

MECH70039 - Data Science and Digitalization in the Energy Sector

Introduction to Machine Learning

with Applications to Energy Systems

Dr. Wangkun Xu
wangkun.xu18@imperial.ac.uk

Department of EEE, Imperial College London

Feb. 26, 2026

Outline I

Introduction

- Course Overview
- Machine Learning: Why and What
- Taxonomy of Machine Learning
- Copyright and Reference

Supervised Learning

- Overview
- Linear Regression
- Logistic Regression
- Support Vector Machine

Unsupervised Learning

- Overview
- Principal Component Analysis
- k-means Clustering

Reinforcement Learning

- Overview Only

Outline II

Performance Measure

Overview

Model Capacity

Underfit and Overfit

Regularization

Hyperparameters and Validation Set

Hyperparameters

Validation Set

Special Topic One: Time Series Foundation Model

No Free Lunch Theorem

What is Foundation Model?

Time Series Foundation Model

Amazon Chronos

Special Topic Two: DATA? DATA!

Where and How to obtain the Data

Outline

Introduction

Course Overview

Machine Learning: Why and What

Taxonomy of Machine Learning

Copyright and Reference

Overview I

- ▶ 4 hour lectures to cover the *BASIC* idea of machine learning, deep learning, and a little bit on *foundation model*, with application to energy sector, especially power systems.
- ▶ Some coding tutorial in the afternoon. Please go to <https://github.com/xuwkk/intro2ml-coding-tutorial> to get the code!
- ▶ Hopefully this will become helpful for the project of the module and your thesis project.
- ▶ About myself: *Wangkun Xu*
 - ▶ Research Associate in the Department of EEE.
 - ▶ Obtained my PhD in 2024 at Imperial.
 - ▶ Has been TAing for this module over the last five year with Dr. Tuncer.
 - ▶ Main research direction on machine learning for robust and secure power system operations.
 - ▶ Website: https://xuwkk.github.io/wangkun_xu/.

Overview II

- ▶ GitHub: <https://github.com/xuwkk>.
- ▶ Contact me via email or MS Teams regarding on the course or any inquiry. Please share with me any typo in the slides and suggestions of this course!

Outline

Introduction

Course Overview

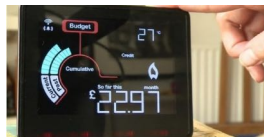
Machine Learning: Why and What

Taxonomy of Machine Learning

Copyright and Reference

Machine Learning: Why and What I

You probably know that the new trend of large penetration of **renewable energy** challenges the old model-based control room techniques.



Machine Learning: Why and What II

New challenge one: The power system with *power electronics/invertors* becomes a **black box** to the system operators. The inverters can cause new stability issues in the grid.

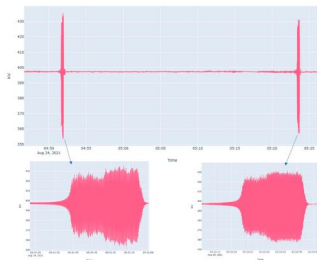


Figure: Sub-synchronous voltage oscillations (8Hz) recorded in GB transmission system, North Scotland, in 2021 [Source: NESO Report on Sub-synchronous oscillations in GB].

Machine Learning: Why and What III

New challenge two: Compared to traditional energy resources, renewable energy is more difficult to manage.

New opportunity: We are entering the era of *big data*. As the advances of *digitalization* in energy system, more and more data from the measurement unit, such as remote terminal units (RTUs), phasor measurement units (PMUs), and smart meters, and industrial sensors are readily available. But how to *extract useful information* from the data is one thing we need to consider.

Machine Learning: Why and What IV

To answer this: This requires automated data analysis methods, which is what *machine learning* (ML) provides.

Definition

(Less formal). In particular, we define machine learning as a set of methods that can automatically detect patterns in data and then use the uncovered patterns to predict future data or to perform other types of decision-making under uncertainty.




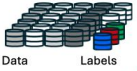

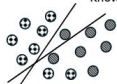
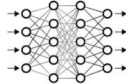
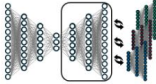
The essentials include:

- ▶ Data.
- ▶ Algorithm/Model.
- ▶ Evaluation/generalization.
- ▶ Automation.

And we will basically introduce ML based on the four factors.

Machine Learning: Why and What V

Machine learning is a very broad and practical topic. Therefore, this lecture only covers the most basic and ready-to-use ones that are mostly related to your project. Though many of the ideas can be used for more advanced topics.

	Expert Systems	Machine Learning	Deep Learning	Foundation Model
				
				
Model Size	Very small	Small	Large	Very Large
Features	Hand-designed	Hand-designed	Learnt	Learnt
Learning	None	Supervised	Supervised	Self-supervised
Data	Very few	Few labeled Data	Labeled data	Very large unlabeled + small labeled data
Adaptability	None	Little	Medium	Large

Machine Learning: Why and What VI

Figure: The evolution of AI (Source: [Hamann et al., 2024]). This course mainly focuses on the *machine learning* (with a little *time series foundation model*) part. The ideas are shared for deep learnings.

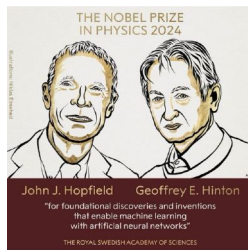
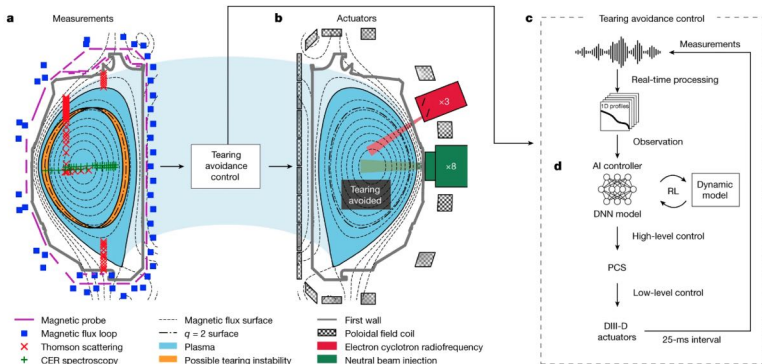


Figure: The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Physics 2024 to John J. Hopfield and Geoffrey Hinton *for foundational discoveries and inventions that enable machine learning with artificial neural networks.*

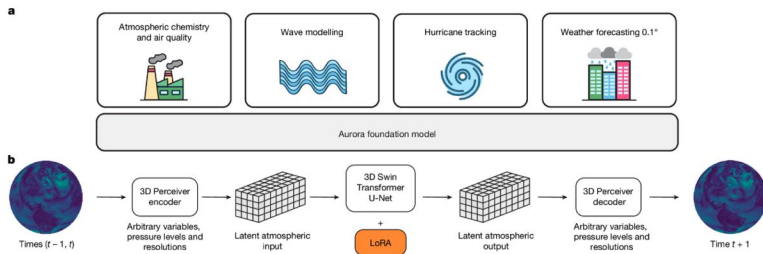
Machine Learning: Why and What VII



a, The selected diagnostic systems used in this work: magnetics, Thomson scattering (TS) and charge-exchange recombination (CER) spectroscopy. The possible tearing instability of $m/n = 2/1$ is shown in orange. **b**, The heating, current drive and control actuators used in this work. **c**, Schematic description of the tearing-avoidance control, including preprocessing, high-level control by a DNN and low-level control by a PCS. **d**, The AI controller based on the DNN.

Figure: Avoiding fusion plasma tearing instability with deep reinforcement learning. Source: Nature.

Machine Learning: Why and What VIII



Icons are for illustrative purposes only. **a**, Aurora is pretrained on several heterogeneous datasets with different resolutions, variables and pressure levels. The model is then fine-tuned for several operational forecasting scenarios at different resolutions: atmospheric chemistry and air quality at 0.4° , wave modelling at 0.25° , hurricane tracking at 0.25° and weather forecasting at 0.1° . **b**, Aurora is a flexible 3D Swin Transformer¹⁹ with 3D Perceiver-based²¹ atmospheric encoders and decoders. The model is able to ingest inputs with different spatial resolutions, numbers of pressure levels and variables.

Figure: A foundation model for the Earth system. Source: Nature.

Outline

Introduction

Course Overview

Machine Learning: Why and What

Taxonomy of Machine Learning

Copyright and Reference

Taxonomy of Machine Learning I

A formal definition:

Definition

A computer program is said to learn from experience E with respect to some class of task T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .

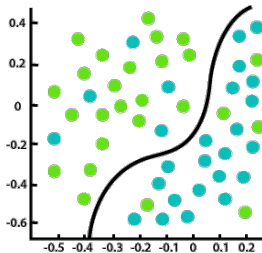
Based on the definition, machine learning algorithms can be classified according to different *tasks* (T), *experience* (E), and *performances* (P),

- ▶ Based on the *learning task* T , we roughly have
 - ▶ *Regression Task*: the computer program is asked to predict a numerical/float value given some input. To solve this task, the learning algorithm is asked to output a function $f: \mathbb{R}^D \rightarrow \mathbb{R}^I$.
 - ▶ *Classification Task*: the computer program is asked to specify which of k categories some input belongs to. To solve this task, the learning algorithm is usually asked to produce a function $f: \mathbb{R}^D \rightarrow \{1, \dots, k\}$. Sometimes, f can produce a probability distribution over classes.

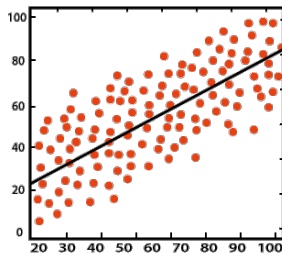
Taxonomy of Machine Learning II

- ▶ Based on the kind of *experience* E the data can have during training, we have
 - ▶ Supervised Learning
 - ▶ Unsupervised Learning
 - ▶ Reinforcement Learning (will not be covered).
- ▶ In order to *evaluate the performance* P of a machine learning algorithm, we must design various quantitative measures of its performance.

Taxonomy of Machine Learning III



Classification



Regression

Figure: Regression vs classification. Source:

<https://www.javatpoint.com/regression-vs-classification-in-machine-learning>

Taxonomy of Machine Learning IV

Question

Determine the tasks T of the following activities in energy system (regression or classification?):

1. Renewable generation or electricity consumption forecast.
2. Detect if there is a congestion in the electricity transmission line.
3. Predict the optimal dispatch plan of generator.
4. Determine if the power system operational point is stable or not.

Question

Determine the experience E of the following activities in energy system (supervised or unsupervised?):

1. Train renewable generation forecaster based on historical data.
2. Compare the common patterns of UK's wind generation in summer and winter.

Outline

Introduction

Course Overview

Machine Learning: Why and What

Taxonomy of Machine Learning

Copyright and Reference

Copyright and Reference I

Some of the materials (including images and mathematical definitions) in this lecture note are taken from the following references:

- ▶ Murphy, K. P. Machine Learning—A probabilistic Perspective. The MIT Press, 2012. *A solid ML textbook with mathematical foundations.*
- ▶ Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning. The MIT Press, 2016¹. *One of the classics.*
- ▶ Bishop, Christopher M., and Nasser M. Nasrabadi. Pattern recognition and machine learning. Vol. 4. No. 4. New York: springer, 2006. *A little old and still classic.*

You are welcome (but not compulsory) to study online. For example,

- ▶ Machine learning specialization by Andrew Ng².

¹<https://www.deeplearningbook.org/>

²https://www.youtube.com/playlist?list=PLkDaE6sCZn6FNC6YRfRQc_FbeQrF8BwGI

Outline

Supervised Learning

Overview

Linear Regression

Logistic Regression

Support Vector Machine

Overview I

Definition

In *supervised learning* approach, the goal is to learn a mapping from inputs x to output y , given a *labeled set of input-output pairs* $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$. Here \mathcal{D} is the *training set*, and N is the number of training examples.

Each x_i is a D -dimensional vector of numbers, representing, say, the weather conditions such as temperature, wind speed, solar radiation, etc. These are called *features* or *attributes* or *contexts*. They are stored in an $N \times D$ *design matrix* X .

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{pmatrix} = \begin{pmatrix} x_1^1 & x_1^2 & \cdots & x_1^D \\ x_2^1 & x_2^2 & \cdots & x_2^D \\ \vdots & \vdots & \ddots & \vdots \\ x_N^1 & x_N^2 & \cdots & x_N^D \end{pmatrix} \in \mathbb{R}^{N \times D}$$

In principal, x_i can contain complex information, such as an image, sentence, etc. In this lecture, we mainly focus on *time series* data.

Overview II

Questions

If you want to build a machine learning model to forecast solar generation, what can be included in X ?

Depending on the format of the label y , we have

- ▶ y_i is real-valued: *regression* or *pattern recognition* task.
- ▶ y_i is categorical: *classification*.

Outline

Supervised Learning

Overview

Linear Regression

Logistic Regression

Support Vector Machine

Linear Regression: Scalar Case I

One of the most widely used models for regression is known as *linear regression*. This asserts that the response is a linear function of the input feature.

$$y(x) = w^T x + \epsilon = \sum_{j=1}^D w_j x_j + \epsilon$$

where D is the number of features and ϵ is the residual error between the linear predictions and the true response. Here, y is a scalar. w is the model parameter that you want to learn.

We often assume that ϵ has a Gaussian distribution. e.g., $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$. Then the linear regression can be denoted as

$$p(y|x, \theta) = \mathcal{N}(y|\mu(x), \sigma^2(x))$$

Linear Regression: Scalar Case II

In this case, it is assumed that y follows a conditional Gaussian distribution with mean given by

$$\mu(x) = w^T x$$

and the variance equals to the *fixed* noise $\sigma^2(x) = \sigma^2$.

For example, suppose the input is one-dimensional,

$$\mu(x) = w_0 + w_1 x = w^T x$$

where w_0 is the *bias* term, w_1 is the slope, and we can defined the vector $x = (1, x)$.

Questions

Can “linear” regression capture “nonlinear” relationships?

Linear Regression: Scalar Case III

The linear regression can capture nonlinear relationship between x and y .

$$p(y|x, \theta) = \mathcal{N}(y|w^T \phi(x), \sigma^2)$$

where $\phi(x)$ is a nonlinear function of x . For example, a *polynomial regression* is defined as

$$\phi(x) = [1, x, x^2, \dots, x^d]$$

Linear Regression: Scalar Case IV

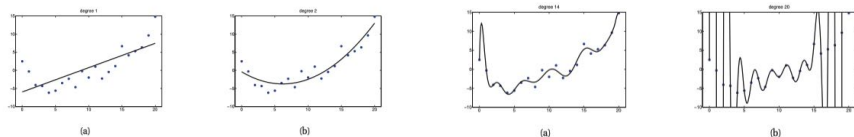


Figure: Linear regression (a) vs nonlinear regression (b).

Figure: Polynomial of degrees 14 (a) and 20 (b).

Questions

Does a perfect model (e.g. (b)) always give you good decision (compared to (a))?

Linear Regression: Scalar Case V

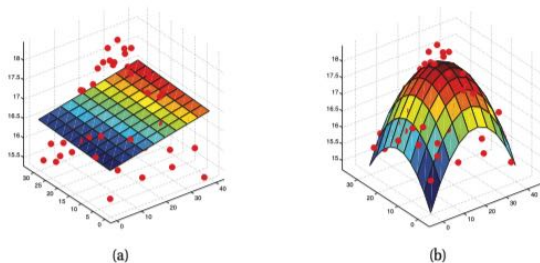


Figure: Linear regression on 2D data. (a). $\mu(x) = w_0 + w_1x_1 + w_2x_2$;
(b). $\mu(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_2^2$

Linear Regression: Maximum Likelihood Estimation I

Linear regression can be modelled and trained by *maximum likelihood estimation* (MLE).

$$\hat{\theta} = \arg \max_{\theta} \log p(\mathcal{D}|\theta)$$

Assuming that the training examples are independent and identically distributed (iid), the log-likelihood is represented as

$$\log p(\mathcal{D}|\theta) = \log \prod_{i=1}^N p(y_i|x_i, \theta) = \sum_{i=1}^N \log p(y_i|x_i, \theta)$$

Recall that a Gaussian distribution is assumed:

$$p(y|x, \theta) = (2\pi\sigma^2)^{-\frac{1}{2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Linear Regression: Maximum Likelihood Estimation II

Plugging into the log likelihood:

$$\ell(\theta) = -\frac{1}{2\sigma^2} \sum_{i=1}^N (y_i - w^T x_i)^2 - \frac{N}{2} \log 2\pi\sigma^2$$

cancel

In the case when σ is fixed, the second term is removed. The so-called *mean squared error* (MSE) loss is derived:

$$\min_w \text{MSE}(w) := \sum_{i=1}^N (y_i - w^T x_i)^2$$

which is also named as *least squares*.

Question

Can you derive the MSE loss by yourself?

Linear Regression: Maximum Likelihood Estimation III

The MSE loss becomes:

$$\text{MSE}(w) = \frac{1}{2} \|y - Xw\|_2^2 = \frac{1}{2} (y - Xw)^T (y - Xw)$$

where $y \in \mathbb{R}^N$, $X \in \mathbb{R}^{N \times D}$, $w \in \mathbb{R}^D$.

Taking the derivative with respect to w results in the first-order optimality condition:

$$X^T y - X^T X w = 0$$

Therefore, the least-square solution is given as:

$$\hat{w}_{ls} = (X^T X)^{-1} X^T y$$

Questions

Under what condition the $X^T X$ is invertible (non-singular)? What if it is not invertible?

Outline

Supervised Learning

Overview

Linear Regression

Logistic Regression

Support Vector Machine

Logistic Regression I

Logistic “regression” is a binary classification, though it is named as regression...

Let's say $y \in \{0, 1\}$. The output of the logistic regression is the probability of one of the class. Therefore, we need

$$\mu(x) = \text{sigm}(w^T x) \in [0, 1]$$

A *sigmoid function* or *logistic function* is defined as

$$\text{sigm}(\eta) = \frac{1}{1 + \exp(-\eta)} = \frac{e^\eta}{e^\eta + 1}$$

For the scalar case, let's assume that the output is the probability of class 1,

$$p(y_i = 1 | x_i, w) = \text{sigm}(w_0 + w_1 x)$$

Logistic Regression II

We say a sample is classified as 1 if it has probability larger than 0.5:

$$\hat{y} = 1 \Leftrightarrow p(y = 1|x) > 0.5$$

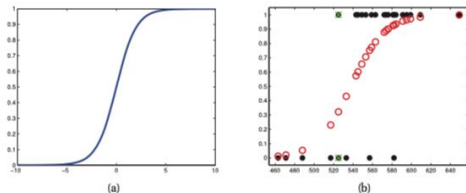


Figure: (a). The sigmoid function; (b). The output of the logistic regression and classification outcomes. The black dots are the input data. The red circles are the predicted probabilities.

Logistic Regression III

The log-likelihood of logistic regression is *cross-entropy* loss:

$$\text{CE}(w) = - \sum_{i=1}^N y_i \log \mu_i + (1 - y_i) \log(1 - \mu_i)$$

where $y_i \in \{0, 1\}$. μ_i is the probability of class 1. Therefore, when $y_i = 1$, only the first-term is active (the probability of class 1); and when $y_i = 0$, only the second-term is active (the probability of class 0).

Question

Does CE have a closed-form solution, such as MSE loss in linear regression case? Why?

Unlike linear regression (MSE loss), cross-entropy loss does not have a closed-form solution. Numerical optimization techniques, such as *gradient*

Logistic Regression IV

descent (first-order method) or *Newton's method* (second-order method) can be used.

The gradient of cross entropy loss is:

$$g = \frac{d}{dw} \text{CE}(w) = \sum_i (\mu_i - y_i) x_i = X^T (\mu - y)$$

where μ should be understood as element-wise sigmoid function on Xw .

Question

Can you derive the gradient?

Then the *gradient descent* is the following iteration:

$$w_{k+1} = w_k - \eta \cdot g_k$$

where η is a *hyperparameter* that needs to be determined before training (we will come back to this point later).

Logistic Regression V

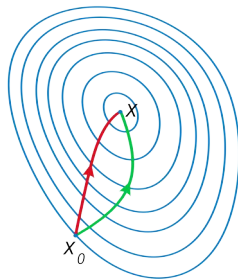


Figure: A comparison of gradient descent (green) and Newton's method (red) for minimizing a function (with small step sizes). Newton's method uses curvature information (i.e. the second derivative) to take a more direct route. Source: Wikipedia.

Logistic Regression VI

Question

What happens when η is too large or too small?

Outline

Supervised Learning

Overview

Linear Regression

Logistic Regression

Support Vector Machine

Support Vector Machine I

Support vector machines (SVMs) are *supervised max-margin* models with associated learning algorithms that analyze data for both *classification* and *regression* analysis.

In addition to linear classification, SVMs can perform *non-linear* classification using the method called kernel trick (we will not cover this in lecture).

Support Vector Machine II

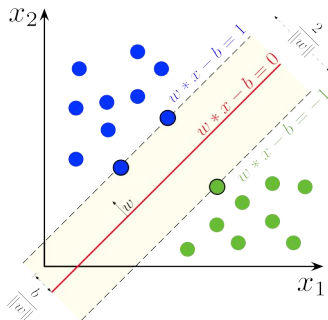


Figure: Max-margin idea in SVM. Source: By Larhman - Own work, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=73710028>

Support Vector Machine III

We want to find a hyperplane (the red line in 2D) that divides the two groups of data (blue and green dots). E.g., the distance between the hyperplane and the nearest point (*the support vector*) from either group is maximized.

Support Vector Machine IV

If the training data is linearly separable (all the data can be eventually perfectly classified!), we can have two parallel hyperplanes that separate the two classes of data with largest distance. The largest distance is the *margin*.

When we have a normalized dataset (e.g., all data falls in range $[0, 1]$, etc), the two parallel hyperplanes can be written as

$$w^T x - b = 1$$

$$w^T x - b = -1$$

The distance between two hyperplanes is $\frac{2}{\|w\|}$. Therefore, maximizing the margin is equivalent to minimizing $\|w\|$. The distance requires some math to derive, feel free to explore online³.

Support Vector Machine V

We also need to put data points outside of the margin area, e.g.,

$$\begin{aligned}w^T x_i - b &\geq 1, & y_i &= 1 \\w^T x_i - b &\leq -1, & y_i &= -1\end{aligned}$$

which can be compactly written as

$$y_i(w^T x_i - b) \geq 1, \quad \text{for any data}$$

Now we have the objective (to maximize the margin) and constraints (to put data outside the margin area), we can formulate a *optimization problem*:

$$\begin{aligned}\min_{w,b} & \quad \frac{1}{2} \|w\|^2 \\ \text{s.t.} & \quad y_i(w^T x_i - b) \geq 1, \quad \text{for any data}\end{aligned}$$

Support Vector Machine VI

Questions

Is the SVM formulation a convex optimization problem, why? What is the good property of convex optimization.

As we mentioned earlier, there are some extensions to the hard-margin SVM:

- ▶ *Soft-margin SVM* allows some data inside the margin area.
- ▶ *Nonlinear or kernel SVM* allows to separate more complex data pattern.

In the coursework, you can test all the different settings (or hyperparameters) using Python packages such as `sklearn`⁴, without knowing the detailed math behind them. (We will have an example in the afternoon)

³<https://www.akshayagrawal.com/lecture-notes/html/hyperplanes.html>

⁴<https://scikit-learn.org/1.5/modules/svm.html>

Outline

Unsupervised Learning

Overview

Principal Component Analysis

k-means Clustering

Unsupervised Learning: Definition I

Questions

Given a set of power system sensor measurement data, let us consider a task to separate samples with irregular voltage profile. What can you do if there is no label given?

Definition

In *unsupervised learning*, there is only given inputs $\mathcal{D} = \{x_i\}_{i=1}^N$ and the goal is to find interesting patterns in the data. This is a much less well-defined problem, since we are not told what kinds of pattern to look for.

Unsupervised Learning: Definition II

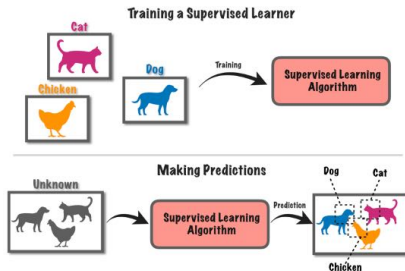


Figure: Supervised vs unsupervised learning.

Some examples include *principal component analysis (PCA)*, *k-means*, etc.

Outline

Unsupervised Learning

Overview

Principal Component Analysis

k-means Clustering

PCA I

Imagine you have a dataset with 10,000 number of features. And you would like to “compress” the dataset into lower dimensions.

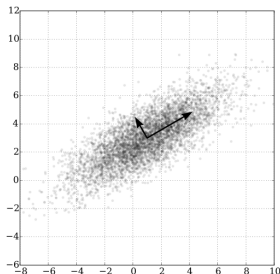
Question

When is a dimension reduction useful?

Question

Consider the 2D data shown below. For the two-marked directions, which direction contains more information, why?

PCA II



PCA⁵ is a *linear dimensionality reduction* method that can project the data into fewer dimensions (the D) in a way that most of the information of the data is kept!

- ▶ Summary: Minimize the dimension while keeping the information.

PCA III

Consider that you have some sensor measurement data with a large number of features, e.g., D is very large,

- ▶ You would like to visualize the data. Then...
- ▶ You would like to first extract useful information, then apply ML algorithms. Then...

Hint: It is always a great practice to visualize your data (in low dimensions) to see if it has some common mode.

Question

Explain why PCA is an unsupervised learning.

Question

Does PCA project onto the existing features?

⁵<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

Outline

Unsupervised Learning

Overview

Principal Component Analysis

k-means Clustering

k-means Clustering

As the name suggests, the k-means⁶ method divides the training set into k different classes in an *unsupervised* way (without label).

The k-means algorithm works as follows

1. Determine the number of clusters k .
2. Initialize k (random) different center of clusters $\{c_1, \dots, c_k\}$.
3. Assign each training example to cluster i , where i is the index of the *nearest* center c_i .
4. Each center c_i is *updated to the mean* of all training examples assigned to cluster i (centroid).
5. Repeat step (3)-(4).

Question

For the task of detecting normal and abnormal sensor measurements from unlabeled data, how many clusters (k) should be assumed in step (1)?

⁶<https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html>

[//scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html](https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html)

Outline

Reinforcement Learning Overview Only

Reinforcement Learning

There is a third type of machine learning, known as *reinforcement learning*, which is somewhat less commonly used. This is useful for learning how to act or behave when given occasional reward or punishment signals. We will not cover this type of ML in this lecture.

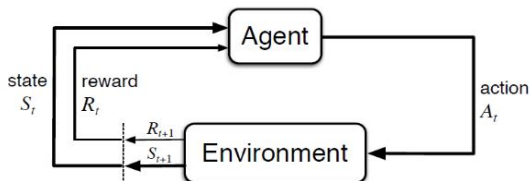


Figure: The reinforcement learning updates the agent by continuously interacting with the environment.

Outline

Performance Measure Overview

Performance Measure I

In order to evaluate the abilities of a machine learning algorithm, we must design a quantitative but straightforward measure of its performance.

For *regression task*, we can use the mean squared error (MSE) as defined before.

For the binary classification task on sensor measurement, we can define more performance measures, including

- ▶ *True positive rate (TPR)*: the proportion of abnormal measurements (class 1) that are correctly marked (as class 1). You can also call it the *detection rate*.
- ▶ *False positive rate (FPR)*: the proportion of normal measurement (class 0) that are incorrectly marked (as class 1). You can also call it *false alarm*.

Performance Measure II

- ▶ *True negative rate (TNR)*: the proportion of normal measurements (class 0) that are correctly marked (as class 0).
- ▶ *False negative rate (FNR)*: the proportion of abnormal measurements (class 1) that are incorrectly marked as (class 0). You can also call it *missing alarm rate*.

How to remember? FPR: falsely classified as positive, etc.

To systematically include all the measures, the F_1 score is defined as

$$F_1 = \frac{2TP}{2TP + FP + FN}$$

Performance Measure III

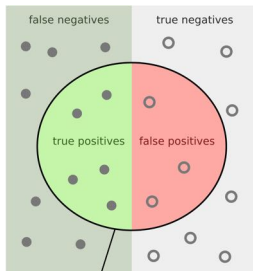


Figure: Performance measure for classifications. The left part represents positive samples (Abnormal measurements) and the right part represents the negative samples (Normal measurements). The middle circle represents the detected positive samples.

Question

Can you identify the TPR, FPR, TNR, and FNR in the above plot?

Outline

Model Capacity

Underfit and Overfit

Regularization

Overfitting and Underfitting I

Having good accuracy on the training dataset is not the ultimate goal of machine learning. The central challenge in machine learning is that we must perform well on *new, unseen* inputs.

The ability to perform well on previously unobserved inputs (you can call it test set) is called *generalization*.

When training a machine learning model, we *only* have access to a training set, we can compute some error measure on the training set called the *training error*, and we reduce this training error as described in the linear regression case through optimization.

$$\text{MSE}^{\text{train}} = \|X^{\text{train}}w - y^{\text{train}}\|_2^2$$

Overfitting and Underfitting II

What separates machine learning from optimization is that we want the generalization error, also called the *test error*, to be low as well.

$$\text{MSE}^{\text{test}} = \|X^{\text{test}}w - y^{\text{test}}\|_2^2$$

Overfitting and Underfitting III

Therefore, the factors determining how well a machine learning algorithm will perform are its ability to:

1. Make the training error small.
2. *Make the gap between training and test error small.*

Definition

Underfit occurs when the model is not able to obtain a sufficiently low error value on the training set. *Overfit* occurs when the gap between the training error and test error is too large.

Overfitting and Underfitting IV

We can control whether a model is more likely to overfit or underfit by altering its *capacity*.

- ▶ Models with insufficient capacity are unable to solve complex tasks.
- ▶ Models with high capacity can solve complex tasks, but when their *capacity is higher than needed* to solve the present task, they may overfit.

One way to control the capacity of a learning algorithm is by choosing its *hypothesis space*, the set of functions that the learning algorithm is allowed to select as being the solution.

As mentioned before, the linear regression algorithm has the set of all linear functions of its input as its hypothesis space. We can generalize linear regression to include polynomials (polynomial regression), rather than just linear functions, in its hypothesis space. Doing so increases the capacity of the model.

Overfitting and Underfitting V

Normally, we can start to choose model with lower capacity (complexity).
For example, a linear regression with polynomial of degree one:

$$\hat{y} = b + wx$$

By introducing x^2 as another feature, we increase the complexity of the regression model:

$$\hat{y} = b + w_1x + w_2x^2$$

We can continue to add more powers of x , for example a polynomial of degree of 9:

$$\hat{y} = b + \sum_{i=1}^9 w_i x^i$$

Overfitting and Underfitting VI

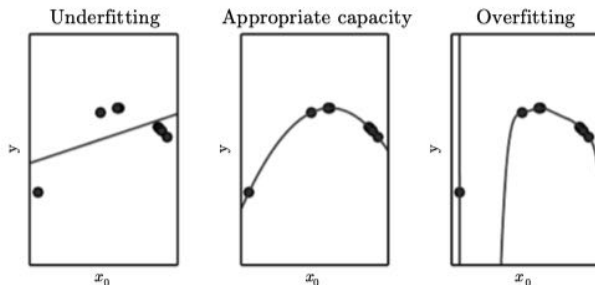


Figure: Underfit vs overfit

Overfitting and Underfitting VII

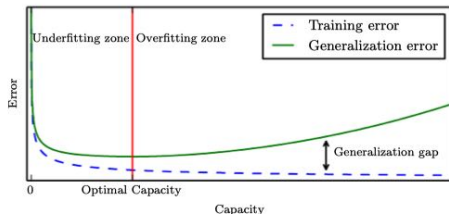


Figure: Typical relationship between capacity and error.

Question

We still have many unsolved problems. For example, we don't have access to the test set... Then how can we know the generalization error (or the performance on test dataset) during training?

Outline

Model Capacity

Underfit and Overfit

Regularization

Regularization I

Degree of complexity: So far, the only method of modifying a learning algorithm is to increase or decrease the model's representational capacity by adding or removing the degree of the features.

Larger hypothesis space: Another way to control the performance (capacity) of the algorithms is to choose what kind of functions that can better fit on the data. For example, linear regression would not perform very well if we try to use it to fit on $\sin(x)$ from x .

Regularization: We can introduce a *preference* in the learning algorithm for certain solutions within its hypothesis space. In other words, the algorithm will choose a less preferred (*more complex*) solution only if it fits the training data *significantly better* than the preferred (simpler) one.

Regularization II

We can include the *weight decay* on the linear regression loss for *regularization* purpose.

$$\text{MSE}^{\text{reg}} = \text{MSE}^{\text{train}} + \lambda w^T w$$

where λ is a value chosen before the final training (a hyperparameter).

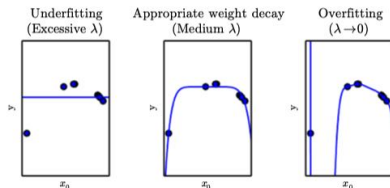


Figure: Different λ on the overfitted polynomial.

Questions

What actually happens if $\lambda \rightarrow 0$ and ∞ ?

Outline

Hyperparameters and Validation Set

Hyperparameters

Validation Set

Hyperparameters I

Definition

Most machine learning algorithms have several settings that we can use to control the behavior of the learning algorithm. These settings are called *hyperparameters*.

NOTE: The values of hyperparameters are *not* adapted by the learning algorithm itself. Why?

The hyperparameters significantly influence the performance of the ML model. For example,

1. In the polynomial regression, $\phi(x) = [1, x, x^2, \dots, x^d]$, the degree of the polynomial, which represents the *capacity* hyperparameter.
2. In the regression with weight decay, $\text{MSE}^{\text{reg}} = \text{MSE}^{\text{train}} + \lambda w^T w$, the λ value used to control the *strength* of weight decay is another example of hyperparameter.
3. Whenever you train an ML model, there always exists a set of hyperparameters which should carefully play with.

Hyperparameters II

SVC

```
class sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale',  
coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200,  
class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr',  
break_ties=False, random_state=None) \[source\]
```

Figure: Sklearn SVM function. There are many hyperparameters to be set. Some of them can be more impactful than the other. You will have example on this.

Question

In the simplest setting, it is not appropriate to learn the hyperparameters on the training set. Based on our discussion on overfitting, explain why.

Outline

Hyperparameters and Validation Set

Hyperparameters

Validation Set

Validation Set I

Recall a previous problem,

Question

We still have many unsolved problems. For example, we don't have access to the test set... Then how can we know the generalization error (or the performance on test dataset) during training?

We would like to *estimate* the generalizability of the ML model in *an unknown dataset*. The solution is to separate a held-out set from the *training dataset*, called *validation set*. Important notice:

1. The validation set should NOT be used to train the model!
2. NO test data should be included in the validation set!

Specifically, we split the training data into two *disjoint* subsets. One of these subsets is used to learn the parameters (e.g., 80%). The other subset is our validation set (e.g., 20%), used to estimate the generalization error during or after training, allowing for the hyperparameters to be updated accordingly.

Validation Set II

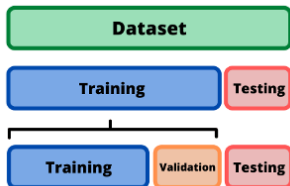


Figure: Train, validation, and test dataset.

Given labeled dataset. The procedure could be:

1. Split the training dataset into 'training dataset' and validation set with a predefined ratio (e.g., 0.8 vs 0.2).
2. Determine an ML model you would like to train on.
3. Set a list of value of all hyperparameters. E.g., $\lambda = [0, 0.01, 0.001]$, $d = 1, 2, \dots$ etc.

Validation Set III

4. Train the ML model under each of the hyperparameters combination, e.g. $(\lambda = 0, d = 1)$, $(\lambda = 0.01, d = 2)$, etc on the *training dataset*. This processing is also called *grid search*.
5. Evaluate the performance of all trained models with different hyperparameter combinations on *validation set*. The one with the best performance is the final model.

Basically, you never touch (you do not have actually) on the test set during model training.

Validation Set IV

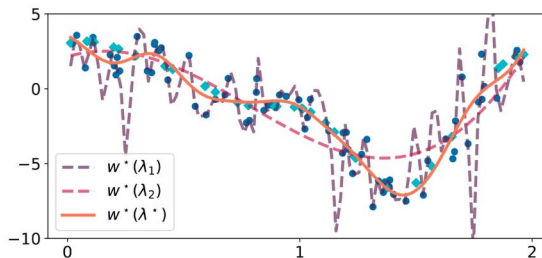


Figure: Tuning hyperparameter λ based on the performance on the **validation set**. This example is taken from [Blondel and Roulet, 2024]. Blue dots are training samples and cyan dots are validation samples.

Question

Can you tell which fitting curve is over-fitted and under-fitted?

Outline

Special Topic One: Time Series Foundation Model

No Free Lunch Theorem

What is Foundation Model?

Time Series Foundation Model

Amazon Chronos

No Free Lunch Theorem I

- ▶ “All models are wrong, but some models are useful.”
- ▶ There is no single best model that works optimally for all kinds of problems. The reason is that a set of assumptions that works well in one domain may *generalize* poorly in another.
- ▶ As a consequence of the *no free lunch theorem*, we need to develop many different types of models, to cover the wide variety of data that occurs in the real world. And for each model, there may be many different algorithms we can use to train the model.
- ▶ But with the development of foundation model, the No Free Lunch Theorem “seems” to be less accurate.

Outline

Special Topic One: Time Series Foundation Model

No Free Lunch Theorem

What is Foundation Model?

Time Series Foundation Model

Amazon Chronos

Foundation Model I

All the techniques we discussed earlier require training by yourself. This is because one model is typically only good at one task.

Definition

A *foundation model* is a *large-scale, pre-trained* model (usually based on deep learning) that learns general-purpose representations from *massive, diverse datasets* and can be *adapted* (fine-tuned or prompted) to perform many downstream tasks.

- ▶ Trained on broad and heterogeneous data (e.g., text, image, code, multimodal data, etc);
- ▶ Use self-supervised or unsupervised learning (e.g. predicting masked tokens, next word, next frame, etc.): predict on itself.
- ▶ Transferable to many domains with prompting (zero-shot) or fine-tuning (few-shot).

Foundation Model II

- ▶ **LARGE:** Billions of parameters.

Questions

Can GPT-5 be seen as a foundation model?

Suppose you have a model like GPT-5 trained on global text data. You can immediately adapt it for:

- ▶ Writing technical reports (text generation).
- ▶ Extracting entities from energy market documents (information extraction).
- ▶ Explaining optimization models (reasoning).

You don't train it from scratch: you just prompt or fine-tune it → this reusability makes it a foundation model.

Outline

Special Topic One: Time Series Foundation Model

No Free Lunch Theorem

What is Foundation Model?

Time Series Foundation Model

Amazon Chronos

Time Series Foundation Model (TSFM) I

Definition

A time series foundation model is a large pre-trained model designed to understand and predict *temporal* patterns across diverse time-dependent datasets. It is trained on massive collections of time series from many domains (e.g., finance, energy, weather, health) and can generalize to new, unseen time series without retraining from scratch.

Questions

Does TSFM follow the properties of general FM?

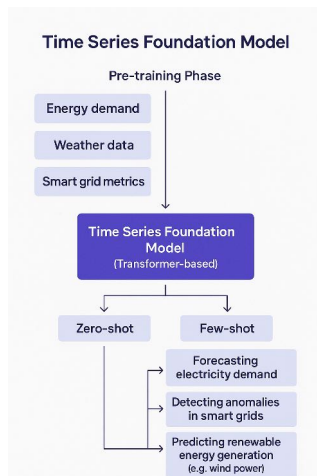


Figure: The concept of TSFM (image generated by GPT-5).

Outline

Special Topic One: Time Series Foundation Model

No Free Lunch Theorem

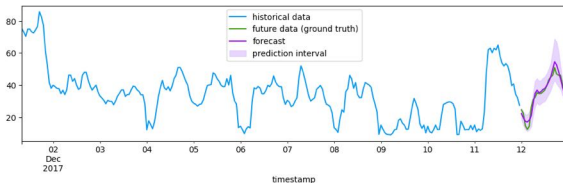
What is Foundation Model?

Time Series Foundation Model

Amazon Chronos

Amazon Chronos I

- ▶ Chronos is a family of pretrained time series forecasting models open sourced by Amazon. The latest version supports zero-shot univariate, multivariate, and covariate-informed forecasting tasks (Can you guess what is covariate?). The largest one has 710M parameters.
- ▶ GitHub:
<https://github.com/amazon-science/chronos-forecasting>.
- ▶ A zero-shot example for energy price forecasting.



Outline

Special Topic Two: DATA? DATA! Where and How to obtain the Data

Why Data is Important? I

- ▶ All the previous discussions assume that you already have *sufficient* dataset with *good quality*.
- ▶ For energy systems, some of the most important data include power consumption and generation, energy price (day-ahead and real-time) in an open market, and operational status such as transmission line congestion level.
- ▶ Believe it or not, high quality data is not easy to get. And there is a famous one: “Garbage in, garbage out” (GIGO).
- ▶ But luckily, the EU (including UK) has enforced many regulations for opening data.

European's Effort on Data Transparency I

- ▶ Open energy data is essential for ensuring transparency, market efficiency, and innovation in the energy sector.
- ▶ It allows regulators, researchers, and consumers to better understand how electricity is produced, transmitted, and consumed, fostering trust and enabling smarter system planning and policy design.
- ▶ The European Union has established several key regulations to enforce this transparency, most notably Regulation (EU) No 543/2013, which mandates that electricity market participants, such as transmission system operators (TSOs), power plant operators, and large consumers, publish detailed operational data on generation, demand, outages, and balancing.

European's Effort on Data Transparency II

- ▶ Complementary frameworks like REMIT (Regulation 1227/2011) ensure integrity in wholesale energy markets, while newer laws such as the Energy Efficiency Directive (2023/1791) and the EU Data Act (2023/2854) extend transparency and data-sharing obligations to other parts of the energy ecosystem.
- ▶ Most of this data is publicly available and freely accessible through the *ENTSO-E Transparency Platform*, which serves as the central hub for Europe-wide electricity system operational data.

Where to Find the Power System Data I

- ▶ Electricity generation, transportation and consumption for the European market: <https://newtransparency.entsoe.eu/> is the main source for obtain market, load, generation, transmission, and balancing data.
- ▶ Weather dataset is important to forecast nearly all factors in electricity market. You may find it useful in ERA5 dataset <https://cds.climate.copernicus.eu/datasets>.
- ▶ More detailed operation and market data may be found on regional operators website, such as FinGrid and Elia.

Reference I



Blondel, M. and Roulet, V. (2024).

The elements of differentiable programming.

arXiv preprint arXiv:2403.14606.



Hamann, H. F., Brunschwiler, T., Gjorgiev, B., Martins, L. S., Puech, A., Varbella, A., Weiss, J., Bernabe-Moreno, J., Massé, A. B., Choi, S., et al. (2024).

A perspective on foundation models for the electric power grid.

arXiv preprint arXiv:2407.09434.