Lecture Note: Mathematics for Machine Learning
Bài Giảng: Toán Học Cho Học Máy

Nguyễn Quản Bá Hồng*

Ngày 23 tháng 5 năm 2025

Tóm tắt nội dung

This text is a part of the series *Some Topics in Advanced STEM & Beyond*:
URL: https://nqbh.github.io/advanced_STEM/.
Latest version:

- *Lecture Note: Mathematics for Machine Learning – Bài Giảng: Toán Học Cho Học Máy.*
  PDF: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/machine_learning/lecture/NQBH_mathematics_for_machine_learning_lecture.pdf.
  TEX: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/machine_learning/lecture/NQBH_mathematics_for_machine_learning_lecture.tex.
- *Slide: Mathematics for Machine Learning – Slide Bài Giảng: Toán Học cho Học Máy.*
  PDF: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/machine_learning/slide/NQBH_mathematics_for_machine_learning_slide.pdf.
  TEX: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/machine_learning/slide/NQBH_mathematics_for_machine_learning_slide.tex.
- *Survey: Machine Learning – Khảo Sát: Học Máy.*
  PDF: URL: https://github.com/NQBH/draft/blob/master/machine_learning/NQBH_machine_learning.pdf.
  TEX: URL: https://github.com/NQBH/draft/blob/master/machine_learning/NQBH_machine_learning.tex.
- Codes:
  - Python for Machine Learning: https://github.com/NQBH/advanced_STEM_beyond/tree/main/machine_learning/Python.
  - C/C++ for Machine Learning:

Mục lục

1 Basic & Big Pictures **2**
1.1 Abbreviation, Conventions, & Notations 2
1.2 Main components 2
1.3 Energy functions – Các hàm năng lượng 3
1.4 Data as vectors 4
1.5 How to describe models 4
1.5.1 Models as functions – Mô hình như hàm số 5
1.5.2 Models as probability distributions – Mô hình như phân phối xác suất 5
1.5.3 Learning is finding parameters 5

2 Introduction to Supervised Learning – Giới Thiệu về Học Có Giám Sát **6**
2.1 Decision Theory 7
2.1.1 Supervised Learning Problems & Loss Functions 7
2.2 Empirical risk minimization – Giảm thiểu rủi ro thực nghiệm 8
2.3 Parameter estimation – Ước tính/lượng tham số 8
2.3.1 Maximum Likelihood Estimation (MLE) 8

3 Linear Regression – Hồi Quy Tuyến Tính **8**
3.1 Introduction to linear regression – Giới thiệu về hồi quy tuyến tính 9
3.2 A general mathematical setting 10

4 Overfitting – Quá Khớp **10**

*A scientist- & creative artist wannabe, a mathematics & computer science lecturer of Department of Artificial Intelligence & Data Science (AIDS), School of Technology (SOT), UMT Trường Đại học Quản lý & Công nghệ TP.HCM, Hồ Chí Minh City, Việt Nam.
E-mail: nguyenquanbahong@gmail.com & hong.nguyenquanba@umt.edu.vn. Website: https://nqbh.github.io/. GitHub: https://github.com/NQBH.

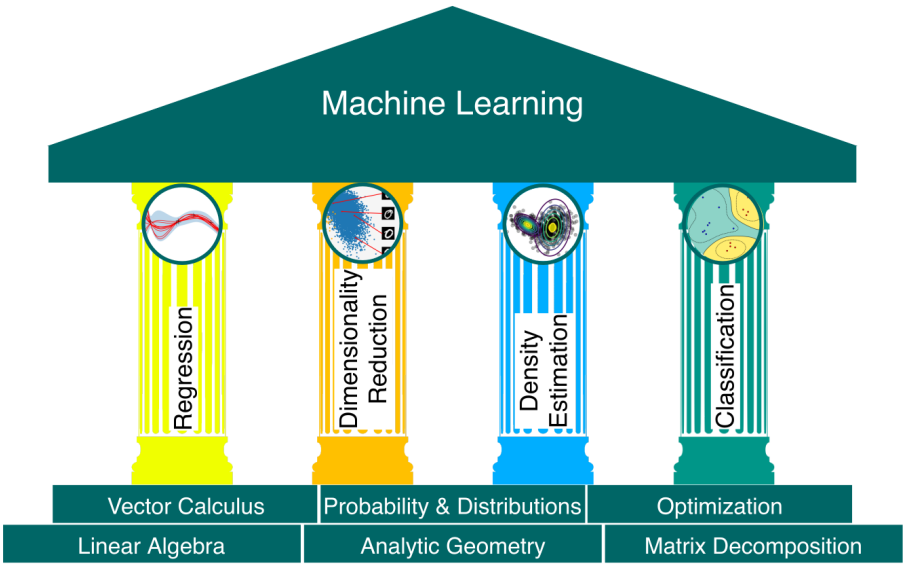
5 K Neighbors – K Lân Cận . . . 10
6 Phân Cụm K-Means . . . 10
7 Artificial Neural Networks (ANNs) – Mạng Neuron Nhân Tạo . . . 10
8 Perception Learning Algorithm – Thuật Toán Học Perceptron . . . 11
9 Logistic Regression – Hồi Quy Logistic . . . 11
10 Softmax Regression – Hồi Quy Softmax . . . 11
11 Deep Neural Networks & Backpropagation – Mạng Neuron Đa Tầng & Lan Truyền Ngược . . . 11
12 Miscellaneous . . . 11
12.1 Contributors . . . 11
Tài liệu . . . 11

1 Basic & Big Pictures

1.1 Abbreviation, Conventions, & Notations

- AI: Artificial Intelligence
- ANN: Artificial Neural Network
- DL: Deep Learning
- LLM: Large Language Model
- ML: Machine Learning

1.2 Main components



Hình 1: Foundations & 4 pillars of ML. Source: [DFO24, Fig. 1.1, p. 14].

- 3 major components of a ML system: data, models, & learning.
- 4 pillars of machine learning – 4 trụ cột của máy học:
  1. Regression – Hồi quy
  2. Dimensionality reduction – Giảm kích thước[1]
  3. Density estimation – Ước tính mật độ

[1] I was used to work in project ROMSOC: Reduced Order Modelling, Simulation & Optimization of Coupled Systems. Maybe "dimensionality reduction" is related to "reduced order modeling" in some sense of mathematics?

4. Classification – Phân loại

Question 1 (Main question of ML)**.** *What do we mean by "good" "models"?*

See, e.g., [DFO24, Chap. 8, Sect. 8.1: Data, Models, & Learning].

1.3 Energy functions – Các hàm năng lượng

Resources – Tài nguyên.

1. [Cho25]. Kyunghyun Cho. *Machine Learning: a Lecture Note*. Chap. 1: An Energy Function.

Definition 1 (Energy function, [Cho25], pp. 1–2)**.** *An* energy function*, or a* negative compatibility score *e assigns a real value to a pair of an observed instance & a latent instance (x, z) & is parameterized by a multidimensional vector $\boldsymbol{\theta}$: $e : \mathcal{X} \times \mathcal{Z} \times \Theta \to \mathbb{R}$, where $\mathcal{X}$ is a set of all possible observed instances, $\mathcal{Z}$ is a set of all possible latent instances, & Θ is a set of all possible parameter configurations.*

When the energy function is low (i.e., the compatibility is high), we say that a given pair (x, z) is highly preferred *given $\boldsymbol{\theta}$. When the energy function is high, unsurprising we say that the given pair is* not as preferred*.*

Định nghĩa 1 (Hàm năng lượng)**.** *Một* hàm năng lượng*, hoặc* điểm tương thích âm *e gán một giá trị thực cho một cặp trường hợp quan sát được & một trường hợp tiềm ẩn (x, z) & được tham số hóa bởi một vectơ đa chiều $\boldsymbol{\theta}$: $e : \mathcal{X} \times \mathcal{Z} \times \Theta \to \mathbb{R}$, trong đó $\mathcal{X}$ là tập hợp tất cả các trường hợp quan sát được có thể, $\mathcal{Z}$ là tập hợp tất cả các trường hợp tiềm ẩn có thể, & Θ là tập hợp tất cả các cấu hình tham số có thể.*

Khi hàm năng lượng thấp (tức là khả năng tương thích cao), chúng ta nói rằng một cặp (x, z) cho trước là được ưa thích cao *với $\boldsymbol{\theta}$. Khi hàm năng lượng cao, không có gì ngạc nhiên khi chúng ta nói rằng cặp cho trước là* không được ưa thích*.*

The latent observation z is, as the name suggests, not observed directly. It nevertheless plays an important role in capturing uncertainty. When we only observe x, but not z, we cannot fully determine how preferable x is. With a certain set of values of z, the energy may be low, while it may be high with other values of z. This gives us a sense of the uncertainty. E.g., we can compute both the mean & variance of the energy of an observed instance by

$$e_\mu(x, \boldsymbol{\theta}) \coloneqq \mathbb{E}[e(x, z, \boldsymbol{\theta})] = \sum_{z \in \mathcal{Z}} p(z) e(x, z, \boldsymbol{\theta}), \tag{1}$$

$$e_{\mathrm{v}}(x, \boldsymbol{\theta}) \coloneqq \mathbb{E}[(e(x, z, \boldsymbol{\theta}) - e_\mu(x, \boldsymbol{\theta}))^2]. \tag{2}$$

– Quan sát tiềm ẩn z, như tên gọi của nó, không được quan sát trực tiếp. Tuy nhiên, nó đóng vai trò quan trọng trong việc nắm bắt sự không chắc chắn. Khi chúng ta chỉ quan sát x, nhưng không quan sát z, chúng ta không thể xác định đầy đủ mức độ ưu tiên của x. Với một tập hợp các giá trị z nhất định, năng lượng có thể thấp, trong khi nó có thể cao với các giá trị z khác. Điều này giúp chúng ta có cảm giác về sự không chắc chắn. Ví dụ, chúng ta có thể tính toán cả giá trị trung bình & phương sai của năng lượng của một trường hợp quan sát được bằng (1) & (2), resp.

Given an energy function e & the parameter $\boldsymbol{\theta}$, we can derive a variety of paradigms in ML by minimizing the energy function w.r.t. different variables. E.g., let observation be partitioned into 2 parts; input & output & assume that there is no latent variable, i.e., $e([x, y], \emptyset, \boldsymbol{\theta})$. Given a new input x', we can solve the problem of *supervised learning* by

$$\hat{y} = \arg\min_{y \in \mathcal{Y}} e([x', y], \emptyset, \boldsymbol{\theta}),$$

where $\mathcal{Y}$ is the set of all possible outcomes y. When $\mathcal{Y}$ consists of discrete items, we call it *classification*. If y is a continuous variable, we call it *regression*.

– Với một hàm năng lượng e & tham số $\boldsymbol{\theta}$, chúng ta có thể suy ra nhiều mô hình khác nhau trong ML bằng cách giảm thiểu hàm năng lượng theo các biến khác nhau. Ví dụ, hãy phân vùng quan sát thành 2 phần; đầu vào & đầu ra & giả sử rằng không có biến tiềm ẩn, tức là $e([x, y], \emptyset, \boldsymbol{\theta})$. Với một đầu vào mới x', chúng ta có thể giải quyết vấn đề *học có giám sát* bằng

$$\hat{y} = \arg\min_{y \in \mathcal{Y}} e([x', y], \emptyset, \boldsymbol{\theta}),$$

trong đó $\mathcal{Y}$ là tập hợp tất cả các kết quả có thể y. Khi $\mathcal{Y}$ bao gồm các mục rời rạc, chúng ta gọi đó là *phân loại*. Nếu y là biến liên tục, chúng ta gọi nó là *hồi quy*.

When $\mathcal{Z}$ is a finite set of discrete items, a given energy function e defines the cluster assignment of an observation x, resulting in *clustering*:

$$\hat{z} = \arg\min_{z \in \mathcal{Z}} e(x, z, \boldsymbol{\theta}).$$

If z is a continuous variable, we would solve the same problem but call it *representation learning*.

– Khi $\mathcal{Z}$ là một tập hợp hữu hạn các mục rời rạc, một hàm năng lượng e cho trước sẽ xác định phép gán cụm của một quan sát x, dẫn đến *clustering*:

$$\hat{z} = \arg\min_{z \in \mathcal{Z}} e(x, z, \boldsymbol{\theta}).$$

Nếu z là một biến liên tục, chúng ta sẽ giải quyết cùng một bài toán nhưng gọi nó là *học biểu diễn*.

All these different paradigms effectively correspond to solving a minimization problem w.r.t. some subset of the inputs to the energy function e. I.e., given a partially-observed input, we infer the unobserved part that minimizes the energy function. This is often why people refer to using any ML model after training as *inference*.

– Tất cả các mô hình khác nhau này thực sự tương ứng với việc giải quyết một vấn đề tối thiểu hóa w.r.t. một số tập hợp con của các đầu vào cho hàm năng lượng e. Tức là, với một đầu vào được quan sát một phần, chúng ta suy ra phần không được quan sát làm tối thiểu hóa hàm năng lượng. Đây thường là lý do tại sao mọi người gọi việc sử dụng bất kỳ mô hình ML nào sau khi đào tạo là *suy luận*.

It is nontrivial to solve such a minimization problem. The level of difficulty depends on a variety of factors, including how the energy function is defined, the dimensionalities of the observed as well as latent variables as well sa the parameters themselves. We will consider different setups in which efficient & effective optimization algorithms are known & used for inference.

– Không dễ để giải quyết một bài toán tối thiểu hóa như vậy. Mức độ khó phụ thuộc vào nhiều yếu tố, bao gồm cách định nghĩa hàm năng lượng, chiều của các biến quan sát cũng như các biến tiềm ẩn cũng như các tham số. Chúng ta sẽ xem xét các thiết lập khác nhau trong đó các thuật toán tối ưu hóa hiệu quả & hiệu quả được biết & sử dụng để suy luận.

As the name 'ML' suggests, a bulk of ML is on estimating $\boldsymbol{\theta}$. Learning is not just $\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{x\sim p_{\text{data}}}[e(x, \emptyset, \boldsymbol{\theta})]$ when there is no latent variable, but, unfortunately, learning is not as easy, since we must ensure that the energy assigned to undesirable observation, i.e., $p_{\text{data}}(x) \downarrow$, must be relatively high. I.e., we must introduce an extra term that regularizes learning:

$$\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{x\sim p_{\text{data}}}[e(x, \emptyset, \boldsymbol{\theta}) - R(\boldsymbol{\theta})].$$

The choice of R must be made appropriately for each problem we solve, & we will learn how to design appropriate regularizers to ensure proper learning.

– Như tên gọi 'ML' gợi ý, phần lớn ML nằm ở việc ước tính $\boldsymbol{\theta}$. Học không chỉ là $\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{x\sim p_{\text{data}}}[e(x, \emptyset, \boldsymbol{\theta})]$ khi không có biến tiềm ẩn, nhưng thật không may, việc học không dễ dàng như vậy, vì chúng ta phải đảm bảo rằng năng lượng được gán cho quan sát không mong muốn, tức là $p_{\text{data}}(x) \downarrow$, phải tương đối cao. Tức là, chúng ta phải giới thiệu một thuật ngữ bổ sung để chính quy hóa việc học:

$$\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{x\sim p_{\text{data}}}[e(x, \emptyset, \boldsymbol{\theta}) - R(\boldsymbol{\theta})].$$

Việc lựa chọn R phải được thực hiện một cách phù hợp cho từng bài toán mà chúng ta giải quyết, & chúng ta sẽ học cách thiết kế các bộ chính quy hóa phù hợp để đảm bảo việc học tập phù hợp.

Of course it becomes even more involved when there are latent (unobserved) variables z, since it require us to solve the problem of inference simultaneously as well. This happens for problems e.g. clustering where the cluster assignment of each observation is unknown & factor analysis where latent factors are unknown in advance. We will learn how to interpret such latent variables & algorithms that allow us to estimate $\boldsymbol{\theta}$ in the absence of latent variables.

– Tất nhiên, nó trở nên phức tạp hơn khi có các biến tiềm ẩn (không quan sát được) z, vì nó đòi hỏi chúng ta phải giải quyết vấn đề suy luận đồng thời. Điều này xảy ra đối với các vấn đề ví dụ như phân cụm trong đó việc gán cụm cho mỗi quan sát là không xác định & phân tích nhân tố trong đó các nhân tố tiềm ẩn không xác định trước. Chúng ta sẽ học cách diễn giải các biến tiềm ẩn như vậy & các thuật toán cho phép chúng ta ước tính $\boldsymbol{\theta}$ khi không có các biến tiềm ẩn.

In summary, there are 3 aspects to every ML problem:

1. defining an energy function e (parametrization)
2. estimating the parameters $\boldsymbol{\theta}$ from data (learning)
3. inferring a missing part given an partial observation (inference).

Across these 3 steps sits 1 energy function, & once we obtain an energy function e, we can easily mix & match these steps from different paradigms of ML.

– Tóm lại, có 3 khía cạnh đối với mọi vấn đề ML:

1. định nghĩa hàm năng lượng e (tham số hóa)
2. ước tính các tham số $\boldsymbol{\theta}$ từ dữ liệu (học tập)
3. suy ra phần còn thiếu khi quan sát một phần (suy luận).

Trong 3 bước này có 1 hàm năng lượng, & khi chúng ta thu được hàm năng lượng e, chúng ta có thể dễ dàng kết hợp & khớp các bước này từ các mô hình ML khác nhau.

1.4 Data as vectors

Input data can be considered as a vector of features of the problem considered.

See, e.g., [DFO24, Chap. 8, Sect. 8.1.1: Data as Vectors].

1.5 How to describe models

2 major approaches to describe models in ML:

- a predictor as a function (i.e., non-probabilistic view of ML)
- a predictor as a probabilistic model (i.e., probabilistic view of ML).

Remark 1 (Cf. Deterministic world vs. Stochastic world)**.**

1.5.1 Models as functions – Mô hình như hàm số

See, e.g., [DFO24, Chap. 8, Sect. 8.1.2: Models as Functions].

A predictor is a function that, when given a particular input example produces an output:

$$\begin{aligned}\text{predictor} : \text{Input Space} &\to \text{Output Space},\\ \text{input example} &\mapsto \text{output} \coloneqq \text{predictor(input example)}.\end{aligned}$$

Input- & Output Spaces in this formula are usually vector spaces. If they are, we usually take Input Space $\coloneqq \mathbb{R}^{n_{\text{input}}}$, Output Space $\coloneqq \mathbb{R}^{n_{\text{output}}}$, or Input Space $\coloneqq \mathbb{C}^{n_{\text{input}}}$, Output Space $\coloneqq \mathbb{C}^{n_{\text{output}}}$, but each complex number can be decomposed as $z = \Re z + i\Im z$, $\forall z \in \mathbb{C}$, so it is equivalent to consider Input Space $\coloneqq \mathbb{R}^{2n_{\text{input}}}$, Output Space $\coloneqq \mathbb{R}^{2n_{\text{output}}}$ instead of Input Space $\coloneqq \mathbb{C}^{n_{\text{input}}}$, Output Space $\coloneqq \mathbb{C}^{n_{\text{output}}}$. Briefly, a pair of Euclidean spaces $(\mathbb{R}^{n_{\text{input}}}, \mathbb{R}^{n_{\text{output}}})$ is good enough to represent input- & output spaces. For simplicity, consider the output to be a single real number, i.e., a real-valued scalar output, which can be written as

$$f : \mathbb{R}^D \to \mathbb{R},$$

where the input vector $\mathbf{x}$ is D-dimensional (has D features), & the function f then applied to it (written as $f(\mathbf{x})$) returns a real number. In [DFO24], the authors did not considered the general case of all functions, which would involve the need for functional analysis (see some classical texts of Functional Analysis, e.g., [Alt16; Bre11; Rud91]). Instead, they considered the special case of linear functions

$$f(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{x} + \theta_0 = \boldsymbol{\theta} \cdot \mathbf{x} + \theta_0 = \sum_{i=1}^{D} \theta_i x_i + \theta_0, \text{ also } f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \mathbf{w} \cdot \mathbf{x} + b = \sum_{i=1}^{D} w_i x_i + b, \tag{linprd}$$

for unknown $\boldsymbol{\theta}, \theta_0$ (or more precisely, unknown but adjustable parameters) (or weights $\mathbf{w}$ & bias b). The linear restriction of possible forms of predictors means that basic concepts of Linear Algebra & Analytic Geometry suffice for precisely stating the notion of a predictor for the non-probabilistic view of machine learning. Linear functions strike a good balance between the generality of the problems that can be solved & the amount of background mathematics, especially for computer science-major students, that is needed.

Remark 2 (Pros & cons of linear world vs. nonlinear world – Ưu & nhược điểm của thế giới tuyến tính so với thế giới phi tuyến tính)**.**

(i) Cf: Linear Functional Analysis vs. Nonlinear Functional Analysis.

(ii) Cf: Linear Partial Differential Equations (linear PDEs) vs. Nonlinear Partial Differential Equations (nonlinear PDEs).

Philosophically & mathematically speaking, in any case, some compromises that are good enough are needed to be established for particular goals.

1.5.2 Models as probability distributions – Mô hình như phân phối xác suất

See, e.g., [DFO24, Chap. 8, Sect. 8.1.3: Models as Probability Distributions].

We often consider data to be noisy observations of some true underlying effect, i.e.,

$$\text{noise data} \coloneqq \text{true data} + \text{noise}, \ \mathbf{x}_{\text{noise}} \coloneqq \mathbf{x}_{\text{true}} + \delta_{\text{noise}},$$

& hope that by apply ML we can identity the signal from the noise (denoise), which requires

- a language for quantifying the effect of noise (probability theory provides a language for quantifying uncertainty), &
- predictors being able to express some sort of uncertainty, e.g., to quantify the confidence we have about the value of the prediction for a particular test data point.

Idea. Instead of considering a predictor as a single function, we could consider predictors to be probabilistic models, i.e., models describing the distribution of possible functions. In [DFO24], the authors limited themselves to the special case of distributions with finite-dimensional parameters, which allows them to describe probabilistic models without needing stochastic processes & random measures. For this special case, we can think about probabilistic models as multivariate probability distributions, which already allow for a rich class of models.

1.5.3 Learning is finding parameters

See, e.g., [DFO24, Chap. 8, Sect. 8.1.4: Learning is Finding Parameters].

The goal of learning is to find a model & its corresponding parameters such that the resulting predictor will perform well on unseen data. There are conceptually 3 distinct algorithmic phases when discussing ML algorithms:

1. Prediction or inference – Dự đoán hoặc suy luận
2. Training or parameter estimation – Đào tạo hoặc ước tính tham số

3. Hyperparameter tuning or model selection – Điều chỉnh siêu tham số hoặc lựa chọn mô hình

The prediction phase is when we use a trained predictor on previously unseen test data, i.e., the parameters & model choice is already fixed & the predictor is applied to new vectors representing new input data points. Recall that we consider 2 schools of ML, corresponding to whether the predictor is a function or a probabilistic model. When we have a probabilistic model (see, e.g., [DFO24, Sect. 8.4]) the prediction phase is called *inference*.

Remark 3 (Usage of word "inference" – Các cách sử dụng từ "suy luận")**.** *"Unfortunately, there is no agreed upon naming for different algorithmic phases. The word "inference" is sometimes also used to mean parameter estimation of a probabilistic model, & less often may be also used to mean prediction for non-probabilistic models." – [DFO24, Rmk, p. 257])*

– Thật không may, không có cách đặt tên thống nhất cho các giai đoạn thuật toán khác nhau. Từ "suy luận" đôi khi cũng được dùng để chỉ ước tính tham số của một mô hình xác suất, & ít thường xuyên hơn có thể được dùng để chỉ dự đoán cho các mô hình không xác suất.

We are interested in learning a model based on data s.t. it performs well on future data. It is not enough for the model to only fit the training data well, the predictor needs to perform well on unseen data. We simulate the behavior of our predictor on future unseen data using *cross-validation*. To check the goal of performing well on unseen data, we will need to balance between fitting well on training data & finding "simple" explanations of the phenomenon. This trade-off is achieved using *regularization* or by *adding a prior*. In philosophy, this is considered to be neither induction nor deduction, but is called *abduction*. According to the *Stanford Encyclopedia of Philosophy, abduction* is the process of inference to the best explanation.

We often need to make high-level modeling decisions about the structure of the predictor, e.g. the number of components to use or the class of probability distributions to consider. The choice of the number of components is an example of a *hyperparameter*, & this choice can affect the performance of the model significantly. The problem of choosing among different models is called *model selection* (see, e.g., [DFO24, Sect. 8.6: Model Selection]). For non-probabilistic models, model selection is often done using *nested cross-validation* (see, e.g., [DFO24, Subsect. 8.6.1: Nested Cross-Validation]). We also use model selection to choose hyperparameters of our model.

Remark 4 (Cf. Parameter vs. Hyperparameter)**.** *The distinction between parameters & hyperparameters is somewhat arbitrary, & is mostly driven by the distinction between what can be numerically optimized vs. what needs to use search techniques. Another way to consider the distinction is to consider parameters as the explicit parameters of a probabilistic model, & to consider hyperparameters (higher-level parameters) as parameters that control the distribution of these explicit parameters.*

3 flavors of ML:

1. Empirical risk minimization, see, e.g., [DFO24, Sect. 8.2: Empirical Risk Minimization];

2. Principle of maximum likelihood, see, e.g., [DFO24, Sect. 8.3: Parameter Estimation];

3. Probabilistic modeling, see, e.g., [DFO24, Sect. 8.4: Probabilistic Modeling & Inference].

The learning part of ML boils down to estimating parameters based on training data.

2 Introduction to Supervised Learning – Giới Thiệu về Học Có Giám Sát

References – Tài nguyên.

1. [Bac24]. Francis Bach. *Learning Theory from First Principles*. Chap. 2: Introduction to Supervised Learning.

Main goal. Give some observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, \dots, n$, of inputs/outputs, features/labels, covariates/responses (which are referred to as "training data"), main goal of supervised learning is to predict a new $y \in \mathcal{Y}$ given a new previously unseen $x \in \mathcal{X}$. Unobserved data are usually referred to as "testing data".

– Đưa ra 1 số quan sát $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, \dots, n$, của các đầu vào/đầu ra, các nhãn/tính năng, các phản hồi của biến phụ thuộc/(được gọi là "dữ liệu huấn luyện/đào tạo"), mục tiêu chính của học có giám sát là dự đoán 1 $y \in \mathcal{Y}$ mới khi biết trước 1 $x \in \mathcal{X}$ mới chưa từng thấy. Dữ liệu chưa quan sát thường được gọi là "dữ liệu thử nghiệm".

Examples. Supervised learning is used in many areas of science, engineering, & industry. There are thus many examples where $\mathcal{X}, \mathcal{Y}$ can be very diverse:

- **Inputs** $x \in \mathcal{X}$: They can be images, sounds, videos, text documents, proteins, sequences of DNA bases, web pages, social network activities, sensors from industry, financial time series, etc. Set $\mathcal{X}$ may thus have a variety of structures that can be leveraged. All learning methods presented in this textbook will use at 1 point a vector space representation of inputs, either by building an explicit mapping from $\mathcal{X}$ to a vector space, e.g., $\mathbb{R}^d$, or implicitly by using a notion of pairwise dissimilarity or similarity between pairs of inputs. Choice of these representations is highly domain-dependent. However, note:

  - common topologies are encountered in many diverse areas (e.g. sequences or 2D or 3D objects), & thus common tools are used, &
  - learning these representations is an active area of research.

  In this textbook, will primarily consider that inputs are d-dimensional vectors, with d potentially large, up to 10^6 or 10^9.

- **Outputs** $y \in \mathcal{Y}$. Most classical examples are binary labels $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{\pm 1\}$, multicategory classification problems with $\mathcal{Y} = \{1, \dots, k\}$, & classical regression with real responses/outputs $\mathcal{Y} = \mathbb{R}$. These will be main examples examined in most of book. Note, however: most of concepts extend to more general *structured prediction* setup, where more general structured outputs (e.g., graph prediction, visual scene analysis, source separation, ranking) can be considered.

Why difficult? Supervised learning is difficult (& thus interesting) for a variety of reasons:

- Label y may not be a deterministic function of x: Given $x \in \mathcal{X}$, outputs are noisy, i.e., y is a random function of x. When $y \in \mathbb{R}$, will often make simplifying "additive noise" assumption: $y = f(x) + \varepsilon$ with some zero-mean noise ε, but in general, only assume: there is a conditional distribution of y given x. This stochasticity is typically due to diverging views between labelers or dependence on random external unobserved quantities (i.e., $y = f(x, z)$, with z random & not observed, which is common, e.g., in medical applications, where need to predict a future occurrence of a disease based on limited information about patients).
- Prediction function f may be quite complex, highly nonlinear when $\mathcal{X}$ is a vector space, & even hard to define when $\mathcal{X}$ is not a vector space.
- Only a few x's are observed: thus need interpolation & potentially extrapolation (diagram for an illustration for $\mathcal{X} = \mathcal{Y} = \mathbb{R}$), & therefore overfitting (predicting well on training data but not as well on testing data) is always a possibility.

  Moreover, training observations may not be uniformly distributed in $\mathcal{X}$. In this book, they will be assumed to be random, but some analyzes will rely on deterministically located inputs to simplify some theoretical arguments.
- Input space $\mathcal{X}$ may be very large (i.e., with high dimension when this is a vector space). This leads to both computational issues (scalability) & statistical issues (generalization to unseen data). One usually refers to this problem as *curse of dimensionality*.
- There may be a weak link between training & testing distributions. I.e., data at training time can have different characteristics than data at testing time.
- Criterion for performance is not always well defined.

2.1 Decision Theory

See, e.g., Wikipedia/decision theory.

Question 2 (Optimal performance)**.** *What is optimal performance, regardless of the finiteness of the training data? Or what should be done if we have a perfect knowledge of the underlying probability distribution of the data?*

2.1.1 Supervised Learning Problems & Loss Functions

We consider a *loss function* $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ (often $\mathbb{R}_+ := (0, \infty)$), where $l(y, z)$ is the loss of predicting z while the true label is y. Some related research communities (e.g., economics) use the concept of "utility" which is then maximized.

Meaning of loss function. Hàm mất mát $l(y, z)$ đóng vai trò như 1 phép đo khoảng cách (a measurement of distances) từ nhãn đúng đến giá trị dự đoán thu được bởi thuật toán học máy.

The loss function only concerns the output space $\mathcal{Y}$ independent of the input space $\mathbf{X}$ (check its definition!).

Example 1.

(i) **Binary classification.** $\mathcal{Y} = \{0, 1\}$ *(or often* $\mathcal{Y} = \{\pm 1\}$*, or, less often, when seen as a subcase of multicategory situation* $\mathcal{Y} = \{1, 2\}$*); the "0–1 loss" defined as*

$$l(y, z) = 1_{y \neq z} = \begin{cases} 0 & \text{if } y = z \text{ (no mistake)}, \\ 1 & \text{otherwise (mistake)}. \end{cases}$$

is the most commonly used.

(ii) **Multicategory classification.** $\mathcal{Y} = \{1, \dots, k\}$, $l(y, z) = 1_{y \neq z}$ *(0–1 loss).*

(iii) **Regression.** $\mathcal{Y} = \mathbb{R}$, $l(y, z) = (y - z)^2$ *(square loss). The absolute loss* $l(y, z) = |y - z|$ *is often used for robust estimation (đánh giá/ước tính mạnh mẽ) (since the penalty for large errors is smaller).*

(iv) **Structured prediction.** *There are many practical problems where* $\mathcal{Y}$ *is more complicated, with associated algorithms & theoretical results. E.g., when* $\mathcal{Y} = \{0, 1\}^k$ *(leading to* multilabel classification*), the* Hamming loss $l(y, z) = \sum_{j=1}^k 1_{y_j \neq z_j}$ *is commonly used; also, ranking problems involve losses on permutations, see, e.g., [Bac24, Chap. 13: Structured Prediction].*

Assumption 1 (Loss function l)**.** *Assume that the loss function is given to us.*

In practice, the final user imposes the loss function, as this is how models will be evaluated. Clearly, a single real number may not be enough to characterize the entire prediction behavior. E.g., in binary classification, there are 2 types of errors, false positives & false negatives, which can be considered simultaneously. Since we now have 2 performance measures (& in any case with ≥ 2 performance measures), we typically need a curve to characterize the performance of a prediction function.

2.2 Empirical risk minimization – Giảm thiểu rủi ro thực nghiệm

See, e.g., [DFO24, Sect. 8.2: Empirical Risk Minimization]. In this subsection, we consider the case of a predictor that is a function.

2.3 Parameter estimation – Ước tính/lượng tham số

See, e.g., [DFO24, Sect. 8.3: Parameter Estimation], [ABT18]. In the previous framework of empirical risk minimization, we did not explicitly model our problem using probability distributions. In this subsection, we learn how to use probability distributions to model our uncertainty due to the observation process & our uncertainty in the parameters of our predictors.

Remark 5 (Relation with Bayesian inverse problem?)**.** *See, e.g., [IJ15].*

2.3.1 Maximum Likelihood Estimation (MLE)

...For data represented by a random variable $\mathbf{x}$ & for a family of probability densities $p(\mathbf{x}|\boldsymbol{\theta})$ parametrized by $\boldsymbol{\theta}$, the *negative log-likelihood* is given by

$$\mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta}) = -\log p(\mathbf{x}|\boldsymbol{\theta}). \tag{3}$$

3 Linear Regression – Hồi Quy Tuyến Tính

References – Tài nguyên.

1. [Bac24]. Francis Bach. *Learning Theory from First Principles*. Chap. 3: Linear Least-Squares Regression.
2. [DFO24]. Marc Peter Deisenroth, A. Aldo Faisal, Cheng Soon Ong. *Mathematics for Machine Learning*. Chap. 9: Linear Regression.
3. DeepLearning.AI's resources: https://www.deeplearning.ai/resources/.
4. Andrew Ng's Machine Learning Specialization slides: https://home-wordpress.deeplearning.ai/wp-content/uploads/2022/03/andrew-ng-machine-learning-yearning.pdf.
5. [Tiệ25]. Vũ Hữu Tiệp. *Machine Learning Cơ Bản*. Chap. 7: Hồi Quy Tuyến Tính.
6. Wikipedia: Wikipedia/linear function. Wikipedia/linear regression.

A general idea – 1 ý tưởng tổng quát. Ý tưởng này có vẻ được mượn từ Lý Thuyết Thống Kê Có Tham Số (Parametric Statistics Theory), see, e.g., Wikipedia/parametric statistics.

For $d \in \mathbb{N}^\star$ pairs $(\mathbf{x}_i, y_i)_{i=1}^d$ of input-output, we represent some (mathematically reasonable) hypotheses h about the data using the parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_{\mathrm{p}})$ where $\mathrm{p} = \text{size}\,\boldsymbol{\theta}$ is the size/dimension (kích thước/số chiều) of the vector of parameters, i.e., the total number of parameters in the chosen hypotheses h. If the data is correctly predicted according to hypothesis $h_{\boldsymbol{\theta}}$, then $y \approx h_{\boldsymbol{\theta}}(\mathbf{x})$.

Example 2 (Some examples of hypothesis function $h_{\boldsymbol{\theta}}$ – Vài ví dụ về hàm giả thiết $h_{\boldsymbol{\theta}}$)**.**

(i) (Simplest: Linear function/affine mapping) *We assume $h_{\boldsymbol{\theta}}$ be a linear function, i.e., $h_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x$ for $d = 1$ & $h_{\boldsymbol{\theta}}(\mathbf{x}) = \theta_0 + [\theta_1, \theta_2, \dots, \theta_d] \cdot \mathbf{x}$ for $d \geq 2$.*

(ii) (Polynomial)*: When $d = 1$, we can assume $h_{\boldsymbol{\theta}} \in \mathbb{R}[\mathbf{x}]$ or even $h_{\boldsymbol{\theta}} \in \mathbb{C}[\mathbf{x}]$, i.e., a polynomial with real/complex coefficients in the variable x:*

$$h_{\boldsymbol{\theta}}(x) = \sum_{i=0}^{\mathrm{p}} \theta_i x^i = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_{\mathrm{p}} x^{\mathrm{p}}.$$

(iii) (Hàm phân thức) *When $d = 1$, we can assume*

$$h_{\boldsymbol{\theta}}(x) = \frac{\sum_{i=0}^{m} \theta_i x^i}{\sum_{m+1}^{\mathrm{p}} \theta_i x^{i-m-1}} = \frac{\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_m x^m}{\theta_{m+1} + \theta_{m+2} x + \cdots + \theta_{\mathrm{p}} x^{p-m-1}}.$$

(iv) (Hàm căn thức) *We can assume*

$$h_{\boldsymbol{\theta}}(x) = \sqrt[n]{\sum_{i=0}^{\mathrm{p}} \theta_i x^i} = \sqrt[n]{\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_{\mathrm{p}} x^{\mathrm{p}}},$$

$$h_{\boldsymbol{\theta}}(x) = \sqrt[n]{\frac{\sum_{i=0}^{m} \theta_i x^i}{\sum_{m+1}^{\mathrm{p}} \theta_i x^{i-m-1}}} = \sqrt[n]{\frac{\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_m x^m}{\theta_{m+1} + \theta_{m+2} x + \cdots + \theta_{\mathrm{p}} x^{\mathrm{p}-m-1}}},$$

for a given (fixed) $n \in \mathbb{R}$ or an adjustable n, i.e., n itself is also a component of $\boldsymbol{\theta}$.

(v) (Elementary function – Hàm sơ cấp) *We can assume* $h_{\boldsymbol{\theta}}(\mathbf{x})$ *to be an elementary function of the variable* $\mathbf{x}$. *For elementary functions, see, e.g., Wikipedia/ elementary function.*

A typical learning algorithm finds the best hypothesis $h_{\boldsymbol{\theta}}$ for the training set $(\mathbf{x}_i, y_i)_{i=1}^N$. We can then estimate the values of y for the test set $(\mathbf{x}_i, y_i)_{i=N+1}^d$ using the "optimal" hypothesis $h_{\boldsymbol{\theta}}(\mathbf{x})$ obtained from our learning algorithm. In particular, if $h_{\boldsymbol{\theta}}$ is a linear function of $x \in \mathbb{R}$, this procedure is called *linear regression.*

3.1 Introduction to linear regression – Giới thiệu về hồi quy tuyến tính

1st impression. Hồi quy tuyến tính (linear regression) là:

- 1 thuật toán hồi quy mà đầu ra là 1 hàm số tuyến tính (linear function, i.e., $y = ax + b$, $a, b \in \mathbb{R}$, $a \neq 0$) của đầu vào. A brief notation:
$$\text{output} = \begin{cases} \texttt{linear\_function}(\text{input}) = a\,\text{input} + b & \text{if } d = 1, \\ \texttt{linear\_function}(\mathbf{inputs}) = \mathbf{w} \cdot \mathbf{inputs} + b & \text{if } d \geq 2, \end{cases}$$
- Thuật toán đơn giản nhất trong nhóm các thuật toán học có giám sát (supervised learning algorithms).

Problem 1 (Housing prices – giá cả nhà đất/bất động sản)**.** *Let* $m \in \mathbb{N}^\star$. *Suppose we have a 2-column table whose the 1st column consists of sizes of* m *houses:* $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ *& the 2nd one consists of their corresponding housing prices* $\mathbf{y} = (y_1, y_2, \ldots, y_m)$. *Here* m *is the number of training examples,* $\mathbf{x}$ *is the vector of input variable or features (size), &* $\mathbf{y}$ *is the vector of output variables or target variables (price).*

Bài toán 1 (Housing price estimation – Ước lượng giá nhà, [Tiệ25], Sect. 7.1, p. 100)**.** *Xét bài toán ước lượng giá của 1 căn phòng rộng* x_1 m^2, *có* x_2 *phòng ngủ, & cách trung tâm thành phố* x_3 *km. Giả sử có 1 tập dữ diệu của* $N = 10000$ *căn nhà trong thành phố đó. Liệu có thể dự đoán được giá* y *của 1 căn nhà mới (i.e., khác* N *căn nhà đã có dữ liệu) thông qua 3 thông số về diện tích* $x_1 \in (0, \infty)$, *số phòng ngủ* $x_2 \in \mathbb{N}$, *& khoảng cách tới trung tâm thành phố* $x_3 \in [0, \infty)$*?*

Mathematical notations – Ký hiệu toán học. Đặt $\mathbf{x} = [x_1, x_2, x_3]^\top \in (0, \infty) \times \mathbb{N} \times [0, \infty)$ là 1 vector cột chứa dữ liệu đầu vào (inputs) gồm 3 thông số diện tích x_1, số phòng ngủ x_2, & khoảng cách đến trung tâm thành phố x_3; $\mathbf{x}$ được gọi là *vector đặc trưng*. Đặt giá nhà (đầu ra/output) $y \in (0, \infty)$.

Reasoning – Biện luận. Nếu diện tích nhà càng lớn thì giá nhà càng cao: $x_1 \uparrow \Rightarrow y \uparrow$, nếu nhà có càng nhiều phòng ngủ thì giá nhà càng cao: $x_2 \uparrow \Rightarrow y \uparrow$, & nếu nhà càng ở gần trung tâm thành phố thì giá nhà càng cao: $x_3 \downarrow 0 \Rightarrow y \uparrow$ (why? economics!). Có thể sử dụng mô hình đầu ra là 1 hàm tuyến tính đơn giản của đầu vào:

$$y \approx \hat{y} := f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_3 x_3 = \sum_{i=1}^{3} w_i x_i = \mathbf{x}^\top \mathbf{w} = \mathbf{w}^\top \mathbf{x} = \mathbf{x} \cdot \mathbf{w} = \langle \mathbf{x}, \mathbf{w} \rangle, \tag{lnr}$$

với $\mathbf{w} = [w_1, w_2, w_2]^\top$ là *vector trọng số* (weight vector) với $w_1, w_2 \in (0, \infty), w_3 \in (-\infty, 0)$ (why?) cần tìm. Mối quan hệ cho bởi (lnr) là 1 mối quan hệ tuyến tính.

Question 3 (Regression factor)**.** *Why is it called* regression*?*

Bản chất. 2 bài toán dự đoán giá nhà trên là 2 bài toán dự đoán giá trị đầu ra dựa trên vector đặc trưng đầu vào. Ngoài ra, giá trị của đầu ra y có thể nhận rất nhiều giá trị thực dương khác nhau. Nên ta cần tính đi tính lại, tính kết quả đầu ra mới dựa trên các kết quả đầu ra cũ, để nhận được 1 kết quả được xem là tối ưu theo nghĩa nào đó. Vì vậy, đây cũng là 1 bài toán hồi quy. Kết hợp bản chất hồi quy với mối quan hê tuyến tính $\hat{y} = \mathbf{x}^\top \mathbf{w}$ cho ra tên gọi *hồi quy tuyến tính.* Về hàm tuyến tính, see, e.g., Wikipedia/linear function.

Some reasons to study linear least-squares regression. From [Bac24, Sect. 3.1, pp. 45–46].

- Linear least-squares regression already captures many of the concepts in learning theory, e.g. bias-variance trade-off, as well as dependence of generalization performance on the underlying dimension of the problem with no regularization, or on dimensionless quantities when regularization is added.

  – Hồi quy tuyến tính bình phương nhỏ nhất đã nắm bắt được nhiều khái niệm trong lý thuyết học tập, ví dụ như sự đánh đổi giữa độ lệch và phương sai, cũng như sự phụ thuộc của hiệu suất tổng quát vào chiều cơ bản của vấn đề không có chính quy hóa hoặc vào các đại lượng không có thứ nguyên khi chính quy hóa được thêm vào.

- Because of its simplicity, many results can be easily derived without the need for complicated mathematics, both in terms of algorithms & statistical analysis (simple linear algebra for the simplest results in the fixed design setting).

  – Do tính đơn giản của nó, nhiều kết quả có thể dễ dàng thu được mà không cần đến các phép toán phức tạp, cả về mặt thuật toán & phân tích thống kê (đại số tuyến tính đơn giản cho kết quả đơn giản nhất trong bối cảnh thiết kế cố định).

- Using nonlinear features, it can lead to arbitrary nonlinear predictions.

  – Sử dụng các tính năng phi tuyến tính, nó có thể dẫn đến những dự đoán phi tuyến tính tùy ý.

Question 4 (Optimal choice of parameters $\boldsymbol{\theta}$)**.** *How to choose parameters* $\boldsymbol{\theta}$ *"optimally"? "Optimal" in which sense?*
– Làm thế nào để chọn tham số $\boldsymbol{\theta}$ *"tối ưu"? "Tối ưu" theo nghĩa nào?*

3.2 A general mathematical setting

Tổng quát, nếu mỗi điểm dữ liệu được mô tả bởi 1 vector đặc trưng d chiều $x \in \mathbb{R}^d$, hàm dự đoán đầu ra được viết dưới dạng

$$y = \mathbf{x}^\top \mathbf{w} = \mathbf{x} \cdot \mathbf{w} = \sum_{i=1}^{d} w_i x_i = w_1 x_1 + \cdots + w_d x_d.$$

Nếu tính thêm bias (e.g., tiện đặt cọc, chi phí phát sinh, giảm giá, voucher discount, etc.) $b \in \mathbb{R}$ thì thêm vào thành:

$$y = \mathbf{x}^\top \mathbf{w} + b = \mathbf{x} \cdot \mathbf{w} + b = \sum_{i=1}^{d} w_i x_i + b = w_1 x_1 + \cdots + w_d x_d + b.$$

Với bài toán hồi quy nói chung, ta cần cực tiểu sự sai khác e (error) giữa đầu ra thực sự y & đầu ra dự đoán $\hat{y}$ với e thường được chọn khoảng cách giữa 2 điểm y & $\hat{y}$, e.g.:

$$\frac{1}{2}e^2 = \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - \mathbf{x}^\top \mathbf{w} - b)^2.$$

(Hệ số $\frac{1}{2}$ đóng vai trò là scaling factor, giúp khi tính đạo hàm thì sẽ đơn giản được hệ số, thật vậy: vì $\left(\frac{1}{2}x^2\right)' = x$ nên hệ số $\frac{1}{2}$ sẽ bị triệ tiêu khi lấy đạo hàm của e theo tham số mô hình $\mathbf{w}$.) Các cách chọn sai số e khác:

1. $e = y - \hat{y}$: Khuyết điểm: $y - \hat{y}$ có thể âm, do đó việc cực tiểu hóa e không có ý nghĩa.
2. $e = |y - \hat{y}|$: Khuyết điểm: Hàm trị tuyệt đối (absolute value function) $f : \mathbb{R} \to [0, \infty)$, $f(x) = |x|$ tuy liên tục nhưng không khả vi tại gốc tọa độ (why?), không thuận tiện cho việc tối ưu.
3. $e = \frac{1}{p}|y - \hat{y}|^p$ với $p \in (1, \infty)$. Hàm số $f(x) = \frac{1}{p}x^p$ có đạo hàm $f'(x) = |x|^{p-1}$. Khuyết điểm: Cần nhiều tính toán hơn số với hàm bình phương, i.e., khi $p = 2$.

Hàm mất mát đóng vai trò là tiêu chí cho việc tối ưu mô hình: việc tìm mô hình tốt nhất/tối ưu nhất (best/optimal) tương đương với việc tìm $\mathbf{w}$ để cực tiểu hàm số:

$$L(\mathbf{w}) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \mathbf{x}_i^\top \mathbf{w})^2.$$

Optimization problem – Bài toán tối ưu:

$$\min_{\mathbf{w}} L(\mathbf{w}).$$

Nếu bài toán có nghiệm duy nhất, ký hiệu:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w}).$$

4 Overfitting – Quá Khớp

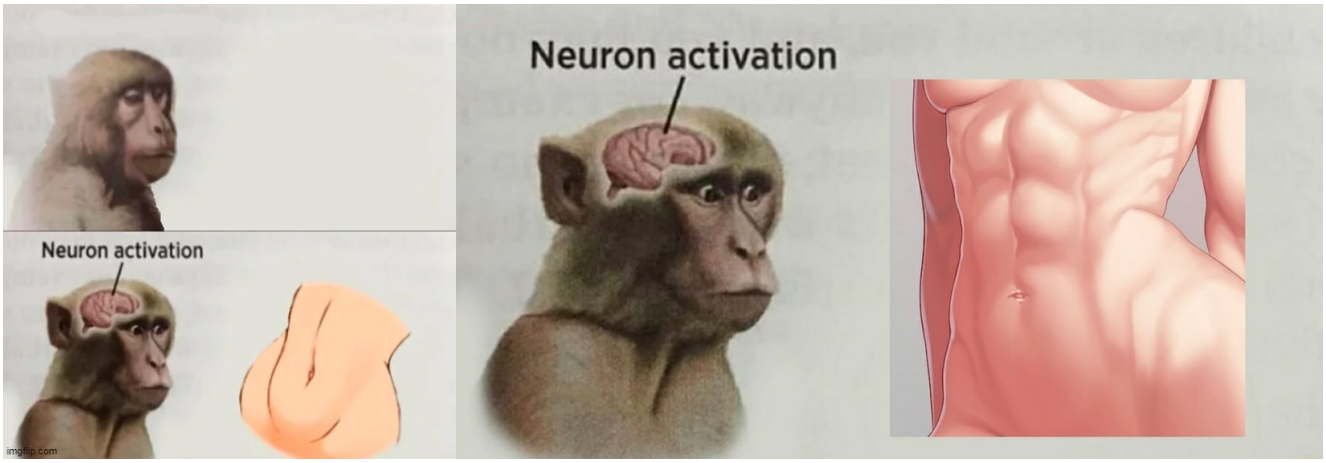
5 K Neighbors – K Lân Cận

6 Phân Cụm K-Means

7 Artificial Neural Networks (ANNs) – Mạng Neuron Nhân Tạo

Distinguish: Artificial Neural Network (ANN) vs. Biological Neural Network (BNN), see, e.g., Wikipedia/neural network (machine learning) vs. Wikipedia/neural network (biology), Psychology Wiki/biological neural network.

Definition 2 (Activation function $\sigma(\cdot)$)**.**



Hình 2: A typical monkey's (maybe man also?) biological neural networks (BNN) gets activated by 2D/anime girl's strong abs.

8 Perception Learning Algorithm – Thuật Toán Học Perceptron

9 Logistic Regression – Hồi Quy Logistic

10 Softmax Regression – Hồi Quy Softmax

11 Deep Neural Networks & Backpropagation – Mạng Neuron Đa Tầng & Lan Truyền Ngược

12 Miscellaneous

12.1 Contributors

Tài liệu

[ABT18] Richard C. Aster, Brian Borchers, and Clifford H. Thurber. *Parameter estimation and inverse problems*. Third. Elsevier/Academic Press, Amsterdam, 2018, pp. xi+392. ISBN: 978-0-12-804651-7. DOI: 10.1016/C2015-0-02458-3. URL: https://doi.org/10.1016/C2015-0-02458-3.

[Alt16] Hans Wilhelm Alt. *Linear functional analysis*. Universitext. An application-oriented introduction, Translated from the German edition by Robert Nürnberg. Springer-Verlag London, Ltd., London, 2016, pp. xii+435. ISBN: 978-1-4471-7279-6; 978-1-4471-7280-2. DOI: 10.1007/978-1-4471-7280-2. URL: https://doi.org/10.1007/978-1-4471-7280-2.

[Bac24] Francis Bach. "Learning Theory from First Principles". In: Adaptive Computation and Machine Learning series (2024), p. 496.

[Bre11] Haim Brezis. *Functional analysis, Sobolev spaces and partial differential equations*. Universitext. Springer, New York, 2011, pp. xiv+599. ISBN: 978-0-387-70913-0.

[Cho25] Kyunghyun Cho. *Machine Learning: a Lecture Note*. 1st version. arXiv, 2025, p. 107.

[DFO24] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. 1st edition. Cambridge University Press, 2024, pp. iii+411.

[IJ15] Kazufumi Ito and Bangti Jin. *Inverse problems*. Vol. 22. Series on Applied Mathematics. Tikhonov theory and algorithms. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2015, pp. x+318. ISBN: 978-981-4596-19-0.

[Rud91] Walter Rudin. *Functional analysis*. Second. International Series in Pure and Applied Mathematics. McGraw-Hill, Inc., New York, 1991, pp. xviii+424. ISBN: 0-07-054236-8.

[Tiệ25] Vũ Khắc Tiệp. *Machine Learning Cơ Bản*. 2025, p. 422.