The A”allon koneoppimisen sanakirja

Alexander Jung[1], Konstantina Olioumtsevits[1], Ekkehard Schnoor[1], Tommi Flores Ryynänen[1], Juliette Gronier[2] ja Salvatore Rastelli[1]
Käännös Mikko Seesto[1], Janita Thusberg[1] ja Helli Manninen[1]
[1]Aalto-yliopisto [2]ENS Lyon

23. lokakuuta 2025

please cite as: A. Jung, K. Olioumtsevits, E. Schnoor, T. Ryynänen, J. Gronier ja S. Rastelli, *Aallon koneoppimisen sanakirja*. Käännös J. Thusberg, H. Manninen ja M. Seesto. Espoo, Suomi: Aalto-yliopisto, 2025.

Acknowledgment

This dictionary of machine learning evolved through the development and teaching of several courses, including CS-E3210 Machine Learning: Basic Principles, CS-C3240 Machine Learning, CS-E4800 Artificial Intelligence, CS-EJ3211 Machine Learning with Python, CS-EJ3311 Deep Learning with Python, CS-E4740 Federated Learning, and CS-E407507 Human-Centered Machine Learning. These courses were offered at Aalto University `https://www.aalto.fi/en`, to adult learners via The Finnish Institute of Technology (FITech) `https://fitech.io/en/`, and to international students through the European University Alliance Unite! `https://www.aalto.fi/en/unite`.

We are grateful to the students who provided valuable feedback that helped shape this dictionary. Special thanks to Mikko Seesto for his meticulous proofreading.

This work was supported by

- the Research Council of Finland (grant 331197, 363624, 349966);
- the European Union (grant 952410);
- the Jane and Aatos Erkko Foundation (grant A835);
- Business Finland, as part of the project Forward-Looking AI Governance in Banking and Insurance (FLAIG).

Sisällys

Lists of Symbols

Sets and Functions

$a \in \mathcal{A}$	The object a is an element of the set $\mathcal{A}$.
$a := b$	We use a as a shorthand for b.
$\lvert\mathcal{A}\rvert$	The cardinality (i.e., number of elements) of a finite set $\mathcal{A}$.
$\mathcal{A} \subseteq \mathcal{B}$	$\mathcal{A}$ is a subset of $\mathcal{B}$.
$\mathcal{A} \subset \mathcal{B}$	$\mathcal{A}$ is a strict subset of $\mathcal{B}$.
$\mathcal{A} \times \mathcal{B}$	The Cartesian product of the sets $\mathcal{A}$ and $\mathcal{B}$.
$\mathbb{N}$	The natural numbers 1, 2, ...
$\mathbb{R}$	The real numbers x [1].
$\mathbb{R}_+$	The nonnegative real numbers $x \geq 0$.
$\mathbb{R}_{++}$	The positive real numbers $x > 0$.
$\{0, 1\}$	The set consisting of the two real numbers 0 and 1.
$[0, 1]$	The closed interval of real numbers x with $0 \leq x \leq 1$.

$\arg\min_{\mathbf{w}} f(\mathbf{w})$	The set of minimizers for a real-valued function $f(\mathbf{w})$. See also: function.
$\mathbb{S}^{(n)}$	The set of unit-norm vectors in $\mathbb{R}^{n+1}$. See also: norm, vector.
$\exp(a)$	The exponential function evaluated at the real number $a \in \mathbb{R}$. See also: function.
$\log a$	The logarithm of the positive number $a \in \mathbb{R}_{++}$.
$f(\cdot): \mathcal{A} \to \mathcal{B} : a \mapsto f(a)$	A function (or map) from a set $\mathcal{A}$ to a set $\mathcal{B}$, which assigns to each input $a \in \mathcal{A}$ a well-defined output $f(a) \in \mathcal{B}$. The set $\mathcal{A}$ is the domain of the function f and the set $\mathcal{B}$ is the co-domain of f. Koneoppiminen aims to learn a function h that maps piirret $\mathbf{x}$ of a data point to a ennuste $h(\mathbf{x})$ for its nimiö y. See also: function, map, koneoppiminen, piirre, data point, ennuste, nimiö.
$\text{epi}(f)$	The epigraph of a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$. See also: epigraph, function.
$\frac{\partial f(w_1, \ldots, w_d)}{\partial w_j}$	The partial derivative (if it exists) of a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ with respect to w_j [2, Ch. 9]. See also: function.

$\nabla f(\mathbf{w})$

The gradientti of a differentiable real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ is the vector $\nabla f(\mathbf{w}) = \big(\partial f/\partial w_1, \ldots, \partial f/\partial w_d\big)^T \in \mathbb{R}^d$ [2, Ch. 9].

See also: gradientti, differentiable, function, vector.

Matrices and Vectors

$\mathbf{x} = \left(x_1, \ldots, x_d\right)^T$	A vector of length d, with its jth entry being x_j. See also: vector.
$\mathbb{R}^d$	The set of vectors $\mathbf{x} = \left(x_1, \ldots, x_d\right)^T$ consisting of d real-valued entries $x_1, \ldots, x_d \in \mathbb{R}$. See also: vector.
$\mathbf{I}_{l\times d}$	A generalized identity matriisi with l rows and d columns. The entries of $\mathbf{I}_{l\times d} \in \mathbb{R}^{l\times d}$ are equal to 1 along the main diagonal and otherwise equal to 0. See also: matriisi.
$\mathbf{I}_d, \mathbf{I}$	A square identity matriisi of size $d \times d$. If the size is clear from context, we drop the subscript. See also: matriisi.
$\left\|\mathbf{x}\right\|_2$	The Euclidean (or ℓ_2) norm of the vector $\mathbf{x} = \left(x_1, \ldots, x_d\right)^T \in \mathbb{R}^d$ defined as $\left\|\mathbf{x}\right\|_2 := \sqrt{\sum_{j=1}^d x_j^2}$. See also: norm, vector.
$\left\|\mathbf{x}\right\|$	Some norm of the vector $\mathbf{x} \in \mathbb{R}^d$ [3]. Unless otherwise specified, we mean the Euclidean norm $\left\|\mathbf{x}\right\|_2$. See also: norm, vector.
$\mathbf{x}^T$	The transpose of a matriisi that has the vector $\mathbf{x} \in \mathbb{R}^d$ as its single column. See also: matriisi, vector.

$\mathbf{X}^{\,T}$	The transpose of a matriisi $\mathbf{X} \in \mathbb{R}^{m \times d}$. A square real-valued matriisi $\mathbf{X} \in \mathbb{R}^{m \times m}$ is called symmetric if $\mathbf{X} = \mathbf{X}^{\,T}$. See also: matriisi.
$\mathbf{X}^{-1}$	The käänteismatriisi of a matriisi $\mathbf{X} \in \mathbb{R}^{d \times d}$. See also: käänteismatriisi, matriisi.
$\mathbf{0} = \big(0, \ldots, 0\big)^{\,T}$	The vector in $\mathbb{R}^d$ with each entry equal to zero. See also: vector.
$\mathbf{1} = \big(1, \ldots, 1\big)^{\,T}$	The vector in $\mathbb{R}^d$ with each entry equal to one. See also: vector.
$\big(\mathbf{v}^{\,T}, \mathbf{w}^{\,T}\big)^{\,T}$	The vector of length $d + d'$ obtained by concatenating the entries of vector $\mathbf{v} \in \mathbb{R}^d$ with the entries of $\mathbf{w} \in \mathbb{R}^{d'}$. See also: vector.
$\text{span}\{\mathbf{B}\}$	The span of a matriisi $\mathbf{B} \in \mathbb{R}^{a \times b}$, which is the subspace of all linear combinations of the columns of $\mathbf{B}$, such that $\text{span}\{\mathbf{B}\} = \big\{\mathbf{B}\mathbf{a} : \mathbf{a} \in \mathbb{R}^b\big\} \subseteq \mathbb{R}^a$. See also: matriisi.
$\text{null}(\mathbf{A})$	The nullspace of a matriisi $\mathbf{A} \in \mathbb{R}^{a \times b}$, which is the subspace of vectors $\mathbf{a} \in \mathbb{R}^b$ such that $\mathbf{A}\mathbf{a} = \mathbf{0}$. See also: nullspace, matriisi, vector.

$\det(\mathbf{C})$	The determinantti of the matriisi $\mathbf{C}$. See also: determinantti, matriisi.
$\mathbf{A} \otimes \mathbf{B}$	The Kronecker product of $\mathbf{A}$ and $\mathbf{B}$ [4]. See also: Kronecker product.

Probability Theory

$\mathbf{x} \sim p(\mathbf{z})$	The satunnaismuuttuja $\mathbf{x}$ is distributed according to the todennäköisyysjakauma $p(\mathbf{z})$ [5], [6]. See also: satunnaismuuttuja, todennäköisyysjakauma.
$\mathbb{E}_p\{f(\mathbf{z})\}$	The expectation of an satunnaismuuttuja $f(\mathbf{z})$ that is obtained by applying a deterministic function f to an satunnaismuuttuja $\mathbf{z}$ whose todennäköisyysjakauma is $\mathbb{P}(\mathbf{z})$. If the todennäköisyysjakauma is clear from context, we just write $\mathbb{E}\{f(\mathbf{z})\}$. See also: expectation, satunnaismuuttuja, function, todennäköisyysjakauma.
$\mathrm{cov}\,(x, y)$	The covariance between two real-valued satunnaismuuttujat defined over a common probability space. See also: covariance, satunnaismuuttuja, todennäköisyysjakauma.
$\mathbb{P}(\mathbf{x}, y)$	A (joint) todennäköisyysjakauma of an satunnaismuuttuja whose realizationt are data points with piirre $\mathbf{x}$ and nimiö y. See also: todennäköisyysjakauma, satunnaismuuttuja, realization, data point, piirre, nimiö.
$\mathbb{P}(\mathbf{x}\|y)$	A conditional todennäköisyysjakauma of an satunnaismuuttuja $\mathbf{x}$ given the value of another satunnaismuuttuja y [7, Sec. 3.5]. See also: todennäköisyysjakauma, satunnaismuuttuja.
$\mathbb{P}(\mathcal{A})$	The probability of the measurable event $\mathcal{A}$. See also: probability, measurable, event.

$\mathbb{P}(\mathbf{x};\mathbf{w})$	A parameterized todennäköisyysjakauma of an satunnaismuuttuja $\mathbf{x}$. The todennäköisyysjakauma depends on a parameter vector $\mathbf{w}$. For example, $\mathbb{P}(\mathbf{x};\mathbf{w})$ could be a multivariate normal distribution with the parameter vector $\mathbf{w}$ given by the entries of the mean vector $\mathbb{E}\{\mathbf{x}\}$ and the kovarianssimatriisi $\mathbb{E}\Big\{\big(\mathbf{x}-\mathbb{E}\{\mathbf{x}\}\big)\big(\mathbf{x}-\mathbb{E}\{\mathbf{x}\}\big)^{T}\Big\}$. See also: todennäköisyysjakauma, parameter, todennäköisyysmalli.
$\mathcal{N}(\mu,\sigma^2)$	The todennäköisyysjakauma of a Gaussian random variable (Gaussian RV) $x \in \mathbb{R}$ with mean (or expectation) $\mu = \mathbb{E}\{x\}$ and variance $\sigma^2 = \mathbb{E}\big\{(x-\mu)^2\big\}$. See also: todennäköisyysjakauma, Gaussian RV.
$\mathcal{N}(\boldsymbol{\mu},\mathbf{C})$	The multivariate normal distribution of a vector-valued Gaussian RV $\mathbf{x} \in \mathbb{R}^d$ with mean (or expectation) $\boldsymbol{\mu} = \mathbb{E}\{\mathbf{x}\}$ and kovarianssimatriisi $\mathbf{C} = \mathbb{E}\big\{\big(\mathbf{x}-\boldsymbol{\mu}\big)\big(\mathbf{x}-\boldsymbol{\mu}\big)^{T}\big\}$. See also: multivariate normal distribution, Gaussian RV.
Ω	A sample space of all possible outcomes of a satunnaiskoe. See also: event.
$\mathcal{F}$	A collection of measurable subsets of a sample space Ω. See also: sample space, event.
$\mathcal{P}$	A probability space that consists of a sample space Ω, a σ-algebra $\mathcal{F}$ of measurable subsets of Ω, and a todennäköisyysjakauma $\mathbb{P}(\cdot)$. See also: sample space, measurable, todennäköisyysjakauma.

Machine Learning

r	An index $r = 1, 2, \ldots$ that enumerates data points. See also: data point.
m	The number of data points in (i.e., the size of) a tietoaineisto. See also: data point, tietoaineisto.
$\mathcal{D}$	A tietoaineisto $\mathcal{D} = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ is a list of individual data points $\mathbf{z}^{(r)}$, for $r = 1, \ldots, m$. See also: tietoaineisto, data point.
d	The number of piirre that characterize a data point. See also: piirre, data point.
x_j	The jth piirre of a data point. The first piirre is denoted by x_1, the second piirre x_2, and so on. See also: data point, piirre.
$\mathbf{x}$	The piirrevektori $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ of a data point. The vector's entries are the individual piirre of a data point. See also: piirevektori, data point, vector, piirre.
$\mathcal{X}$	The feature space $\mathcal{X}$ is the set of all possible values that the piirre $\mathbf{x}$ of a data point can take on. See also: feature space, piirre, data point.

$\mathbf{z}$	Instead of the symbol $\mathbf{x}$, we sometimes use $\mathbf{z}$ as another symbol to denote a vector whose entries are the individual piirre of a data point. We need two different symbols to distinguish between raw and learned piirre [8, Ch. 9]. See also: vector, piirre, data point.
$\mathbf{x}^{(r)}$	The piirevektori of the rth data point within a tietoaineisto. See also: piirevektori, data point, tietoaineisto.
$x_j^{(r)}$	The jth piirre of the rth data point within a tietoaineisto. See also: piirre, data point, tietoaineisto.
$\mathcal{B}$	A mini-batch (or subset) of randomly chosen data points. See also: batch, data point.
B	The size of (i.e., the number of data points in) a mini-batch. See also: data point, batch.
y	The nimiö (or quantity of interest) of a data point. See also: nimiö, data point.
$y^{(r)}$	The nimiö of the rth data point. See also: nimiö, data point.
$\left(\mathbf{x}^{(r)}, y^{(r)}\right)$	The piirre and nimiö of the rth data point. See also: piirre, nimiö, data point.

$\mathcal{Y}$	The label space $\mathcal{Y}$ of an koneoppiminen method consists of all potential nimiö values that a data point can carry. The nominal label space might be larger than the set of different nimiö values arising in a given tietoaineisto (e.g., a training set). Koneoppiminen problems (or methods) using a numeric label space, such as $\mathcal{Y} = \mathbb{R}$ or $\mathcal{Y} = \mathbb{R}^3$, are referred to as regressio problems (or methods). Koneoppiminen problems (or methods) that use a discrete label space, such as $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{cat, dog, mouse\}$, are referred to as luokittelu problems (or methods). See also: label space, koneoppiminen, nimiö, data point, tietoaineisto, training set, regressio, luokittelu.
η	Oppimisnopeus (or step size) used by gradient-based methods. See also: oppimisnopeus, step size, gradient-based methods.
$h(\cdot)$	A hypothesis map that maps the piirre of a data point to a ennuste $\hat{y} = h(\mathbf{x})$ for its nimiö y. See also: hypothesis, map, piirre, data point, ennuste, nimiö.
$\mathcal{Y}^{\mathcal{X}}$	Given two sets $\mathcal{X}$ and $\mathcal{Y}$, we denote by $\mathcal{Y}^{\mathcal{X}}$ the set of all possible hypothesis maps $h : \mathcal{X} \to \mathcal{Y}$. See also: hypothesis, map.
$\mathcal{H}$	A hypothesis space or malli used by an koneoppiminen method. The hypothesis space consists of different hypothesis maps $h : \mathcal{X} \to \mathcal{Y}$, between which the koneoppiminen method must choose. See also: hypothesis space, malli, koneoppiminen, hypothesis, map.

$d_{\text{eff}}(\mathcal{H})$	The effective dimension of a hypothesis space $\mathcal{H}$. See also: effective dimension, hypothesis space.
B^2	The squared harha of a learned hypothesis $\hat{h}$, or its parameters. Note that $\hat{h}$ becomes an satunnaismuuttuja if it is learned from data points being satunnaismuuttujat themselves. See also: harha, hypothesis, parameter, satunnaismuuttuja, data point.
V	The variance of a learned hypothesis $\hat{h}$, or its parameters. Note that $\hat{h}$ becomes an satunnaismuuttuja if it is learned from data points being satunnaismuuttujat themselves. See also: variance, hypothesis, parameter, satunnaismuuttuja, data point.
$L((\mathbf{x}, y), h)$	The häviö incurred by predicting the nimiö y of a data point using the ennuste $\hat{y} = h(\mathbf{x})$. The ennuste $\hat{y}$ is obtained by evaluating the hypothesis $h \in \mathcal{H}$ for the piirevektori $\mathbf{x}$ of the data point. See also: häviö, nimiö, data point, ennuste, hypothesis, piirevektori.
E_v	The validointivirhe of a hypothesis h, which is its average häviö incurred over a validation set. See also: validointivirhe, hypothesis, häviö, validation set.
$\widehat{L}\big(h\|\mathcal{D}\big)$	The empirical risk, or average häviö, incurred by the hypothesis h on a tietoaineisto $\mathcal{D}$. See also: empirical risk, häviö, hypothesis, tietoaineisto.

E_t	The opetusvirhe of a hypothesis h, which is its average häviö incurred over a training set. See also: opetusvirhe, hypothesis, häviö, training set.
t	A discrete-time index $t = 0, 1, \ldots$ used to enumerate sequential events (or time instants). See also: event.
t	An index that enumerates learning tasks within a multitask learning problem. See also: learning task, multitask learning.
α	A regularisointi parameter that controls the amount of regularisointi. See also: regularisointi, parameter.
$\lambda_j(\mathbf{Q})$	The jth eigenvalue (sorted in either ascending or descending order) of a positive semi-definite (psd) matriisi $\mathbf{Q}$. We also use the shorthand λ_j if the corresponding matriisi is clear from context. See also: eigenvalue, psd, matriisi.
$\sigma(\cdot)$	The activation function used by an artificial neuron within an artificial neural network (ANN). See also: activation function, ANN.
$\mathcal{R}_{\hat{y}}$	A päätösalue within a feature space. See also: päätösalue, feature space.

$\mathbf{w}$	A parameter vector $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$ of a malli, e.g., the weights of a linear model or an ANN. See also: parameter, vector, malli, weights, linear model, ANN.
$h^{(\mathbf{w})}(\cdot)$	A hypothesis map that involves tunable model parameters $w_1, \ldots, w_d$ stacked into the vector $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$. See also: hypothesis, map, model parameters, vector.
$\phi(\cdot)$	A feature map $\phi : \mathcal{X} \rightarrow \mathcal{X}' : \mathbf{x} \mapsto \phi\big(\mathbf{x}\big)$ that transforms the piirevektori $\mathbf{x}$ of a data point into a new piirevektori $\mathbf{x}' = \phi\big(\mathbf{x}\big) \in \mathcal{X}'$. See also: feature map.
$K\big(\cdot, \cdot\big)$	Given some feature space $\mathcal{X}$, a ydinfunktio is a map $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{C}$ that is psd. See also: feature space, ydinfunktio, map, psd.
$\mathrm{VCdim}\,(\mathcal{H})$	The Vapnik–Chervonenkis dimension (VC dimension) of the hypothesis space $\mathcal{H}$. See also: VC dimension, hypothesis space.

Federated Learning

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	An undirected graph whose nodes $i \in \mathcal{V}$ represent devicet within a federated learning network (FL network). The undirected weighted edges $\mathcal{E}$ represent connectivity between devices and statistical similarities between their tietoaineistot and learning tasks. See also: graph, device, FL network, tietoaineisto, learning task.
$i \in \mathcal{V}$	A node that represents some device within an FL network. The device can access a local dataset and train a local model. See also: device, FL network, local dataset, local model.
$\mathcal{G}^{(\mathcal{C})}$	The induced subgraph of $\mathcal{G}$ using the nodes in $\mathcal{C} \subseteq \mathcal{V}$.
$\mathbf{L}^{(\mathcal{G})}$	The Laplacian matrix of a graph $\mathcal{G}$. See also: Laplacian matrix, graph.
$\mathbf{L}^{(\mathcal{C})}$	The Laplacian matrix of the induced graph $\mathcal{G}^{(\mathcal{C})}$. See also: Laplacian matrix, graph.
$\mathcal{N}^{(i)}$	The neighborhood of the node i in a graph $\mathcal{G}$. See also: neighborhood, graph.
$d^{(i)}$	The weighted node degree $d^{(i)} := \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'}$ of node i. See also: node degree.
$d^{(\mathcal{G})}_{\max}$	The maximum weighted node degree of a graph $\mathcal{G}$. See also: maximum, node degree, graph.

$\mathcal{D}^{(i)}$	The local dataset $\mathcal{D}^{(i)}$ carried by node $i \in \mathcal{V}$ of an FL network. See also: local dataset, FL network.
m_i	The number of data points (i.e., sample size) contained in the local dataset $\mathcal{D}^{(i)}$ at node $i \in \mathcal{V}$. See also: data point, sample size, local dataset.
$\mathbf{x}^{(i,r)}$	The piirre of the rth data point in the local dataset $\mathcal{D}^{(i)}$. See also: piirre, data point, local dataset.
$y^{(i,r)}$	The nimiö of the rth data point in the local dataset $\mathcal{D}^{(i)}$. See also: nimiö, data point, local dataset.
$\mathbf{w}^{(i)}$	The local model parameters of device i within an FL network. See also: model parameters, device, FL network.
$L_i\left(\mathbf{w}\right)$	The local häviöfunktio used by device i to measure the usefulness of some choice $\mathbf{w}$ for the local model parameters. See also: häviöfunktio, device, model parameters.
$L^{(\mathrm{d})}\left(\mathbf{x}, h\big(\mathbf{x}\big), h'\big(\mathbf{x}\big)\right)$	The häviö incurred by a hypothesis h' on a data point with piirre $\mathbf{x}$ and nimiö $h\big(\mathbf{x}\big)$ that is obtained from another hypothesis. See also: häviö, hypothesis, data point, piirre, nimiö.

$\text{stack}\left\{\mathbf{w}^{(i)}\right\}_{i=1}^{n}$

The vector $\left(\left(\mathbf{w}^{(1)}\right)^T, \ldots, \left(\mathbf{w}^{(n)}\right)^T\right)^T \in \mathbb{R}^{dn}$ that is obtained by vertically stacking the local model parameters $\mathbf{w}^{(i)} \in \mathbb{R}^d$, for $i = 1, \ldots, n$.

See also: vector, model parameters.

Tools

characteristic function The characteristic function of a real-valued satunnaismuuttuja x is the function [6, Sec. 26]

$$\phi_x(t) := \mathbb{E}\exp(jtx) \text{ with } j = \sqrt{-1}.$$

The characteristic function uniquely determines the todennäköisyysjakauma of x.

See also: satunnaismuuttuja, todennäköisyysjakauma.

continuous A function $f : \mathbb{R}^d \to \mathbb{R}$ is continuous at a point $\mathbf{x}' \in \mathbb{R}^d$ if for every $\epsilon > 0$ there is a $\delta > 0$ such that for all $\mathbf{x} \in \mathbb{R}^d$ with $\|\mathbf{x} - \mathbf{x}'\|_2 < \delta$, it holds that $|f(\mathbf{x}) - f(\mathbf{x}')| < \epsilon$ [2]. In other words, we can make $f(\mathbf{x})$ arbitrarily close to $f(\mathbf{x}')$ by choosing $\mathbf{x}$ sufficiently close to $\mathbf{x}'$. If f is continuous at every point $\mathbf{x}' \in \mathbb{R}^d$, then f is said to be continuous on $\mathbb{R}^d$. The notion of a continuous function can be naturally extended to functions between general metric spaces [2].

See also: Euclidean space, metric.

convergence TBD.

convex optimization TBD.

determinantti The determinant $\det(\mathbf{A})$ of a square matriisi $\mathbf{A} = \left(\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(d)}\right) \in \mathbb{R}^{d \times d}$ is a function of its columns $\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(d)} \in \mathbb{R}^d$, i.e., it satisfies the following properties [**?**]:

- Normalized:

$$\det(\mathbf{I}) = 1$$

- Multilinear:

$$\det\left(\mathbf{a}^{(1)}, \ldots, \alpha\mathbf{u} + \beta\mathbf{v}, \ldots, \mathbf{a}^{(d)}\right) = \alpha \det\left(\mathbf{a}^{(1)}, \ldots, \mathbf{u}, \ldots, \mathbf{a}^{(d)}\right) + \beta \det\left(\mathbf{a}^{(1)}, \ldots, \mathbf{v}, \ldots, \mathbf{a}^{(d)}\right)$$

- Antisymmetric:

$$\det\left(\ldots, \mathbf{a}^{(j)}, \ldots, \mathbf{a}^{(j')}, \ldots\right) = -\det\left(\ldots, \mathbf{a}^{(j')}, \ldots, \mathbf{a}^{(j)}, \ldots\right).$$

We can interpret a matriisi $\mathbf{A}$ as a linear transformation on $\mathbb{R}^d$. The determinant $\det(\mathbf{A})$ characterizes how volumes in $\mathbb{R}^d$ (and their orientation) are altered by this transformation (see Fig. 1) [3], [**?**]. In particular, $\det(\mathbf{A}) > 0$ preserves orientation, $\det(\mathbf{A}) < 0$ reverses orientation, and $\det(\mathbf{A}) = 0$ collapses volume entirely, indicating that $\mathbf{A}$ is non-invertible. The determinant also satisfies $\det(\mathbf{AB}) = \det(\mathbf{A}) \cdot \det(\mathbf{B})$, and if $\mathbf{A}$ is diagonalizable with eigenvalues $\lambda_1, \ldots, \lambda_d$, then $\det(\mathbf{A}) = \prod_{j=1}^{d} \lambda_j$ [**?**]. For the special cases $d = 2$ (i.e., two-dimensional or 2-D) and $d = 3$ (i.e., three-dimensional or 3-D), the determinant can be interpreted as an oriented area or volume spanned by the column vectors of $\mathbf{A}$.

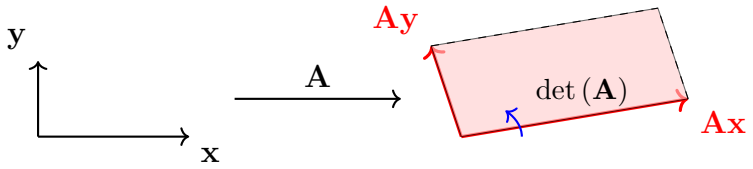


Fig. 1. We can interpret a square matriisi $\mathbf{A}$ as a linear transformation of $\mathbb{R}^d$ into itself. The determinant $\det(\mathbf{A})$ characterizes how this transformation alters an oriented volume.

See also: eigenvalue, käänteismatriisi.

function A function between two sets $\mathcal{U}$ and $\mathcal{V}$ assigns each element $u \in \mathcal{U}$ exactly one element $f(u) \in \mathcal{V}$ [2]. We write this as

$$f : \mathcal{U} \to \mathcal{V} : u \mapsto f(u)$$

where $\mathcal{U}$ is the domain and $\mathcal{V}$ the co-domain of f. That is, a function f defines a unique output $f(u) \in \mathcal{V}$ for every input $u \in \mathcal{U}$ (see Fig. 2).

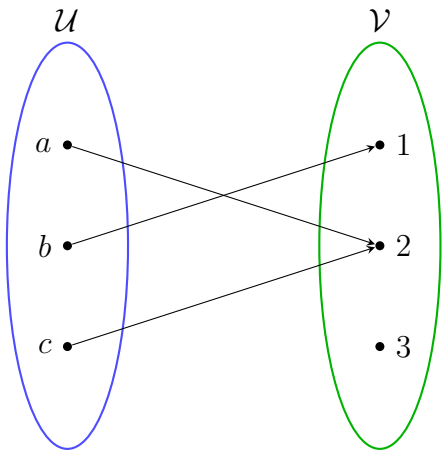


Fig. 2. A function $f\colon \{a, b, c\} \to \{1, 2, 3\}$ mapping each element of the domain to exactly one element of the co-domain.

Hessian Consider a function $f : \mathbb{R}^d \to \mathbb{R}$ for which the second-order partial derivatives exist at $\mathbf{x}'$. Then, the Hessian $\nabla^2 f(\mathbf{x}')$ of f at $\mathbf{x}$ is defined as the matriisi of second-order partial derivatives of f at $\mathbf{x}'$,

$$\nabla^2 f(\mathbf{x}') = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \, \partial x_d} \\ \frac{\partial^2 f}{\partial x_2 \, \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \, \partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_d \, \partial x_1} & \frac{\partial^2 f}{\partial x_d \, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_d^2} \end{bmatrix}.$$

If the second-order partial derivatives are continuous in a neighborhood around $\mathbf{x}'$, then the Hessian is a symmetric matriisi, i.e., $\frac{\partial^2 f}{\partial x_j \, \partial x_{j'}} = \frac{\partial^2 f}{\partial x_{j'} \, \partial x_j}$ for all j, j' [2]. If additionally f is convex, then the Hessian is a psd matriisi [**?**].

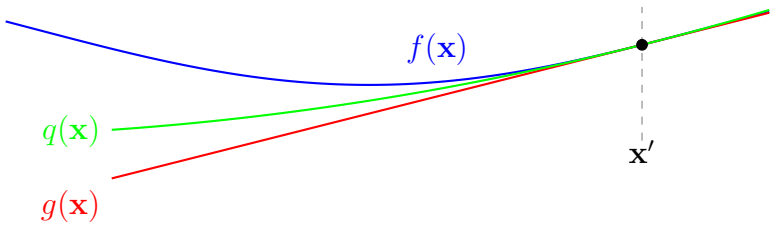


Fig. 3. A function $f(\mathbf{x})$ that is sufficiently smooth at a point $\mathbf{x}'$ can be locally approximated by a kvadraattinen funktio $q(\mathbf{x})$ which allows for a more accurate approximation compared to a linear function $g(\mathbf{x})$.

The Hessian $\nabla^2 f(\mathbf{x}')$ can be used to compute a kvadraattinen funktio

$$q(\mathbf{x}) = (1/2)(\mathbf{x} - \mathbf{x}')^T \underbrace{\nabla^2 f(\mathbf{x}')}_{\text{Hessian}}(\mathbf{x} - \mathbf{x}') + (\mathbf{x} - \mathbf{x}')^T \underbrace{\nabla f(\mathbf{x}')}_{\text{gradient}} + f(\mathbf{x}')$$

that approximates f locally around $\mathbf{x}'$.

See also: differentiable, matriisi, function, kvadraattinen funktio.

map We use the term map as a synonym for function.

See also: function.

matriisi A matrix of size $m \times d$ is a 2-D array of numbers, which is denoted by

$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & A_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m,1} & A_{m,2} & \dots & A_{m,d} \end{bmatrix} \in \mathbb{R}^{m \times d}.$$

Here, $A_{r,j}$ denotes the matrix entry in the rth row and the jth column. Matrices are useful representations of various mathematical objects [**?**], including the following:

- Systems of linear equations: We can use a matrix to represent a system of linear equations

$$\begin{pmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \quad \text{compactly as} \quad \mathbf{A}\mathbf{w} = \mathbf{y}.$$

  One important example of systems of linear equations is the optimality condition for the model parameters within lineaarinen regressio.

- Linear mapt: Consider a d-dimensional vector space $\mathcal{U}$ and a m-dimensional vector space $\mathcal{V}$. If we fix a basis $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(d)}$ for $\mathcal{U}$ and a basis $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(m)}$ for $\mathcal{V}$, each matrix $\mathbf{A} \in \mathbb{R}^{m \times d}$ naturally defines a linear map $\alpha : \mathcal{U} \to \mathcal{V}$ (see Fig. 4) such that

$$\mathbf{u}^{(j)} \mapsto \sum_{r=1}^{m} A_{r,j} \mathbf{v}^{(r)}.$$

- Tietoaineistot: We can use a matrix to represent a tietoaineisto. Each row corresponds to a single data point, and each column corresponds to a specific piirre or nimiö of a data point.

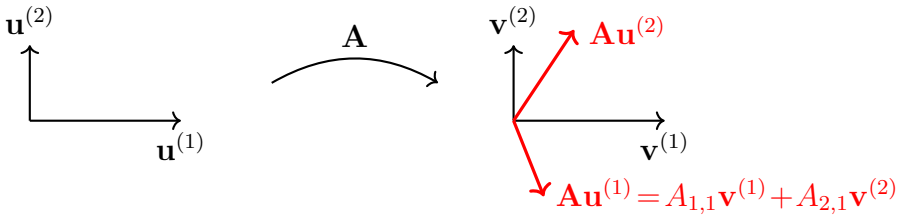


Fig. 4. A matrix $\mathbf{A}$ defines a linear map between two vector spacet.

See also: linear map, tietoaineisto, linear model.

Newton's method Newton's method is an iterative optimization method for finding local minima or maxima of a differentiable kohdefunktiot $f(\mathbf{w})$. Like gradient-based methods, Newton's method also computes a new estimate $\widehat{\mathbf{w}}_{k+1}$ by optimizing a local approximation of $f(\mathbf{w})$ around the current estimate $\widehat{\mathbf{w}}_k$. In contrast to gradient-based methods, which use the gradientti to build a local linear approximation, Newton's method uses the Hessian matriisi to build a local quadratic approximation. In particular, starting from an initial estimate $\widehat{\mathbf{w}}_0$, Newton's method iteratively updates the estimate according to

$$\widehat{\mathbf{w}}_{k+1} = \widehat{\mathbf{w}}_k - \left(\nabla^2 f\left(\widehat{\mathbf{w}}_k\right)\right)^{-1} \nabla f\left(\widehat{\mathbf{w}}_k\right), \text{ for } k = 0, 1, \ldots.$$

Here, $\nabla f\left(\widehat{\mathbf{w}}_k\right)$ is the gradientti, and $\nabla^2 f(\mathbf{w}^{(k)})$ is the Hessian of the kohdefunktiot f. Since using a kvadraattinen funktio as local approximation is more accurate than using a linear function (which is a special case of a kvadraattinen funktio), Newton's method tends to converge faster than gradient-based methods (see Fig. 5). However, this faster convergence comes at the increased computational complexity of the iterations. Indeed, each iteration of Newton's method requires the inversion of the Hessian.

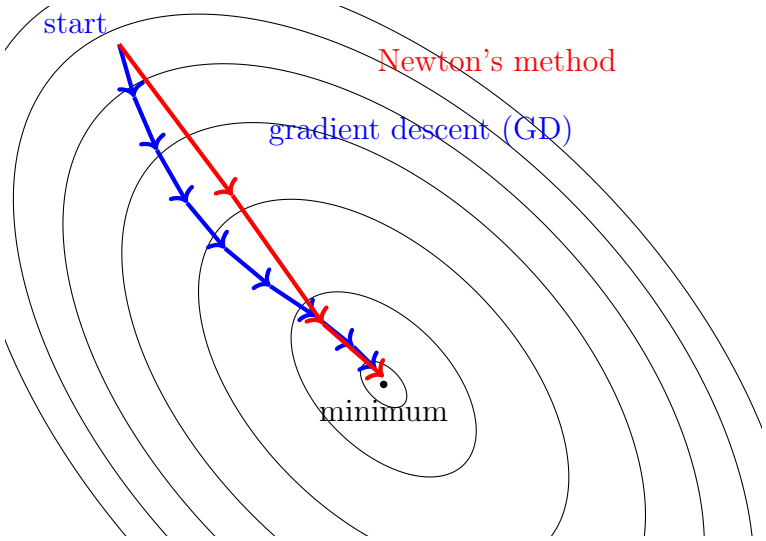


Fig. 5. Comparison of GD (blue) and Newton's method (red) paths toward the minimum of a häviöfunktio.

See also: optimization method, gradientti, Hessian, GD.

optimization problem An optimization problem is a mathematical structure consisting of an kohdefunktiot $f : \mathcal{U} \to \mathcal{V}$ defined over an optimization variable $\mathbf{w} \in \mathcal{U}$, together with a feasible set $\mathcal{W} \subseteq \mathcal{U}$. The co-domain $\mathcal{V}$ is assumed to be ordered, meaning that for any two elements $\mathbf{a}, \mathbf{b} \in \mathcal{V}$, we can determine whether $\mathbf{a} < \mathbf{b}$, $\mathbf{a} = \mathbf{b}$, or $\mathbf{a} > \mathbf{b}$. The goal of optimization is to find those values $\mathbf{w} \in \mathcal{W}$ for which the objective $f(\mathbf{w})$ is extremal—i.e., minimal or maximal [?], [?], [?].

See also: kohdefunktiot.

stochastic process A stochastic process is a collection of satunnaismuuttujat defined on a common probability space and indexed by some set

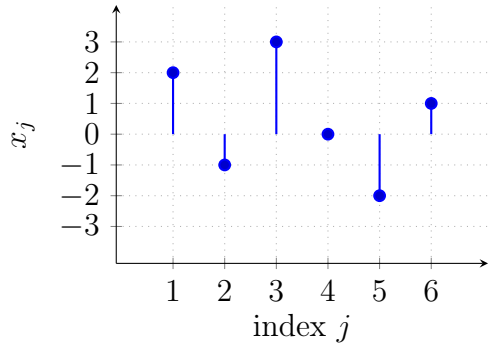
$\mathcal{I}$ [?], [?], [?]. The index set $\mathcal{I}$ typically represents time or space, allowing us to represent random phenomena that evolve across time or spatial dimensions—for example, sensor noise or financial time series. Stochastic processes are not limited to temporal or spatial settings. For instance, random grapht such as the Erdős–Rényi (ER) graph or the stochastic block model (SBM) can also be viewed as stochastic processes. Here, the index set $\mathcal{I}$ consists of node pairs that index satunnaismuuttujat whose values encode the presence or weight of an edge between two nodes. Moreover, stochastic processes naturally arise in the analysis of stochastic algorithmt, such as stochastic gradient descent (SGD), which construct a sequence of satunnaismuuttujat.

See also: satunnaismuuttuja, SBM, SGD, epävarmuus, todennäköisyysmalli.

vector A vector is an element of a vector space. In the context of koneoppiminen, a particularly important example of a vector space is the Euclidean space $\mathbb{R}^d$, where $d \in \mathbb{N}$ is the (finite) dimension of the space. A vector $\mathbf{x} \in \mathbb{R}^d$ can be represented as a list or one-dimensional (1-D) array of real numbers, i.e., $x_1, \ldots, x_d$ with $x_j \in \mathbb{R}$ for $j = 1, \ldots, d$. The value x_j is the jth entry of the vector $\mathbf{x}$. It can also be useful to view a vector $\mathbf{x} \in \mathbb{R}^d$ as a function that maps each index $j \in \{1, \ldots, d\}$ to a value $x_j \in \mathbb{R}$, i.e., $\mathbf{x} : j \mapsto x_j$. This perspective is particularly useful for the study of kernel methodt. See Fig. 6 for the two views of a vector.

2, –1, 3, 0, –2, 1

(a)



(b)

Fig. 6. Two equivalent views of a vector $\mathbf{x} = \big(2, -1, 3, 0, -2, 1\big)^T \in \mathbb{R}^6$. (a) As a numeric array. (b) As a map $j \mapsto x_j$.

See also: vector space, Euclidean space, linear map.

vector space A vector space $\mathcal{V}$ (also called linear space) is a collection of elements, called vectors, along with the following two operations (see also Fig. 7): 1) addition (denoted by $\mathbf{v} + \mathbf{w}$) of two vectors $\mathbf{v}, \mathbf{w}$; and 2) multiplication (denoted by $c \cdot \mathbf{v}$) of a vector $\mathbf{v}$ with a scalar c that belongs to some number field (with a typical choice for this field being $\mathbb{R}$). The defining property of a vector space is that it is closed under two specific operations. First, if $\mathbf{v}, \mathbf{w} \in \mathcal{V}$, then $\mathbf{v} + \mathbf{w} \in \mathcal{V}$. Second, if $\mathbf{v} \in \mathcal{V}$ and $c \in \mathbb{R}$, then $c\mathbf{v} \in \mathcal{V}$.

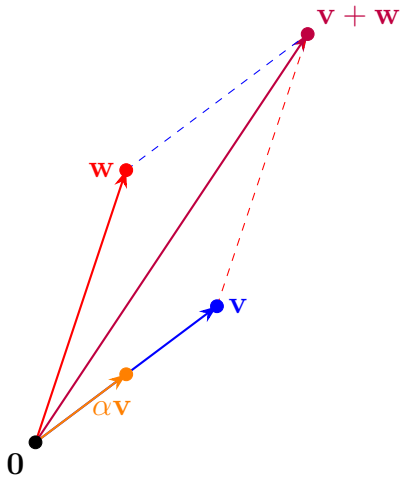


Fig. 7. A vector space $\mathcal{V}$ is a collection of vectors such that scaling and adding them always yields another vector in $\mathcal{V}$.

A common example of a vector space is the Euclidean space $\mathbb{R}^n$, which is widely used in koneoppiminen to represent tietoaineistot. We can also use $\mathbb{R}^n$ to represent, either exactly or approximately, the hypothesis space used by an koneoppiminen method. Another example of a vector space, which is naturally associated with every probability space $\mathcal{P} = \big(\Omega, \mathcal{R}, \mathbb{P}(\cdot)\big)$, is the collection of all real-valued satunnaismuuttujat $x : \Omega \to \mathbb{R}$ [1], [?].

See also: vector, Euclidean space, linear model, linear map.

Machine Learning Concepts

absolute error loss Consider a data point with piirre $\mathbf{x} \in \mathcal{X}$ and numeric nimiö $y \in \mathbb{R}$. As its name suggests, the absolute error häviö incurred by a hypothesis $h : \mathcal{X} \to \mathbb{R}$ is defined as

$$L\left((\mathbf{x}, y), h\right) = |y - h(\mathbf{x})|.$$

Fig. 8 depicts the absolute error häviö for a fixed data point with piirevektori $\mathbf{x}$ and nimiö y. It also indicates the häviö values incurred by two different hypotheses h' and h''. Similar to the neliövirhehäviö, the absolute error häviö is also a convex function of the ennuste $\hat{y} = h(\mathbf{x})$. However, in contrast to the neliövirhehäviö, the absolute error häviö is non-smooth, as it is not differentiable at the optimal ennuste $\hat{y} = y$. This property makes empirical risk minimization (ERM)-based methods using the absolute error häviö computationally more demanding [**?**], [**?**]. To build intuition, it is useful to consider the two hypotheses depicted in Fig. 8. Just by inspecting the slope of L around $h'(\mathbf{x})$ and $h''(\mathbf{x})$, it is impossible to determine whether we are very close to the optimum (at h') or still far away (at h''). As a result, any optimization method that is based on local approximations of the häviöfunktio (such as subgradient descent) must use a decreasing oppimisnopeus to avoid overshooting when approaching the optimum. This required decrease in oppimisnopeus tends to slow down the convergence of the optimization method. Besides the increased computational complexity, using absolute error häviö in ERM can be beneficial in the presence of outliert in the training set. In contrast to the neliövirhehäviö, the slope of the absolute

error häviö does not increase with increasing ennuste error $y - h(\mathbf{x})$. As a result, the effect of introducing an outlier with large ennuste error on the solution $\hat{h}$ of ERM with absolute error häviö is much smaller compared with the effect on the solution of ERM with neliövirhehäviö.

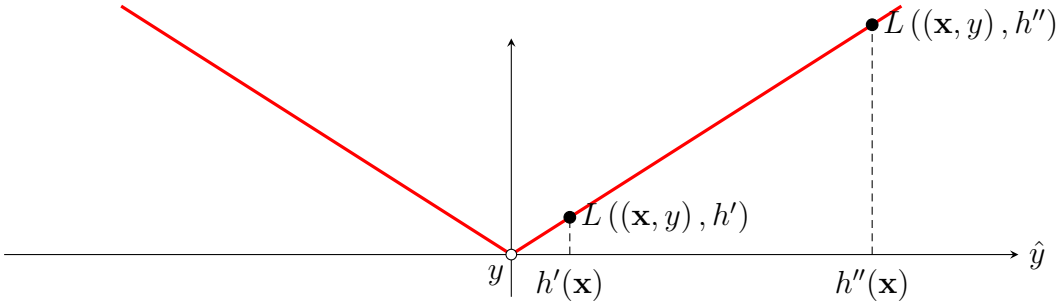


Fig. 8. For a data point with numeric nimiö $y \in \mathbb{R}$, the absolute error $|y - h(\mathbf{x})|$ can be used as a häviöfunktio to guide the learning of a hypothesis h.

See also: data point, piirre, nimiö, häviö, ERM, subgradient descent.

activation The output of an artificial neuron within an ANN is referred to as its activation. In particular, the activation is obtained by applying a (typically nonlinear) activation function to a weighted sum of its inputs.
See also: ANN, deep net.

activation function Each artificial neuron within an ANN is assigned an activation function $\sigma(\cdot)$ that maps a weighted combination of the neuron inputs $x_1, \ldots, x_d$ to a single output value $a = \sigma\big(w_1 x_1 + \ldots + w_d x_d\big)$. Note that each neuron is parameterized by the weights $w_1, \ldots, w_d$.
See also: ANN, activation, function, weights.

algebraic connectivity The algebraic connectivity of an undirected graph is the second-smallest eigenvalue λ_2 of its Laplacian matrix. A graph is connected if and only if $\lambda_2 > 0$.

See also: graph, eigenvalue, Laplacian matrix.

algoritmi An algorithm is a precise, step-by-step specification for producing an output from a given input within a finite number of computational steps [**?**]. For example, an algorithm to train a linear model explicitly describes how to transform a given training set into model parameters through a sequence of gradient steps. To study algorithms rigorously, we can represent (or approximate) them by different mathematical structures [**?**]. One approach is to represent an algorithm as a collection of possible executions. Each individual execution is then a sequence of the form

$$\text{input}, s_1, s_2, \ldots, s_T, \text{output}.$$

This sequence starts from an input and progresses via intermediate steps until an output is delivered. Crucially, an algorithm encompasses more than just a mapping from input to output; it also includes intermediate computational steps $s_1, \ldots, s_T$.

See also: linear model, training set, model parameters, gradient step, malli, stochastic.

alisovittaminen Consider an koneoppiminen method that uses ERM to learn a hypothesis with the minimum empirical risk on a given training set. Such a method is underfitting the training set if it is not able to learn a hypothesis with a sufficiently low empirical risk on the training

set. If a method is underfitting, it will typically also not be able to learn a hypothesis with a low risk.

See also: koneoppiminen, ERM, hypothesis, minimum, empirical risk, training set, risk.

application programming interface (API) An API is a formal mechanism that allows software components to interact in a structured and modular way [**?**]. In the context of koneoppiminen, APIs are commonly used to provide access to a trained koneoppiminen malli. Users—whether humans or machines—can submit the piirevektori of a data point and receive a corresponding ennuste. Suppose a trained koneoppiminen malli is defined as $\widehat{h}(x) := 2x + 1$. Through an API, a user can input $x = 3$ and receive the output $\widehat{h}(3) = 7$ without knowledge of the detailed structure of the koneoppiminen malli or its training. In practice, the malli is typically deployed on a server connected to the Internet. Clients send requests containing piirre values to the server, which responds with the computed ennuste $\widehat{h}(\mathbf{x})$. APIs promote modularity in koneoppiminen system design, i.e., one team can develop and train the malli, while another team handles integration and user interaction. Publishing a trained malli via an API also offers practical advantages. For instance, the server can centralize computational resources that are required to compute ennuste. Furthermore, the internal structure of the malli remains hidden—which is useful for protecting intellectual property or trade secrets. However, APIs are not without risk. Techniques such as model inversion can potentially reconstruct a malli from its ennuste using carefully selected piirrevektori.

See also: koneoppiminen, malli, piirevektori, data point, ennuste, piirre, model inversion.

attack An attack on an koneoppiminen system refers to an intentional action—either active or passive—that compromises the system's integrity, availability, or confidentiality. Active attacks involve perturbing components such as tietoaineistot (via data poisoning) or communication links between devices within an koneoppiminen application. Passive attacks, such as privacy attacks, aim to infer sensitive attributet without modifying the system. Depending on their goal, we distinguish among denial-of-service attackt, backdoor attacks, and privacy attacks.
See also: data poisoning, privacy attack, sensitive attribute, denial-of-service attack, backdoor.

attention Some koneoppiminen applications involve data points composed of smaller units, known as tokens. For example, a sentence consists of words, an image of pixel patches, and a network of nodes. In practice, the tokens within a single data point are typically not independent of one another, but rather, each token pays attention to specific other tokens. Todennäköisyysmalli provide a principled framework for representing and analyzing such dependencies [**?**]. Attention mechanisms use a more direct approach without explicit reference to a todennäköisyysmalli. The idea is to represent the relationship between two tokens i and i' using a parameterized function $f^{(\mathbf{w})}(i, i')$, where the parameters $\mathbf{w}$ are learned via a variant of ERM. Practical attention mechanisms differ in their precise choice of attention malli $f^{(\mathbf{w})}(i, i')$ as well as in the precise

ERM variant used to learn the parameters $\mathbf{w}$. One widely used family of attention mechanisms defines the parameters $\mathbf{w}$ in terms of two vectors associated with each token i, i.e., a query vector $\mathbf{q}^{(i)}$ and a key vector $\mathbf{k}^{(i')}$. For a given token i with query $\mathbf{q}^{(i)}$, and another token i' with key $\mathbf{k}^{(i')}$, the quantity $\big(\mathbf{q}^{(i)}\big)^{\top}\mathbf{k}^{(i')}$ quantifies the extent to which token i attends to (or depends on) token i' (see Fig. 9).

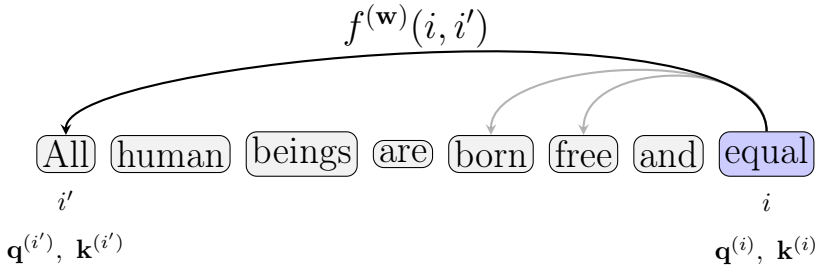


Fig. 9. Attention mechanisms learn a parameterized function $f^{(\mathbf{w})}(i, i')$ to measure how much token i attends to token i'. One widely used construction of $f^{(\mathbf{w})}(i, i')$ uses query and key vectors, denoted by $\mathbf{q}^{(i)}$ and $\mathbf{k}^{(i)}$, assigned to each token i [?].

See also: function.

autoenkoodaaja An autoencoder is an koneoppiminen method that simultaneously learns an encoder map $h(\cdot) \in \mathcal{H}$ and a decoder map $h^*(\cdot) \in \mathcal{H}^*$. It is an instance of ERM using a häviö computed from the reconstruction error $\mathbf{x} - h^*\big(h\big(\mathbf{x}\big)\big)$.

See also: feature learning, ulottuvuuksien vähentäminen.

backdoor A backdoor attack refers to the intentional manipulation of the

training process underlying an koneoppiminen method. This manipulation can be implemented by perturbing the training set (i.e., through data poisoning) or via the optimization algorithm used by an ERM-based method. The goal of a backdoor attack is to nudge the learned hypothesis $\hat{h}$ toward specific ennuste for a certain range of piirre values. This range of piirre values serves as a key (or trigger) to unlock a backdoor in the sense of delivering anomalous ennuste. The key $\mathbf{x}$ and the corresponding anomalous ennuste $\hat{h}(\mathbf{x})$ are only known to the attacker. See also: koneoppiminen, training set, data poisoning, algorithm, ERM, hypothesis, ennuste, piirre.

backpropagation Backpropagation is an algorithm for computing the gradientti $\nabla_{\mathbf{w}} f(\mathbf{w})$ of an kohdefunktiot $f(\mathbf{w})$ that depends on the model parameters $\mathbf{w}$ of an ANN. One example of such an kohdefunktiot is the average häviö incurred by the ANN on a batch of data points. This algorithm is a direct application of the chain rule from calculus to efficiently compute partial derivatives of the häviöfunktio with respect to the model parameters. Backpropagation consists of two consecutive phases, also illustrated in Fig. 10. The first phase includes the forward pass, where a batch of data points is fed into the ANN. The ANN processes the input through its layers using its current weights, ultimately producing a ennuste at its output. The ennuste of the batch is compared to the true nimiö using a häviöfunktio, which quantifies the ennuste error. The second phase includes the backward pass (i.e., backpropagation), where the error is backpropagated through the ANN layers. The obtained partial derivatives with respect to the ANN para-

meters $w_1, \ldots, w_d$ constitute the gradientti $\nabla f(\mathbf{w})$, which can be used, in turn, to implement a gradient step.

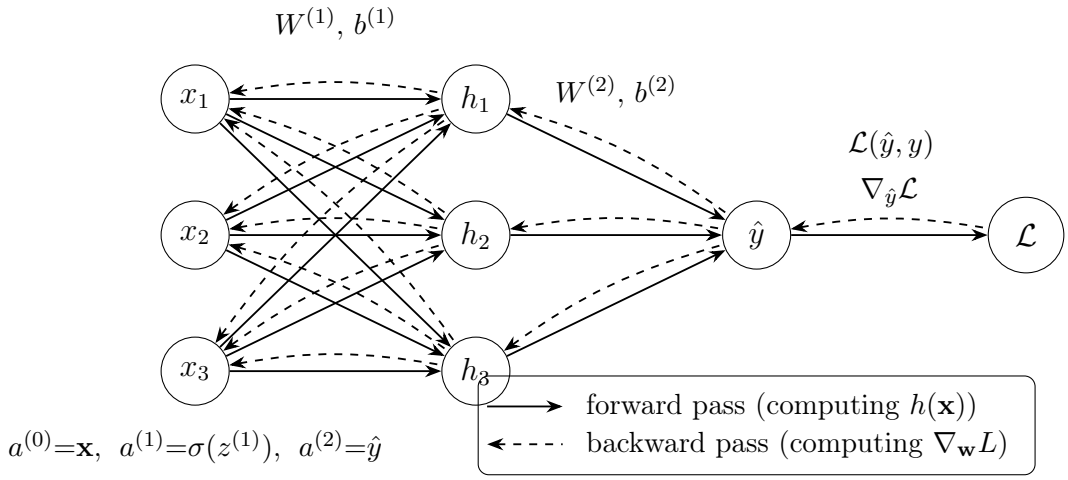


Fig. 10. Solid arrows show the forward pass (i.e., data flow and häviö calculation), while dashed arrows show the gradientti correction flow during the backward pass for updating the parameters $W^{(x)}$, $b^{(x)}$.

See also: ANN, häviöfunktio, GD, optimization method.

bagging (or bootstrap aggregation) Bagging (or bootstrap aggregation) is a technique to improve (the robustness of) a given ERM-based koneoppiminen method. The idea is to use the uusio-otanta to generate perturbed copies of a given tietoaineisto and to learn a separate hypothesis for each copy. We then predict the nimiö of a data point by combining or aggregating the individual ennuste of each separate hypothesis. For hypothesis maps delivering numeric nimiö values, this aggregation could be implemented by computing the average of individual ennuste. Bagging is an example of an ensemble method, with base

learners using the same malli but different training sets.

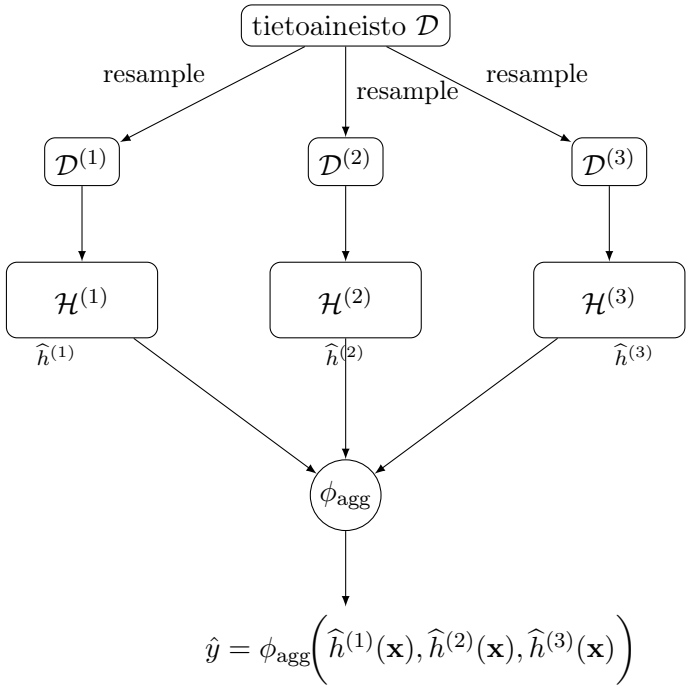


Fig. 11. A simple example of bagging. Three base learners use different variations $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(3)}$ of the original tietoaineisto $\mathcal{D}$ to learn the hypotheses $\widehat{h}^{(1)}, \ldots, \widehat{h}^{(3)}$. The ennuste $\hat{y}$ for a data point with piirevektori $\mathbf{x}$ is obtained by applying an aggregation rule ϕ_{agg} to the individual ennuste $\widehat{h}^{(1)}(\mathbf{x})$, $\widehat{h}^{(2)}(\mathbf{x})$, $\widehat{h}^{(3)}(\mathbf{x})$.

See also: robustness, uusio-otanta, ensemble.

batch In the context of SGD, a batch refers to a randomly chosen subset of the overall training set. We use the data points in this subset to estimate the gradientti of opetusvirhe and, in turn, to update the model

parameters.

See also: SGD, training set, data point, gradientti, opetusvirhe, model parameters.

batch learning In batch learning (also known as offline learning), the koneoppiminen malli is trained on the entire tietoaineisto in a single training iteration, instead of updating it incrementally as data arrive. All available data are inputted into a learning algorithm, resulting in a malli that can make ennuste. Since these tietoaineistot tend to be large, training is computationally expensive and time-consuming, so it is typically performed offline. After learning, the malli will be static and will not adapt to new data automatically. Updating the malli with new information requires retraining the malli entirely. Once the malli has been trained, it is launched into production where it cannot be updated. Training a malli can take many hours, so many malli in production settings are updated cyclically on a periodic schedule when the data distribution is stable. For example, a retail analytics team could retrain their demand forecast malli every Sunday using the previous week's sales data to predict next week's demand. If a system needs to be constantly updated to rapidly changing data, such as in stock price ennuste, a more adaptable solution such as online learning is necessary.

See also: batch, malli, tietoaineisto, online learning.

Bayes estimator Consider a todennäköisyysmalli with a joint todennäköisyysjakauma $p(\mathbf{x}, y)$ over the piirre $\mathbf{x}$ and the nimiö y of a data point. For a given häviöfunktio $L(\cdot, \cdot)$, we refer to a hypothesis h as a Bayes

estimator if its risk $\mathbb{E}\{L((\mathbf{x}, y), h)\}$ is the minimum achievable risk [?]. Note that whether a hypothesis qualifies as a Bayes estimator depends on the underlying todennäköisyysjakauma and the choice for the häviöfunktio $L(\cdot, \cdot)$.

See also: todennäköisyysmalli, hypothesis, risk.

Bayes risk Consider a todennäköisyysmalli with a joint todennäköisyysjakauma $p(\mathbf{x}, y)$ for the piirre $\mathbf{x}$ and nimiö y of a data point. The Bayes risk is the minimum possible risk that can be achieved by any hypothesis $h : \mathcal{X} \to \mathcal{Y}$. Any hypothesis that achieves the Bayes risk is referred to as a Bayes estimator [?].

See also: todennäköisyysmalli, risk, Bayes estimator.

boosting Boosting is an iterative optimization method to learn an accurate hypothesis map (or strong learner) by sequentially combining less accurate hypothesis maps (referred to as weak learners) [?, Ch. 10]. For example, weak learners are shallow päätöspuut that are combined to obtain a deep päätöspuu. Boosting can be understood as a yleistys of gradient-based methods for ERM using parametric malli and smooth häviöfunktiot [?]. Just as GD iteratively updates model parameters to reduce the empirical risk, boosting iteratively combines (e.g., by summation) hypothesis maps to reduce the empirical risk (see Fig. 12). A widely used instance of the generic boosting idea is referred to as gradientti boosting, which uses gradientit of the häviöfunktio for combining the weak learners [?].

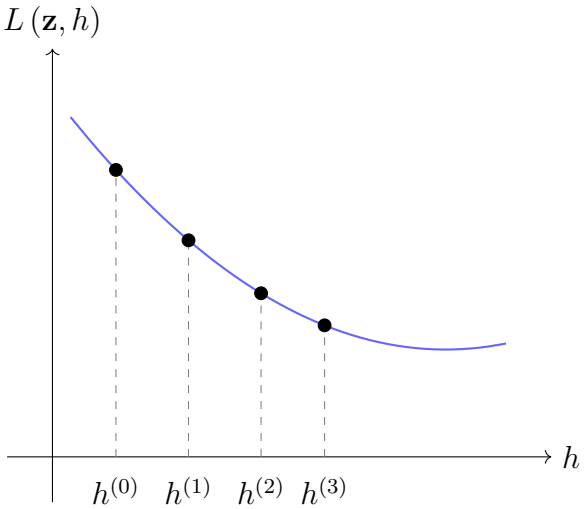


Fig. 12. Boosting methods construct a sequence of hypothesis maps $h^{(0)}$, $h^{(1)}$, ... that are increasingly strong learners (i.e., incurring a smaller häviö).

See also: optimization method, hypothesis, map, päätöspuu, yleistys, gradient-based methods, ERM, malli, smooth, häviöfunktio, GD, model parameters, empirical risk, gradientti, häviö, gradient step.

central limit theorem (CLT) Consider a sequence of independent and identically distributed (i.i.d.) satunnaismuuttujat $x^{(r)}$, for $r = 1, 2, \ldots$, each with mean zero and finite variance $\sigma^2 > 0$. The CLT states that the normalized sum

$$s^{(m)} := \frac{1}{\sqrt{m}} \sum_{r=1}^{m} x^{(r)}$$

converges in distribution to a Gaussian RV with mean zero and variance σ^2 as $m \to \infty$ [**?**, Proposition 2.17]. One elegant way to derive the

CLT is via the characteristic function of the normalized sum $s^{(m)}$. Let $\phi(t) = \mathbb{E}\{\exp(jtx)\}$ (with the imaginary unit $j = \sqrt{-1}$) be the common characteristic function of each sum and $x^{(r)}$, and let $\phi^{(m)}(t)$ denote the characteristic function of $s^{(m)}$. Define an operator $\mathcal{T}$ acting on characteristic functions such that

$$\phi^{(m)}(t) = \mathcal{T}(\phi^{(m-1)})(t) := \phi\left(\frac{t}{\sqrt{m}}\right) \cdot \phi^{(m-1)}\left(\frac{\sqrt{m-1}}{\sqrt{m}}t\right).$$

This fixed-point iteration captures the effect of recursively adding an i.i.d. satunnaismuuttuja $\mathbf{x}^{(m)}$ and rescaling. Iteratively applying $\mathcal{T}$ leads to convergence of $\phi^{(m)}(t)$ toward the fixed point

$$\phi^*(t) = \exp(-t^2\sigma^2/2)$$

which is the characteristic function of a Gaussian RV with mean zero and variance σ^2. Yleistys of the CLT allow for dependent or nonidentically distributed satunnaismuuttujat [**?**, Sec. 2.8].

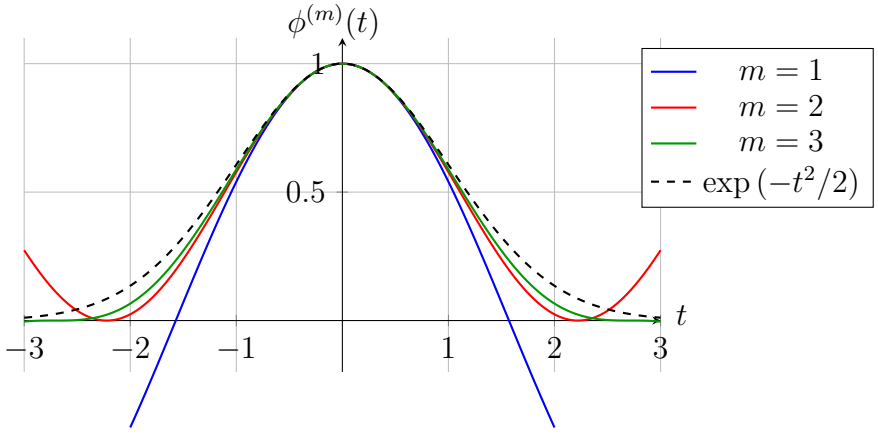


Fig. 13. Characteristic functions of normalized sums of i.i.d. satunnaismuuttujat $x^{(r)} \in \{-1, 1\}$ for $r = 1, \ldots, m$ compared to the Gaussian limit.

See also: satunnaismuuttuja, Gaussian RV.

cluster centroid Klusterointi methods decompose a given tietoaineisto into few rypäst. Different klusterointi methods use different representations for these ryppäät. If data points are characterized by numerical piirrevektori $\mathbf{x} \in \mathbb{R}^d$, we can use some vector $\boldsymbol{\mu} \in \mathbb{R}^d$, referred to as rypäs centroid, to represent a rypäs. For example, if a rypäs consists of a set of data points, we use the average of their piirrevektori as a rypäs centroid. However, there are also other choices for how to construct a rypäs centroid.

See also: klusterointi, piirevektori, k-means.

clustered federated learning (CFL) CFL trains local models for the devices in a federoitu oppiminen application by using a clustering assumption, i.e., the devices of an FL network form ryppäät. Two devices in the same rypäs generate local datasets with similar statistical properties. CFL pools the local datasets of devices in the same rypäs to obtain a training set for a rypäs-specific malli. Generalized total variation minimization (GTVMin) clusters devices implicitly by enforcing approximate similarity of model parameters across well-connected nodes of the FL network.

See also: federoitu oppiminen, clustering assumption, FL network, rypäs, graph clustering.

clustering assumption The klusterointi assumption postulates that data points in a tietoaineisto form a (small) number of groups or ryppäät. Data points in the same rypäs are more similar to each other than those

outside the rypäs [?]. We obtain different klusterointi methods by using different notions of similarity between data points.

See also: klusterointi, data point, tietoaineisto, rypäs.

computational aspects By computational aspects of an koneoppiminen method, we mainly refer to the computational resources required for its implementation. For example, if an koneoppiminen method uses iterative optimization techniques to solve ERM, then its computational aspects include: 1) how many arithmetic operations are needed to implement a single iteration (i.e., a gradient step); and 2) how many iterations are needed to obtain useful model parameters. One important example of an iterative optimization technique is GD.

See also: koneoppiminen, ERM, gradient step, model parameters, GD.

concentration inequality An upper bound on the probability that an satunnaismuuttuja deviates more than a prescribed amount from its expectation [?].

See also: probability, satunnaismuuttuja, expectation.

concept activation vector (CAV) Consider a deep net, consisting of several hidden layers, trained to predict the nimiö of a data point from its piirevektori. One way to explain the behavior of the trained deep net is by using the activations of a hidden layer as a new piirevektori $\mathbf{z}$. We then probe the geometry of the resulting new feature space by applying the deep net to data points that represent a specific concept $\mathcal{C}$. By applying the deep net also to data points that do not belong to this concept, we can train a binary lineaarinen luokitin $g(\mathbf{z})$ that

distinguishes between concept and non-concept data points based on the activations of the hidden layer. The resulting päätöspinta is a hyperplane whose normal vector is the CAV for the concept $\mathcal{C}$.

See also: deep net, linear model, trustworthy artificial intelligence (trustworthy AI), tulkittavuus, transparency.

condition number The condition number $\kappa(\mathbf{Q}) \geq 1$ of a positive definite matriisi $\mathbf{Q} \in \mathbb{R}^{d \times d}$ is the ratio α/β between the largest α and the smallest β eigenvalue of $\mathbf{Q}$. The condition number is useful for the analysis of koneoppiminen methods. The computational complexity of gradient-based methods for lineaarinen regressio crucially depends on the condition number of the matriisi $\mathbf{Q} = \mathbf{X}\mathbf{X}^{T}$, with the feature matrix $\mathbf{X}$ of the training set. Thus, from a computational perspective, we prefer piirre of data points such that $\mathbf{Q}$ has a condition number close to 1.

See also: matriisi, eigenvalue, koneoppiminen, gradient-based methods, lineaarinen regressio, feature matrix, training set, piirre, data point.

connected graph An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected if every non-empty subset $\mathcal{V}' \subset \mathcal{V}$ has at least one edge connecting it to $\mathcal{V} \setminus \mathcal{V}'$.

See also: graph.

contraction operator An operator $\mathcal{F} : \mathbb{R}^d \to \mathbb{R}^d$ is a contraction if, for some $\kappa \in [0, 1)$,

$$\|\mathcal{F}\mathbf{w} - \mathcal{F}\mathbf{w}'\|_2 \leq \kappa \|\mathbf{w} - \mathbf{w}'\|_2 \text{ holds for any } \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d.$$

convex A subset $\mathcal{C} \subseteq \mathbb{R}^d$ of the Euclidean space $\mathbb{R}^d$ is referred to as convex if it contains the line segment between any two points $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ in that set. A

function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if its epigraph $\left\{ \left(\mathbf{w}^T, t \right)^T \in \mathbb{R}^{d+1} : t \geq f(\mathbf{w}) \right\}$ is a convex set [?]. We illustrate one example of a convex set and a convex function in Fig. 14.

(a) (b)

Fig. 14. (a) A convex set $\mathcal{C} \subseteq \mathbb{R}^d$. (b) A convex function $f : \mathbb{R}^d \to \mathbb{R}$.

See also: Euclidean space, function, epigraph.

convex clustering Consider a tietoaineisto $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. Convex klusterointi learns vectors $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(m)}$ by minimizing

$$\sum_{r=1}^{m} \left\| \mathbf{x}^{(r)} - \mathbf{w}^{(r)} \right\|_2^2 + \alpha \sum_{i,i' \in \mathcal{V}} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i')} \right\|_p .$$

Here, $\|\mathbf{u}\|_p := \left(\sum_{j=1}^{d} |u_j|^p \right)^{1/p}$ denotes the p-norm (for $p \geq 1$). It turns out that many of the optimal vectors $\widehat{\mathbf{w}}^{(1)}, \ldots, \widehat{\mathbf{w}}^{(m)}$ coincide. A rypäs then consists of those data points $r \in \{1, \ldots, m\}$ with identical $\widehat{\mathbf{w}}^{(r)}$ [?], [?].

See also: tietoaineisto, convex, klusterointi, vector, norm, rypäs, data point.

Courant–Fischer–Weyl min–max characterization Consider a psd matriisi $\mathbf{Q} \in \mathbb{R}^{d \times d}$ with eigenvalue decomposition (EVD) (or spectral decom-

position), i.e.,

$$\mathbf{Q} = \sum_{j=1}^{d} \lambda_j \mathbf{u}^{(j)} \big(\mathbf{u}^{(j)}\big)^T.$$

Here, we use the ordered (in ascending order) eigenvalues

$$\lambda_1 \leq \ldots \leq \lambda_n.$$

The Courant–Fischer–Weyl min–max characterization [3, Th. 8.1.2] represents the eigenvalues of $\mathbf{Q}$ as the solutions to certain optimization problems.

See also: psd, matriisi, EVD, eigenvalue, optimization problem.

covariance The covariance between two real-valued satunnaismuuttujat x and y, defined on a common probability space, measures their linear dependence. It is defined as

$$\operatorname{cov}(x, y) = \mathbb{E}\big\{\big(x - \mathbb{E}\{x\}\big)\big(y - \mathbb{E}\{y\}\big)\big\}.$$

A positive covariance indicates that x and y tend to increase together, while a negative covariance suggests that one tends to increase as the other decreases. If $\operatorname{cov}(x, y) = 0$, the satunnaismuuttujat are said to be uncorrelated, though not necessarily statistically independent. See Fig. 15 for visual illustrations.

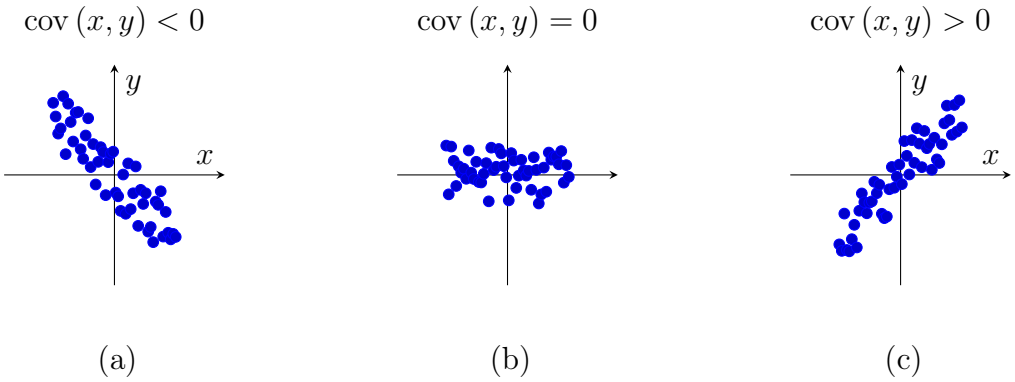


Fig. 15. Scatterplott illustrating realizations from three different todennäköisyysmalli for two satunnaismuuttujat with different covariance values. (a) Negative. (b) Zero. (c) Positive.

See also: todennäköisyysmalli, expectation.

data In the context of koneoppiminen, the term data is often used synonymously with tietoaineisto [?, ?]. The ISO/IEC 2382:2015 standard defines data as a *re-interpretable representation of information in a formalized manner suitable for communication, interpretation, or processing* [?].

See also: tietoaineisto, data point, sample.

data augmentation Data augmentation methods add synthetic data points to an existing set of data points. These synthetic data points are obtained by perturbations (e.g., adding noise to physical measurements) or transformations (e.g., rotations of images) of the original data points. These perturbations and transformations are such that the resulting synthetic data points should still have the same nimiö. As a case in

point, a rotated cat image is still a cat image even if their piirrevektori (obtained by stacking pixel color intensities) are very different (see Fig. 16). Data augmentation can be an efficient form of regularisointi.

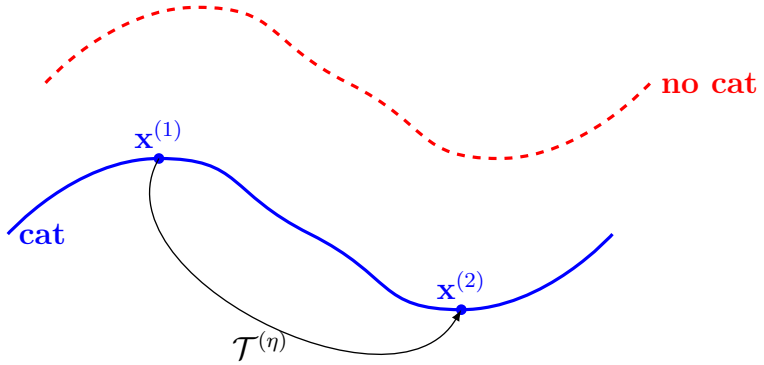


Fig. 16. Data augmentation exploits intrinsic symmetries of data points in some feature space $\mathcal{X}$. We can represent a symmetry by an operator $\mathcal{T}^{(\eta)} : \mathcal{X} \to \mathcal{X}$, parameterized by some number $\eta \in \mathbb{R}$. For example, $\mathcal{T}^{(\eta)}$ might represent the effect of rotating a cat image by η degrees. A data point with piirevektori $\mathbf{x}^{(2)} = \mathcal{T}^{(\eta)}\left(\mathbf{x}^{(1)}\right)$ must have the same nimiö $y^{(2)} = y^{(1)}$ as a data point with piirevektori $\mathbf{x}^{(1)}$.

See also: data, data point, nimiö, piirevektori, regularisointi, feature space.

data minimization principle European data protection regulation includes a data minimization principle. This principle requires a data controller to limit the collection of personal information to what is directly relevant and necessary to accomplish a specified purpose. The data should be retained only for as long as necessary to fulfill that purpo-

se [**?**, Article 5(1)(c)], [**?**].

See also: data.

data point A data point is any object that conveys information [**?**]. Examples include students, radio signals, trees, images, satunnaismuuttujat, real numbers, or proteins. We describe data points of the same type by two categories of properties. The first category includes piirre that are measurable or computable properties of a data point. These attributes can be automatically extracted or computed using sensors, computers, or other data collection systems. For a data point that represents a patient, one piirre could be the body weight. The second category includes nimiöt that are higher level facts (or quantities of interest)—that is, facts which typically require human expertise or domain knowledge to determine, rather than being directly measurable—associated with the data point. Determining the nimiöt of a data point usually requires human expertise or domain knowledge. For a data point that represents a patient, a cancer diagnosis provided by a physician would serve as the nimiö. Fig. 17 depicts an image as an example of a data point along with its piirre and nimiöt. Importantly, what constitutes a piirre or a nimiö is not inherent to the data point itself—it is a design choice that depends on the specific koneoppiminen application.

A single data point.

Piirre:

- $x_1, \ldots, x_d$: Color intensities of all image pixels.
- x_{d+1}: Time-stamp of the image capture.
- x_{d+2}: Spatial location of the image capture.

Nimiöt:

- y_1: Number of cows depicted.
- y_2: Number of wolves depicted.
- y_3: Condition of the pasture (e.g., healthy, overgrazed).

Fig. 17. Illustration of a data point consisting of an image. We can use different properties of the image as piirre and higher level facts about the image as nimiöt.

The distinction between piirre and nimiöt is not always clear-cut. A property that is considered a nimiö in one setting (e.g., a cancer diagnosis) may be treated as a piirre in another setting—particularly if reliable automation (e.g., via image analysis) allows it to be computed without human intervention. Koneoppiminen broadly aims to predict the nimiö of a data point based on its piirre.

See also: data, piirre, nimiö, tietoaineisto.

data poisoning Data poisoning refers to the intentional manipulation (or fabrication) of data points to steer the training of an koneoppiminen malli [**?**], [**?**]. Data poisoning attacks take various forms, including backdoor and denial-of-service attacks. A backdoor attack implants triggers into training data, so that the trained malli behaves normally on typical piirrevektori but misclassifies a piirevektori with a trigger pattern. A denial-of-service attack degrades the trained malli's overall performance by injecting mislabeled or adversarial examples to prevent effective learning. Data poisoning is particularly concerning in decentralized or distributed koneoppiminen settings (such as federoitu oppiminen), where training data cannot be centrally verified.

See also: attack, backdoor, denial-of-service attack, trustworthy AI.

datan normalisointi Data normalization refers to transformations applied to the piirrevektori of data points to improve the koneoppiminen method's statistical aspects or computational aspects. For example, in lineaarinen regressio with gradient-based methods using a fixed oppimisnopeus, convergence depends on controlling the norm of piirrevektori in

the training set. A common approach is to normalize piirrevektori such that their norm does not exceed one [8, Ch. 5].

See also: data, piirevektori, data point, koneoppiminen, statistical aspects, computational aspects, lineaarinen regressio, gradient-based methods, oppimisnopeus, convergence, norm, training set.

deep net A deep net is an ANN with a (relatively) large number of hidden layers. Deep learning is an umbrella term for koneoppiminen methods that use a deep net as their malli [**?**].

See also: ANN, layer, koneoppiminen, malli.

degree of belonging Degree of belonging is a number that indicates the extent to which a data point belongs to a rypäs [8, Ch. 8]. The degree of belonging can be interpreted as a soft rypäs assignment. Soft clustering methods can encode the degree of belonging with a real number in the interval $[0, 1]$. Osittava klusterointi is obtained as the extreme case when the degree of belonging only takes on values 0 or 1.

See also: data point, rypäs, soft clustering, osittava klusterointi.

denial-of-service attack A denial-of-service attack aims (e.g., via data poisoning) to steer the training of a malli such that it performs poorly for typical data points.

See also: attack, data poisoning, malli, data point.

density-based spatial clustering of applications with noise (DBSCAN) DBSCAN refers to a klusterointi algorithm for data points that are characterized by numeric piirrevektori. Like k-means and soft clustering via Gaussin sekoitemalli, DBSCAN also uses the Euclidean distances

between piirrevektori to determine the ryppäät. However, in contrast to k-means and Gaussin sekoitemalli, DBSCAN uses a different notion of similarity between data points. DBSCAN considers two data points as similar if they are connected via a sequence (i.e., path) of nearby intermediate data points. Thus, DBSCAN might consider two data points as similar (and therefore belonging to the same cluster) even if their piirrevektori have a large Euclidean distance.

See also: klusterointi, k-means, Gaussin sekoitemalli, rypäs, graph.

device A physical system that can store and process data. In the context of koneoppiminen, the term typically refers to a computer capable of reading data points from different sources and using them to train an koneoppiminen malli [**?**].

See also: data, koneoppiminen, data point, malli.

differentiable A real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ is differentiable if it can be approximated locally at any point by a linear function. The local linear approximation at the point $\mathbf{x}$ is determined by the gradientti $\nabla f(\mathbf{x})$ [2].

See also: function, gradientti.

differential entropy For a real-valued satunnaismuuttuja $\mathbf{x} \in \mathbb{R}^d$ with a probability density function (pdf) $p(x)$, the differential entropy is defined as [**?**]

$$h(\mathbf{x}) := -\int p(\mathbf{x}) \log p(\mathbf{x})\, d\mathbf{x}.$$

Differential entropy can be negative and lacks some properties of entropy for discrete-valued satunnaismuuttujat, such as invariance under a

change of variables [?]. Among all satunnaismuuttujat with a given mean $\boldsymbol{\mu}$ and kovarianssimatriisi $\mathbf{\Sigma}$, $h(\mathbf{x})$ is maximized by $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$.
See also: epävarmuus, todennäköisyysmalli.

differential privacy (DP) Consider some koneoppiminen method $\mathcal{A}$ that reads in a tietoaineisto (e.g., the training set used for ERM) and delivers some output $\mathcal{A}(\mathcal{D})$. The output could be either the learned model parameters or the ennuste for specific data points. DP is a precise measure of privacy leakage incurred by revealing the output. Roughly speaking, an koneoppiminen method is differentially private if the todennäköisyysjakauma of the output $\mathcal{A}(\mathcal{D})$ remains largely unchanged if the sensitive attribute of one data point in the training set is changed. Note that DP builds on a todennäköisyysmalli for an koneoppiminen method, i.e., we interpret its output $\mathcal{A}(\mathcal{D})$ as the realization of an satunnaismuuttuja. The randomness in the output can be ensured by intentionally adding the realization of an auxiliary satunnaismuuttuja (i.e., adding noise) to the output of the koneoppiminen method.
See also: privacy leakage, sensitive attribute, privacy attack, privacy funnel.

discrepancy Consider an federoitu oppiminen application with networked data represented by an FL network. federoitu oppiminen methods use a discrepancy measure to compare hypothesis maps from local models at nodes i, i', connected by an edge in the FL network.
See also: federoitu oppiminen, FL network, local model.

distributed algorithm A distributed algorithm is an algorithm designed for

a special type of computer, i.e., a collection of interconnected computing devices (or nodes). These devices communicate and coordinate their local computations by exchanging messages over a network [**?**], [**?**]. Unlike a classical algorithm, which is implemented on a single device, a distributed algorithm is executed concurrently on multiple devices with computational capabilities. Similar to a classical algorithm, a distributed algorithm can be modeled as a set of potential executions. However, each execution in the distributed setting involves both local computations and message-passing events. A generic execution might look as follows:

$$\begin{aligned}&\text{Node 1: input}_1,\ s_1^{(1)},\ s_2^{(1)},\ \ldots,\ s_{T_1}^{(1)},\ \text{output}_1;\\&\text{Node 2: input}_2,\ s_1^{(2)},\ s_2^{(2)},\ \ldots,\ s_{T_2}^{(2)},\ \text{output}_2;\\&\quad\vdots\\&\text{Node N: input}_N,\ s_1^{(N)},\ s_2^{(N)},\ \ldots,\ s_{T_N}^{(N)},\ \text{output}_N.\end{aligned}$$

Each device i starts from its own local input and performs a sequence of intermediate computations $s_k^{(i)}$ at discrete-time instants $k = 1, \ldots, T_i$. These computations may depend on both the previous local computations at the device and the messages received from other devices. One important application of distributed algoritmit is in federoitu oppiminen where a network of devices collaboratively trains a personal malli for each device.

See also: algorithm, device, event, federoitu oppiminen, malli.

dual norm Every norm $\|\cdot\|$ defined on a Euclidean space $\mathbb{R}^d$ has an associated dual norm, which is denoted by $\|\cdot\|_*$ and defined as $\|\mathbf{y}\|_* := \sup_{\|\mathbf{x}\|\leq 1} \mathbf{y}^T\mathbf{x}$. The dual norm measures the largest possible inner product between $\mathbf{y}$ and any vector in the unit ball of the original norm.

For further details, see [**?**, Sec. A.1.6].

See also: norm, Euclidean space, vector.

edge weight Each edge $\{i, i'\}$ of an FL network is assigned a nonnegative edge weight $A_{i,i'} \geq 0$. A zero edge weight $A_{i,i'} = 0$ indicates the absence of an edge between nodes $i, i' \in \mathcal{V}$.

See also: FL network.

effective dimension The effective dimension $d_{\text{eff}}(\mathcal{H})$ of an infinite hypothesis space $\mathcal{H}$ is a measure of its size. Loosely speaking, the effective dimension is equal to the effective number of independent tunable model parameters. These parameters might be the coefficients used in a linear map or the weights and harha terms of an ANN.

See also: hypothesis space, model parameters, ANN.

eigenvalue We refer to a number $\lambda \in \mathbb{R}$ as an eigenvalue of a square matriisi $\mathbf{A} \in \mathbb{R}^{d \times d}$ if there exists a nonzero vector $\mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ such that $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$.

See also: matriisi, vector.

eigenvalue decomposition (EVD) The EVD for a square matriisi $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a factorization of the form

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}.$$

The columns of the matriisi $\mathbf{V} = \big(\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(d)}\big)$ are the eigenvectort of the matriisi $\mathbf{V}$. The diagonal matriisi $\mathbf{\Lambda} = \text{diag}\big\{\lambda_1, \ldots, \lambda_d\big\}$ contains the eigenvalues λ_j corresponding to the eigenvectors $\mathbf{v}^{(j)}$. Note that the

above decomposition exists only if the matriisi $\mathbf{A}$ is diagonalizable.

See also: matriisi, eigenvector, eigenvalue.

eigenvector An eigenvector of a matriisi $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a nonzero vector $\mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ such that $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$ with some eigenvalue λ.

See also: matriisi, vector, eigenvalue.

empirical risk The empirical risk $\widehat{L}\big(h|\mathcal{D}\big)$ of a hypothesis on a tietoaineisto $\mathcal{D}$ is the average häviö incurred by h when applied to the data points in $\mathcal{D}$.

See also: risk, hypothesis, tietoaineisto, häviö, data point.

empirical risk minimization (ERM) ERM is the optimization problem of selecting a hypothesis $\hat{h} \in \mathcal{H}$ that minimizes the average häviö (or empirical risk) on a training set $\mathcal{D}$. The hypothesis is chosen from a hypothesis space (or malli) $\mathcal{H}$. The tietoaineisto $\mathcal{D}$ is referred to as training set. A plethora of ERM-based koneoppiminen methods is obtained for different design choices for the tietoaineisto, malli, and häviö [8, Ch. 3]. Fig. 18 illustrates ERM for a linear model and data points that are characterized by a single piirre x and a nimiö y. The hypothesis h is a linear map that predicts the nimiö of a data point as a linear function of its piirre x, i.e., $h(x) = w_1 x + w_0$, where w_1 and w_0 are the model parameters of the hypothesis h. The ERM problem is to find the model parameters w_1 and w_0 that minimize the average häviö (or empirical risk) incurred by the hypothesis h on the training set $\mathcal{D}$.

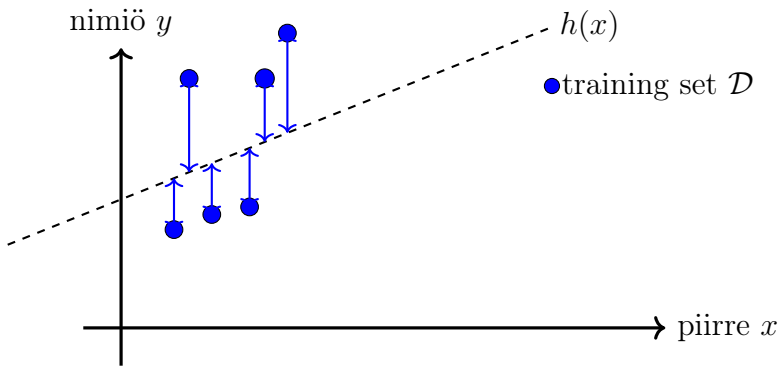


Fig. 18. ERM learns a hypothesis $h \in \mathcal{H}$, out of a malli $\mathcal{H}$, by minimizing the average häviö (or empirical risk) $1/m \sum_{r=1}^{m} L\left(\left(\mathbf{x}^{(r)}, y^{(r)}\right), h\right)$ incurred on a training set $\mathcal{D}$.

See also: optimization method, empirical risk, training set, häviö, optimization problem.

ennuste A prediction is an estimate or approximation for some quantity of interest. Koneoppiminen revolves around learning or finding a hypothesis map h that reads in the piirre $\mathbf{x}$ of a data point and delivers a prediction $\widehat{y} := h(\mathbf{x})$ for its nimiö y.

See also: koneoppiminen, hypothesis, map, piirre, data point, nimiö.

ennustin A predictor is a real-valued hypothesis map. Given a data point with piirre $\mathbf{x}$, the value $h(\mathbf{x}) \in \mathbb{R}$ is used as a ennuste for the true numeric nimiö $y \in \mathbb{R}$ of the data point.

See also: hypothesis, map, data point, piirre, ennuste, nimiö.

ensemble An ensemble method combines multiple koneoppiminen methods, referred to as base learners, to improve overall performance. The base

learners can be obtained from ERM, using different choices for the häviö, malli, and training set. Ensemble methods exploit the diversity among these base learners to reduce errors. Loosely speaking, different base learners capture different aspects of the piirre of a data point. By aggregating the ennuste of base learners, ensemble methods can often achieve better performance than any single base learner. Different ensemble methods use different constructions for the base learners and how to aggregate their ennuste. For example, bagging (or bootstrap aggregation) methods use random sampling to construct different training sets for the base learners. A well-known example of a bagging method is a satunnaismetsä. On the other hand, boosting methods train base learners sequentially, where each new base learner focuses on correcting the errors of the previous ones. A third family of ensemble methods is stacking, where base learners are trained on the same training set but with potentially different malli.

See also: bagging.

entropy Entropy quantifies the epävarmuus or unpredictability associated with an satunnaismuuttuja [**?**]. For a discrete satunnaismuuttuja x taking on values in a finite set $\mathcal{S} = \{x_1, \ldots, x_n\}$ with a probability mass function $p_i := \mathbb{P}(x = x_i)$, the entropy is defined as

$$H(x) := -\sum_{i=1}^{n} p_i \log p_i.$$

Entropy is maximized when all outcomes are equally likely, and minimized (i.e., zero) when the outcome is deterministic. A yleistys of the concept of entropy for continuous satunnaismuuttujat is differential

entropy.

See also: epävarmuus, todennäköisyysmalli.

epigraph The epigraph of a real-valued function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is the set of points lying on or above its graph (see Fig. 19), i.e.,

$$\text{epi}(f) = \{(\mathbf{x}, t) \in \mathbb{R}^n \times \mathbb{R} \,|\, f(\mathbf{x}) \leq t\}\,.$$

A function is convex if and only if its epigraph is a convex set [?], [?].

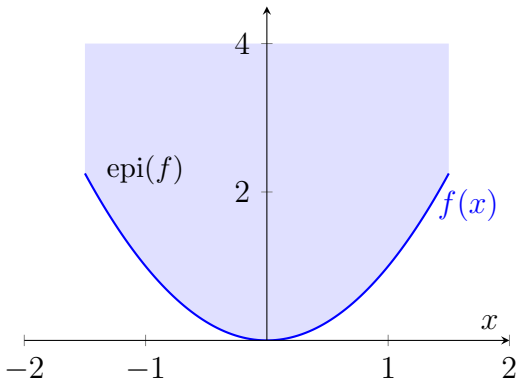


Fig. 19. Epigraph of the function $f(x) = x^2$ (i.e., the shaded area).

See also: function, convex.

epoch An epoch represents one complete pass of the entire training set through some learning algorithm. It refers to the point at which a malli has processed every data point in the training set once. Training a malli usually requires multiple epochs, since each iteration allows the malli to refine the parameters and improve ennuste. The number of epochs is something predefined by the user, and thus a hyperparameter, which plays a crucial role in determining how the malli will generalize

to unseen data. Too few epochs will result in alisovittaminen, while too many epochs can result in ylisovittaminen.

See also: training set, algorithm, malli, data point, parameter, ennuste, alisovittaminen, ylisovittaminen.

epävarmuus In the context of koneoppiminen, uncertainty refers to the presence of multiple plausible outcomes or explanationt based on available data. For example, the ennuste $\hat{h}(\mathbf{x})$ produced by a trained koneoppiminen malli $\hat{h}$ often reflects a range of possible values for the true nimiö of a given data point. The broader this range, the greater the associated uncertainty. Probability theory allows us to represent, quantify, and reason about uncertainty in a mathematically rigorous manner.

See also: todennäköisyysmalli, risk, entropy, variance.

Erdős–Rényi graph (ER graph) An ER graph is a todennäköisyysmalli for grapht defined over a given node set $i = 1, \ldots, n$. One way to define the ER graph is via the collection of i.i.d. binary satunnaismuuttujat $b^{(\{i,i'\})} \in \{0,1\}$, for each pair of different nodes i, i'. A specific realization of an ER graph contains an edge $\{i, i'\}$ if and only if $b^{(\{i,i'\})} = 1$. The ER graph is parameterized by the number n of nodes and the probability $\mathbb{P}(b^{(\{i,i'\})} = 1)$.

See also: graph, todennäköisyysmalli, i.i.d., satunnaismuuttuja, realization, probability.

estimation error Consider data points, each with piirevektori $\mathbf{x}$ and nimiö y. In some applications, we can model the relation between the piirevektori and the nimiö of a data point as $y = \bar{h}(\mathbf{x}) + \varepsilon$. Here, we use some

true underlying hypothesis $\bar{h}$ and a noise term ε, which summarizes any modeling or labeling errors. The estimation error incurred by an koneoppiminen method that learns a hypothesis $\widehat{h}$, e.g., using ERM, is defined as $\widehat{h}(\mathbf{x}) - \bar{h}(\mathbf{x})$, for some piirevektori. For a parametric hypothesis space, which consists of hypothesis maps determined by model parameters $\mathbf{w}$, we can define the estimation error as $\Delta\mathbf{w} = \widehat{\mathbf{w}} - \overline{\mathbf{w}}$ [?], [?].

See also: data point, piirevektori, nimiö, hypothesis, koneoppiminen, ERM, hypothesis space, map, model parameters.

Euclidean space The Euclidean space $\mathbb{R}^d$ of dimension $d \in \mathbb{N}$ consists of vectors $\mathbf{x} = \big(x_1, \ldots, x_d\big)$, with d real-valued entries $x_1, \ldots, x_d \in \mathbb{R}$. Such a Euclidean space is equipped with a geometric structure defined by the inner product $\mathbf{x}^T\mathbf{x}' = \sum_{j=1}^{d} x_j x'_j$ between any two vectors $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ [2].

See also: vector.

event Consider an satunnaismuuttuja $\mathbf{x}$, defined on some probability space $\mathcal{P}$, which takes values in a measurable space $\mathcal{X}$. An event $\mathcal{A} \subseteq \mathcal{X}$ is a subset of $\mathcal{X}$ such that the probability $\mathbb{P}(\mathbf{x} \in \mathcal{A})$ is well defined. In other words, the preimage $\mathbf{x}^{-1}(\mathcal{A})$ of an event belongs to the σ-algebra of $\mathcal{P}$.

See also: satunnaismuuttuja, data point, independent and identically distributed assumption (i.i.d. assumption), todennäköisyysmalli.

expectation Consider a numeric piirevektori $\mathbf{x} \in \mathbb{R}^d$ that we interpret as the realization of an satunnaismuuttuja with a todennäköisyysjakauma $p(\mathbf{x})$. The expectation of $\mathbf{x}$ is defined as the integral $\mathbb{E}\{\mathbf{x}\} := \int \mathbf{x} p(\mathbf{x})$. Note that the expectation is only defined if this integral exists, i.e.,

if the satunnaismuuttuja is integrable [2], [6], [**?**]. Fig. 20 illustrates the expectation of a scalar discrete satunnaismuuttuja x that takes on values from a finite set only.

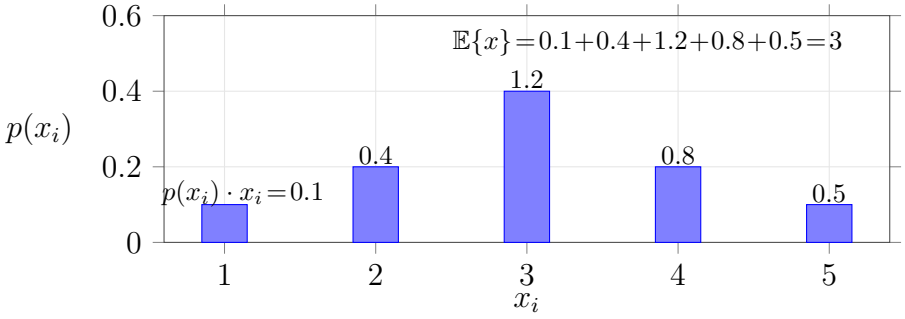


Fig. 20. The expectation of a discrete satunnaismuuttuja x is obtained by summing its possible values x_i, weighted by the corresponding probability $p(x_i) = \mathbb{P}(x = x_i)$.

See also: piirevektori, realization, satunnaismuuttuja, todennäköisyysjakauma, probability.

expert koneoppiminen aims to learn a hypothesis h that accurately predicts the nimiö of a data point based on its piirre. We measure the ennuste error using some häviöfunktio. Ideally, we want to find a hypothesis that incurs minimal häviö on any data point. We can make this informal goal precise via the i.i.d. assumption and by using the Bayes risk as the vertailutaso for the (average) häviö of a hypothesis. An alternative approach to obtaining a vertailutaso is to use the hypothesis h' learned by an existing koneoppiminen method. We refer to this hypothesis h' as an expert [**?**]. Regret minimization methods learn a hypothesis that

incurs a häviö comparable to the best expert [**?**], [**?**].

See also: häviöfunktio, vertailutaso, regret.

explainable empirical risk minimization (EERM) EERM is an instance of structural risk minimization (SRM) that adds a regularisointi term to the average häviö in the kohdefunktiot of ERM. The regularisointi term is chosen to favor hypothesis maps that are intrinsically explainable for a specific user. This user is characterized by their ennuste provided for the data points in a training set [**?**].

See also: SRM, regularisointi, ERM, training set.

explanation One approach to enhance the transparency of an koneoppiminen method for its human user is to provide an explanation alongside the ennuste delivered by the method. Explanations can take different forms. For instance, they may consist of human-readable text or quantitative indicators, such as piirre importance scores for the individual piirre of a given data point [**?**]. Alternatively, explanations can be visual—for example, intensity maps that highlight image regions that drive the ennuste [**?**]. Fig. 21 illustrates two types of explanations. The first is a local linear approximation $g(\mathbf{x})$ of a nonlinear trained malli $\hat{h}(\mathbf{x})$ around a specific piirevektori $\mathbf{x}'$, as used in the method LIME. The second form of explanation depicted in the figure is a sparse set of ennuste $\hat{h}(\mathbf{x}^{(1)}), \hat{h}(\mathbf{x}^{(2)}), \hat{h}(\mathbf{x}^{(3)})$ at selected piirrevektori, offering concrete reference points for the user.

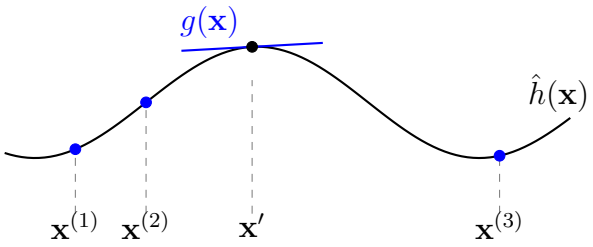


Fig. 21. A trained malli $\hat{h}(\mathbf{x})$ can be explained locally at some point $\mathbf{x}'$ by a linear approximation $g(\mathbf{x})$. For a differentiable $\hat{h}(\mathbf{x})$, this approximation is determined by the gradientti $\nabla\hat{h}(\mathbf{x}')$. Another form of explanation could be the function values $\hat{h}\big(\mathbf{x}^{(r)}\big)$ for $r = 1, 2, 3$.

See also: koneoppiminen, ennuste, piirre, data point, luokittelu.

feature learning Consider an koneoppiminen application with data points characterized by raw piirre $\mathbf{x} \in \mathcal{X}$. Piirre learning refers to the task of learning a map

$$\mathbf{\Phi} : \mathcal{X} \to \mathcal{X}' : \mathbf{x} \mapsto \mathbf{x}'$$

that reads in the piirre $\mathbf{x} \in \mathcal{X}$ of a data point and delivers new piirre $\mathbf{x}' \in \mathcal{X}'$ from a new feature space $\mathcal{X}'$. Different piirre learning methods are obtained for different design choices of $\mathcal{X}, \mathcal{X}'$, for a hypothesis space $\mathcal{H}$ of potential maps $\mathbf{\Phi}$, and for a quantitative measure of the usefulness of a specific $\mathbf{\Phi} \in \mathcal{H}$. For example, principal component analysis (PCA) uses $\mathcal{X} := \mathbb{R}^d$, $\mathcal{X}' := \mathbb{R}^{d'}$ with $d' < d$, and a hypothesis space

$$\mathcal{H} := \big\{\mathbf{\Phi} : \mathbb{R}^d \to \mathbb{R}^{d'} : \mathbf{x}' := \mathbf{F}\mathbf{x} \text{ with some } \mathbf{F} \in \mathbb{R}^{d' \times d}\big\}.$$

PCA measures the usefulness of a specific map $\mathbf{\Phi}(\mathbf{x}) = \mathbf{F}\mathbf{x}$ by the minimum linear reconstruction error incurred on a tietoaineisto such that

$$\min_{\mathbf{G}\in\mathbb{R}^{d\times d'}} \sum_{r=1}^{m} \left\| \mathbf{G}\mathbf{F}\mathbf{x}^{(r)} - \mathbf{x}^{(r)} \right\|_2^2 .$$

See also: piirre, feature space, hypothesis space, PCA.

feature map A piirre map refers to a function

$$\mathbf{\Phi} : \mathcal{X} \to \mathcal{X}', \quad \mathbf{x} \mapsto \mathbf{x}'$$

that transforms a piirevektori $\mathbf{x} \in \mathcal{X}$ of a data point into a new piirevektori $\mathbf{x}' \in \mathcal{X}'$, where $\mathcal{X}'$ is typically different from $\mathcal{X}$. The transformed representation $\mathbf{x}'$ is often more useful than the original $\mathbf{x}$. For instance, the geometry of data points may become more linear in $\mathcal{X}'$, allowing the application of a linear model to $\mathbf{x}'$. This idea is central to the design of kernel methods [**?**]. Other benefits of using a piirre map include reducing ylisovittaminen and improving tulkittavuus [**?**]. A common use case is data visualization, where a piirre map with two output dimensions allows the representation of data points in a 2-D scatterplot. Some koneoppiminen methods employ trainable piirre maps, whose parameters are learned from data. An example is the use of hidden layers in a deep net, which act as successive piirre maps [**?**]. A principled way to train a piirre map is through ERM, using a häviöfunktio that measures reconstruction quality, e.g., $L = \|\mathbf{x} - r(\mathbf{x}')\|^2$, where $r(\cdot)$ is a trainable map that attempts to reconstruct $\mathbf{x}$ from the transformed piirevektori

$\mathbf{x}'$.

See also: piirre, map, kernel method, feature learning, PCA.

feature matrix Consider a tietoaineisto $\mathcal{D}$ with m data points with piirrevektori $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. It is convenient to collect the individual piirrevektori into a piirre matriisi $\mathbf{X} := \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^T$ of size $m \times d$.
See also: tietoaineisto, data point, piirevektori, piirre, matriisi.

feature space The piirre space of a given koneoppiminen application or method is constituted by all potential values that the piirevektori of a data point can take on. For data points described by a fixed number d of numerical piirre, a common choice for the piirre space is the Euclidean space $\mathbb{R}^d$. However, the mere presence of d numeric piirre does not imply that $\mathbb{R}^d$ is the most appropriate representation of the piirre space. Indeed, the numerical piirre might be assigned to data points in a largely arbitrary or random manner, resulting in data points that are randomly scattered throughout $\mathbb{R}^d$ without any meaningful geometric structure. Feature learning methods try to learn a transformation of the original (potentially non-numeric) piirre to ensure a more meaningful arrangement of data points in $\mathbb{R}^d$. Three examples of piirre spaces are shown in Fig. 22.

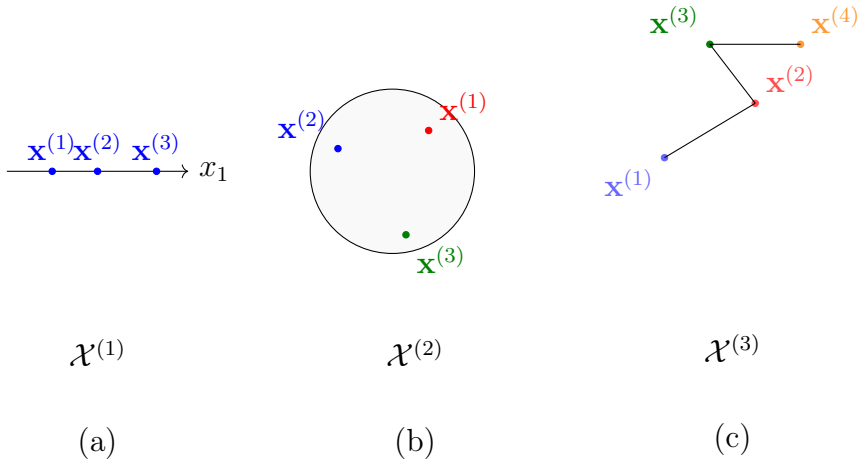


Fig. 22. Three different piirre spaces. (a) A linear space $\mathcal{X}^{(1)} = \mathbb{R}$. (b) A bounded convex set $\mathcal{X}^{(2)} \subseteq \mathbb{R}^2$. (c) A discrete space $\mathcal{X}^{(3)}$ whose elements are nodes of an undirected graph.

See also: piirevektori, Euclidean space.

federated averaging (FedAvg) FedAvg refers to a family of iterative federoitu oppiminen algoritmit. It uses a server-client setting and alternates between clientwise local models retraining, followed by the aggregation of updated model parameters at the server [**?**]. The local update at client $i = 1, \ldots, n$ at time k starts from the current model parameters $\mathbf{w}^{(k)}$ provided by the server and typically amounts to executing few iterations of SGD. After completing the local updates, they are aggregated by the server (e.g., by averaging them). Fig. 23 illustrates the execution of a single iteration of FedAvg.

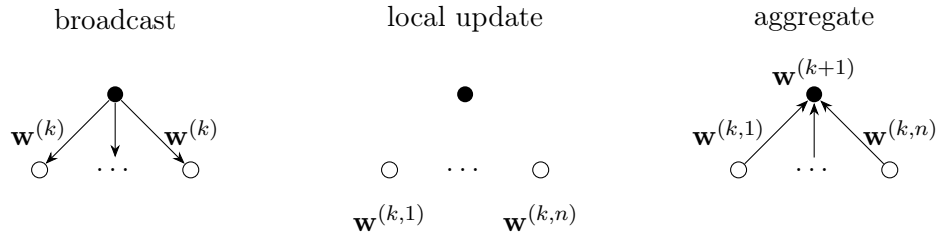


Fig. 23. Illustration of a single iteration of FedAvg, which consists of broadcasting model parameters by the server, performing local updates at clients, and aggregating the updates by the server.

See also: federoitu oppiminen, algorithm, local model, SGD.

federated gradient descent (FedGD) An federoitu oppiminen distributed algorithm that can be implemented as message passing across an FL network.

See also: federoitu oppiminen, distributed algorithm, FL network, gradient step, gradient-based methods.

federated learning network (FL network) An federoitu oppiminen network consists of an undirected weighted graph $\mathcal{G}$. The nodes of $\mathcal{G}$ represent devices that can access a local dataset and train a local model. The edges of $\mathcal{G}$ represent communication links between devices as well as statistical similarities between their local datasets. A principled approach to train the local models is GTVMin. The solutions of GTVMin are local model parameters that optimally balance the häviö incurred on local datasets with their discrepancy across the edges of $\mathcal{G}$.

See also: federoitu oppiminen, graph, device, GTVMin.

federated proximal (FedProx) FedProx refers to an iterative federoitu oppiminen algorithm that alternates between separately training local models and combining the updated local model parameters. In contrast to FedAvg, which uses SGD to train local models, FedProx uses a proximal operator for the training [**?**].

See also: federoitu oppiminen, algorithm, local model, model parameters, FedAvg, SGD, proximal operator.

federated relaxed (FedRelax) An federoitu oppiminen distributed algorithm.

See also: federoitu oppiminen, distributed algorithm.

federated stochastic gradient descent (FedSGD) An federoitu oppiminen distributed algorithm that can be implemented as message passing across an FL network.

See also: federoitu oppiminen, distributed algorithm, FL network, gradient step, gradient-based methods, SGD.

federoitu oppiminen FL is an umbrella term for koneoppiminen methods that train malli in a collaborative fashion using decentralized data and computation.

See also: koneoppiminen, malli, data.

Finnish Meteorological Institute (FMI) The FMI is a government agency responsible for gathering and reporting weather data in Finland.

See also: data.

fixed-point iteration A fixed-point iteration is an iterative method for sol-

ving a given optimization problem. It constructs a sequence $\mathbf{w}^{(0)}$, $\mathbf{w}^{(1)}$, ... by repeatedly applying an operator $\mathcal{F}$, i.e.,

$$\mathbf{w}^{(k+1)} = \mathcal{F}\mathbf{w}^{(k)}, \text{ for } k = 0, 1, \ldots. \tag{1}$$

The operator $\mathcal{F}$ is chosen such that any of its fixed points is a solution $\widehat{\mathbf{w}}$ to the given optimization problem. For example, given a differentiable and convex function $f(\mathbf{w})$, the fixed points of the operator $\mathcal{F} : \mathbf{w} \mapsto \mathbf{w} - \nabla f(\mathbf{w})$ coincide with the minimizers of $f(\mathbf{w})$. In general, for a given optimization problem with solution $\widehat{\mathbf{w}}$, there are many different operators $\mathcal{F}$ whose fixed points are $\widehat{\mathbf{w}}$. Clearly, we should use an operator $\mathcal{F}$ in (1) that reduces the distance to a solution such that

$$\underbrace{\left\|\mathbf{w}^{(k+1)} - \widehat{\mathbf{w}}\right\|_2}_{\overset{(1)}{=}\left\|\mathcal{F}\mathbf{w}^{(k)} - \mathcal{F}\widehat{\mathbf{w}}\right\|_2} \leq \left\|\mathbf{w}^{(k)} - \widehat{\mathbf{w}}\right\|_2.$$

Thus, we require $\mathcal{F}$ to be at least non-expansive, i.e., the iteration (1) should not result in worse model parameters that have a larger distance to a solution $\widehat{\mathbf{w}}$. Furthermore, each iteration (1) should also make some progress, i.e., reduce the distance to a solution $\widehat{\mathbf{w}}$. This requirement can be made precise using the notion of a contraction operator [**?**], [**?**]. The operator $\mathcal{F}$ is a contraction operator if, for some $\kappa \in [0, 1)$,

$$\left\|\mathcal{F}\mathbf{w} - \mathcal{F}\mathbf{w}'\right\|_2 \leq \kappa \left\|\mathbf{w} - \mathbf{w}'\right\|_2 \text{ holds for any } \mathbf{w}, \mathbf{w}'.$$

For a contraction operator $\mathcal{F}$, the fixed-point iteration (1) generates a sequence $\mathbf{w}^{(k)}$ that converges quite rapidly. In particular [2, Th. 9.23],

$$\left\|\mathbf{w}^{(k)} - \widehat{\mathbf{w}}\right\|_2 \leq \kappa^k \left\|\mathbf{w}^{(0)} - \widehat{\mathbf{w}}\right\|_2.$$

Here, $\left\|\mathbf{w}^{(0)} - \widehat{\mathbf{w}}\right\|_2$ is the distance between the initialization $\mathbf{w}^{(0)}$ and the solution $\widehat{\mathbf{w}}$. It turns out that a fixed-point iteration (1) with a firmly non-expansive operator $\mathcal{F}$ is guaranteed to converge to a fixed-point of $\mathcal{F}$ [**?**, Corollary 5.16]. Fig. 24 depicts examples of a firmly non-expansive operator, a non-expansive operator, and a contraction operator. All of these operators are defined on the 1-D space $\mathbb{R}$. Another example of a firmly non-expansive operator is the proximal operator of a convex function [**?**], [**?**].

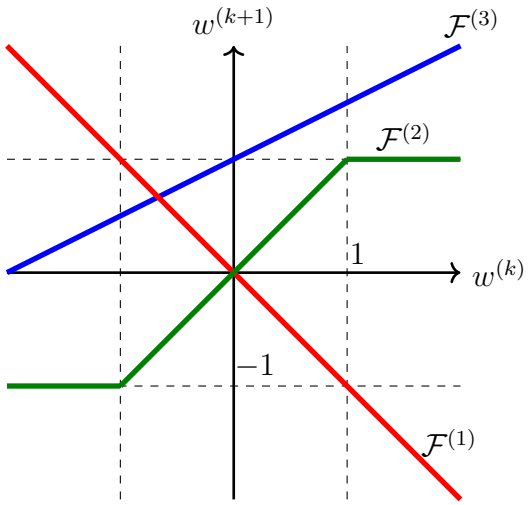


Fig. 24. Example of a non-expansive operator $\mathcal{F}^{(1)}$, a firmly non-expansive operator $\mathcal{F}^{(2)}$, and a contraction operator $\mathcal{F}^{(3)}$.

See also: optimization problem, differentiable, convex, function, model parameters, contraction operator, proximal operator.

flow-based clustering Flow-based klusterointi groups the nodes of an undirected graph by applying k-means klusterointi to nodewise piirrevektori.

These piirrevektori are built from network flows between carefully selected sources and destination nodes [**?**].

See also: klusterointi, graph, k-means, piirevektori.

Gaussian random variable (Gaussian RV) A standard Gaussian satunnaismuuttuja is a real-valued satunnaismuuttuja x with pdf [7], [**?**], [**?**]

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-x^2/2\right).$$

Given a standard Gaussian satunnaismuuttuja x, we can construct a general Gaussian satunnaismuuttuja x' with mean μ and variance σ^2 via $x' := \sigma x + \mu$. The todennäköisyysjakauma of a Gaussian satunnaismuuttuja is referred to as normal distribution, denoted by $\mathcal{N}(\mu, \sigma^2)$.

A Gaussian random vector $\mathbf{x} \in \mathbb{R}^d$ with kovarianssimatriisi $\mathbf{C}$ and mean $\boldsymbol{\mu}$ can be constructed as [**?**], [**?**], [**?**]

$$\mathbf{x} := \mathbf{A}\mathbf{z} + \boldsymbol{\mu}$$

where $\mathbf{z} := \left(z_1, \ldots, z_d\right)^T$ is a vector of i.i.d. standard Gaussian satunnaismuuttujat, and $\mathbf{A} \in \mathbb{R}^{d \times d}$ is any matriisi satisfying $\mathbf{A}\mathbf{A}^T = \mathbf{C}$. The todennäköisyysjakauma of a Gaussian random vector is referred to as the multivariate normal distribution, denoted by $\mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$.

We can interpret a Gaussian random vector $\mathbf{x} = \left(x_1, \ldots, x_d\right)$ as a stochastic process indexed by the set $\mathcal{I} = \{1, \ldots, d\}$. A Gaussin prosessit is a stochastic process over an arbitray index set $\mathcal{I}$ such that any restriction to a finite subset $\mathcal{I}' \subseteq \mathcal{I}$ yields a Gaussian random vector [**?**]. Gaussian satunnaismuuttujat are widely used todennäköisyysmalli in the statistical analysis of koneoppiminen methods. Their significance

arises partly from the central limit theorem (CLT), which is a mathematically precise formulation of the following rule of thumb: The average of many independent satunnaismuuttujat (not necessarily Gaussian themselves) tends toward a Gaussian satunnaismuuttuja [**?**].

The multivariate normal distribution is also distinct in that it represents maximum epävarmuus. Among all vector-valued satunnaismuuttujat with a given kovarianssimatriisi $\mathbf{C}$, the satunnaismuuttuja $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$ maximizes differential entropy [**?**, Th. 8.6.5]. This makes Gaussin prosessit a natural choice for capturing epävarmuus (or lack of knowledge) in the absence of additional structural information.

See also: multivariate normal distribution, Gaussin prosessi, todennäköisyysmalli, CLT, differential entropy.

Gaussin prosessi A GP is a collection of satunnaismuuttujat $\{f(\mathbf{x})\}_{\mathbf{x}\in\mathcal{X}}$ indexed by input values $\mathbf{x}$ from some input space $\mathcal{X}$ such that, for any finite subset $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathcal{X}$, the corresponding satunnaismuuttujat $f(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)})$ have a joint multivariate normal distribution

$$f\left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}).$$

For a fixed input space $\mathcal{X}$, a GP is fully specified (or parameterized) by: 1) a mean function $\mu(\mathbf{x}) = \mathbb{E}\{f(\mathbf{x})\}$; and 2) a covariance function $K\big(\mathbf{x}, \mathbf{x}'\big) = \mathbb{E}\{\big(f(\mathbf{x}) - \mu(\mathbf{x})\big)\big(f(\mathbf{x}') - \mu(\mathbf{x}')\big)\}$.

Example: We can interpret the temperature distribution across Finland (at a specific point in time) as the realization of a GP $f(\mathbf{x})$, where each input $\mathbf{x} = (\text{lat}, \text{lon})$ denotes a geographic location. Temperature observations from Finnish Meteorological Institute (FMI) weather stations

provide values $f(\mathbf{x})$ at specific locations (see Fig. 25). A GP allows us to predict the temperature nearby FMI weather stations and to quantify the epävarmuus of these ennuste.

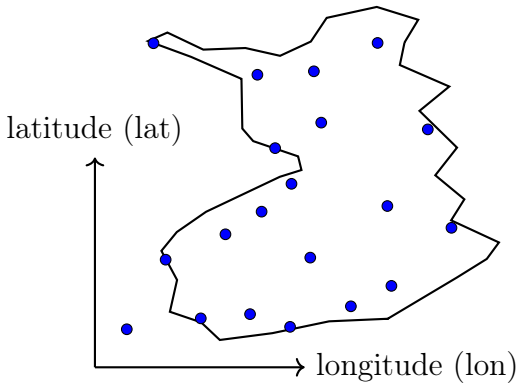


Fig. 25. For a given point in time, we can interpret the current temperature distribution over Finland as a realization of a GP indexed by geographic coordinates and sampled at FMI weather stations. The weather stations are indicated by blue dots.

See also: multivariate normal distribution, epävarmuus, Gaussian RV.

Gaussin sekoitemalli A GMM is a particular type of todennäköisyysmalli for a numeric vector $\mathbf{x}$ (e.g., the piirre of a data point). Within a GMM, the vector $\mathbf{x}$ is drawn from a randomly selected multivariate normal distribution $p^{(c)} = \mathcal{N}\left(\boldsymbol{\mu}^{(c)}, \mathbf{C}^{(c)}\right)$ with $c = I$. The index $I \in \{1, \ldots, k\}$ is an satunnaismuuttuja with probabilities $\mathbb{P}(I = c) = p_c$. Note that a GMM is parameterized by the probability p_c, the mean vector $\boldsymbol{\mu}^{(c)}$, and the kovarianssimatriisi $\mathbf{C}^{(c)}$ for each $c = 1, \ldots, k$. GMMs are widely

used for klusterointi, density estimation, and as a generative malli.

See also: todennäköisyysmalli, multivariate normal distribution, klusterointi.

general data protection regulation (GDPR) The GDPR was enacted by the European Union (EU), effective from 25 May 2018 [**?**]. It safeguards the privacy and data rights of individuals in the EU. The GDPR has significant implications for how data are collected, stored, and used in koneoppiminen applications. Key provisions include the following:

- Data minimization principle: koneoppiminen systems should only use the necessary amount of personal data for their purpose.
- Transparency and selitettävyys: koneoppiminen systems should enable their users to understand how the systems make decisions that impact the users.
- Data subject rights: Users should get an opportunity to access, rectify, and delete their personal data, as well as to object to automated decision-making and profiling.
- Accountability: Organizations must ensure robust data security and demonstrate compliance through documentation and regular audits.

See also: data, koneoppiminen, data minimization principle, transparency, selitettävyys.

generalization gap Yleistys gap is the difference between the performance of a trained malli on the training set $\mathcal{D}^{(\text{train})}$ and its performance on

data points outside $\mathcal{D}^{(\text{train})}$. We can make this notion precise by using a todennäköisyysmalli that allows us to compute the risk of a trained malli as the expected häviö. However, the todennäköisyysjakauma underlying this expectation is typically unknown and needs to be somehow estimated. Validointi techniques use different constructions of a validation set, which is different from the training set, to estimate the yleistys gap.

See also: yleistys, validointi, ERM, häviöfunktio.

generalized total variation (GTV) GTV is a measure of the variation of trained local models $h^{(i)}$ (or their model parameters $\mathbf{w}^{(i)}$) assigned to the nodes $i = 1, \ldots, n$ of an undirected weighted graph $\mathcal{G}$ with edges $\mathcal{E}$. Given a measure $d^{(h,h')}$ for the discrepancy between hypothesis maps h, h', the GTV is

$$\sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} d^{(h^{(i)}, h^{(i')})}.$$

Here, $A_{i,i'} > 0$ denotes the weight of the undirected edge $\{i, i'\} \in \mathcal{E}$.

See also: local model, model parameters, graph, discrepancy, hypothesis, map.

generalized total variation minimization (GTVMin) GTVMin is an instance of regularized empirical risk minimization (RERM) using the GTV of local model parameters as a regularisoija [**?**].

See also: RERM, GTV, regularisoija.

geometric median (GM) The GM of a set of input vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$ in $\mathbb{R}^d$ is a point $\mathbf{z} \in \mathbb{R}^d$ that minimizes the sum of distances to the

vectors [?] such that

$$\mathbf{z} \in \operatorname*{arg\,min}_{\mathbf{y}\in\mathbb{R}^d} \sum_{r=1}^{m} \left\|\mathbf{y} - \mathbf{x}^{(r)}\right\|_2 . \tag{2}$$

Fig. 26 illustrates a fundamental property of the GM: If $\mathbf{z}$ does not coincide with any of the input vectors, then the unit vectors pointing from $\mathbf{z}$ to each $\mathbf{x}^{(r)}$ must sum to zero—this is the zero-subgradient (optimality) condition for (2). It turns out that the solution to (2) cannot be arbitrarily pulled away from trustworthy input vectors as long as they are the majority [?, Th. 2.2].

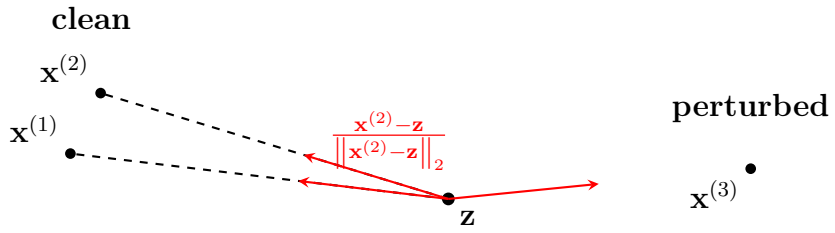


Fig. 26. Consider a solution $\mathbf{z}$ of (2) that does not coincide with any of the input vectors. The optimality condition for (2) requires that the unit vectors from $\mathbf{z}$ to the input vectors sum to zero.

See also: vector, subgradient.

gradient descent (GD) GD is an iterative method for finding the minimum of a differentiable function $f : \mathbb{R}^d \to \mathbb{R}$. GD generates a sequence of estimates $\mathbf{w}^{(0)}$, $\mathbf{w}^{(1)}$, $\mathbf{w}^{(2)}$, ... that (ideally) converge to a minimum of f. At each iteration k, GD refines the current estimate $\mathbf{w}^{(k)}$ by taking a step in the direction of the steepest descent of a local linear approximation.

This direction is given by the negative gradientti $\nabla f(\mathbf{w}^{(k)})$ of the function f at the current estimate $\mathbf{w}^{(k)}$. The resulting update rule is given by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \nabla f(\mathbf{w}^{(k)}) \tag{3}$$

where $\eta > 0$ is a suitably small step size. For a suitably choosen step size η, the update typically reduces the function value, i.e., $f(\mathbf{w}^{(k+1)}) < f(\mathbf{w}^{(k)})$. Fig. 27 illustrates a single GD step.

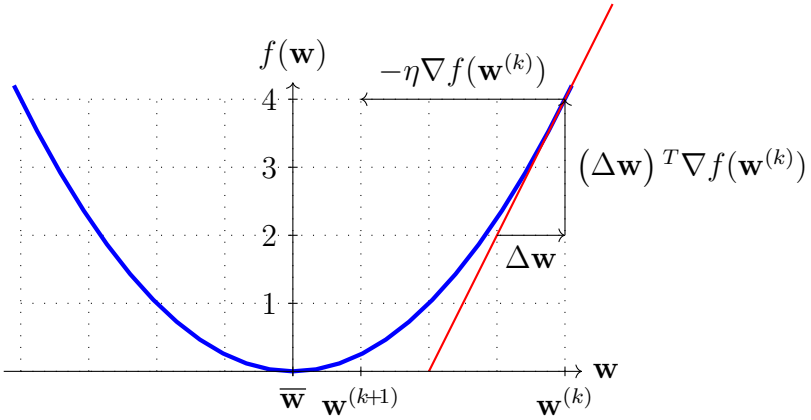


Fig. 27. A single gradient step (3) toward the minimizer $\overline{\mathbf{w}}$ of $f(\mathbf{w})$.

See also: minimum, differentiable, gradientti, step size, gradient step.

gradient step Given a differentiable real-valued function $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ and a vector $\mathbf{w} \in \mathbb{R}^d$, the gradientti step updates $\mathbf{w}$ by adding the scaled negative gradientti $\nabla f(\mathbf{w})$ to obtain the new vector (see Fig. 28)

$$\widehat{\mathbf{w}} := \mathbf{w} - \eta \nabla f(\mathbf{w}). \tag{4}$$

Mathematically, the gradientti step is an operator $\mathcal{T}^{(f,\eta)}$ that is parametrized by the function f and the step size η.

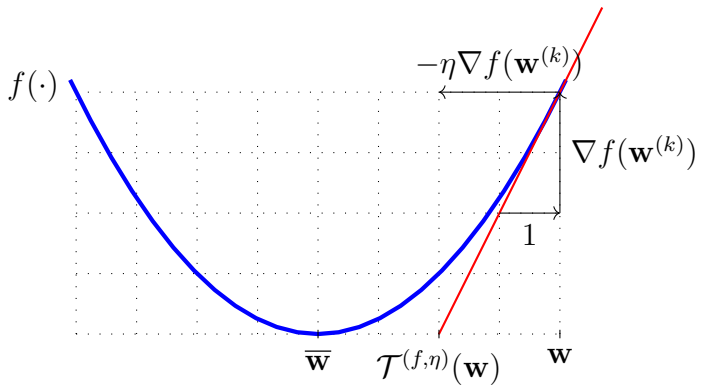


Fig. 28. The basic gradientti step (4) maps a given vector $\mathbf{w}$ to the updated vector $\mathbf{w}'$. It defines an operator $\mathcal{T}^{(f,\eta)}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d : \mathbf{w} \mapsto \widehat{\mathbf{w}}$.

Note that the gradientti step (4) optimizes locally—in a neighborhood whose size is determined by the step size η—a linear approximation to the function $f(\cdot)$. A natural yleistys of (4) is to locally optimize the function itself—instead of its linear approximation—such that

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}' \in \mathbb{R}^d} f(\mathbf{w}') + \frac{1}{\eta} \|\mathbf{w} - \mathbf{w}'\|_2^2 . \tag{5}$$

We intentionally use the same symbol η for the parameter in (5) as we used for the step size in (4). The larger the η we choose in (5), the more progress the update will make toward reducing the function value $f(\widehat{\mathbf{w}})$. Note that, much like the gradientti step (4), the update (5) also defines an operator that is parameterized by the function $f(\cdot)$ and the oppimisnopeus η. For a convex function $f(\cdot)$, this operator is known as the proximal operator of $f(\cdot)$ [**?**].

See also: differentiable, function, vector, gradientti, step size, neighborhood, yleistys, parameter, oppimisnopeus, convex, proximal operator.

gradient-based methods Gradientti-based methods are iterative techniques for finding the minimum (or maximum) of a differentiable kohdefunktiot of the model parameters. These methods construct a sequence of approximations to an optimal choice for model parameters that results in a minimum (or maximum) value of the kohdefunktiot. As their name indicates, gradientti-based methods use the gradientit of the kohdefunktiot evaluated during previous iterations to construct new, (hopefully) improved model parameters. One important example of a gradientti-based method is GD.

See also: gradientti, minimum, maximum, differentiable, kohdefunktiot, model parameters, GD.

gradientti For a real-valued function $f : \mathbb{R}^d \to \mathbb{R} : \mathbf{w} \mapsto f(\mathbf{w})$, if a vector $\mathbf{g}$ exists such that $\lim_{\mathbf{w} \to \mathbf{w}'} f(\mathbf{w}) - \big(f(\mathbf{w}') + \mathbf{g}^T(\mathbf{w} - \mathbf{w}')\big)/\|\mathbf{w} - \mathbf{w}'\| = 0$, it is referred to as the gradient of f at $\mathbf{w}'$. If it exists, the gradient is unique and denoted by $\nabla f(\mathbf{w}')$ or $\nabla f(\mathbf{w})\big|_{\mathbf{w}'}$ [2].

See also: function, vector.

graph A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a pair that consists of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$. In its most general form, a graph is specified by a map that assigns each edge $e \in \mathcal{E}$ a pair of nodes [**?**]. One important family of graphs is simple undirected graphs. A simple undirected graph is obtained by identifying each edge $e \in \mathcal{E}$ with two different nodes $\{i, i'\}$. Weighted graphs also specify numeric weights A_e for each edge $e \in \mathcal{E}$.

See also: map, weights.

graph clustering Graph klusterointi aims to cluster data points that are

represented as the nodes of a graph $\mathcal{G}$. The edges of $\mathcal{G}$ represent pairwise similarities between data points. We can sometimes quantify the extent of these similarities by an edge weight [?], [?].

See also: graph, klusterointi, data point, edge weight.

harha Consider an koneoppiminen method using a parameterized hypothesis space $\mathcal{H}$. It learns the model parameters $\mathbf{w} \in \mathbb{R}^d$ using the tietoaineisto

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(r)}, y^{(r)}\right) \right\}_{r=1}^{m}.$$

To analyze the properties of the koneoppiminen method, we typically interpret the data points as realizations of i.i.d. satunnaismuuttujat,

$$y^{(r)} = h^{(\overline{\mathbf{w}})}\left(\mathbf{x}^{(r)}\right) + \varepsilon^{(r)}, r = 1, \ldots, m.$$

We can then interpret the koneoppiminen method as an estimator $\widehat{\mathbf{w}}$ computed from $\mathcal{D}$ (e.g., by solving ERM). The (squared) bias incurred by the estimate $\widehat{\mathbf{w}}$ is then defined as $B^2 := \left\|\mathbb{E}\{\widehat{\mathbf{w}}\} - \overline{\mathbf{w}}\right\|_2^2$.

See also: i.i.d., satunnaismuuttuja, todennäköisyysmalli, estimation error.

harjaregressio Consider a regressio problem where the goal is to learn a hypothesis $h^{(\mathbf{w})}$ for predicting the numeric nimiö of a data point based on its piirevektori. Ridge regressio learns the parameters $\mathbf{w}$ by minimizing the penalized average neliövirhehäviö. The average neliövirhehäviö is measured on a set of labeled data pointt (i.e., the training set)

$$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right).$$

The penalty term is the scaled squared Euclidean norm $\alpha\|\mathbf{w}\|_2^2$ with a regularisointi parameter $\alpha > 0$. The purpose of the penalty term is regularisointi, i.e., to prevent ylisovittaminen in the high-dimensional regime, where the number of piirre d exceeds the number of data points m in the training set. Adding $\alpha\|\mathbf{w}\|_2^2$ to the average neliövirhehäviö is equivalent to computing the average neliövirhehäviö on an augmented training set. This augmented training set is obtained by replacing each data point $\big(\mathbf{x}^{(r)}, y^{(r)}\big)$ in the original training set by the realization of infinitely many i.i.d. satunnaismuuttujat whose todennäköisyysjakauma is centered at $\big(\mathbf{x}^{(r)}, y^{(r)}\big)$.

See also: regressio, regularisointi, map, data augmentation.

high-dimensional regime The high-dimensional regime of ERM is characterized by the effective dimension of the malli being larger than the sample size, i.e., the number of (labeled) data points in the training set. For example, lineaarinen regressio methods operate in the high-dimensional regime whenever the number d of piirre used to characterize data points exceeds the number of data points in the training set. Another example of koneoppiminen methods that operate in the high-dimensional regime is large neuroverkot, which have far more tunable weights (and bias terms) than the total number of data points in the training set. High-dimensional statistics is a recent main thread of probability theory that studies the behavior of koneoppiminen methods in the high-dimensional regime [**?**], [**?**].

See also: ERM, effective dimension, ylisovittaminen, regularisointi.

Hilbert space A Hilbert space is a complete inner product space [?]. That is, it is a vector space equipped with an inner product between pairs of vectors, and it satisfies the additional requirement of completeness, i.e., every Cauchy sequence of vectors converges to a limit within the space. A canonical example of a Hilbert space is the Euclidean space $\mathbb{R}^d$, for some dimension d, consisting of vectors $\mathbf{u} = \left(u_1, \ldots, u_d\right)^T$ and the standard inner product $\mathbf{u}^T\mathbf{v}$.

See also: vector space, vector, Euclidean space.

hinge loss Consider a data point characterized by a piirevektori $\mathbf{x} \in \mathbb{R}^d$ and a binary nimiö $y \in \{-1, 1\}$. The hinge häviö incurred by a real-valued hypothesis map $h(\mathbf{x})$ is defined as

$$L\left((\mathbf{x}, y), h\right) := \max\{0, 1 - yh(\mathbf{x})\}. \tag{6}$$

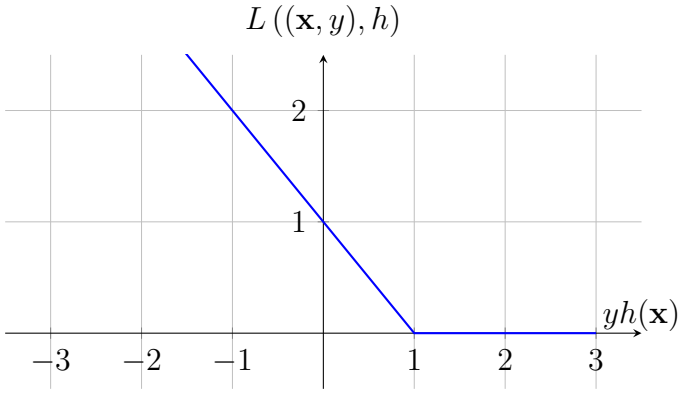


Fig. 29. The hinge häviö incurred by the ennuste $h(\mathbf{x}) \in \mathbb{R}$ for a data point with nimiö $y \in \{-1, 1\}$. A regularized variant of the hinge häviö is used by the support vector machine (SVM) [?].

See also: SVM, luokittelu, luokitin.

histogrammi Consider a tietoaineisto $\mathcal{D}$ that consists of m data points $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$, each of them belonging to some cell $[-U, U] \times \ldots \times [-U, U] \subseteq \mathbb{R}^d$ with side length U. We partition this cell evenly into smaller elementary cells with side length Δ. The histogram of $\mathcal{D}$ assigns each elementary cell to the corresponding fraction of data points in $\mathcal{D}$ that fall into this elementary cell. A visual example of such a histogram is provided in Fig. 30.

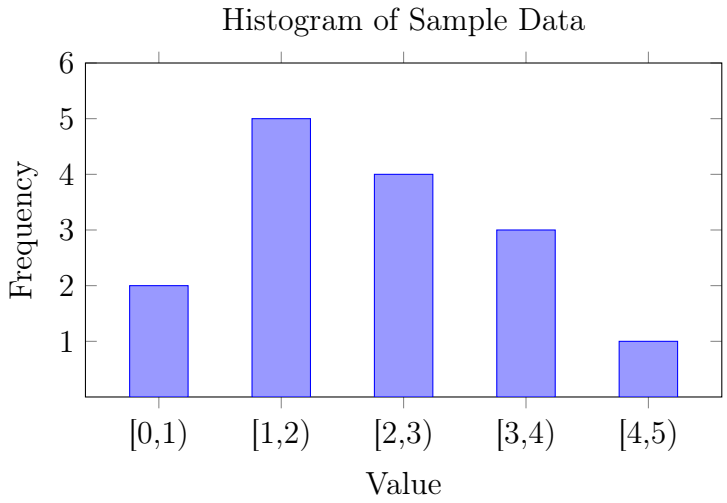


Fig. 30. A histogram representing the frequency of data points falling within discrete value ranges (i.e., bins). Each bar height shows the count of samples in the corresponding interval.

See also: tietoaineisto, data point, sample.

horizontal federated learning (HFL) HFL uses local datasets constitut-

ed by different data points but uses the same piirre to characterize them [?]. For example, weather forecasting uses a network of spatially distributed weather (observation) stations. Each weather station measures the same quantities, such as daily temperature, air pressure, and precipitation. However, different weather stations measure the characteristics or piirre of different spatiotemporal regions. Each spatiotemporal region represents an individual data point, each characterized by the same piirre (e.g., daily temperature or air pressure).

See also: semi-supervised learning (SSL), federoitu oppiminen, vertical federated learning (VFL).

Huber loss The Huber häviö unifies the neliövirhehäviö and the absolute error loss.

See also: häviö, neliövirhehäviö, absolute error loss.

Huber regression Huber regressio refers to ERM-based methods that use the Huber loss as a measure of the ennuste error. Two important special cases of Huber regressio are least absolute deviation regression and lineaarinen regressio. Tuning the threshold parameter of the Huber loss allows the user to trade the robustness of the absolute error loss against the computational benefits of the smooth neliövirhehäviö.

See also: least absolute deviation regression, lineaarinen regressio, absolute error loss, neliövirhehäviö.

hypothesis A hypothesis refers to a map (or function) $h : \mathcal{X} \to \mathcal{Y}$ from the feature space $\mathcal{X}$ to the label space $\mathcal{Y}$. Given a data point with piirre $\mathbf{x}$, we use a hypothesis map h to estimate (or approximate) the nimiö y

using the ennuste $\hat{y} = h(\mathbf{x})$. Koneoppiminen is all about learning (or

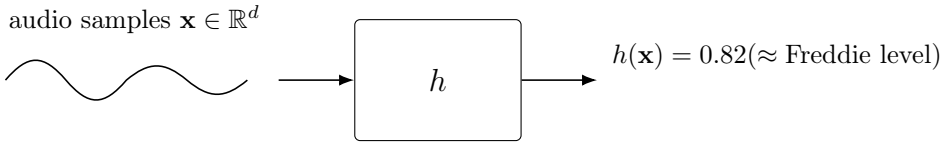


Fig. 31. A hypothesis $h : \mathcal{X} \to \mathcal{Y}$ maps the piirre $\mathbf{x} \in \mathcal{X}$ of a data point to a ennuste $h(\mathbf{x}) \in \mathcal{Y}$ of the nimiö. For example, the koneoppiminen application `https://freddiemeter.withyoutube.com/` uses the samples of an audio recording as piirre predict how closely a person's singing resembles that of Freddie Mercury.

finding) a hypothesis map h such that $y \approx h(\mathbf{x})$ for any data point (with piirre $\mathbf{x}$ and nimiö y). Practical koneoppiminen methods, limited by finite computational resources, must restrict learning to a subset of all possible hypothesis maps. This subset is called the hypothesis space or simply the malli underlying the method.

See also: map, function, ennuste, malli.

hypothesis space A hypothesis space is a mathematical malli that characterizes the learning capacity of an koneoppiminen method. The goal of such a method is to learn a hypothesis map that maps piirre of a data point to a ennuste of its nimiö. Given a finite amount of computational resources, a practical koneoppiminen method typically explores only a restricted set of all possible maps from the feature space to the label space. Such a restricted set is referred to as a hypothesis space $\mathcal{H}$ underlying the koneoppiminen method (see Fig. 32). For the analysis of a given koneoppiminen method, the choice of a hypothesis space $\mathcal{H}$ is not

unique, i.e., any superset containing all maps the method can learn is also a valid hypothesis space.

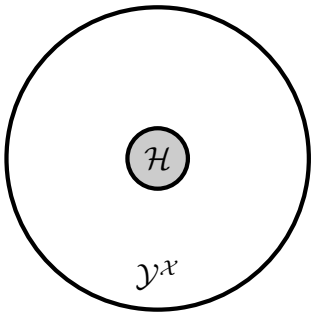


Fig. 32. The hypothesis space $\mathcal{H}$ of an koneoppiminen method is a (typically very small) subset of the (typically very large) set $\mathcal{Y}^{\mathcal{X}}$ of all possible maps from the feature space $\mathcal{X}$ into the label space $\mathcal{Y}$.

On the other hand, from an koneoppiminen engineering perspective, the hypothesis space $\mathcal{H}$ is a design choice for ERM-based methods. This design choice can be guided by the available computational resources and statistical aspects. For instance, if efficient matriisi operations are feasible and a roughly linear relation exists between piirre and nimiöt, a linear model can be a useful choice for $\mathcal{H}$.

See also: hypothesis, malli, map, linear model.

häviö koneoppiminen methods use a häviöfunktio $L\left(\mathbf{z}, h\right)$ to measure the error incurred by applying a specific hypothesis to a specific data point. With a slight abuse of notation, we use the term loss for both the häviöfunktio L itself and the specific value $L\left(\mathbf{z}, h\right)$, for a data point $\mathbf{z}$ and hypothesis h.

See also: häviöfunktio, empirical risk.

häviöfunktio A häviö function is a map

$$L : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}_+ : \big(\big(\mathbf{x}, y\big), h\big) \mapsto L\left((\mathbf{x}, y), h\right).$$

It assigns a nonnegative real number (i.e., the häviö) $L\left((\mathbf{x}, y), h\right)$ to a pair that consists of a data point, with piirre $\mathbf{x}$ and nimiö y, and a hypothesis $h \in \mathcal{H}$. The value $L\left((\mathbf{x}, y), h\right)$ quantifies the discrepancy between the true nimiö y and the ennuste $h(\mathbf{x})$. Lower (closer to zero) values $L\left((\mathbf{x}, y), h\right)$ indicate a smaller discrepancy between ennuste $h(\mathbf{x})$ and nimiö y. Fig. 33 depicts a häviö function for a given data point, with piirre $\mathbf{x}$ and nimiö y, as a function of the hypothesis $h \in \mathcal{H}$.

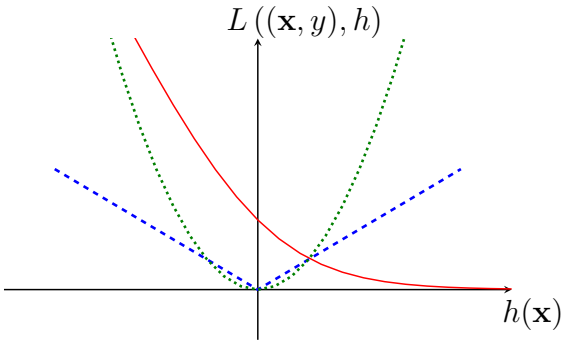


Fig. 33. Some häviö function $L\left((\mathbf{x}, y), h\right)$ for a fixed data point, with piirevektori $\mathbf{x}$ and nimiö y, and a varying hypothesis h. koneoppiminen methods try to find (or learn) a hypothesis that incurs minimal häviö.

See also: häviö, nimiö, piirevektori, ERM.

independent and identically distributed (i.i.d.) A collection of satunnaismuuttujat

$\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$ is referred to as i.i.d. if each $\mathbf{z}^{(r)}$ follows the same todennäköisyysjakauma, and the satunnaismuuttujat are mutually independent. That is, for any collection of events $\mathcal{A}_1, \ldots, \mathcal{A}_m$, we have

$$\mathbb{P}(\mathbf{z}^{(1)} \in \mathcal{A}_1, \ldots, \mathbf{z}^{(m)} \in \mathcal{A}_m) = \prod_{r=1}^{m} \mathbb{P}(\mathbf{z}^{(r)} \in \mathcal{A}_r).$$

See also: satunnaismuuttuja, todennäköisyysjakauma, event, data point, i.i.d. assumption.

independent and identically distributed assumption (i.i.d. assumption) The i.i.d. assumption interprets data points of a tietoaineisto as the realizations of i.i.d. satunnaismuuttujat.

See also: i.i.d., data point, tietoaineisto, realization, satunnaismuuttuja.

käänteismatriisi An inverse matriisi $\mathbf{A}^{-1}$ is defined for a square matriisi $\mathbf{A} \in \mathbb{R}^{n \times n}$ that is of full rank, meaning its columns are linearly independent. In this case, $\mathbf{A}$ is said to be invertible, and its inverse satisfies

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}.$$

A square matriisi is invertible if and only if its determinantti is nonzero. Inverse matriisit are fundamental in solving systems of linear equations and in the closed-form solution of lineaarinen regressio [**?**], [**?**]. The concept of an inverse matriisi can be extended to matriisit that are not square or not full rank. One may define a "left inverse" $\mathbf{B}$ satisfying $\mathbf{B}\mathbf{A} = \mathbf{I}$ or a "right inverse" $\mathbf{C}$ satisfying $\mathbf{A}\mathbf{C} = \mathbf{I}$. For general rectangular or singular matriisit, the Moore–Penrose pseudokäänteisluku $\mathbf{A}^{+}$ provides a unified concept of a generalized inverse matriisi [3].

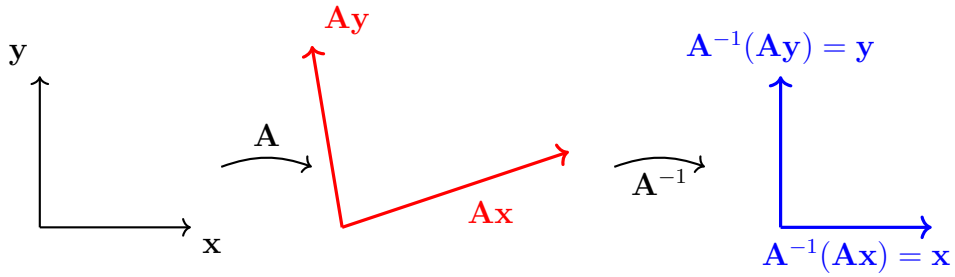


Fig. 34. A matriisi $\mathbf{A}$ represents a linear transformation of $\mathbb{R}^2$. The inverse matriisi $\mathbf{A}^{-1}$ represents the inverse transformation.

See also: matriisi, determinantti, lineaarinen regressio, pseudokäänteisluku.

Jacobi method The Jacobi method is an algorithm for solving systems of linear equations (i.e., a linear system) of the form $\mathbf{Ax} = \mathbf{b}$. Here, $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a square matriisi with nonzero main diagonal entries. The method constructs a sequence $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots$ by updating each entry of $\mathbf{x}^{(k)}$ according to

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right).$$

Note that all entries $x_1^{(k)}, \ldots, x_d^{(k)}$ are updated simultaneously. The above iteration converges to a solution, i.e., $\lim_{k \to \infty} \mathbf{x}^{(k)} = \mathbf{x}$, under certain conditions on the matriisi $\mathbf{A}$, e.g., being strictly diagonally dominant or symmetric positive definite [3], [?], [?]. Jacobi-type methods are appealing for large linear systems due to their parallelizable structure [?]. We can interpret the Jacobi method as a fixed-point iteration. Indeed,

using the decomposition $\mathbf{A} = \mathbf{D} + \mathbf{R}$, with $\mathbf{D}$ being the diagonal of $\mathbf{A}$, allows us to rewrite the linear equation $\mathbf{Ax} = \mathbf{b}$ as a fixed-point equation

$$\mathbf{x} = \underbrace{\mathbf{D}^{-1}(\mathbf{b} - \mathbf{Rx})}_{\mathcal{F}\mathbf{x}}$$

which leads to the iteration $\mathbf{x}^{(k+1)} = \mathbf{D}^{-1}(\mathbf{b} - \mathbf{R}\mathbf{x}^{(k)})$.

As an example, for the linear equation $\mathbf{Ax} = \mathbf{b}$, where

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

the Jacobi method updates each component of $\mathbf{x}$ as follows:

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{a_{11}} \left(b_1 - a_{12} x_2^{(k)} - a_{13} x_3^{(k)} \right); \\ x_2^{(k+1)} &= \frac{1}{a_{22}} \left(b_2 - a_{21} x_1^{(k)} - a_{23} x_3^{(k)} \right); \\ x_3^{(k+1)} &= \frac{1}{a_{33}} \left(b_3 - a_{31} x_1^{(k)} - a_{32} x_2^{(k)} \right). \end{aligned}$$

See also: algorithm, matriisi, fixed-point iteration, optimization method.

k-kertainen ristiinvalidointi k-fold CV is a method for learning and validating a hypothesis using a given tietoaineisto. This method divides the tietoaineisto evenly into k subsets or folds and then executes k repetitions of malli training (e.g., via ERM) and validointi. Each repetition uses a different fold as the validation set and the remaining $k - 1$ folds as a training set. The final output is the average of the validointivirhe obtained from the k repetitions.

See also: ERM, validointi, validation set, training set, validointivirhe.

k-means The k-means principle is an optimization-based approach to the klusterointi of data points that are characterized by a numeric piirevektori [8, Ch. 8]. As a osittava klusterointi approach, k-means partitions a tietoaineisto into k disjoint subsets (or ryppäät), which are indexed by $c = 1, \ldots, k$. Each rypäs $\mathcal{C}$ is characterized by the average piirevektori of data points that belong to it. This average (or mean) piirevektori is referred to as the cluster centroid $\boldsymbol{\mu}^{(c)}$. A visual illustration is provided in Fig. 35.

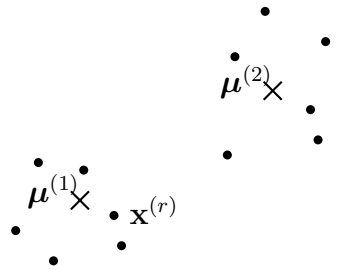


Fig. 35. A scatterplot of data points, indexed by $r = 1, \ldots, m$ and characterized by piirrevektori $\mathbf{x}^{(r)} \in \mathbb{R}^2$. The scatterplot also includes two cluster centroids $\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(2)} \in \mathbb{R}^2$.

In general, the k-means problem is a challenging optimization problem [**?**]. However, there is a simple iterative method for finding approximately optimal cluster centroids. This method, referred to as Lloyd's method, alternates between: 1) updating the rypäs assignments based on the nearest current cluster centroid; and 2) recalculating the cluster centroids given the updated rypäs assignments [**?**].

See also: osittava klusterointi, rypäs.

kernel method A ydinfunktio method is an koneoppiminen method that uses a ydinfunktio K to map the original (i.e., raw) piirevektori $\mathbf{x}$ of a data point to a new (transformed) piirevektori $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$ [?], [?]. The motivation for transforming the piirrevektori is that, by using a suitable ydinfunktio, the data points have a more "pleasant" geometry in the transformed feature space. For example, in a binary luokittelu problem, using transformed piirrevektori $\mathbf{z}$ might allow us to use linear models, even if the data points are not linearly separable in the original feature space (see Fig. 36).

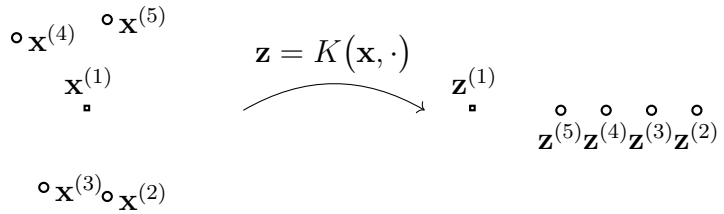


Fig. 36. Five data points characterized by piirrevektori $\mathbf{x}^{(r)}$ and nimiöt $y^{(r)} \in \{\circ, \square\}$, for $r = 1, \ldots, 5$. With these piirrevektori, there is no way to separate the two classes by a straight line (representing the päätöspinta of a lineaarinen luokitin). In contrast, the transformed piirrevektori $\mathbf{z}^{(r)} = K\big(\mathbf{x}^{(r)}, \cdot\big)$ allow us to separate the data points using a lineaarinen luokitin.

See also: ydinfunktio, piirevektori, feature space, lineaarinen luokitin.

klusterointi Clustering methods decompose a given set of data points into a few subsets, which are referred to as ryppäät. Each rypäs consists of data points that are more similar to each other than to data points

outside the rypäs. Different clustering methods use different measures for the similarity between data points and different forms of rypäs representations. The clustering method k-means uses the average piirevektori of a rypäs (i.e., the rypäs mean) as its representative. A popular soft clustering method based on Gaussin sekoitemalli represents a rypäs by a multivariate normal distribution.

See also: rypäs, k-means, soft clustering, Gaussin sekoitemalli.

kohdefunktio An objective function is a map that assigns a numeric objective value $f(\mathbf{w})$ to each choice $\mathbf{w}$ of some variable that we want to optimize (see Fig. 37). In the context of koneoppiminen, the optimization variable could be the model parameters of a hypothesis $h^{(\mathbf{w})}$. Common objective functions include the risk (i.e., expected häviö) or the empirical risk (i.e., average häviö over a training set). koneoppiminen methods apply optimization techniques, such as gradient-based methods, to find the choice $\mathbf{w}$ with the optimal value (e.g., the minimum or the maximum) of the objective function.

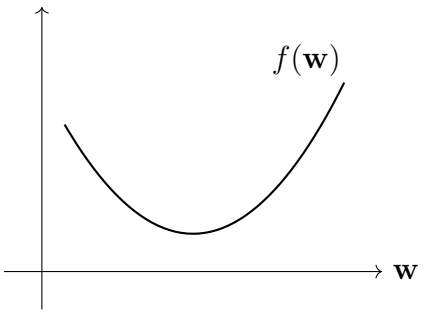


Fig. 37. An objective function maps each possible value $\mathbf{w}$ of an optimization variable, such as the model parameters of an koneoppiminen malli, to a value $f(\mathbf{w})$ that measures the usefulness of $\mathbf{w}$.

See also: häviö, empirical risk, ERM, optimization problem.

koneoppiminen ML aims to predict a nimiö from the piirre of a data point. ML methods achieve this by learning a hypothesis from a hypothesis space (or malli) through the minimization of a häviöfunktio [8], [**?**]. One precise formulation of this principle is ERM. Different ML methods are obtained from different design choices for data points (i.e., their piirre and nimiö), the malli, and the häviöfunktio [8, Ch. 3].
See also: malli, data, häviö.

kovarianssimatriisi The covariance matriisi of an satunnaismuuttuja $\mathbf{x} \in \mathbb{R}^d$ is defined as $\mathbb{E}\Big\{ \big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)\big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)^T \Big\}$.
See also: covariance, matriisi, satunnaismuuttuja.

Kronecker product The Kronecker product of two matriisit $\mathbf{A} \in \mathbb{R}^{m \times n}$

and $\mathbf{B} \in \mathbb{R}^{p \times q}$ is a block matriisi denoted by $\mathbf{A} \otimes \mathbf{B}$ and defined as [3], [**?**]

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix} \in \mathbb{R}^{mp \times nq}.$$

The Kronecker product is a special case of the tensor product for matriisit and is widely used in multivariate statistics, linear algebra, and structured koneoppiminen malli. It satisfies the identity $(\mathbf{A} \otimes \mathbf{B})(\mathbf{x} \otimes \mathbf{y}) = (\mathbf{A}\mathbf{x}) \otimes (\mathbf{B}\mathbf{y})$ for vectors $\mathbf{x}$ and $\mathbf{y}$ of compatible dimensions.

See also: matriisi, koneoppiminen, malli, vector.

Kullback–Leibler divergence (KL divergence) The KL divergence is a quantitative measure of how different one todennäköisyysjakauma is from another [**?**].

See also: todennäköisyysjakauma.

label space In a koneoppiminen application, each data point is described by a set of piirre together with an associated nimiö. The set of all admissible nimiö values is called the label space, denoted by $\mathcal{Y}$. Importantly, $\mathcal{Y}$ may include values that no observed data point has as its nimiö value. To a large extent, the choice of $\mathcal{Y}$ is up to the koneoppiminen engineer and depends on the problem formulation. Fig. 38 shows some examples of label spacet that are commonly used in koneoppiminen applications.

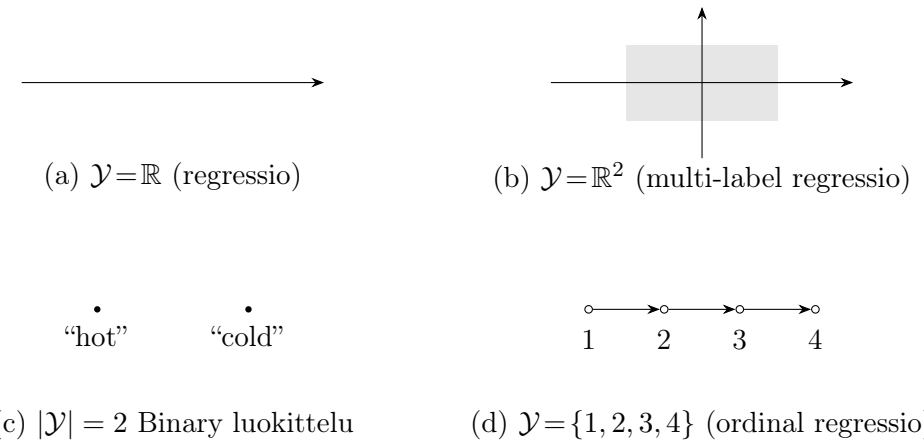


Fig. 38. Examples of label spacet and corresponding flavours of koneoppiminen.

The choice of label space $\mathcal{Y}$ determines the flavour of koneoppiminen methods appropriate for the application at hand. Regressio methods use the $\mathcal{Y} = \mathbb{R}$ while binary luokittelu methods use a nimiö space $\mathcal{Y}$ that consists of two different elements, i.e., $|\mathcal{Y}| = 2$. Ordinal regressio methods use a finite, ordered set of nimiö values, e.g., $\mathcal{Y} = \{1, 2, 3, 4\}$ with the natural ordering $1 < 2 < 3 < 4$.

See also: data point,nimiö, regressio, luokittelu.

labeled data point A data point whose nimiö is known or has been determined by some means that might require human labor.

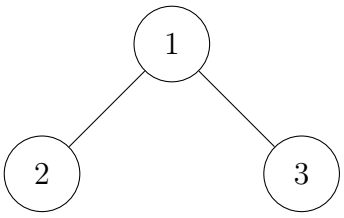
See also: data point, nimiö.

Laplacian matrix The structure of a graph $\mathcal{G}$, with nodes $i = 1, \ldots, n$, can be analyzed using the properties of special matriisit that are associated with $\mathcal{G}$. One such matriisi is the graph Laplacian matriisi $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{n \times n}$,

which is defined for an undirected and weighted graph [?], [?]. It is defined elementwise as (see Fig. 39)

$$L^{(\mathcal{G})}_{i,i'} := \begin{cases} -A_{i,i'}, & \text{for } i \neq i', \{i,i'\} \in \mathcal{E}; \\ \sum\limits_{i'' \neq i} A_{i,i''}, & \text{for } i = i'; \\ 0, & \text{else.} \end{cases}$$

Here, $A_{i,i'}$ denotes the edge weight of an edge $\{i,i'\} \in \mathcal{E}$.



(a)

$$\mathbf{L}^{(\mathcal{G})} = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

(b)

Fig. 39. (a) Some undirected graph $\mathcal{G}$ with three nodes $i = 1, 2, 3$. (b) The Laplacian matriisi $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{3 \times 3}$ of $\mathcal{G}$.

See also: graph, matriisi, edge weight.

large language model (LLM) LLM is an umbrella term for koneoppiminen methods that process and generate humanlike text. These methods typically use deep nets with billions (or even trillions) of parameters. A widely used choice for the network architecture is referred to as Transformers [?]. The training of LLMs is often based on the task of predicting a few words that are intentionally removed from a large text corpus. Thus, we can construct labeled data points simply by selecting some

words from a given text as nimiöt and the remaining words as piirre of data points. This construction requires very little human supervision and allows for generating sufficiently large training sets for LLMs.

See also: deep net, labeled data point.

law of large numbers The law of large numbers refers to the convergence of the average of an increasing (large) number of i.i.d. satunnaismuuttujat to the mean of their common todennäköisyysjakauma. Different instances of the law of large numbers are obtained by using different notions of convergence [**?**].

See also: convergence, i.i.d., satunnaismuuttuja, mean, todennäköisyysjakauma.

layer A deep net is an ANN that consists of consecutive layers, indexed by $\ell = 1, 2, \ldots, L$. The ℓ-th layer consists of artificial neurons $a_1^{(\ell)}, \ldots, a_{d^{(\ell)}}^{(\ell)}$ with the layer width $d^{(\ell)}$. Each of these artificial neurons evaluates an activation function for a weighted sum of the outputs (or activations) of the previous layer $\ell - 1$. The input to layer $\ell = 1$ is formed from weighted sums of the piirre of the data point for which the deep net computes a ennuste. The outputs of the neurons in layer ℓ are then, in turn, used to form the inputs for the neurons in the next layer. The final (output) layer consists of a single neuron whose output is used as the ennuste delivered by the deep net.

See also: deep net, ANN.

learning task Consider a tietoaineisto $\mathcal{D}$ consisting of multiple data points $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$. For example, $\mathcal{D}$ can be a collection of images in an image

database. A learning task is defined by specifying those properties (or attributes) of a data point that are used as its piirre and nimiöt. Given a choice of malli $\mathcal{H}$ and häviöfunktio, a learning task leads to an instance of ERM and can thus be represented by the associated kohdefunktiot $\widehat{L}\big(h|\mathcal{D}\big)$ for $h \in \mathcal{H}$. Importantly, multiple distinct learning tasks can be constructed from the same tietoaineisto by selecting different sets of piirre and nimiöt (see Fig. 40).

An image showing cows grazing in the Austrian countryside.

Task 1 (regressio):
Piirre are the RGB values of all image pixels, and the nimiö is the number of cows depicted.

Task 2 (luokittelu):
Piirre include the average green intensity of the image, and the nimiö indicates whether cows should be moved to another location (i.e., yes/no).

Fig. 40. Two learning tasks constructed from a single image tietoaineisto. These tasks differ in piirre selection and choice of nimiö (i.e., the objective), but are both derived from the same tietoaineisto.

Different learning tasks arising from the same underlying tietoaineisto are often coupled. For example, when a todennäköisyysmalli is used to generate data points, statistical dependencies among different nimiöt induce dependencies among the corresponding learning tasks. In general,

solving learning tasks jointly, e.g., using multitask learning methods, tends to be more effective than solving them independently (thereby ignoring dependencies among learning tasks) [?], [?], [?].

See also: tietoaineisto, malli, häviöfunktio, kohdefunktiot, multitask learning, label space.

least absolute deviation regression Least absolute deviation regression is an instance of ERM using the absolute error loss. It is a special case of Huber regression.

See also: ERM, absolute error loss, Huber regression.

least absolute shrinkage and selection operator (Lasso) The Lasso is an instance of SRM. It learns the weights $\mathbf{w}$ of a linear map $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ from a training set. Lasso is obtained from lineaarinen regressio by adding the scaled ℓ_1-norm $\alpha \|\mathbf{w}\|_1$ to the average neliövirhehäviö incurred on the training set.

See also: SRM, weights, linear map, training set, lineaarinen regressio, norm, neliövirhehäviö.

lineaarinen luokitin Consider data points characterized by numeric piirre $\mathbf{x} \in \mathbb{R}^d$ and a nimiö $y \in \mathcal{Y}$ from some finite label space $\mathcal{Y}$. A linear luokitin is characterized by having päätösalue that are separated by hyperplanes in $\mathbb{R}^d$ [8, Ch. 2].

See also: data point, piirre, nimiö, label space, luokitin, päätösalue.

lineaarinen regressio Linear regressio aims to learn a linear hypothesis map to predict a numeric nimiö based on the numeric piirre of a data point. The quality of a linear hypothesis map is measured using the

average neliövirhehäviö incurred on a set of labeled data points, which we refer to as the training set.

See also: regressio, hypothesis, map, nimiö, piirre, data point, neliövirhehäviö, labeled data point, training set.

linear map A linear map $f : \mathbb{R}^n \to \mathbb{R}^m$ is a function that satisfies additivity, i.e., $f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$, and homogeneity, i.e., $f(c\mathbf{x}) = cf(\mathbf{x})$, for all vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ and scalars $c \in \mathbb{R}$. In particular, $f(\mathbf{0}) = \mathbf{0}$. Any linear map can be represented as a matriisi multiplication $f(\mathbf{x}) = \mathbf{A}\mathbf{x}$ for some matriisi $\mathbf{A} \in \mathbb{R}^{m \times n}$. The collection of real-valued linear maps for a given dimension n constitute a linear model, which is used in many koneoppiminen methods.

See also: map, function, vector, matriisi, linear model, koneoppiminen.

linear model Consider an koneoppiminen application involving data points, each represented by a numeric piirevektori $\mathbf{x} \in \mathbb{R}^d$. A linear malli defines a hypothesis space consisting of all real-valued linear maps from $\mathbb{R}^d$ to $\mathbb{R}$ such that

$$\mathcal{H}^{(d)} := \left\{ h : \mathbb{R}^d \to \mathbb{R} \mid h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} \text{ for some } \mathbf{w} \in \mathbb{R}^d \right\}.$$

Each value of d defines a different hypothesis space, corresponding to the number of piirre used to compute the ennuste $h(\mathbf{x})$. The choice of d is often guided not only by computational aspects (e.g., fewer features reduce computation) and statistical aspects (e.g., more features typically reduce harha and risk), but also by tulkittavuus. A linear malli using a small number of well-chosen piirre is generally considered more interpretable [**?**], [**?**]. The linear malli is attractive because it can

typically be trained using scalable convex optimization methods [?], [?]. Moreover, linear malli often permit rigorous statistical analysis, including fundamental limits on the minimum achievable risk [?]. They are also useful for analyzing more complex nonlinear malli such as neuroverkot. For instance, a deep net can be viewed as the composition of a feature map—implemented by the input and hidden layers—and a linear malli in the output layer. Similarly, a päätöspuu can be interpreted as applying a one-hot-encoded feature map based on päätösalue, followed by a linear malli that assigns a ennuste to each region. More generally, any trained malli $\hat{h} \in \mathcal{H}$ that is differentiable at some $\mathbf{x}'$ can be locally approximated by a linear map $g(\mathbf{x})$. Fig. 41 illustrates such a local linear approximation, defined by the gradientti $\nabla \hat{h}(\mathbf{x}')$. Note that the gradientti is only defined where $\hat{h}$ is differentiable. To ensure robustness in the context of trustworthy AI, one may prefer malli whose associated map $\hat{h}$ is Lipschitz continuous. A classic result in mathematical analysis—Rademacher's Theorem—states that if $\hat{h}$ is Lipschitz continuous with some constant L over an open set $\Omega \subseteq \mathbb{R}^d$, then $\hat{h}$ is differentiable almost everywhere in Ω [?, Th. 3.1].

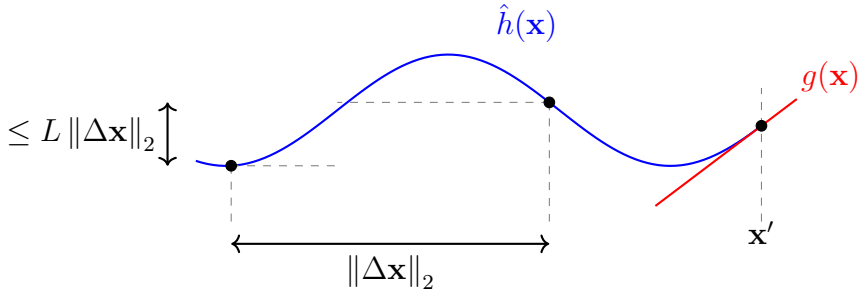


Fig. 41. A trained malli $\hat{h}(\mathbf{x})$ that is differentiable at a point $\mathbf{x}'$ can be locally approximated by a linear map $g \in \mathcal{H}^{(d)}$. This local approximation is determined by the gradientti $\nabla \hat{h}(\mathbf{x}')$.

See also: malli, hypothesis space, linear map, tulkittavuus, LIME.

local dataset The concept of a local tietoaineisto is in between the concept of a data point and a tietoaineisto. A local tietoaineisto consists of several individual data points characterized by piirre and nimiöt. In contrast to a single tietoaineisto used in basic koneoppiminen methods, a local tietoaineisto is also related to other local tietoaineistot via different notions of similarity. These similarities might arise from todennäköisyysmalli or communication infrastructure and are encoded in the edges of an FL network.

See also: tietoaineisto, data point, piirre, nimiö, koneoppiminen, todennäköisyysmalli, FL network.

local interpretable model-agnostic explanations (LIME) Consider a trained malli (or learned hypothesis) $\widehat{h} \in \mathcal{H}$, which maps the piirevektori of a data point to the ennuste $\widehat{y} = \widehat{h}$. LIME is a technique for explaining the behavior of $\widehat{h}$, locally around a data point with piirevektori $\mathbf{x}^{(0)}$ [?].

The explanation is given in the form of a local approximation $g \in \mathcal{H}'$ of $\widehat{h}$ (see Fig. 42). This approximation can be obtained by an instance of ERM with a carefully designed training set. In particular, the training set consists of data points with piirrevektori centered around $\mathbf{x}^{(0)}$ and the (pseudo-)nimiö $\widehat{h}(\mathbf{x})$. Note that we can use a different malli $\mathcal{H}'$ for the approximation from the original malli $\mathcal{H}$. For example, we can use a päätöspuu to locally approximate a deep net. Another widely used choice for $\mathcal{H}'$ is the linear model.

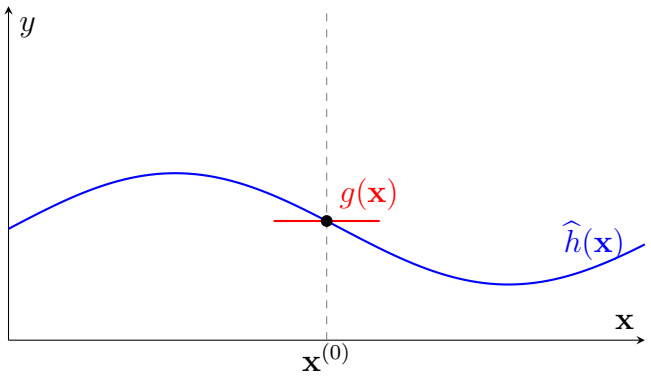


Fig. 42. To explain a trained malli $\widehat{h} \in \mathcal{H}$, around a given piirevektori $\mathbf{x}^{(0)}$, we can use a local approximation $g \in \mathcal{H}'$.

See also: malli, explanation, ERM, training set, nimiö, päätöspuu, deep net, linear model.

local model Consider a collection of devices that are represented as nodes $\mathcal{V}$ of an FL network. A local malli $\mathcal{H}^{(i)}$ is a hypothesis space assigned to a node $i \in \mathcal{V}$. Different nodes can have different hypothesis spaces, i.e., in general, $\mathcal{H}^{(i)} \neq \mathcal{H}^{(i')}$ for different nodes $i, i' \in \mathcal{V}$.

See also: device, FL network, malli, hypothesis space.

logistic loss Consider a data point characterized by the piirre $\mathbf{x}$ and a binary nimiö $y \in \{-1, 1\}$. We use a real-valued hypothesis h to predict the nimiö y from the piirre $\mathbf{x}$. The logistic häviö incurred by this ennuste is defined as

$$L\left((\mathbf{x}, y), h\right) := \log\left(1 + \exp\left(-yh(\mathbf{x})\right)\right). \tag{7}$$

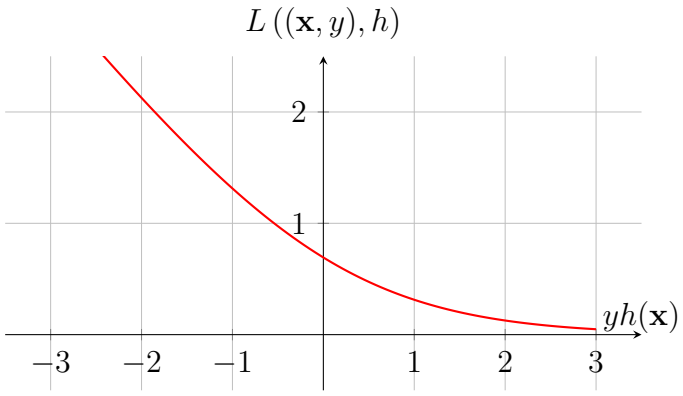


Fig. 43. The logistic häviö incurred by the ennuste $h(\mathbf{x}) \in \mathbb{R}$ for a data point with nimiö $y \in \{-1, 1\}$.

Note that the expression (7) for the logistic häviö applies only for the label space $\mathcal{Y} = \{-1, 1\}$ and when using the thresholding rule (8).

See also: data point, piirre, nimiö, hypothesis, häviö, ennuste, label space.

logistic regression Logistic regressio learns a linear hypothesis map (or luokitin) $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ to predict a binary nimiö y based on the numeric piirevektori $\mathbf{x}$ of a data point. The quality of a linear hypothesis map is

measured by the average logistic loss on some labeled data points (i.e., the training set).

See also: regressio, hypothesis, map, luokitin, nimiö, piirevektori, data point, logistic loss, labeled data point, training set.

luokitin A classifier is a hypothesis (i.e., a map) $h(\mathbf{x})$ used to predict a nimiö taking on values from a finite label space. We might use the function value $h(\mathbf{x})$ itself as a ennuste $\hat{y}$ for the nimiö. However, it is customary to use a map $h(\cdot)$ that delivers a numeric quantity. The ennuste is then obtained by a simple thresholding step. For example, in a binary luokittelu problem with a label space $\mathcal{Y} \in \{-1, 1\}$, we might use a real-valued hypothesis map $h(\mathbf{x}) \in \mathbb{R}$ as a classifier. A ennuste $\hat{y}$ can then be obtained via thresholding,

$$\hat{y} = 1 \text{ for } h(\mathbf{x}) \geq 0 \text{ and } \hat{y} = -1 \text{ otherwise.} \tag{8}$$

We can characterize a classifier by its päätösalue $\mathcal{R}_a$, for every possible nimiö value $a \in \mathcal{Y}$.

See also: hypothesis, luokittelu, päätösalue.

luokittelu Classification is the task of determining a discrete-valued nimiö y for a given data point, based solely on its piirre $\mathbf{x}$. The nimiö y belongs to a finite set, such as $y \in \{-1, 1\}$ or $y \in \{1, \ldots, 19\}$, and represents the category to which the corresponding data point belongs.

See also: nimiö, data point, piirre.

malli The study and design of koneoppiminen methods is often based on a mathematical model [**?**]. Maybe the most widely used example of a mat-

hematical model for koneoppiminen is a hypothesis space. A hypothesis space consists of hypothesis maps that are used by an koneoppiminen method to predict nimiöt from the piirre of data points. Another important type of mathematical model is a todennäköisyysmalli, which consists of todennäköisyysjakaumat that describe how data points are generated. Unless stated otherwise, we use the term model to refer specifically to the hypothesis space underlying an koneoppiminen method. We illustrate one example of a hypothesis space and a todennäköisyysmalli in Fig. 44.

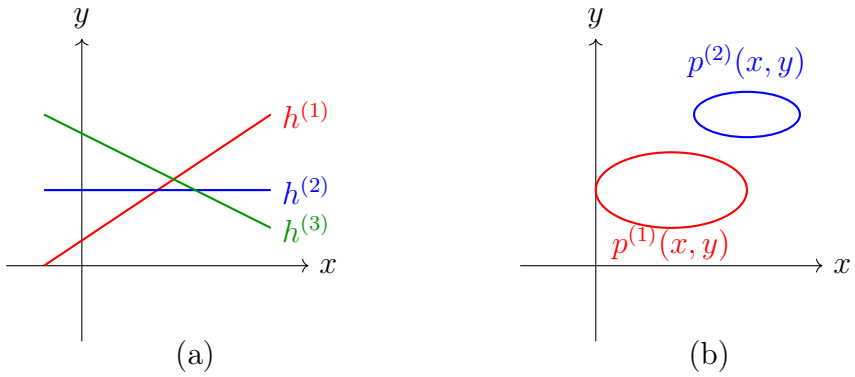


Fig. 44. Two types of mathematical models used in koneoppiminen. (a) A hypothesis space consisting of three linear maps. (b) A todennäköisyysmalli consisting of todennäköisyysjakauma over the plane spanned by the piirre and nimiö values of a data point.

See also: hypothesis space, todennäköisyysmalli, todennäköisyysjakauma.

mallin valinta In koneoppiminen, malli selection refers to the process of

choosing between different candidate malli. In its most basic form, malli selection amounts to: 1) training each candidate malli; 2) computing the validointivirhe for each trained malli; and 3) choosing the malli with the smallest validointivirhe [8, Ch. 6].

See also: koneoppiminen, malli, validointivirhe.

Markov decision process (MDP) An MDP is a mathematical structure that can be used to study reinforcement learning (RL) applications. An MDP formalizes how reward signals depend on the ennuste (and corresponding actions) made by an RL method. Formally, an MDP is a specific type of stochastic process defined by

- a state space $\mathcal{S}$;
- an action space $\mathcal{A}$ (where each action $a \in \mathcal{A}$ corresponds to a specific ennuste made by the RL method);
- a transition function $\mathbb{P}(s' \mid s, a)$ specifying the todennäköisyysjakauma over the next state $s' \in \mathcal{S}$, given the current state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$;
- a reward function $r(s, a) \in \mathbb{R}$ that assigns a numerical reward to each state-action pair.

The defining property of an MDP is the Markov property. That is, the next state s' and reward only depend on the current state s and action a, not on the entire history of interactions.

See also: RL, reward, ennuste, stochastic process, function, todennäköisyysjakauma.

maximum The maximum of a set $\mathcal{A} \subseteq \mathbb{R}$ of real numbers is the greatest element in that set, if such an element exists. A set $\mathcal{A}$ has a maximum if it is bounded above and attains its supremum (or least upper bound) [2, Sec. 1.4].

See also: supremum.

mean The mean of an satunnaismuuttuja $\mathbf{x}$, which takes on values in a Euclidean space $\mathbb{R}^d$, is its expectation $\mathbb{E}\{\mathbf{x}\}$. It is defined as the Lebesgue integral of $\mathbf{x}$ with respect to the underlying todennäköisyysjakauma P (e.g., see [2] or [6]), i.e.,

$$\mathbb{E}\{\mathbf{x}\} = \int_{\mathbb{R}^d} \mathbf{x} \, \mathrm{d}P(\mathbf{x}).$$

It is useful to think of the mean as the solution of the following risk minimization problem [7]:

$$\mathbb{E}\{\mathbf{x}\} = \operatorname*{arg\,min}_{\mathbf{c} \in \mathbb{R}^d} \mathbb{E}\big\{ \|\mathbf{x} - \mathbf{c}\|_2^2 \big\}.$$

We also use the term to refer to the average of a finite sequence $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. However, these two definitions are essentially the same. Indeed, we can use the sequence $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$ to construct a discrete satunnaismuuttuja $\widetilde{\mathbf{x}} = \mathbf{x}^{(I)}$, with the index I being chosen uniformly at random from the set $\{1, \ldots, m\}$. The mean of $\widetilde{\mathbf{x}}$ is precisely the average $(1/m) \sum_{r=1}^{m} \mathbf{x}^{(r)}$.

See also: satunnaismuuttuja, expectation, todennäköisyysjakauma.

mean squared estimation error (MSEE) Consider an koneoppiminen method that learns model parameters $\widehat{\mathbf{w}}$ based on some tietoaineisto $\mathcal{D}$. If we interpret the data points in $\mathcal{D}$ as i.i.d. realizations of an

satunnaismuuttuja $\mathbf{z}$, we define the estimation error $\Delta\mathbf{w} := \widehat{w} - \overline{\mathbf{w}}$. Here, $\overline{\mathbf{w}}$ denotes the true model parameters of the todennäköisyysjakauma of $\mathbf{z}$. The MSEE is defined as the expectation $\mathbb{E}\{\|\Delta\mathbf{w}\|^2\}$ of the squared Euclidean norm of the estimation error [?], [?].

See also: satunnaismuuttuja, estimation error, todennäköisyysmalli, neliövirhehäviö.

measurable Consider a satunnaiskoe, such as recording the air temperature at an FMI weather station. The corresponding sample space Ω consists of all possible outcomes ω (e.g., all possible temperature values in degree Celsius). In many koneoppiminen applications, we are not interested in the exact outcome ω, but only whether it belongs to a subset $\mathcal{A} \subseteq \Omega$ (e.g., "is the temperature below zero degrees?"). We call such a subset $\mathcal{A}$ measurable if it is possible to decide, for any outcome ω, whether $\omega \in \mathcal{A}$ or not (see Fig. 45).

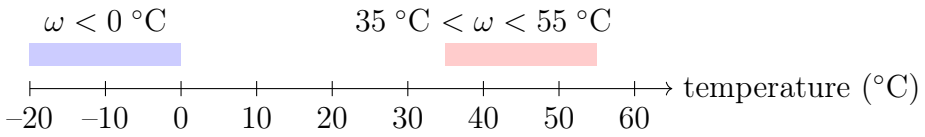


Fig. 45. A sample space constituted by all possible temperature values ω that may be experienced at an FMI station. Two measurable subsets of temperature values, denoted by $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$, are highlighted. For any actual temperature value ω, it is possible to determine whether $\omega \in \mathcal{A}^{(1)}$ and whether $\omega \in \mathcal{A}^{(2)}$.

In principle, measurable sets could be chosen freely (e.g., depending on the resolution of the measuring equipment). However, it is often useful to impose certain completeness requirements on the collection of measurable sets. For example, the sample space itself should be measurable, and the union of two measurable sets should also be measurable. These completeness requirements can be formalized via the concept of σ-algebra (or σ-field) [1], [6], [?]. A measurable space is a pair $\left(\mathcal{X}, \mathcal{F}\right)$ that consists of an arbitrary set $\mathcal{X}$ and a collection $\mathcal{F}$ of measurable subsets of $\mathcal{X}$ that form a σ-algebra.

See also: sample space, probability.

median A median $\text{med}\,(x)$ of a real-valued satunnaismuuttuja x is any number $m \in \mathbb{R}$ such that $\mathbb{P}(x \leq m) \geq 1/2$ and $\mathbb{P}(x \geq m) \geq 1/2$ (see Fig. 46) [?].

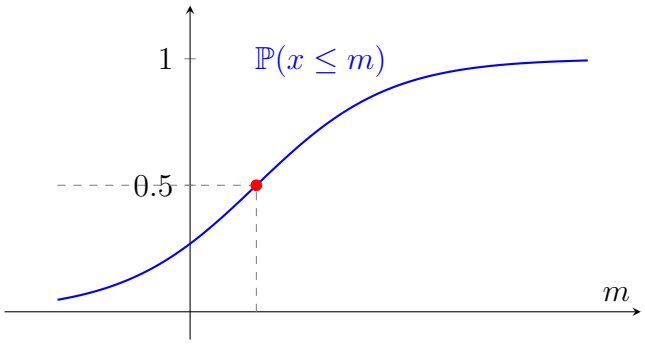


Fig. 46. A representation of a median.

We can define the median $\text{med}\,(\mathcal{D})$ of a tietoaineisto $\mathcal{D} = \{x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}\}$ via a specific satunnaismuuttuja $\tilde{x}$ that is naturally associated with $\mathcal{D}$. In particular, this satunnaismuuttuja is constructed by $\tilde{x} = x^{(I)}$, with

the index I being chosen uniformly at random from the set $\{1, \ldots, m\}$, i.e., $\mathbb{P}(I = r) = 1/m$ for all $r = 1, \ldots, m$. If the satunnaismuuttuja x is integrable, a median of x is the solution of the following optimization problem:

$$\min_{x' \in \mathbb{R}} \mathbb{E}|x - x'|.$$

Like the mean, the median of a tietoaineisto $\mathcal{D}$ can also be used to estimate parameters of an underlying todennäköisyysmalli. Compared with the mean, the median is more robust to outliert. For example, a median of a tietoaineisto $\mathcal{D}$ with more than one data point does not change even if we arbitrarily increase the largest element of $\mathcal{D}$ (see Fig. 47). In contrast, the mean will increase arbitrarily.

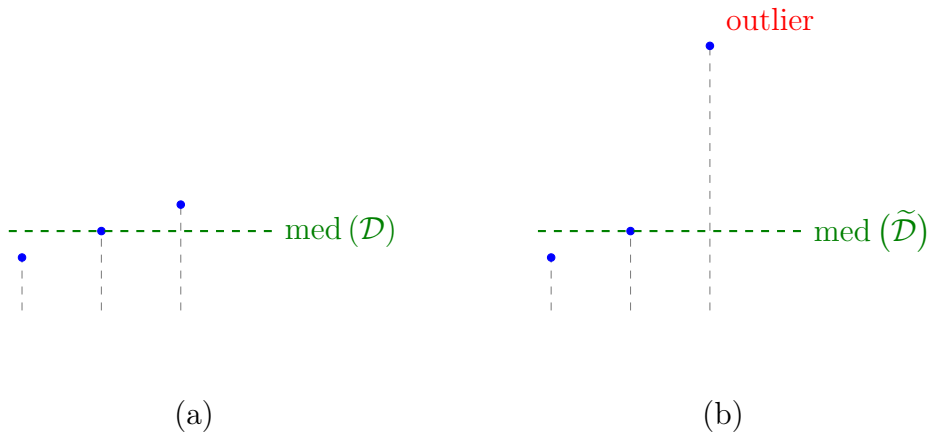


Fig. 47. The median is robust against outlier contamination. (a) Original tietoaineisto $\mathcal{D}$. (b) Noisy tietoaineisto $\widetilde{\mathcal{D}}$ including an outlier.

See also: mean, outlier, robustness.

metric In its most general form, a metric is a quantitative measure used to

compare or evaluate objects. In mathematics, a metric measures the distance between two points and must follow specific rules, i.e., the distance is always nonnegative, zero only if the points are the same, symmetric, and it satisfies the triangle inequality [2]. In koneoppiminen, a metric is a quantitative measure of how well a malli performs. Examples include tarkkuus, precision, and the average 0/1 loss on a test set [**?**], [**?**]. A häviöfunktio is used to train malli, while a metric is used to compare trained malli.

See also: koneoppiminen, malli, tarkkuus, 0/1 loss, test set, häviöfunktio, häviö, mallin valinta.

minimum Given a set of real numbers, the minimum is the smallest of those numbers. Note that for some sets, such as the set of negative real numbers, the minimum does not exist.

missing data Consider a tietoaineisto constituted by data points collected via some physical device. Due to imperfections and failures, some of the piirre or nimiö values of data points might be corrupted or simply missing. Data imputation aims to estimate these missing values [**?**]. We can interpret data imputation as an koneoppiminen problem where the nimiö of a data point is the value of the corrupted piirre.

See also: piirre, nimiö.

model inversion A malli inversion is a form of privacy attack on an koneoppiminen system. An adversary seeks to infer sensitive attributes of individual data points by exploiting partial access to a trained malli $\hat{h} \in \mathcal{H}$. This access typically consists of querying the malli for ennuste

$\hat{h}(\mathbf{x})$ using carefully chosen inputs. Basic malli inversion techniques have been demonstrated in the context of facial image luokittelu, where images are reconstructed using the (gradientti of) malli outputs combined with auxiliary information such as a person's name [**?**] (see Fig. 48).

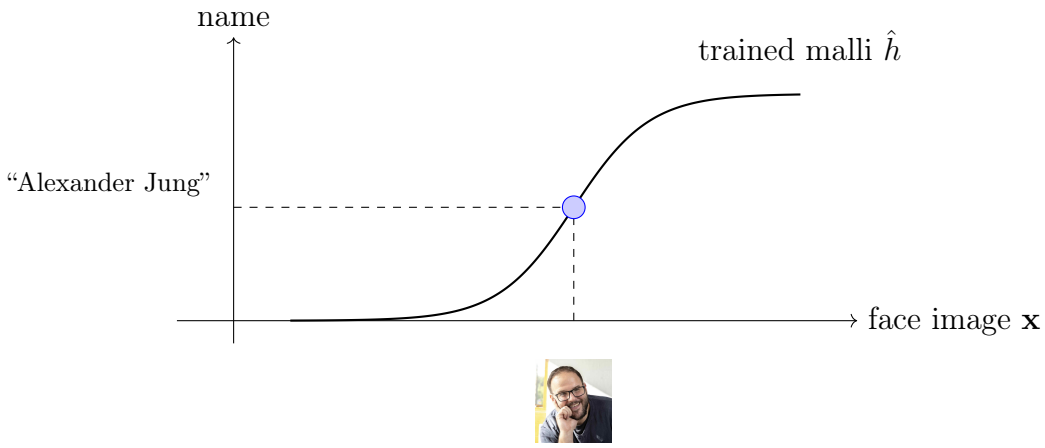


Fig. 48. Model inversion techniques implemented in the context of facial image classification.

See also: malli, privacy attack, koneoppiminen, sensitive attribute, data point, ennuste, luokittelu, gradientti, trustworthy AI, privacy protection.

model parameters Malli parameters are quantities that are used to select a specific hypothesis map from a malli. We can think of a list of malli parameters as a unique identifier for a hypothesis map, similar to how a social security number identifies a person in Finland.

See also: malli, parameter, hypothesis, map.

multi-label classification Multi-nimiö luokittelu problems and methods

use data points that are characterized by several nimiöt. As an example, consider a data point representing a picture with two nimiöt. One nimiö indicates the presence of a human in this picture and another nimiö indicates the presence of a car.

See also: nimiö, luokittelu, data point.

multiarmed bandit (MAB) An MAB problem is a precise mathematical formulation of a sequential decision-making task under epävarmuus. At each discrete time step k, a learner selects one of several possible actions—called arms—from a finite set $\mathcal{A}$. Pulling arm a at time k yields a reward $r^{(a,k)}$ that is drawn from an unknown todennäköisyysjakauma $\mathbb{P}(r^{(a,k)})$. We obtain different classes of MAB problems by placing different restrictions on this todennäköisyysjakauma. In the simplest setting, the todennäköisyysjakauma $\mathbb{P}(r^{(a,k)})$ does not depend on t. Given an MAB problem, the goal is to construct koneoppiminen methods that maximize the cumulative reward over time by strategically balancing exploration (i.e., gathering information about uncertain arms) and exploitation (i.e., selecting arms known to perform well). MAB problems form an important special case of RL problems [?], [?].

See also: reward, regret.

multitask learning Multitask learning aims to leverage relations between different learning tasks. Consider two learning tasks obtained from the same tietoaineisto of webcam snapshots. The first task is to predict the presence of a human, while the second task is to predict the presence of a car. It may be useful to use the same deep net structure for both tasks

and only allow the weights of the final output layer to be different.

See also: learning task, tietoaineisto, deep net, weights, layer.

multivariate normal distribution The multivariate normal distribution, which is denoted by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, is a fundamental todennäköisyysmalli for numerical piirrevektori of fixed dimension d. It defines a family of todennäköisyysjakaumat over vector-valued satunnaismuuttujat $\mathbf{x} \in \mathbb{R}^d$ [7], [**?**], [**?**]. Each distribution in this family is fully specified by its mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and kovarianssimatriisi $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$. When the kovarianssimatriisi $\boldsymbol{\Sigma}$ is invertible, the corresponding todennäköisyysjakauma is characterized by the following pdf:

$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{\Sigma})}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right].$$

Note that this pdf is only defined when $\boldsymbol{\Sigma}$ is invertible. More generally, any satunnaismuuttuja $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ admits the following representation:

$$\mathbf{x} = \mathbf{A}\mathbf{z} + \boldsymbol{\mu}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is a standard normal vector and $\mathbf{A} \in \mathbb{R}^{d \times d}$ satisfies $\mathbf{A}\mathbf{A}^\top = \boldsymbol{\Sigma}$. This representation remains valid even when $\boldsymbol{\Sigma}$ is singular, in which case $\mathbf{A}$ is not full rank [**?**, Ch. 23]. The family of multivariate normal distributions is exceptional among todennäköisyysmalli for numerical quantities, at least for the following reasons. First, the family is closed under affine transformations, i.e.,

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \text{ implies } \mathbf{B}\mathbf{x}+\mathbf{c} \sim \mathcal{N}\big(\mathbf{B}\boldsymbol{\mu} + \mathbf{c}, \mathbf{B}\boldsymbol{\Sigma}\mathbf{B}^T\big).$$

Second, the todennäköisyysjakauma $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ maximizes the differential entropy among all distributions with the same kovarianssimatriisi $\boldsymbol{\Sigma}$ [**?**].

See also: todennäköisyysmalli, todennäköisyysjakauma, standard normal vector, differential entropy, Gaussian RV.

mutual information (MI) The MI $I(\mathbf{x}; y)$ between two satunnaismuuttujat $\mathbf{x}$, y defined on the same probability space is given by [**?**]

$$I(\mathbf{x}; y) := \mathbb{E}\left\{\log \frac{p(\mathbf{x}, y)}{p(\mathbf{x})p(y)}\right\}.$$

It is a measure of how well we can estimate y based solely on $\mathbf{x}$. A large value of $I(\mathbf{x}; y)$ indicates that y can be well predicted solely from $\mathbf{x}$. This ennuste could be obtained by a hypothesis learned by an ERM-based koneoppiminen method.

See also: satunnaismuuttuja, probability space, ennuste, hypothesis, ERM, koneoppiminen.

nearest neighbor (NN) NN methods learn a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ whose function value $h(\mathbf{x})$ is solely determined by the NNs within a given tietoaineisto. Different methods use different metrics for determining the NNs. If data points are characterized by numeric piirrevektori, we can use their Euclidean distances as the metric.

See also: hypothesis, function, tietoaineisto, metric, data point, piirrevektori, neighbors.

neighborhood Consider some metric space $\mathcal{X}$ with metric $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$. The neighborhood of a point $\mathbf{x} \in \mathcal{X}$ is the set of other points having a sufficiently small distance to $\mathbf{x}$. For example, the ϵ-neighborhood of $\mathbf{x}$ is defined as

$$\{\mathbf{x}' \in \mathcal{X} : d(\mathbf{x}, \mathbf{x}') \leq \epsilon\}.$$

If $\mathcal{X}$ is an undirected graph, which is a special case of a metric space, the neighborhood of a node $i \in \mathcal{V}$ is the set of its neighbors.

See also: neighbors, metric.

neighbors The neighbors of a node $i \in \mathcal{V}$ within an FL network are those nodes $i' \in \mathcal{V} \setminus \{i\}$ that are connected (via an edge) to node i.

See also: FL network.

neliövirhehäviö The squared error häviö measures the ennuste error of a hypothesis h when predicting a numeric nimiö $y \in \mathbb{R}$ from the piirre $\mathbf{x}$ of a data point. It is defined as

$$L\left((\mathbf{x}, y), h\right) := \big(y - \underbrace{h(\mathbf{x})}_{=\hat{y}}\big)^2.$$

See also: häviö, ennuste, hypothesis, nimiö, piirre, data point.

networked data Networked data consist of local datasets that are related by some notion of pairwise similarity. We can represent networked data using a graph whose nodes carry local datasets and whose edges encode pairwise similarities. An example of networked data can be found in federoitu oppiminen applications where local datasets are generated by spatially distributed devices.

See also: data, local dataset, graph, federoitu oppiminen, device.

networked exponential families (nExpFam) A collection of exponential families, each of them assigned to a node of an FL network. The model parameters are coupled via the network structure by requiring

them to have a small GTV [?].

See also: FL network, model parameters, GTV.

networked federated learning (NFL) NFL refers to methods that learn personalized malli in a distributed fashion. These methods learn from local datasets that are related by an intrinsic network structure.

See also: malli, local dataset, federoitu oppiminen.

networked model A networked malli over an FL network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ assigns a local model (i.e., a hypothesis space) to each node $i \in \mathcal{V}$ of the FL network $\mathcal{G}$.

See also: malli, FL network, local model, hypothesis space.

neuroverkko An ANN is a graphical (signal-flow) representation of a function that maps piirre of a data point at its input to a ennuste for the corresponding nimiö at its output. The fundamental unit of an ANN is the artificial neuron, which applies an activation function to its weighted inputs. The outputs of these neurons serve as inputs for other neurons, forming interconnected layers.

See also: function, piirre, data point, ennuste, nimiö, activation function, layer.

nimiö A higher-level fact or quantity of interest associated with a data point. For example, if the data point is an image, the label could indicate whether the image contains a cat or not. Synonyms for label, commonly used in specific domains, include "response variable," "output variable," and "target" [?], [?], [?].

See also: data point, label space.

node degree The degree $d^{(i)}$ of a node $i \in \mathcal{V}$ in an undirected graph is the number of its neighbors, i.e., $d^{(i)} := \left|\mathcal{N}^{(i)}\right|$.

See also: graph, neighbors.

non-smooth We refer to a function as non-smooth if it is not smooth [?].

See also: function, smooth.

norm A norm is a function that maps each (vector) element of a vector space to a nonnegative real number. This function must be homogeneous and definite, and it must satisfy the triangle inequality [?].

See also: function, vector, vector space.

nullspace The nullspace of a matriisi $\mathbf{A} \in \mathbb{R}^{d' \times d}$, denoted by null($\mathbf{A}$), is the set of all vectors $\mathbf{n} \in \mathbb{R}^{d}$ such that

$$\mathbf{A}\mathbf{n} = \mathbf{0}.$$

Consider a feature learning method that uses the matriisi $\mathbf{A}$ to transform a piirevektori $\mathbf{x} \in \mathbb{R}^{d}$ of a data point into a new piirevektori $\mathbf{z} = \mathbf{A}\mathbf{x} \in \mathbb{R}^{d'}$. The nullspace null($\mathbf{A}$) characterizes all directions in the original feature space $\mathbb{R}^{d}$ along which the transformation $\mathbf{A}\mathbf{x}$ remains unchanged. In other words, adding any vector from the nullspace to a piirevektori $\mathbf{x}$ does not affect the transformed representation $\mathbf{z}$. This property can be exploited to enforce invariances in the ennuste (computed from $\mathbf{A}\mathbf{x}$). Fig. 49 illustrates one such invariance. It shows rotated versions of two handwritten digits, which approximately lie along 1-D curves in the original feature space. These curves are aligned with a direction vector $\mathbf{n} \in \mathbb{R}^{d}$. To ensure that the trained malli is invariant to such rotations,

we can choose the transformation matriisi $\mathbf{A}$ such that $\mathbf{n} \in \text{null}(\mathbf{A})$. This ensures that $\mathbf{Ax}$, and hence the resulting ennuste, is approximately insensitive to rotations of the input image.

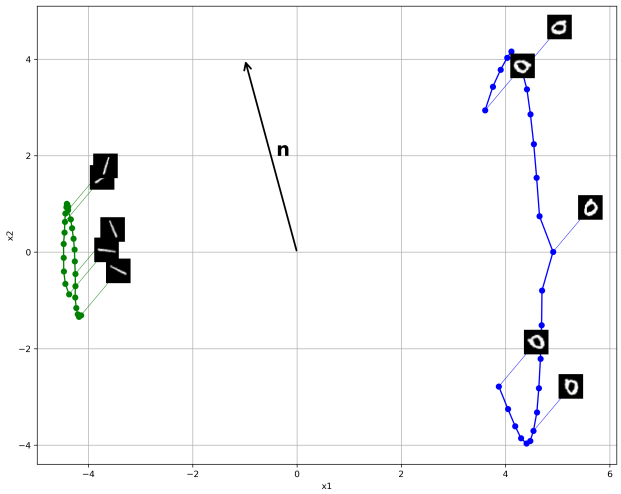


Fig. 49. Rotated handwritings of two different digits. The rotations are approximately aligned along straight lines parallel to the vector $\mathbf{n}$. For a binary luokitin distinguishing between these digits, a natural choice is a linear feature map $\mathbf{x} \mapsto \mathbf{Ax}$ with a matriisi $\mathbf{A}$ whose nullspace contains $\mathbf{n}$, i.e., $\mathbf{n} \in \text{null}(\mathbf{A})$.

See also: matriisi, feature map, feature learning.

Python demo: click me

odotusarvon maksimointi Consider a todennäköisyysmalli $\mathbb{P}(\mathbf{z}; \mathbf{w})$ for the data points $\mathcal{D}$ generated in some koneoppiminen application. The suurimman uskottavuuden menetelmä estimator for the model parameters $\mathbf{w}$ is obtained by maximizing $\mathbb{P}(\mathcal{D}; \mathbf{w})$. However, the resulting optimiza-

tion problem might be computationally challenging. EM approximates the suurimman uskottavuuden menetelmä estimator by introducing a latent satunnaismuuttuja $\mathbf{z}$ such that maximizing $\mathbb{P}(\mathcal{D}, \mathbf{z}; \mathbf{w})$ would be easier [?], [?], [?]. Since we do not observe $\mathbf{z}$, we need to estimate it from the observed tietoaineisto $\mathcal{D}$ using a conditional expectation. The resulting estimate $\widehat{\mathbf{z}}$ is then used to compute a new estimate $\widehat{\mathbf{w}}$ by solving $\max_{\mathbf{w}} \mathbb{P}(\mathcal{D}, \widehat{\mathbf{z}}; \mathbf{w})$. The crux is that the conditional expectation $\widehat{\mathbf{z}}$ depends on the model parameters $\widehat{\mathbf{w}}$, which we have updated based on $\widehat{\mathbf{z}}$. Thus, we have to recalculate $\widehat{\mathbf{z}}$, which, in turn, results in a new choice $\widehat{\mathbf{w}}$ for the model parameters. In practice, we repeat the computation of the conditional expectation (i.e., the E-step) and the update of the model parameters (i.e., the M-step) until some stopping criterion is met. See also: todennäköisyysmalli, suurimman uskottavuuden menetelmä, optimization problem.

online gradient descent (online GD) Consider an koneoppiminen method that learns model parameters $\mathbf{w}$ from some parameter space $\mathcal{W} \subseteq \mathbb{R}^d$. The learning process uses data points $\mathbf{z}^{(t)}$ that arrive at consecutive time instants $t = 1, 2, \ldots$. Let us interpret the data points $\mathbf{z}^{(t)}$ as i.i.d. copies of an satunnaismuuttuja $\mathbf{z}$. The risk $\mathbb{E}\{L(\mathbf{z}, \mathbf{w})\}$ of a hypothesis $h^{(\mathbf{w})}$ can then (under mild conditions) be obtained as the limit $\lim_{T\to\infty}(1/T)\sum_{t=1}{}^{T} L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$. We might use this limit as the kohdefunktiot for learning the model parameters $\mathbf{w}$. Unfortunately, this limit can only be evaluated if we wait infinitely long in order to collect all data points. Some koneoppiminen applications require methods that learn online, i.e., as soon as a new data point $\mathbf{z}^{(t)}$ arrives at time t,

we update the current model parameters $\mathbf{w}^{(t)}$. Note that the new data point $\mathbf{z}^{(t)}$ contributes the component $L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$ to the risk. As its name suggests, online GD updates $\mathbf{w}^{(t)}$ via a (projected) gradient step such that

$$\mathbf{w}^{(t+1)} := P_{\mathcal{W}}\big(\mathbf{w}^{(t)} - \eta_t \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)\big). \tag{9}$$

Note that (9) is a gradient step for the current component $L\left(\mathbf{z}^{(t)}, \cdot\right)$ of the risk. The update (9) ignores all previous components $L\left(\mathbf{z}^{(t')}, \cdot\right)$, for $t' < t$. It might therefore happen that, compared with $\mathbf{w}^{(t)}$, the updated model parameters $\mathbf{w}^{(t+1)}$ increase the retrospective average häviö $\sum_{t'=1}^{t-1} L\left(\mathbf{z}^{(t')}, \cdot\right)$. However, for a suitably chosen oppimisnopeus η_t, online GD can be shown to be optimal in practically relevant settings. By optimal, we mean that the model parameters $\mathbf{w}^{(T+1)}$ delivered by online GD after observing T data points $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(T)}$ are at least as good as those delivered by any other learning method [?], [?].

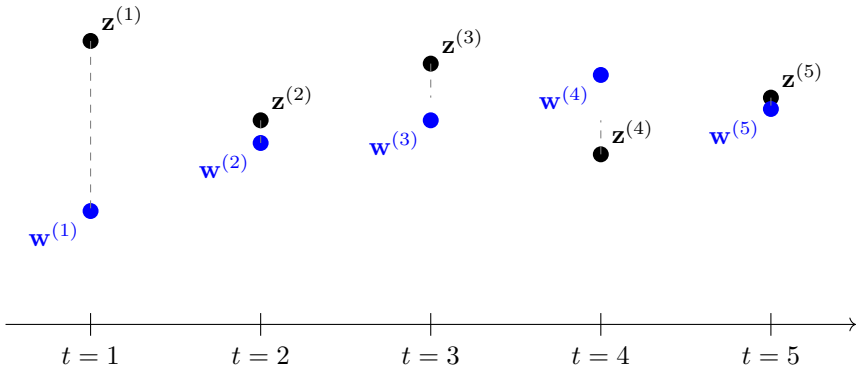


Fig. 50. An instance of online GD that updates the model parameters $\mathbf{w}^{(t)}$ using the data point $\mathbf{z}^{(t)} = x^{(t)}$ arriving at time t. This instance uses the neliövirhehäviö $L\left(\mathbf{z}^{(t)}, w\right) = (x^{(t)} - w)^2$.

See also: kohdefunktiot, GD, gradient step, online learning.

online learning Some koneoppiminen methods are designed to process data in a sequential manner, updating their model parameters one at a time, as new data points become available. A typical example is time-series data, such as daily minimum and maximum temperatures recorded by an FMI weather station. These values form a chronological sequence of observations. During each time step t, online learning methods update (or refine) the current hypothesis $h^{(t)}$ (or model parameters $\mathbf{w}^{(t)}$) based on the newly observed data point $\mathbf{z}^{(t)}$.

See also: online gradient descent (online GD), online-algoritmi.

online-algoritmi An online algorithm processes input data incrementally, receiving data points sequentially and making decisions or producing outputs (or decisions) immediately without having access to the entire input in advance [**?**], [**?**]. Unlike an offline algorithm, which has the entire input available from the start, an online algorithm must handle epävarmuus about future inputs and cannot revise past decisions. Similar to an offline algorithm, we represent an online algorithm formally as a collection of possible executions. However, the execution sequence for an online algorithm has a distinct structure as follows:

$$\text{in}_1,\ s_1,\ \text{out}_1,\ \text{in}_2,\ s_2,\ \text{out}_2,\ \ldots,\ \text{in}_T,\ s_T,\ \text{out}_T.$$

Each execution begins from an initial state (i.e., in_1) and proceeds through alternating computational steps, outputs (or decisions), and inputs. Specifically, at step k, the algorithm performs a computational step s_k, generates an output out_k, and then subsequently receives the

next input (data point) in_{k+1}. A notable example of an online algorithm in koneoppiminen is online GD, which incrementally updates model parameters as new data points arrive.

See also: algorithm, data, data point, epävarmuus, koneoppiminen, online GD, model parameters, online learning.

opetusvirhe The average häviö of a hypothesis when predicting the nimiöt of the data points in a training set. We sometimes also refer to training error as the minimal average häviö that is achieved by a solution of ERM.

See also: häviö, hypothesis, nimiö, data point, training set, ERM.

oppimisnopeus Consider an iterative koneoppiminen method for finding or learning a useful hypothesis $h \in \mathcal{H}$. Such an iterative method repeats similar computational (update) steps that adjust or modify the current hypothesis to obtain an improved hypothesis. One well-known example of such an iterative learning method is GD and its variants, SGD and projected gradient descent (projected GD). A key parameter of an iterative method is the learning rate. The learning rate controls the extent to which the current hypothesis can be modified during a single iteration. A well-known example of such a parameter is the step size used in GD [8, Ch. 5].

See also: koneoppiminen, hypothesis, GD, SGD, projected GD, parameter, step size.

optimism in the face of uncertainty koneoppiminen methods learn model parameters $\mathbf{w}$ according to some performance criterion $\bar{f}(\mathbf{w})$. Howe-

ver, they usually cannot access $\bar{f}(\mathbf{w})$ directly but rely on an estimate (or approximation) $f(\mathbf{w})$ of $\bar{f}(\mathbf{w})$. As a case in point, ERM-based methods use the average häviö on a given tietoaineisto (i.e., the training set) as an estimate for the risk of a hypothesis. Using a todennäköisyysmalli, one can construct a confidence interval $\left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$ for each choice $\mathbf{w}$ for the model parameters. One simple construction is $l^{(\mathbf{w})} := f(\mathbf{w}) - \sigma/2$, $u^{(\mathbf{w})} := f(\mathbf{w}) + \sigma/2$, with σ being a measure of the (expected) deviation of $f(\mathbf{w})$ from $\bar{f}(\mathbf{w})$. We can also use other constructions for this interval as long as they ensure that $\bar{f}(\mathbf{w}) \in \left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$ with a sufficiently high probability. An optimist chooses the model parameters according to the most favorable—yet still plausible—value $\tilde{f}(\mathbf{w}) := l^{(\mathbf{w})}$ of the performance criterion (see Fig. 51). Two examples of koneoppiminen methods that use such an optimistic construction of an kohdefunktiot are SRM [**?**, Ch. 11] and upper confidence bound (UCB) methods for sequential decision making [**?**, Sec. 2.2].

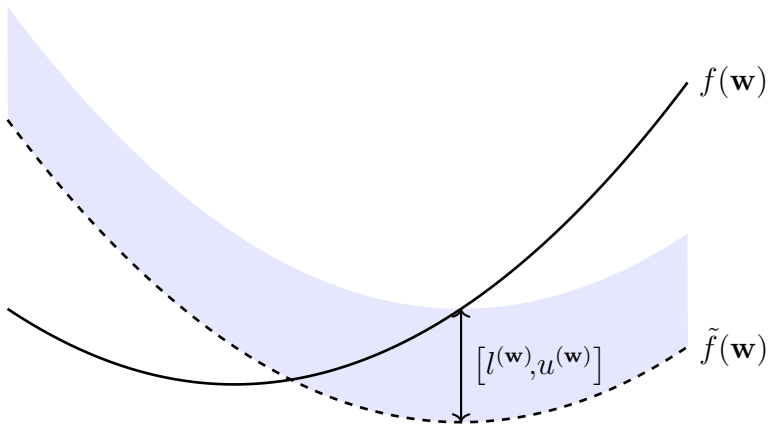


Fig. 51. koneoppiminen methods learn model parameters $\mathbf{w}$ by using some estimate of $f(\mathbf{w})$ for the ultimate performance criterion $\bar{f}(\mathbf{w})$. Using a todennäköisyysmalli, one can use $f(\mathbf{w})$ to construct confidence intervals $\left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$, which contain $\bar{f}(\mathbf{w})$ with a high probability. The best plausible performance measure for a specific choice $\mathbf{w}$ of model parameters is $\tilde{f}(\mathbf{w}) := l^{(\mathbf{w})}$.

See also: koneoppiminen, model parameters, ERM, häviö, tietoaineisto, training set, risk, hypothesis, todennäköisyysmalli, probability, kohdefunktiot, SRM, UCB.

optimization method An optimization method is an algorithm that reads in a representation of an optimization problem and delivers an (approximate) solution as its output [**?**], [**?**], [**?**].

See also: algorithm, optimization problem.

osittava klusterointi Hard klusterointi refers to the task of partitioning a given set of data points into (a few) nonoverlapping ryppäät. The most widely used hard klusterointi method is k-means.

See also: klusterointi, data point, rypäs, k-means.

outlier Many koneoppiminen methods are motivated by the i.i.d. assumption, which interprets data points as realizations of i.i.d. satunnaismuuttujat with a common todennäköisyysjakauma. The i.i.d. assumption is useful for applications where the statistical properties of the data generation process are stationary (or time-invariant) [**?**]. However, in some applications, the data consist of a majority of regular data points that conform with the i.i.d. assumption as well as a small number of data points that have fundamentally different statistical properties compared with the regular data points. We refer to a data point that substantially deviates from the statistical properties of most data points as an outlier. Different methods for outlier detection use different measures for this deviation. Statistical learning theory studies fundamental limits on the ability to mitigate outliers reliably [**?**], [**?**].

See also: robustness, stability, Huber regression, todennäköisyysmalli.

parameter The parameter of an koneoppiminen malli is a tunable (i.e., learnable or adjustable) quantity that allows us to choose between different hypothesis maps. For example, the linear model $\mathcal{H} := \{h^{(\mathbf{w})} : h^{(\mathbf{w})}(x) = w_1 x + w_2\}$ consists of all hypothesis maps $h^{(\mathbf{w})}(x) = w_1 x + w_2$ with a particular choice for the parameters $\mathbf{w} = \big(w_1, w_2\big)^T \in \mathbb{R}^2$. Another example of a malli parameter is the weights assigned to a connection between two neurons of an ANN.

See also: koneoppiminen, malli, hypothesis, map, linear model, weights, ANN.

parameter space The parameter space $\mathcal{W}$ of an koneoppiminen malli $\mathcal{H}$ is the set of all feasible choices for the model parameters (see Fig. 52). Many important koneoppiminen methods use a malli that is parameterized by vectors of the Euclidean space $\mathbb{R}^d$. Two widely used examples of parameterized malli are linear models and deep nets. The parameter space is then often a subset $\mathcal{W} \subseteq \mathbb{R}^d$, e.g., all vectors $\mathbf{w} \in \mathbb{R}^d$ with a norm smaller than one.

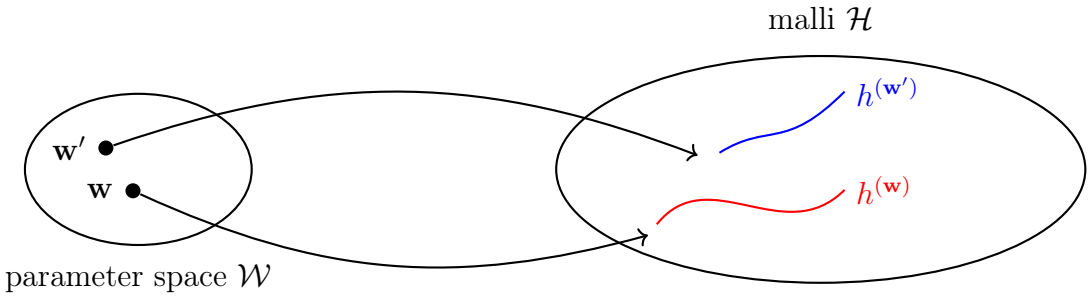


Fig. 52. The parameter space $\mathcal{W}$ of an koneoppiminen malli $\mathcal{H}$ consists of all feasible choices for the model parameters. Each choice $\mathbf{w}$ for the model parameters selects a hypothesis map $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: parameter, malli, model parameters.

parametric model A parametric malli $\mathcal{H}$ is a malli that is parameterized by a finite number of model parameters. In particular, each hypothesis $h \in \mathcal{H}$ is uniquely identified by a list of model parameters $w_1, w_2, \ldots$ (see Fig. 52). For many important koneoppiminen methods , this list has a fixed length d which we refer to as the number of model parameters. We then stack the model parameters into a vector $\mathbf{w} \in \mathbb{R}^d$. Two widely

used examples of a paramtric malli are a linear model and a neuroverkot. The parameter space is then often a subset $\mathcal{W} \subseteq \mathbb{R}^d$, e.g., all vectors $\mathbf{w} \in \mathbb{R}^d$ with a norm smaller than one.

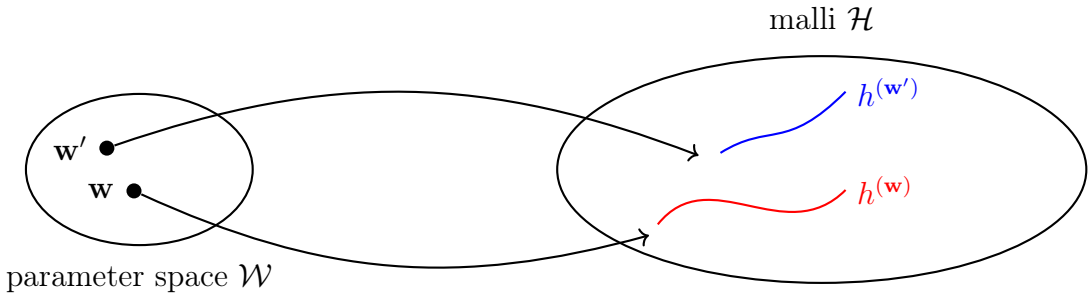


Fig. 53. The parameter space $\mathcal{W}$ of an koneoppiminen malli $\mathcal{H}$ consists of all feasible choices for the model parameters. Each choice $\mathbf{w}$ for the model parameters selects a hypothesis map $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: parameter space, malli, model parameters.

piirre A feature of a data point is one of its properties that can be measured or computed easily without the need for human supervision. For example, if a data point is a digital image (e.g., stored as a `.jpeg` file), then we could use the red–green–blue (RGB) intensities of its pixels as features. Another example is shown in Fig. 54, where the the signal samples of a finite-duration audio signal are used as its features. Domain-specific synonyms for the term feature are "covariate," "explanatory variable," "independent variable," "input (variable)," "predictor (variable)," or "regressor" [?], [?], [?].

See also: data point.

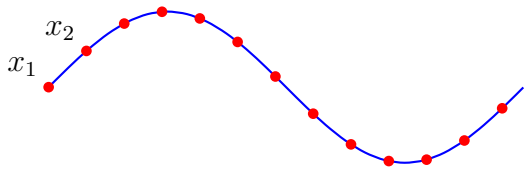


Fig. 54. An audio signal (blue waveform) and its discretized signal samples (red dots) which can be used as its features $x_1, \ldots, x_d$.

piirrevektori Piirre vector refers to a vector $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ whose entries are individual piirre $x_1, \ldots, x_d$. Many koneoppiminen methods use piirre vectors that belong to some finite-dimensional Euclidean space $\mathbb{R}^d$. For some koneoppiminen methods, however, it can be more convenient to work with piirre vectors that belong to an infinite-dimensional vector space (e.g., see kernel method).

See also: piirre, vector, koneoppiminen, Euclidean space, vector space.

polynomial regression Polynomial regressio is an instance of ERM that learns a polynomial hypothesis map to predict a numeric nimiö based on the numeric piirre of a data point. For data points characterized by a single numeric piirre, polynomial regressio uses the hypothesis space $\mathcal{H}_d^{(\text{poly})} := \{h(x) = \sum_{j=0}^{d-1} x^j w_j\}$. The quality of a polynomial hypothesis map is measured using the average neliövirhehäviö incurred on a set of labeled data points (which we refer to as the training set).

See also: regressio, ERM, neliövirhehäviö.

positive semi-definite (psd) A (real-valued) symmetric matriisi $\mathbf{Q} = \mathbf{Q}^T \in \mathbb{R}^{d \times d}$ is referred to as psd if $\mathbf{x}^T \mathbf{Q} \mathbf{x} \geq 0$ for every vector $\mathbf{x} \in \mathbb{R}^d$. The property of being psd can be extended from matriisit to (real-valued)

symmetric ydinfunktio maps $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (with $K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x})$) as follows: For any finite set of piirrevektori $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$, the resulting matriisi $\mathbf{Q} \in \mathbb{R}^{m \times m}$ with entries $Q_{r,r'} = K\big(\mathbf{x}^{(r)}, \mathbf{x}^{(r')}\big)$ is psd [**?**].

See also: matriisi, vector, ydinfunktio, map, piirevektori.

preimage Consider a function $f : \mathcal{U} \to \mathcal{V}$ between two sets. The preimage $f^{-1}(\mathcal{B})$ of a subset $\mathcal{B} \subseteq \mathcal{V}$ is the set of all inputs $u \in \mathcal{U}$ that are mapped into $\mathcal{B}$ by f, i.e.,

$$f^{-1}(\mathcal{B}) := \{u \in \mathcal{U} \mid f(u) \in \mathcal{B}\}.$$

The preimage is well defined even if the function f is non-invertible [2].

See also: function.

principal component analysis (PCA) PCA determines a linear feature map such that the new piirre allow us to reconstruct the original piirre with the minimum reconstruction error [8].

See also: feature map, piirre, minimum.

privacy attack A privacy attack on an koneoppiminen system aims to infer sensitive attributes of individuals by exploiting partial access to a trained koneoppiminen malli. One form of a privacy attack is model inversion.

See also: attack, sensitive attribute, model inversion, trustworthy AI, general data protection regulation (GDPR).

privacy funnel The privacy funnel is a method for learning privacy-friendly piirre of data points [**?**].

See also: piirre, data point.

privacy leakage Consider an koneoppiminen application that processes a tietoaineisto $\mathcal{D}$ and delivers some output, such as the ennuste obtained for new data points. Privacy leakage arises if the output carries information about a private (or sensitive) piirre of a data point of $\mathcal{D}$ (such as a human). Based on a todennäköisyysmalli for the data generation, we can measure the privacy leakage via the MI between the output and the sensitive piirre. Another quantitative measure of privacy leakage is DP. The relations between different measures of privacy leakage have been studied in the literature (see [**?**]).

See also: MI, DP, privacy attack, GDPR.

privacy protection Consider some koneoppiminen method $\mathcal{A}$ that reads in a tietoaineisto $\mathcal{D}$ and delivers some output $\mathcal{A}(\mathcal{D})$. The output could be the learned model parameters $\widehat{\mathbf{w}}$ or the ennuste $\hat{h}(\mathbf{x})$ obtained for a specific data point with piirre $\mathbf{x}$. Many important koneoppiminen applications involve data points representing humans. Each data point is characterized by piirre $\mathbf{x}$, potentially a nimiö y, and a sensitive attribute s (e.g., a recent medical diagnosis). Roughly speaking, privacy protection means that it should be impossible to infer, from the output $\mathcal{A}(\mathcal{D})$, any of the sensitive attributes of data points in $\mathcal{D}$. Mathematically, privacy protection requires non-invertibility of the map $\mathcal{A}(\mathcal{D})$. In general, just making $\mathcal{A}(\mathcal{D})$ non-invertible is typically insufficient for privacy protection. We need to make $\mathcal{A}(\mathcal{D})$ sufficiently non-invertible.

See also: koneoppiminen, tietoaineisto, model parameters, ennuste, data point, piirre, nimiö, sensitive attribute, map.

probabilistic principal component analysis (PPCA) PPCA extends basic PCA by using a todennäköisyysmalli for data points. The todennäköisyysmalli of PPCA frames the task of ulottuvuuksien vähentäminen as an estimation problem that can be solved using odotusarvon maksimointi [**?**].

See also: PCA, todennäköisyysmalli, ulottuvuuksien vähentäminen, odotusarvob maksimointi.

probability We assign a probability value, typically chosen in the interval $[0, 1]$, to each event that can occur in a satunnaiskoe [6], [7], [**?**], [**?**].

See also: event, satunnaiskoe.

probability density function (pdf) The pdf $p(x)$ of a real-valued satunnaismuuttuja $x \in \mathbb{R}$ is a particular representation of its todennäköisyysjakauma. If the pdf exists, it can be used to compute the probability that x takes on a value from a measurable set $\mathcal{B} \subseteq \mathbb{R}$ via $\mathbb{P}(x \in \mathcal{B}) = \int_{\mathcal{B}} p(x')dx'$ [7, Ch. 3]. If the pdf of a vector-valued satunnaismuuttuja $\mathbf{x} \in \mathbb{R}^d$ exists, it allows us to compute the probability of $\mathbf{x}$ belonging to a measurable region $\mathcal{R}$ via $\mathbb{P}(\mathbf{x} \in \mathcal{R}) = \int_{\mathcal{R}} p(\mathbf{x}')dx'_1 \ldots dx'_d$ [7, Ch. 3].

See also: satunnaismuuttuja, todennäköisyysjakauma, probability, measurable, vector.

probability space A probability space is a mathematical structure that allows us to reason about a satunnaiskoe, e.g., the observation of a physical phenomenon. Formally, a probability space $\mathcal{P}$ is a triplet $(\Omega, \mathcal{F}, \mathbb{P}(\cdot))$ where

- Ω is a sample space containing all possible outcomes of a satunnaiskoe;
- $\mathcal{F}$ is a σ-algebra, i.e., a collection of subsets of Ω (called events) that satisfies certain closure properties under set operations;
- $\mathbb{P}(\cdot)$ is a todennäköisyysjakauma, i.e., a function that assigns a probability $P(\mathcal{A}) \in [0, 1]$ to each event $\mathcal{A} \in \mathcal{F}$. This function must satisfy $\mathbb{P}(\Omega) = 1$ and $\mathbb{P}(\bigcup_{i=1}^{\infty} \mathcal{A}_i) = \sum_{i=1}^{\infty} \mathbb{P}(\mathcal{A}_i)$ for any countable sequence of pairwise disjoint events $\mathcal{A}_1$, $\mathcal{A}_2$, ... in $\mathcal{F}$.

Probability spaces provide the foundation of todennäköisyysmalli that can be used to study the behavior of koneoppiminen methods [6], [**?**], [**?**]. See also: probability, satunnaiskoe, sample space, event, todennäköisyysjakauma, function, todennäköisyysmalli, koneoppiminen.

projected gradient descent (projected GD) Consider an ERM-based method that uses a parameterized malli with parameter space $\mathcal{W} \subseteq \mathbb{R}^d$. Even if the kohdefunktiot of ERM is smooth, we cannot use basic GD, as it does not take into account constraints on the optimization variable (i.e., the model parameters). Projected GD extends basic GD to address this issue. A single iteration of projected GD consists of first taking a gradient step and then projecting the result back onto the parameter space. See Fig. 55 for a visual illustration.

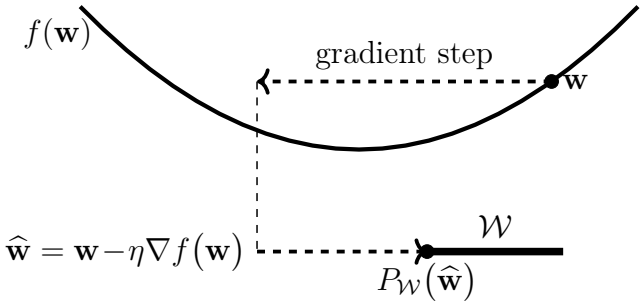


Fig. 55. Projected GD augments a basic gradient step with a projection back onto the constraint set $\mathcal{W}$.

See also: ERM, malli, parameter space, kohdefunktiot, smooth, GD, model parameters, gradient step, projection.

projection Consider a subset $\mathcal{W} \subseteq \mathbb{R}^d$ of the d-dimensional Euclidean space. We define the projection $P_{\mathcal{W}}(\mathbf{w})$ of a vector $\mathbf{w} \in \mathbb{R}^d$ onto $\mathcal{W}$ as

$$P_{\mathcal{W}}(\mathbf{w}) = \operatorname*{arg\,min}_{\mathbf{w}' \in \mathcal{W}} \|\mathbf{w} - \mathbf{w}'\|_2 .$$

In other words, $P_{\mathcal{W}}(\mathbf{w})$ is the vector in $\mathcal{W}$ that is closest to $\mathbf{w}$. The projection is only well defined for subsets $\mathcal{W}$ for which the above minimum exists [**?**].

See also: Euclidean space, vector, minimum.

proximable A convex function for which the proximal operator can be computed efficiently is sometimes referred to as proximable or simple [**?**].

See also: convex, function, proximal operator.

proximal operator Given a convex function $f(\mathbf{w}')$, we define its proximal

operator as [?], [?]

$$\mathbf{prox}_{f(\cdot),\rho}(\mathbf{w}) := \underset{\mathbf{w}' \in \mathbb{R}^d}{\arg\min} \left[f(\mathbf{w}') + \frac{\rho}{2} \|\mathbf{w} - \mathbf{w}'\|_2^2 \right] \text{ with } \rho > 0.$$

As illustrated in Fig. 56, evaluating the proximal operator amounts to minimizing a penalized variant of $f(\mathbf{w}')$. The penalty term is the scaled squared Euclidean distance to a given vector $\mathbf{w}$ (which is the input to the proximal operator). The proximal operator can be interpreted as a yleistys of the gradient step, which is defined for a smooth convex function $f(\mathbf{w}')$. Indeed, taking a gradient step with step size η at the current vector $\mathbf{w}$ is the same as applying the proximal operator of the function $\tilde{f}(\mathbf{w}') = \big(\nabla f(\mathbf{w})\big)^T(\mathbf{w}' - \mathbf{w})$ and using $\rho = 1/\eta$.

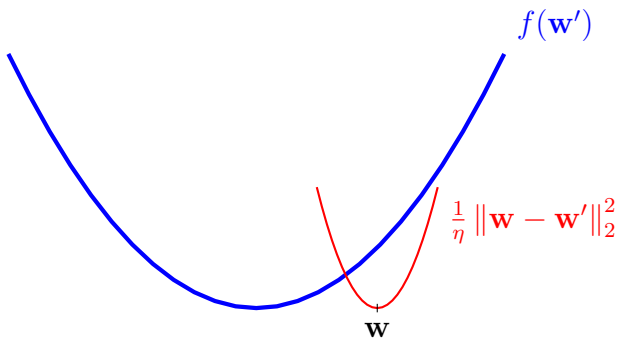


Fig. 56. The proximal operator updates a vector $\mathbf{w}$ by minimizing a penalized version of the function $f(\cdot)$. The penalty term is the scaled squared Euclidean distance between the optimization variable $\mathbf{w}'$ and the given vector $\mathbf{w}$.

See also: convex, function, vector, yleistys, gradient step, smooth, step size.

pseudokäänteisluku The Moore–Penrose pseudoinverse $\mathbf{A}^{+}$ of a matriisi $\mathbf{A} \in \mathbb{R}^{m \times d}$ generalizes the notion of an käänteismatriisi [3]. The pseudoinverse arises naturally within harjaregressio when applied to a tietoaineisto with arbitrary nimiöt $\mathbf{y}$ and a feature matrix $\mathbf{X} = \mathbf{A}$ [**?**, Ch. 3]. The model parameters learned by harjaregressio are given by

$$\widehat{\mathbf{w}}^{(\alpha)} = \left(\mathbf{A}^{T}\mathbf{A} + \alpha\mathbf{I}\right)^{-1}\mathbf{A}^{\top}\mathbf{y}, \quad \alpha > 0.$$

We can then define the pseudoinverse $\mathbf{A}^{+} \in \mathbb{R}^{d \times m}$ via the limit [**?**, Ch. 3]

$$\lim_{\alpha \to 0^{+}} \widehat{\mathbf{w}}^{(\alpha)} = \mathbf{A}^{+}\mathbf{y}.$$

See also: matriisi, käänteismatriisi, harjaregressio.

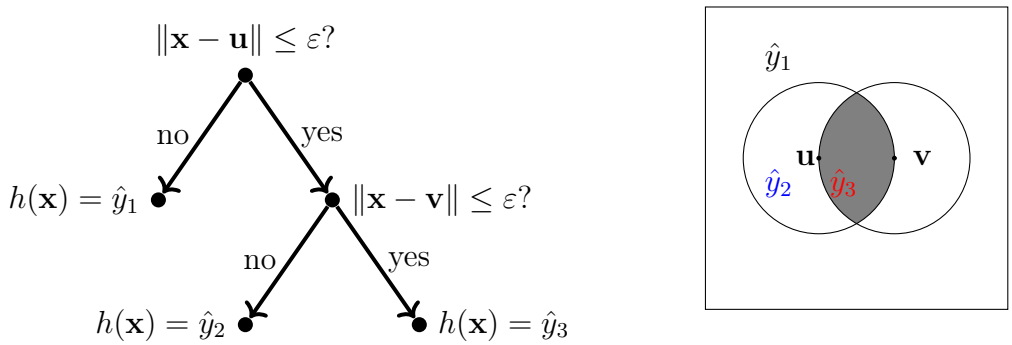
päätösalue Consider a hypothesis map h that delivers values from a finite set $\mathcal{Y}$. For each nimiö value (i.e., category) $a \in \mathcal{Y}$, the hypothesis h determines a subset of piirre values $\mathbf{x} \in \mathcal{X}$ that result in the same output $h(\mathbf{x}) = a$. We refer to this subset as a decision region of the hypothesis h.

See also: hypothesis, map, nimiö, piirre.

päätöspinta Consider a hypothesis map h that reads in a piirevektori $\mathbf{x} \in \mathbb{R}^{d}$ and delivers a value from a finite set $\mathcal{Y}$. The decision boundary of h is the set of vectors $\mathbf{x} \in \mathbb{R}^{d}$ that lie between different päätösalue. More precisely, a vector $\mathbf{x}$ belongs to the decision boundary if and only if each neighborhood $\{\mathbf{x}' : \|\mathbf{x} - \mathbf{x}'\| \leq \varepsilon\}$, for any $\varepsilon > 0$, contains at least two vectors with different function values.

See also: hypothesis, map, piirevektori, vector, päätösalue, neighborhood, function.

päätöspuu A decision tree is a flowchart-like representation of a hypothesis map h. More formally, a decision tree is a directed graph containing a root node that reads in the piirevektori $\mathbf{x}$ of a data point. The root node then forwards the data point to one of its child nodes based on some elementary test on the piirre $\mathbf{x}$. If the receiving child node is not a leaf node, i.e., it has child nodes itself, it represents another test. Based on the test result, the data point is forwarded to one of its descendants. This testing and forwarding of the data point is continued until the data point ends up in a leaf node without any children. See Fig. 57 for visual illustrations.



(a) (b)

Fig. 57. (a) A decision tree is a flowchart-like representation of a piecewise constant hypothesis $h : \mathcal{X} \to \mathbb{R}$. Each piece is a päätösalue $\mathcal{R}_{\hat{y}} := \big\{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = \hat{y}\big\}$. The depicted decision tree can be applied to numeric piirrevektori, i.e., $\mathcal{X} \subseteq \mathbb{R}^d$. It is parameterized by the threshold $\varepsilon > 0$ and the vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$. (b) A decision tree partitions the feature space $\mathcal{X}$ into päätösalue. Each päätösalue $\mathcal{R}_{\hat{y}} \subseteq \mathcal{X}$ corresponds to a specific leaf node in the decision tree.

See also: päätösalue.

kvadraattinen funktio A function $f : \mathbb{R}^d \to \mathbb{R}$ of the form

$$f(\mathbf{w}) = \mathbf{w}^T \mathbf{Q} \mathbf{w} + \mathbf{q}^T \mathbf{w} + a$$

with some matriisi $\mathbf{Q} \in \mathbb{R}^{d \times d}$, vector $\mathbf{q} \in \mathbb{R}^d$, and scalar $a \in \mathbb{R}$.

See also: function, matriisi, vector.

Rademacher complexity TBD.

See also: hypothesis space, yleistys, koneoppiminen, effective dimension, VC dimension.

satunnaiskoe A random experiment is a physical (or abstract) process that produces an outcome ω from a set of possibilities Ω. This set of all possible outcomes is referred to as the sample space of the experiment. The key characteristic of a random experiment is that its outcome is unpredictable (or uncertain). Any measurement or observation of the outcome is an satunnaismuuttuja, i.e., a function of the outcome $\omega \in \Omega$. Probability theory uses a probability space as a mathematical structure for the study of random experiments. A key conceptual property of a random experiment is that it can be repeated under identical conditions. Strictly speaking, repeating a random experiment a given number of m times defines a new random experiment. The outcomes of this new experiment are length-m sequences of outcomes from the original experiment (see Fig. 58). While the outcome of a single experiment is uncertain, the long-run behavior of the outcomes of repeated experiments tends to become increasingly predictable. This informal claim can be made precise via fundamental results of probability theory, such as the law of large numbers and the CLT.

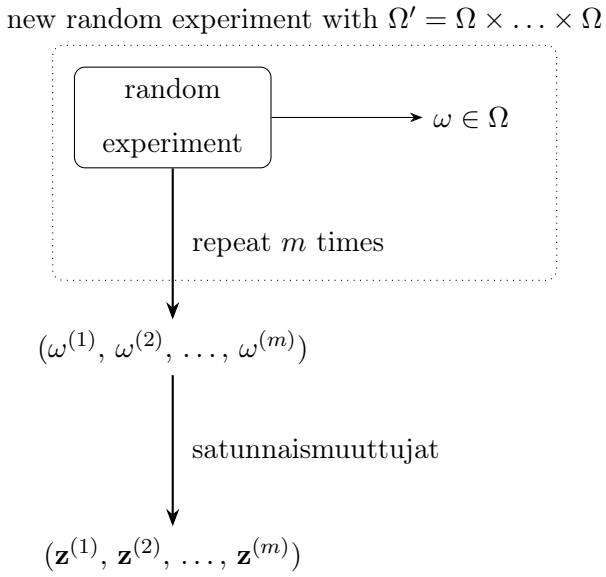


Fig. 58. A random experiment produces an outcome $\omega \in \Omega$ from a set of possibilities (or sample space) *ga*. Repeating the experiment m times yields another random experiment, whose outcomes are sequences $(\omega^{(1)}, \omega^{(2)}, \ldots, \omega^{(m)}) \in \Omega \times \ldots \times \Omega$. One example of a random experiment arising in many koneoppiminen applications is the gathering of a training set $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$.

Examples for random experiments arising in koneoppiminen applications include the following:

- Data collection: The data points collected in ERM-based methods can be interpreted as satunnaismuuttujat, i.e., as functions of the outcome $\omega \in \Omega$ of a random experiment.
- SGD uses a random experiment at each iteration to select a subset of the training set.

- Privacy protection methods use random experiments to generate noise that is added to the outputs of an koneoppiminen method to ensure DP.

See also: sample space, satunnaismuuttuja, function, probability, probability space, law of large numbers, CLT, sample space, koneoppiminen, training set, data, data point, ERM, SGD, privacy protection, DP.

realization Consider an satunnaismuuttuja $\mathbf{x}$ that maps each outcome $\omega \in \mathcal{P}$ of a probability space $\mathcal{P}$ to an element a of a measurable space $\mathcal{N}$ [2], [6], [**?**]. A realization of $\mathbf{x}$ is any element $\mathbf{a} \in \mathcal{N}$ such that there exists an element $\omega' \in \mathcal{P}$ with $\mathbf{x}(\omega') = \mathbf{a}$.

See also: satunnaismuuttuja, probability space, measurable.

rectified linear unit (ReLU) The ReLU is a popular choice for the activation function of a neuron within an ANN. It is defined as $\sigma(z) = \max\{0, z\}$, with z being the weighted input of the artificial neuron.

See also: activation function, ANN.

regressio Regression problems revolve around the ennuste of a numeric nimiö solely from the piirre of a data point [8, Ch. 2].

See also: ennuste, nimiö, piirre, data point.

regret The regret of a hypothesis h relative to another hypothesis h', which serves as a vertailutaso, is the difference between the häviö incurred by h and the häviö incurred by h' [**?**]. The vertailutaso hypothesis h' is also referred to as an expert.

See also: vertailutaso, häviö, expert.

regularisoija A regularizer assigns each hypothesis h from a hypothesis space $\mathcal{H}$ a quantitative measure $\mathcal{R}\big\{h\big\}$ conveying to what extent its ennuste errors might differ on data points on and outside a training set. Harjaregressio uses the regularizer $\mathcal{R}\big\{h\big\} := \|\mathbf{w}\|_2^2$ for linear hypothesis maps $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [8, Ch. 3]. Lasso uses the regularizer $\mathcal{R}\big\{h\big\} := \|\mathbf{w}\|_1$ for linear hypothesis maps $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [8, Ch. 3].
See also: harjaregressio, Lasso, häviö, kohdefunktiot.

regularisointi A key challenge of modern koneoppiminen applications is that they often use large malli, which have an effective dimension in the order of billions. Training a high-dimensional malli using basic ERM-based methods is prone to ylisovittaminen, i.e., the learned hypothesis performs well on the training set but poorly outside the training set. Regularization refers to modifications of a given instance of ERM in order to avoid ylisovittaminen, i.e., to ensure that the learned hypothesis does not perform much worse outside the training set. There are three routes for implementing regularization:

1) Malli pruning: We prune the original malli $\mathcal{H}$ to obtain a smaller malli $\mathcal{H}'$. For a parametric malli, the pruning can be implemented via constraints on the model parameters (such as $w_1 \in [0.4, 0.6]$ for the weight of piirre x_1 in lineaarinen regressio).

2) Häviö penalization: We modify the kohdefunktiot of ERM by adding a penalty term to the opetusvirhe. The penalty term estimates how much higher the expected häviö (or risk) is compared with the average häviö on the training set.

3) Data augmentation: We can enlarge the training set $\mathcal{D}$ by adding perturbed copies of the original data points in $\mathcal{D}$. One example for such a perturbation is to add the realization of an satunnaismuuttuja to the piirevektori of a data point.

Fig. 59 illustrates the above three routes to regularization. These routes are closely related and sometimes fully equivalent. Data augmentation using Gaussian RVs to perturb the piirrevektori in the training set of lineaarinen regressio has the same effect as adding the penalty $\lambda \|\mathbf{w}\|_2^2$ to the opetusvirhe (which is nothing but harjaregressio). The decision on which route to use for regularization can be based on the available computational infrastructure. For example, it might be much easier to implement data augmentation than malli pruning.

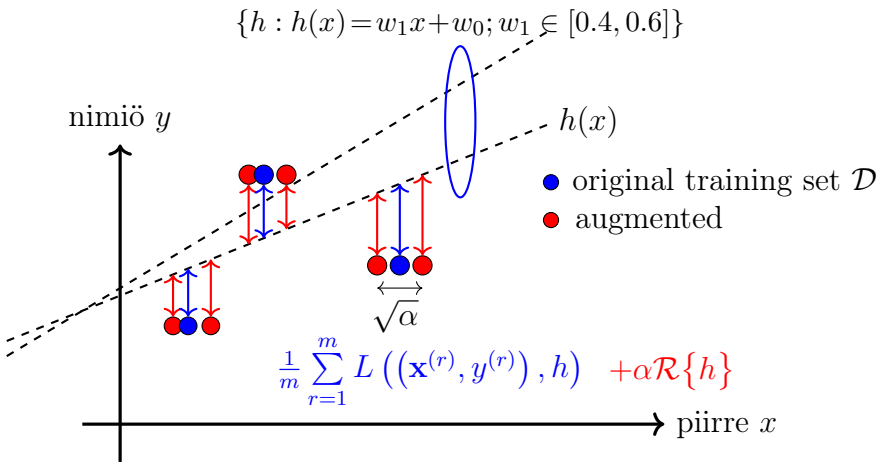


Fig. 59. Three approaches to regularization: 1) data augmentation; 2) häviö penalization; and 3) malli pruning (via constraints on model parameters).

See also: ylisovittaminen, data augmentation, validointi, mallin valinta.

regularized empirical risk minimization (RERM) Basic ERM learns a hypothesis (or trains a malli) $h \in \mathcal{H}$ based solely on the empirical risk $\widehat{L}\big(h|\mathcal{D}\big)$ incurred on a training set $\mathcal{D}$. To make ERM less prone to ylisovittaminen, we can implement regularisointi by including a (scaled) regularisoija $\mathcal{R}\big\{h\big\}$ in the learning objective. This leads to RERM such that

$$\hat{h} \in \arg\min_{h \in \mathcal{H}} \widehat{L}\big(h|\mathcal{D}\big) + \alpha \mathcal{R}\big\{h\big\}. \tag{10}$$

The parameter $\alpha \geq 0$ controls the regularisointi strength. For $\alpha = 0$, we recover standard ERM without regularisointi. As α increases, the learned hypothesis is increasingly biased toward small values of $\mathcal{R}\big\{h\big\}$. The component $\alpha \mathcal{R}\big\{h\big\}$ in the kohdefunktiot of (10) can be intuitively understood as a surrogate for the increased average häviö that may occur when predicting nimiöt for data points outside the training set. This intuition can be made precise in various ways. For example, consider a linear model trained using neliövirhehäviö and the regularisoija $\mathcal{R}\big\{h\big\} = \|\mathbf{w}\|_2^2$. In this setting, $\alpha \mathcal{R}\big\{h\big\}$ corresponds to the expected increase in häviö caused by adding Gaussian RVs to the piirrevektori in the training set [8, Ch. 3]. A principled construction for the regularisoija $\mathcal{R}\big\{h\big\}$ arises from approximate upper bounds on the yleistys error. The resulting RERM instance is known as SRM [**?**, Sec. 7.2].

See also: ERM, regularisointi, häviö, SRM.

regularized loss minimization (RLM) See RERM.

reinforcement learning (RL) RL refers to an online learning setting where

we can only evaluate the usefulness of a single hypothesis (i.e., a choice of model parameters) at each time step t. In particular, RL methods apply the current hypothesis $h^{(t)}$ to the piirevektori $\mathbf{x}^{(t)}$ of the newly received data point. The usefulness of the resulting ennuste $h^{(t)}(\mathbf{x}^{(t)})$ is quantified by a reward signal $r^{(t)}$ (see Fig. 60).

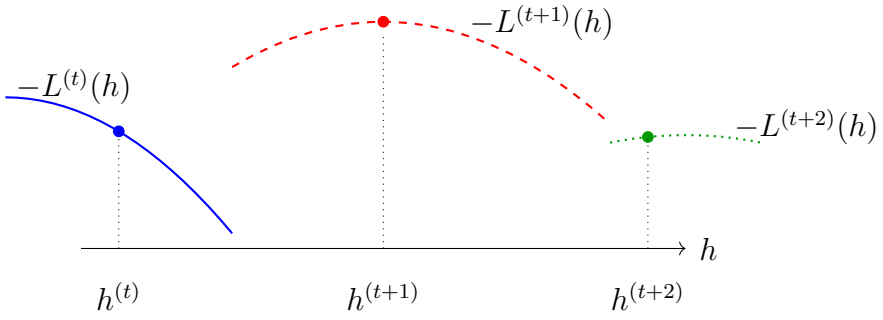


Fig. 60. Three consecutive time steps $t, t+1, t+2$ with corresponding häviöfunktiot $L^{(t)}, L^{(t+1)}, L^{(t+2)}$. During time step t, an RL method can evaluate the häviöfunktio only for one specific hypothesis $h^{(t)}$, resulting in the reward signal $r^{(t)} = -L^{(t)}(h^{(t)})$.

In general, the reward depends also on the previous ennuste $h^{(t')}\big(\mathbf{x}^{(t')}\big)$ for $t' < t$. The goal of RL is to learn $h^{(t)}$, for each time step t, such that the (possibly discounted) cumulative reward is maximized [8], [**?**].

See also: reward, häviöfunktio, koneoppiminen.

Rényi divergence The Rényi divergence measures the (dis)similarity between two todennäköisyysjakaumat [**?**].

See also: todennäköisyysjakauma.

reward A reward refers to some observed (or measured) quantity that allows us to estimate the häviö incurred by the ennuste (or decision) of a hypothesis $h(\mathbf{x})$. For example, in an koneoppiminen application to self-driving vehicles, $h(\mathbf{x})$ could represent the current steering direction of a vehicle. We could construct a reward from the measurements of a collision sensor that indicate if the vehicle is moving toward an obstacle. We define a low reward for the steering direction $h(\mathbf{x})$ if the vehicle moves dangerously toward an obstacle.

See also: häviö, MAB, RL.

risk Consider a hypothesis h used to predict the nimiö y of a data point based on its piirre $\mathbf{x}$. We measure the quality of a particular ennuste using a häviöfunktio $L\left((\mathbf{x}, y), h\right)$. If we interpret data points as the realizations of i.i.d. satunnaismuuttujat, the $L\left((\mathbf{x}, y), h\right)$ also becomes the realization of an satunnaismuuttuja. The i.i.d. assumption allows us to define the risk of a hypothesis as the expected häviö $\mathbb{E}\big\{L\left((\mathbf{x}, y), h\right)\big\}$. Note that the risk of h depends on both the specific choice for the häviöfunktio and the todennäköisyysjakauma of the data points.

See also: hypothesis, nimiö, data point, piirre, ennuste, häviöfunktio, realization, i.i.d. satunnaismuuttuja, i.i.d. assumption, häviö, todennäköisyysjakauma.

robustness Robustness is a key requirement for trustworthy AI. It refers to the property of an koneoppiminen system to maintain acceptable performance even when subjected to different forms of perturbations. These perturbations may affect the piirre of a data point in order to

manipulate the ennuste delivered by a trained koneoppiminen malli. Robustness also includes the stability of ERM-based methods against perturbations of the training set. Such perturbations can occur within data poisoning attacks.

See also: trustworthy AI,stability,data poisoning, attack.

rypäs A cluster is a subset of data points that are more similar to each other than to the data points outside the cluster. The quantitative measure of similarity between data points is a design choice. If data points are characterized by Euclidean piirrevektori $\mathbf{x} \in \mathbb{R}^d$, we can define the similarity between two data points via the Euclidean distance between their piirrevektori. An example of such clusters is shown in Fig. 61.

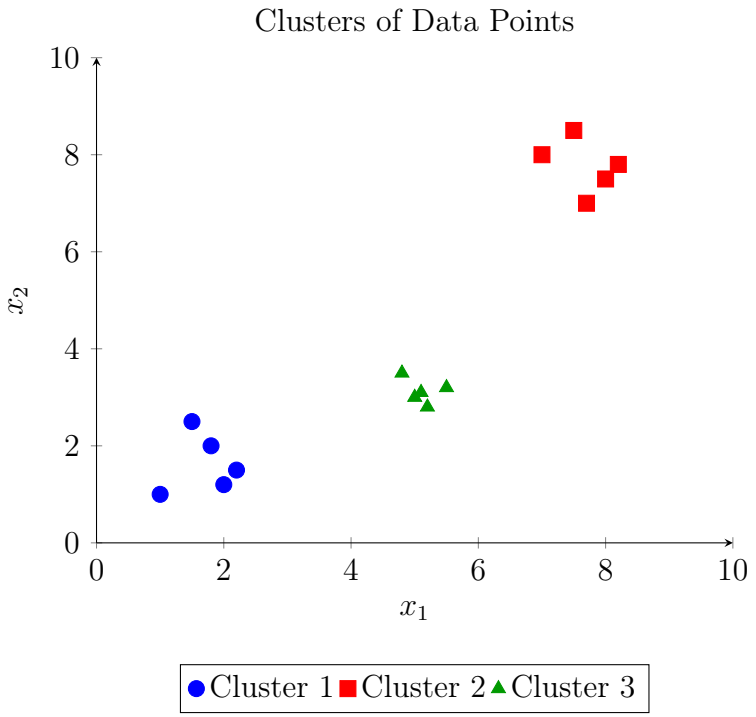


Fig. 61. Illustration of three clusters in a 2-D feature space. Each cluster groups data points that are more similar to each other than to those in other clusters, based on the Euclidean distance.

See also: data point, piirevektori, feature space.

sample In the context of koneoppiminen, a sample is a finite sequence (of length m) of data points, $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$. The number m is called the sample size. ERM-based methods use a sample to train a malli (or learn a hypothesis) by minimizing the average häviö (the empirical risk) over that sample. Since a sample is defined as a sequence, the same data point may appear more than once. By contrast, some authors in statistics

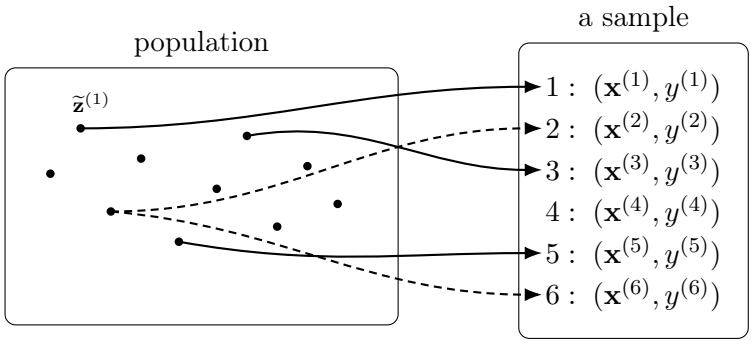


Fig. 62. Illustration of a sample as a finite sequence. Each sequence element consists of the piirevektori and the nimiö of some data point which belongs to an underlying population. Depending on the application, the same data point is used to obtain multiple sample elements.

define a sample as a set of data points, in which case duplicates are not allowed [**?**, **?**]. These two views can be reconciled by regarding a sample as a sequence of piirre–nimiö pairs, $\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right)$. The r-th pair consists of the piirre $\mathbf{x}^{(r)}$ and the nimiö $y^{(r)}$ of an unique underlying data point $\widetilde{\mathbf{z}}^{(r)}$. While the underlying data points $\widetilde{\mathbf{z}}^{(1)}, \ldots, \widetilde{\mathbf{z}}^{(m)}$ are unique, some of them can have identical piirre and nimiöt. For the analysis of koneoppiminen methods, it is common to interpret a sample as the realization of a stochastic process indexed by $\{1, \ldots, m\}$. A widely used assumption is the i.i.d. assumption, where sample elements $\left(\mathbf{x}^{(r)}, y^{(r)}\right)$, for $r = 1, \ldots, m$, are i.i.d. satunnaismuuttujat with common todennäköisyysjakauma $p(\mathbf{x}, y)$.

See also: data point, realization, i.i.d., satunnaismuuttuja, todennäköisyysjakauma, sample size, ERM.

sample covariance matrix The sample kovarianssimatriisi $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$ for a given set of piirrevektori $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$ is defined as

$$\widehat{\mathbf{\Sigma}} = \frac{1}{m} \sum_{r=1}^{m} (\mathbf{x}^{(r)} - \widehat{\mathbf{m}})(\mathbf{x}^{(r)} - \widehat{\mathbf{m}})^T.$$

Here, we use the sample mean $\widehat{\mathbf{m}}$.

See also: sample, kovarianssimatriisi, piirevektori, sample mean.

sample mean The sample mean $\mathbf{m} \in \mathbb{R}^d$ for a given tietoaineisto, with piirrevektori $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$, is defined as

$$\mathbf{m} = \frac{1}{m} \sum_{r=1}^{m} \mathbf{x}^{(r)}.$$

See also: sample, mean, tietoaineisto, piirevektori.

sample size The number of individual data points contained in a sample.

See also: data point, tietoaineisto.

sample space A sample space is the set of all possible outcomes of a satunnaiskoe [6], [7], [**?**], [**?**].

See also: probability space.

satunnaismetsä A random forest is a set of different päätöspuut. Each of these päätöspuut is obtained by fitting a perturbed copy of the original tietoaineisto.

See also: päätöspuu, tietoaineisto.

satunnaismuuttuja An RV is a function that maps the outcomes of a satunnaiskoe to a value space [6], [**?**]. Mathematically, an RV is a function

$x : \Omega \to \mathcal{X}$ that is defined on the sample space Ω of a probability space. Different types of RVs include

- binary RVs, which map each outcome to an element of a binary set (e.g., $\{-1, 1\}$ or $\{\text{cat}, \text{no cat}\}$);
- real-valued RVs, which take on values in the real numbers $\mathbb{R}$;
- vector-valued RVs, which map outcomes to the Euclidean space $\mathbb{R}^d$.

Probability theory uses the concept of measurable spaces to rigorously define and study the properties of collections of RVs [6].
See also: function, satunnaiskoe, sample space, probability space, vector, Euclidean space, probability, measurable.

scatterplot A visualization technique that depicts data points using markers in a 2-D plane. Fig. 63 depicts an example of a scatterplot.

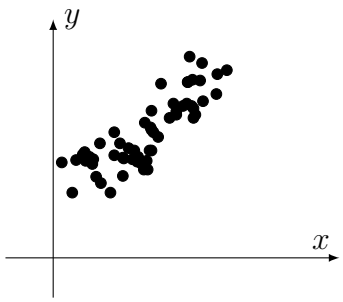


Fig. 63. A scatterplot with circle markers, where the data points represent daily weather conditions in Finland. Each data point is characterized by its minimum daytime temperature x as the piirre and its maximum daytime temperature y as the nimiö. The temperatures have been measured at the FMI weather station Helsinki Kaisaniemi during 1 September 2024—28 October 2024.

A scatterplot can enable the visual inspection of data points that are naturally represented by piirrevektori in high-dimensional spaces.
See also: data point, minimum, piirre, maximum, nimiö, FMI, piirevektori, ulottuvuuksien vähentäminen.

sekaannusmatriisi Consider data points characterized by piirre $\mathbf{x}$ and corresponding nimiöt y. The nimiöt take on values in a finite label space $\mathcal{Y} = \{1, \ldots, k\}$. For a given hypothesis h, the confusion matriisi is a $k \times k$ matriisi where each row corresponds to a different value of the true nimiö $y \in \mathcal{Y}$ and each column to a different value of the ennuste $h(\mathbf{x}) \in \mathcal{Y}$. The (c, c')th entry of the confusion matriisi represents the fraction of data points with a true nimiö $y = c$ that are predicted as $h(\mathbf{x}) = c'$. The main diagonal of the confusion matriisi contains the fractions of correctly classified data points (i.e., those for which $y = h(\mathbf{x})$).

The off-diagonal entries contain the fractions of data points that are misclassified by h.

See also: nimiö, label space, hypothesis, matriisi, luokittelu.

selitettävyys We define the (subjective) explainability of an koneoppiminen method as the level of simulatability [?] of the ennuste delivered by an koneoppiminen system to a human user. Quantitative measures for the (subjective) explainability of a trained malli can be constructed by comparing its ennuste with the ennuste provided by a user on a test set [?], [?]. Alternatively, we can use todennäköisyysmalli for data and measure the explainability of a trained koneoppiminen malli via the conditional (or differential) entropy of its ennuste, given the user's ennuste [?], [?].

See also: trustworthy AI, regularisointi.

selitettävä koneoppiminen XML methods aim to complement each ennuste with an explanation of how the ennuste has been obtained. The construction of an explicit explanation might not be necessary if the koneoppiminen method uses a sufficiently simple (or interpretable) malli [?].

See also: ennuste, explanation, koneoppiminen, malli.

semi-supervised learning (SSL) SSL methods use unlabeled data points to support the learning of a hypothesis from labeled data points [?]. This approach is particularly useful for koneoppiminen applications that offer a large number of unlabeled data points, but only a limited number of labeled data points.

See also: data point, hypothesis, labeled data point, koneoppiminen.

sensitive attribute koneoppiminen revolves around learning a hypothesis map that allows us to predict the nimiö of a data point from its piirre. In some applications, we must ensure that the output delivered by an koneoppiminen system does not allow us to infer sensitive attributes of a data point. Which part of a data point is considered a sensitive attribute is a design choice that varies across different application domains.
See also: koneoppiminen, hypothesis, map, nimiö, data point, piirre.

similarity graph Some koneoppiminen applications generate data points that are related by a domain-specific notion of similarity. These similarities can be represented conveniently using a similarity graph $\mathcal{G} = \big(\mathcal{V} := \{1, \ldots, m\}, \mathcal{E}\big)$. The node $r \in \mathcal{V}$ represents the rth data point. Two nodes are connected by an undirected edge if the corresponding data points are similar.
See also: koneoppiminen, data point, graph.

singular value decomposition (SVD) The SVD for a matriisi $\mathbf{A} \in \mathbb{R}^{m \times d}$ is a factorization of the form

$$\mathbf{A} = \mathbf{V} \mathbf{\Lambda} \mathbf{U}^T$$

with orthonormal matriisit $\mathbf{V} \in \mathbb{R}^{m \times m}$ and $\mathbf{U} \in \mathbb{R}^{d \times d}$ [3]. The matriisi $\mathbf{\Lambda} \in \mathbb{R}^{m \times d}$ is only nonzero along the main diagonal, whose entries $\Lambda_{j,j}$ are nonnegative and referred to as singular values.
See also: matriisi.

smooth A real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ is smooth if it is differentiable and its gradientti $\nabla f(\mathbf{w})$ is continuous at all $\mathbf{w} \in \mathbb{R}^d$ [**?**], [**?**]. A smooth function f is referred to as β-smooth if the gradientti $\nabla f(\mathbf{w})$ is Lipschitz continuous with Lipschitz constant β, i.e.,

$$\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\| \leq \beta \|\mathbf{w} - \mathbf{w}'\|, \text{ for any } \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d.$$

The constant β quantifies the smoothness of the function f: the smaller the β, the smoother f is. Optimization problems with a smooth kohdefunktiot can be solved effectively by gradient-based methods. Indeed, gradient-based methods approximate the kohdefunktiot locally around a current choice $\mathbf{w}$ using its gradientti. This approximation works well if the gradientti does not change too rapidly. We can make this informal claim precise by studying the effect of a single gradient step with step size $\eta = 1/\beta$ (see Fig. 64).

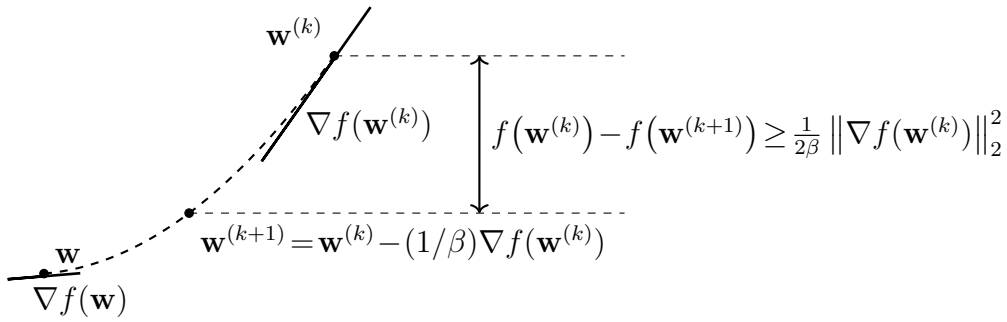


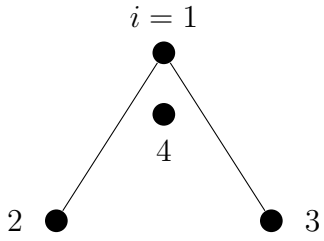
Fig. 64. Consider an kohdefunktiot $f(\mathbf{w})$ that is β-smooth. Taking a gradient step, with step size $\eta = 1/\beta$, decreases the objective by at least $1/2\beta \left\|\nabla f(\mathbf{w}^{(k)})\right\|_2^2$ [?], [?], [?]. Note that the step size $\eta = 1/\beta$ becomes larger for smaller β. Thus, for smoother kohdefunktiot (i.e., those with smaller β), we can take larger steps.

See also: function, differentiable, gradientti, gradient-based methods.

soft clustering Soft klusterointi refers to the task of partitioning a given set of data points into (a few) overlapping ryppäät. Each data point is assigned to several different ryppäät with varying degrees of belonging. Soft klusterointi methods determine the degree of belonging (or soft rypäs assignment) for each data point and each rypäs. A principled approach to soft klusterointi is by interpreting data points as i.i.d. realizations of a Gaussin sekoitemalli. The conditional probability of a data point belonging to a specific mixture component is then a natural choice for the degree of belonging.

See also: klusterointi, data point, rypäs, degree of belonging, i.i.d., realization, Gaussin sekoitemalli, probability.

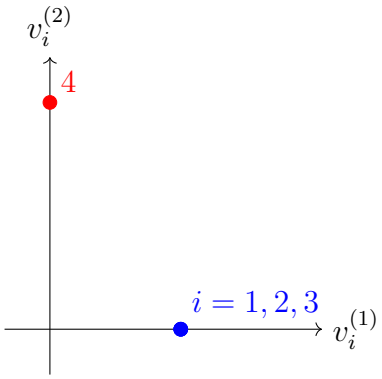
spectral clustering Spectral klusterointi is a particular instance of graph clustering, i.e., it clusters data points represented as the nodes $i = 1, \ldots, n$ of a graph $\mathcal{G}$. Spectral klusterointi uses the eigenvectors of the Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$ to construct piirrevektori $\mathbf{x}^{(i)} \in \mathbb{R}^d$ for each node (i.e., for each data point) $i = 1, \ldots, n$. We can feed these piirrevektori into Euclidean space-based klusterointi methods, such as k-means or soft clustering via Gaussin sekoitemalli. Roughly speaking, the piirrevektori of nodes belonging to a well-connected subset (or rypäs) of nodes in $\mathcal{G}$ are located nearby in the Euclidean space $\mathbb{R}^d$ (see Fig. 65).



(a)

$$\mathbf{L}^{(\mathcal{G})} = \begin{pmatrix} 2 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$$

(b)



(c)

$$\mathbf{V} = \left(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)}, \mathbf{v}^{(4)}\right)$$

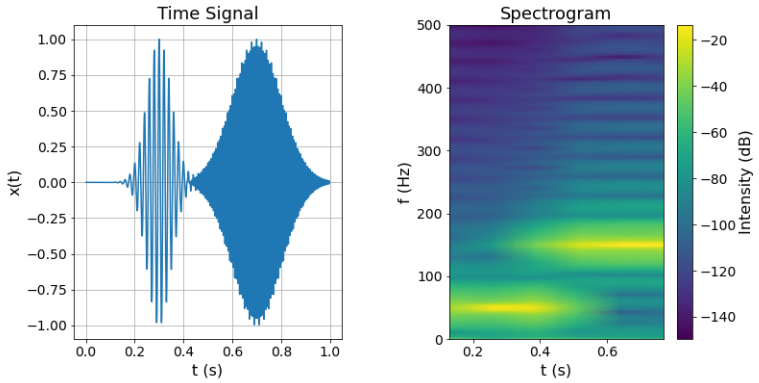
$$\mathbf{v}^{(1)} = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \; \mathbf{v}^{(2)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

(d)

Fig. 65. (a) An undirected graph $\mathcal{G}$ with four nodes $i = 1, 2, 3, 4$, each representing a data point. (b) The Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{4\times 4}$ and its EVD. (c) A scatterplot of data points using the piirrevektori $\mathbf{x}^{(i)} = \left(v_i^{(1)}, v_i^{(2)}\right)^T$. (d) Two eigenvectors $\mathbf{v}^{(1)}, \mathbf{v}^{(2)} \in \mathbb{R}^d$ corresponding to the eigenvalue $\lambda = 0$ of the Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$.

See also: klusterointi, graph clustering, Laplacian matrix, eigenvalue.

spectrogram A spectrogram represents the time-frequency distribution of the energy of a time signal $x(t)$. Intuitively, it quantifies the amount of signal energy present within a specific time segment $[t_1, t_2] \subseteq \mathbb{R}$ and frequency interval $[f_1, f_2] \subseteq \mathbb{R}$. Formally, the spectrogram of a signal is defined as the squared magnitude of its short-time Fourier transform (STFT) [**?**]. Fig. 66 depicts a time signal along with its spectrogram.



(a) (b)

Fig. 66. (a) A time signal consisting of two modulated Gaussian pulses. (b) An intensity plot of the spectrogram.

The intensity plot of its spectrogram can serve as an image of a signal. A simple recipe for audio signal luokittelu is to feed this signal image into deep nets originally developed for image luokittelu and object detection [**?**]. It is worth noting that, beyond the spectrogram, several alternative representations exist for the time-frequency distribution of signal energy [**?**], [**?**].

See also: luokittelu, deep net.

stability Stability is a desirable property of an koneoppiminen method $\mathcal{A}$ that maps a tietoaineisto $\mathcal{D}$ (e.g., a training set) to an output $\mathcal{A}(\mathcal{D})$. The output $\mathcal{A}(\mathcal{D})$ can be the learned model parameters or the ennuste delivered by the trained malli for a specific data point. Intuitively, $\mathcal{A}$ is stable if small changes in the input tietoaineisto $\mathcal{D}$ lead to small changes in the output $\mathcal{A}(\mathcal{D})$. Several formal notions of stability exist that enable bounds on the yleistys error or risk of the method (see [**?**, Ch. 13]). To build intuition, consider the three tietoaineistot depicted in Fig. 67, each of which is equally likely under the same data-generating todennäköisyysjakauma. Since the optimal model parameters are determined by this underlying todennäköisyysjakauma, an accurate koneoppiminen method $\mathcal{A}$ should return the same (or very similar) output $\mathcal{A}(\mathcal{D})$ for all three tietoaineistot. In other words, any useful $\mathcal{A}$ must be robust to variability in sample realizations from the same todennäköisyysjakauma, i.e., it must be stable.

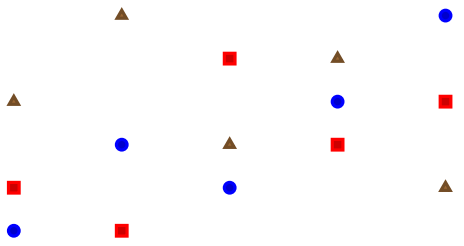


Fig. 67. Three tietoaineistot $\mathcal{D}^{(*)}$, $\mathcal{D}^{(\square)}$, and $\mathcal{D}^{(\triangle)}$, each sampled independently from the same data-generating todennäköisyysjakauma. A stable koneoppiminen method should return similar outputs when trained on any of these tietoaineistot.

See also: koneoppiminen, tietoaineisto, training set, model parameters, ennuste, malli, data point, yleistys, risk, data, todennäköisyysjakauma, sample, realization.

standard normal vector A standard normal vector is a random vector $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ whose entries are i.i.d. Gaussian RVs $x_j \sim \mathcal{N}(0,1)$. It is a special case of a multivariate normal distribution, $\mathbf{x} \sim (\mathbf{0}, \mathbf{I})$.

See also: vector, i.i.d., Gaussian RV, multivariate normal distribution, satunnaismuuttuja.

statistical aspects By statistical aspects of an koneoppiminen method, we refer to (properties of) the todennäköisyysjakauma of its output under a todennäköisyysmalli for the data fed into the method.

See also: koneoppiminen, todennäköisyysjakauma, todennäköisyysmalli, data.

step size See oppimisnopeus.

stochastic We refer to a method as stochastic if it involves a random component or is governed by probabilistic laws. Koneoppiminen methods use randomness to reduce computational complexity (e.g., see SGD) or to capture epävarmuus in todennäköisyysmalli.

See also: SGD, epävarmuus, todennäköisyysmalli.

stochastic algorithm A stochastic algorithm uses a random mechanism during its execution. For example, SGD uses a randomly selected subset of data points to compute an approximation for the gradientti of an kohdefunktiot. We can represent a stochastic algorithm by a stochastic processes, i.e., the possible execution sequence is the possible outcomes of a satunnaiskoe [7], [**?**], [**?**].

See also: stochastic, algorithm, SGD, data point, gradientti, kohdefunktiot, stochastic process, satunnaiskoe, optimization method, gradient-based methods.

stochastic block model (SBM) The SBM is a probabilistic generative malli for an undirected graph $\mathcal{G} = \big(\mathcal{V}, \mathcal{E}\big)$ with a given set of nodes $\mathcal{V}$ [**?**]. In its most basic variant, the SBM generates a graph by first randomly assigning each node $i \in \mathcal{V}$ to a rypäs index $c_i \in \{1, \ldots, k\}$. A pair of different nodes in the graph is connected by an edge with probability $p_{i,i'}$ that depends solely on the nimiöt $c_i, c_{i'}$. The presence

of edges between different pairs of nodes is statistically independent. See also: malli, graph, rypäs, probability, nimiö.

stochastic gradient descent (SGD) SGD is obtained from GD by replacing the gradientti of the kohdefunktiot with a stochastic approximation. A main application of SGD is to train a parameterized malli via ERM on a training set $\mathcal{D}$ that is either very large or not readily available (e.g., when data points are stored in a database distributed globally). To evaluate the gradientti of the empirical risk (as a function of the model parameters $\mathbf{w}$), we need to compute a sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over all data points in the training set. We obtain a stochastic approximation to the gradientti by replacing the sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ with a sum $\sum_{r \in \mathcal{B}} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over a randomly chosen subset $\mathcal{B} \subseteq \{1, \ldots, m\}$ (see Fig. 68). We often refer to these randomly chosen data points as a batch. The batch size $|\mathcal{B}|$ is an important parameter of SGD. SGD with $|\mathcal{B}| > 1$ is referred to as mini-batch SGD [**?**].

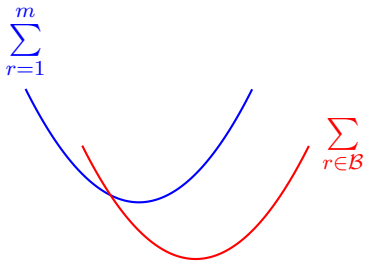


Fig. 68. SGD for ERM approximates the gradientti by replacing the sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over all data points in the training set (indexed by $r = 1, \ldots, m$) with a sum over a randomly chosen subset $\mathcal{B} \subseteq \{1, \ldots, m\}$.

See also: GD, gradientti, kohdefunktiot, stochastic, malli, ERM, training set, data point, empirical risk, function, model parameters, batch, parameter.

stopping criterion Many koneoppiminen methods use iterative algoritmit that construct a sequence of model parameters in order to minimize the opetusvirhe. For example, gradient-based methods iteratively update the parameters of a parametric malli, such as a linear model or a deep net. Given a finite amount of computational resources, we need to stop updating the parameters after a finite number of iterations. A stopping criterion is any well-defined condition for deciding when to stop updating.

See also: algorithm, gradient-based methods.

strongly convex A continuously differentiable real-valued function $f(\mathbf{x})$ is strongly convex with coefficient σ if $f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^{T}(\mathbf{y}-\mathbf{x}) + (\sigma/2)\left\|\mathbf{y}-\mathbf{x}\right\|_{2}^{2}$ [?], [?, Sec. B.1.1].

See also: differentiable, function, convex.

structural risk minimization (SRM) SRM is an instance of RERM, with which the malli $\mathcal{H}$ can be expressed as a countable union of submodels such that $\mathcal{H} = \bigcup_{n=1}^{\infty} \mathcal{H}^{(n)}$. Each submodel $\mathcal{H}^{(n)}$ permits the derivation of an approximate upper bound on the yleistys error incurred when applying ERM to train $\mathcal{H}^{(n)}$. These individual bounds—one for each submodel—are then combined to form a regularisoija used in the RERM objective. These approximate upper bounds (one for each $\mathcal{H}^{(n)}$) are then combined to construct a regularisoija for RERM [?, Sec. 7.2].

See also: RERM, malli, yleistys, ERM, regularisoija, risk.

subgradient For a real-valued function $f : \mathbb{R}^d \to \mathbb{R} : \mathbf{w} \mapsto f(\mathbf{w})$, a vector $\mathbf{a}$ such that $f(\mathbf{w}) \geq f(\mathbf{w}') + \left(\mathbf{w} - \mathbf{w}'\right)^T \mathbf{a}$ is referred to as a subgradient of f at $\mathbf{w}'$ [?], [?].

See also: function, vector.

subgradient descent Subgradient descent is a yleistys of GD that does not require differentiability of the function to be minimized. This yleistys is obtained by replacing the concept of a gradientti with that of a subgradient. Similar to gradientit, subgradients allow us to construct local approximations of an kohdefunktiot. The kohdefunktiot might be the empirical risk $\widehat{L}\left(h^{(\mathbf{w})} \middle| \mathcal{D}\right)$ viewed as a function of the model parameters $\mathbf{w}$ that select a hypothesis $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: subgradient, yleistys, GD, function, gradientti, kohdefunktiot, empirical risk, model parameters, hypothesis.

support vector machine (SVM) The SVM is a binary luokittelu method that learns a linear hypothesis map. Thus, like lineaarinen regressio and logistic regression, it is also an instance of ERM for the linear model. However, the SVM uses a different häviöfunktio from the one used in those methods. As illustrated in Fig. 69, it aims to maximally separate data points from the two different classes in the feature space (i.e., maximum margin principle). Maximizing this separation is equivalent to minimizing a regularized variant of the hinge loss (6) [?], [?], [?].

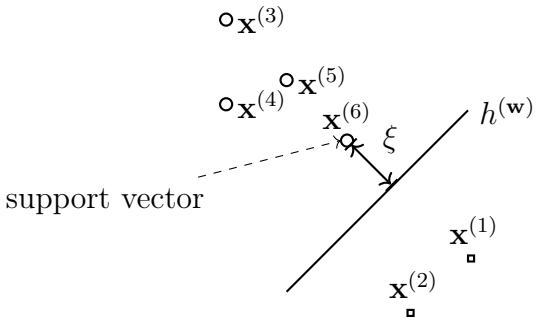


Fig. 69. The SVM learns a hypothesis (or luokitin) $h^{(\mathbf{w})}$ with minimal average soft-margin hinge loss. Minimizing this häviö is equivalent to maximizing the margin ξ between the päätöspinta of $h^{(\mathbf{w})}$ and each class of the training set.

The above basic variant of SVM is only useful if the data points from different categories can be (approximately) linearly separated. For an koneoppiminen application where the categories are not derived from a ydinfunktio.

See also: luokittelu, linear model, luokitin, hinge loss.

supremum (or least upper bound) The supremum of a set of real numbers is the smallest number that is greater than or equal to every element in the set. More formally, a real number a is the supremum of a set $\mathcal{A} \subseteq \mathbb{R}$ if: 1) a is an upper bound of $\mathcal{A}$; and 2) no number smaller than a is an upper bound of $\mathcal{A}$. Every non-empty set of real numbers that is bounded above has a supremum, even if it does not contain its supremum as an element [2, Sec. 1.4].

suurimman uskottavuuden menetelmä Consider data points $\mathcal{D} = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ that are interpreted as the realizations of i.i.d. satunnaismuuttujat

with a common todennäköisyysjakauma $\mathbb{P}(\mathbf{z}; \mathbf{w})$, which depends on the model parameters $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^n$. Maximum likelihood methods learn model parameters $\mathbf{w}$ by maximizing the probability (density) $\mathbb{P}(\mathcal{D}; \mathbf{w}) = \prod_{r=1}^{m} \mathbb{P}(\mathbf{z}^{(r)}; \mathbf{w})$ of the observed tietoaineisto. Thus, the maximum likelihood estimator is a solution to the optimization problem $\max_{\mathbf{w} \in \mathcal{W}} \mathbb{P}(\mathcal{D}; \mathbf{w})$.

See also: todennäköisyysjakauma, optimization problem, todennäköisyysmalli.

tarkkuus Consider data points characterized by piirre $\mathbf{x} \in \mathcal{X}$ and a categorical nimiö y that takes on values from a finite label space $\mathcal{Y}$. The accuracy of a hypothesis $h : \mathcal{X} \to \mathcal{Y}$, when applied to the data points in a tietoaineisto $\mathcal{D} = \left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right) \right\}$, is then defined as $1 - (1/m) \sum_{r=1}^{m} L^{(0/1)} \left(\left(\mathbf{x}^{(r)}, y^{(r)}\right), h \right)$ using the 0/1 loss $L^{(0/1)}(\cdot, \cdot)$.

See also: 0/1 loss, häviö, metric.

tekoäly AI refers to systems that behave rationally in the sense of maximizing a long-term reward. The koneoppiminen-based approach to AI is to train a malli to predict optimal actions. These ennuste are computed from observations about the state of the environment. The choice of häviöfunktio sets AI applications apart from more basic koneoppiminen applications. AI systems rarely have access to a labeled training set that allows the average häviö to be measured for any possible choice of model parameters. Instead, AI systems use observed reward signals to estimate the häviö incurred by the current choice of model parameters.

See also: koneoppiminen, RL.

test set A set of data points that have been used neither to train a malli (e.g., via ERM) nor to choose between different malli in a validation set. See also: data point, malli, ERM, validation set.

tietoaineisto A dataset is a set of distinct data points. In contrast to a sample, which is defined as a sequence of data points and may contain repetitions, a dataset is an unordered collection without duplicates. koneoppiminen methods use datasets to train and validate malli. The notion of a dataset is broad: data points may represent concrete physical entities (such as humans or animals) or abstract objects (such as numbers). For illustration, Fig. 70 depicts a dataset whose data points are cows.

Fig. 70. A cow herd somewhere in the Alps.

Quite often, an koneoppiminen engineer does not have direct access to the underlying dataset. For instance, accessing the dataset in Fig. 70 would require visiting the cow herd. In practice, we work with a more convenient representation (or approximation) of the dataset. Various mathematical malli have been developed for this purpose [**?**], [**?**], [**?**], [**?**]. One of the most widely used is the relational malli, which organizes data

as a table (or relation) [?], [?]. A table consists of rows and columns: each row corresponds to a single data point, while each column represents a specific attribute of a data point. koneoppiminen methods typically interpret these attributes as piirre or as a nimiö of a data point. As an illustration, Table I shows a relational representation of the dataset from Fig. 70. In the relational malli, the order of rows is immaterial, and each attribute (column) is associated with a domain that specifies the set of admissible values. In koneoppiminen applications, these attribute domains correspond to the feature space and the label space.

TABLE I

A Relation (or Table) That Represents the Dataset in Fig. 70

Name	**Weight**	**Age**	**Height**	**Stomach temperature**
Zenzi	100	4	100	25
Berta	140	3	130	23
Resi	120	4	120	31

While the relational malli is useful for the study of many koneoppiminen applications, it may be insufficient regarding the requirements for trustworthy AI. Modern approaches like datasheets for datasets provide more comprehensive documentation, including details about the data collection process, intended use, and other contextual information [?]. See also: data point, data, piirre, sample, feature space, label space.

todennäköisyysjakauma To analyze koneoppiminen methods, it can be useful to interpret data points as i.i.d. realizations of an satunnaismuut-

tuja. The typical properties of such data points are then governed by the probability distribution of this satunnaismuuttuja. The probability distribution of a binary satunnaismuuttuja $y \in \{0,1\}$ is fully specified by the probabilities $\mathbb{P}(y=0)$ and $\mathbb{P}(y=1)=1-\mathbb{P}(y=0)$. The probability distribution of a real-valued satunnaismuuttuja $x \in \mathbb{R}$ might be specified by a pdf $p(x)$ such that $\mathbb{P}(x \in [a,b]) \approx p(a)|b-a|$. In the most general case, a probability distribution is defined by a probability measure [6], [**?**].

See also: i.i.d., realization, satunnaismuuttuja, probability, pdf.

todennäköisyysmalli A probabilistic malli interprets data points as realizations of satunnaismuuttujat with a joint todennäköisyysjakauma. This joint todennäköisyysjakauma typically involves parameters that have to be manually chosen or learned via statistical inference methods such as suurimman uskottavuuden menetelmä estimation [**?**].

See also: malli, data point, realization, satunnaismuuttuja, todennäköisyysjakauma, parameter, suurimman uskottavuuden menetelmä.

total variation See GTV.

training In the context of koneoppiminen, training refers to the process of learning a useful hypothesis $\hat{h}$ out of a malli $\mathcal{H}$. The training of a malli $\mathcal{H}$ is guided by the häviö incurred on a set of data points, which serve as the training set. For parametric models, where each hypothesis $h^{(\mathbf{w})}$ is characterized by a specific choice for the model parameters, training amounts to finding an optimal choice for the model parameterst $\mathbf{w}$. A widely-used approach to training is ERM, which learns a hypothesis

by minimizing the average häviö incurred on a training set. One of the main challenges in koneoppiminen is to control the discrepancy between the häviö incurred on the training set and the häviö incurred on other (unseen) data points.

See also: ERM, häviö, malli.

training set A training set is a tietoaineisto $\mathcal{D}$ that consists of some data points used in ERM to learn a hypothesis $\hat{h}$. The average häviö of $\hat{h}$ on the training set is referred to as the opetusvirhe. The comparison of the opetusvirhe with the validointivirhe of $\hat{h}$ allows us to diagnose the koneoppiminen method and informs how to improve the validointivirhe (e.g., using a different hypothesis space or collecting more data points) [8, Sec. 6.6].

See also: tietoaineisto, data point, ERM, hypothesis, häviö, opetusvirhe, validointivirhe, koneoppiminen, hypothesis space.

transfer learning Transfer learning aims at leveraging information obtained while solving an existing learning task to solve another learning task.

See also: learning task, multitask learning

transparency Transparency is a fundamental requirement for trustworthy AI [?]. In the context of koneoppiminen methods, transparency is often used interchangeably with selitettävyys [?], [?]. However, in the broader scope of AI systems, transparency extends beyond selitettävyys and includes providing information about the system's limitations, reliability, and intended use. In medical diagnosis systems, transparency requires disclosing the confidence level for the ennuste delivered by a trained malli.

In credit scoring, AI-based lending decisions should be accompanied by explanations of contributing factors, such as income level or credit history. These explanations allow humans (e.g., a loan applicant) to understand and contest automated decisions. Some koneoppiminen methods inherently offer transparency. For example, logistic regression provides a quantitative measure of luokittelu reliability through the value $|h(\mathbf{x})|$. Päätöspuut are another example, as they allow human-readable decision rules [**?**]. Transparency also requires a clear indication when a user is engaging with an AI system. For example, AI-powered chatbots should notify users that they are interacting with an automated system rather than a human. Furthermore, transparency encompasses comprehensive documentation detailing the purpose and design choices underlying the AI system. For instance, malli datasheets [**?**] and AI system cards [**?**] help practitioners understand the intended use cases and limitations of an AI system [**?**].

See also: trustworthy AI, selitettävyys.

trustworthy artificial intelligence (trustworthy AI) Besides the computational aspects and statistical aspects, a third main design aspect of koneoppiminen methods is their trustworthiness [**?**]. The EU has put forward seven key requirements (KRs) for trustworthy AI (which typically build on koneoppiminen methods) [**?**]:

1) KR1 – Human agency and oversight;
2) KR2 – Technical robustness and safety;
3) KR3 – Privacy and data governance;

4) KR4 – Transparency;

5) KR5 – Diversity, non-discrimination and fairness;

6) KR6 – Societal and environmental well-being;

7) KR7 – Accountability.

See also: computational aspects, statistical aspects, koneoppiminen, AI, robustness, data, transparency.

tulkittavuus An koneoppiminen method is interpretable for a human user if they can comprehend the decision process of the method. One approach to develop a precise definition of interpretability is via the concept of simulatability, i.e., the ability of a human to mentally simulate the malli behavior [**?**], [**?**], [**?**], [**?**], [**?**]. The idea is as follows: If a human user understands an koneoppiminen method, then they should be able to anticipate its ennuste on a test set. We illustrate such a test set in Fig. 71, which also depicts two learned hypotheses $\hat{h}$ and $\hat{h}'$. The koneoppiminen method producing the hypothesis $\hat{h}$ is interpretable to a human user familiar with the concept of a linear map. Since $\hat{h}$ corresponds to a linear map, the user can anticipate the ennuste of $\hat{h}$ on the test set. In contrast, the koneoppiminen method delivering $\hat{h}'$ is not interpretable, because its behavior is no longer aligned with the user's expectations.

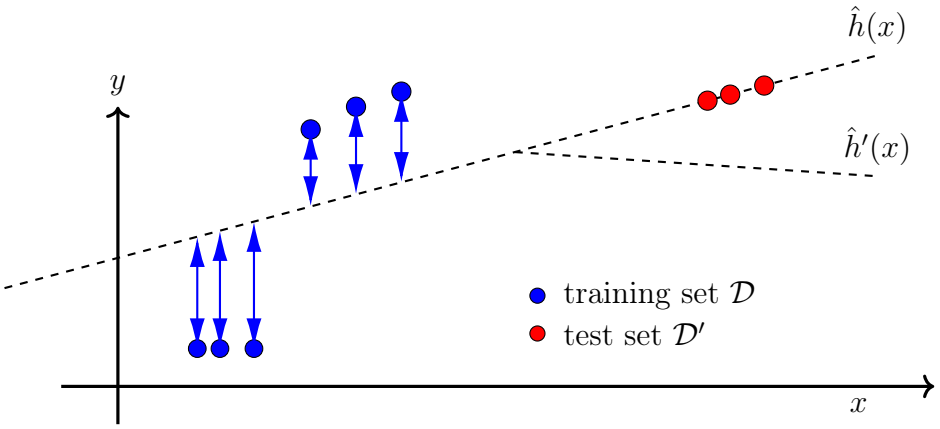


Fig. 71. We can assess the interpretability of trained koneoppiminen malli $\hat{h}$ and $\hat{h}'$ by comparing their ennuste to pseudo-nimiöt generated by a human user for $\mathcal{D}'$.

The notion of interpretability is closely related to the notion of selitettävyys, as both aim to make koneoppiminen methods more understandable for humans. In the context of Fig. 71, interpretability of an koneoppiminen method $\hat{h}$ requires that the human user can anticipate its ennuste on an arbitrary test set. This contrasts with selitettävyys, where the user is supported by external explanations—such as saliency maps or reference examples from the training set—to understand the ennuste of $\hat{h}$ on a specific test set $\mathcal{D}'$.

See also: selitettävyys, trustworthy AI, regularisointi, LIME.

ulottuvuuksien vähentäminen Dimensionality reduction refers to methods that learn a transformation $h : \mathbb{R}^d \to \mathbb{R}^{d'}$ of a (typically large) set of raw piirre $x_1, \ldots, x_d$ into a smaller set of informative piirre

$z_1, \ldots, z_{d'}$. Using a smaller set of piirre is beneficial in several ways:

- Statistical benefit: It typically reduces the risk of ylisovittaminen, as reducing the number of piirre often reduces the effective dimension of a malli.
- Computational benefit: Using fewer piirre means less computation for the training of koneoppiminen malli. As a case in point, lineaarinen regressio methods need to invert a matriisi whose size is determined by the number of piirre.
- Visualization: Dimensionality reduction is also instrumental for data visualization. For example, we can learn a transformation that delivers two piirre z_1, z_2, which we can use, in turn, as the coordinates of a scatterplot. Fig. 72 depicts the scatterplot of handwritten digits that are placed using transformed piirre. Here, the data points are naturally represented by a large number of greyscale values (one value for each pixel).

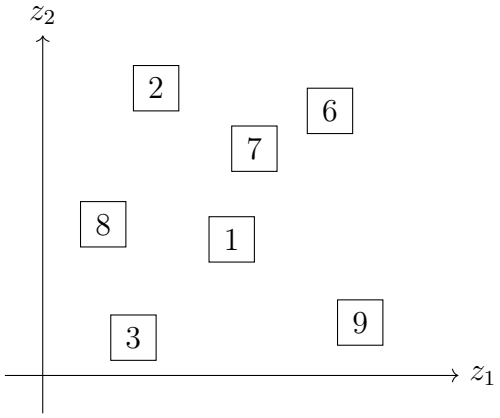


Fig. 72. Example of dimensionality reduction: High-dimensional image data (e.g., high-resolution images of handwritten digits) embedded into 2-D using learned piirre (z_1, z_2) and visualized in a scatterplot.

See also: ylisovittaminen, effective dimension, malli, scatterplot.

upper confidence bound (UCB) Consider an koneoppiminen application that requires selecting, at each time step k, an action a_k from a finite set of alternatives $\mathcal{A}$. The utility of selecting action a_k is quantified by a numeric reward signal $r^{(a_k)}$. A widely used todennäköisyysmalli for this type of sequential decision-making problem is the stochastic MAB setting [**?**]. In this malli, the reward $r^{(a)}$ is viewed as the realization of an satunnaismuuttuja with unknown mean $\mu^{(a)}$. Ideally, we would always choose the action with the largest expected reward $\mu^{(a)}$, but these means are unknown and must be estimated from observed data. Simply choosing the action with the largest estimate $\widehat{\mu}^{(a)}$ can lead to suboptimal outcomes due to estimation epävarmuus. The UCB strategy addresses this by selecting actions not only based on their estimated

means but also by incorporating a term that reflects the epävarmuus in these estimates—favoring actions with a high-potential reward and high epävarmuus. Theoretical guarantees for the performance of UCB strategies, including logarithmic regret bounds, are established in [**?**].

See also: koneoppiminen, reward, todennäköisyysmalli, stochastic, MAB, malli, realization, satunnaismuuttuja, mean, data, epävarmuus, regret.

uusio-otanta For the analysis of koneoppiminen methods, it is often useful to interpret a given set of data points $\mathcal{D} = \left\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\right\}$ as realizations of i.i.d. satunnaismuuttujat drawn from a common todennäköisyysjakauma $p(\mathbf{z})$. In practice, the todennäköisyysjakauma $p(\mathbf{z})$ is unknown and must be estimated from $\mathcal{D}$. The bootstrap approach uses the histogrammi of $\mathcal{D}$ as an estimator for $p(\mathbf{z})$.

See also: i.i.d., satunnaismuuttuja, todennäköisyysjakauma, histogrammi.

validation set A set of data points used to estimate the risk of a hypothesis $\hat{h}$ that has been learned by some koneoppiminen method (e.g., solving ERM). The average häviö of $\hat{h}$ on the validointi set is referred to as the validointivirhe and can be used to diagnose an koneoppiminen method (see [8, Sec. 6.6]). The comparison between opetusvirhe and validointivirhe can inform directions for the improvement of the koneoppiminen method (such as using a different hypothesis space).

See also: data point, risk, hypothesis, koneoppiminen, ERM, häviö, validointi, validointivirhe, opetusvirhe, hypothesis space.

validointi Consider a hypothesis $\hat{h}$ that has been learned via some koneop-

piminen method, e.g., by solving ERM on a training set $\mathcal{D}$. Validation

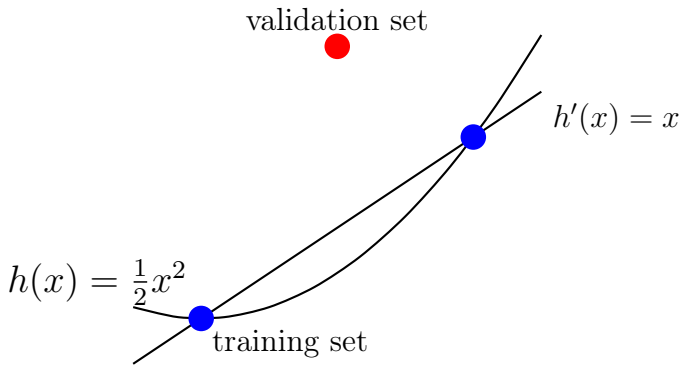


Fig. 73. Illustration of validation. The blue points represent the data points in the training set, while the red point represents a data point in the validation set. The hypothesis $\hat{h}$ (black curve) fits the data points in the training set perfectly, but incurs a large häviö on the data point in the validation set.

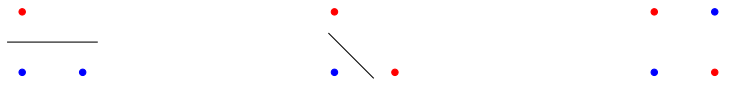
refers to the process of evaluating the häviö incurred by the hypothesis $\hat{h}$ on a set of data points that are not contained in the training set $\mathcal{D}$. This set of data points is called the validation set. The average häviö of $\hat{h}$ on the validation set is referred to as the validointivirhe.

See also: training set, ylisovittaminen, yleistys, validointivirhe, validation set.

validointivirhe Consider a hypothesis $\hat{h}$ that is obtained by some koneoppiminen method, e.g., using ERM on a training set. The average häviö of $\hat{h}$ on a validation set, which is different from the training set, is referred to as the validointi error.

See also: hypothesis, koneoppiminen, ERM, training set, häviö, validation set, validointi.

Vapnik–Chervonenkis dimension (VC dimension) The statistical properties of an ERM-based method depends critically on the expressive capacity of its hypothesis space (or malli) $\mathcal{H}$. A standard measure of this capacity is the VC dimension $\text{VCdim}(\mathcal{H})$ [**?**]. Formally, it is the largest integer m such that there exists a tietoaineisto $\mathcal{D} = \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \subseteq \mathcal{X}$ that can be perfectly classified (or shattered) by some $h \in \mathcal{H}$. In particular, for every one of the 2^m possible assignments of binary nimiöt to each piirevektori in $\mathcal{D}$, there exists some hypothesis $h \in \mathcal{H}$ that realizes this labeling. Intuitively, the VC dimension quantifies how well $\mathcal{H}$ can fit arbitrary nimiö assignments, and thus captures its approximate power. It plays a central role in deriving bounds on the generalization gap. Fig. 74 illustrates the definition of the VC dimension for a linear model $\mathcal{H}^{(2)}$ with $d = 2$ piirre. Fig. 74(a) and 74(b) show the same set of three noncollinear piirrevektori under two different binary labelings. In both cases, a separating hyperplane exists that realizes the labeling. Since this holds for all $2^3 = 8$ possible binary labelings of the three piirrevektori, the set is shattered. Fig. 74(c) depicts four piirrevektori with a specific labeling. No linear separator can correctly classify all data points in this case. Thus, $\text{VCdim}\left(\mathcal{H}^{(2)}\right) = 3$.



(a) (b) (c)

Fig. 74. Illustration of the VC dimension for a linear model $\mathcal{H}^{(2)}$ that is used to learn a lineaarinen luokitin in the feature space $\mathbb{R}^2$.

More generally, for a linear model $\mathcal{H}^{(d)}$, the VC dimension equals $d+1$. In other words, for linear models, the VC dimension essentially matches the dimension of the underlying feature space $\mathbb{R}^d$. For more complex hypothesis spaces, such as päätöspuut or neuroverkot, the relation between VC dimension and the dimension of the feature space is far less direct. In these cases, alternative complexity measures, such as the Rademacher complexity, can be more useful for analyzing ERM-based methods.
See also: hypothesis space, Rademacher complexity, yleistys, koneoppiminen, effective dimension.

variance The variance of a real-valued satunnaismuuttuja x is defined as the expectation $\mathbb{E}\big\{\big(x-\mathbb{E}\{x\}\big)^2\big\}$ of the squared difference between x and its expectation $\mathbb{E}\{x\}$. We extend this definition to vector-valued satunnaismuuttujat $\mathbf{x}$ as $\mathbb{E}\big\{\big\|\mathbf{x}-\mathbb{E}\{\mathbf{x}\}\big\|_2^2\big\}$.
See also: satunnaismuuttuja, expectation, vector.

vertailutaso Consider some koneoppiminen method that produces a learned hypothesis (or trained malli) $\hat{h} \in \mathcal{H}$. We evaluate the quality of a trained malli by computing the average häviö on a test set. But how can we assess whether the resulting test set performance is sufficiently good?

How can we determine if the trained malli performs close to optimal such that there is little point in investing more resources (for data collection or computation) to improve it? To this end, it is useful to have a reference (or baseline) level against which we can compare the performance of the trained malli.

Such a reference value might be obtained from human performance, e.g., the misclassification rate of dermatologists who diagnose cancer from visual inspection of skin [**?**]. Another source for a baseline is an existing, but for some reason unsuitable, koneoppiminen method. For example, the existing koneoppiminen method might be computationally too expensive for the intended koneoppiminen application. Nevertheless, its test set error can still serve as a baseline. Another, somewhat more principled, approach to constructing a baseline is via a todennäköisyysmalli. In many cases, given a todennäköisyysmalli $p(\mathbf{x}, y)$, we can precisely determine the minimum achievable risk among any hypotheses (not even required to belong to the hypothesis space $\mathcal{H}$) [**?**].

This minimum achievable risk (referred to as the Bayes risk) is the risk of the Bayes estimator for the nimiö y of a data point, given its piirre $\mathbf{x}$. Note that, for a given choice of häviöfunktio, the Bayes estimator (if it exists) is completely determined by the todennäköisyysjakauma $p(\mathbf{x}, y)$ [**?**, Ch. 4]. However, computing the Bayes estimator and Bayes risk presents two main challenges. First, the todennäköisyysjakauma $p(\mathbf{x}, y)$ is unknown and must be estimated from observed data. Second, even if $p(\mathbf{x}, y)$ were known, computing the Bayes risk exactly may be computationally infeasible [**?**]. A widely used todennäköisyysmalli is

the multivariate normal distribution $(\mathbf{x}, y) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for data points characterized by numeric piirre and nimiöt. Here, for the neliövirhehäviö, the Bayes estimator is given by the posterior mean $\mu_{y|\mathbf{x}}$ of the nimiö y, given the piirre $\mathbf{x}$ [?], [?]. The corresponding Bayes risk is given by the posterior variance $\sigma^2_{y|\mathbf{x}}$ (see Fig. 75).

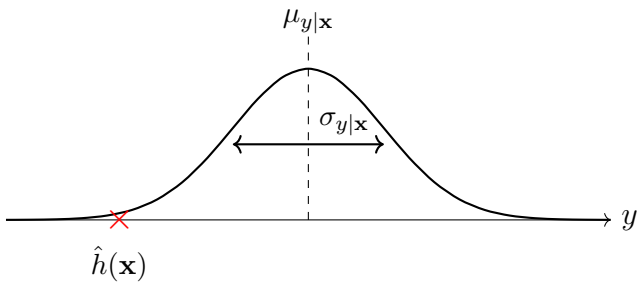


Fig. 75. If the piirre and the nimiö of a data point are drawn from a multivariate normal distribution, we can achieve the minimum risk (under neliövirhehäviö) by using the Bayes estimator $\mu_{y|\mathbf{x}}$ to predict the nimiö y of a data point with piirre $\mathbf{x}$. The corresponding minimum risk is given by the posterior variance $\sigma^2_{y|\mathbf{x}}$. We can use this quantity as a baseline for the average häviö of a trained malli $\hat{h}$.

See also: Bayes risk, Bayes estimator.

vertical federated learning (VFL) VFL refers to federoitu oppiminen applications where devices have access to different piirre of the same set of data points [?]. Formally, the underlying global tietoaineisto is

$$\mathcal{D}^{(\text{global})} := \left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right) \right\}.$$

We denote by $\mathbf{x}^{(r)} = \left(x_1^{(r)}, \ldots, x_{d'}^{(r)}\right)^T$, for $r = 1, \ldots, m$, the complete piirrevektori for the data points. Each device $i \in \mathcal{V}$ observes only a subset $\mathcal{F}^{(i)} \subseteq \{1, \ldots, d'\}$ of piirre, resulting in a local dataset $\mathcal{D}^{(i)}$ with piirrevektori

$$\mathbf{x}^{(i,r)} = \left(x_{j_1}^{(r)}, \ldots, x_{j_d}^{(r)}\right)^T.$$

Some of the devices may also have access to the nimiöt $y^{(r)}$, for $r = 1, \ldots, m$, of the global tietoaineisto (see Fig. 76).

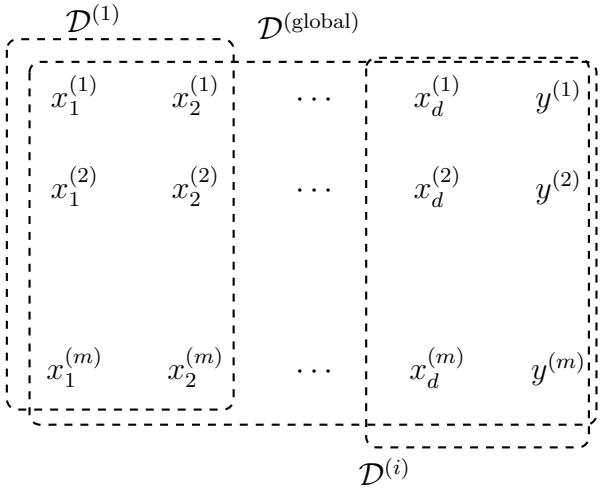


Fig. 76. VFL uses local datasets that are derived from the data points of a common global tietoaineisto. The local datasets differ in the choice of piirre used to characterize the data points.

One potential application of VFL is to enable collaboration between different healthcare providers. Each provider collects distinct types of measurements—such as blood values, electrocardiography, and lung X-rays—for the same patients. Another application is a national social

insurance system, where health records, financial indicators, consumer behavior, and mobility data are collected by different institutions. VFL enables joint learning across these parties while allowing well-defined levels of privacy protection.

See also: federoitu oppiminen, privacy protection.

weights Consider a parameterized hypothesis space $\mathcal{H}$. We use the term weights for numeric model parameters that are used to scale piirre or their transformations in order to compute $h^{(\mathbf{w})} \in \mathcal{H}$. A linear model uses weights $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$ to compute the linear combination $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. Weights are also used in neuroverkot to form linear combinations of piirre or the outputs of neurons in hidden layers (see Fig. 77).

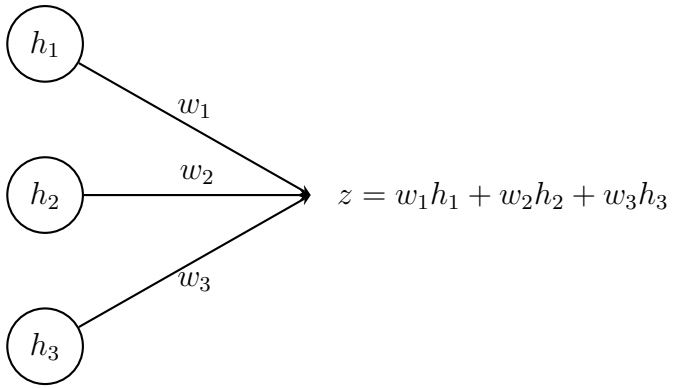


Fig. 77. A section of an ANN that contains a hidden layer with outputs (or activations) h_1,h_2, and h_3. These outputs are combined linearly to compute z, which can be used either as output of the ANN or as input to another layer.

See also: hypothesis space, model parameters, piirre, linear model, ANN,

layer, activation.

ydinfunktio Consider a set of data points, each represented by a piirevektori $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ denotes the feature space. A (real-valued) kernel is a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that assigns to every pair of piirrevektori $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ a real number $K\big(\mathbf{x}, \mathbf{x}'\big)$. This value is typically interpreted as a similarity measure between $\mathbf{x}$ and $\mathbf{x}'$. The defining property of a kernel is that it is symmetric, i.e., $K\big(\mathbf{x}, \mathbf{x}'\big) = K\big(\mathbf{x}', \mathbf{x}\big)$, and that for any finite set of piirrevektori $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$, the matriisi

$$\mathbf{K} = \begin{pmatrix} K\big(\mathbf{x}_1, \mathbf{x}_1\big) & K\big(\mathbf{x}_1, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_1, \mathbf{x}_n\big) \\ K\big(\mathbf{x}_2, \mathbf{x}_1\big) & K\big(\mathbf{x}_2, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_2, \mathbf{x}_n\big) \\ \vdots & \vdots & \ddots & \vdots \\ K\big(\mathbf{x}_n, \mathbf{x}_1\big) & K\big(\mathbf{x}_n, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_n, \mathbf{x}_n\big) \end{pmatrix} \in \mathbb{R}^{n \times n}$$

is psd. A kernel naturally defines a transformation of a piirevektori $\mathbf{x}$ into a function $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$. The function $\mathbf{z}$ maps an input $\mathbf{x}' \in \mathcal{X}$ to the value $K\big(\mathbf{x}, \mathbf{x}'\big)$. We can view the function $\mathbf{z}$ as a new piirevektori that belongs to a feature space $\mathcal{X}'$ that is typically different from $\mathcal{X}$. This new feature space $\mathcal{X}'$ has a particular mathematical structure, i.e., it is a reproducing kernel Hilbert space (RKHS) [?], [?]. Since $\mathbf{z}$ belongs to a RKHS, which is a vector space, we can interpret it as a generalized piirevektori. Note that a finite-length piirevektori $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T \in \mathbb{R}^d$ can be viewed as a function $\mathbf{x} : \{1, \ldots, d\} \to \mathbb{R}$ that assigns a real value to each index $j \in \{1, \ldots, d\}$.

See also: piirevektori, feature space, Hilbert space, kernel method.

yleistys Generalization refers to the ability of a malli trained on a training

set to make accurate ennuste on new unseen data points. This is a central goal of koneoppiminen and AI, i.e., to learn patterns that extend beyond the training set. Most koneoppiminen systems use ERM to learn a hypothesis $\hat{h} \in \mathcal{H}$ by minimizing the average häviö over a training set of data points $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$, which is denoted by $\mathcal{D}^{(\text{train})}$. However, success on the training set does not guarantee success on unseen data—this discrepancy is the challenge of generalization.

To study generalization mathematically, we need to formalize the notion of "unseen" data. A widely used approach is to assume a todennäköisyysmalli for data generation, such as the i.i.d. assumption. Here, we interpret data points as independent satunnaismuuttujat with an identical todennäköisyysjakauma $p(\mathbf{z})$. This todennäköisyysjakauma, which is assumed fixed but unknown, allows us to define the risk of a trained malli $\hat{h}$ as the expected häviö

$$\bar{L}\big(\hat{h}\big) = \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\big\{L(\hat{h}, \mathbf{z})\big\}.$$

The difference between risk $\bar{L}\big(\hat{h}\big)$ and empirical risk $\widehat{L}\big(\hat{h}|\mathcal{D}^{(\text{train})}\big)$ is known as the generalization gap. Tools from probability theory, such as concentration inequalities and uniform convergence, allow us to bound this gap under certain conditions [**?**].

Generalization without probability: Probability theory is one way to study how well a malli generalizes beyond the training set, but it is not the only way. Another option is to use simple deterministic changes to the data points in the training set. The basic idea is that a good malli $\hat{h}$ should be robust, i.e., its ennuste $\hat{h}(\mathbf{x})$ should not change much if we slightly change the piirre $\mathbf{x}$ of a data point $\mathbf{z}$. For example, an object

detector trained on smartphone photos should still detect the object if a few random pixels are masked [**?**]. Similarly, it should deliver the same result if we rotate the object in the image [**?**]. See Fig. 78 for a visual illustration.

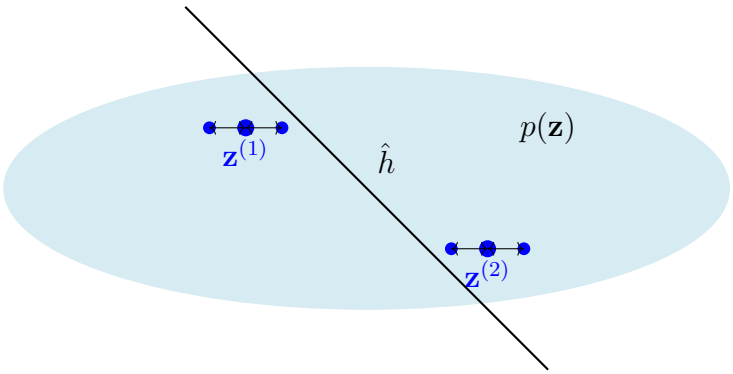


Fig. 78. Two data points $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}$ that are used as a training set to learn a hypothesis $\hat{h}$ via ERM. We can evaluate $\hat{h}$ outside $\mathcal{D}^{(\text{train})}$ either by an i.i.d. assumption with some underlying todennäköisyysjakauma $p(\mathbf{z})$ or by perturbing the data points.

See also: ERM, i.i.d. assumption, ylisovittaminen, validointi.

ylisovittaminen Consider an koneoppiminen method that uses ERM to learn a hypothesis with the minimum empirical risk on a given training set. Such a method is overfitting the training set if it learns a hypothesis with a low empirical risk on the training set but a significantly higher häviö outside the training set.

See also: ERM, yleistys, validointi, generalization gap.

zero-gradient condition Consider the unconstrained optimization problem

$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w})$ with a smooth and convex kohdefunktiot $f(\mathbf{w})$. A necessary and sufficient condition for a vector $\widehat{\mathbf{w}} \in \mathbb{R}^d$ to solve this problem is that the gradientti $\nabla f\left(\widehat{\mathbf{w}}\right)$ is the zero vector such that

$$\nabla f\left(\widehat{\mathbf{w}}\right) = \mathbf{0} \Leftrightarrow f\left(\widehat{\mathbf{w}}\right) = \min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}).$$

See also: optimization problem, smooth, convex, kohdefunktiot, vector, gradientti.

0/1 loss The 0/1 häviö $L^{(0/1)}\left((\mathbf{x}, y), h\right)$ measures the quality of a luokitin $h(\mathbf{x})$ that delivers a ennuste $\hat{y}$ (e.g., via thresholding (8)) for the nimiö y of a data point with piirre $\mathbf{x}$. It is equal to 0 if the ennuste is correct, i.e., $L^{(0/1)}\left((\mathbf{x}, y), h\right) = 0$ when $\hat{y} = y$. It is equal to 1 if the ennuste is wrong, i.e., $L^{(0/1)}\left((\mathbf{x}, y), h\right) = 1$ when $\hat{y} \neq y$.

See also: häviö, luokitin, ennuste, nimiö, data point, piirre.

Hakemisto

Viitteet

[1] W. Rudin, *Real and Complex Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1987.

[2] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1976.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 2013.

[4] G. H. Golub and C. F. Van Loan, "An analysis of the total least squares problem," *SIAM J. Numer. Anal.*, vol. 17, no. 6, pp. 883–893, Dec. 1980, doi: 10.1137/0717073.

[5] A. Klenke, *Probability Theory: A Comprehensive Course*, 3rd ed. Cham, Switzerland: Springer Nature, 2020.

[6] P. Billingsley, *Probability and Measure*, 3rd ed. New York, NY, USA: Wiley, 1995.

[7] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability*, 2nd ed. Belmont, MA, USA: Athena Scientific, 2008.

[8] A. Jung, *Machine Learning: The Basics*. Singapore, Singapore: Springer Nature, 2022.