Session 33

Classification II

DigitalSkola
Uncover The World Of Digital Skills With Us

Table of Content

Apa yang Akan Kita Pelajari Hari Ini?

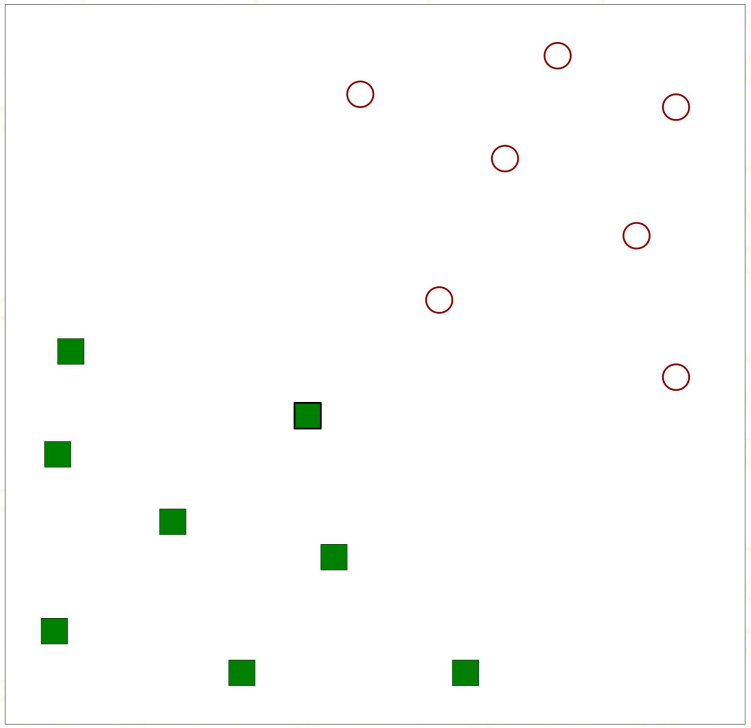
1. SVM
2. Neural Network
3. Multiclass classification

DigitalSkola
Uncover The World Of Digital Skills With Us

SVM

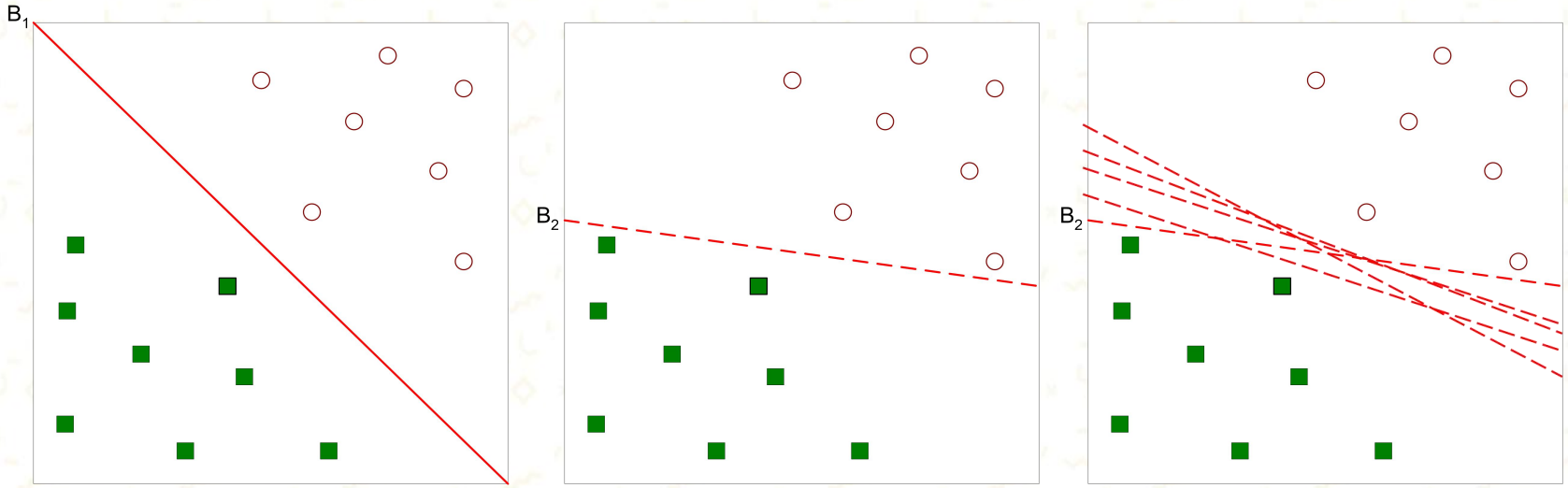
DigitalSkola
Uncover The World Of Digital Skills With Us

Support Vector Machines



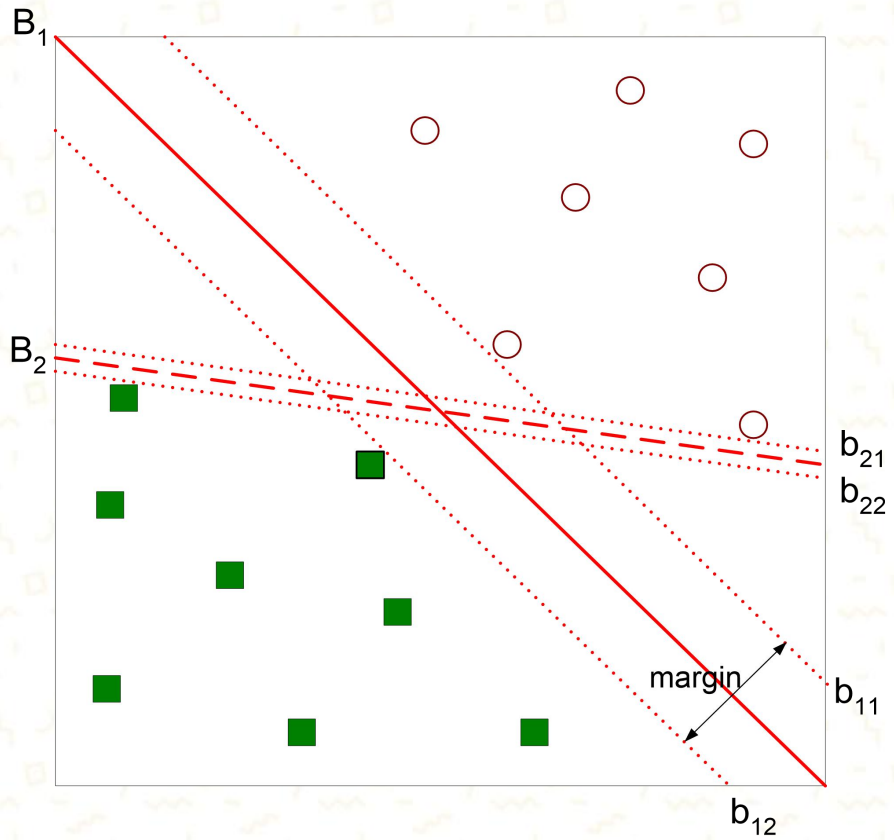
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



- There are many possible solution

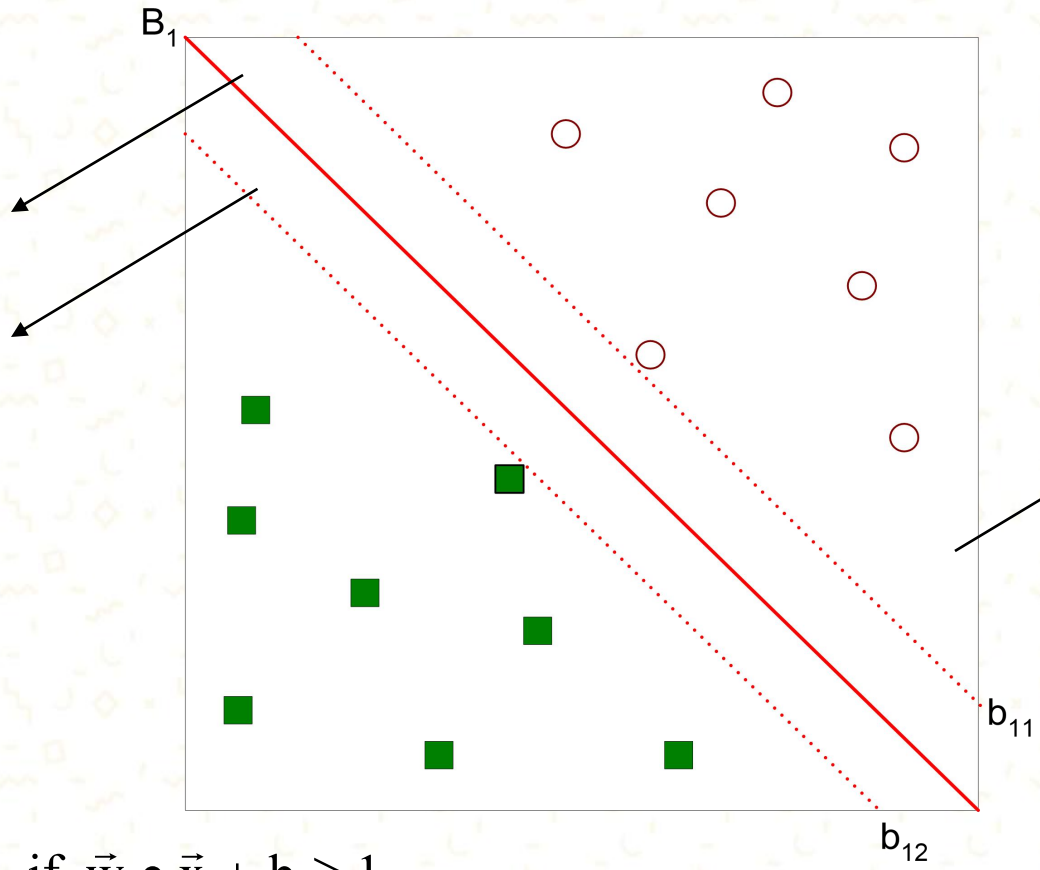
Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

DigitalSkola
Uncover The World Of Digital Skills With Us

Support Vector Machines



$\vec{w} \bullet \vec{x} + b = 0$

$\vec{w} \bullet \vec{x} + b = -1$

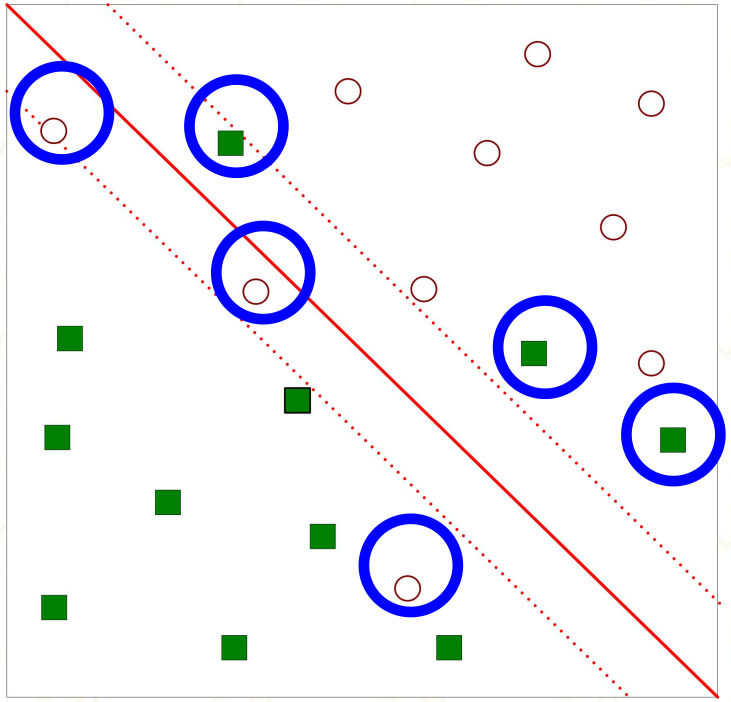
$\vec{w} \bullet \vec{x} + b = +1$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

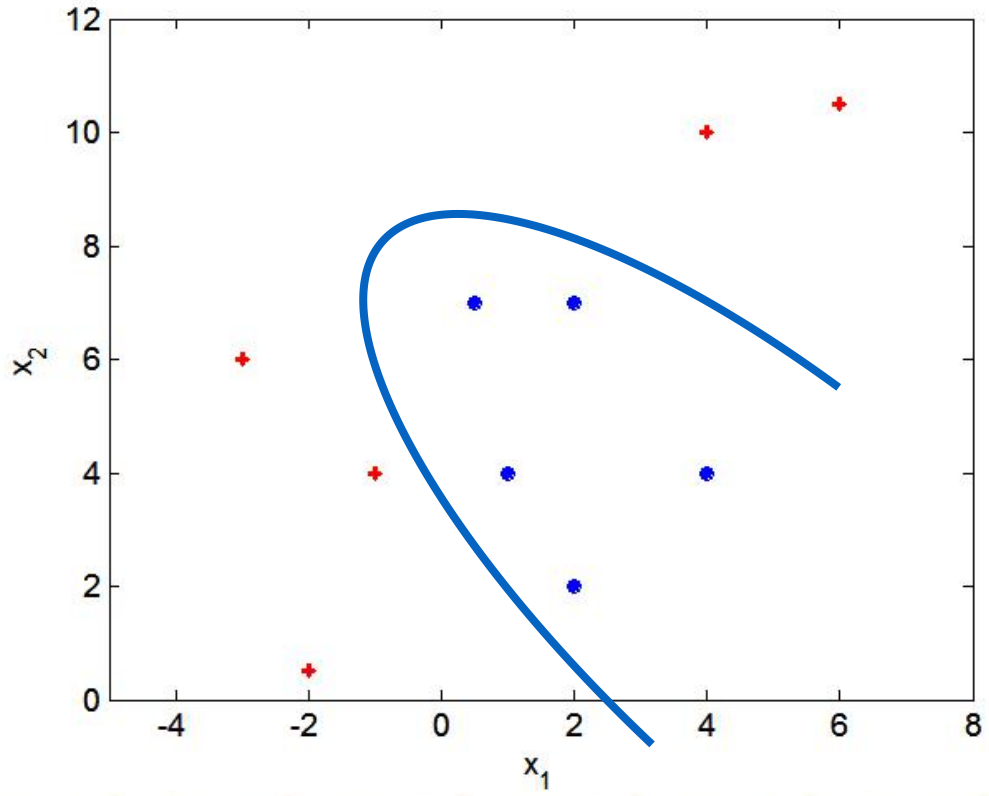
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

    - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$

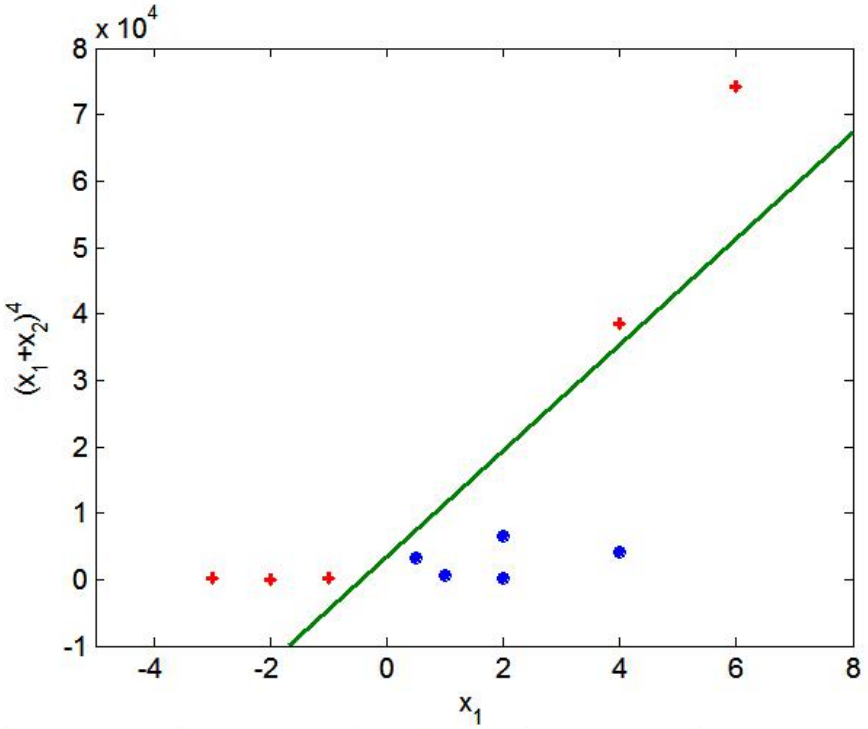
Nonlinear Support Vector Machines

- What if decision boundary is not linear?



Nonlinear Support Vector Machines

- Transform data into higher dimensional space



DigitalSkola
Uncover The World Of Digital Skills With Us

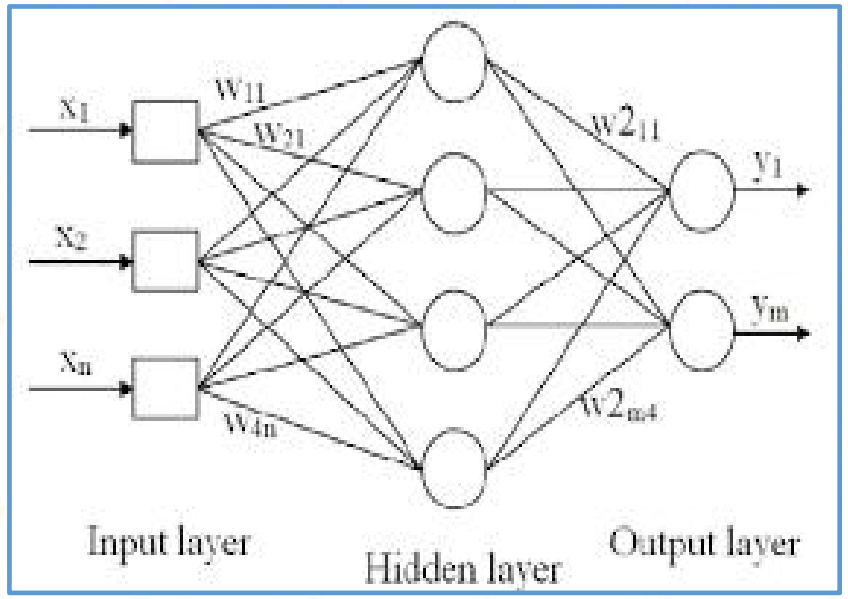
Neural Network

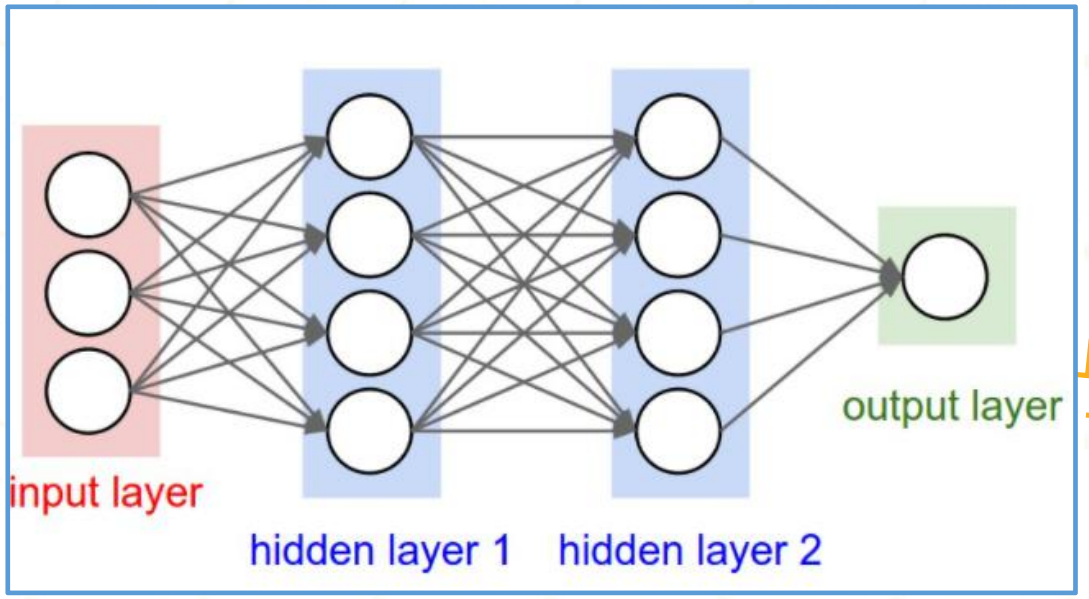
Brief History

- **History of ANN**
- 1943: McCulloch and Pitts: First mathematic model of neuron
- 1958: Rosenblatt: Perceptron - Single layer NN
- 1986: Rumelhart: Back Propagation algorithm
- 1995: Y. LeCun, Y. Bengio, et al.: Convolutional neural network
- 2006: G. E. Hinton, et al.: Deep belief nets.

Artificial Neural Network

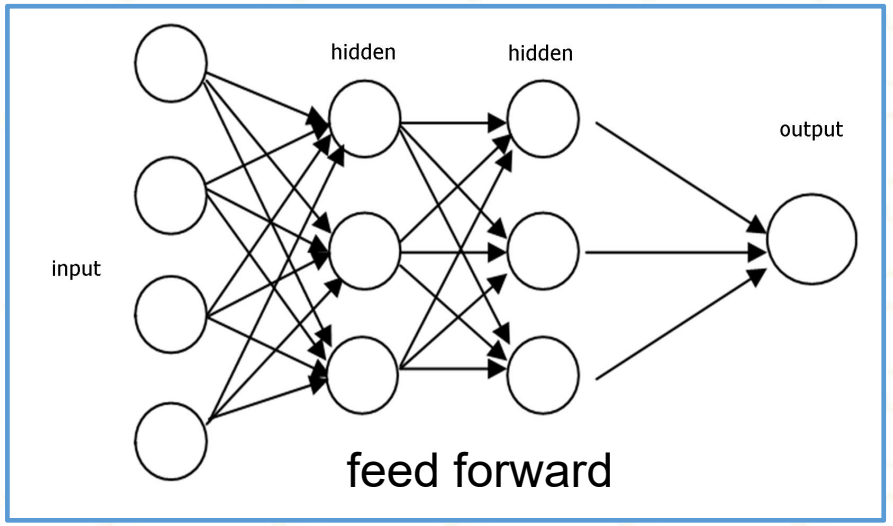
- Artificial Neural Network (Jaringan saraf tiruan) terdiri dari kumpulan unit pemrosesan sederhana yang berkomunikasi dengan mengirimkan sinyal satu sama lain melalui sejumlah besar koneksi berbobot.
- Model yang terinspirasi oleh bagaimana neuron dalam otak manusia bekerja

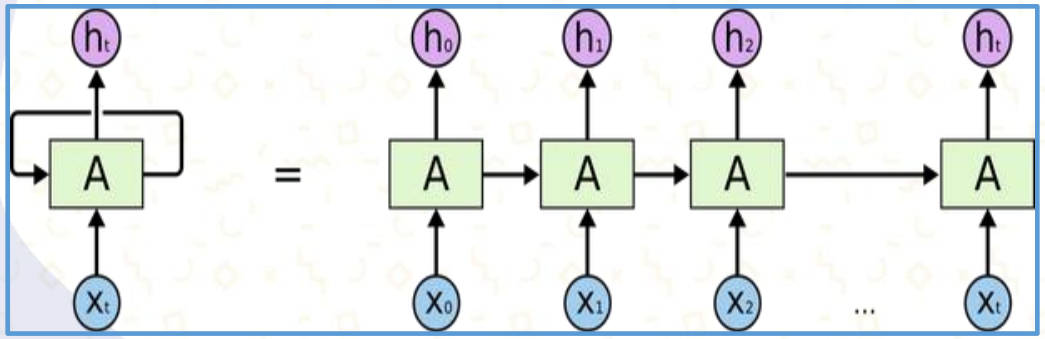




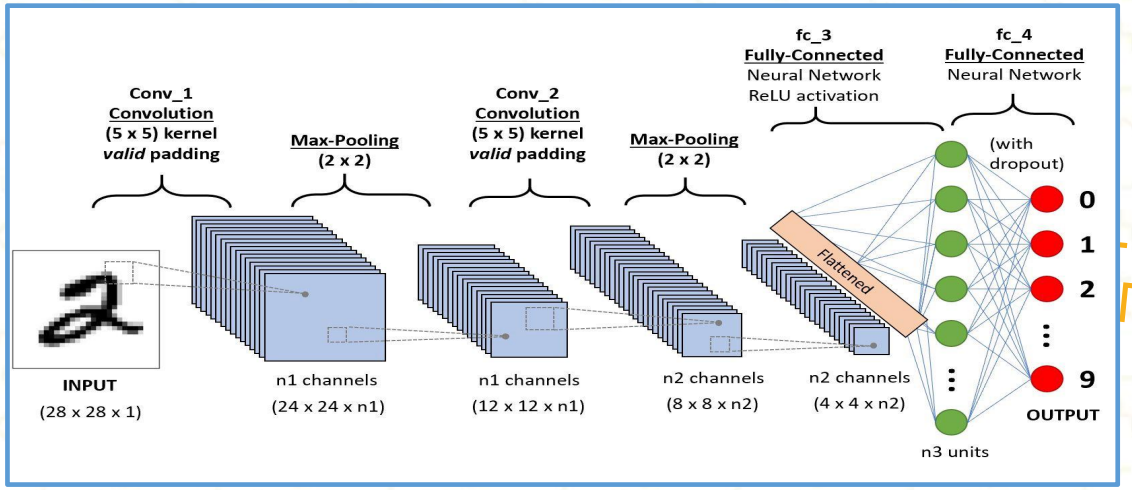
Types of ANN

- **Types of ANN**
  - Feedforward neural network
  - Recurrent neural network (RNN)
  - Convolutional neural network (CNN)
  - etc,





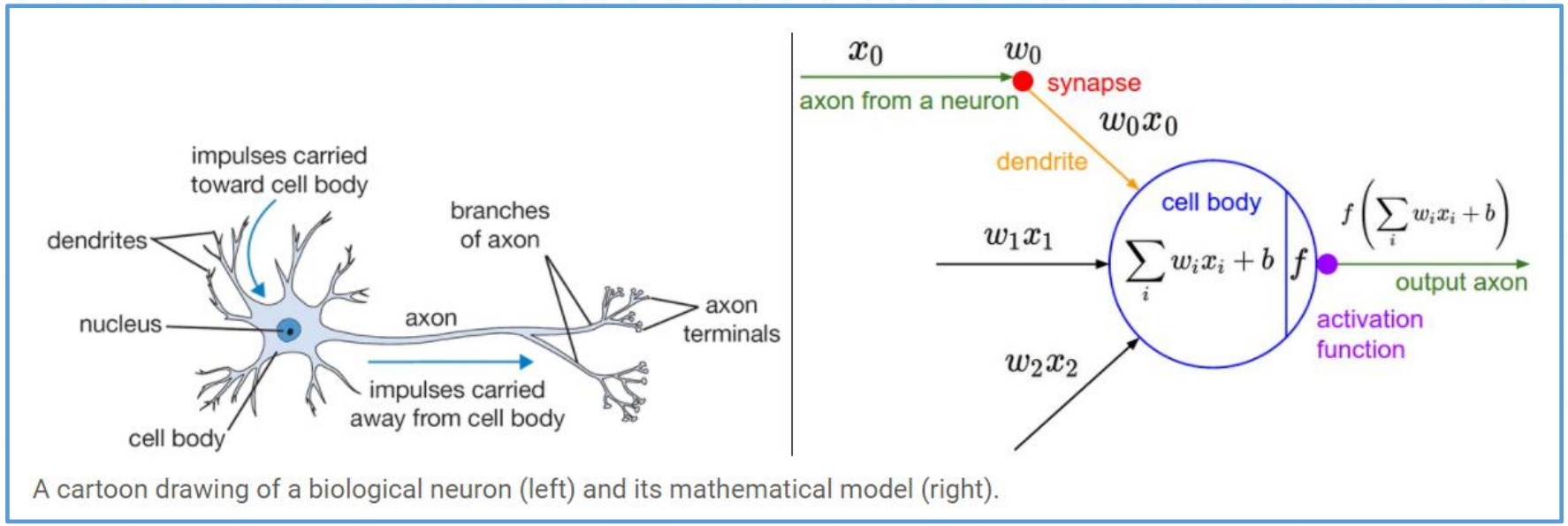
RNN



CNN

Biological motivation

- Neural Networks awalnya terinspirasi oleh tujuan pemodelan sistem saraf biologis.
- Unit komputasi dasar otak adalah neuron.
- Sekitar 86 miliar neuron dapat ditemukan di sistem saraf manusia

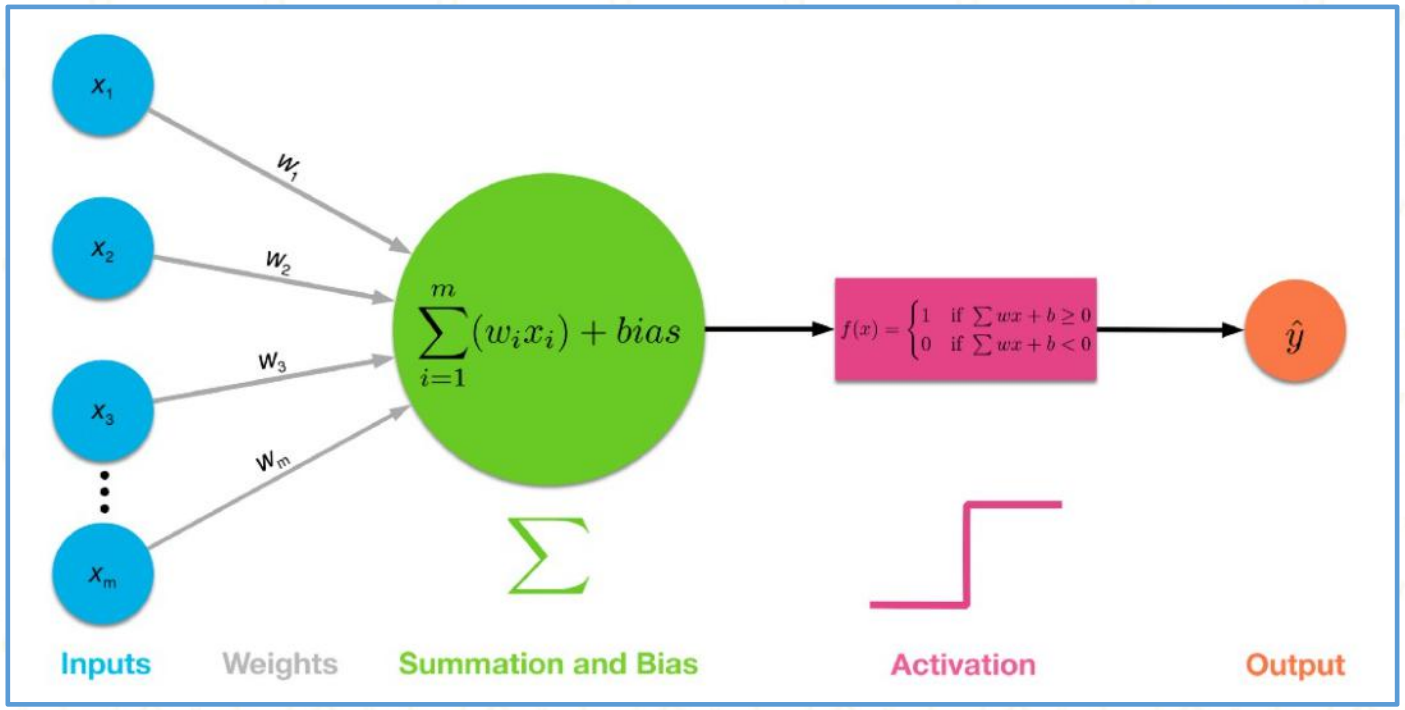


A cartoon drawing of a biological neuron (left) and its mathematical model (right).

https://cs231n.github.io/neural-networks-1/

Perceptron

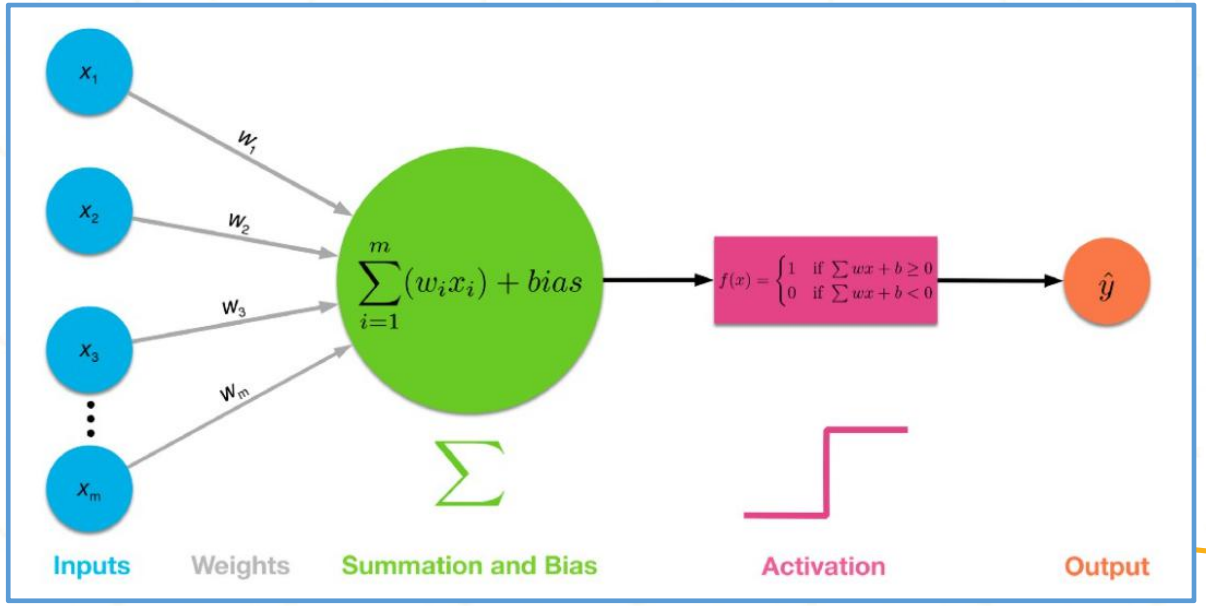
- Perceptron adalah pengklasifikasi linier (linear classifier).
- Frank Rosenblatt, seorang psikolog Amerika, mengusulkan model perceptron klasik pada tahun 1958



https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

Perceptron

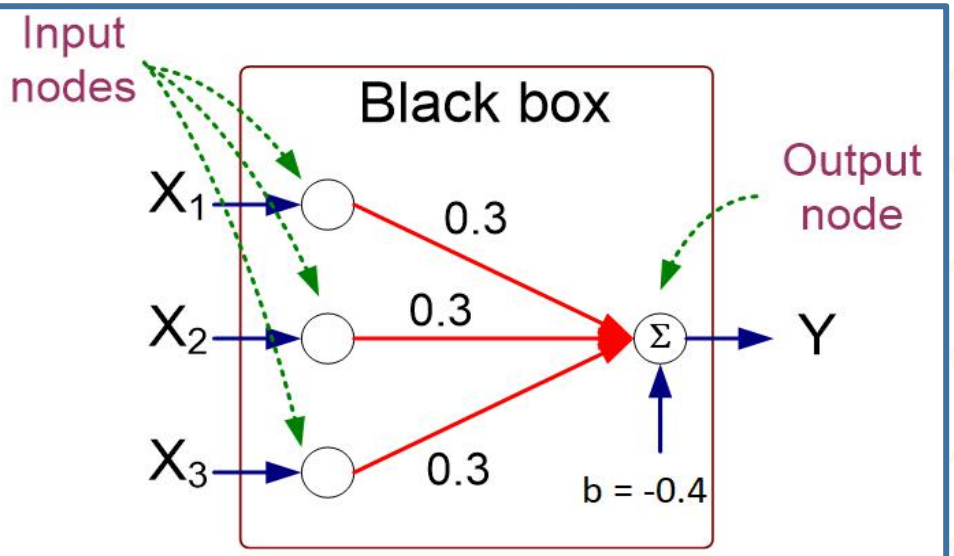
- Digunakan untuk mengklasifikasikan data input yang diberikan.
- The perceptron terdiri dari 4 bagian
  - Input values atau one input layer
  - Weights dan Bias
  - Net sum
  - Activation Function



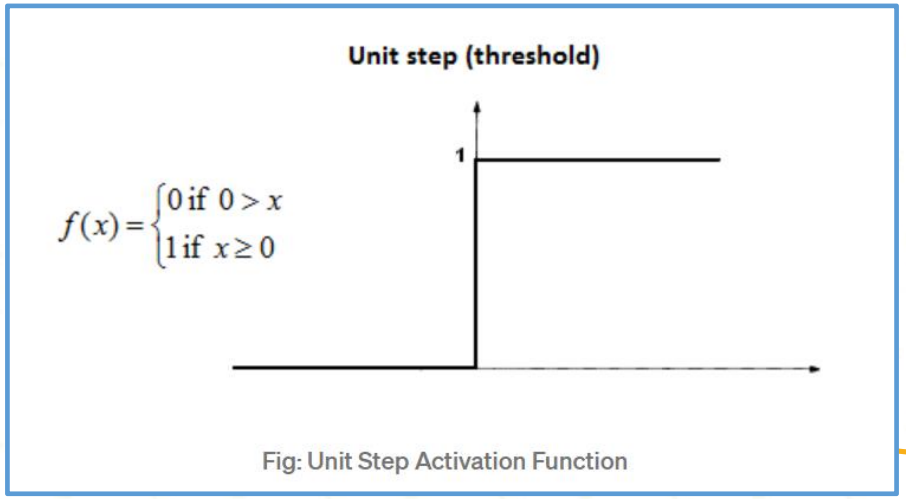
Perceptron (how does it work?)

- Semua inputs x dikalikan dengan weights w
- Jumlahkan semua hasil perkalikan tersebut (kita sebut *Weighted Sum*)
- Mengaplikasikan activation function ke weighted sum

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

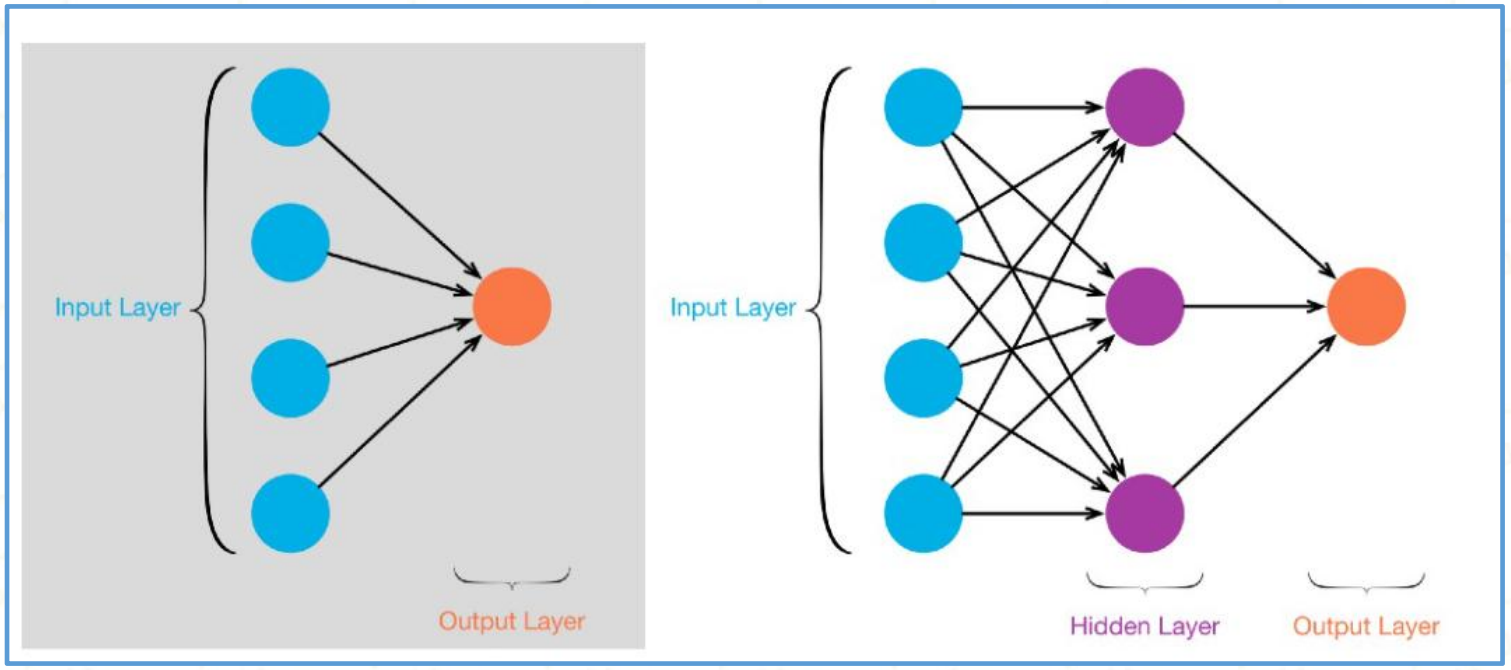


y = f(0.3*x1 + 0.3*x2 + 0.3*x3 - 0.4)

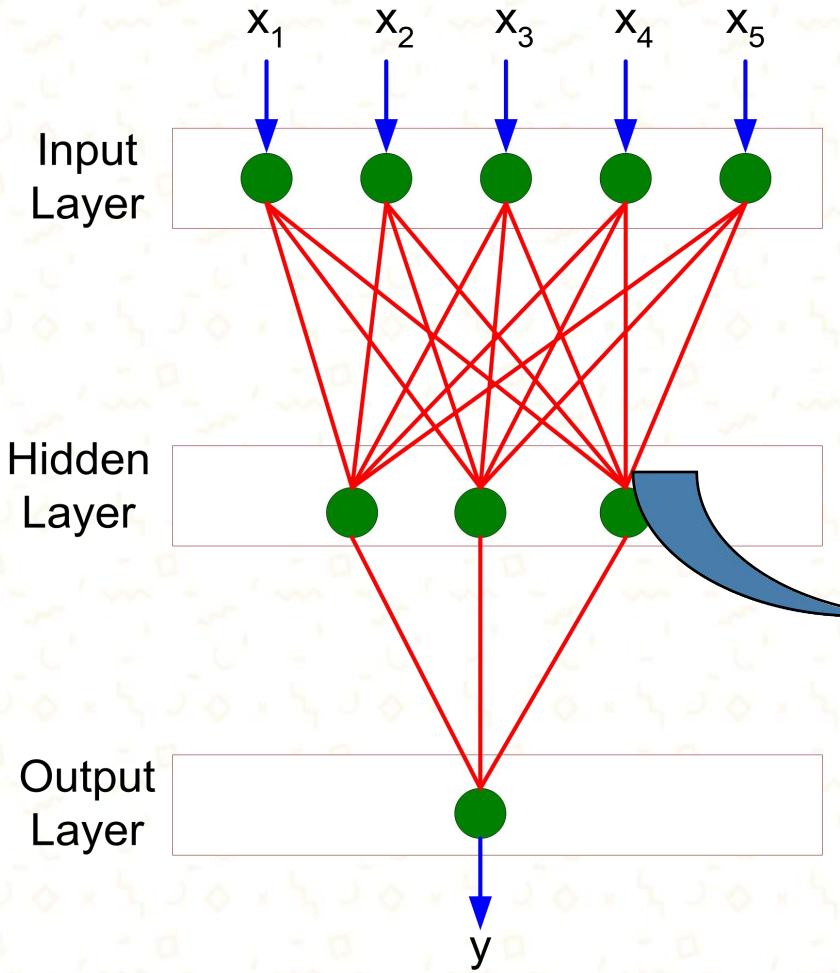


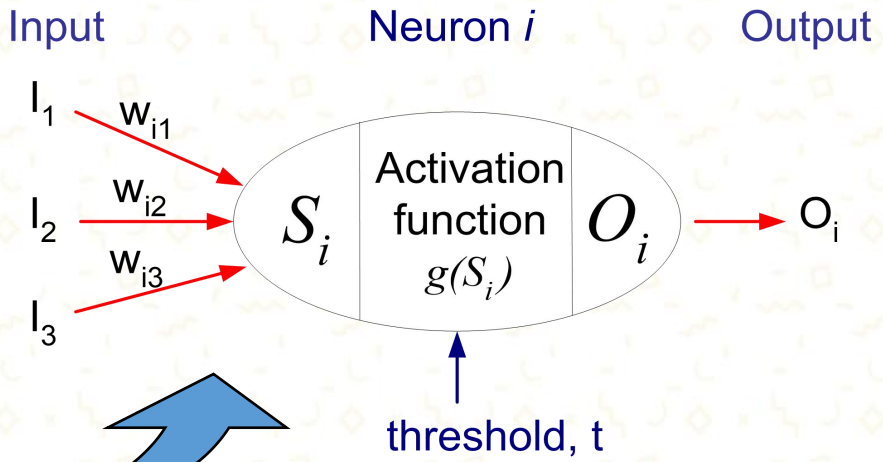
Multilayer perceptron

- Model ini terdiri dari tiga jenis layer — input layer, hidden layer, output layer.
- Kecuali node input, setiap node adalah neuron yang menggunakan fungsi aktivasi nonlinier.
- MLP menggunakan backpropagation untuk training-nya.



General Structure of MLP



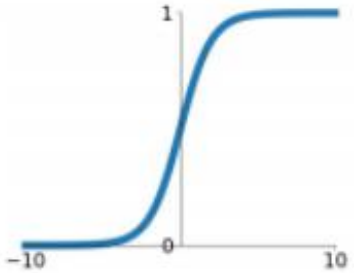


Training ANN means learning the weights of the neurons

Activation function

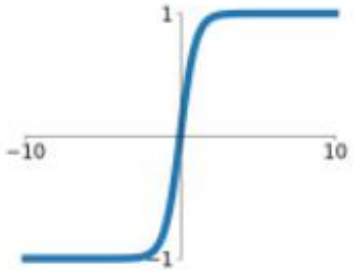
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



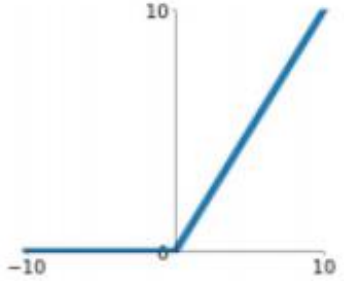
tanh

$$\tanh(x)$$

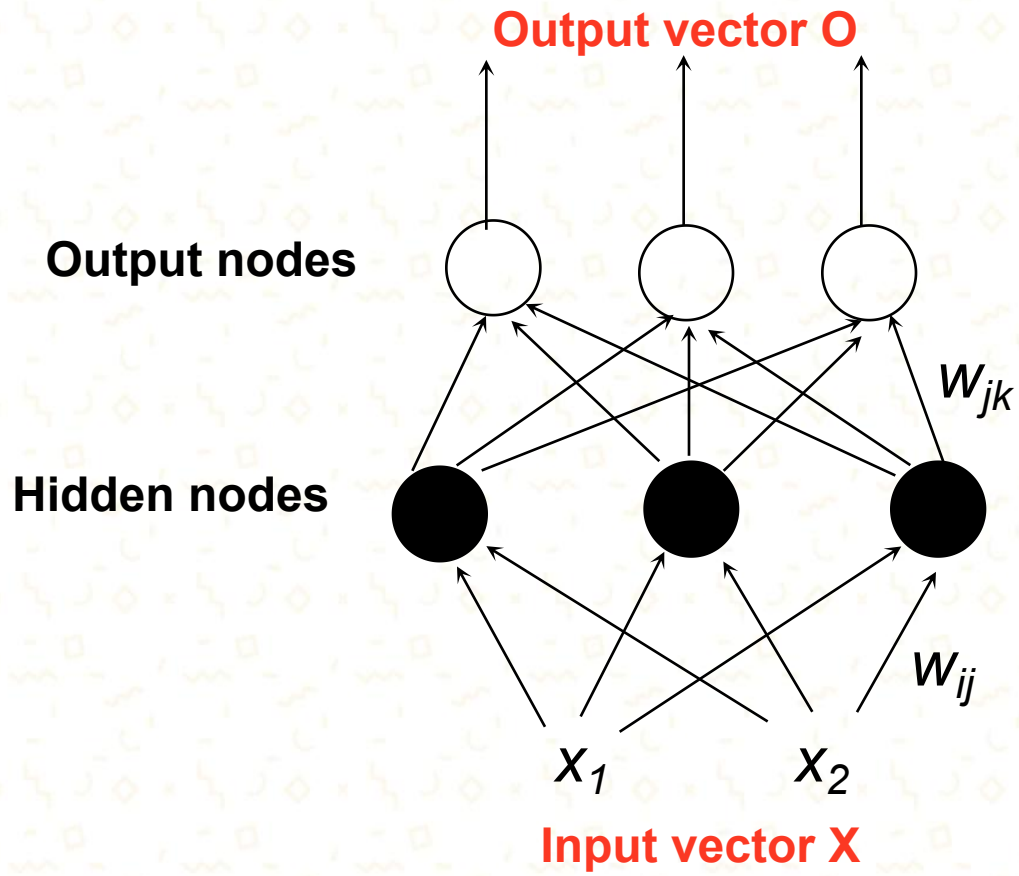


ReLU

$$\max(0, x)$$



Multilayer perceptron



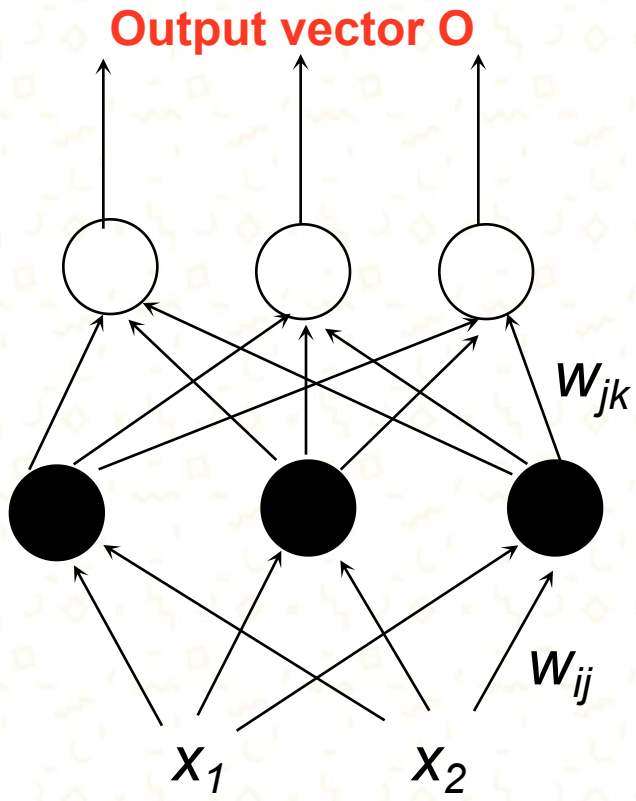
$$O_k = \frac{1}{1+e^{-\sum h_j w_{jk} + \theta_k}}$$

$$h_j = \frac{1}{1+e^{-\sum x_i w_{ij} + \theta_j}}$$

Network Training

- Tujuan dari training
  - Mendapatkan weights yang membuat hampir semua sampel dalam training data dapat diklasifikasikan dengan benar
- Langkah-langkah
  - Inisialisasi weights w_{ij} dengan nilai acak(random)
  - Masukkan (feed) sample training ***X*** ke dalam jaringan satu persatu
  - Untuk setiap unit
    - Hitung output value ***O*** dengan mengaplikasikan activation function
    - Hitung error ***E***
    - Update weights w_{ij} dan biases

Backpropagation Learning



$$Err_k = (T_k - O_k)$$

O_k

h_j

Given (X, T)

$$\delta_k = O_k(1-O_k)(T_k - O_k)$$

$$w_{jk} = w_{jk} + (l)\delta_k O_k$$

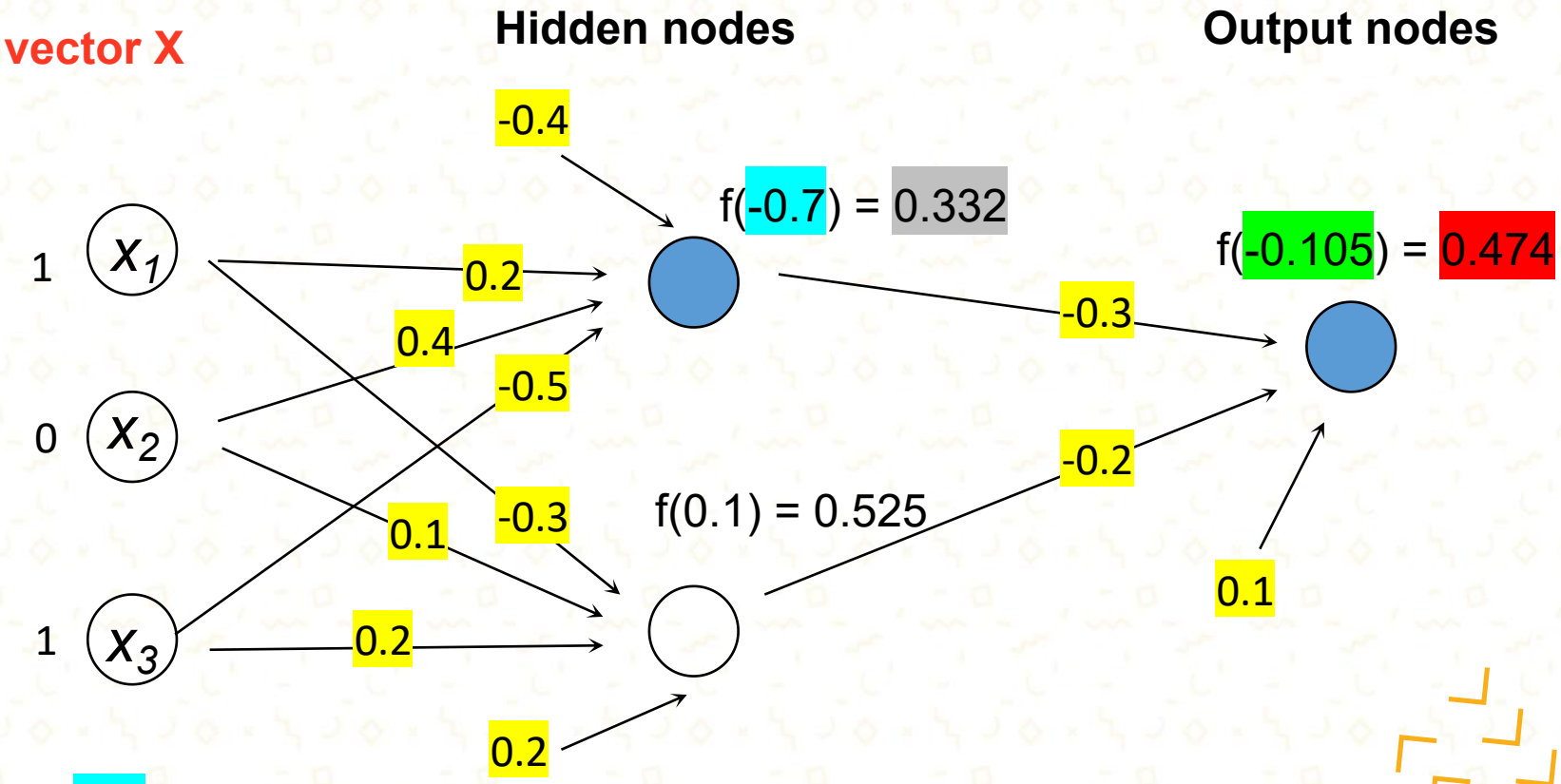
$$\delta_j = h_j(1-h_j)\sum_k \delta_k w_{jk}$$

$$w_{ij} = w_{ij} + (l)\delta_j h_j$$

Contoh

Input vector X

X_1	X_2	X_3	Y
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

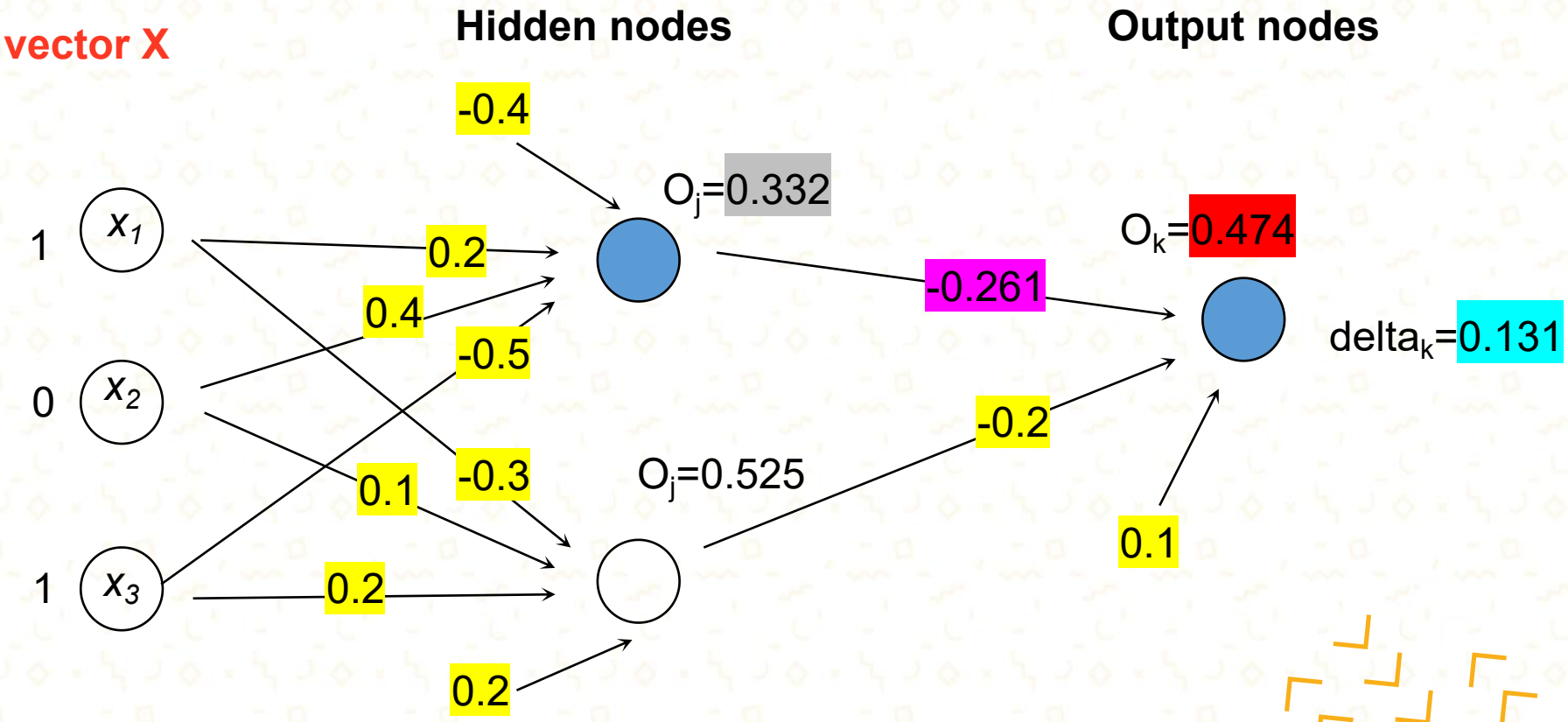


- Net Input I_j = 0.2 + 0 + -0.5 -0.4 = -0.7
- Output O_j = $(1/(1+e^{0.7})$=0.332

- Net Input I_k = -0.3(0.332) - (0.2)(0.525) + 0.1 = -0.105
- Output O_k = $(1/(1+e^{0.105})$ = 0.474

Contoh

X_1	X_2	X_3	Y
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



- $delta_k$=(0.474)(1-0.474)(1-0.474)=0.131
- w_{jk} = -0.3+(0.9)(0.1311)(0.332)=-0.261

- Dan seterusnya..

Discussion on NN

- Keuntungan
  - **Robust** -berfungsi baik ketika training set mengandung error
  - Output bisa discrete, real-valued, atau vector
- Criticism
  - Waktu yang lama saat training
  - Sulit untuk dipahami

MLP in Sklearn

- Sklearn menyediakan class MLPClassifier

```
class sklearn.neural_network.MLPClassifier(hidden_layer_sizes=(100), activation='relu', *, solver='adam', alpha=0.0001,
batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None,
tol=0.0001, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False,
validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10, max_fun=15000)   [source]
```

Hyperparameter	Description
hidden_layer_sizes	Jumlah hidden layer dan node-nya
activation	Activation function
max_iter	Jumlah iterasi

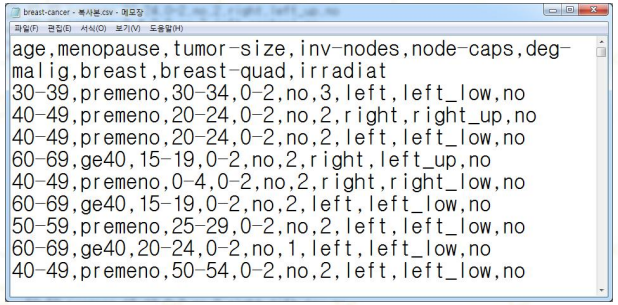
DigitalSkola
Uncover The World Of Digital Skills With Us

Multiclass

Basic Classification in ML

Input $\mathbf{x} \in \mathcal{X}$	Output $y \in \mathcal{Y}$

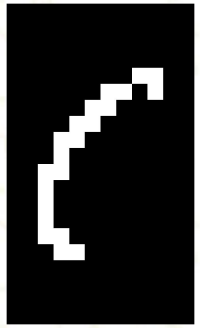
Cancer detection



Binary

- Cancer
- Not cancer

Character recognition

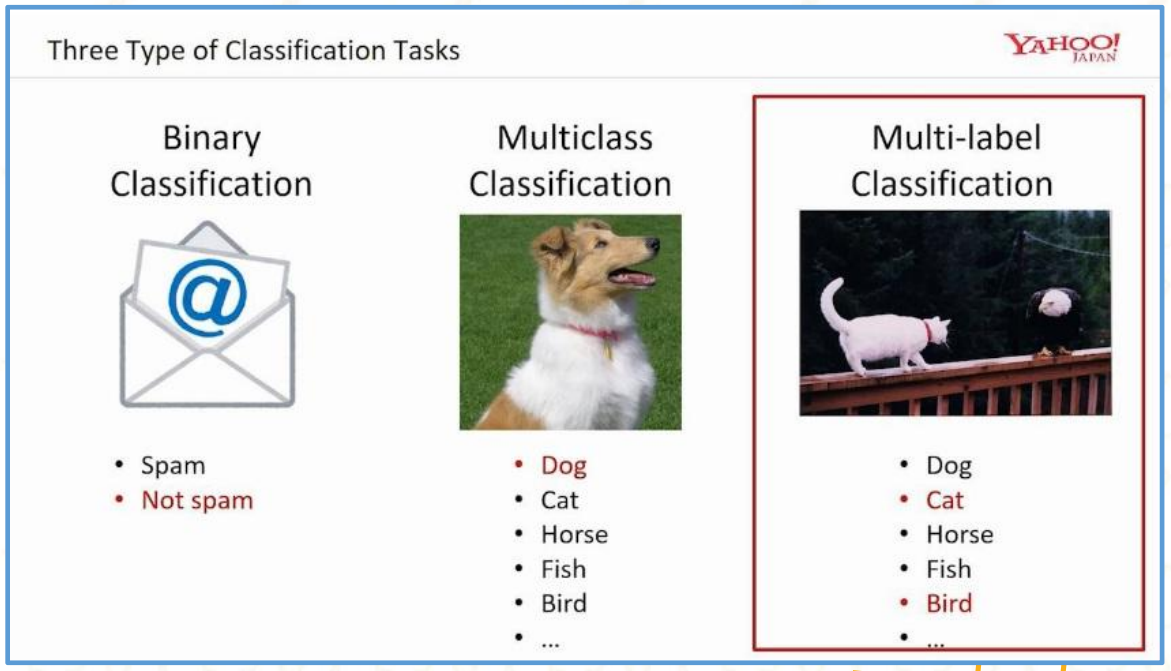


Multi-Class

- A
- B
- C
- …

Multiclass classification

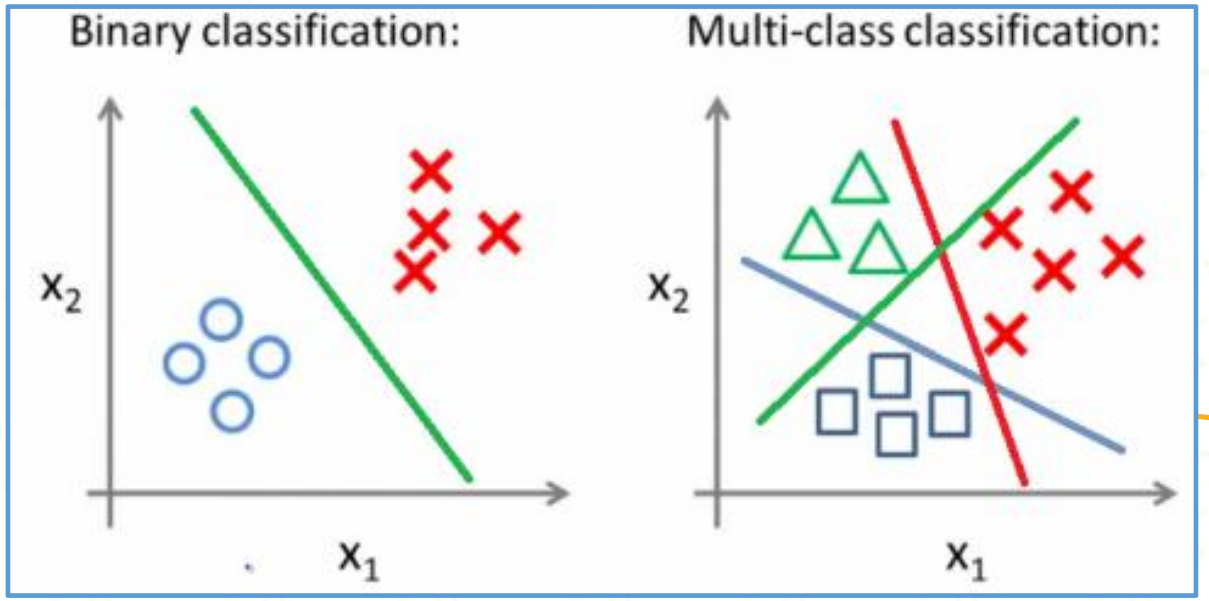
- Multiclass classification
  - Klasifikasi yang melibatkan lebih dari dua Class
  - Setiap data hanya dapat dimiliki oleh satu Class
- Multilabel classification
  - Klasifikasi yang melibatkan lebih dari dua Class
  - Setiap data dapat dimiliki oleh beberapa Class



Binary to multiclass

- Can we use a binary classifier to construct a multiclass classifier?
  - Decompose the prediction into multiple binary decisions

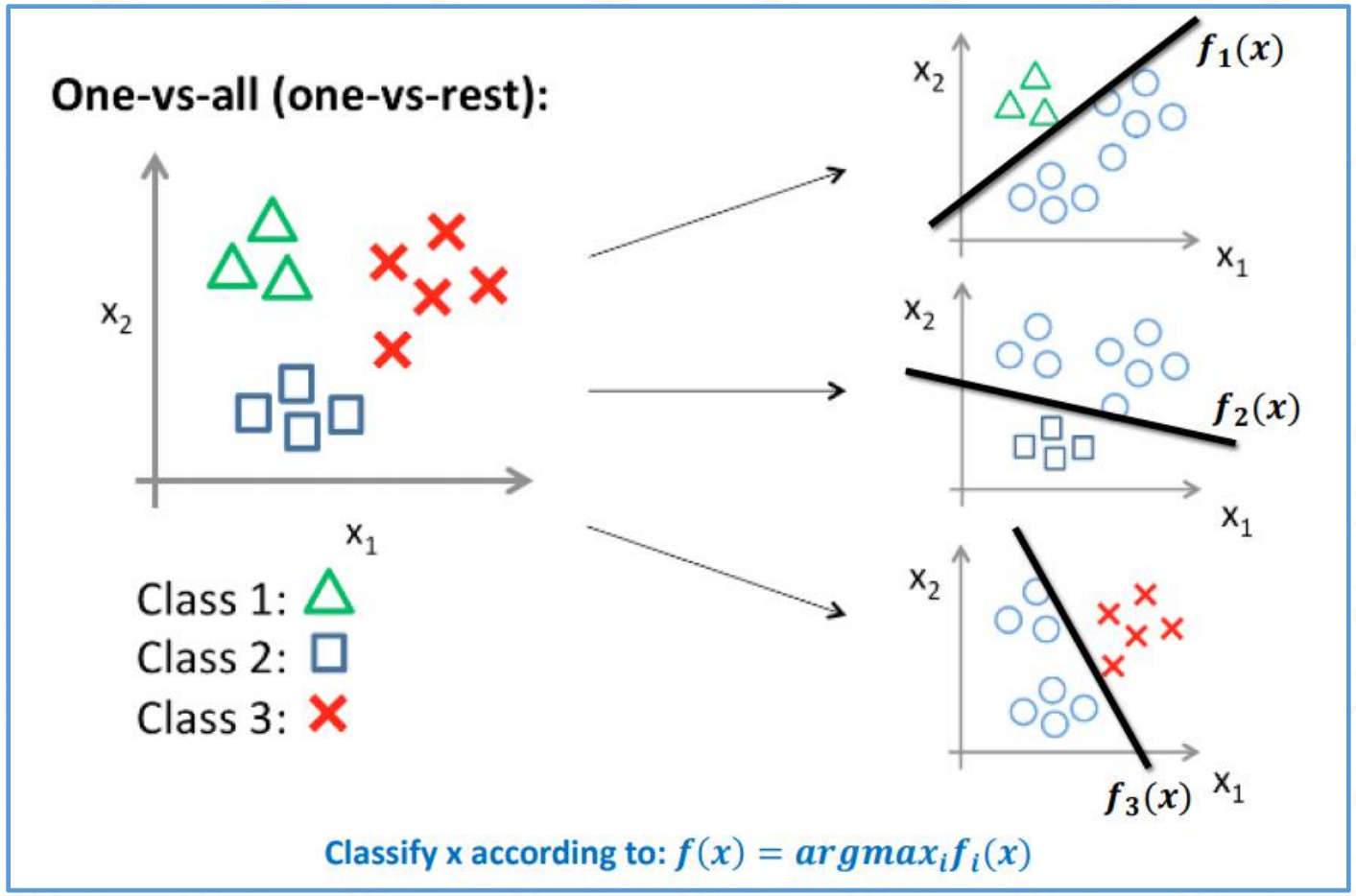
- How to decompose?
  - One-vs-all (One vs Rest/ OvR)
  - One vs One (OvO)



One vs Rest (OvR)

- Learn a classifier one at a time
- Given m classes, train m classifiers: one for each class
- Classifier j: treat tuples in class j as positive & all others as negative
- To classify a tuple X, choose the classifier with maximum value

Visualizing one vs rest



One vs Rest learning algorithm

- Learning: Given a dataset $D = \{(x_i, y_i)\}$
  $x_i \in R^n$, $y_i \in \{1,2,3, \ldots K\}$
- Decompose into K binary classification tasks
  - Learn K models: $w_1, w_2, w_3, \ldots w_K$
  - For class k, construct a binary classification task as:
    - Positive examples: Elements of D with label k
    - Negative examples: All other elements of D
  - The binary classification can be solved by any algorithm we have seen

One vs Rest inference algorithm

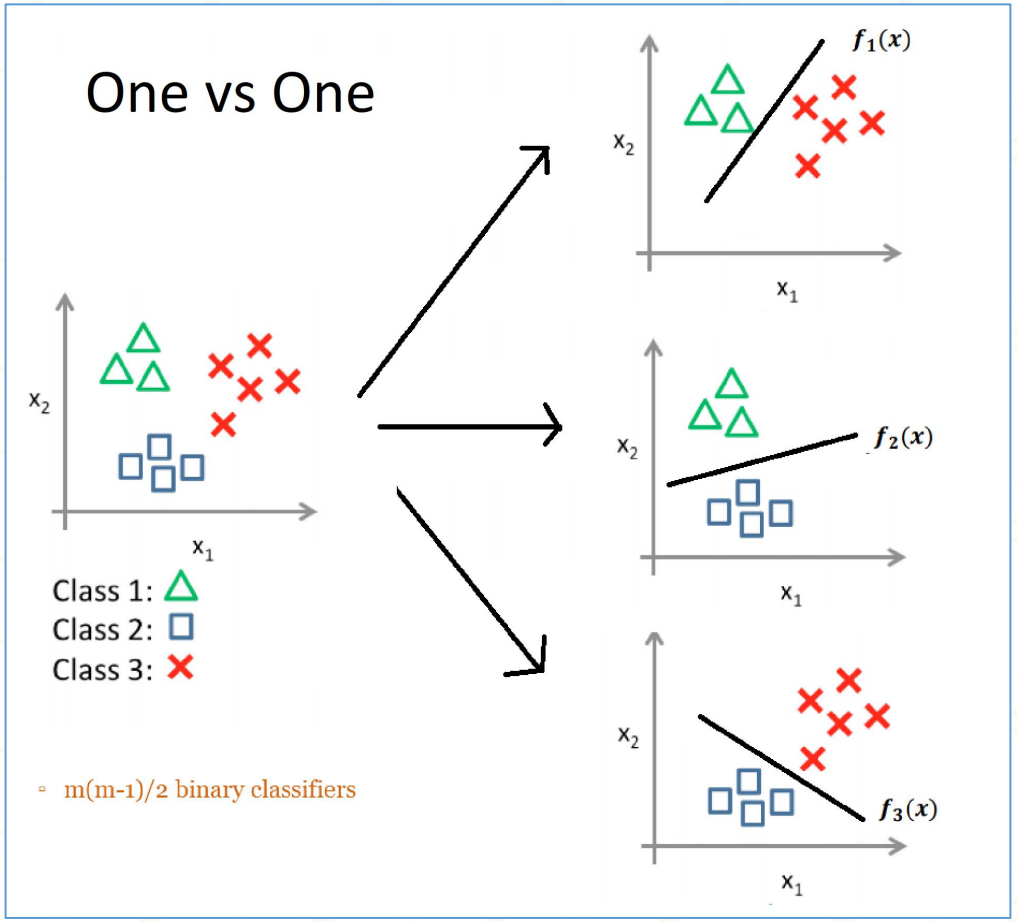
- Inference: “Winner takes all”
  - $\hat{y} = \text{argmax}_{y \in \{1,2,\ldots K\}} \, w_y^T \, x$

For example: $\text{y} = \text{argmax}(w_{red}^T \, x \, , \; w_{blue}^T \, x, \; w_{green}^T \, x)$

One vs One (OvO)

- Learn a classifier for each pair of classes
- Given m classes, construct *m(m-1)/2* binary classifiers
- A classifier is trained using tuples of the two classes
- To classify a tuple X, each classifier votes.
- X is assigned to the class with maximal vote

One vs One (OvO)



Classify x according to majority voting

One vs One learning algorithm

- Learning: Given a dataset $D = \{(x_i, y_i)\}$
  $x_i \in R^n,\ y_i \in \{1,2,3, \ldots K\}$
- Decompose into K(K-1)/2 binary classification tasks
  - Learn K(K-1)/2 models: $w_1,\ w_2,\ w_3, \ldots w_{K*(K-1)/2}$
  - For each class pair (i,j), construct a binary classification task as:
    - Positive examples: Elements of D with label i
    - Negative examples Elements of D with label j
    - The binary classification can be solved by any algorithm

One vs One Inference algorithm

- Prediction:
    - Majority: Pick the label with maximum votes

DigitalSkola
Uncover The World Of Digital Skills With Us

Lets Practice!

DigitalSkola

Uncover The World Of Digital Skills With Us

Thank YOU