Machine Learning Basics

Zhiyao Zhang

May 2025

THE OHIO STATE UNIVERSITY
COLLEGE OF ENGINEERING

Outline

1. What is Machine Learning (ML)?
    a. Definitions
    b. Examples
    c. Important Concepts (Performance, Experience, Underfitting / Overfitting, *Tasks*)

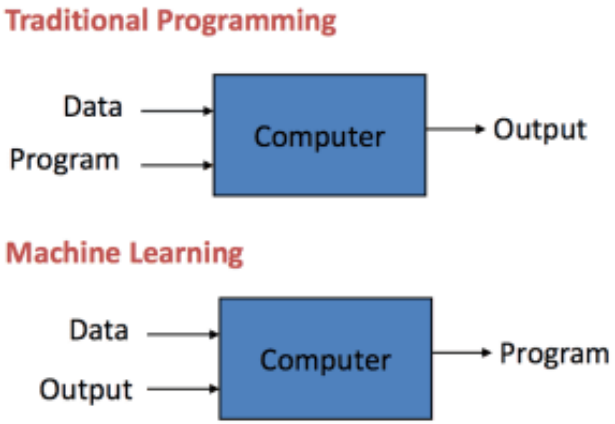
2. ML Tasks
    a. Regression / Classification / Clustering
    b. Supervised Learning / Unsupervised Learning & Others

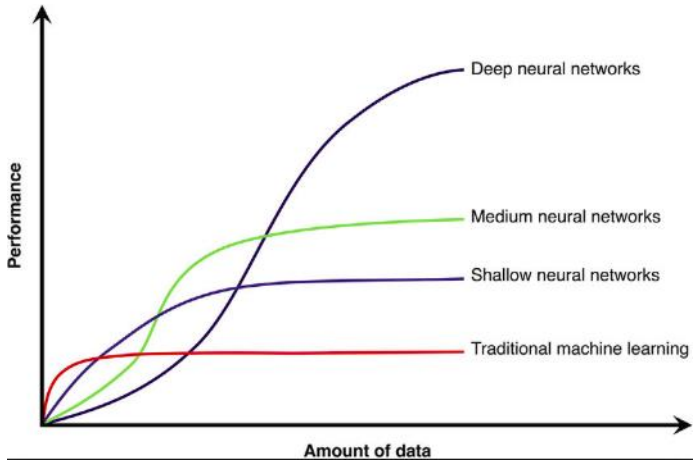
3. Case Study
    a. Problem Statement
    b. Analysis
    c. More…

Definitions

"Field of study that gives computers the ability to **learn without being explicitly programmed**." ------Arthur Samuel (1959)

"A computer program is said to learn from experience E with respect to some class of **tasks** T and performance measure P, if its **performance** at tasks in T, as measured by P, **improves with experience** E." ------Tom M. Mitchell (1997)

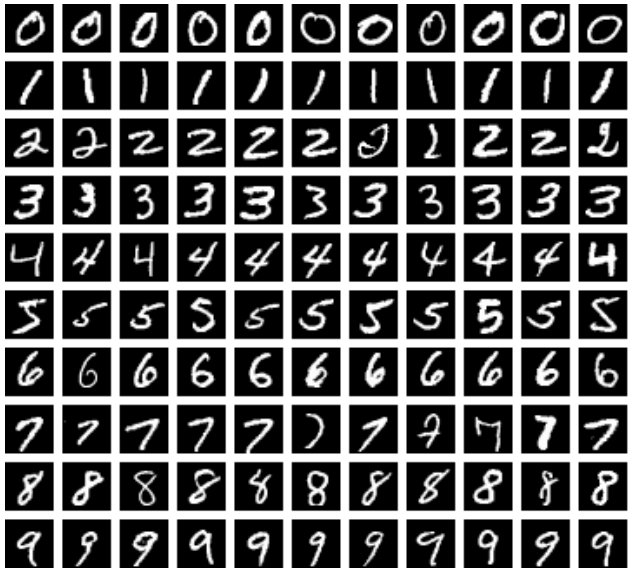




Examples

Handwriting recognition learning problem

- **Task** T : Recognizing and classifying handwritten words within images
- **Performance** P : Percent of words correctly classified
- **Experience** E : A dataset of handwritten words with given classifications



Examples

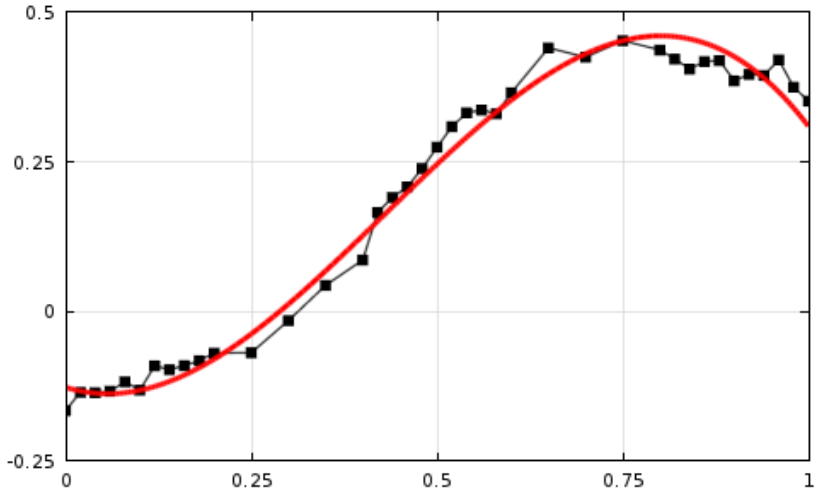
A robot driving learning problem

- **Task T** : Driving on highways using vision sensors
- **Performance P** : Average distance traveled before an error/accident
- **Experience E** : A sequence of images & videos and steering commands recorded while observing a human driver

Examples

A function fitting problem

- **Task T** : Fit the function using partially observable data
- **Performance P** : The total error between fitted curve and original function
- **Experience E** : A collection of (noisy) data points



Important Concepts ------ Performance

Performance measure (metric) is used to evaluate how well a program performs task T. There are different kinds of performance metrics.

For example:
In **Handwriting recognition learning problem**, we consider the ***classification accuracy*** (the proportion of examples for which the model produces the correct output) of the program.

$$\text{Accuracy} = \frac{\#\text{ correct output}}{\#\text{ total examples}} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\{\hat{y}_i = y_i\}$$

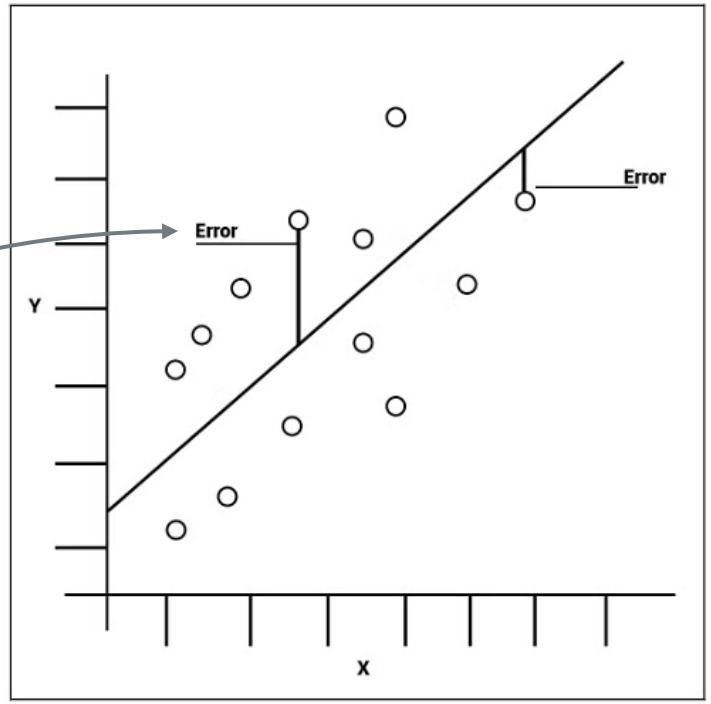
Important Concepts ------ Performance

In **function fitting problem**, we consider the ***Mean Squared Error (MSE)***, which characterizes the "total distance" from the points on the fitted curve and the input data points.

Mean

Error Squared

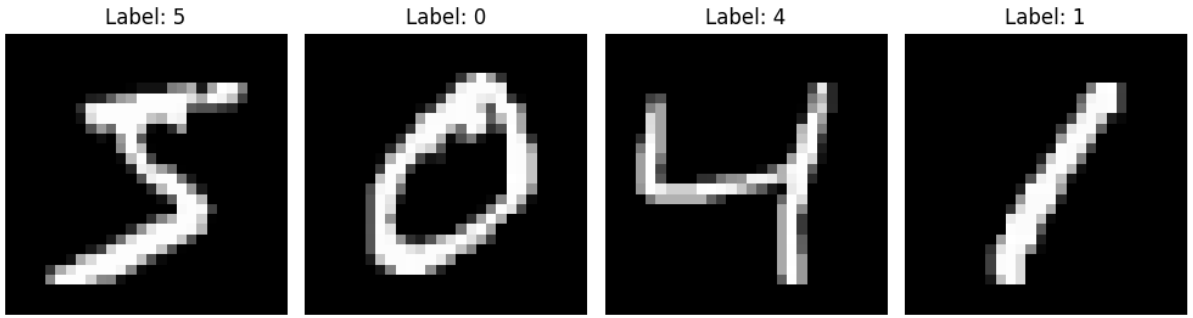
$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$



Important Concepts ------ Experience

Experience (sample) is fed to a program for learning to improve the performance for the corresponding task.

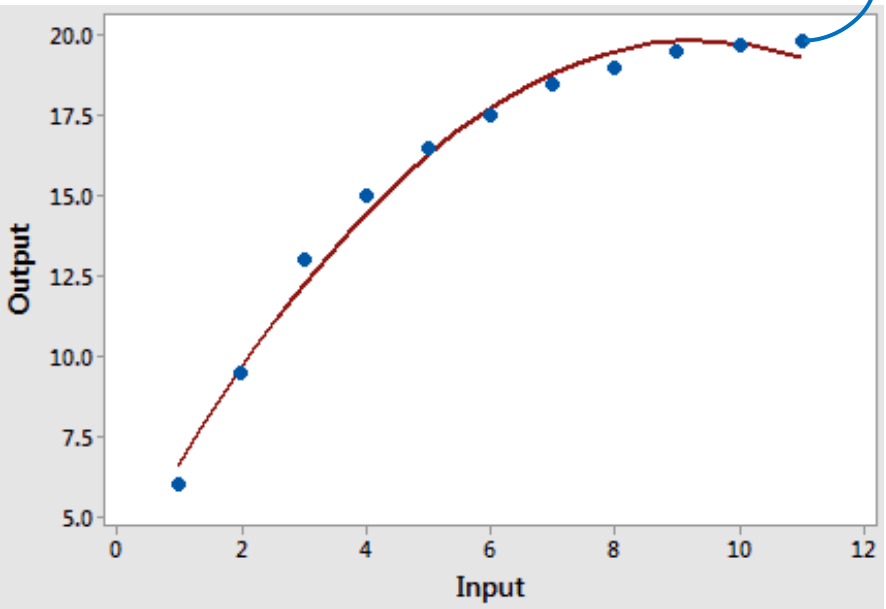
For example:
In **Handwriting recognition learning problem**, we input the handwritten digits with their corresponding labels.

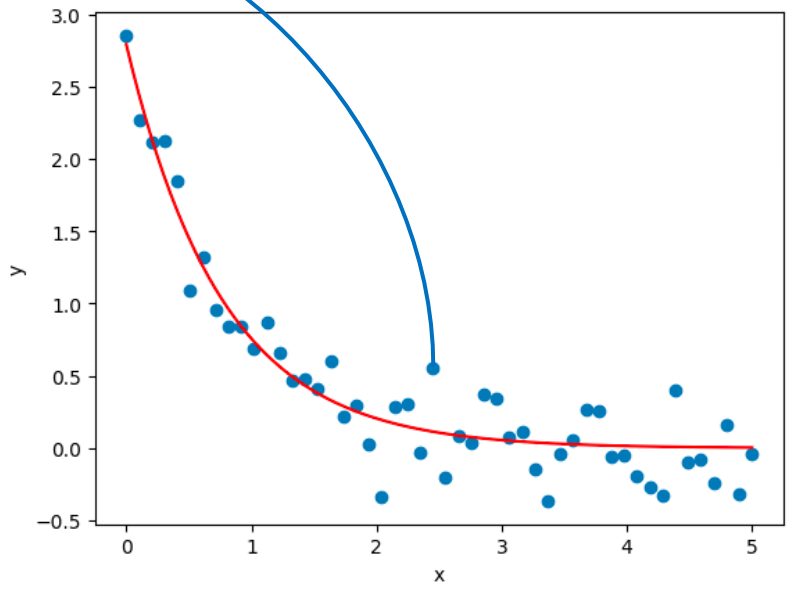


Important Concepts ------ Experience

In **function fitting problem**, we input the data points along with their function values (can be noisy):

$(x_i, \hat{y}_i)$





Important Concepts ------ Experience

We often split the samples into three datasets: **training set**, **validation set** (is ignored sometimes), and **test set**.

The samples in **training set** are fed to the program for learning task.

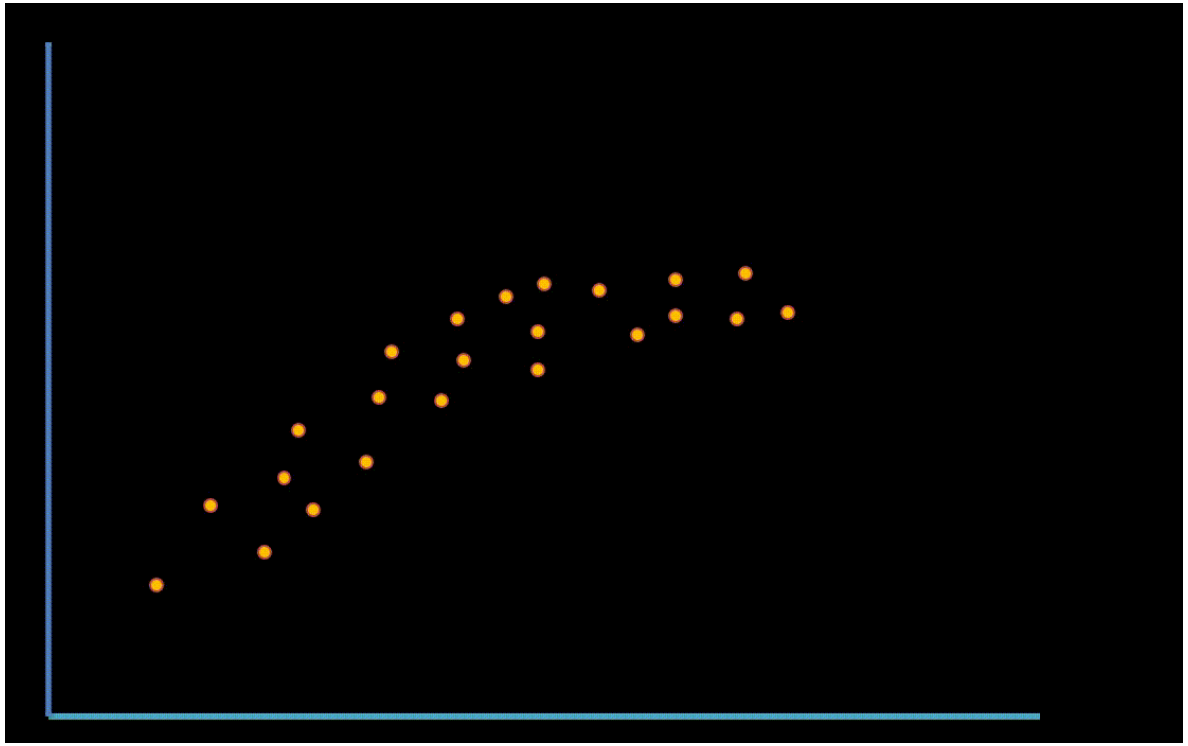
After training process, **validation set** is used to fine-tune the program.

Finally, **test set** is used to evaluate if the program is learning well.

Important Concepts ------ Underfitting/Overfitting

When the learning is at its infancy stage, experience is insufficient, the program is underfitting, the performance metric is bad.

When the learning is "too much", experience is redundant, the program is overfitting, the generalization capability is bad.



ML Tasks ------ Learning Targets

According to different **learning targets**, we can classify ML tasks into the following categories:

- Regression: Predict continuous values.
- Classification: Predict discrete values.
- Clustering: Determine the structure of data.

ML Tasks ------ Regression

Essentially, **Regression** task is to learn a function:

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

For example, the function fitting problem is a regression task.

In practice, regression task can be applied in house price prediction: The input can be the number of rooms (an 1-dim integer), the location of the house (which can be a function of coordinates itself), etc., and the output is the predicted price of the house (an 1-dim real number).

ML Tasks ------ Classification

Essentially, **Classification** task is to learn a function:

$$f : \mathbb{R}^n \rightarrow \mathcal{C}$$

where *C* is a discrete set. For example, the handwriting digit recognition problem is a classification task, where $C=\{0, 1, \ldots, 9\}$.

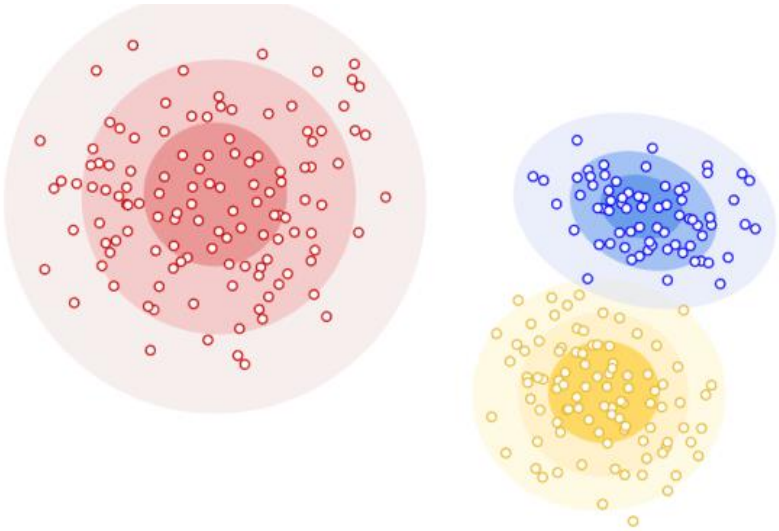
Notably, the labels in *C* are often without order. In price prediction (regression), 10,000$ is similar to 10,001$; however, in handwriting digit recognition (classification), labels *1* and *2* are **not closer** than labels *1* and *9*.

ML Tasks ------ Clustering

Clustering task is to learn the structure of data.

For example, e-commerce companies need to classify users to provide better services (recommendations, etc.), thus increase benefits. These users **do not have explicit labels**.

Instead of classification task, clustering task needs to divide these users into different categories based on their purchase records, browsing records, where no labels are available.



ML Tasks ------ Supervision

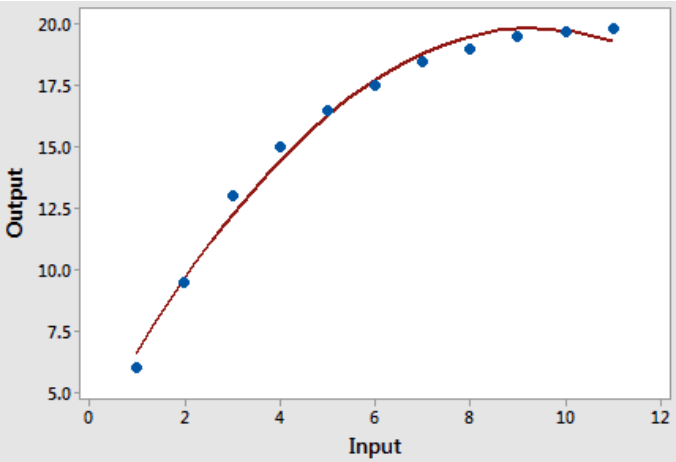
According to the **feature of experience (samples),** we can classify ML tasks into the following categories:

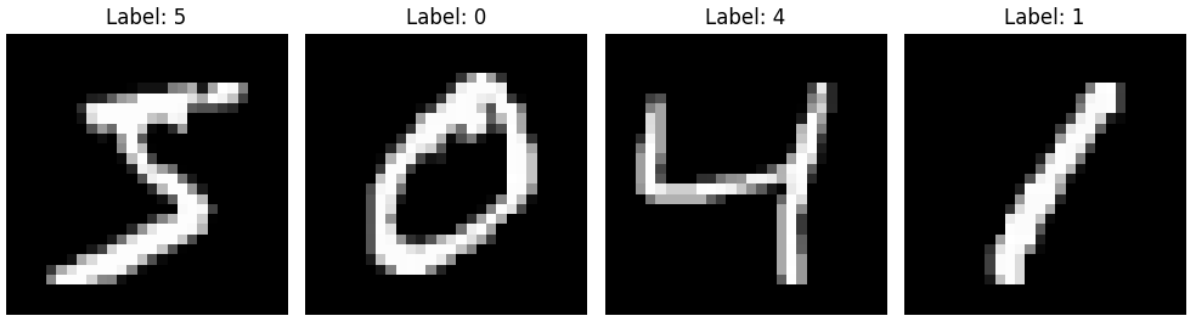
- Supervised Learning
- Unsupervised Learning
- More…

ML Tasks ------ Supervised Learning

Supervised Learning is a paradigm where a model is trained using input objects and desired output values (also known as a supervisory signal), which are often human-made **labels**.

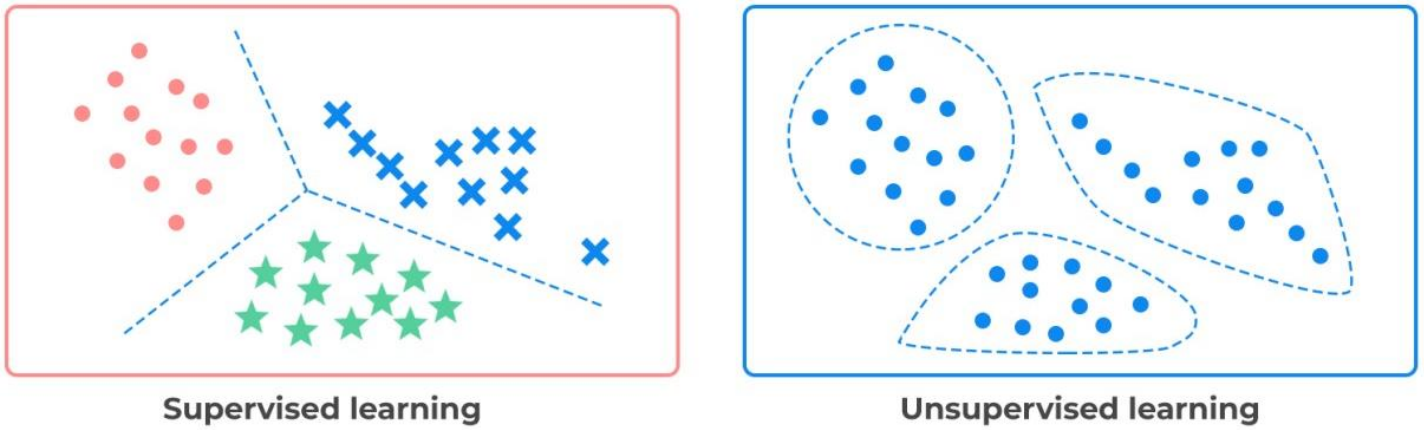
For example, **Regression tasks** and **Classification tasks** are supervised learning problems.





ML Tasks ------ Unsupervised Learning

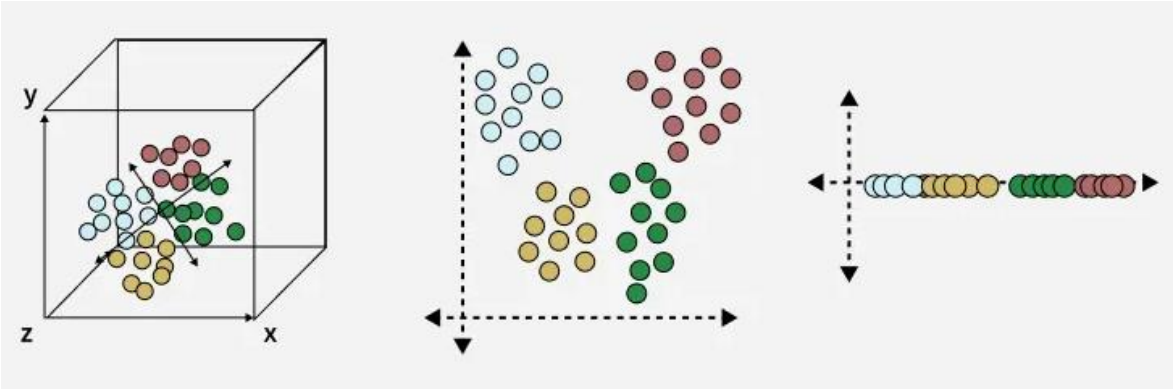
Unsupervised Learning learns from data without human supervision. Unlike supervised learning, unsupervised machine learning models are given **unlabeled data** and allowed to discover patterns and insights without any explicit guidance or instruction.



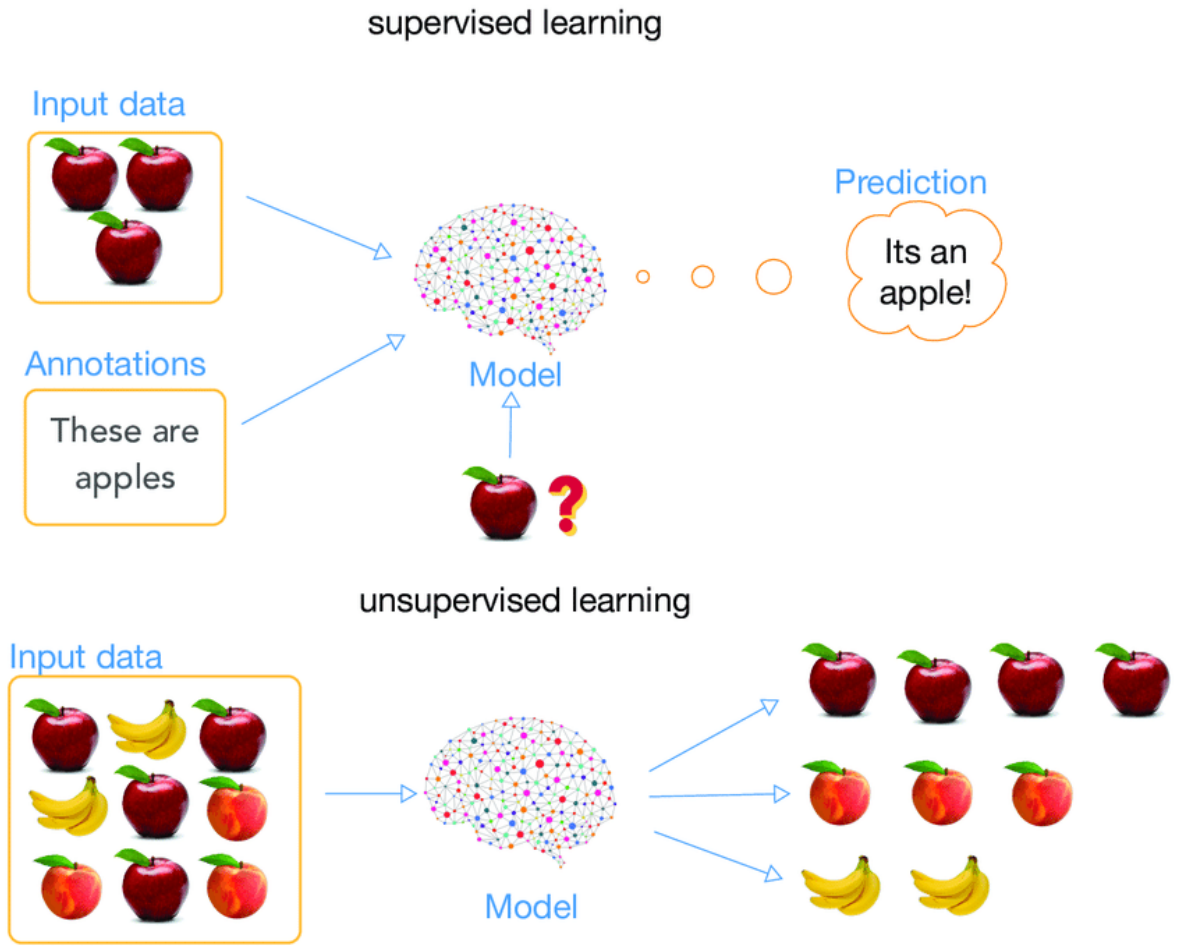
ML Tasks ------ Unsupervised Learning

Except for **Clustering** tasks, **Dimensionality Reduction** is another common unsupervised learning task.

Dimensionality reduction algorithms reduce the number of input variables in a dataset while preserving as much of the original information as possible. This is useful for reducing the complexity of a dataset and making it easier to visualize and analyze.

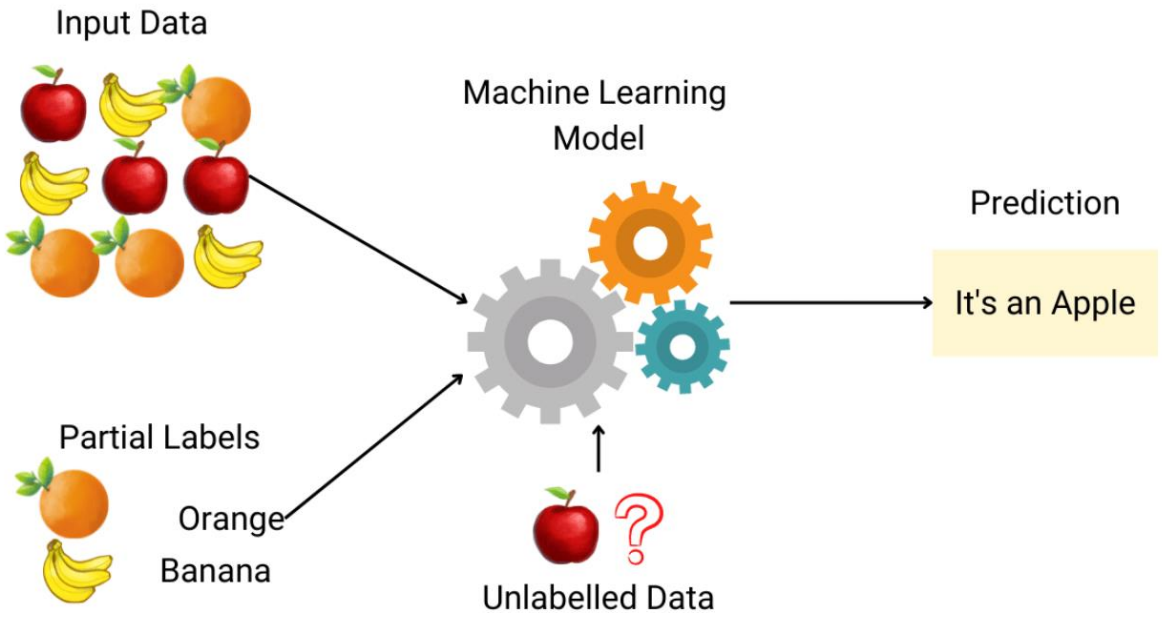


ML Tasks ------ Supervised / Unsupervised Learning



ML Tasks ------ Semi-supervised Learning

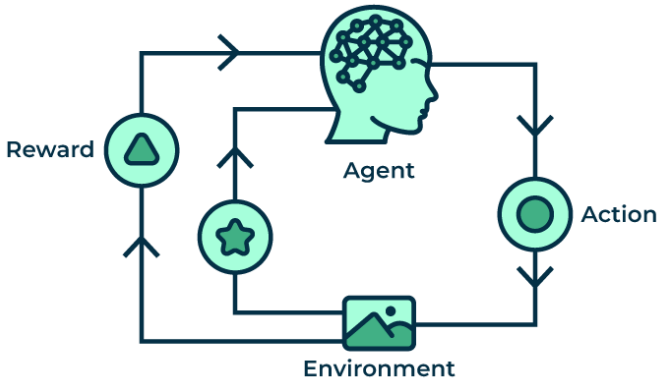
Semi-supervised Learning is the problems with a large amount of input data, but only some of the data is labeled. Semi-supervised learning falls between unsupervised learning and supervised learning.



ML Tasks ------ Reinforcement Learning

Reinforcement Learning (RL) is the problem of getting an agent learns to interact with an environment by performing actions and receiving rewards or penalties based on its actions.

The goal of RL is to learn a policy, which is a mapping from states to actions, that maximizes the expected cumulative reward over time.



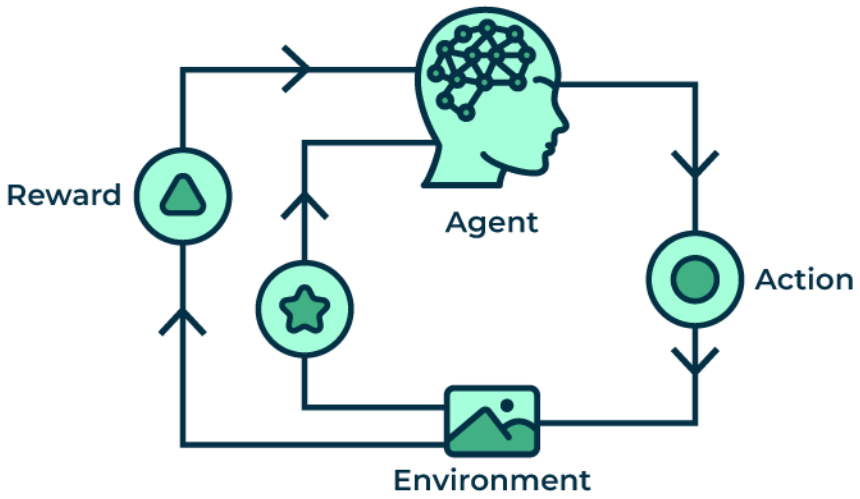
ML Tasks ------ Reinforcement Learning

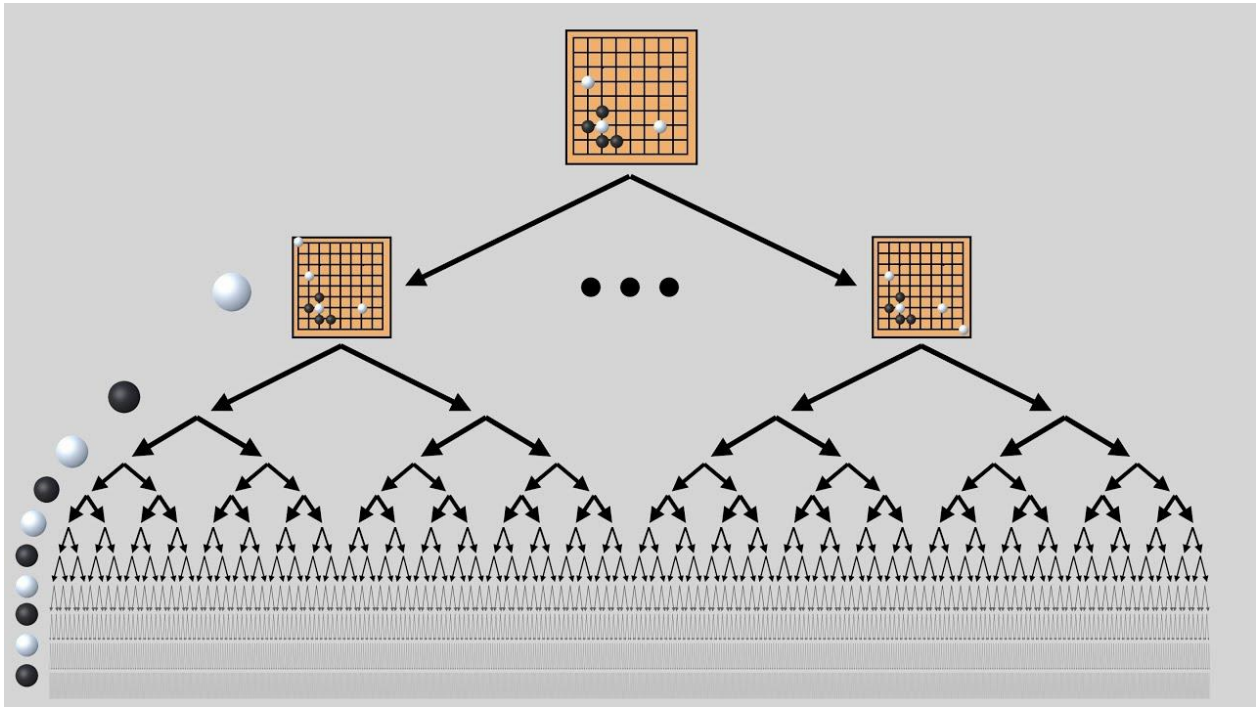
RL is distinct in two key ways: it involves **sequential decision-making** and **receives delayed feedback**. Although RL differs fundamentally from supervised and unsupervised learning, it shares some techniques with them and is therefore often considered a subfield of ML.

Sequential decision-making: The current decision will impact the next decision, since the data is sequential. Note that in conventional ML tasks, data is static.

Delayed feedback: Instead of receiving correct / incorrect predictions, the agent receives rewards. Besides, the reward is not received immediately.

ML Tasks ------ Reinforcement Learning

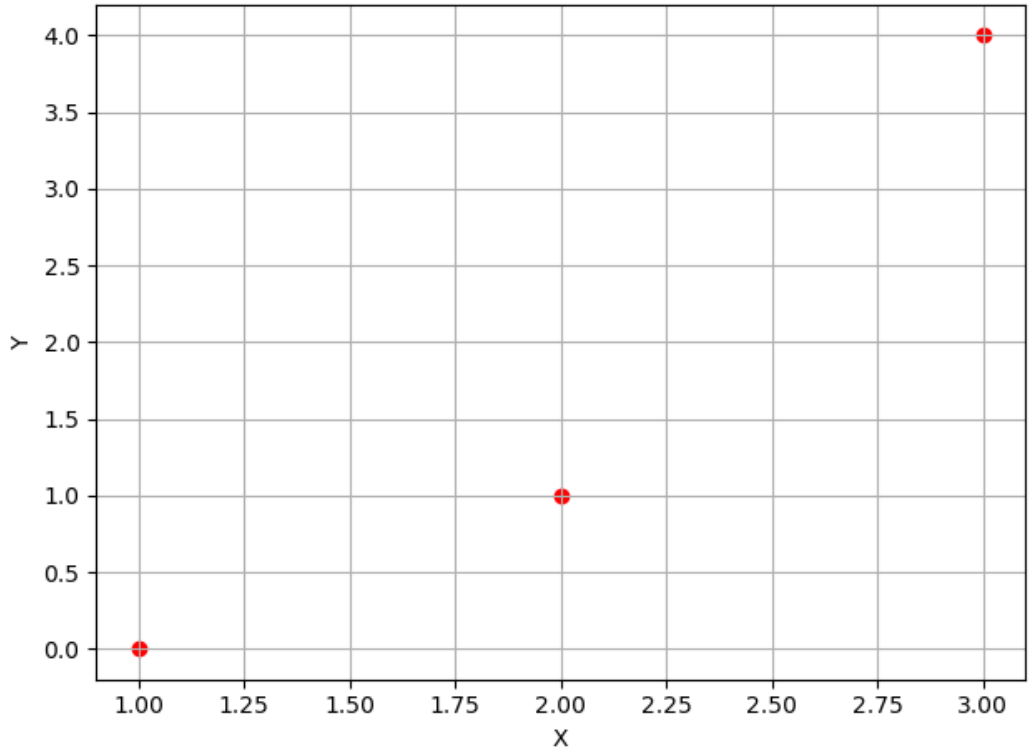




Case Study ------ Problem Statement

Finally, let's analyze a concrete example to ***1)*** understand ML better, and ***2)*** learn some common techniques in ML. Here's the statement of problem:

Given three data points (1, 0), (2, 1), (3, 4), please find a linear function $y=kx+b$ to fit the input data points, ensuring that ***MSE*** is minimized.



Case Study ------ Analysis

First, let's recall today's lecture.

Q1: What's TPE in this problem?

Q2: Is this problem a regression / classification / clustering problem?

Q3: Is this problem a supervised learning / unsupervised learning / reinforcement learning problem?

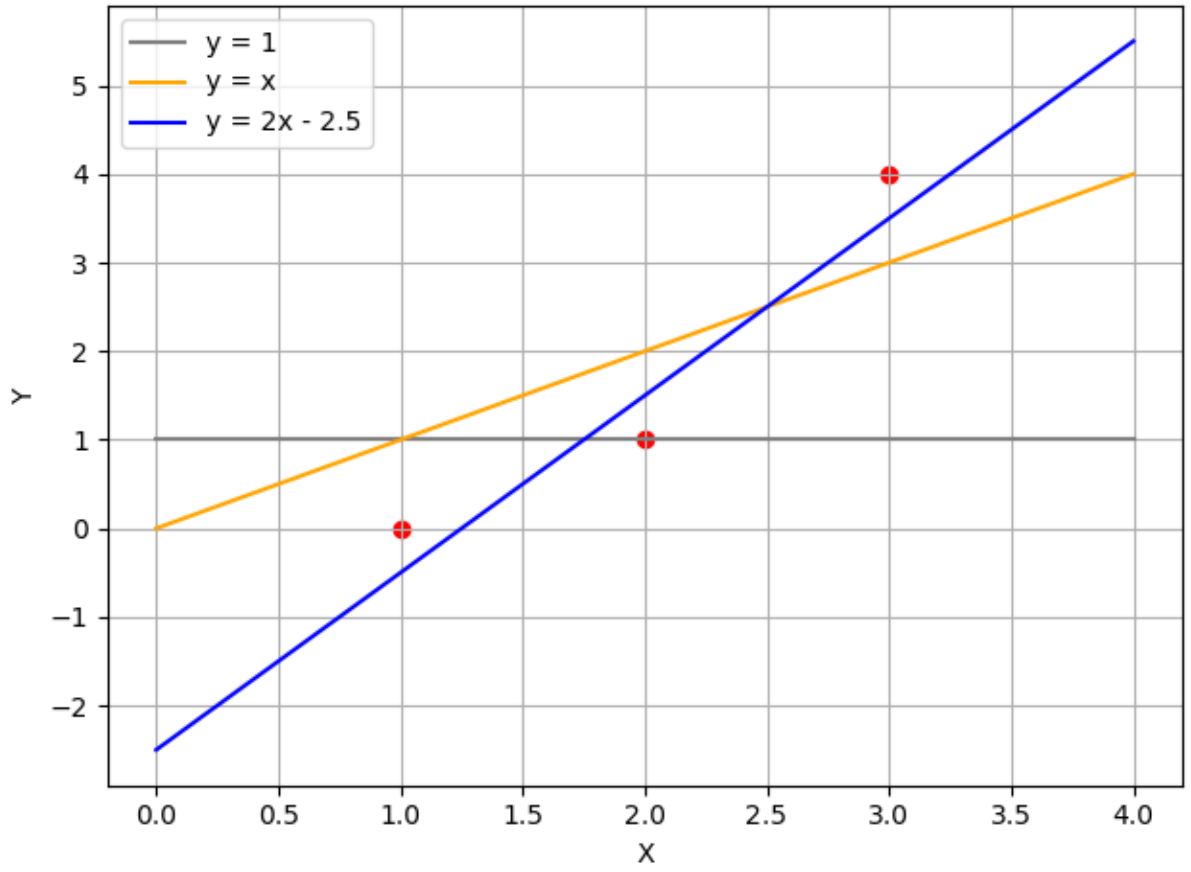
Case Study ------ Analysis

A1: *1)* Task: To learn a linear function, which should be as "*close*" as possible to the data points. *2)* Performance measure: Mean square error (MSE). *3)* Experience: Three given data points.

A2: It's a regression problem, since the outputs of $y=kx+b$ for different x's are continuous.

A3: It's a supervised problem, since every the label y of data point x is given.

Case Study ------ Analysis



Case Study ------ Analysis

Denote $(x_1, y_1)=(1, 0)$, $(x_2, y_2)=(2, 1)$, $(x_3, y_3)=(3, 4)$. The MSE for a function $y=kx+b$ is:

$$
\begin{aligned}
MSE(k,b) &= \frac{1}{N}\sum_{i=1}^{N}(y_i-(kx_i+b))^2 \\
&= \frac{1}{3}\sum_{i=1}^{3}(y_i-(kx_i+b))^2 \\
&= \frac{1}{3}\left((k+b)^2+(2k+b-1)^2+(3k+b-4)^2\right).
\end{aligned}
$$

We call it **Loss (Cost) Function**. We need to minimize this loss function L.

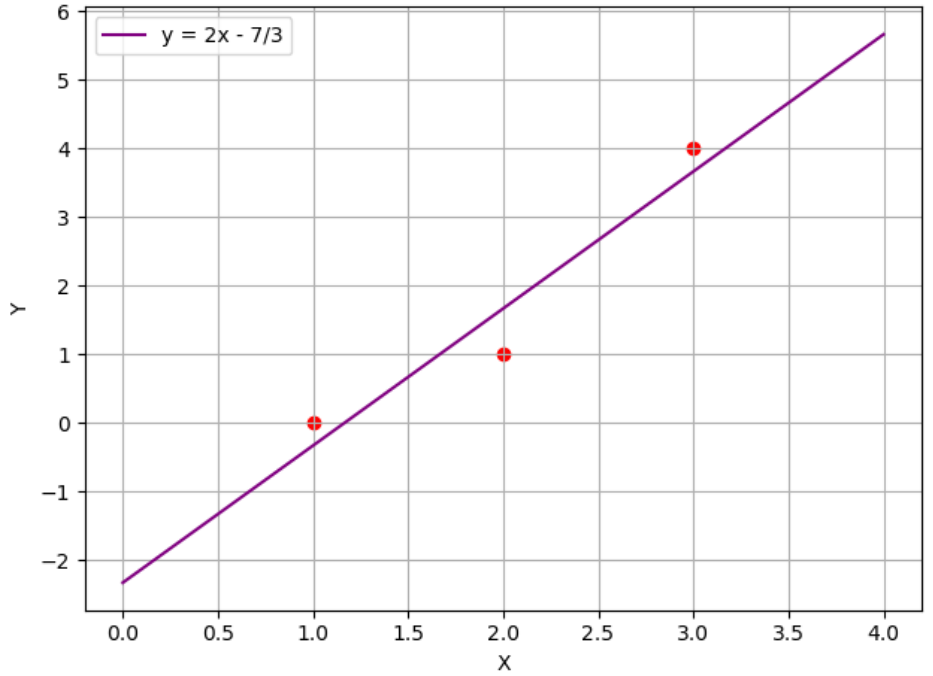
------ How to do?

Case Study ------ Analysis

The loss function enjoys an easy expression with good properties, implying that we can even compute the closed form of the optimal pair in this problem.

Theoretically, we can compute the gradients, and set them to *0*. As the loss function is convex, this gives us the optimal pair (k, b), which serves as the solution of this problem:

$$(k, b) = (2, -\frac{7}{3})$$



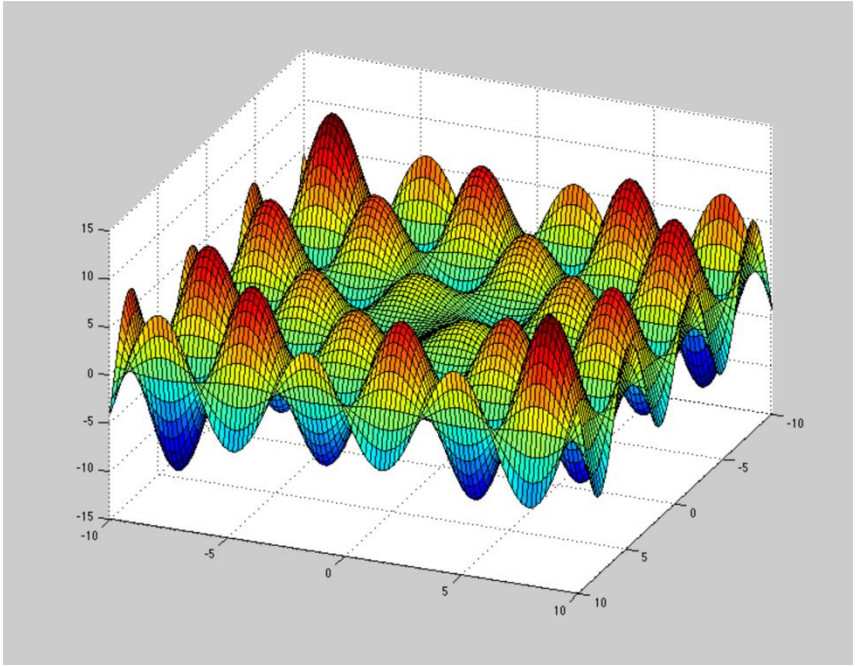
Case Study ------ Analysis

However, this does not work in practice.

1. The loss function can be complicated, with "ugly" shapes, implying that the high dimensional problem is sometimes impossible to solve:

$$\partial_\theta L(\theta) = 0$$

2. Even if we can solve the complicated system, it's computationally expensive.

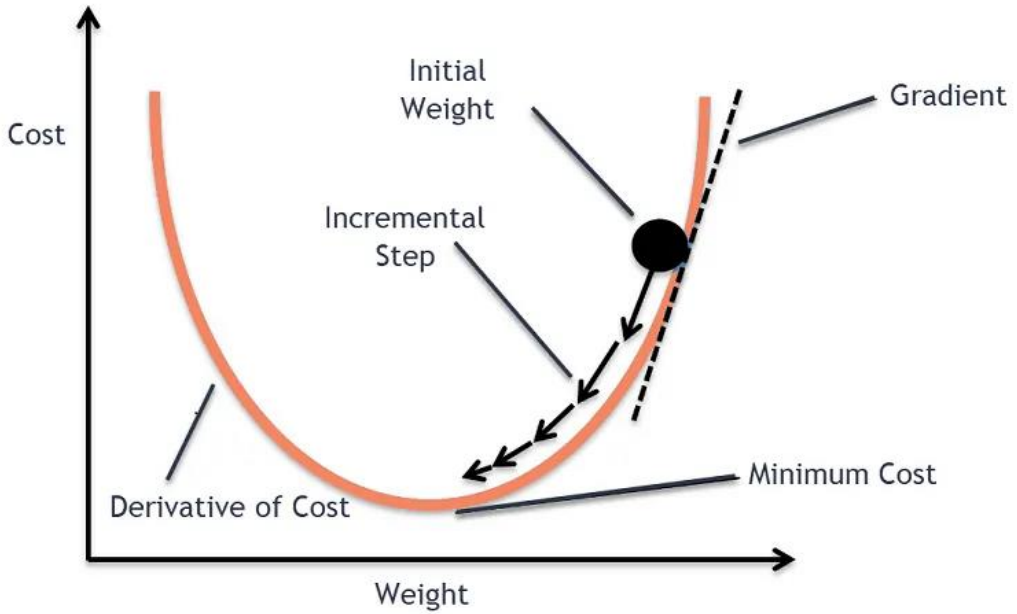


Case Study ------ Analysis

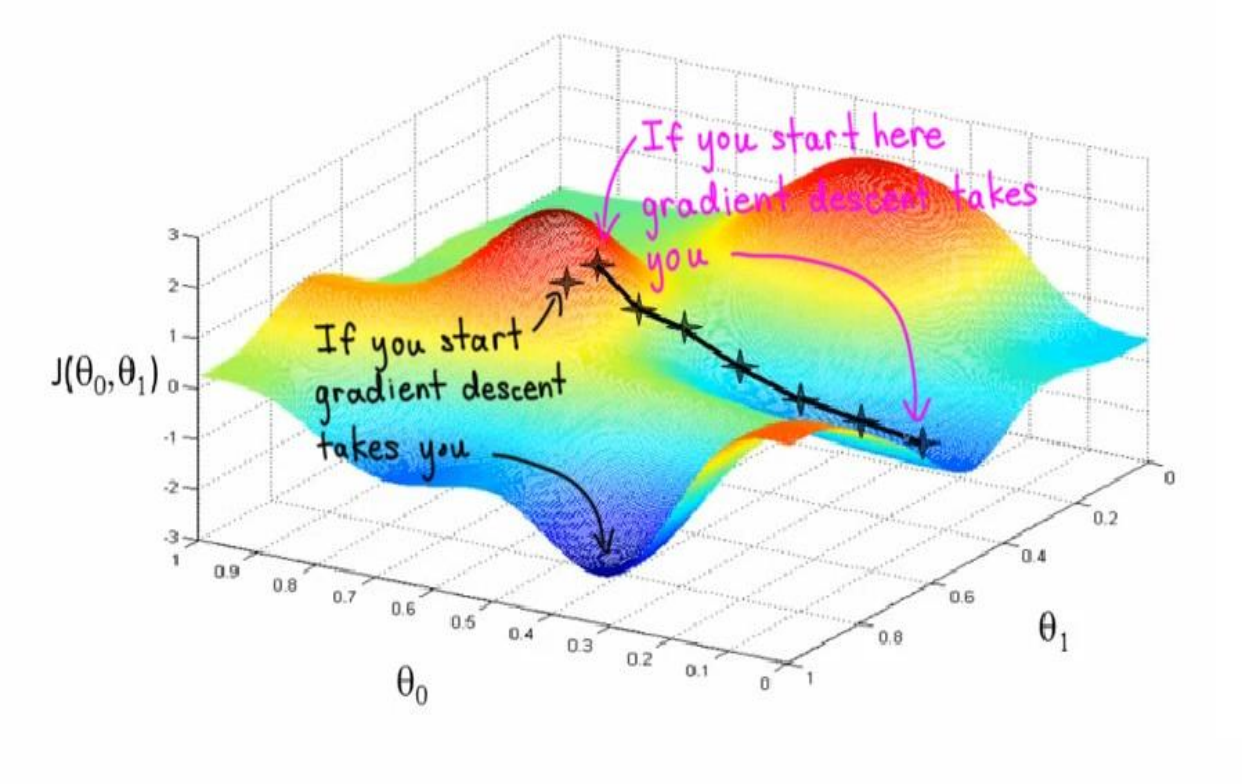
Therefore, we use **Gradient Descent** method (GD, and its extensions such as stochastic GD, momentum, etc.) to solve the problem numerically.

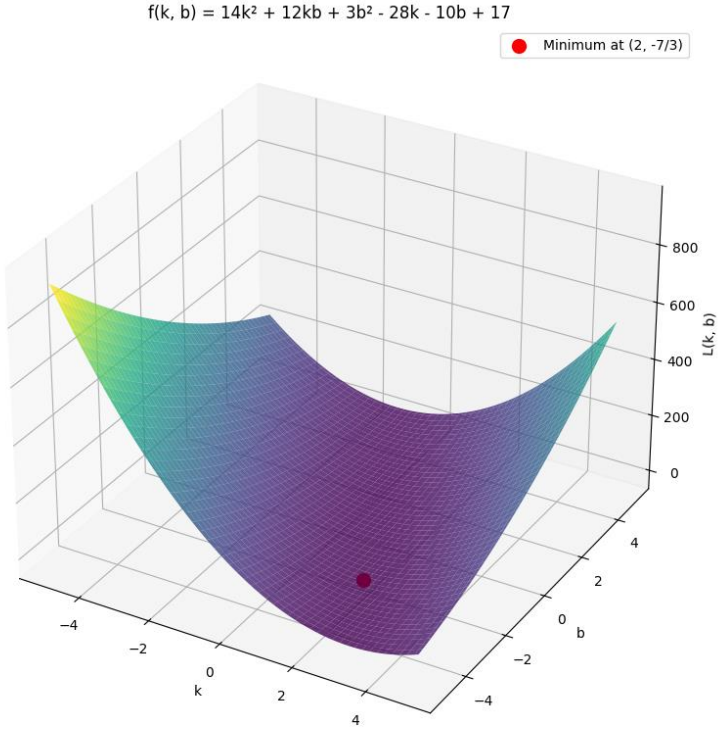
$$\theta_{k+1} = \theta_k - \alpha_k \frac{\partial L(\theta_k)}{\partial \theta}$$

Given the current parameter, GD computes the gradient, and takes a tiny step towards to descent direction.



Case Study ------ Analysis



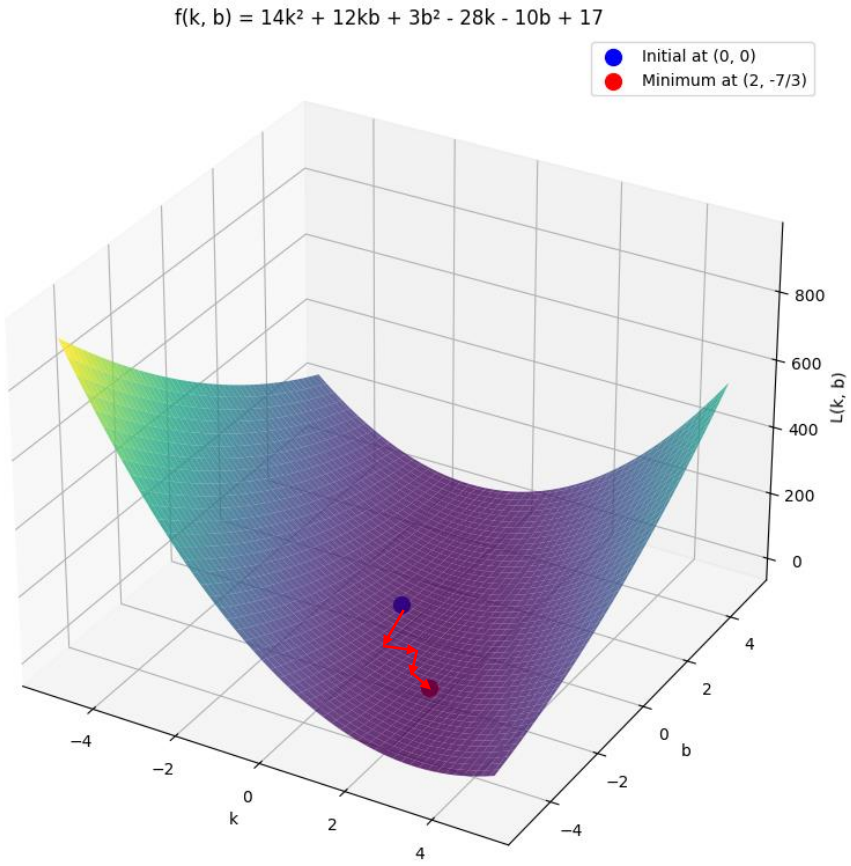


Case Study ------ Analysis

For this problem, we can select $(k, b) = (0, 0)$ as initial values. We compute the partial gradients as follow:

$$\frac{\partial L}{\partial k}(0,0) = -\frac{28}{3},$$
$$\frac{\partial L}{\partial b}(0,0) = -\frac{10}{3},$$

Then, we select a small $\boldsymbol{\alpha}$, and take a tiny step to update parameters.



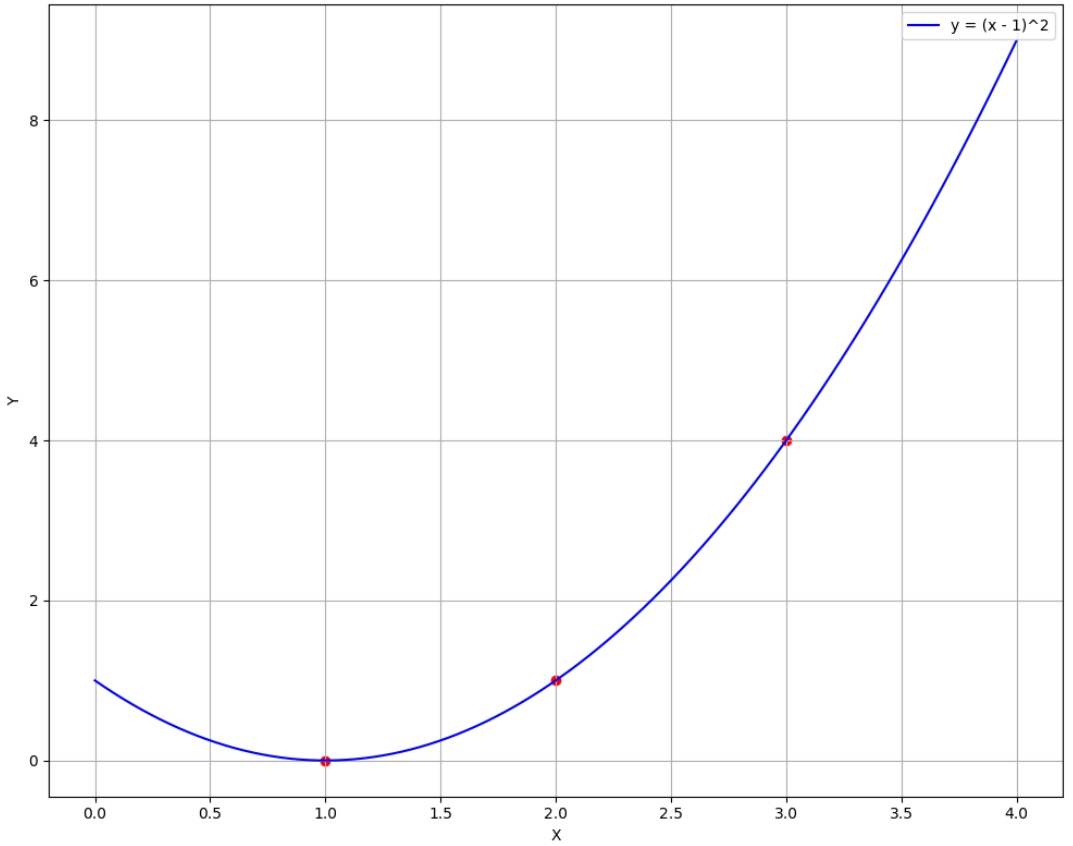
Case Study ------ More…

In our problem, when we achieve the optimal solution pair, the loss function is still positive.

What if we relax the limitation of "linearity"? Can quadratic function perform better?

Yes!

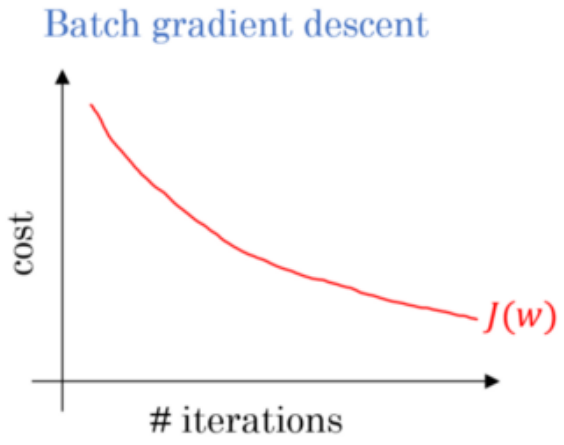
Besides, we have L=0! But… (Overfitting issue)

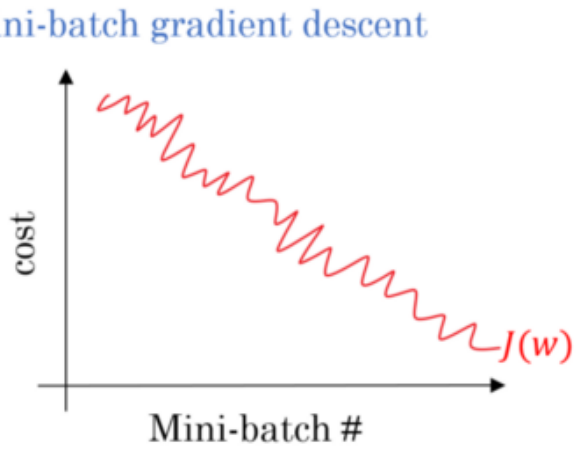


Case Study ------ More…

If we have 100,000,000,000,000 data points, can GD still work?

Theoretically, yes. But… The computation of gradients becomes extremely hard. Therefore, we sometimes use **mini-batch GD** as the surrogate method. At every time step, we randomly select some data in the dataset to compute the gradients.





Case Study ------ More…

Recommended course for more knowledge about ML: Machine Learning course by Andrew Ng.

Thanks!

Zhiyao Zhang

zhang.15178@osu.edu