

# **FIRST STEP TO PRACTICAL MACHINE LEARNING**

KNOWLEDGE SHARING FOR CPE/SKE STUDENTS

SIRAKORN LAMYAI

STUDENT, KASETSART U.

OCTOBER 30, 2018

Resources on this slides are mainly adapted and rearranged from

- W. Jitkrittum: Machine Learning Fundamentals I
- K. Muandet: Machine Learning Fundamentals II
- Google's Machine Learning Crash Course

Make sure these are installed on your computer.

This page is a guide for installing on Windows

- Python 3.6: Download and install at <https://www.python.org>
- NumPy, Scipy, Matplotlib, Scikit-learn, MLxtend:  
Run `pip install numpy scipy matplotlib sklearn mlxtend`

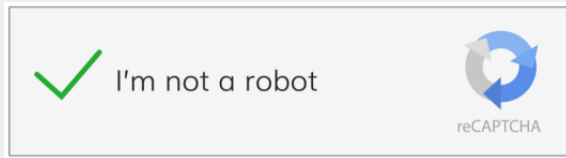
# OUTLINE

- 1 Introduction to Machine Learning
  - What is Machine Learning?
- 2 Machine Learning Problems
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning
- 3 Model
  - The Goal of Machine Learning
- 4 Machine Learning Process
  - Evaluating Machine Learning Performance
- 5 Algorithms for Machine Learning Classification Problem
- 6 Problems for Machine Learning
  - Handwriting recognition
- 7 Neural Networks

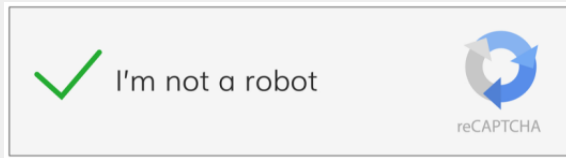
# **INTRODUCTION TO MACHINE LEARNING**

# WHAT IS MACHINE LEARNING?

# WHAT IS MACHINE LEARNING?



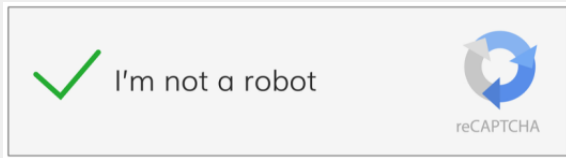
# WHAT IS MACHINE LEARNING?



- This is Recaptcha.

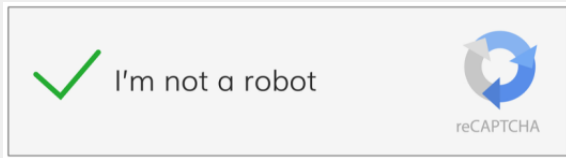


# WHAT IS MACHINE LEARNING?



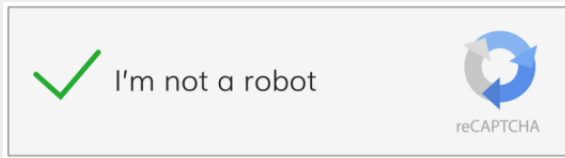
- This is Recaptcha.
  - ▶ Recaptcha helps stop millions of spam a day.

# WHAT IS MACHINE LEARNING?



- This is Recaptcha.
  - ▶ Recaptcha helps stop millions of spam a day.
  - ▶ In some old days, we have to type Captcha texts to distinguish ourself from bots.

# WHAT IS MACHINE LEARNING?



## ■ This is Recaptcha.

- ▶ Recaptcha helps stop millions of spam a day.
- ▶ In some old days, we have to type Captcha texts to distinguish ourself from bots.
- ▶ How is it possible that with a single click, an automated system can distinguish bots from humans?

Machine Learning

Machine Learning

= Improves performance on a specific task.

# **MACHINE LEARNING PROBLEMS**

# TYPES OF MACHINE LEARNING PROBLEMS

# TYPES OF MACHINE LEARNING PROBLEMS

1. Supervised learning



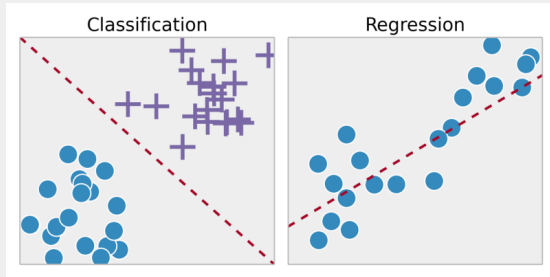
# TYPES OF MACHINE LEARNING PROBLEMS

1. Supervised learning
2. Unsupervised learning

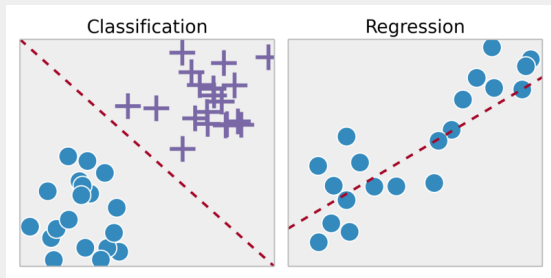
# TYPES OF MACHINE LEARNING PROBLEMS

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

# SUPERVISED LEARNING

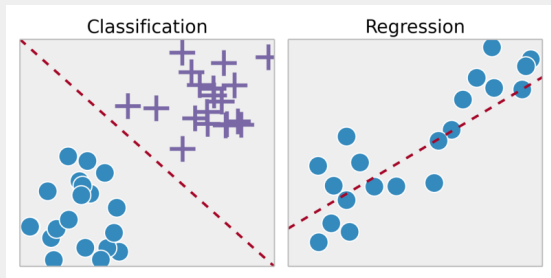


# SUPERVISED LEARNING



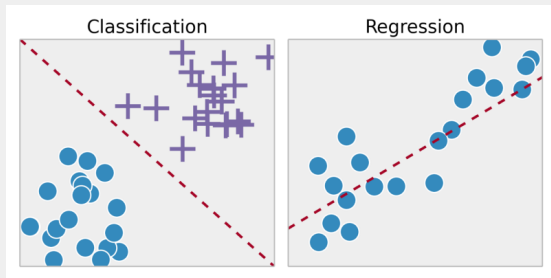
- Given a **training set** for the data, find a **model** to **generalise** well to **unseen** data.

# SUPERVISED LEARNING



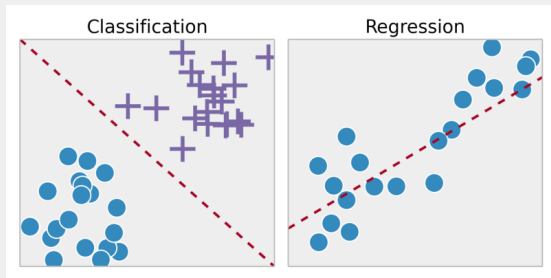
- Given a **training set** for the data, find a **model** to **generalise** well to **unseen** data.
- Two main supervised learning problems

# SUPERVISED LEARNING



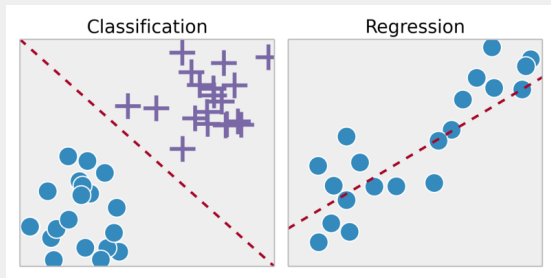
- Given a **training set** for the data, find a **model** to **generalise** well to **unseen** data.
- Two main supervised learning problems
  - ▶ Classification: On the discrete data

# SUPERVISED LEARNING



- Given a **training set** for the data, find a **model** to **generalise** well to **unseen** data.
- Two main supervised learning problems
  - ▶ Classification: On the discrete data
  - ▶ Regression: On the continuous data

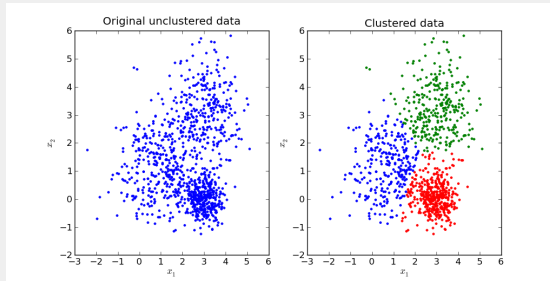
# SUPERVISED LEARNING



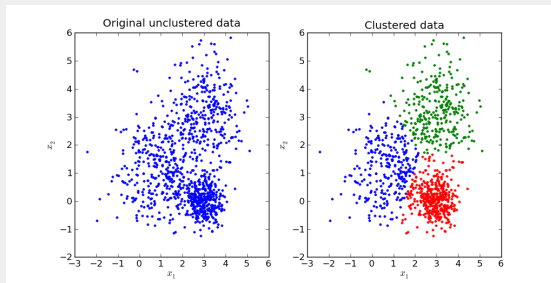
- Given a **training set** for the data, find a **model** to **generalise** well to **unseen** data.
- Two main supervised learning problems
  - ▶ Classification: On the discrete data
  - ▶ Regression: On the continuous data
- Example problems: Spam E-mail detection, Facial recognition



# UNSUPERVISED LEARNING

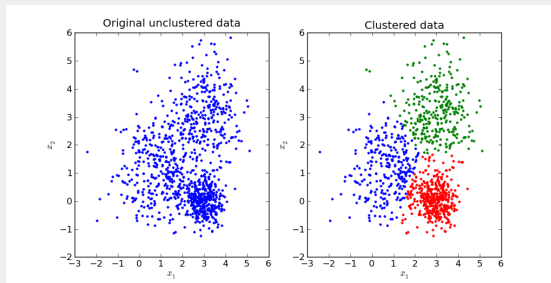


# UNSUPERVISED LEARNING



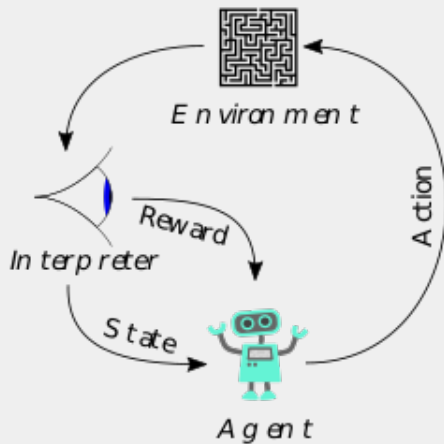
- Discover **hidden** structure in **non-labelled** data.

# UNSUPERVISED LEARNING



- Discover **hidden** structure in **non-labelled** data.
- Example: Clustering, Generative models

# REINFORCEMENT LEARNING



**MODEL**



- A result of the combination between...

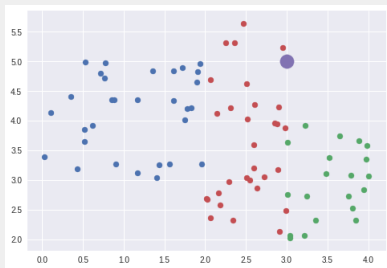
- A result of the combination between...
  - ▶ a **method** to recognise the data, and



- A result of the combination between...
  - ▶ a **method** to recognise the data, and
  - ▶ **sample datas** for such the method

# MODEL

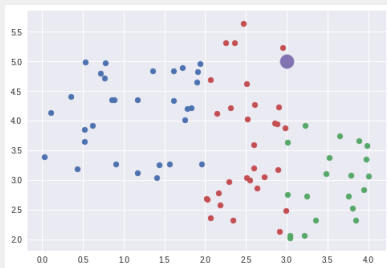
- A result of the combination between...
  - ▶ a **method** to recognise the data, and
  - ▶ **sample datas** for such the method



Data

# MODEL

- A result of the combination between...
  - ▶ a **method** to recognise the data, and
  - ▶ **sample datas** for such the method



Data

Determine which group should the purple dot be in (red/green/blue) by **checking the colour of its nearest dot.**

Method

# Generalise

- We want our model to know how the **data pattern** looks like
- Our model should not "adhere" to one set of data, but instead knows the pattern of all the data.
- Therefore, we want our model to **generalise** over any set of data, given the small portion of data that we used to teach the model.

# BEGINNING WITH OUR FIRST MODEL

- We're going to write our **first own** machine learning algorithm called ***k*-Nearest Neighbour** (*k*-NN)

- We're going to write our **first own** machine learning algorithm called ***k*-Nearest Neighbour** (*k*-NN)
  - ▶ *k*-NN is known to be very simple, with its concept as

- We're going to write our **first own** machine learning algorithm called ***k*-Nearest Neighbour** (*k*-NN)
  - ▶ *k*-NN is known to be very simple, with its concept as

## *k*-NN algorithm

To classify label of a data point, get *k* nearest data points to the data point, and select the major label among those data points.



# **MACHINE LEARNING PROCESS**

# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

## CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

- What if we choose  $k = \#$  of all points?

# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

- What if we choose  $k = \#$  of all points?
  - ▶ What will happen if our dataset's got 3 labels of A, B, C with 10, 20, and 30 data points of each?

# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

- What if we choose  $k = \#$  of all points?
  - ▶ What will happen if our dataset's got 3 labels of A, B, C with 10, 20, and 30 data points of each?
  - ▶ Answer: Our model will always answer the labels with the highest data point count.

# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

- What if we choose  $k = \#$  of all points?
  - ▶ What will happen if our dataset's got 3 labels of A, B, C with 10, 20, and 30 data points of each?
  - ▶ Answer: Our model will always answer the labels with the highest data point count.
- What if we choose  $k = 1$ ?

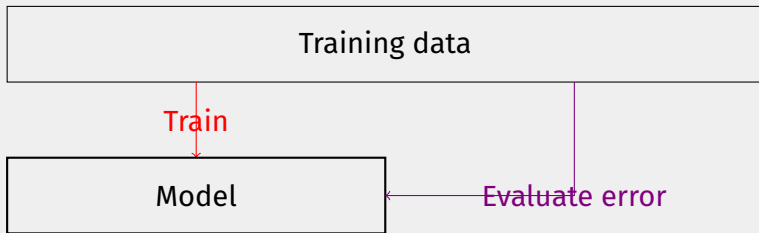
# CHOOSING THE PARAMETER FOR $k$ -NN ALGORITHM

What is the bad way to choose  $k$ ?

- What if we choose  $k = \#$  of all points?
  - ▶ What will happen if our dataset's got 3 labels of A, B, C with 10, 20, and 30 data points of each?
  - ▶ Answer: Our model will always answer the labels with the highest data point count.
- What if we choose  $k = 1$ ?
  - ▶ Let's try!



# TRAINING ERROR



## Loss function

$$L(y, \hat{y}) = \begin{cases} 0 & y = \hat{y} \\ 1 & y \neq \hat{y} \end{cases}$$

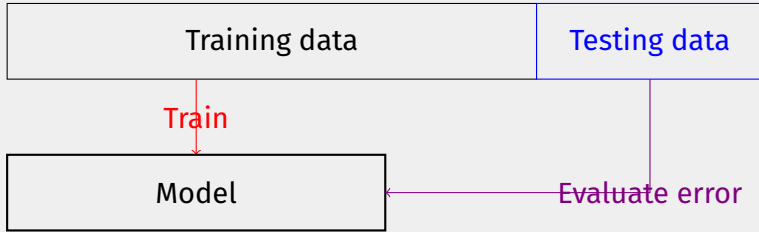
## Error

$$\sigma = \frac{1}{N} \sum_{i=0}^N L(y_i, \hat{y}_i)$$

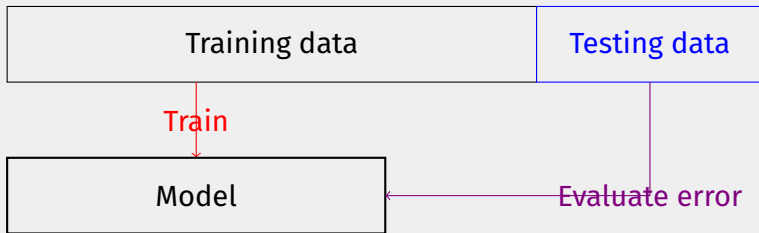
# WHAT WENT WRONG WITH OUR INTUITION?

- Evaluating error with a data points that our model had already seen is bad.
- Why? Because our model already knows the answer to that data point! It could just simply answer by looking at the "answer key"
- So how should we evaluate our model?

# TRAINING AND TESTING SET

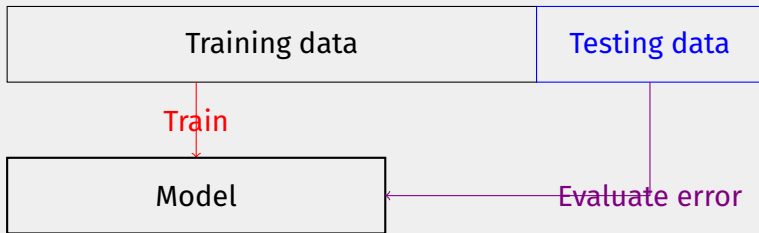


# TRAINING AND TESTING SET



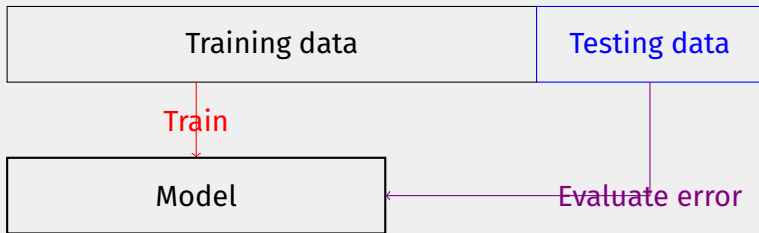
- We separate our dataset into 2 parts: the **training set** and **testing set**

# TRAINING AND TESTING SET



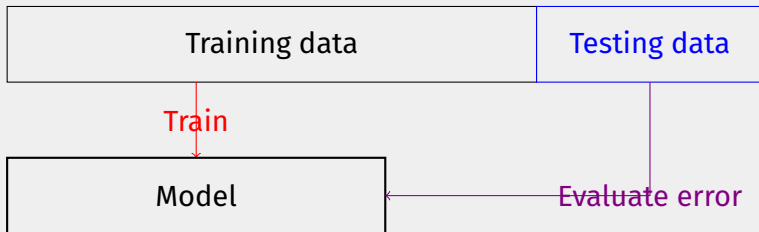
- We separate our dataset into 2 parts: the **training set** and **testing set**
  - ▶ Our model sees the correct label of the training set, but not the testing set.

# TRAINING AND TESTING SET



- We separate our dataset into 2 parts: the **training set** and **testing set**
  - ▶ Our model sees the correct label of the training set, but not the testing set.
  - ▶ We all know the correct label of both the training and testing set.

# TRAINING AND TESTING SET



- We separate our dataset into 2 parts: the **training set** and **testing set**
  - ▶ Our model sees the correct label of the training set, but not the testing set.
  - ▶ We all know the correct label of both the training and testing set.
  - ▶ Teach our model with the training set, see how it performs with the testing set.

# CHOOSING THE BEST $k$

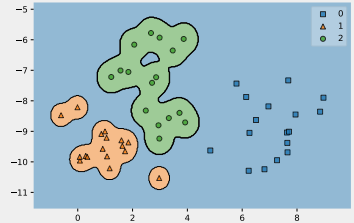
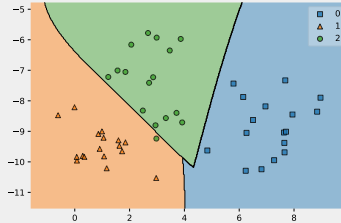
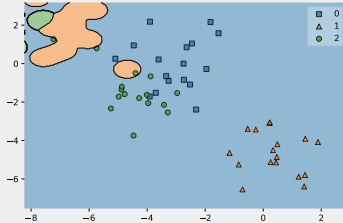
What will happen if...

- our  $k$  is too small?
- our  $k$  is too large?



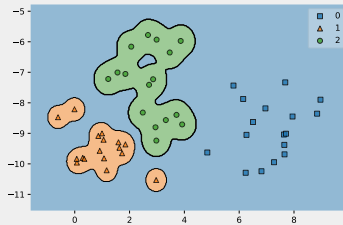
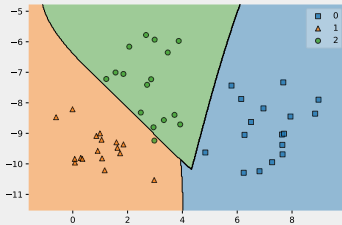
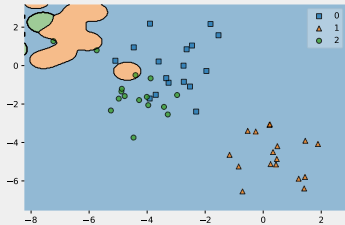
# OVERFITTING AND UNDERFITTING

Which decision region is good?



# OVERFITTING AND UNDERFITTING

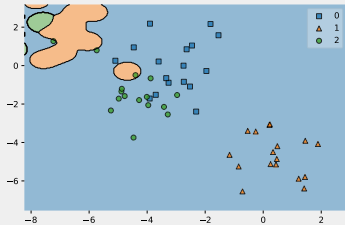
Which decision region is good?



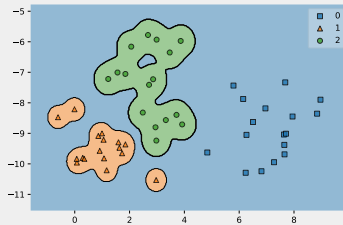
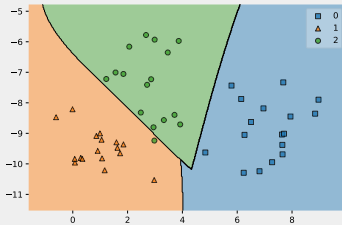
**Underfit:** The model fails to recognise data pattern

# OVERFITTING AND UNDERFITTING

Which decision region is good?



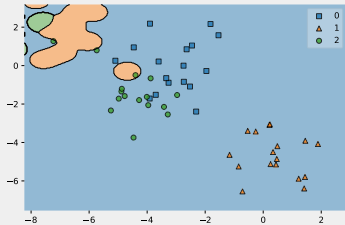
**Underfit:** The model fails to recognise data pattern



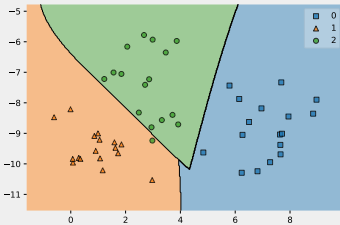
**Overfit:** The model **remembers** data pattern instead of generalising.

# OVERFITTING AND UNDERFITTING

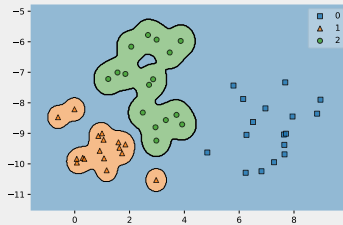
Which decision region is good?



**Underfit:** The model fails to recognise data pattern



**Good fit:** The model recognises data pattern generally



**Overfit:** The model **remembers** data pattern instead of generalising.

Good model must **generalise**

- Actually, the key point in  $k$ -NN algorithm is choosing  $k$  points with the least **distant**.
- What is **distant**?

# NORM FOR $k$ -NN ALGORITHM

## Norm

In linear algebra, a **norm** is a function that assigns a strictly positive length or size to each vector in a vector space - except for the zero vector, which is assigned a length of zero.

Given  $\vec{x}$  as an  $N$ -dimension vector of  $[x_1 \ x_2 \ \dots \ x_n]$

- $l_1$  Norm:  $|x|_1 = \sum_{i=0}^N |x_i|$  (Manhattan)
- $l_2$  Norm:  $|x|_2 = \sqrt{\sum_{i=0}^N x_i^2}$  (Euclidian)
- $l_p$  Norm:  $|x|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$  (Minkowski)

# **ALGORITHMS FOR MACHINE LEARNING CLASSIFI- CATION PROBLEM**



$k$ -NN is a very simple intuition for machine learning algorithms. However, there exists more algorithm that performs well to other problems.

Example algorithms:

- Naïve Bayes
- SVM
- Decision Tree
- Logistic Regression

# NAÏVE BAYES

	Gender	Hair
1	M	Long
2	M	Short
3	F	Long
4	F	Long
5	F	Short

Can we *guess* the gender from hair's length?

■  $P(\text{Male}|\text{Long hair}) = \frac{1}{3}$

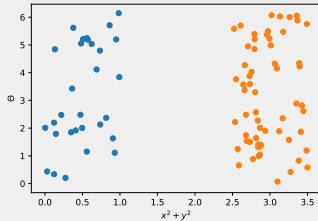
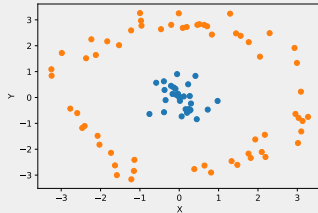
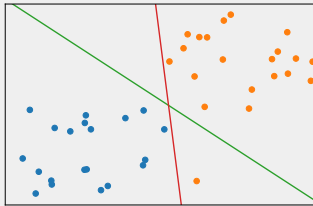
■  $P(\text{Female}|\text{Long hair}) = \frac{2}{3}$

Therefore, we guess that the long-haired person is more likely to be a female.

## Bayes Theorem

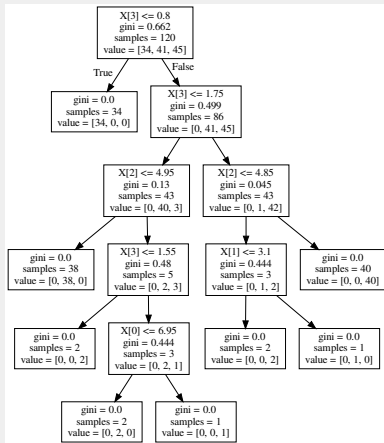
$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A) \times P(A)}{P(B)}$$

# SUPPORT VECTOR MACHINES (SVM)

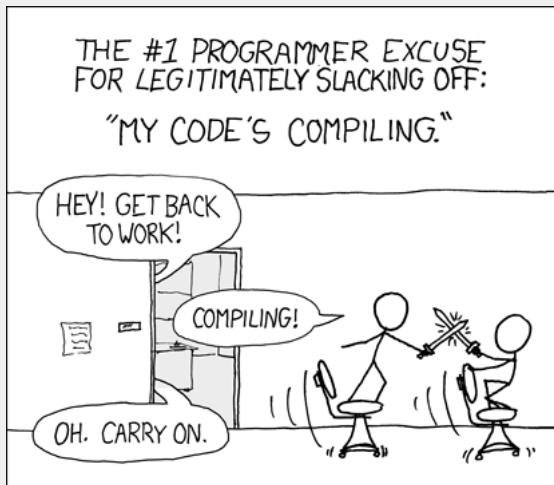


- Goal: to draw a line to separate groups of data
- Ideal good line: maximising the distance between the line and classes of data points
- What if the data is not linearly separable? **Kernel tricks**

# DECISION TREE



- Creating an if-else conditions automatically
- Nested conditions with a parameter to determine how does the separating of the "tree" performs.



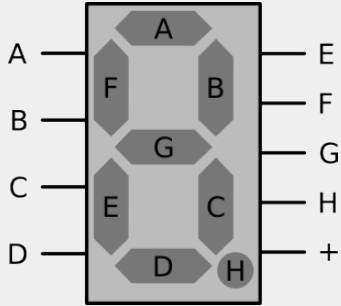
**Figure:** xkcd - Compiling



**Figure:** xkcd - Compiling

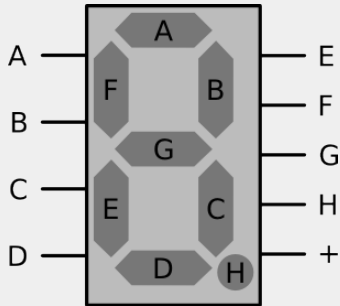
# **PROBLEMS FOR MACHINE LEARNING**

# 7-SEGMENT DISPLAY



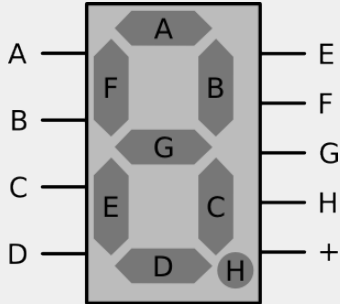


# 7-SEGMENT DISPLAY



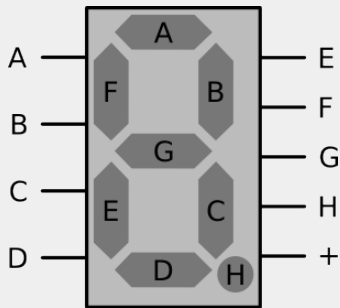
■ This is a 7-segment display.

# 7-SEGMENT DISPLAY



- This is a 7-segment display.
- It consists of a bulb labelled from A-G that could form a number.

# 7-SEGMENT DISPLAY

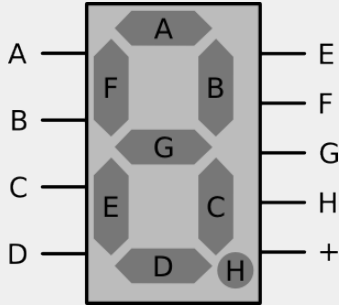


- This is a 7-segment display.
- It consists of a bulb labelled from A-G that could form a number.

## Problem

When the list of the bulb that went on were given, can we determine the number?

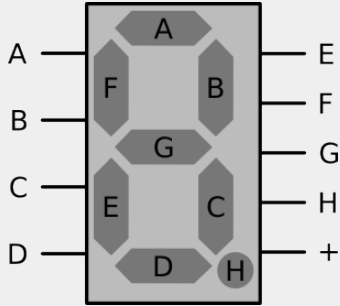
# 7-SEGMENT DISPLAY



## Problem

When the list of the bulb that went on were given, can we determine the number?

# 7-SEGMENT DISPLAY

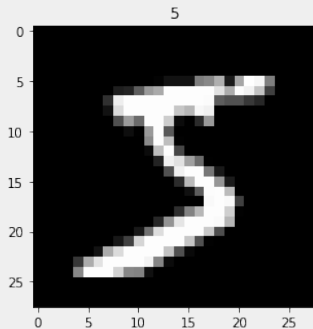


## Problem

When the list of the bulb that went on were given, can we determine the number?

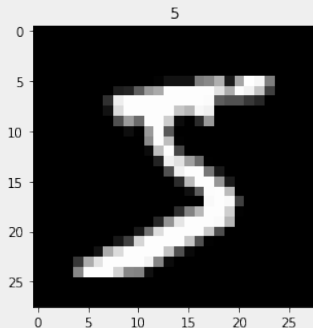
Not only yes, but *easily* yes!

```
if led_on == (b, c):  
    return 1  
elif led_on == (a, b, g, e, d):  
    return 2  
...
```



## Problem

When the image of the handwriting were given, can we determine the number?



## Problem

When the image of the handwriting were given, can we determine the number?

With an **explicit algorithm**? Obviously no! There are too many ways of drawing the number!

# MNIST DATASET





# MNIST DATASET



- 28\*28 pixel images of handwritten numbers (0-9)

# MNIST DATASET



- 28\*28 pixel images of handwritten numbers (0-9)
- 60,000 training images

# MNIST DATASET



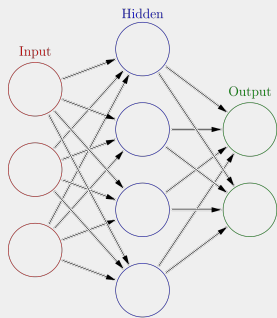
- 28\*28 pixel images of handwritten numbers (0-9)
- 60,000 training images
- 10,000 testing images

- Training: Pretty fast, no calculations on training phase
- Testing: *\*thinking\**
  - ▶ 60,000 data points to calculate the distant + 10,000 data points to test
  - ▶ = 600,000,000 calculations to be made  
(this excludes sorting, of which is a  $\mathcal{O}(n)$  process)
  - ▶ = **(relatively) slow**

- Training: Pretty fast, no calculations on training phase
- Testing: *\*thinking\**
  - ▶ 60,000 data points to calculate the distant + 10,000 data points to test
  - ▶ = 600,000,000 calculations to be made  
(this excludes sorting, of which is a  $\mathcal{O}(n)$  process)
  - ▶ = **(relatively) slow**



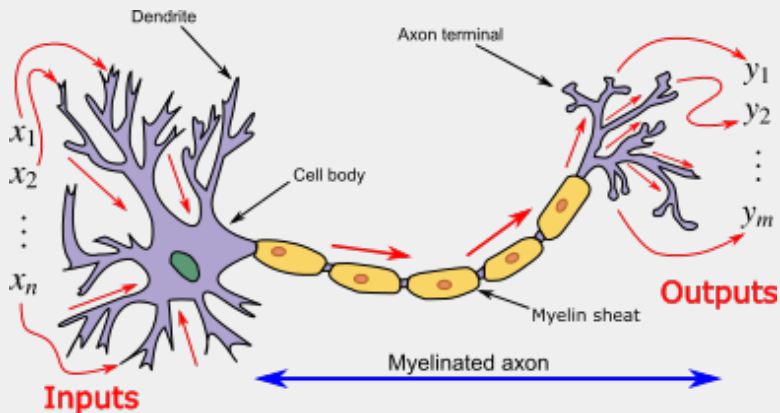
# ARTIFICIAL NEURAL NETWORKS (ANN)



This seems complex, right? We'll get start a little by little...

**Figure:** Neural network  
(Courtesy: Glosser.ca from  
Wikimedia Commons)

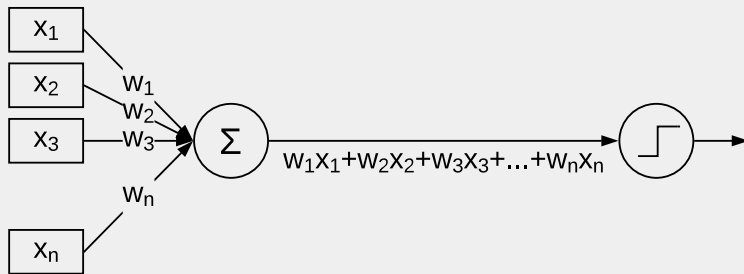
# NEURON



**Figure:** Neuron (Courtesy: Egm4313.s12 from Wikimedia Commons)

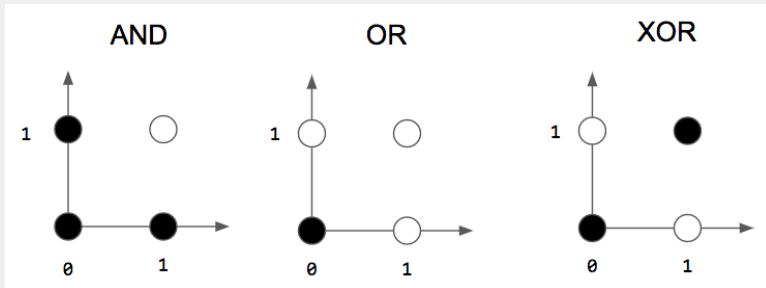


# PERCEPTRON



- **Inputs** consisting of  $n$  inputs from  $x_1, x_2 \dots$  to  $x_n$ .
- **Weights** of each inputs, namely  $w_1, w_2, \dots, w_n$
- **Summation** of all the weighted inputs  $\Sigma = w_1x_1 + w_2x_2 + \dots + w_nx_n$
- **Activation function** in either the form of  $\Sigma > k$  or  $\Sigma < k$

# LOGIC GATES WITH PERCEPTRON



# LOGIC GATES WITH PERCEPTRON

## AND gate

- $w_1 = 1$
- $w_2 = 1$
- $\Sigma = 1x_1 + 1x_2$
- $f : \Sigma \geq 2$

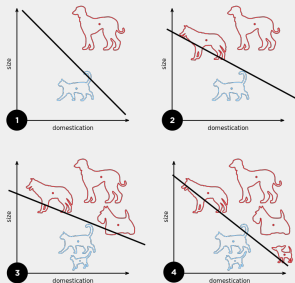
## OR gate

- $w_1 = 1$
- $w_2 = 1$
- $\Sigma = 1x_1 + 1x_2$
- $f : \Sigma \geq 1$

## XOR gate

Why can't XOR gate be created using a perceptron?

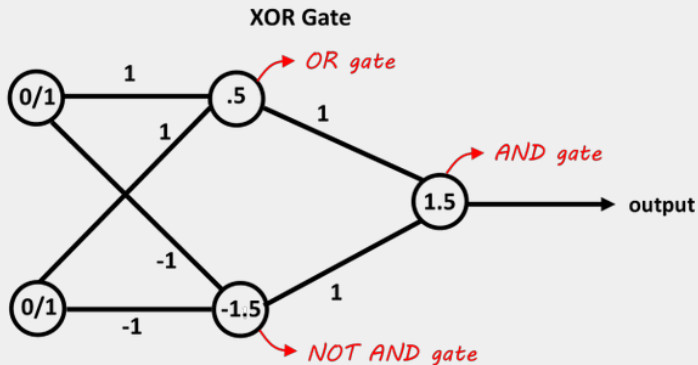
# LINEARLY SEPARABLE PROBLEM



**Figure:** Linearly Separable Problem  
(Courtesy: Elizabeth Goodspeed  
from Wikimedia Commons)

- Now our problem is that the perceptron is a **linear classifier**, that means it could only separate datas that is linearly separable.
- Real-world problems are not that easy to separate
- How can we solve this problem?

# NEURAL NETWORKS



**Figure:** XOR gate with Perceptrons connected together (Courtesy: Parth Udawant)