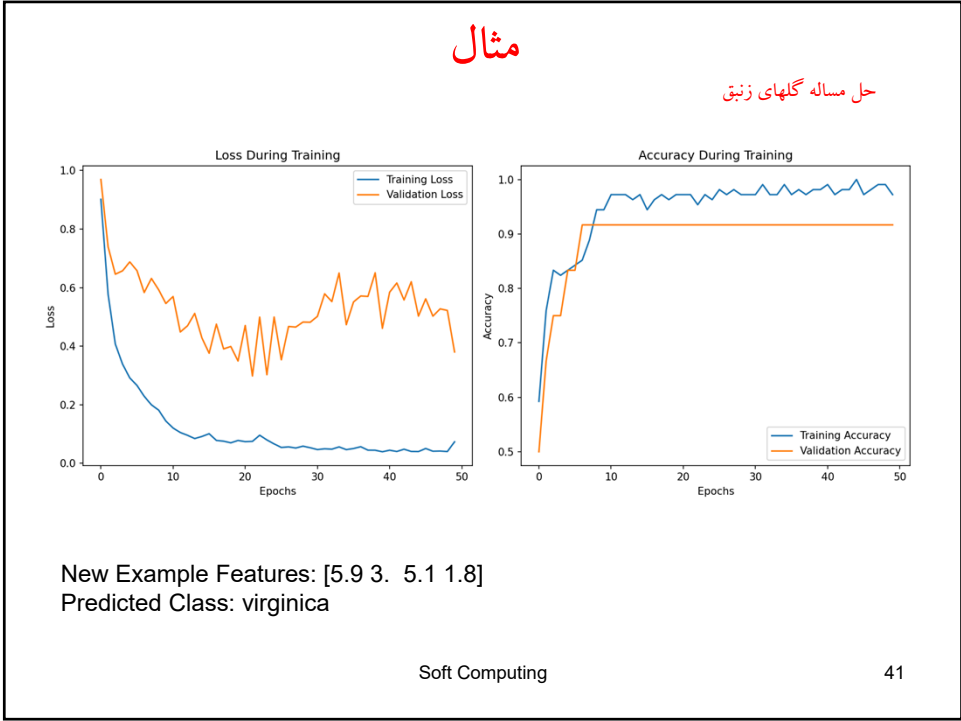
Soft Computing

34









معماری شبکه هوش مصنوعی

•First hidden layer, the number of neurons is often:  $n > n_{input\ features} + n_{output\ targets}$

•Subsequent layers, try reducing the number of neurons progressively (e.g.,  $64 \rightarrow 32 \rightarrow 16$ ).

Example

1. Binary Classification

•Input Features: 10  
•Output Classes: 2

•Suggested Architecture:

•Input Layer: 10 neurons (input size).  
•Hidden Layer 1: 16 neurons (capture interactions).  
•Hidden Layer 2: 8 neurons (reduce complexity).  
•Output Layer: 2 neuron (sigmoid activation).

2. Multiclass Classification

•Input Features: 20  
•Output Classes: 5

•Suggested Architecture:

•Input Layer: 20 neurons.  
•Hidden Layer 1: 32 neurons (capture higher-level features).  
•Hidden Layer 2: 16 neurons.  
•Output Layer: 5 neurons (softmax activation).

3. Regression

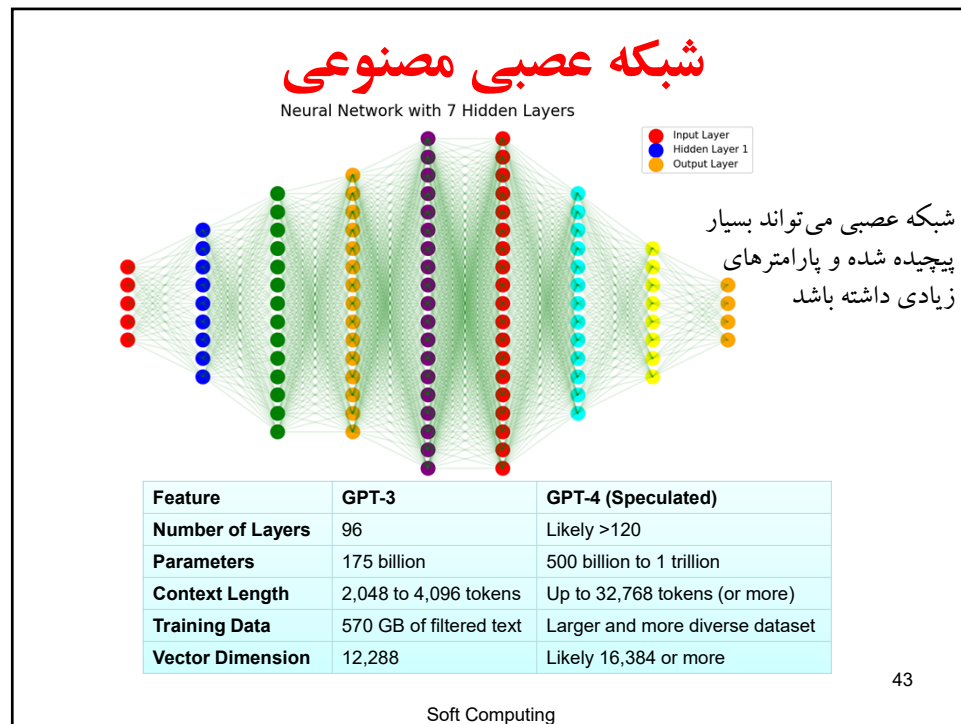
•Input Features: 5  
•Output: Continuous value

•Suggested Architecture:

•Input Layer: 5 neurons.  
•Hidden Layer 1: 10 neurons (nonlinear mapping).  
•Output Layer: 1 neuron (linear activation).

Use learning curves to analyze underfitting (high training loss) or overfitting (low training loss, high validation loss)

Soft Computing42



## شبکه عصبی مصنوعی کانولوشنی

### CNN: Convolutional Neural Networks

نوعی از شبکه های عصبی مصنوعی است که می تواند داده های تصویری را به صورت خودکار تحلیل و یاد بگیرد. برخلاف شبکه های عصبی سنتی که نیاز به پیش پردازش دستی ویژگی ها دارند، CNN ویژگی های مهم تصویر را به صورت خودکار استخراج می کند

توضیح

فیلترهایی (Kernel) را روی تصویر حرکت می دهد تا ویژگی هایی مثل لبه، گوشه یا بافت را شناسایی کند.

عملیات غیرخطی (مثل  $\text{ReLU: max}(0, x)$ ) برای افزایش توانایی مدل در یادگیری روابط پیچیده.

اندازه تصویر را کاهش می دهد و به مدل کمک می کند تا نسبت به جابه جایی ها مقاوم شود.

پس از استخراج ویژگی ها، تصمیم گیری نهایی (مثلاً طبقه بندی) را انجام می دهد.

خروجی نهایی برای دسته بندی (مثلاً احتمال تعلق تصویر به هر کلاس).

Soft Computing

مؤلفه

لایه کانولوشن  
Convolution Layer

لایه فعال سازی  
ReLU

لایه زیر نمونه گیری  
Pooling

لایه اتصال کامل  
Fully Connected

Softmax

44

## تمرین برنامه نویسی

تمرین: یک برنامه به زبان پایتون بنویسید که یک فایل داده را خوانده و به روش شبکه عصبی مصنوعی موارد زیر را انجام دهد.

۱- تعریف یک مدل MLP

۲- تعیین و تنظیم تعداد لایه‌های پنهان و نرونها

۳- تعیین توابع فعالسازی مناسب

Data set: "Concrete Compressive Strength"

Input.: 8 Features

Output: Concrete strength

Sample No.: 1030

۴- پیش بینی مناسب مقاومت فشاری بتن

۵- تعیین میزان خطا

**PyTorch**: کتابخانه متن‌باز برای یادگیری عمیق که در سال ۲۰۱۶ توسط فیسبوک توسعه داده شده است. علاوه بر سادگی و پشتیبانی قوی در انجمن‌ها و شبکه‌های اجتماعی دارای ویژگیهای زیر است:

- کتابخانه‌های غنی. مثل torchvision (کار با تصاویر)، torchttext (پردازش متن) و torchaudio (کار با صدا).
- محاسبات گراف‌های پویا (قابلیت تعریف و تغییر مدل و گراف محاسباتی حین اجرا)
- سازگاری با سایر کتابخانه‌ها نظیر Numpy و Scikit-Learn
- پشتیبانی از کارت‌های گرافیک

Soft Computing

45