

**Mecha**tronics

# Day 1

## Introduction to AI and Machine Learning

# Outline

- Overview
- Application of AI
- Type of Machine Learning
- Challenges of AI in industries
- Basic Math concepts for AI

# Overview of AI, ML, and DL

- Artificial Intelligence
- Machine Learning
- Deep Learning
- Generative AI
- Data Science

# Activity 1: What is AI?

Please describe something that you know about AI?

# AI – Artificial Intelligence

It refers to computer programs that can complete cognitive tasks typically associated with human intelligence.



Optimus Gen 2  
humanoid robot by Tesla

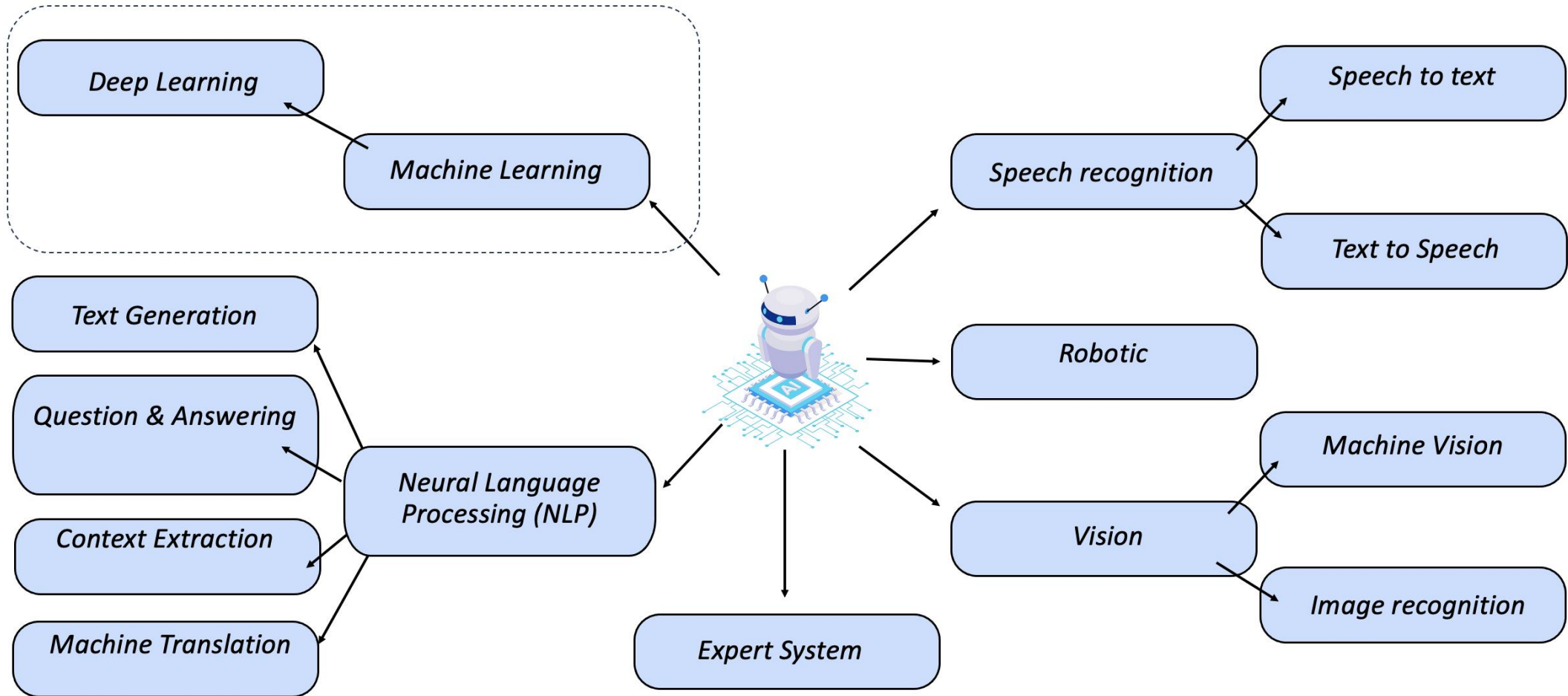
# AI – Artificial Intelligence



AI Vision Systems

Source: (<https://bhtechies.com/ai-vision-systems/>)

# AI – Artificial Intelligence



# AI – Artificial Intelligence

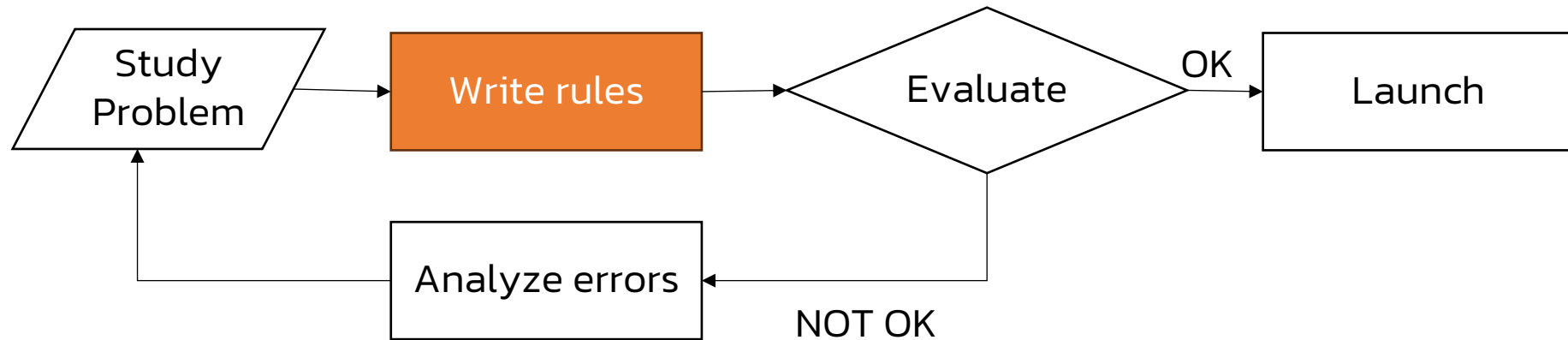
There are two main techniques used to design AI programs

- Rule-based techniques
- Machine learning techniques



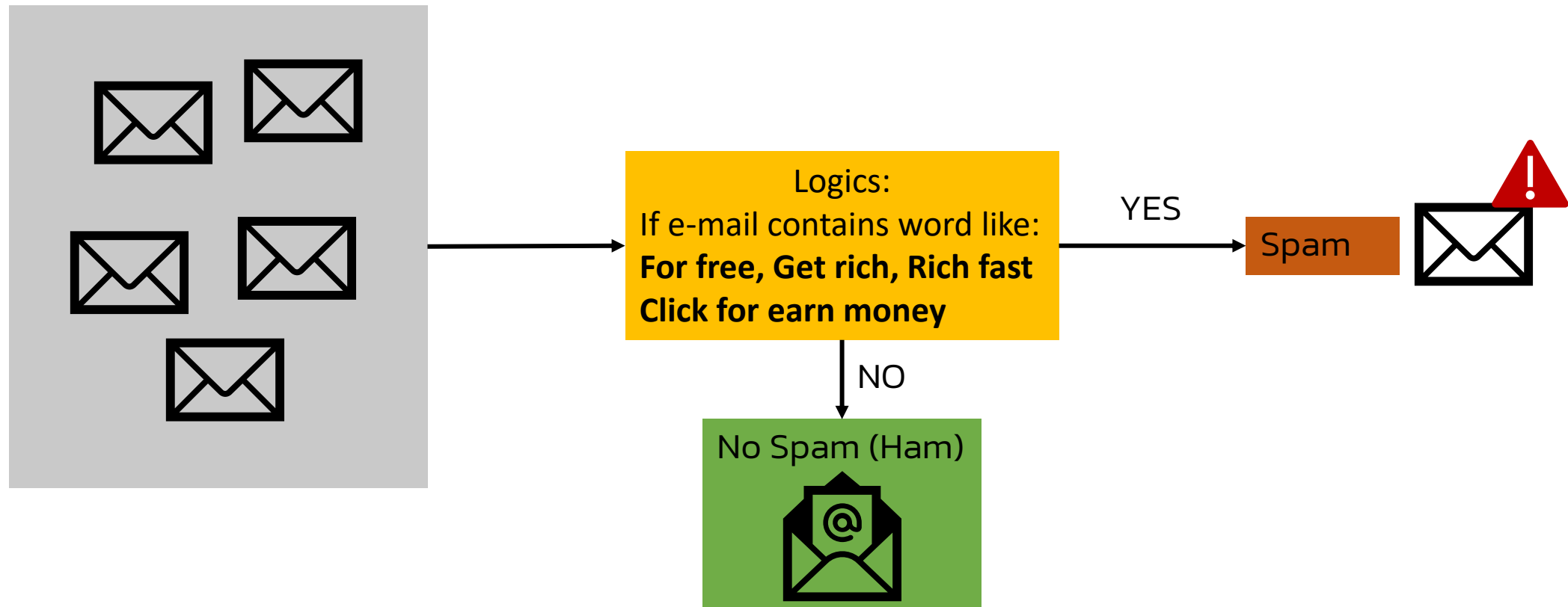
# Rule-based techniques

involve creating AI programs that strictly follow predefined rules to make decisions.



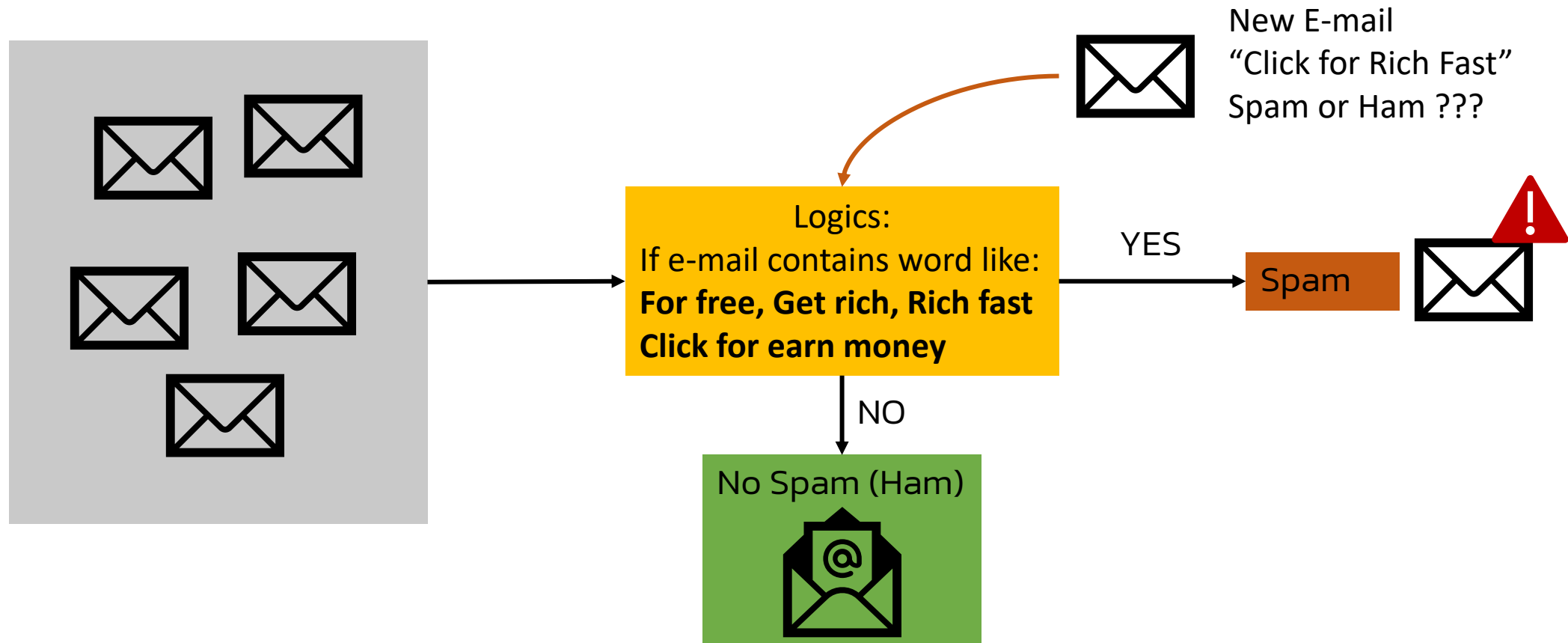
# Rule-based techniques

For example, a spam filter using rule-based techniques might block emails that contain specific keywords using its predefined logic.



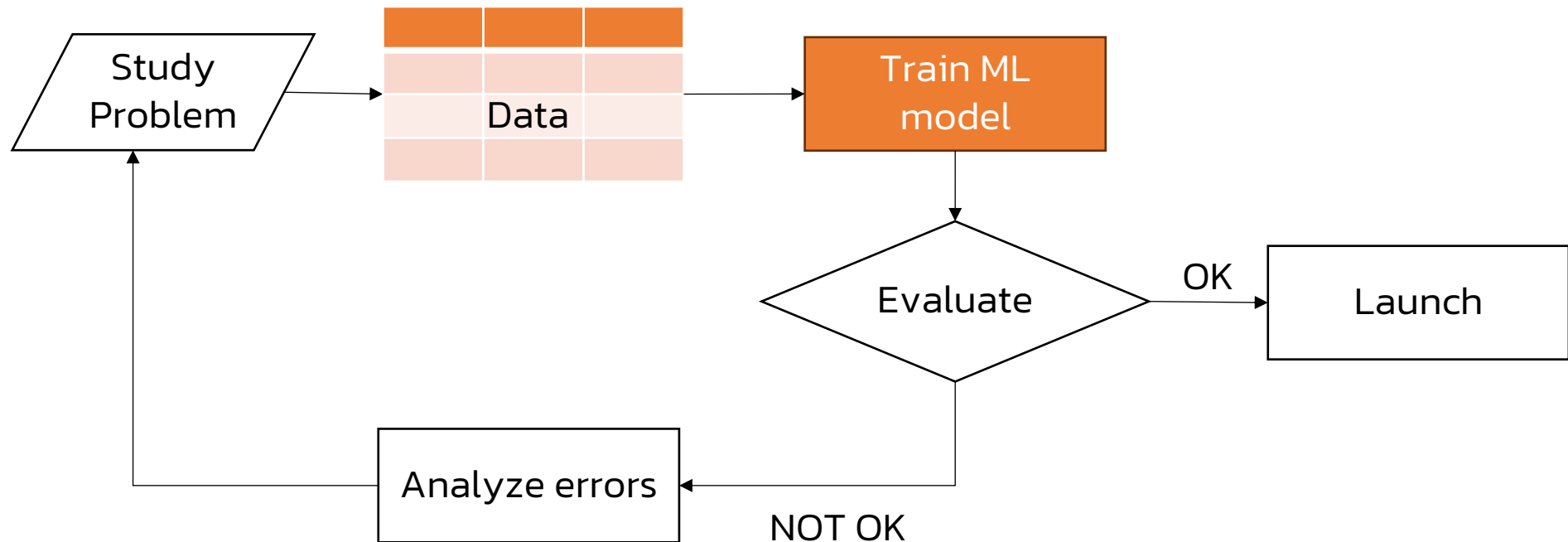
# Rule-based techniques

For example, a spam filter using rule-based techniques might block emails that contain specific keywords using its predefined logic.



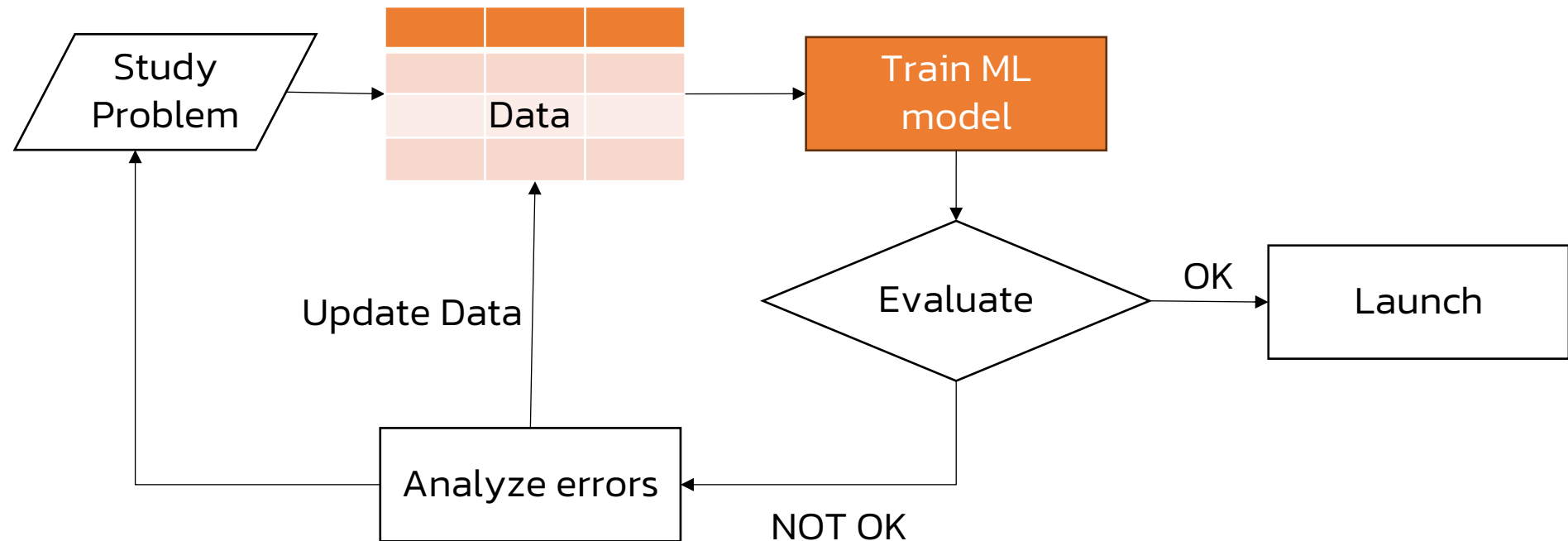
# Machine learning techniques

involve creating AI programs that can analyze and learn from patterns in data to make independent decisions.



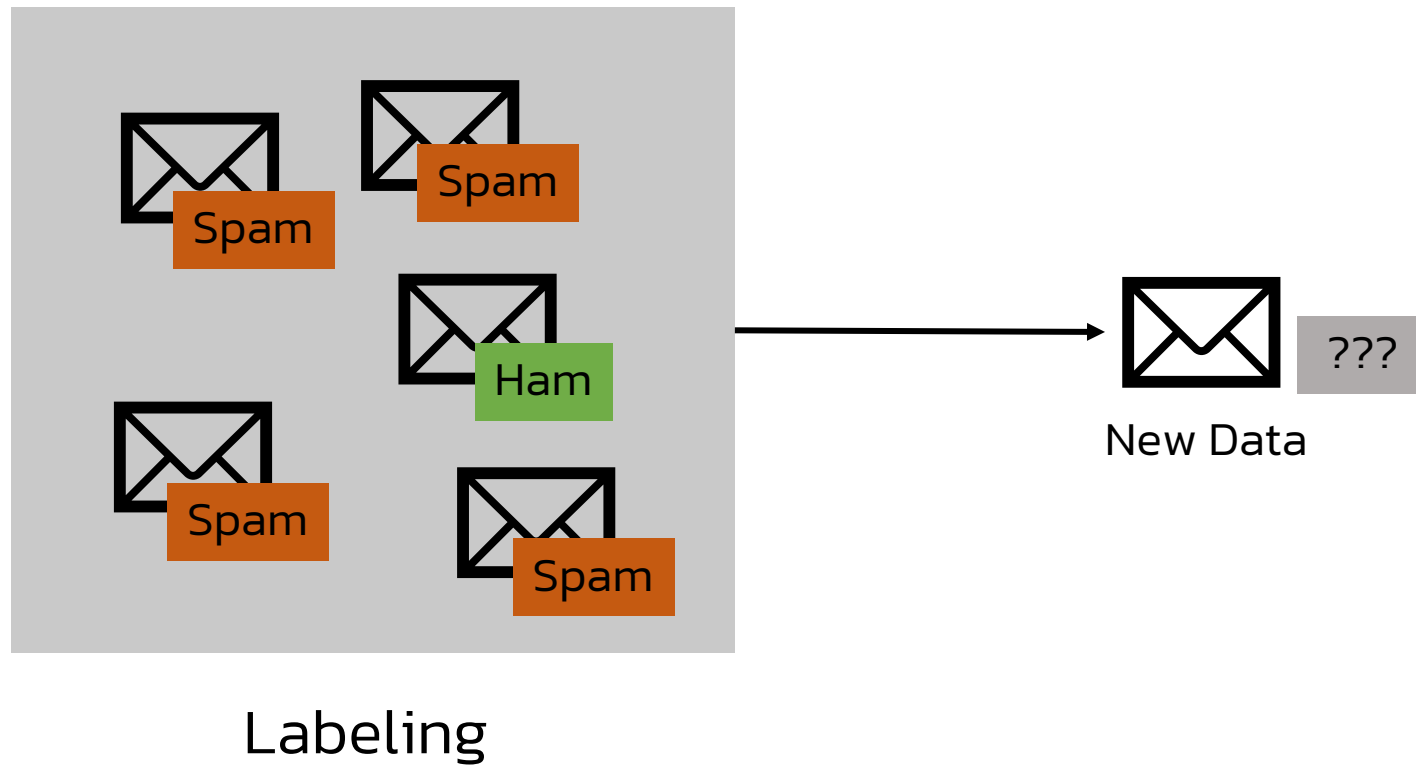
# Machine learning techniques

involve creating AI programs that can analyze and learn from patterns in data to make independent decisions.



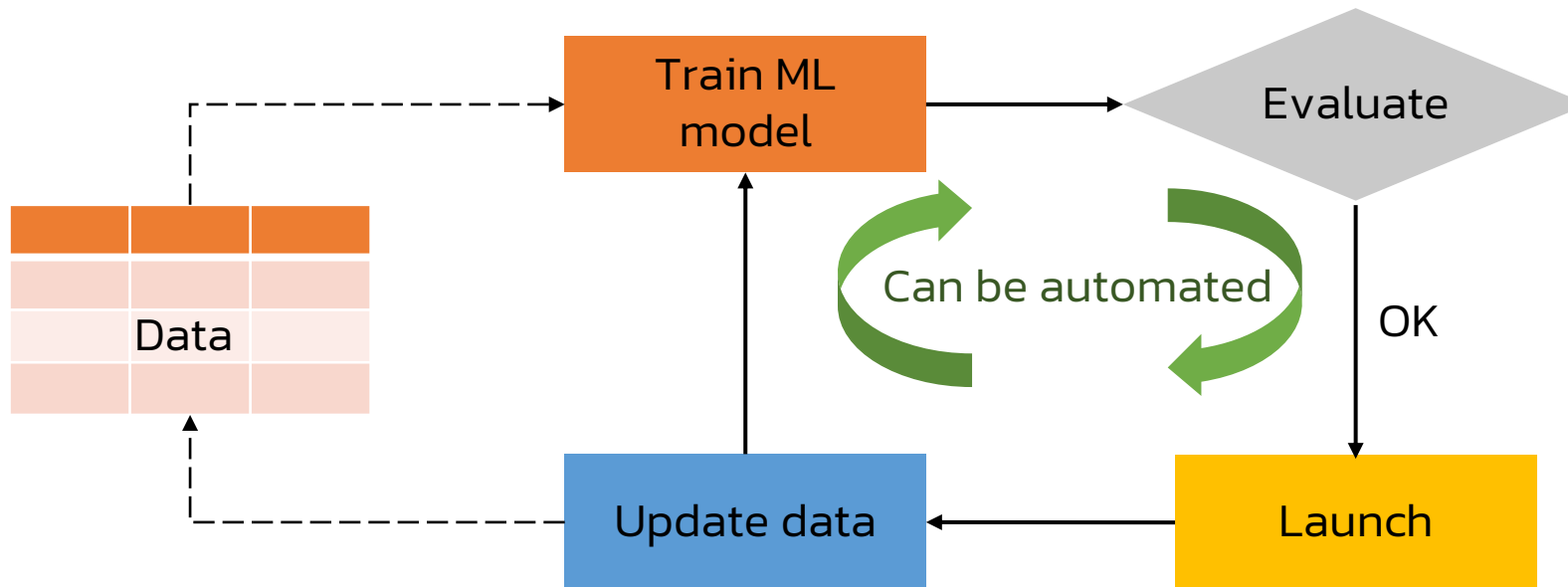
# Machine learning techniques

For example, a spam filter using these techniques might flag potential spam for the recipient to review, preventing automatic blocking.



# Machine learning techniques

If the recipient marks emails from trusted sources as safe, the spam filter learns and adapts its logic to include similar emails from that sender in the future.



# ML – Machine Learning

“Field of study that gives computers the ability to learn without being explicitly programmed.”

Arthur Samuel (1959)



Source:  
(<https://www.forbes.com/sites/gilpress/2021/05/28/on-thinking-machines-machine-learning-and-how-ai-took-over-statistics/>)

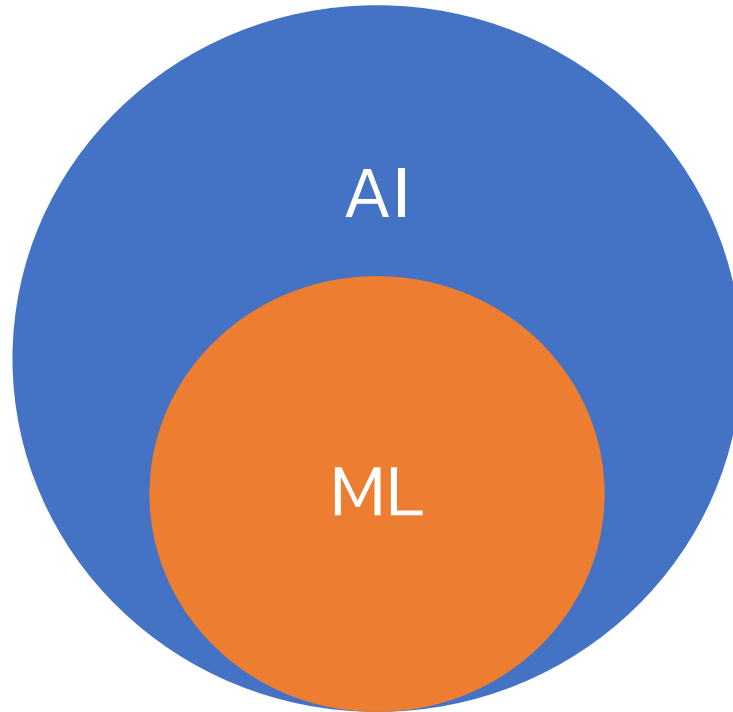


# ML – Machine Learning

It is a subset of AI focused on developing computer programs that can analyze data to make decisions or predictions.

It doesn't have the limitations of rule-based techniques.

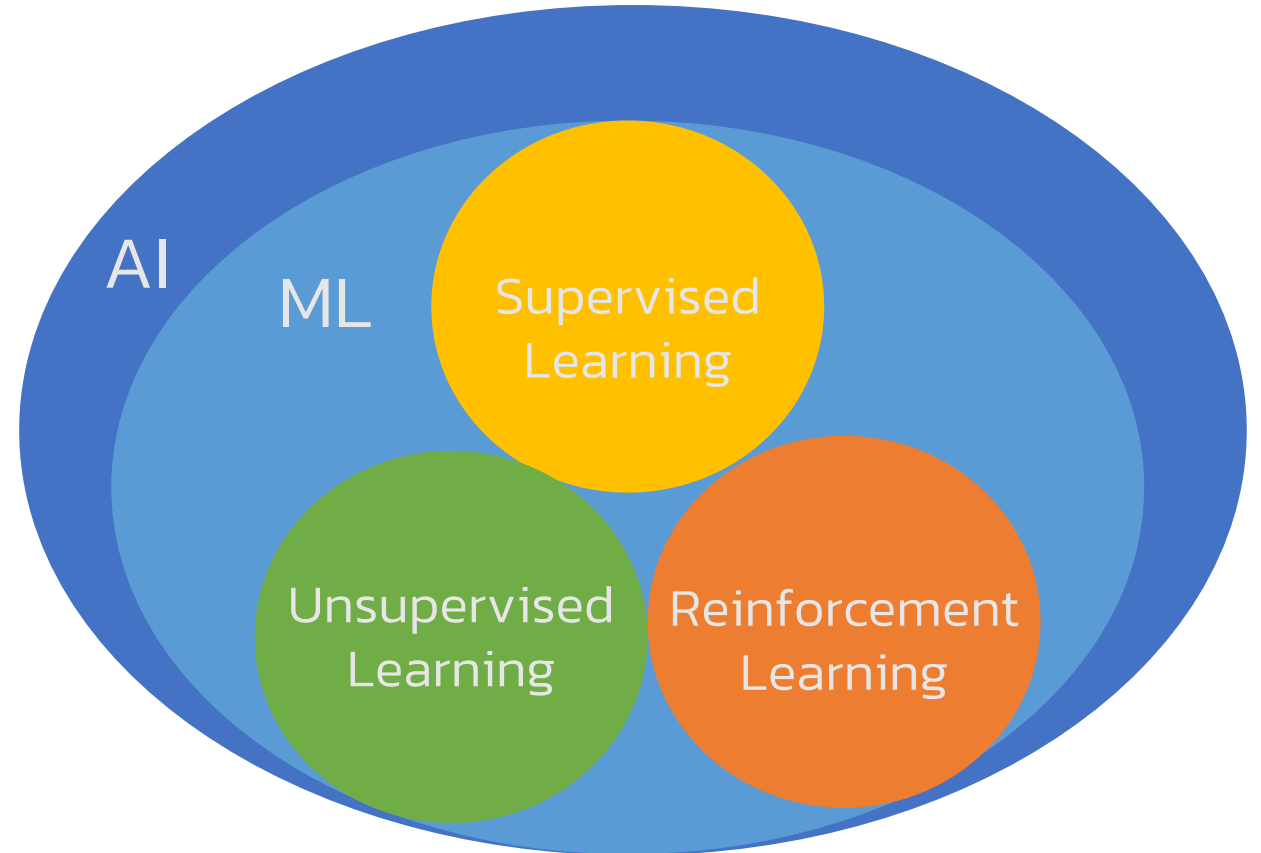
# ML – Machine Learning



# ML – Machine Learning

There are three common approaches to training ML programs.

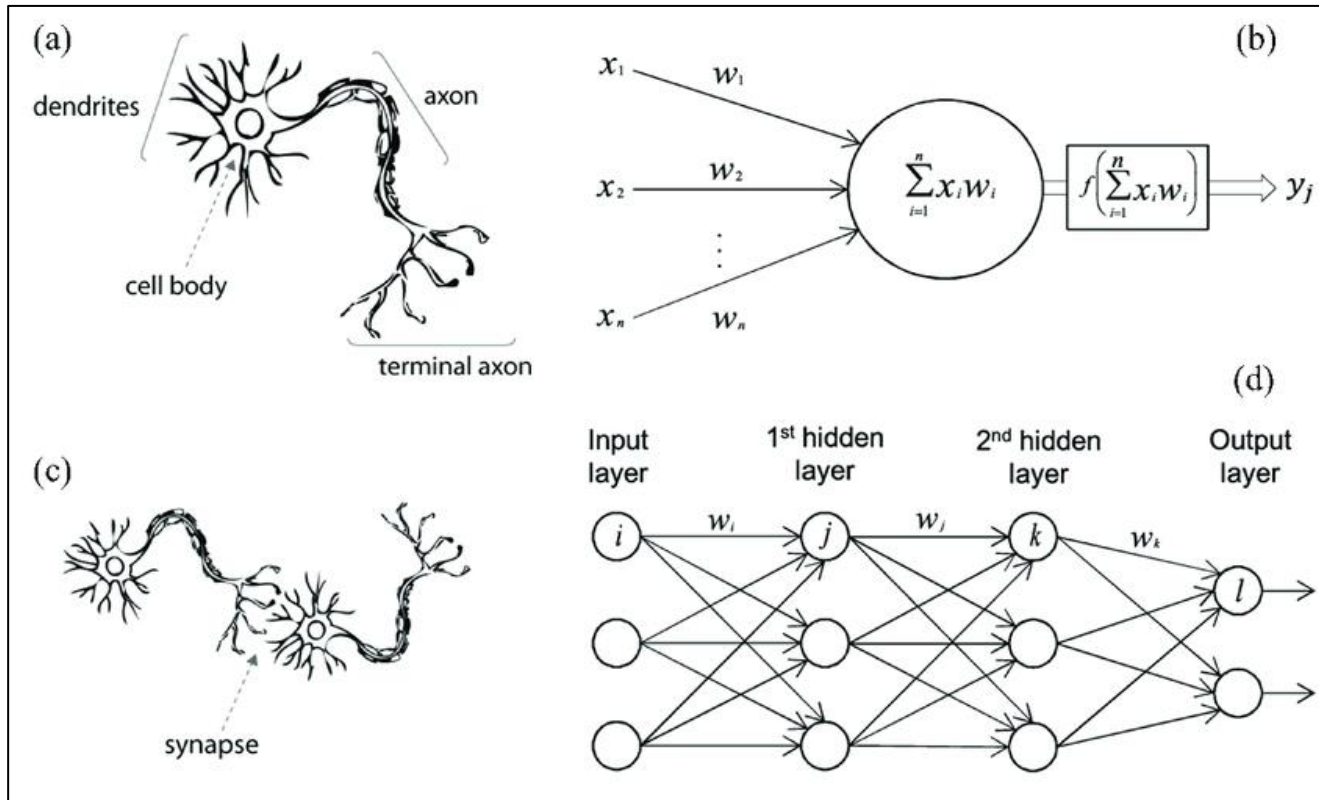
1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning



# DL – Deep Learning

It is a type of machine learning that uses **artificial neural networks** with multiple layers (hence "deep") to learn from data. These neural networks are inspired by the human brain and can be used to solve a wide variety of problems.

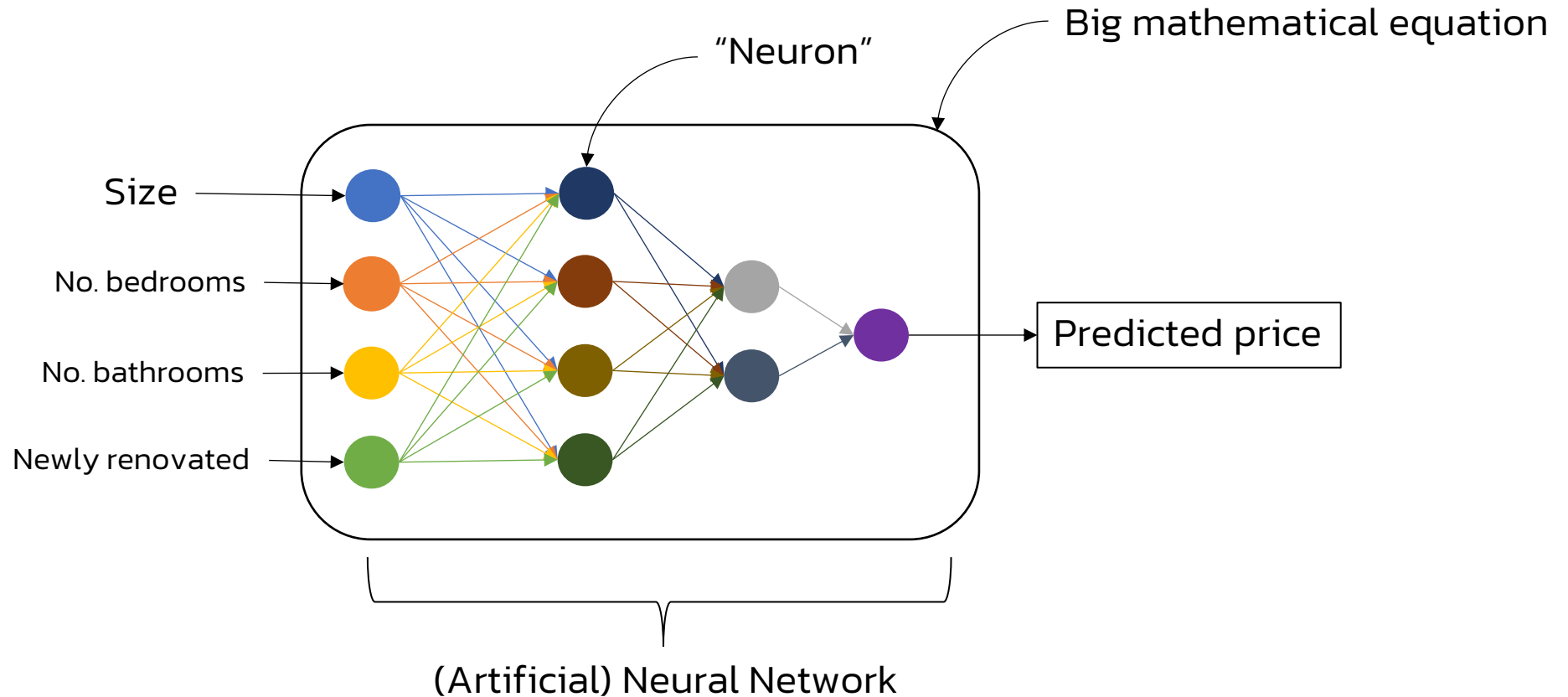
# DL – Deep Learning



Neural Networks were originally inspired by the brain, but the details of how they work are almost completely unrelated to how biological brains work.

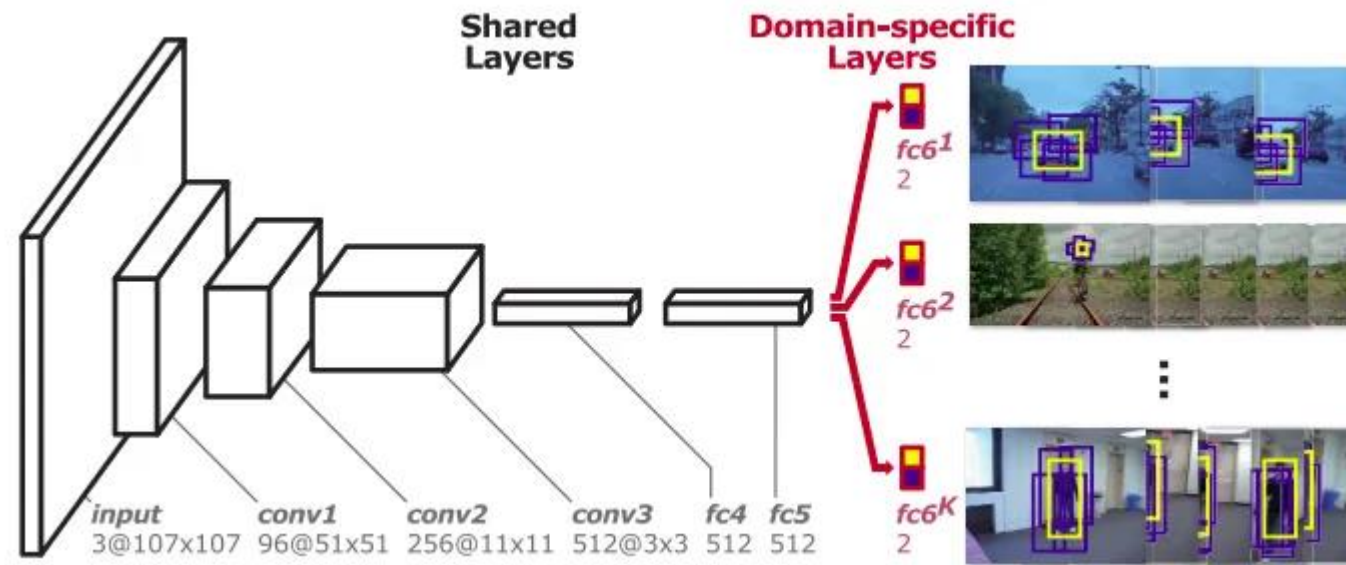
Source:  
(Using a Data Driven Approach to Predict Waves Generated by Gravity Driven Mass Flows – Scientific Figure on ResearchGate. Available from: [https://www.researchgate.net/figure/A-biological-neuron-in-comparison-to-an-artificial-neural-network-a-human-neuron-b\\_fig2\\_339446790](https://www.researchgate.net/figure/A-biological-neuron-in-comparison-to-an-artificial-neural-network-a-human-neuron-b_fig2_339446790) [accessed 4 Jan 2025])

# DL – Deep Learning



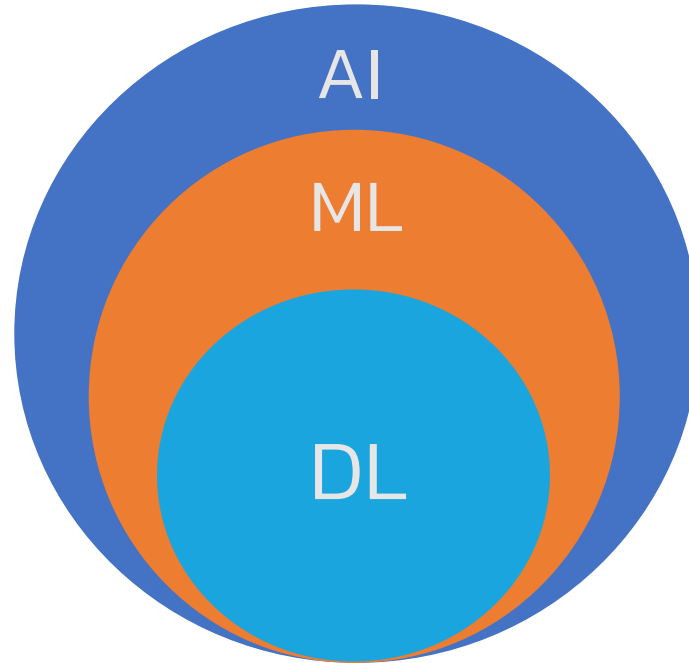
# DL – Deep Learning

## Convolution Neural Network



Source: (<https://medium.com/@r.rohit5557/mdnet-demystified-learning-multi-domain-convolutional-neural-networks-for-visual-tracking-2bb3b7948f61>)

# DL – Deep Learning





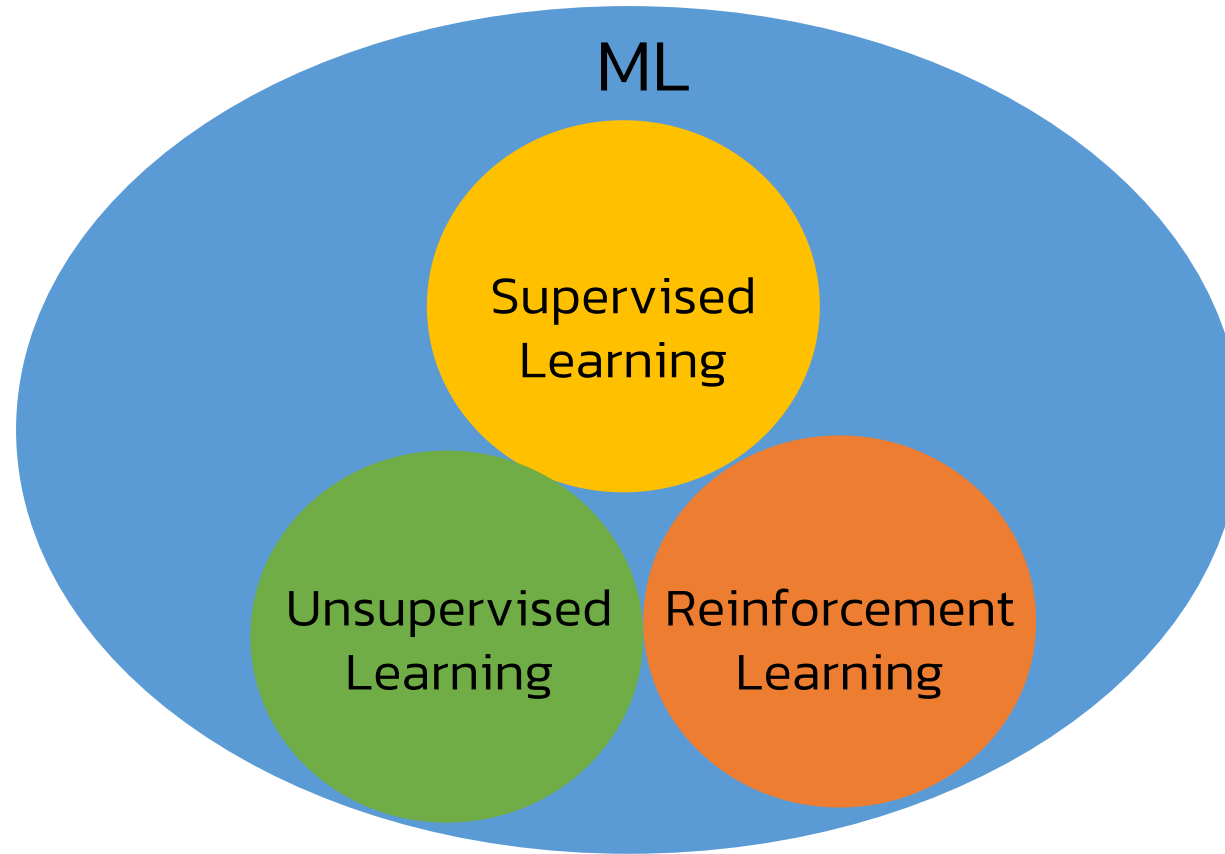
# Data Science

It is a broad field that encompasses the processes of **collecting, cleaning, analyzing, and visualizing data** to extract meaningful insights and inform decision-making. It involves a multidisciplinary approach, drawing from statistics, mathematics, computer science, and domain expertise.

# Data Science vs Machine Learning

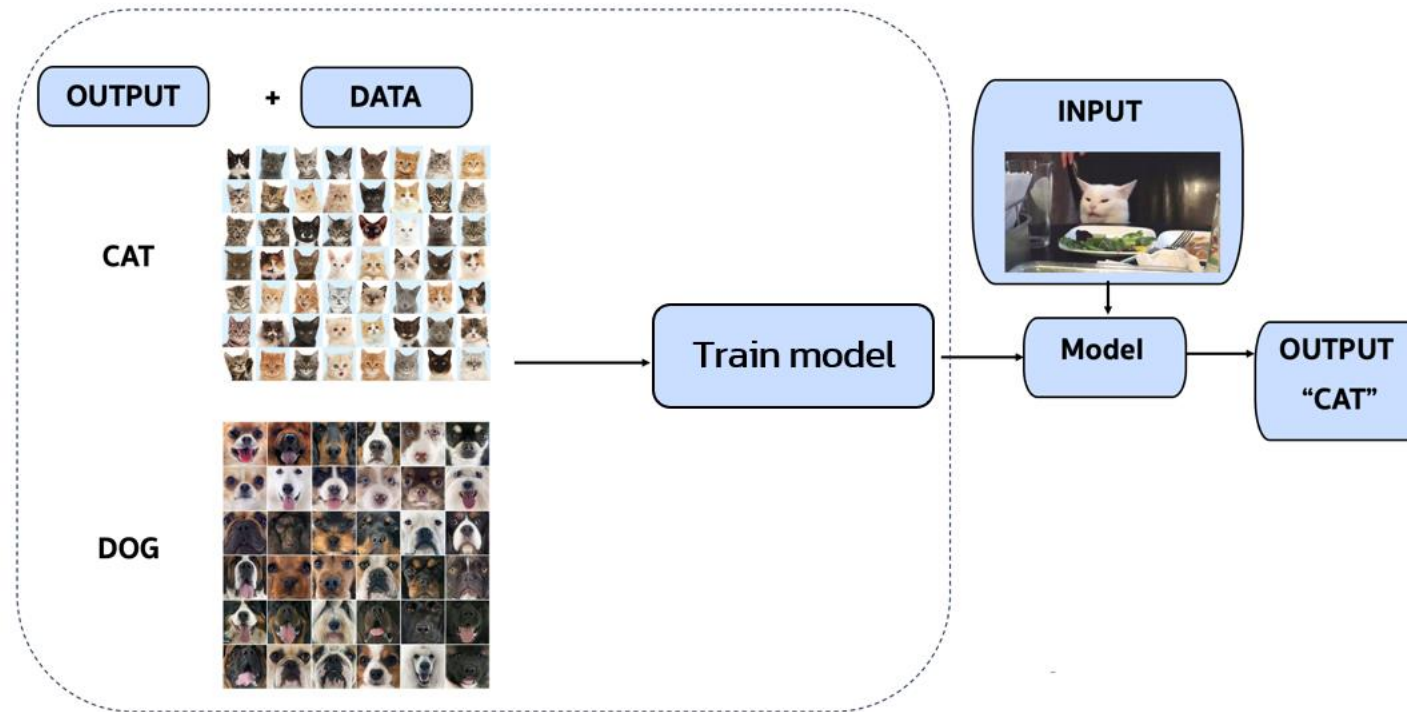
Data science	Machine learning
It is like a toolbox containing various tools, including machine learning.	It is a specific tool within the data science toolbox that is used for building predictive models.

# Types of Machine Learning



# Supervised Learning

Learns from being given “right answer” or “labeled data”



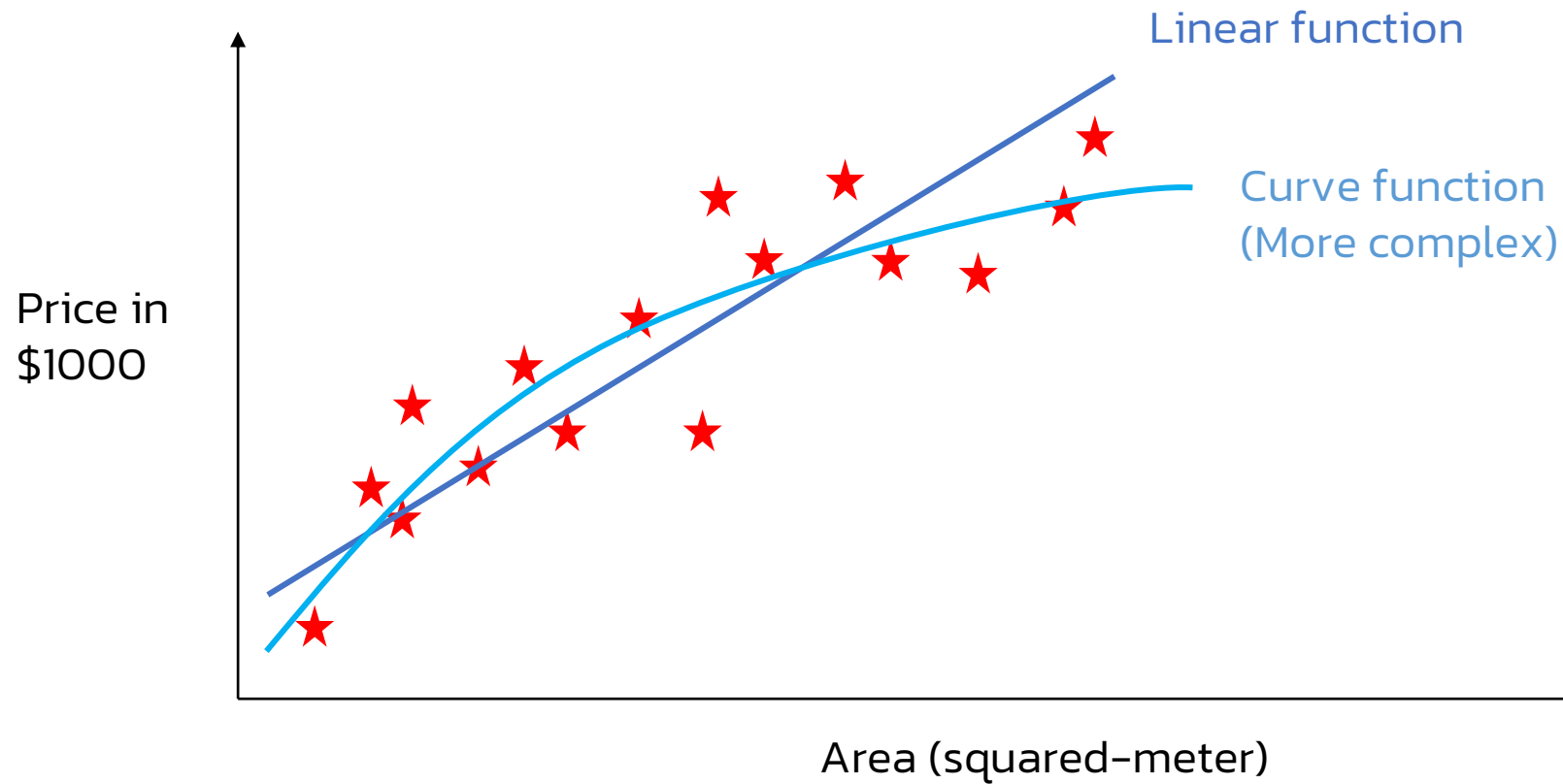
# Supervised Learning

Learns from being given “right answer”

Regression	Classification
Predict a number infinitely many possible outputs	Predict categories Small number of possible outputs

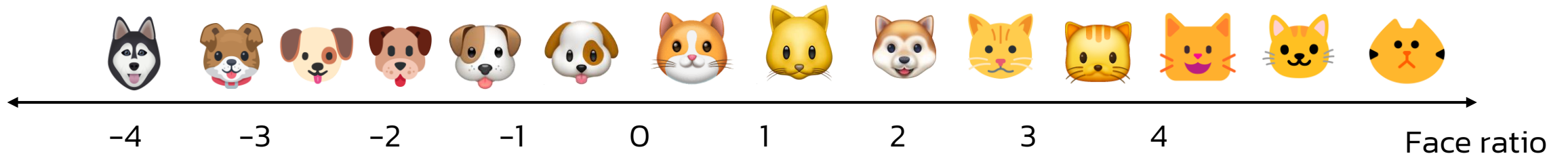
# Supervised Learning

Regression : Housing price prediction



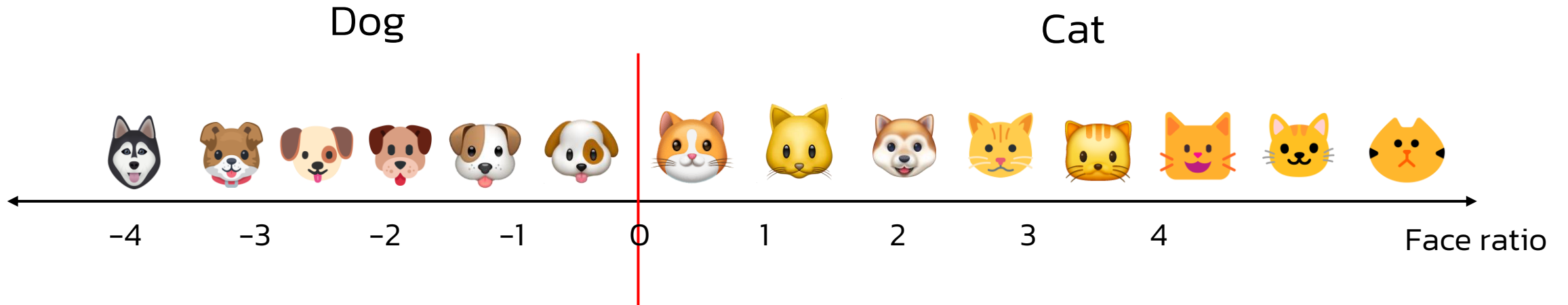
# Supervised Learning

**Classification** : Classify Cat or Dog



# Supervised Learning

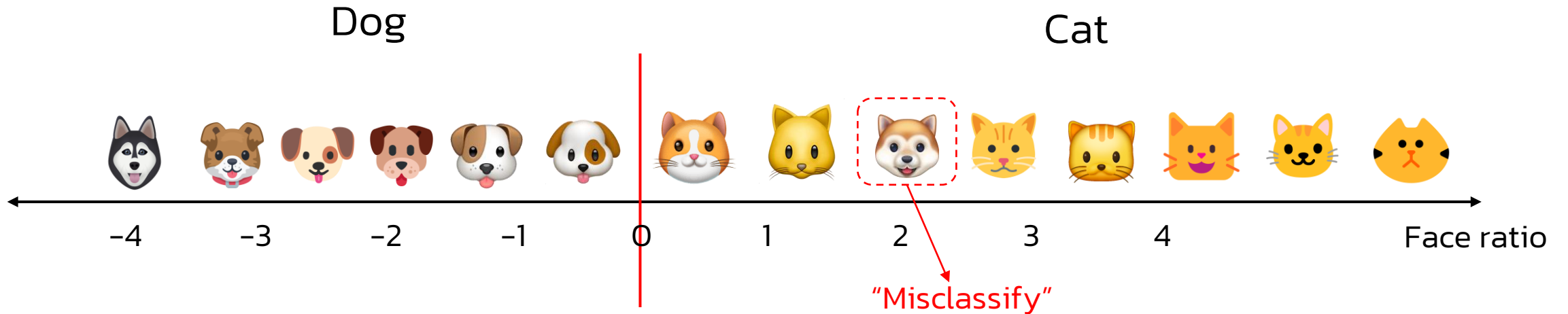
**Classification** : Classify Cat or Dog





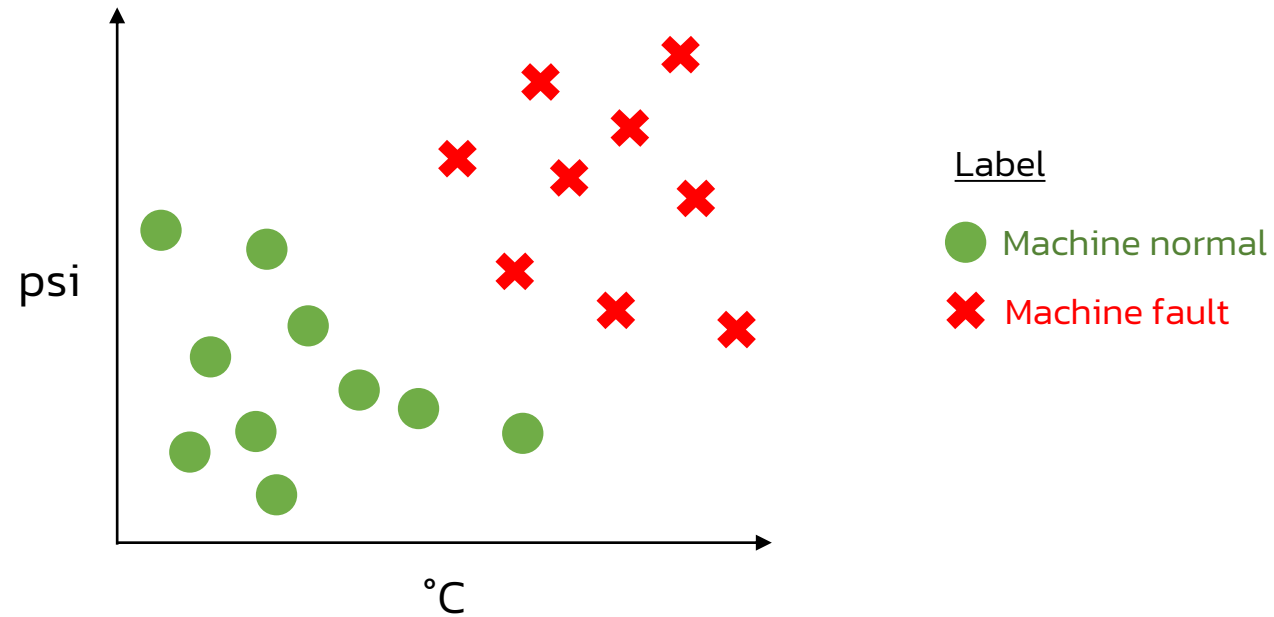
# Supervised Learning

**Classification** : Classify Cat or Dog



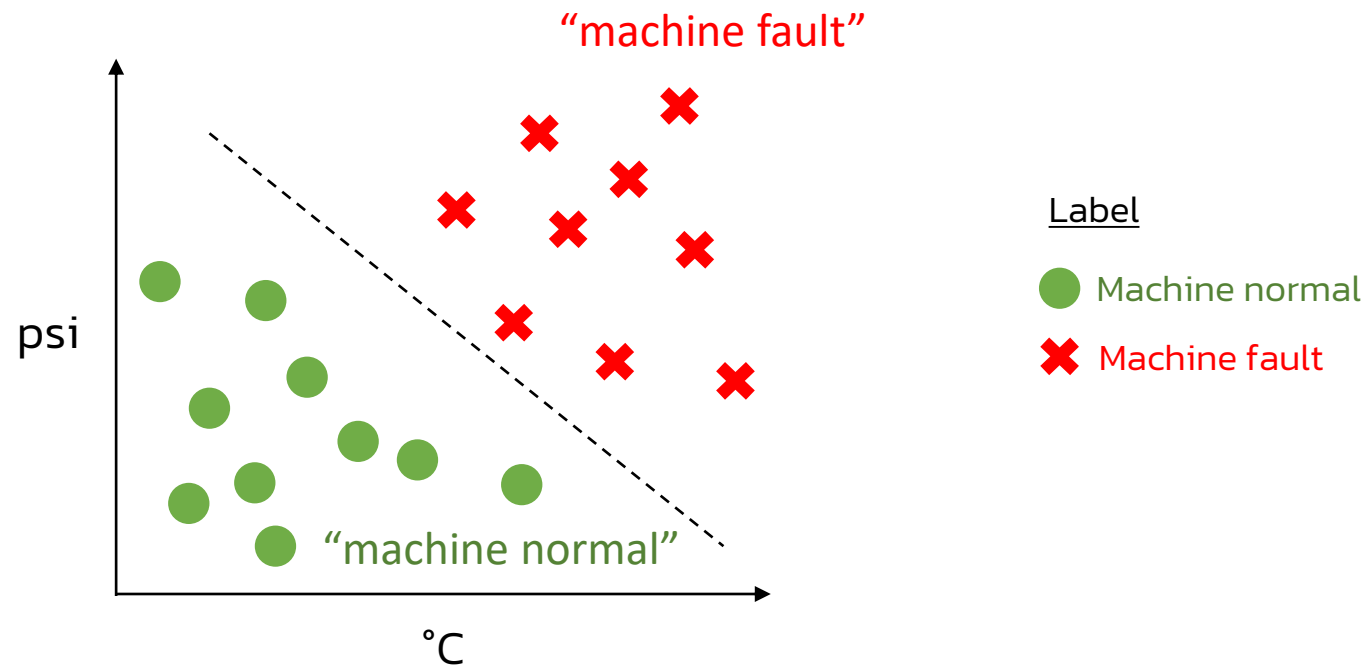
# Supervised Learning

**Classification** : Machine fault prediction



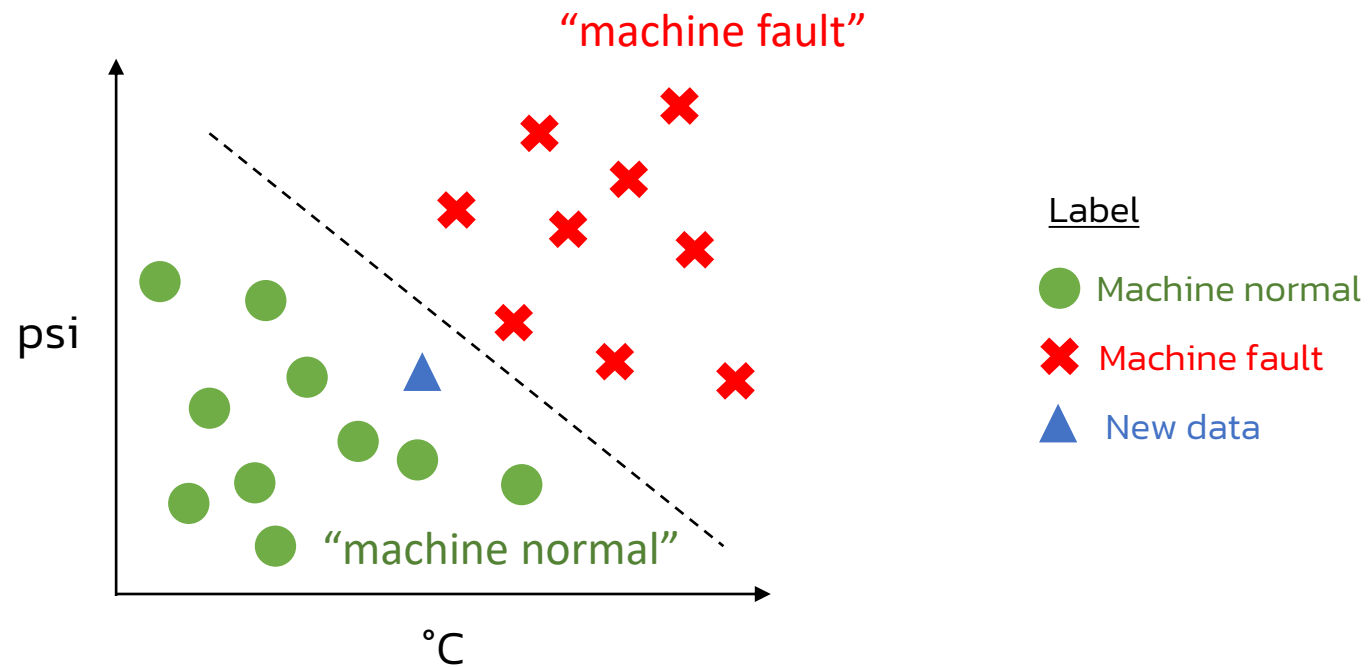
# Supervised Learning

Classification : Machine fault prediction



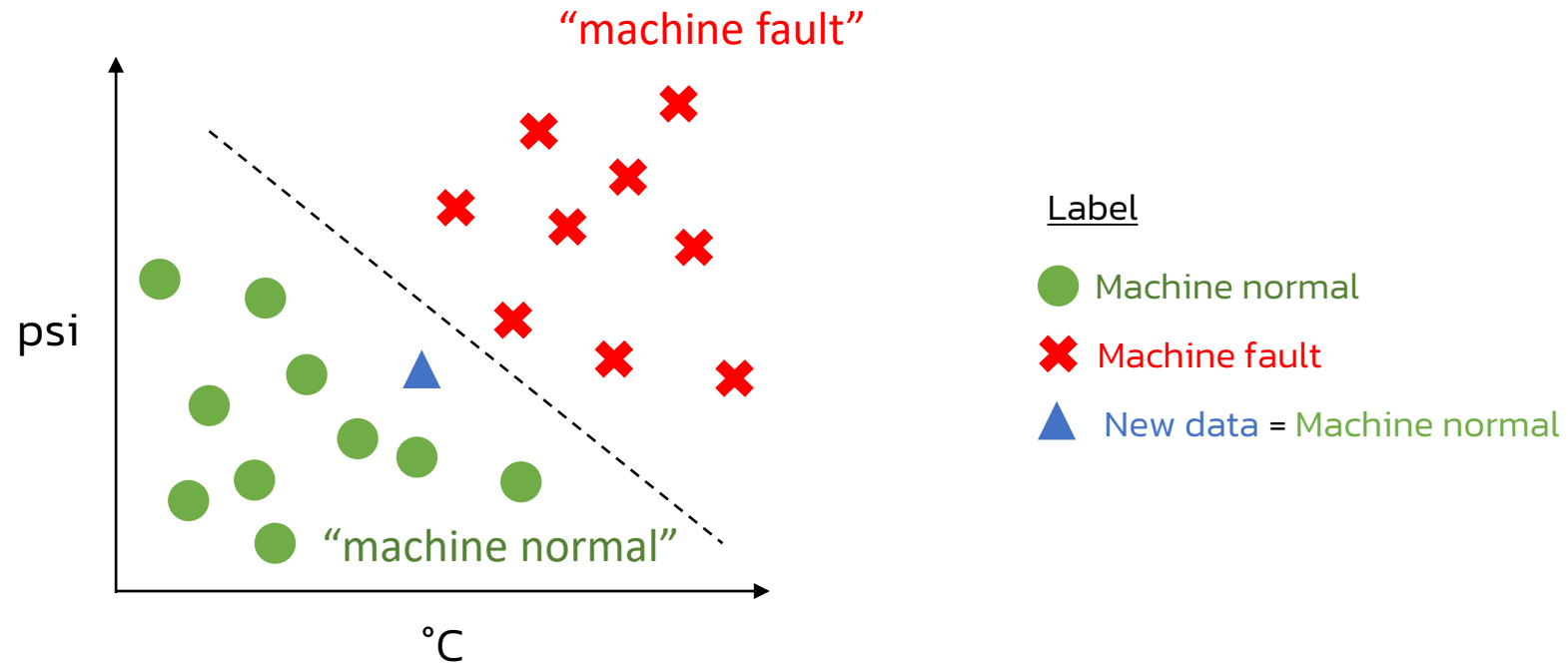
# Supervised Learning

Classification : Machine fault prediction



# Supervised Learning

Classification : Machine fault prediction



# Activity 2 : Supervised Learning applications

Please write words below to complete this table.

Self-driving car  
Spam filtering

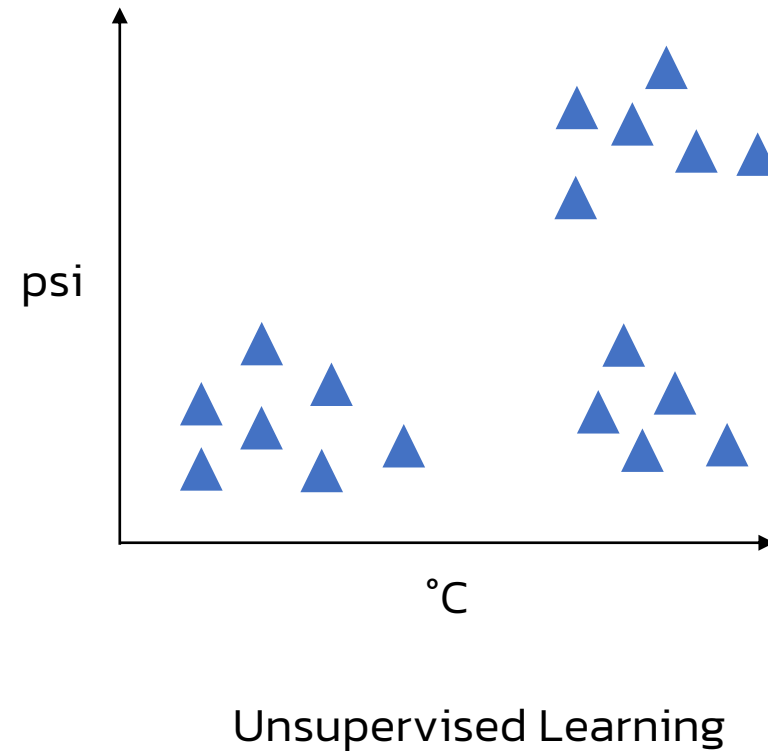
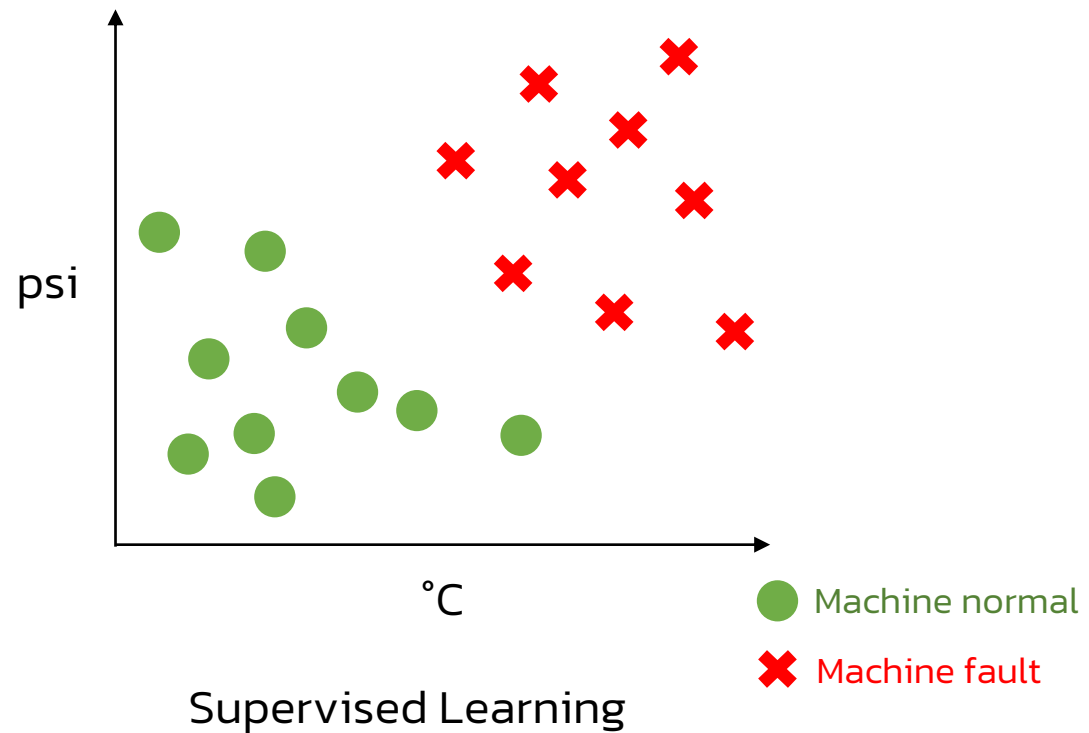
Visual inspection  
Machine translation

Speech recognition  
Online Advertisement

Input (x)	Output (y)	Applications
E-mail	Spam (Yes/NO)	
Audio	Text	
Thai	English	
Advertisement	Click (Yes/No)	
Image and sensor data	Position of Other cars	
Image data	Defect (Yes/No)	

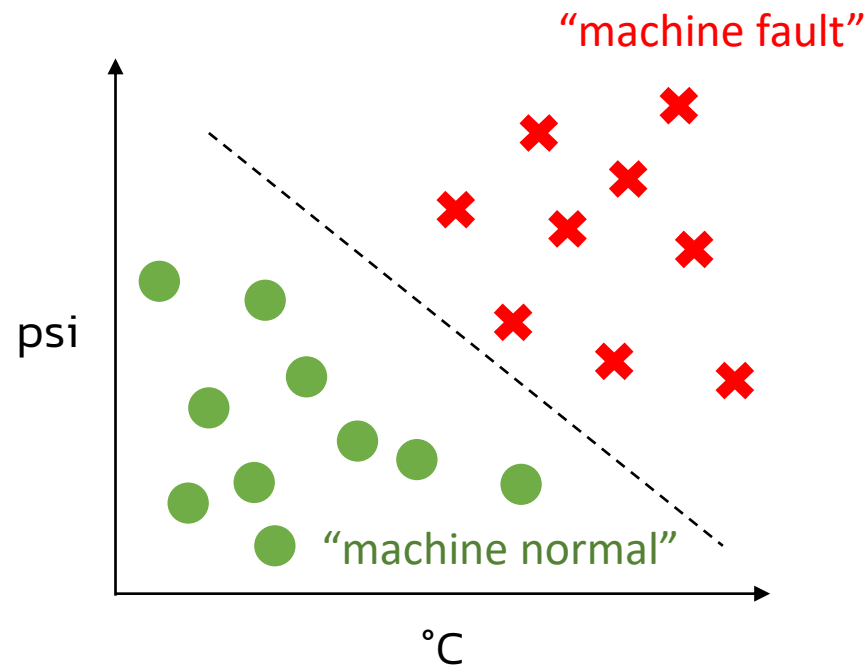
# Unsupervised Learning

Find something interesting in **unlabeled** data

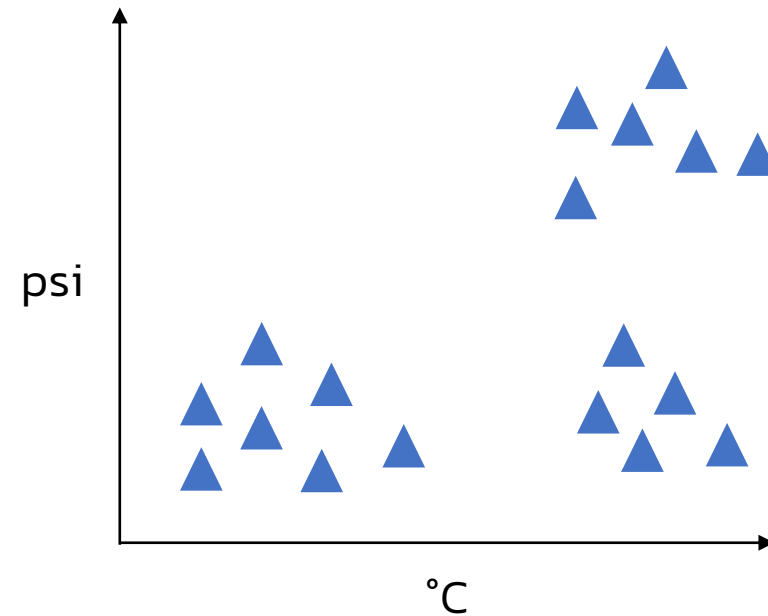


# Unsupervised Learning

Find something interesting in **unlabeled** data



Supervised Learning

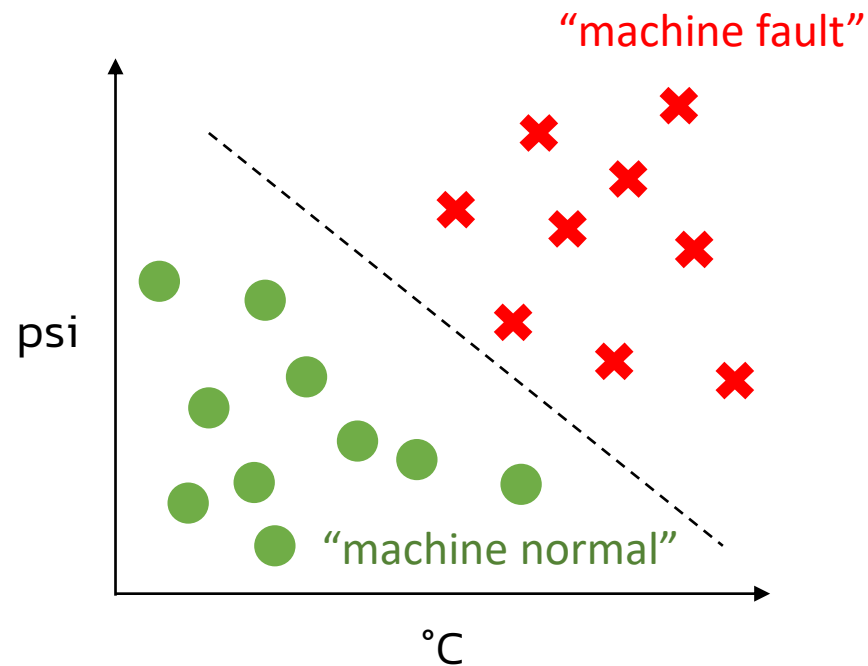


Unsupervised Learning

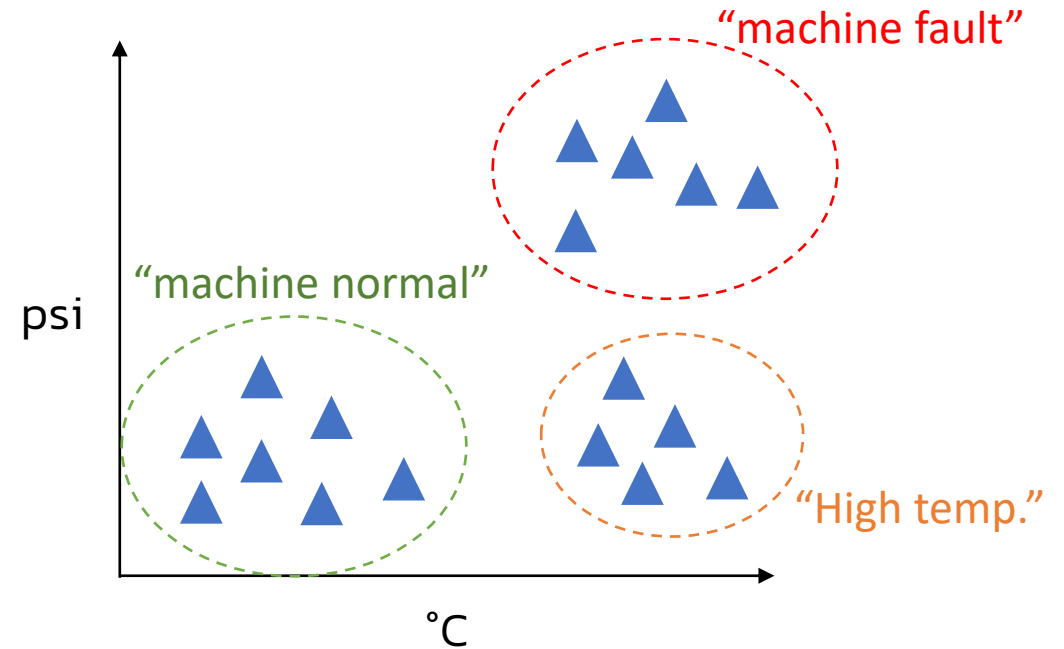


# Unsupervised Learning

Find something interesting in **unlabeled** data



Supervised Learning



Unsupervised Learning

# Reinforcement Learning

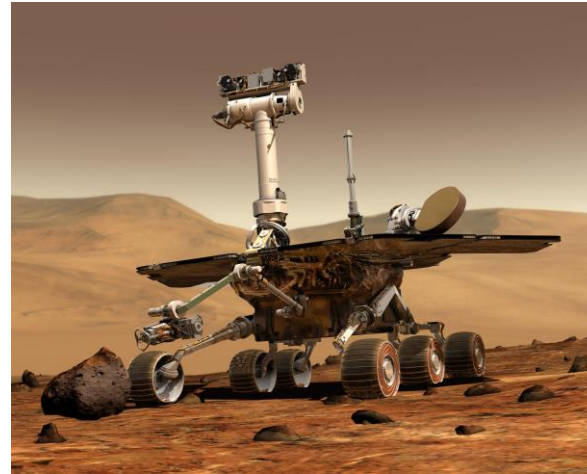
It is a type of machine learning where an agent learns to make decisions by interacting with an environment and receiving feedback in the form of rewards or penalties.

Key concepts:

Agent	The decision-maker or learner.
Environment	The world the agent interacts with.
State	The current situation or condition of the environment.
Action	The choices the agent can make.
Reward	A signal indicating how good or bad an action was.
Policy	The strategy the agent uses to select actions.

# Reinforcement Learning

"Mars Rover"



Source:  
(<https://www.jpl.nasa.gov/missions/mars-exploration-rover-opportunity-mer/>)

Environment : Mars




Terminal state

Action

Terminal state






Reward

State

				
100	0	0	0	40
1	2	3	4	6

# Reinforcement Learning

Find a policy ( $\pi$ ) that tell you what action ( $a = \pi(s)$ ) to take every state ( $s$ ) so as to maximize the return.

Action	100			  	40
Reward	100	0	0	0	40

State-action value function or "Q-function"

$$Q(s, a) = R(s) + \gamma(\max_{a'} Q(s', a'))$$

Reward you get  
Right away
Return from behaving  
optimally starting from state  $s'$

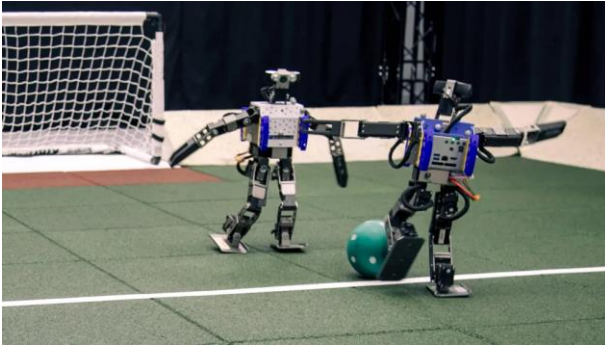
$s$  = current state  
 $a$  = current action  
 $s'$  = state which get after taking action  $a$   
 $a'$  = action which take in state  $s'$   
 $\gamma$  = discount factor  
 $R(s)$  = reward on current state

Source:  
 (Watkins, Chris; Dayan, Peter (1992). "Q-learning". *Machine Learning*. **8** (3–4): 279–292. doi:10.1007/BF00992698. hdl:21.11116/0000-0002-D738-D.)

Bellman equation

# Reinforcement Learning Applications

## Controlling robot

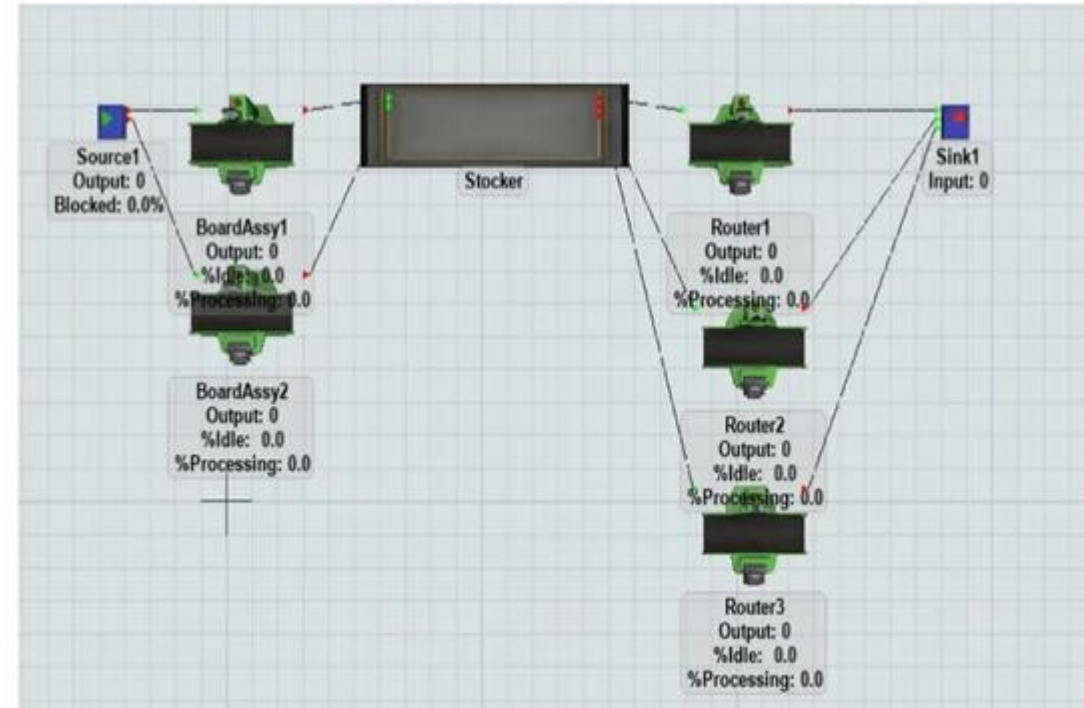


Sources:

(<https://www.sciencenews.org/article/reinforcement-learn-ai-humanoid-robots>)

(<https://ai.stanford.edu/~ang/originalHomepage.html>)

## Factory optimization

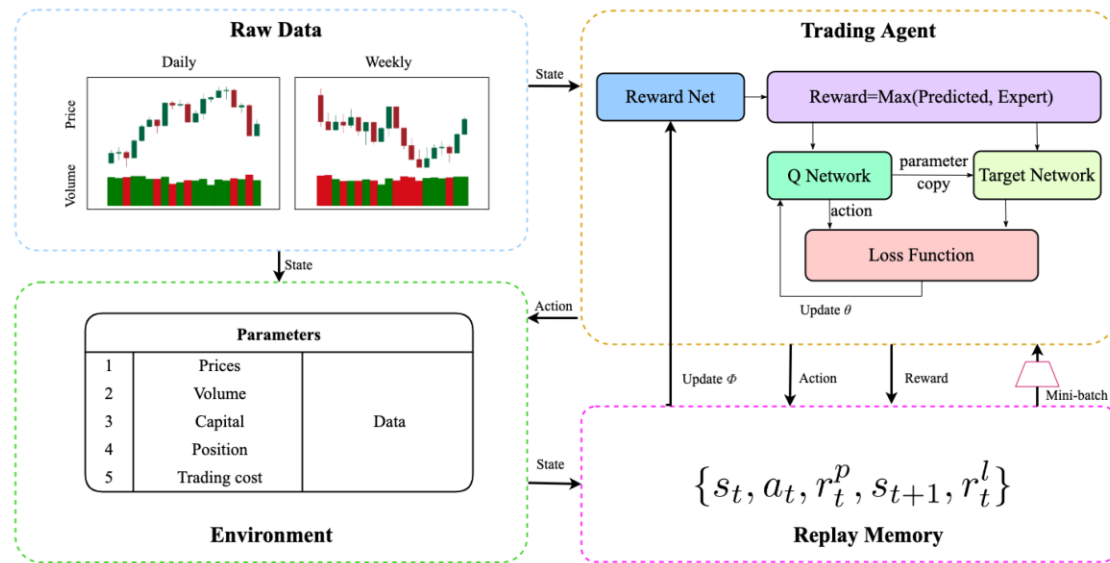


Source:

(Lim, J.-B.; Jeong, J. Factory Simulation of Optimization Techniques Based on Deep Reinforcement Learning for Storage Devices. *Appl. Sci.* **2023**, *13*, 9690. <https://doi.org/10.3390/app13179690>)

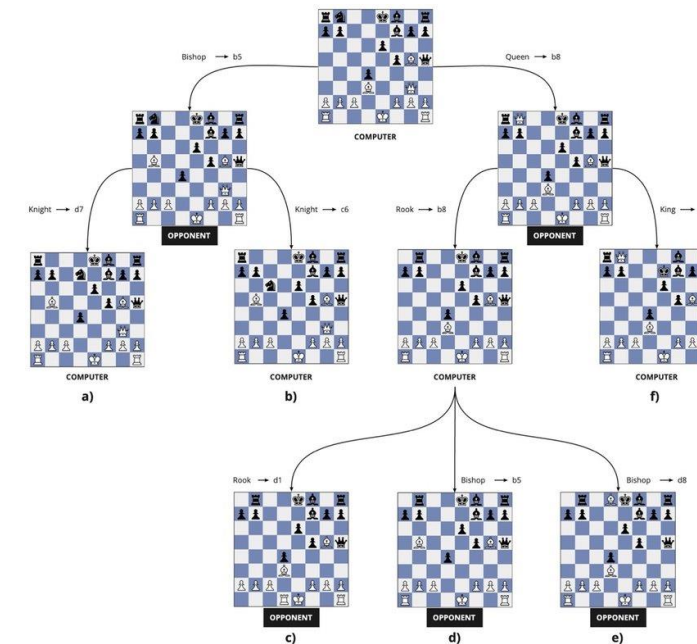
# Reinforcement Learning Applications

## Stock trading



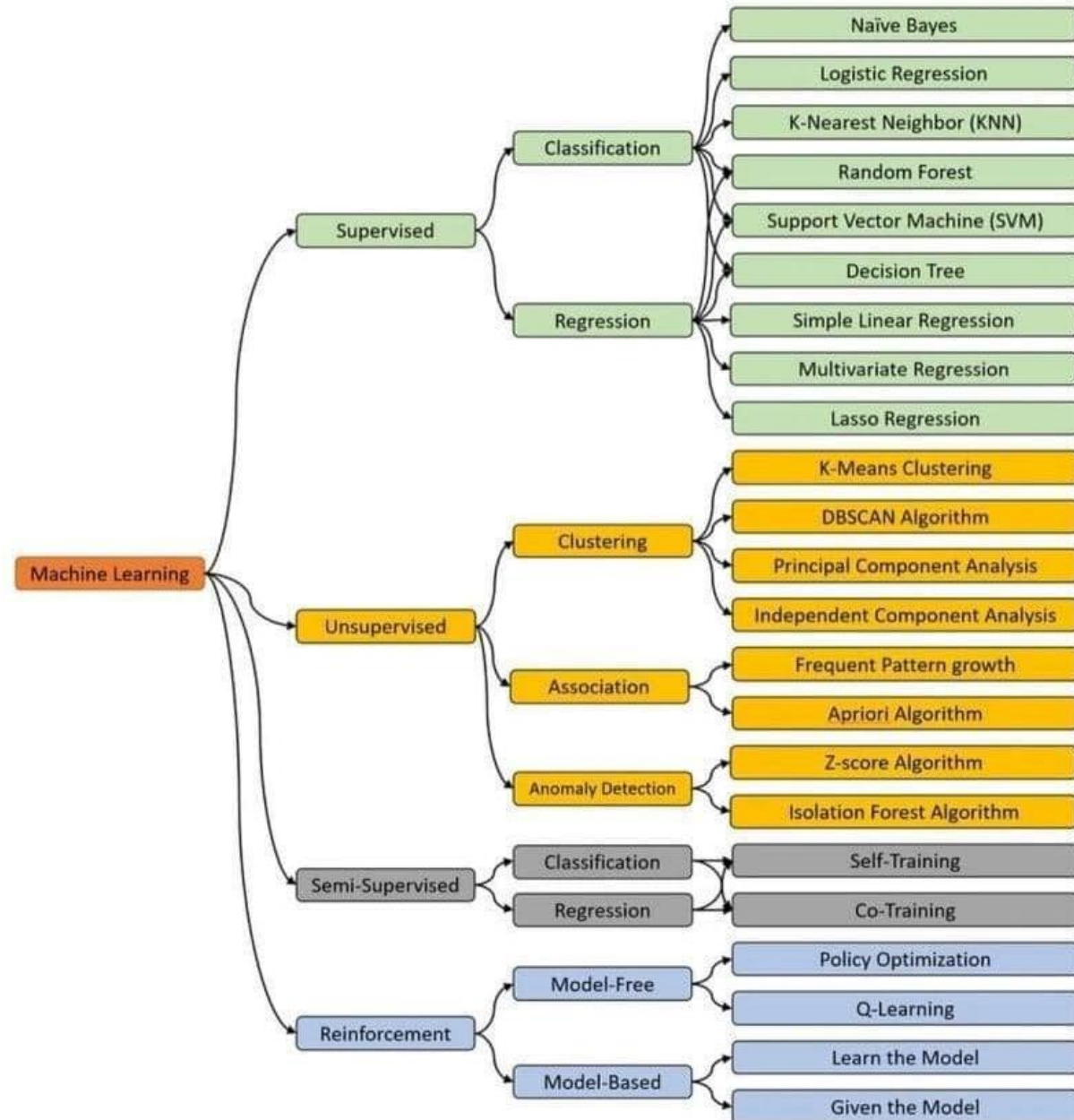
Source:  
 (Huang, Y.; Zhou, C.; Zhang, L.; Lu, X. A Self-Rewarding Mechanism in Deep Reinforcement Learning for Trading Strategy Optimization. *Mathematics* **2024**, *12*, 4020. <https://doi.org/10.3390/math12244020>)

## Playing games

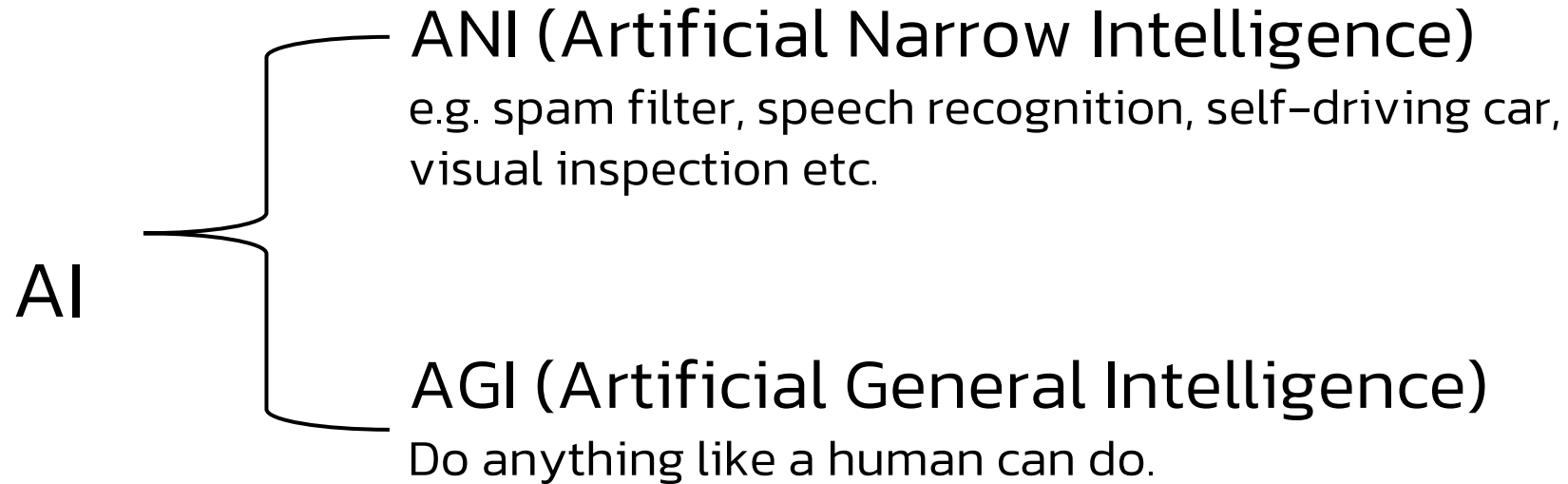


Source:  
 (The Importance of (Exponentially More) Computing Power – Scientific Figure on ResearchGate. Available from: [https://www.researchgate.net/figure/Small-subset-of-a-decision-tree-for-chess\\_fig1\\_361605568](https://www.researchgate.net/figure/Small-subset-of-a-decision-tree-for-chess_fig1_361605568) [accessed 4 Jan 2025])

# Machine Learning models

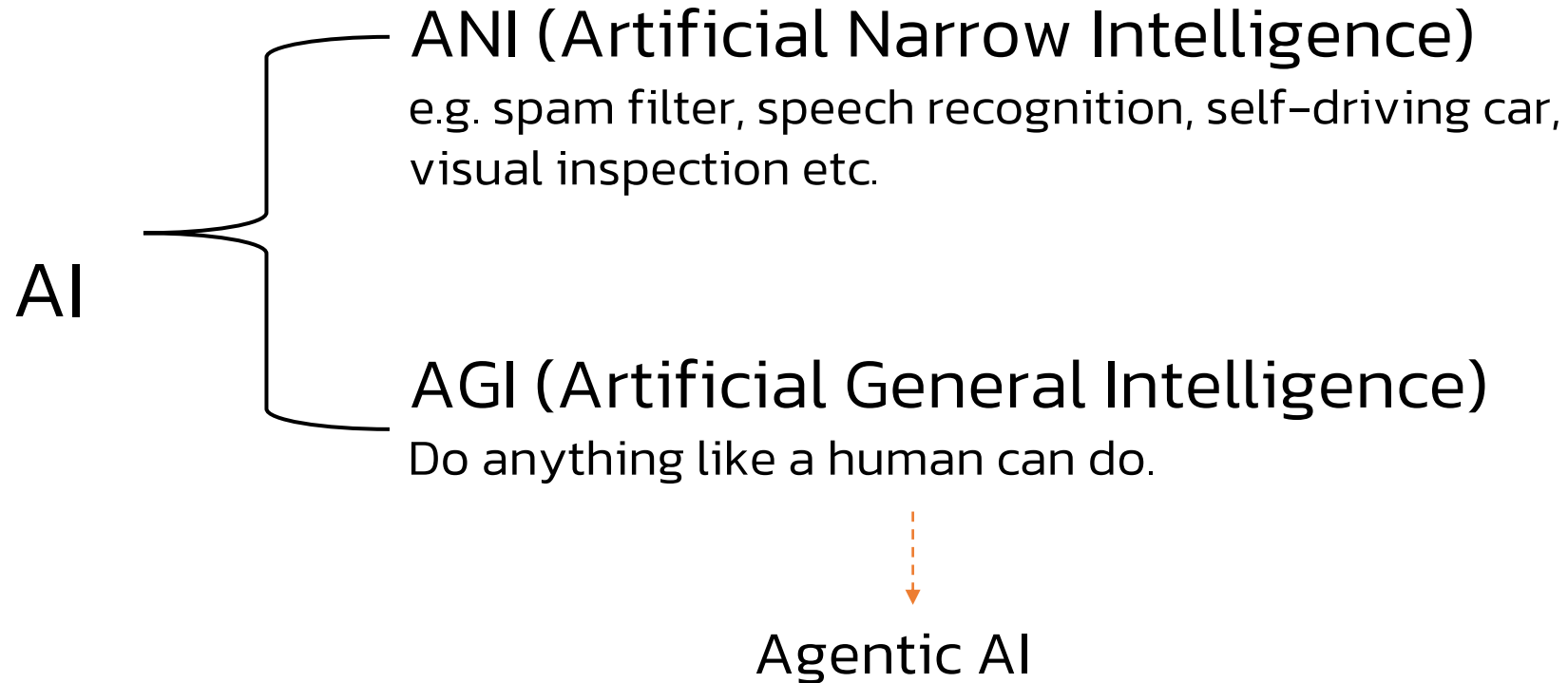


# AI – Artificial Intelligence



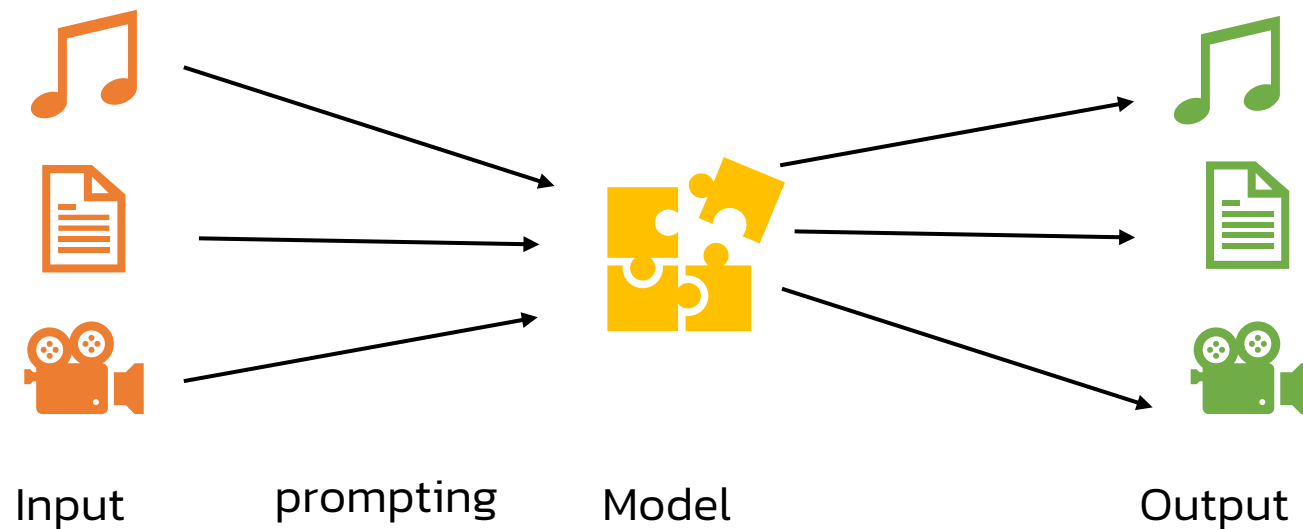


# AI – Artificial Intelligence





















































# Generative AI

It is AI that can generate new content, like text, images, or other media. When users provide input in the form of a **prompt**. (text input that provides instructions to the AI model on how to generate output)



# Generative AI tools

1.  ChatGPT	11.  IIElevenLabs	21.  PhotoRoom	31.  PIXAI	41.  MaxAI.me
2.  Gemini*	12.  Hugging Face	22.  YODAYO	32.  ideogram	42.  Craiyon
3.  character.ai	13.  Leonardo.AI	23.  Clipchamp	33.  invideo AI	43.  OpusClip
4.  liner	14.  Midjourney	24.  runway	34.  Replicate	44.  BLACKBOX AI
5.  QuillBot	15.  SpicyChat	25.  YOU	35.  Playground	45.  CHATPDF
6.  Poe	16.  Gamma	26.  DeepAI	36.  Suno	46.  PIXELCUT
7.  perplexity	17.  Crushon AI	27.  Eightify	37.  Chub.ai	47.  Vectorizer.AI
8.  JanitorAI	18.  cutout.pro	28.  candy.ai	38.  Speechify	48.  DREAMGF
9.  CIVITAI	19.  PIXLR	29.  NightCafe	39.  phind	49.  Photomyne
10.  Claude	20.  VEED.IO	30.  VocalRemover	40.  NovelAI	50.  Otter.ai

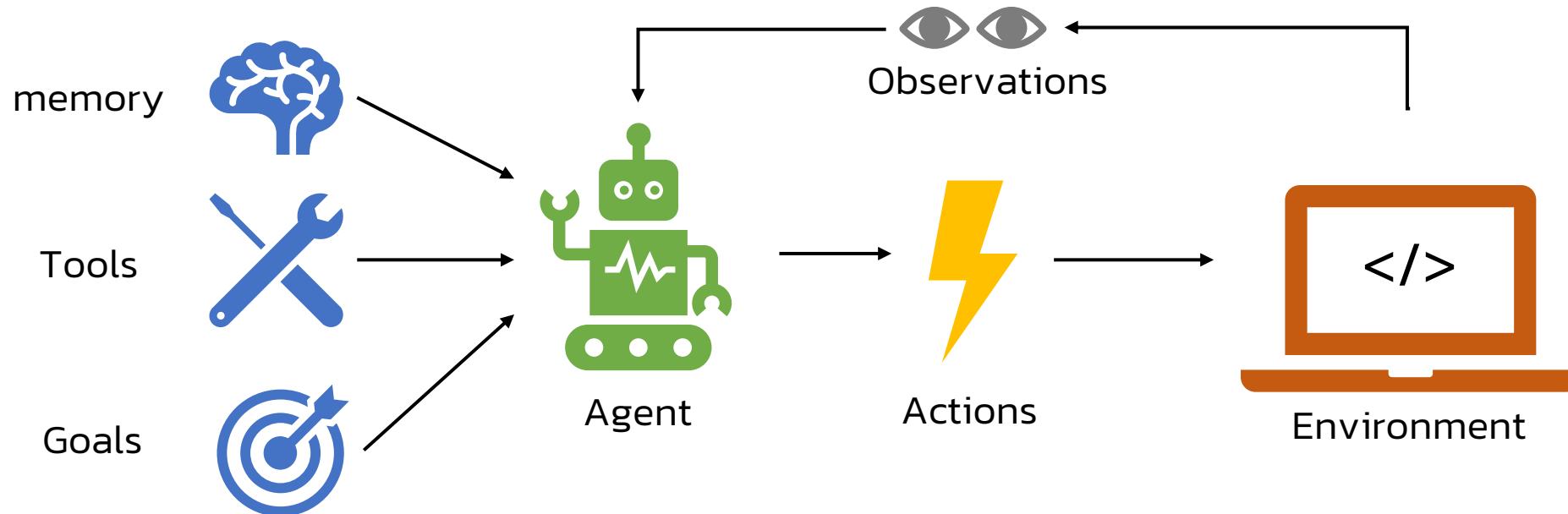
Source: (<https://isarta.com/news/what-are-the-most-used-generative-ai-tools-in-2024/>)

## Activity 3 : Generative AI in daily life

Please tell what you know about Generative AI, and have you ever utilized it before?

# Agentic AI

It is a more recent concept that builds upon AGI. It refers to AI systems that are not only intelligent but also have agency, meaning they can act independently and pursue their own goals.



# Agentic AI

Agentic AI | Updated on Dec 25, 2024

## Agentic AI: 8 Use Cases with Real-life Examples ['25]

By [Cem Dilmegani](#) with Mert Palazoglu

in X

We follow [ethical norms](#) & our [process](#) for objectivity. This research is not funded by any sponsors.

### Table of contents

- 1. AI agents as developers
- 2. AI agents as human-like gaming characters
- 3. AI agents as writers
- 4. AI agents as insurance assistants
- 5. AI agents as human resources (HR) assistants
- 6. AI agents in retail and e-commerce
- 7. AI agents as research assistants
- 8. Building AI agents
- What is agentic AI?

### AGENTIC AI SIMPLIFIED



AIMultiple

A challenge with artificial intelligence (AI), especially [generative AI](#), is that in most complex processes, it requires humans in the loop. Therefore, productivity savings from generative AI remain limited.

[Agentic AI](#) can perform tasks with autonomy, decision-making capabilities, and goal-directed behavior.

Unlike traditional AI, which typically follows predefined instructions, agentic AI can interact with its environment, adapt to new situations, and make decisions. For example, agentic AI can leverage [GenAI](#) and [LLMs](#) can assist employees with writing software code.

## What is agentic AI?

At its core, agentic AI is a type of AI that prioritizes autonomy. This means it can make decisions, take actions, and learn on its own to achieve specific objectives. It's similar to having a virtual assistant who can think, reason, and adapt to changing conditions without constant direction. Agentic AI operates in four major stages:

- **Perception:** It gathers data from its surroundings.
- **Reasoning:** It analyzes this data to identify what it will be used on.
- **Action:** It chooses what to do based on its reasoning.
- **Learning:** It improves and adapts over time, learning from feedback and experience.

## Agentic AI vs artificial intelligence (AI)— key differences

- **Autonomous decision making:** Unlike traditional AI, which is developed for specialized tasks, agentic AI aims to perform complicated goals and workflows with minimal human intervention.

For example, AI agents can not only book your flights but also handle unanticipated delays by rerouting your flight destination.

- **Planning and adapting:** Agentic AI can divide complex tasks into subtasks, reason about them, and make decisions based on the circumstances. This allows it to adjust to changing conditions.

For example, an agentic AI inventory management system can connect to internal warehouse management systems to modify ordering patterns in response to real-time sales data.

Source: (<https://research.aimultiple.com/agentic-ai/>)

# Challenges of AI in industry

1. Data
2. What AI today can and cannot do
3. CPU vs. GPU
4. Brainstorming framework
5. Key steps of an AI project

# (1) Data

## Use and mis-use of data





Don't throw data at an AI and Assume it will be valuable.



# Dataset

Example of table of Data or "Dataset"

Area m-square	No. of rooms	Price (\$1000)
50	2	100
75	3	200
75	4	250
100	3	320
100	4	400
125	5	1000
150	4	1200

Image	Label
	Cat
	Not Cat
	Not Cat
	Cat

# Acquiring Data

- Manual labeling



"Cat"



"Not Cat"



"Not Cat"

- From observing

machine	Temperature (°C)	Pressure (psi)	Machine Fault
A	100	50	N
A	200	50	N
B	100	50	N
B	200	50	Y

- Download from websites/partnerships

# Data is disorganized or “messy”

- Data problems
  - Incorrect label
  - Missing value
- Multiple types of data
  - Number
  - Text
  - Image
  - Audio
- Garbage in, garbage out

machine	Temperature (°C)	Pressure (psi)	Machine Fault
A	100	50	N
A	200	50	N
unknown	250	50	N
B	100	50	N
B	200	1000	Y
C	0.01	50	unknown
C	200	50	N

## (2) What AI today can and cannot do

AI tends to work well when:

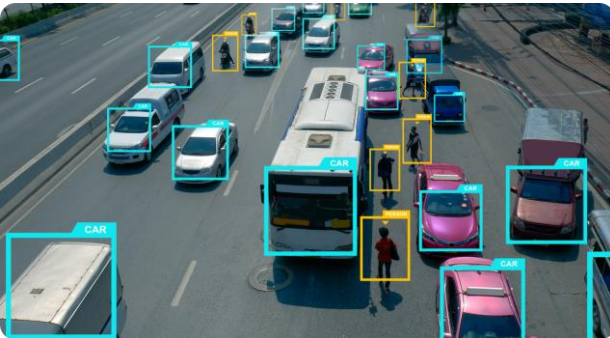
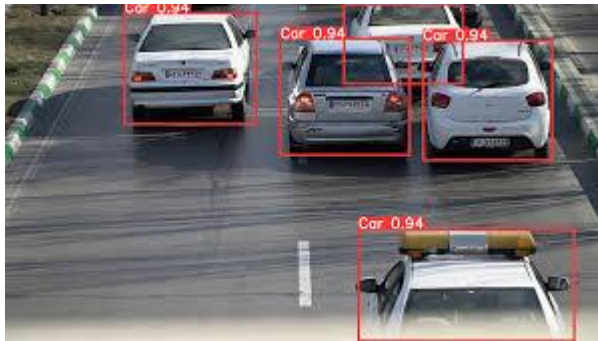
1. Learning a simple concept
2. Lots of data available

AI tends to work poorly when:

1. Learning complex functions from small amounts of data.
2. It is asked to perform on new types of data that it learned from.

# Self-driving car

Can do



Cannot do



stop



hitchhiker



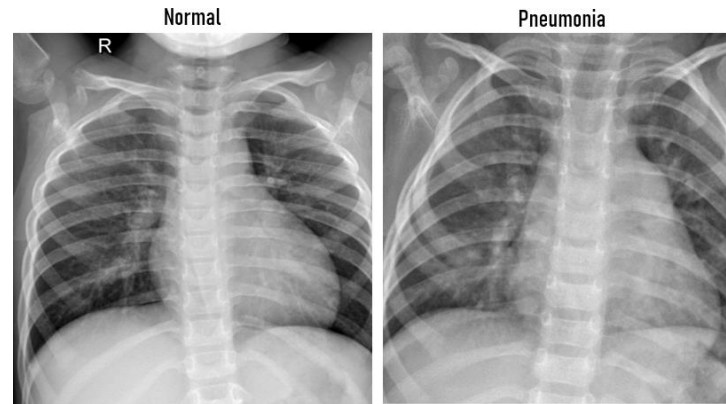
Bike turn left  
signal

Required :

1. More Data
2. High accuracy

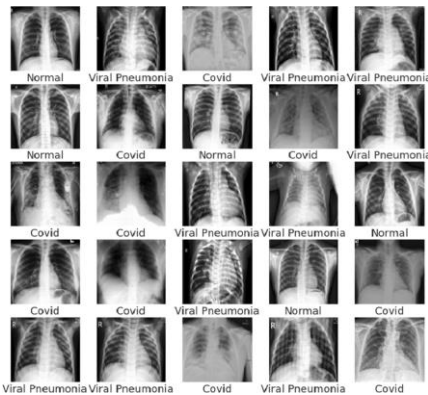
# X-ray diagnosis

Can do



Cannot do

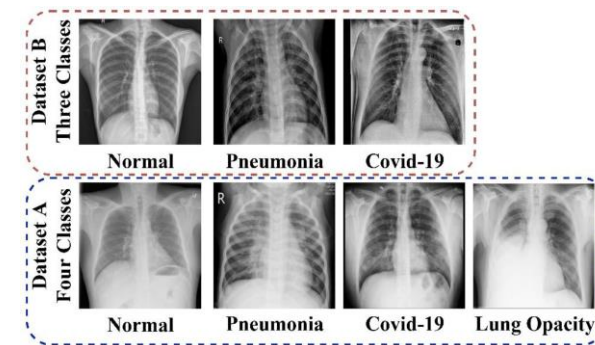
Diagnose pneumonia from  
more than 10,000 labeled images



X 400 sets of images

Source: (Tekerek, A., Al-Rawe, I.A.M. A Novel Approach for Prediction of Lung Disease Using Chest X-ray Images Based on DenseNet and MobileNet. Wireless Pers Commun (2023). <https://doi.org/10.1007/s11277-023-10489-y>)

Diagnose pneumonia from  
7 images



Source: (Ukwuoma, C.C.; Qin, Z.; Heyat, M.B.B.; Akhtar, F.; Smahi, A.; Jackson, J.K.; Furqan Qadri, S.; Muaad, A.Y.; Monday, H.N.; Nneji, G.U. Automated Lung-Related Pneumonia and COVID-19 Detection Based on Novel Feature Extraction Framework and Vision Transformer Approaches Using Chest X-ray Images. Bioengineering 2022, 9, 709. <https://doi.org/10.3390/bioengineering9110709>)

# (3) CPU vs. GPU

CPU : Central Processing Unit (Computer processor)



GPU : Graphics Processing Unit



Cloud vs. On-premises



## (4) Brainstorming framework

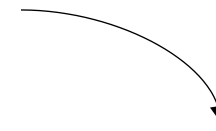
- Think about optimizing tasks rather than automating job.  
E.g. Call center routing, radiologists.
- What are the main drivers of business value?
- What are the main points in your business?

AI Expert



What AI can do?

Domain Expert



Things valuable for your business



# Due diligence on AI project

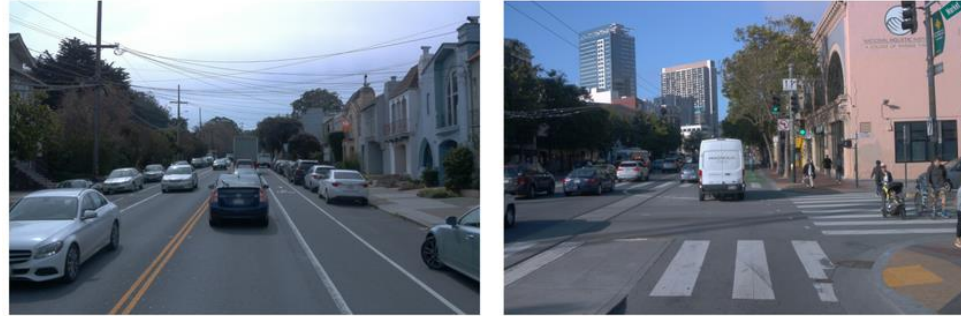
Technical diligence	Business diligence
<ul style="list-style-type: none"><li>• Can AI system meet desired performance?</li><li>• How much data is needed?</li><li>• Engineering timeline</li></ul>	<u>Current business</u> <ul style="list-style-type: none"><li>• Lower costs</li><li>• Increase revenue</li></ul> <u>New business</u> <ul style="list-style-type: none"><li>• Launch new product or business</li></ul>

# (5) Key steps of an AI project

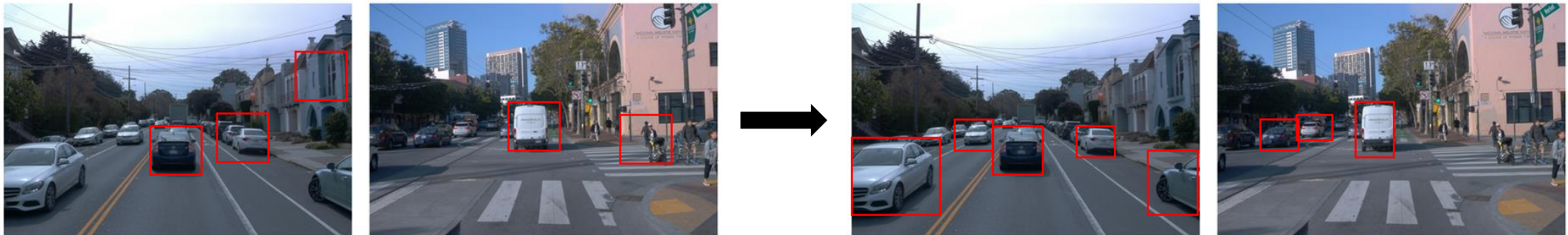
1. Collect data
2. Train model : Iterate many times until good enough
3. Deploy model : Get data back, Maintain or update model

# Self-driving car (Vehicle detection)

1. Collect data



2. Train model : Iterate many times until good enough



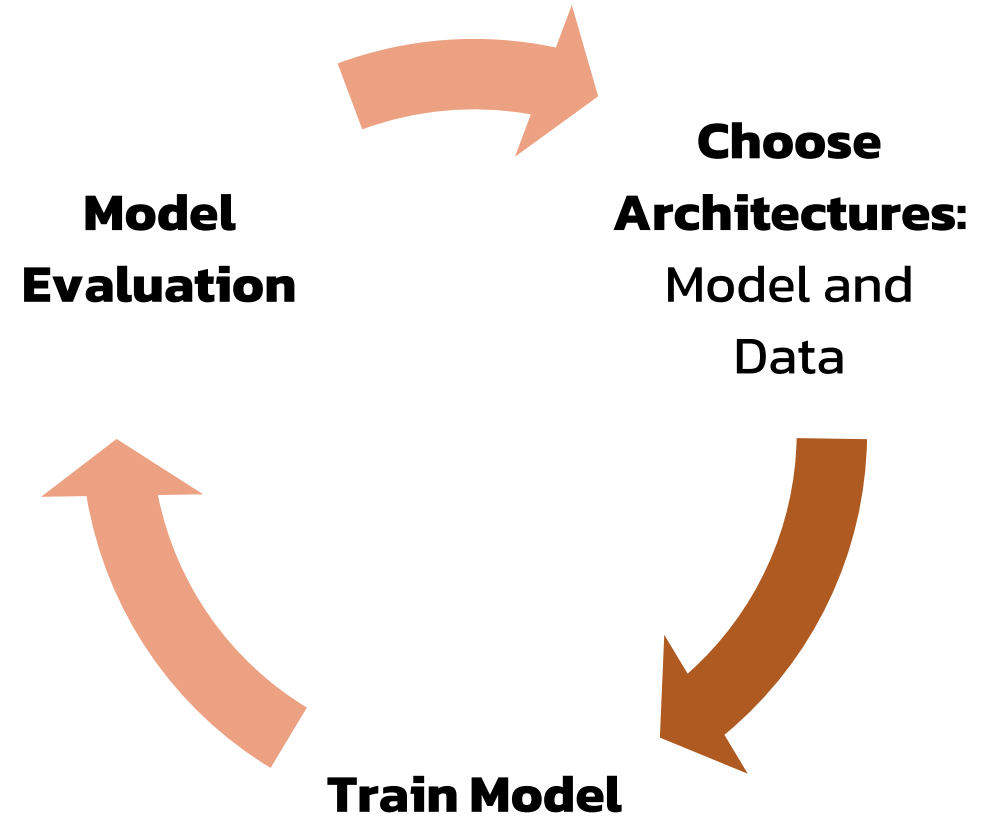
Train 1 time

Train 100 times

3. Deploy model : Get data back, Maintain or update model

# Processes for develop ML Model

1. Define Problem
2. Gathering and Prepare Data
3. Choose Model
4. Train Model
5. Evaluate Model
6. Tuning Model
7. Prediction
8. Deploy Model



# Open-source frameworks

## Machine Learning frameworks

- TensorFlow
- Pytorch
- Keras
- Scikit-learn
- Weka
- MXNet
- CNTK
- Caffe
- PaddlePaddle

## Research publication:

- Arxiv

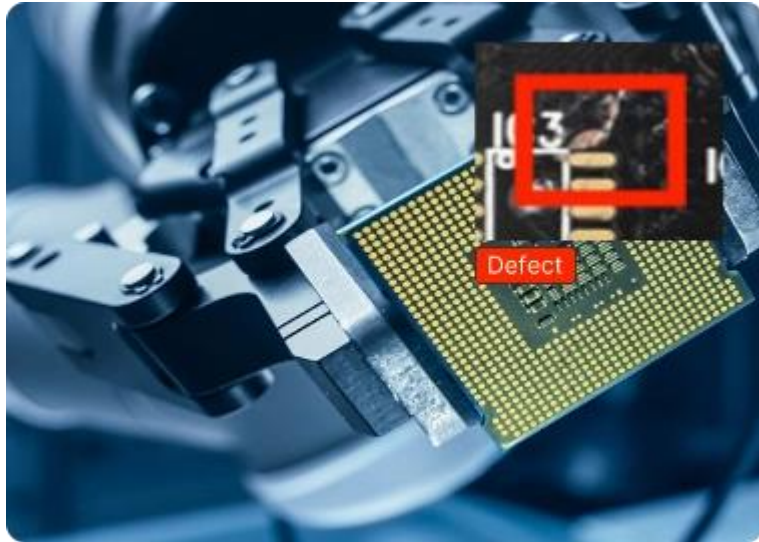
## Open-source repository:

- GitHub

# Examples of AI for industrial inspection

# AI in industry applications

## Integrated Circuit Inspection



Electronics manufacturers must visually inspect integrated circuits (ICs) for defects such as bent, missing, or twisted pins. Not catching these defects — some of which exist on the micron level — can lead to failures, reworks, damaged customer relationships, or worse. Machine vision systems can inspect for such errors, but the large number of defect types poses a challenge to rules-based systems. Deep learning AI software in electronics manufacturing can help classify defects that traditional systems cannot.

# AI in industry applications

## PCB Inspection



Some of the most common artificial intelligence inspection applications in electronics manufacturing involve printed circuit board (PCB) inspection. PCB production involves several different assembly steps with small components. Conducting failure mode and effects analysis for each wafer, pin, joint, and environmental change is a staggering task, and rules-based machine vision makes defining defects difficult or impossible. Deep learning software allows manufacturers to detect and classify these defects.



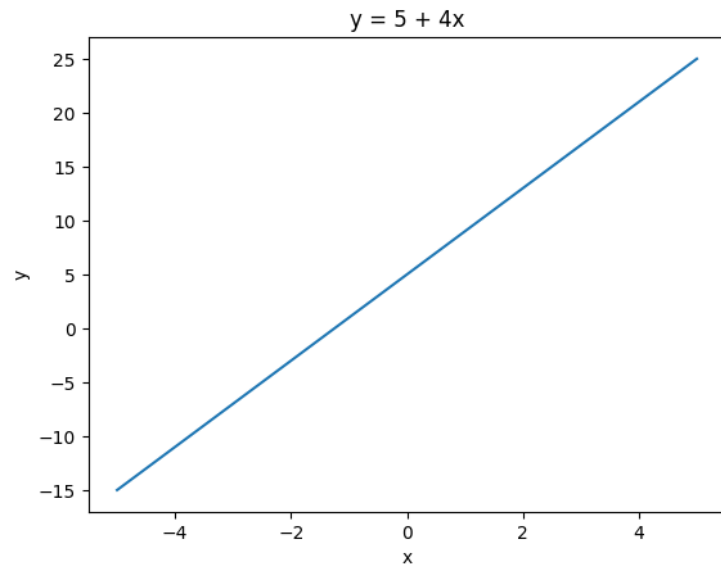
# Activity 4 : AI for your work or business

Please tell about your ideas?

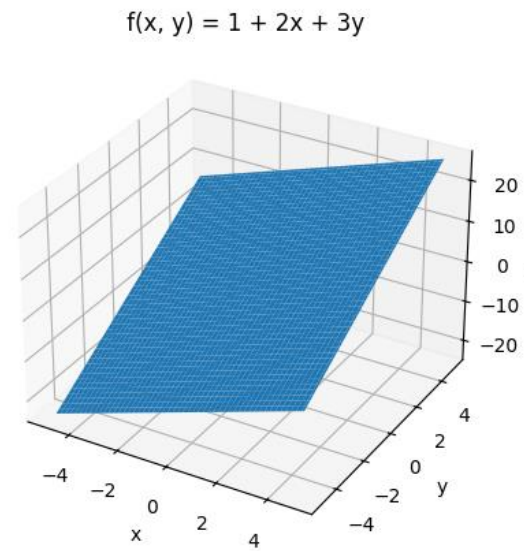
# Basic Math concepts for AI

- Linear algebra
- Calculus
- Probability and Statistics

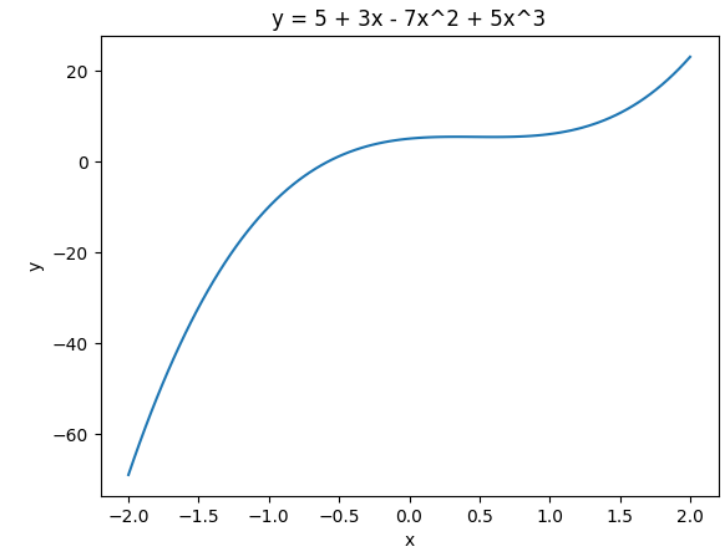
# (1) Linear Algebra



Linear equation



Multivariable equation



Polynomial equation  
(Non-linear equation)

# Linear equation

```
import numpy as np
import matplotlib.pyplot as plt

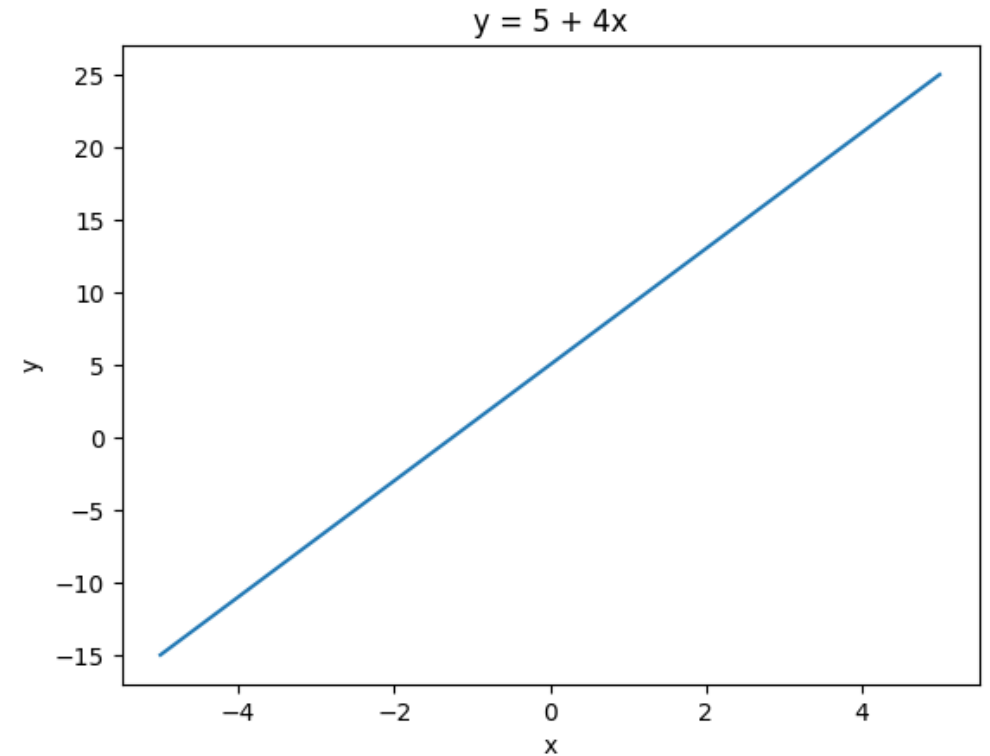
# Define the range of x values
x = np.linspace(-5, 5, 100) # Generate 100 evenly spaced points between -5 and 5

# Calculate the corresponding y values
y = 5 + 4 * x

# Create the plot
plt.plot(x, y)

# Add labels and title
plt.xlabel('x')
plt.ylabel('y')
plt.title('y = 5 + 4x')

# Show the plot
plt.show()
```



# Multivariable linear equation

```
import numpy as np
import matplotlib.pyplot as plt

# Create the x and y values
x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)

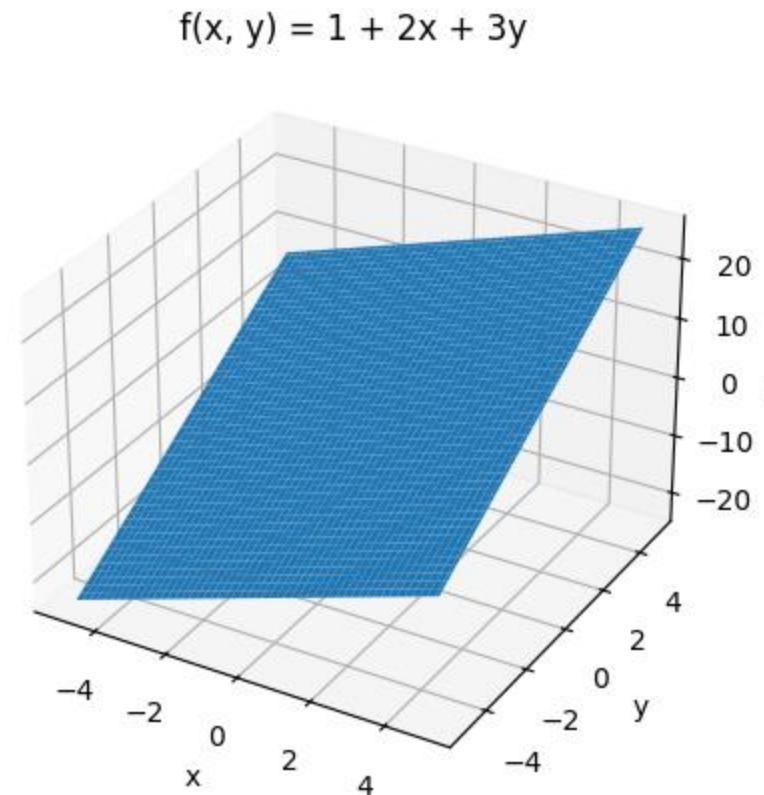
# Calculate f(x, y)
Z = 1 + 2*X + 3*Y

# Create the 3D plot
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Plot the surface
ax.plot_surface(X, Y, Z)

# Set the axis labels
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('f(x, y)')

# Display the plot
plt.title('f(x, y) = 1 + 2x + 3y')
plt.show()
```



# Polynomial equation

```
import numpy as np
import matplotlib.pyplot as plt

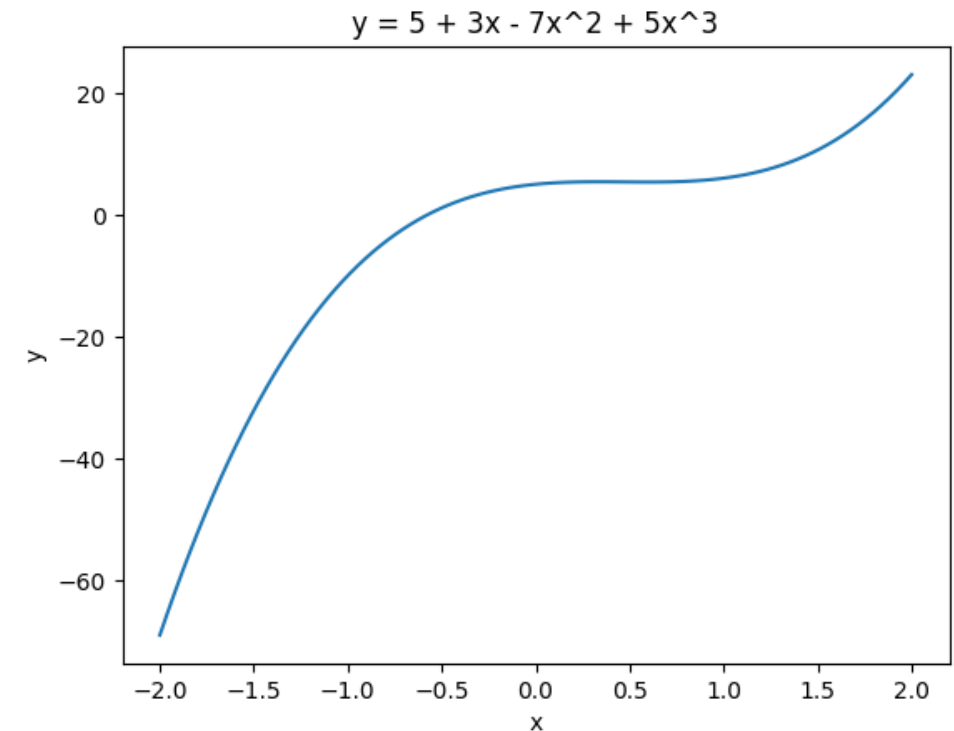
# Define the range of x values
x = np.linspace(-2, 2, 100) # Generate 100 evenly spaced points between -2 and 2

# Calculate the corresponding y values
y = 5 + 3*x - 7*x**2 + 5*x**3

# Create the plot
plt.plot(x, y)

# Add labels and title
plt.xlabel('x')
plt.ylabel('y')
plt.title('y = 5 + 3x - 7x^2 + 5x^3')

# Show the plot
plt.show()
```



## Question 1 : Plot graph and find linear equation from this data

NO.	Weight (g)	Price (THB)
1	200	100
2	300	200
3	500	350
4	800	500
5	1000	600

# Question 1 : Plot graph and find linear equation from this data

```
import numpy as np
import matplotlib.pyplot as plt

# Given data points
x = np.array([200, 300, 500, 800, 1000])
y = np.array([100, 200, 350, 500, 600])

# Plot the data points
plt.scatter(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Scatter Plot of Data Points')

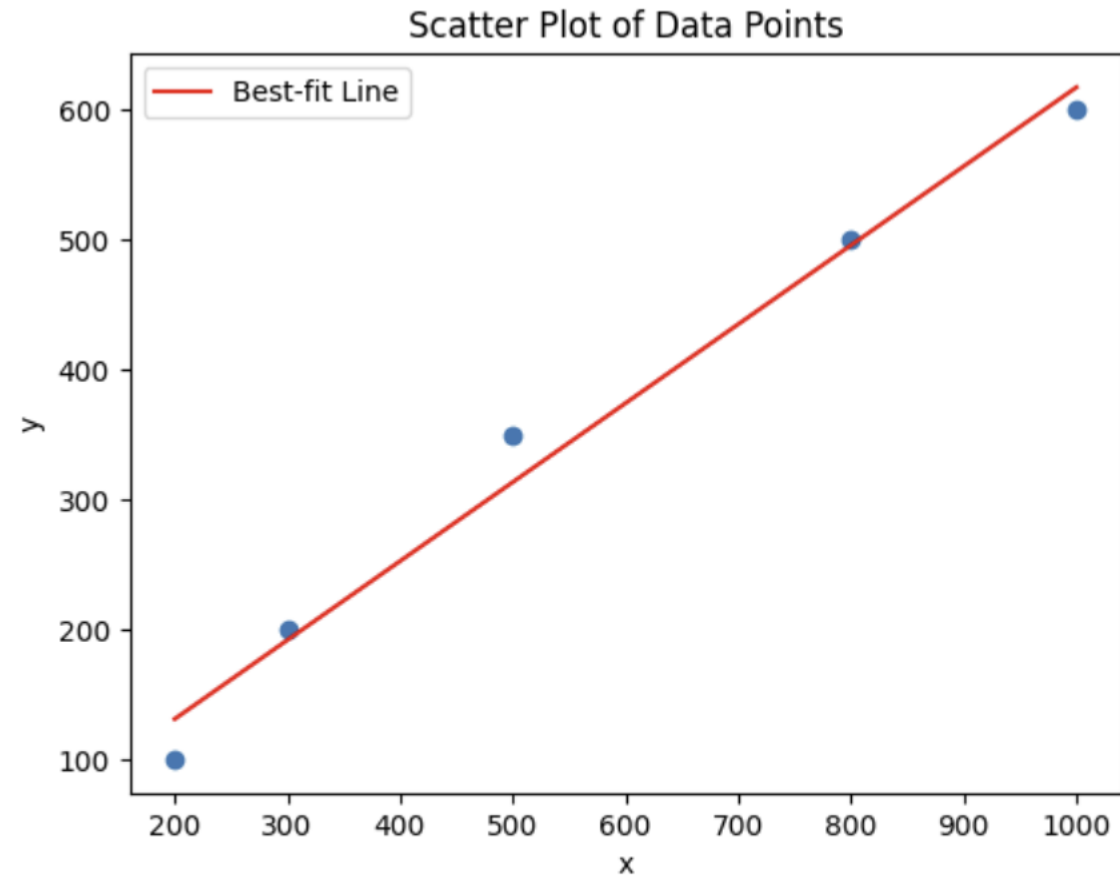
# Find the best-fit line (linear regression)
coefficients = np.polyfit(x, y, 1) # Fit a first-degree polynomial (line)
polynomial = np.poly1d(coefficients)

# Print the equation of the line
print("Equation of the line: y =", polynomial)

# Plot the best-fit line
x_line = np.linspace(min(x), max(x), 100)
y_line = polynomial(x_line)
plt.plot(x_line, y_line, color='red', label='Best-fit Line')

# Show the plot
plt.legend()
plt.show()
```

Equation of the line:  $y = 0.6084x + 9.292$





# Question 1 : Plot graph and find linear equation from this data

```
import numpy as np
import matplotlib.pyplot as plt

# Given data points
x = np.array([200, 300, 500, 800, 1000])
y = np.array([100, 200, 350, 500, 600])

# Plot the data points
plt.scatter(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Scatter Plot of Data Points')

# Find the best-fit line (linear regression)
coefficients = np.polyfit(x, y, 1) # Fit a first-degree polynomial (line)
polynomial = np.poly1d(coefficients)

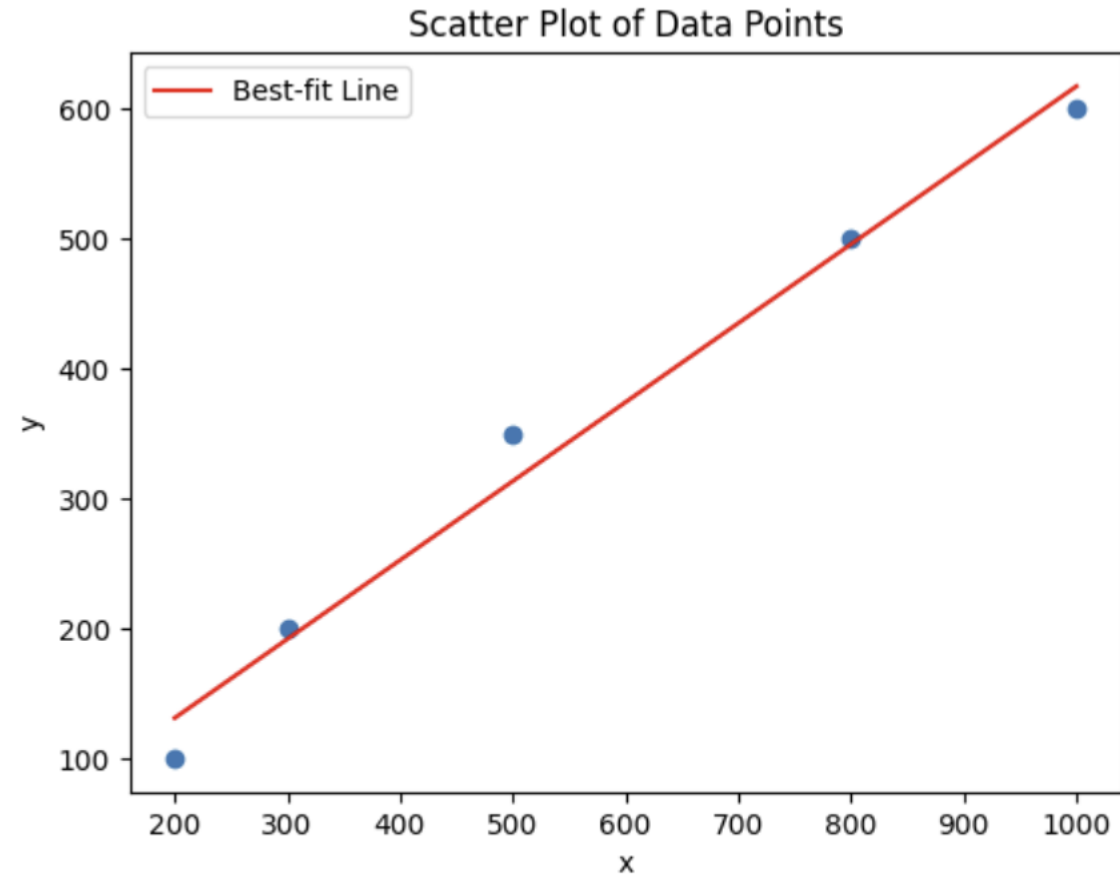
# Print the equation of the line
print("Equation of the line: y =", polynomial)

# Plot the best-fit line
x_line = np.linspace(min(x), max(x), 100)
y_line = polynomial(x_line)
plt.plot(x_line, y_line, color='red', label='Best-fit Line')

# Show the plot
plt.legend()
plt.show()
```

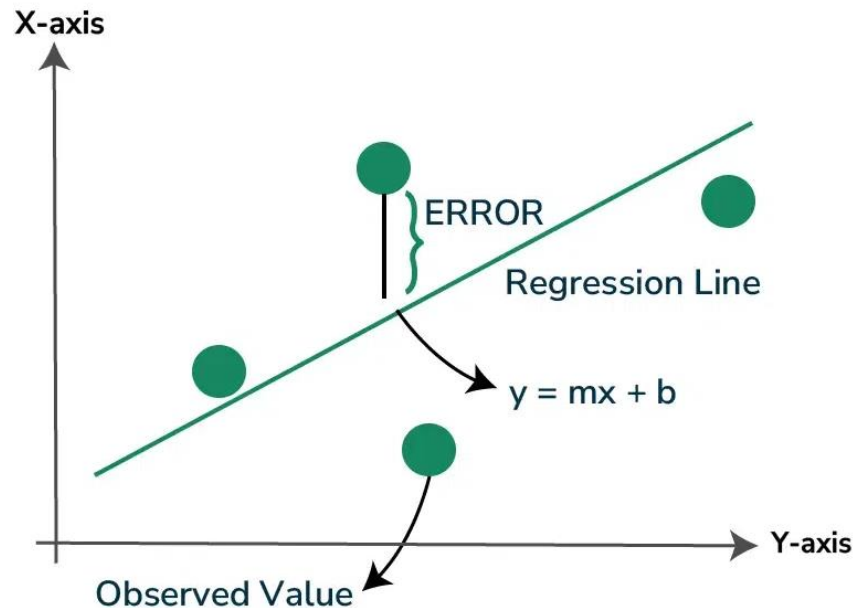
Equation of the line:  $y = 0.6084x + 9.292$

Question: How np.polyfit works?



# Least squares method

The least squares method is a form of mathematical regression analysis used to determine the line of best fit for a set of data, providing a visual demonstration of the relationship between the data points.



Source: (<https://www.geeksforgeeks.org/least-square-method/>)

Slope (m) Formula

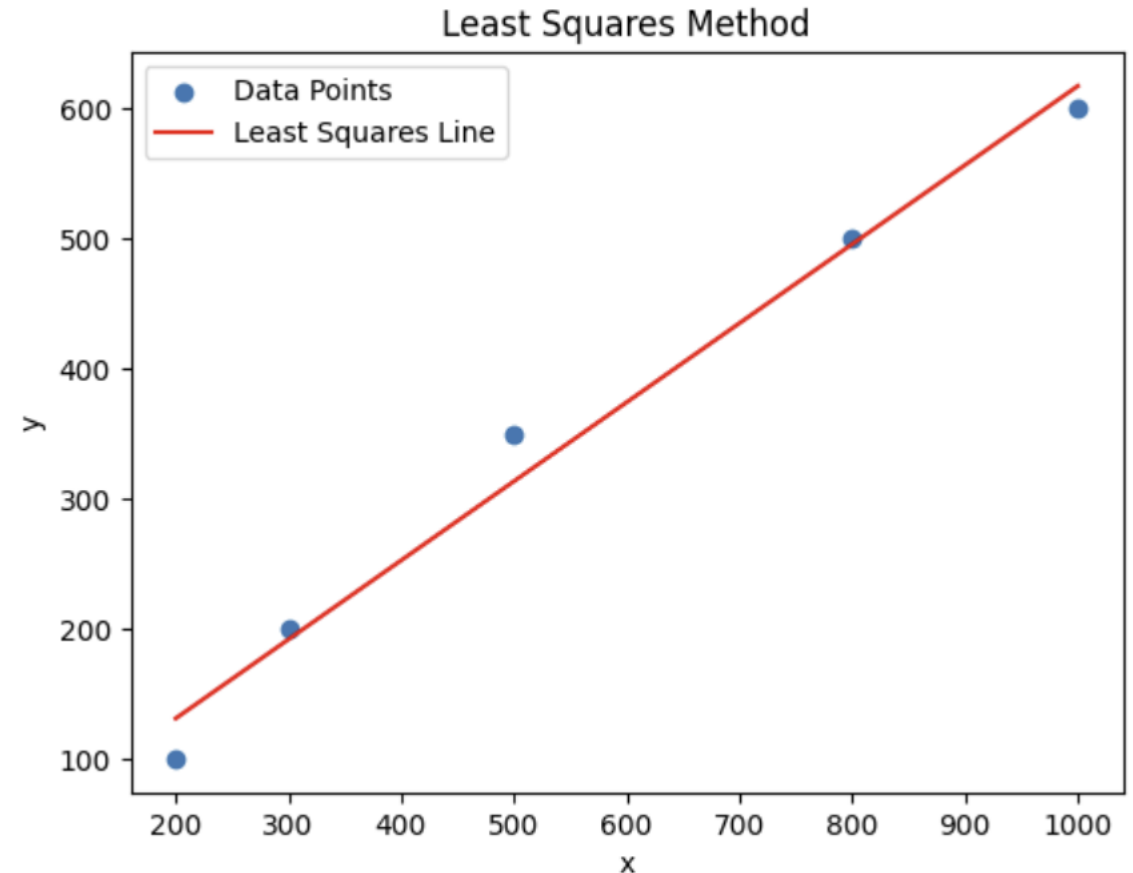
$$m = (n * \sum xy - \sum x * \sum y) / (n * \sum x^2 - (\sum x)^2)$$

Intercept (c) Formula

$$c = (\sum y) - a(\sum x) / n$$

# Least squares method

```
import numpy as np
import matplotlib.pyplot as plt
# Data points
x = np.array([200, 300, 500, 800, 1000])
y = np.array([100, 200, 350, 500, 600])
# Calculate slope and intercept using least squares method
def least_squares_fit(x, y):
    n = len(x)
    sum_x = np.sum(x)
    sum_y = np.sum(y)
    sum_xy = np.sum(x * y)
    sum_x_squared = np.sum(x**2)
    slope = (n * sum_xy - sum_x * sum_y) / (n * sum_x_squared - sum_x**2)
    intercept = (sum_y - slope * sum_x) / n
    return slope, intercept
# Calculate slope and intercept
slope, intercept = least_squares_fit(x, y)
# Generate predicted y values
y_pred = slope * x + intercept
# Plot the data points and the least squares line
plt.scatter(x, y, label='Data Points')
plt.plot(x, y_pred, color='red', label='Least Squares Line')
# Add labels and title
plt.xlabel('x')
plt.ylabel('y')
plt.title('Least Squares Method')
plt.legend()
# Show the plot
plt.show()
# Print the equation
print(f"Equation: y = {slope:.4f}x + {intercept:.4f}")
```



Equation:  $y = 0.6084x + 9.2920$

## Question 2 : Solve these equation using Metrix

$$x + 2y - z = 3$$

$$2x - y + z = 7$$

$$x - y - 3z = -6$$

## Question 2 : Solve these equation using Metrix

$$x + 2y - z = 3$$

$$2x - y + z = 7$$

$$x - y - 3z = -6$$

$$\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

## Question 2 : Solve these equations using Metrix

1. Matrix Representation

$$\begin{bmatrix} & \\ & \end{bmatrix} \begin{bmatrix} \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

A      X      B

3. Multiply Inversed A by B

$$\begin{bmatrix} & \\ & \end{bmatrix} \begin{bmatrix} \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

Therefore, the solution is

2. Find Metrix Inverse

$$X = A^{-1}B$$

$$\det(A) =$$

$$\text{adj}(A) = \begin{bmatrix} & \\ & \end{bmatrix}$$

$$A^{-1} = \frac{1}{\det(A)} * \text{adj}(A) = \begin{bmatrix} & \\ & \end{bmatrix}$$

## Question 2 : Solve these equations using Metrix

$$\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

```
import numpy as np

# Coefficient matrix
A = np.array([[1, 2, -1],
              [2, -1, 1],
              [1, -1, -3]])

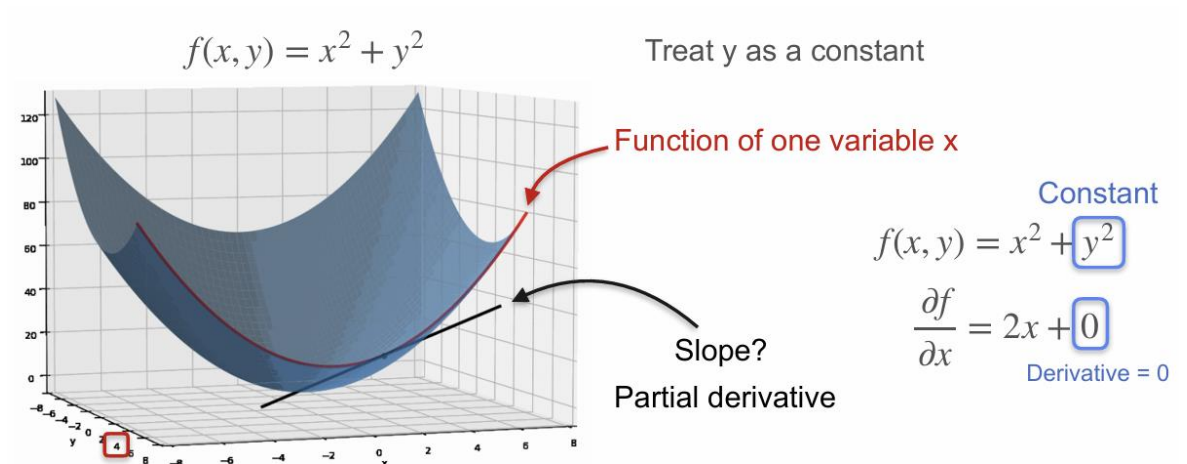
# Constant matrix
B = np.array([3, 7, -6])

# Solve for the variables
X = np.linalg.solve(A, B)

# Print the solution
print(X)
```

## (2) Calculus

- Derivative
- Partial derivative
- Gradient
- Chain rule



Source: (<https://www.deeplearning.ai/resources/#course-slides>)



**Question 3 : Plot this equation and find a minimum point**

$$f(x, y) = (x - 3)^2 + (y - 2)^2$$

## Question 3 : Plot this equation and find a minimum point

$$f(x, y) = (x - 3)^2 + (y - 2)^2$$

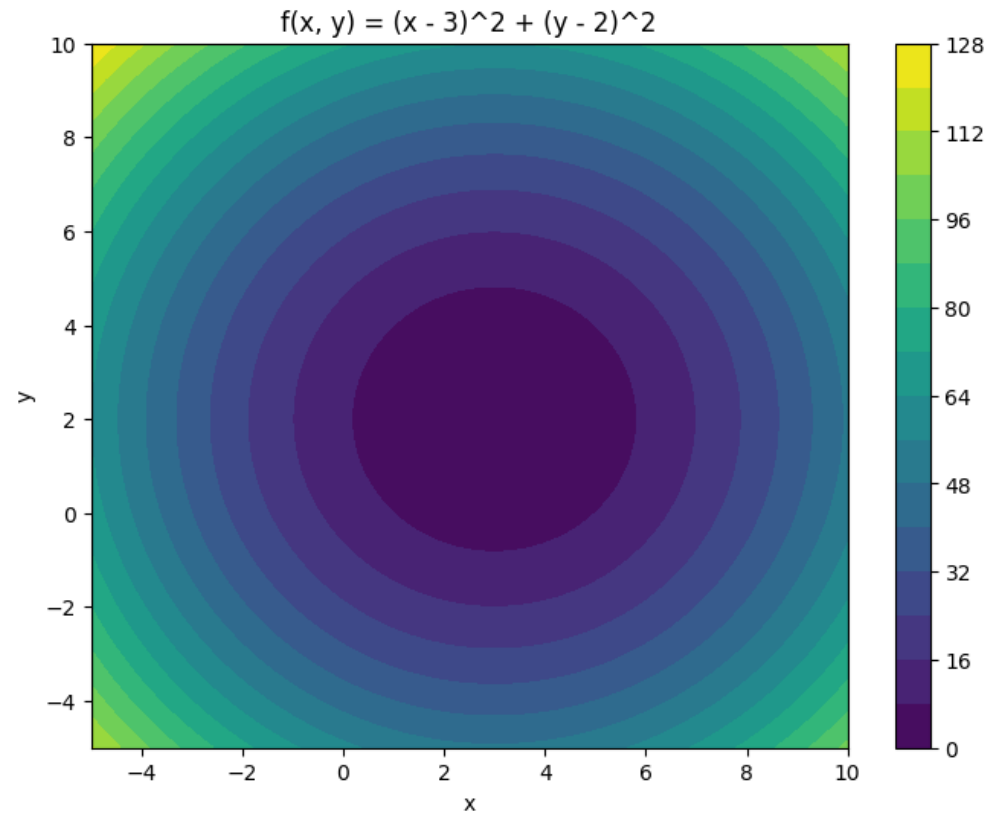
```
import numpy as np
import matplotlib.pyplot as plt

# Define the function
def F(x, y):
    return (x - 3)**2 + (y - 2)**2

# Create a meshgrid
x = np.linspace(-5, 10, 100)
y = np.linspace(-5, 10, 100)
X, Y = np.meshgrid(x, y)

# Calculate the function values
Z = F(X, Y)

# Create the plot
plt.figure(figsize=(8, 6))
plt.contourf(X, Y, Z, levels=20, cmap='viridis')
plt.colorbar()
plt.xlabel('x')
plt.ylabel('y')
plt.title('f(x, y) = (x - 3)^2 + (y - 2)^2')
plt.show()
```



## Question 3 : Plot this equation and find a minimum point

$$f(x, y) = (x - 3)^2 + (y - 2)^2$$

To find the minimum point

### 1. Partial Derivatives

Calculate the partial derivatives of  $f(x, y)$  with respect to  $x$  and  $y$

$$\partial F / \partial x =$$

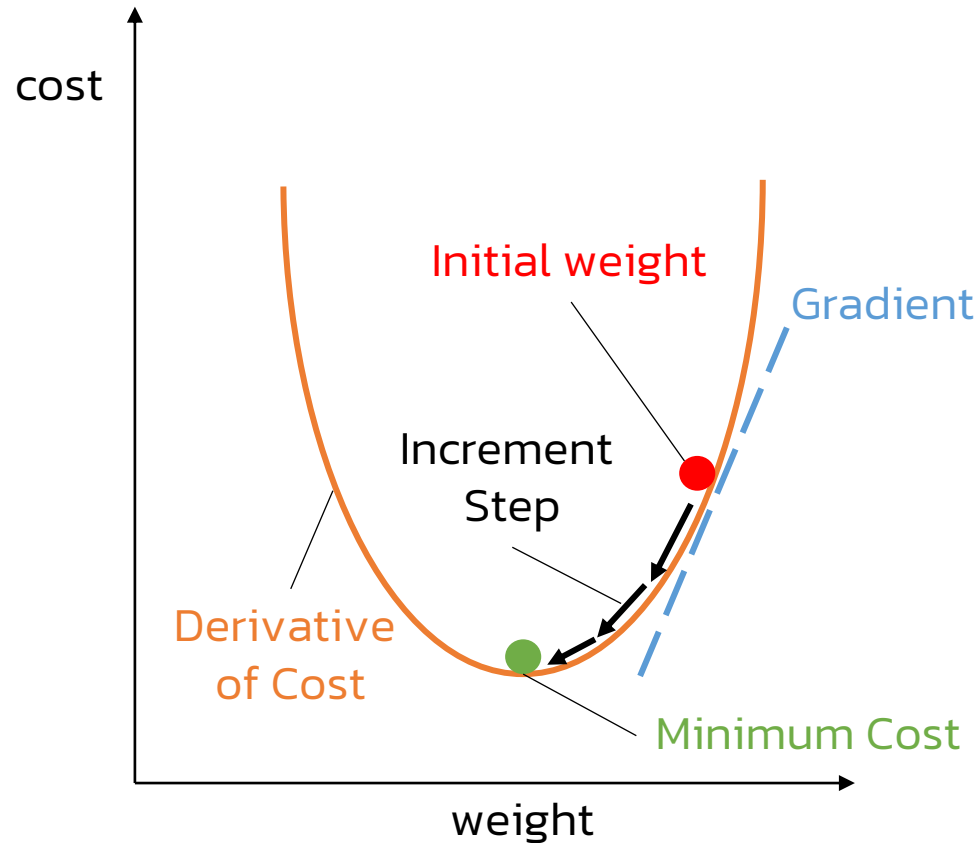
$$\partial F / \partial y =$$

### 2. Set Derivatives to Zero

To find the critical points (potential minimum or maximum), set both partial derivatives to zero

Therefore,  $f(x, y)$  has a minimum at  $(x, y) = ( \quad )$ .

# Gradient descent



Hypothesis :  $h_{\theta}(x) = \theta_0 + \theta_1 x$

Parameters :  $\theta_0$  and  $\theta_1$

Cost Function :  $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$

Goal :  $minimize J(\theta_0, \theta_1)$

Gradient Descent algorithms

Repeat until converge :

$$\theta_{j=1} = \theta_{j=0} - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

## Question 3 : Plot this equation and find a minimum point

$$f(x, y) = (x - 3)^2 + (y - 2)^2$$

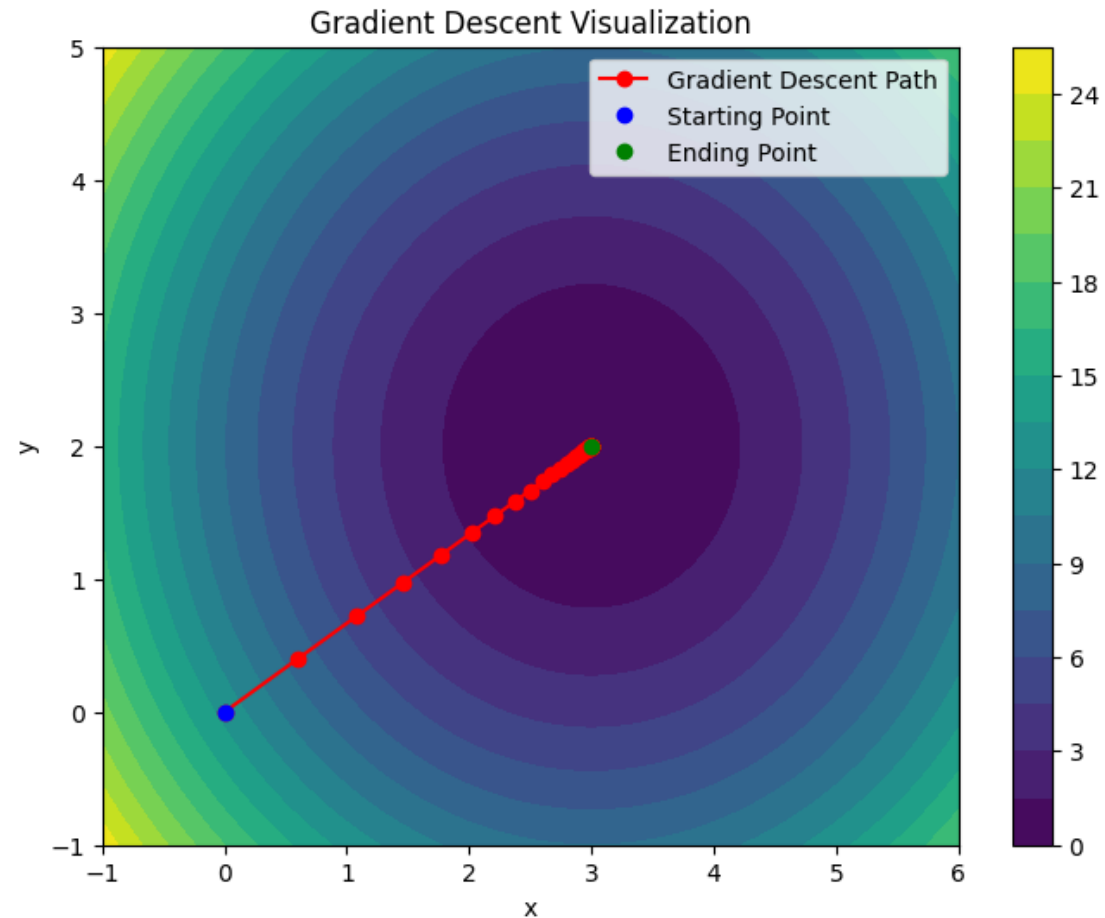
```
import numpy as np
import matplotlib.pyplot as plt
# Define the function to minimize (e.g., a simple quadratic function)
def f(x, y):
    return (x - 3)**2 + (y - 2)**2
# Gradient of the function
def gradient_f(x, y):
    return 2 * (x - 3), 2 * (y - 2)
# Initial point
x0, y0 = 0, 0
# Learning rate
learning_rate = 0.1
# Number of iterations
num_iterations = 50
# Create a meshgrid for plotting
x = np.linspace(-1, 6, 100)
y = np.linspace(-1, 5, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)
# Initialize lists to store the path of gradient descent
path_x = [x0]
path_y = [y0]
```

```
# Gradient descent iterations
for i in range(num_iterations):
    # Calculate the gradient at the current point
    grad_x, grad_y = gradient_f(x0, y0)
    # Update the point using gradient descent
    x0 -= learning_rate * grad_x
    y0 -= learning_rate * grad_y
    # Store the path
    path_x.append(x0)
    path_y.append(y0)
# Plot the contour plot
plt.figure(figsize=(8, 6))
plt.contourf(X, Y, Z, levels=20, cmap='viridis')
plt.colorbar()
# Plot the path of gradient descent
plt.plot(path_x, path_y, '-ro', label='Gradient Descent Path')
# Plot the starting and ending points
plt.plot(path_x[0], path_y[0], 'bo', label='Starting Point')
plt.plot(path_x[-1], path_y[-1], 'go', label='Ending Point')

plt.xlabel('x')
plt.ylabel('y')
plt.title('Gradient Descent Visualization')
plt.legend()
plt.show()
```

### Question 3 : Plot this equation and find a minimum point

$$f(x, y) = (x - 3)^2 + (y - 2)^2$$



# Math concepts used in training an AI model

- Gradients
- Derivatives
- Optimization
- Loss and Cost functions
- Gradient Descent

# (3) Probability and Statistics

## 1. Descriptive Statistics

Mean, median, mode, variance, standard deviation.

## 2. Probability Distributions

Normal, Bernoulli, Binomial, Poisson, etc.

## 3. Hypothesis Testing

t-tests, chi-squared tests, ANOVA.

## 4. Regression Analysis

Linear regression, logistic regression.

## 5. Dimensionality Reduction

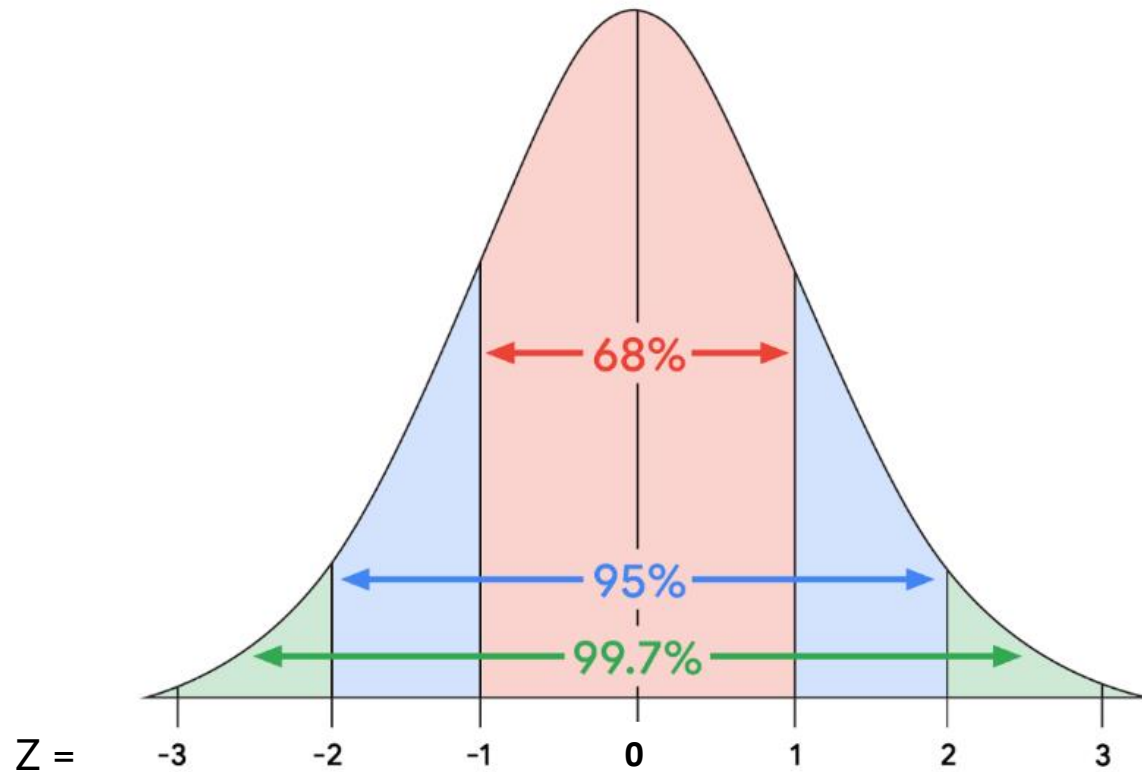
PCA, t-SNE.

## 6. Bayesian Inference

Bayes' theorem, prior and posterior probabilities.



# Normal distribution



$$Z = \frac{x - \mu}{\sigma}$$

$x$  = Data

$\mu$  = Mean

$\sigma$  = Standard deviation

# Plot Histogram of Normal distribution

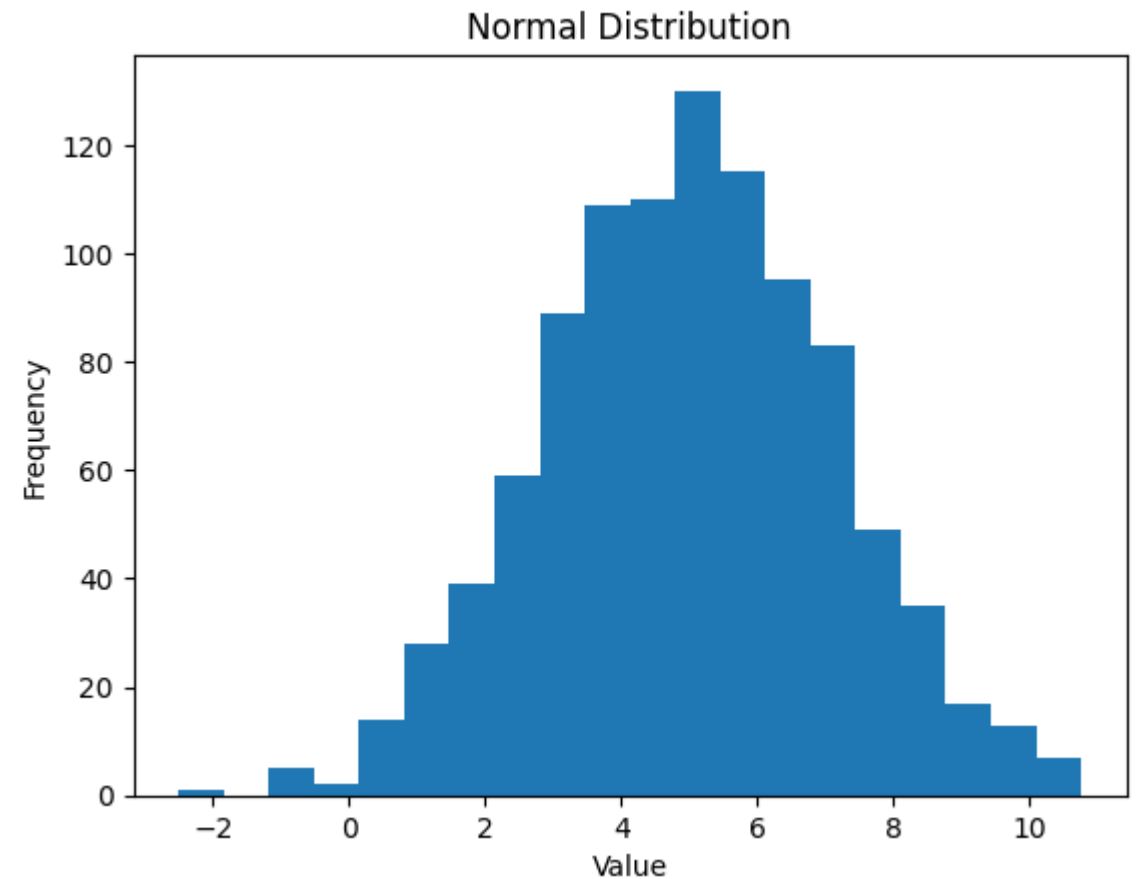
```
import numpy as np
import matplotlib.pyplot as plt

# Create a normal distribution
mean = 5 # Mean of the distribution
std_dev = 2 # Standard deviation of the distribution
num_samples = 1000 # Number of samples

normal_data = np.random.normal(mean, std_dev, num_samples)

# Create a histogram
plt.hist(normal_data, bins=20)
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.title('Normal Distribution')

# Show the histogram
plt.show()
```



# Books

- Grus, J. (2015) Data Science from Scratch: First Principles with Python. O'Reilly, Sebastopol.
- Geron, A. (2019) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2nd Edition, O'Reilly Media, Inc., Sebastopol.
- บัญชา ปะสิละเตสัง. (2563). จัดการและวิเคราะห์ข้อมูลด้วย Python Data Science. กรุงเทพฯ :บริษัท ซี เอ็ดดูเคชั่น จำกัด (มหาชน)

# Courses

- Google Advanced Data Analytics Professional Certificate (Google Career Certificates)
- Google AI Essentials (Google)
- Supervised Machine Learning: Regression and Classification (DeepLearning.AI & Stanford University)



INSTITUTE OF  
**ENGINEERING**

**Mechatronics** Modern Automotive