AI in Smart Grid (I)

MCUT, EE

Yu-Hsiu Lin
yhlin@mail.mcut.edu.tw

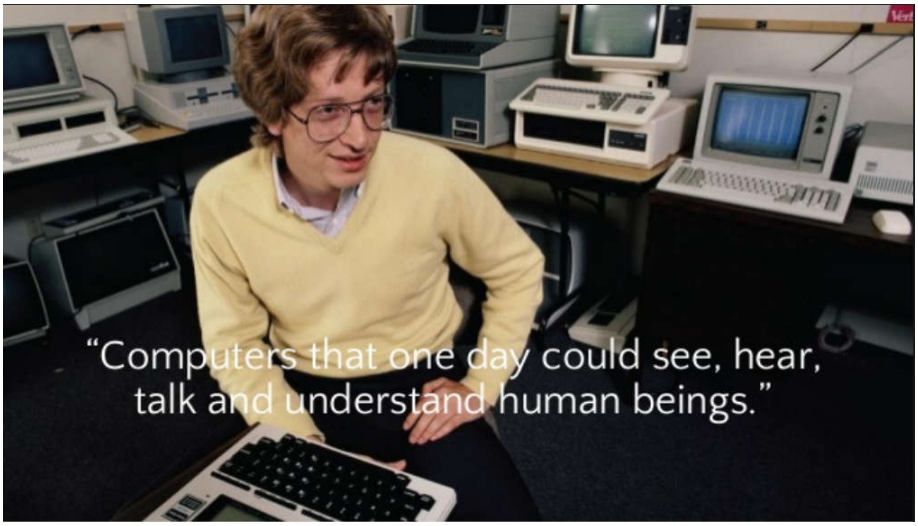
Outline

- Introduction to AI
- Machine Learning VS. Deep Learning
- Supervised learning -- Perceptron

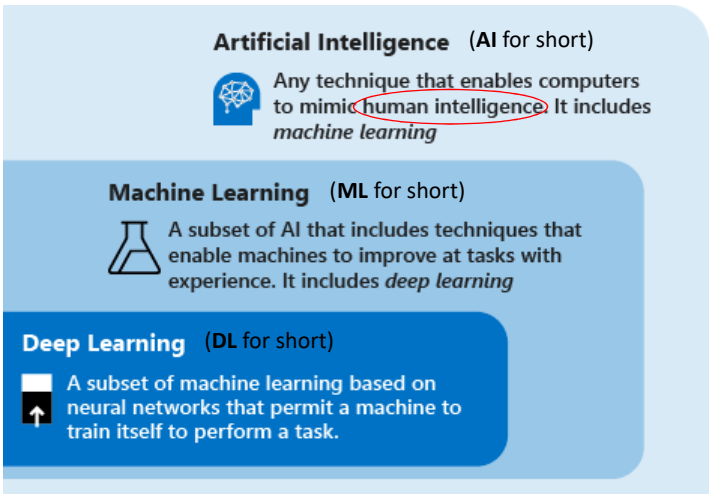
Outline

- Introduction to AI
- Machine Learning VS. Deep Learning
- Supervised learning -- Perceptron

What is AI?



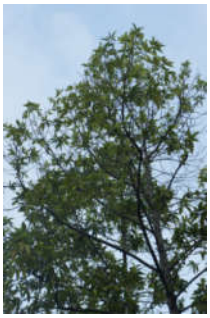
Bill Gates, 1991



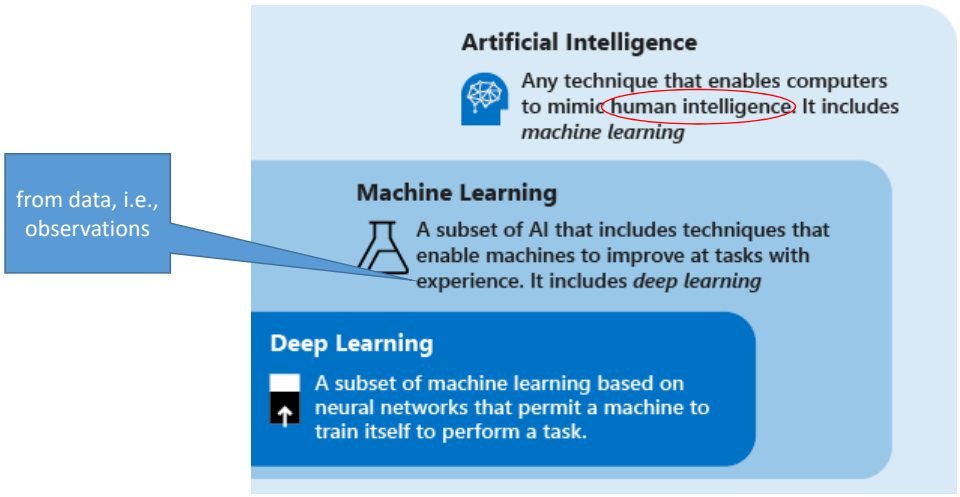
What is human intelligence (VS. AI)?

… in the world



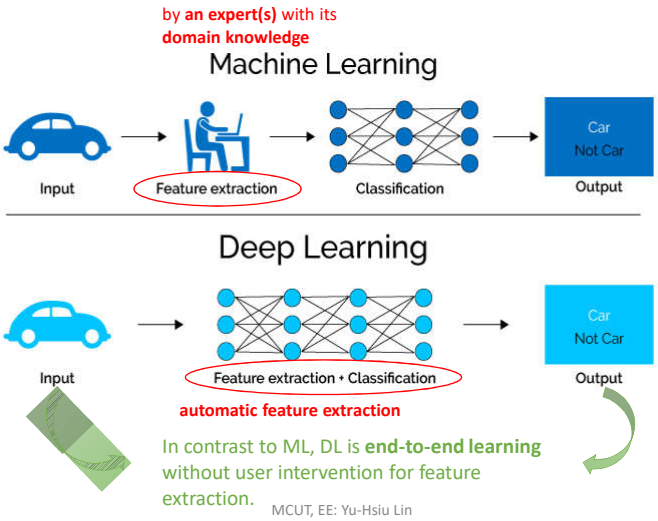
- Define trees and Hand-program: **difficult**
- Learn from observations (data) and Recognize observations: **easy** -- a **3-year-old can do so**



Outline

- Introduction to AI
- Machine Learning VS. Deep Learning
- Supervised learning -- Perceptron

Comparing DL with ML

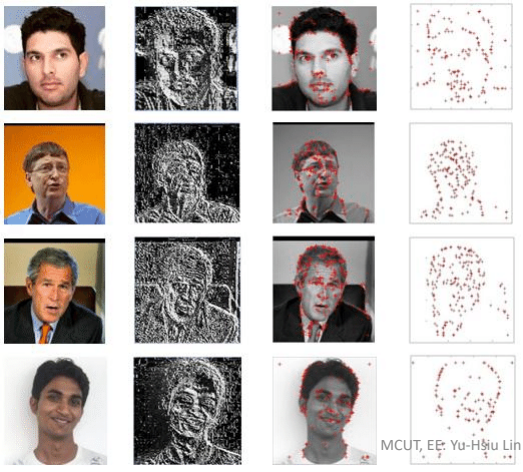


Source: https://semiengineering.com/deep-learning-spreads/.

In contrast to ML, DL is **end-to-end learning** without user intervention for feature extraction.

What is feature extraction/feature engineering?

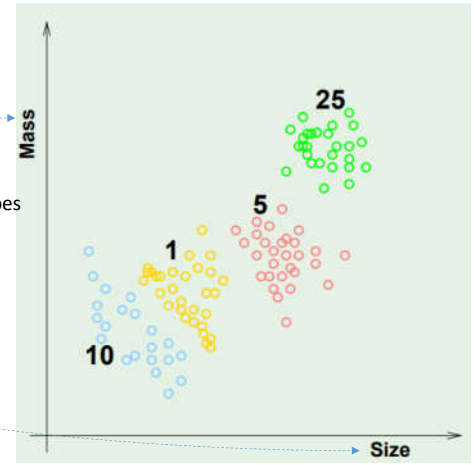
- To a *face recognition* problem



Scale Invariant Feature Transform (SIFT) in digital image processing

To distinguish different persons, you need to
- extract their *facial features* from digital images through some techniques such as SIFT
- train an AI/ML model learning from extracted facial features (facial data)

- To a *coin recognition* problem



Numerical features, size and mass, extracted from different types of coins can be considered.

A feature is an input variable of an AI/ML model.

A simple model might use a single feature, while a more sophisticated model could use millions of features.

- Why there were NO technical term “deep learning (DL)” mentioned?

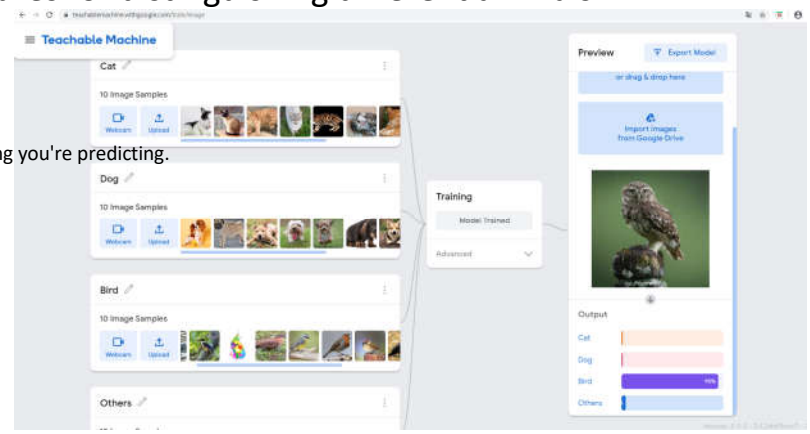
Teachable Machine

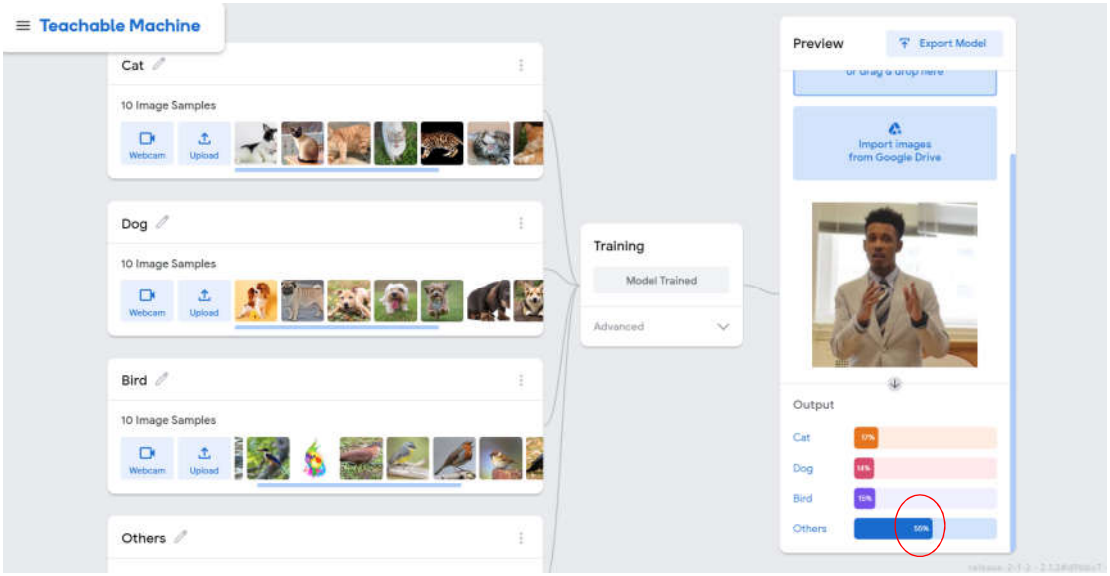
- https://teachablemachine.withgoogle.com/

- What is Teachable Machine?
  - Teachable Machine is a web-based tool that makes creating *Deep Learning* models fast, easy, and accessible to everyone.
  - Using Teachable Machine, you can train a computer, a machine, to recognize your own images, sounds and poses based on *Deep Learning* techniques for your own applications -- sites, apps and more.

- In this demonstration, you do NOT need to define animals' facial features for distinguishing different animals.

A *label* is the thing you're predicting.





The level of accuracy achieved from the model distinguishing animals can be improved by more image data collected and trained by the model.

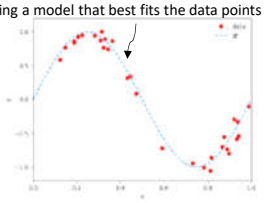
Exercise

- Try for building something interesting of yours! (~30 mins)

What AI/ML/DL can do?

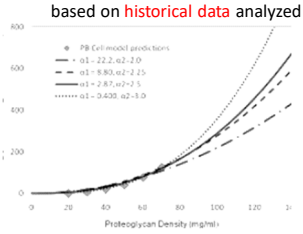
- Classification
  - ➢Image classification
  - ➢…
- Regression/function approximation
  - ➢Curve fitting
  - ➢…
- …more

finding a model that best fits the data points



the interpretation of historical data

aiming at making predictions about future outcomes based on historical data analyzed and modelled



Outline

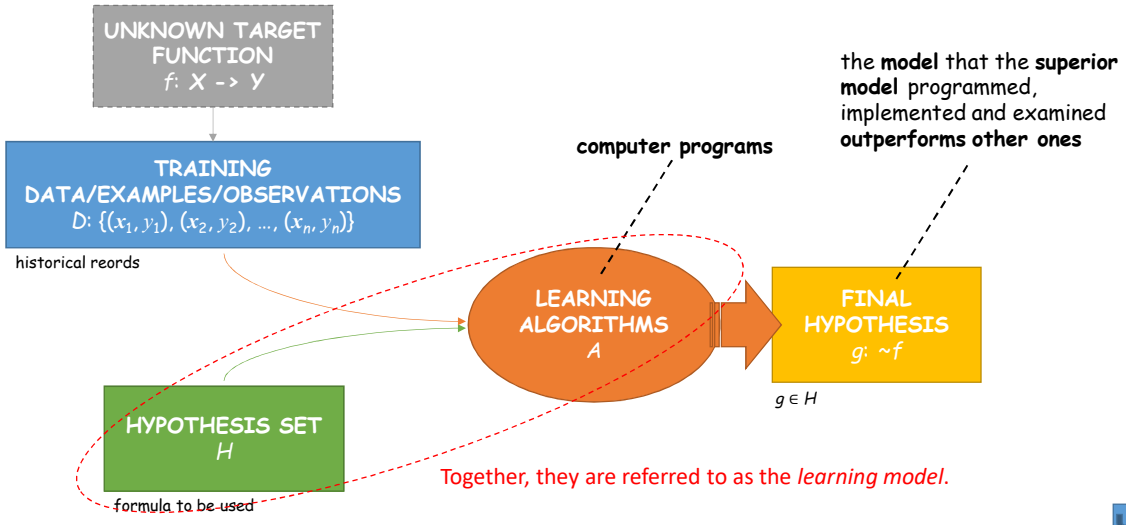
- Introduction to AI
- Machine Learning VS. Deep Learning
- Supervised learning -- Perceptron

Perceptron

Supervised learning: $y_1, y_2, y_3, \ldots$ are given/labeled for learning

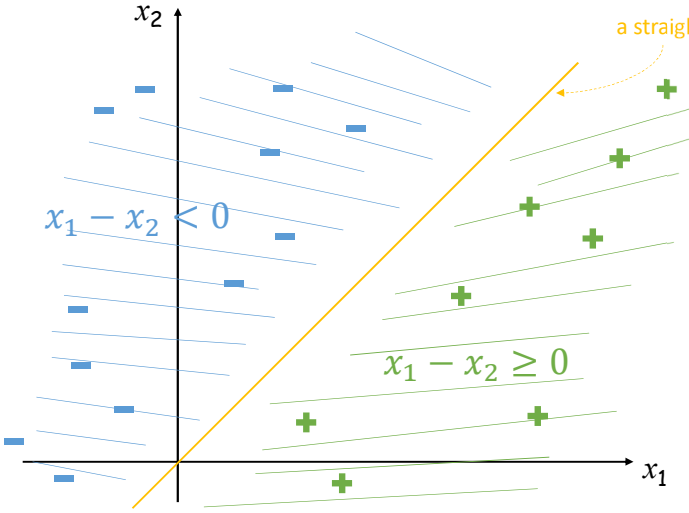
- Components of *Learning*
  - Input: ***X*** applicant information such as (**age, gen… …ual salary, years in job, …**) for a new customer
  - Output: ***y*** a good/bad customer?
  - Unknown target function: f: ***X*** → ***y*** the *ideal* credit approval formula
  - Data: D: $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), (\boldsymbol{x}_3, y_3), \ldots\}$ historical data (customer reords) by existing customers
  - Hypothesis: g: ***X*** → ***y*** an approximated formula to be used

g, a well-trained model from A taking D and H, builds the relationship between ***X*** and ***y*** for f.



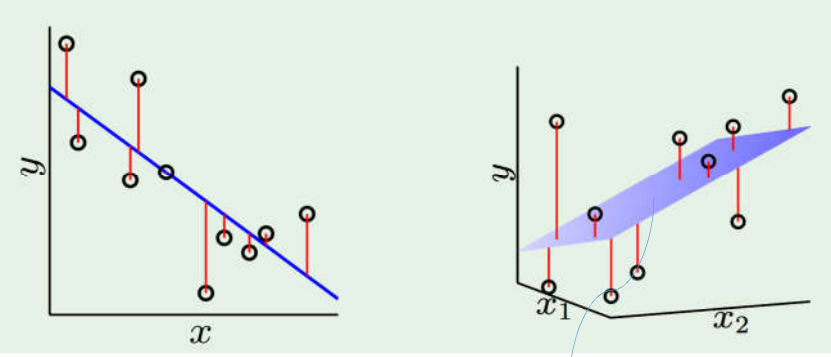
- ***Perceptron*** is one of the simplest hypotheses

How a *Perceptron* can work to separate the two groups of data?



$$y = \begin{cases} Class\ "+", & if\ x_1 - x_2 \geq 0 \\ Class\ "-", & if\ x_1 - x_2 < 0 \end{cases}$$

$$y = \begin{cases} Class\ "+", & if \sum_{i=1}^{n} w_i x_i + w_0 \geq 0 \\ Class\ "-", & if \sum_{i=1}^{n} w_i x_i + w_0 < 0 \end{cases}$$

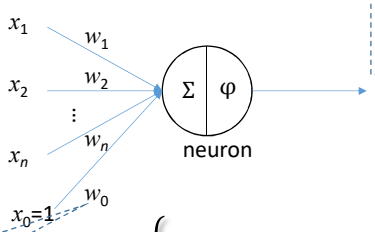


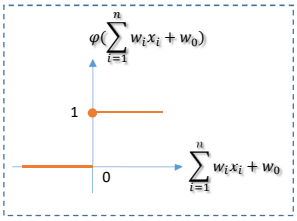
The plane is called a hyperplane when more than two input variables are considered.

How does a *Perceptron* work to perform a *classification* task?

- The basic mathematical model of a perceptron:

$$\sum_{i=1}^{n} w_i x_i + w_0$$





The *bias* is viewed and treated as a trainable weight coefficient, w_0, whose corresponding input variable, x_0, is 1s.

$$y = \begin{cases} Class\ "+", & if \sum_{i=1}^{n} w_i x_i + w_0 \geq 0 \\ Class\ "-", & if \sum_{i=1}^{n} w_i x_i + w_0 < 0 \end{cases}$$

- Perceptron Learning Algorithm (PLA)

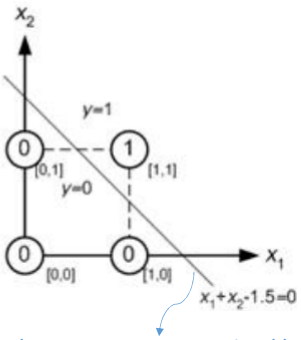
$$\Delta \boldsymbol{w} = \begin{cases} \pm \eta \boldsymbol{x}, & if \text{ there exists a misclassification to a data point } \boldsymbol{x} \text{ inputted} \\ 0, & if \text{ no misclassification to that data point} \end{cases}$$

* For more details about *the updating rule*, refer to https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975.

A very simple example

The truth table of an AND gate

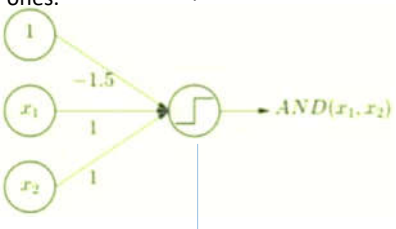
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1



The perceptron was trained by PLA, and is used to classify data.

x_0 is always ones.

Perceptron



To a classification task, the perceptron conducts a step function, a hard/discontinuous function, as its activation function.

- the code used to implement Perceptron on your machine that learns, to classify data, from data

```
import numpy as np

inputs = []  # to prepare your artificial data, according to the truth table, to be learned through PLA
# inputs is an array where its row indicates the number of data points and its column stands for the input variables.
inputs.append(np.array([1, 1, 1]))
inputs.append(np.array([1, 0, 1]))
inputs.append(np.array([0, 1, 1]))
inputs.append(np.array([0, 0, 1]))

labels = np.array([1, 0, 0, 0])  # the desired outputs for the corresponding input data points

Iters = 10  # the maximum number of training epochs

no_of_inputs = 2  # x0 is appended for bias w0.
weights = np.random.randn(no_of_inputs + 1)  # to initialize the initial weight coefficients at random
print("initial: " + str(weights))  # to print the initial weight coefficients: [w1, w2, w0]

learning_rate = 0.15  # to specify the learning rate
```

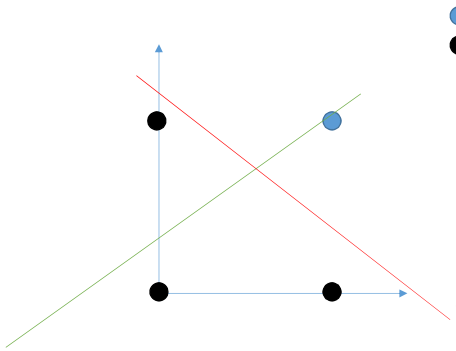


```
# cont'd
for _ in range(Iters):
    for _input, label in zip(inputs, labels):

        summation = np.dot(_input, weights)  # to compute the weighted sum

        if summation > 0:  # to make a decision based on the step activation function
            predicted = 1
        else:
            predicted = 0

        weights += learning_rate * (label - predicted) * _input  # to adjust weights!?

print("trained: " + str(weights))  # to print the trained weight coefficients
```

w_1 w_2 w_0

initial: [0.52404678, -0.92793152, 0.23835091]

trained: [0.22404678, 0.27206848, -0.36164909]



the final decision boundary that correctly classifies the two different groups of data

Google™ Colab

- Colab is a free notebook environment that runs your **Python** codes entirely in the cloud from improving your Python programming skills to work with AI.

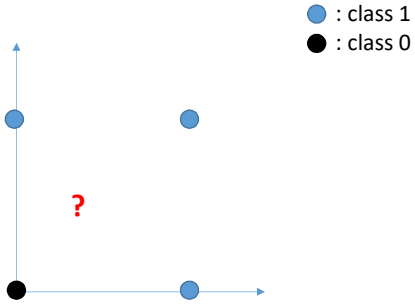
- https://colab.research.google.com/
  - Sign in *your Google account*.
  - …
  - Copy & Paste the code shown previously.
  - Run the code in Colab.

Exercise

- Write a computer program that permits the perceptron to solve the **OR-gate** binary classification problem. (~30 mins)

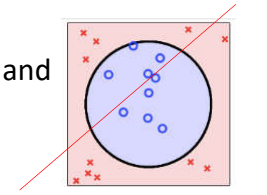
- Depict the initial and resulting ws.

~~• Show the trajectories of the training process where different learning rates are considered and the initial weight coefficients are fixed.~~



More about PLA

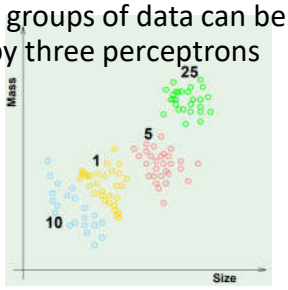
- Perceptron using a step activation function can NOT deal with **non-linearly separable** data VERY WELL
  - ➢More robust models instead of perceptron will be introduced and used to address non-linearly separable data



- There always exist misclassifications
- Note: although there always exist misclassifications, you can still train a perceptron that as few data points as possible are misclassified.

More about PLA (Cont'd)

- How to address *multi-class* classification problems based on PLA?
  - ➢Employ a set of *perceptrons*
  - ➢To a binary classification problem, two different groups of data can be separated by a decision boundary produced by one perceptron
  - ➢To a four-class classification problem, four different groups of data can be separated by three decision boundaries produced by three perceptrons
  - ➢To a *n*-class classification problem, a total of (*n*-1) perceptrons are needed



- *End*