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

5. tammikuuta 2026

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 [?].
$\mathbb{R}_+$	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 $0 \leq x \leq 1$.

$\arg\min_{\mathbf{w}\in\mathcal{C}} f(\mathbf{w})$	The set of minimizers for a real-valued ?? $f : \mathcal{C} \to \mathbb{R}$. See also: ??.
$\mathbb{S}^{(n)}$	The set of unit-?? ?? in $\mathbb{R}^{n+1}$. See also: ??, ??.
$\exp(a)$	The exponential ?? evaluated at the real number $a \in \mathbb{R}$. See also: ??.
$\log a$	The logarithm of the positive number $a \in \mathbb{R}_{++}$.
$f(\cdot) : \mathcal{A} \to \mathcal{B} : a \mapsto f(a)$	A ?? (or ??) 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 ?? of the ?? f and the set $\mathcal{B}$ is the ?? of f. Koneoppiminen aims to learn a ?? h that maps piirret $\mathbf{x}$ of a data point to a ennuste $h(\mathbf{x})$ for its nimiö y. See also: ??, ??, output, ??, ??, koneoppiminen, piirre, data point, ennuste, nimiö.
$\mathrm{epi}(f)$	The ?? of a real-valued ?? $f : \mathbb{R}^d \to \mathbb{R}$. See also: ??, ??.
$(a_r)_{r\in\mathbb{N}}$, $(a^{(r)})_{r\in\mathbb{N}}$, $\{a^{(r)}\}_{r\in\mathbb{N}}$	A ?? of elements. See also: ??.

$\mathbb{I}_{\mathcal{A}}(x)$	The indicator ?? of a set $\mathcal{A}$ delivers $f(x) = 1$ for any $x \in \mathcal{A}$ and $f(x) = 0$ otherwise. See also: ??.
$\frac{\partial f(w_1, \ldots, w_d)}{\partial w_j}$	The ?? (if it exists) of a real-valued ?? $f : \mathbb{R}^d \rightarrow \mathbb{R}$ with respect to w_j [**?**, Ch. 9]. See also: ??, ??.
$\nabla f(\mathbf{w})$	The ?? of a ?? real-valued ?? $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is the ?? $\nabla f(\mathbf{w}) = \big(\partial f/\partial w_1, \ldots, \partial f/\partial w_d\big)^T \in \mathbb{R}^d$ [**?**, Ch. 9]. See also: ??, ??, ??, ??.

Matrices and Vectors

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

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

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

Probability Theory

$\mathbf{x} \sim \mathbb{P}^{(\mathbf{z})}$	The ?? $\mathbf{x}$ is distributed according to the ?? $\mathbb{P}^{(\mathbf{z})}$ [**?**], [**?**]. See also: ??, ??.
$\mathbb{E}_p\{f(\mathbf{z})\}$	The expectation of an ?? $f(\mathbf{z})$ that is obtained by applying a deterministic ?? f to an ?? $\mathbf{z}$ whose ?? is $\mathbb{P}^{(\mathbf{z})}$. If the ?? is clear from context, we just write $\mathbb{E}\{f(\mathbf{z})\}$. See also: expectation, ??, ??, ??.
$\mathrm{cov}\,(x, y)$	The ?? between two real-valued ?? defined over a common ??. See also: ??, ??, ??.
$\mathbb{P}^{(\mathbf{x},y)}$	A (joint) ?? of an ?? whose realizationt are data points with piirteet $\mathbf{x}$ and nimiö y. See also: ??, ??, realization, data point, piirre, nimiö.
$\mathbb{P}^{(y\|\mathbf{x})}$	A ?? of an ?? y given (or conditioned on) the value of another ?? $\mathbf{x}$ [**?**, Sec. 3.5]. See also: ??, ??.
$\mathbb{P}\left(\mathcal{A}\right)$	The todennäköisyys of the ?? ?? $\mathcal{A}$. See also: todennäköisyys, ??, ??.

$M_x\left(t\right)$	The ?? of an ?? x. See also: ??, ??.
$\mathbb{P}^{(\mathcal{D})}$	The empirical distribution of a tietoaineisto $\mathcal{D}$. See also: empirical distribution, tietoaineisto, uusio-otanta.
$\mathbb{P}^{(\mathbf{x};\mathbf{w})}$	A parameterized ?? of an ?? $\mathbf{x}$. The ?? depends on a parameter ?? $\mathbf{w}$. For example, $\mathbb{P}^{(\mathbf{x};\mathbf{w})}$ could be a ?? with the parameter ?? $\mathbf{w}$ given by the entries of the ?? ?? $\mathbb{E}\{\mathbf{x}\}$ and the ?? $\mathbb{E}\Big\{\big(\mathbf{x}-\mathbb{E}\{\mathbf{x}\}\big)\big(\mathbf{x}-\mathbb{E}\{\mathbf{x}\}\big)^{T}\Big\}$. See also: ??, parameter, ??.
$\mathcal{N}(\mu,\sigma^2)$	The ?? of a ?? $x \in \mathbb{R}$ with ?? (or expectation) $\mu = \mathbb{E}\{x\}$ and ?? $\sigma^2 = \mathbb{E}\big\{(x-\mu)^2\big\}$. See also: ??, ??.
$\mathcal{N}(\boldsymbol{\mu},\mathbf{C})$	The ?? of a ??-valued ?? $\mathbf{x} \in \mathbb{R}^d$ with ?? (or expectation) $\boldsymbol{\mu} = \mathbb{E}\{\mathbf{x}\}$ and ?? $\mathbf{C} = \mathbb{E}\big\{\big(\mathbf{x}-\boldsymbol{\mu}\big)\big(\mathbf{x}-\boldsymbol{\mu}\big)^{T}\big\}$. See also: ??, ??.
Δ^k	The ??, which consists of all ?? $\mathbf{p} = \big(p_1, \ldots, p_k\big)^{T} \in \mathbb{R}^k$ with nonnegative entries that sum to one, i.e., $p_c \geq 0$ for $c = 1, \ldots, k$ and $\sum_{c=1}^{k} p_c = 1$. See also: ??.

$H(x)$	The ?? of a ?? x. See also: ??, ??.
Ω	A sample space of all possible ?? of a ??. See also: ??.
Σ	A collection of ?? subsets of a sample space Ω. See also: sample space, ??.
$\mathcal{P}$	A ?? that consists of a sample space Ω, a ?? Σ of ?? subsets of Ω, and a ?? $\mathbb{P}(\cdot)$. See also: sample space, ??, ??.

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 piirteet 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 ??'s entries are the individual piirteet of a data point. See also: piirrevektori, data point, ??, piirre.
$\mathcal{X}$	The feature space $\mathcal{X}$ is the set of all possible values that the piirteet $\mathbf{x}$ 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 ?? whose entries are the individual piirteet of a data point. We need two different symbols to distinguish between raw and learned piirteet [**?**, Ch. 9]. See also: ??, piirre, data point.
$\mathbf{x}^{(r)}$	The piirrevektori of the rth data point within a tietoaineisto. See also: piirrevektori, 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 piirteet 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 method.
$h(\cdot)$	A hypothesis ?? that maps the piirteet of a data point to a ennuste $\hat{y} = h(\mathbf{x})$ for its nimiö y. See also: hypothesis, ??, 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 ?? $h : \mathcal{X} \to \mathcal{Y}$. See also: hypothesis, ??.
$\mathcal{H}$	A hypothesis space or model used by an koneoppiminen method. The hypothesis space consists of different hypothesis ?? $h : \mathcal{X} \to \mathcal{Y}$, between which the koneoppiminen method must choose. See also: hypothesis space, model, koneoppiminen, hypothesis, ??.

$d_{\text{eff}}(\mathcal{H})$	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 ?? if it is learned from data points being ?? themselves. See also: harha, hypothesis, parameter, ??, data point.
V	The ?? of a learned hypothesis $\hat{h}$, or its parameters. Note that $\hat{h}$ becomes an ?? if it is learned from data points being ?? themselves. See also: ??, hypothesis, parameter, ??, data point.
$L((\mathbf{x}, y), h)$	The loss 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 piirrevektori $\mathbf{x}$ of the data point. See also: loss, nimiö, data point, ennuste, hypothesis, piirrevektori.
E_v	The validointivirhe of a hypothesis h, which is its average loss incurred over a validation set. See also: validointivirhe, hypothesis, loss, validation set.
$\widehat{L}\big(h\|\mathcal{D}\big)$	The empiirinen riski, or average loss, incurred by the hypothesis h on a tietoaineisto $\mathcal{D}$. See also: empiirinen riski, loss, hypothesis, tietoaineisto.

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

$\mathbf{w}$	A parameter ?? $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$ of a model, e.g., the weights of a lineaarinen malli or an ANN. See also: parameter, ??, model, weights, lineaarinen malli, ANN.
$h^{(\mathbf{w})}(\cdot)$	A hypothesis ?? that involves tunable model parameters $w_1, \ldots, w_d$ stacked into the ?? $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$. See also: hypothesis, ??, model parameter, ??.
$\phi(\cdot)$	A feature map $\phi : \mathcal{X} \to \mathcal{X}' : \mathbf{x} \mapsto \phi\big(\mathbf{x}\big)$ that transforms the piirrevektori $\mathbf{x}$ of a data point into a new piirrevektori $\mathbf{x}' = \phi\big(\mathbf{x}\big) \in \mathcal{X}'$. See also: feature map.
$K\big(\cdot, \cdot\big)$	Given some feature space $\mathcal{X}$, a ydinfunktio is a ?? $K : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ that is ??. See also: feature space, ydinfunktio, ??, ??.
$\mathrm{VCdim}\,(\mathcal{H})$	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 ?? 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 oppimistehtävät. See also: ??, device, FL network, tietoaineisto, oppimistehtävä.
$i \in \mathcal{V}$	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 ?? $\mathcal{G}$. See also: Laplacian matrix, ??.
$\mathbf{L}^{(\mathcal{C})}$	The Laplacian matrix of the induced ?? $\mathcal{G}^{(\mathcal{C})}$. See also: Laplacian matrix, ??.
$\mathcal{N}^{(i)}$	The naapurusto of the node i in a ?? $\mathcal{G}$. See also: naapurusto, ??.
$d^{(i)}$	The weighted ?? $d^{(i)} := \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'}$ of node i. See also: ??.
$d^{(\mathcal{G})}_{\max}$	The ?? weighted ?? of a ?? $\mathcal{G}$. See also: ??, ??, ??.

$\mathcal{D}^{(i)}$	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 piirteet 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 parameter, device, FL network.
$L_i\left(\mathbf{w}\right)$	The local loss function used by device i to measure the usefulness of some choice $\mathbf{w}$ for the local model parameters. See also: loss function, device, model parameter.
$L^{(\mathrm{d})}\left(\mathbf{x}, h\big(\mathbf{x}\big), h'\big(\mathbf{x}\big)\right)$	The loss incurred by a hypothesis h' on a data point with piirteet $\mathbf{x}$ and nimiö $h\big(\mathbf{x}\big)$ that is obtained from another hypothesis. See also: loss, hypothesis, data point, piirre, nimiö.

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

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

See also: ??, stacking, model parameter.

Tools

empirical distribution Consider a tietoaineisto $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ consisting of m distinct data points, each characterized by the piirrevektori $\mathbf{x}^{(r)} \in \mathcal{X}$ for $r = 1, \ldots, m$. For a given ?? Σ over the feature space $\mathcal{X}$, the empirical distribution of $\mathcal{D}$ is the ?? $\mathbb{P}^{(\mathcal{D})}$ defined via

$$\mathbb{P}^{(\mathcal{D})}(\mathcal{A}) = (1/m)\big|\big\{r : \mathbf{x}^{(r)} \in \mathcal{A}\big\}\big|, \quad \text{for any ?? } \mathcal{A} \in \Sigma.$$

In other words, the empirical distribution assigns to any ?? set $\mathcal{A} \in \Sigma$ the fraction of data points in $\mathcal{D}$ that fall into $\mathcal{A}$. If the feature space is ordered, the empirical distribution can also be characterized by its empirical ??

$$F^{(\mathcal{D})}(\mathbf{x}) = (1/m)\big|\big\{r : \mathbf{x}^{(r)} \preceq \mathbf{x}\big\}\big|, \quad \text{for any } \mathbf{x} \in \mathcal{X}$$

where $\preceq$ denotes the ordering relation on $\mathcal{X}$.

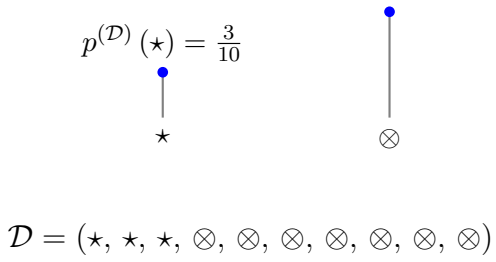


Fig. 1. A tietoaineisto $\mathcal{D}$ consisting of $m = 10$ data points, each characterized by a value from the finite feature space $\mathcal{X} = \{\star, \otimes\}$. The empirical ?? $p^{(\mathcal{D})}(\mathbf{x})$ assigns to each possible value $\mathbf{x} \in \mathcal{X}$ the fraction of data points in $\mathcal{D}$ whose piirre takes on this value. Here, three out of ten data points take on the piirre value $\star$, resulting in $p^{(\mathcal{D})}(\star) = 3/10$.

If the feature space $\mathcal{X}$ is finite, the empirical distribution of $\mathcal{D}$ can also be characterized by the empirical ??

$$p^{(\mathcal{D})}(\mathbf{x}) = (1/m)\big|\big\{r : \mathbf{x}^{(r)} = \mathbf{x}\big\}\big|, \quad \text{for any } \mathbf{x} \in \mathcal{X}.$$

See also: ??, ??.

Machine Learning Concepts

absolute error loss Consider a data point with piirteet $\mathbf{x} \in \mathcal{X}$ and numeric nimiö $y \in \mathbb{R}$. As its name suggests, the absolute error loss 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. **??** depicts the absolute error loss for a fixed data point with piirrevektori $\mathbf{x}$ and nimiö y. It also indicates the loss values incurred by two different hypotheses h' and h''. Similar to the neliövirhehäviö, the absolute error loss is also a ?? ?? of the ennuste $\hat{y} = h(\mathbf{x})$. However, in contrast to the neliövirhehäviö, the absolute error loss is ??, as it is not ?? at the optimal ennuste $\hat{y} = y$. This property makes empirical risk minimization (ERM)-based methods using the absolute error loss computationally more demanding [**?**], [**?**]. To build intuition, it is useful to consider the two hypotheses depicted in Fig. **??**. 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 ?? that is based on local approximations of the loss function (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 ?? of the ??. Besides the increased computational complexity, using absolute error loss 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 loss 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 loss is much smaller compared with the effect on the solution of ERM with neliövirhehäviö.

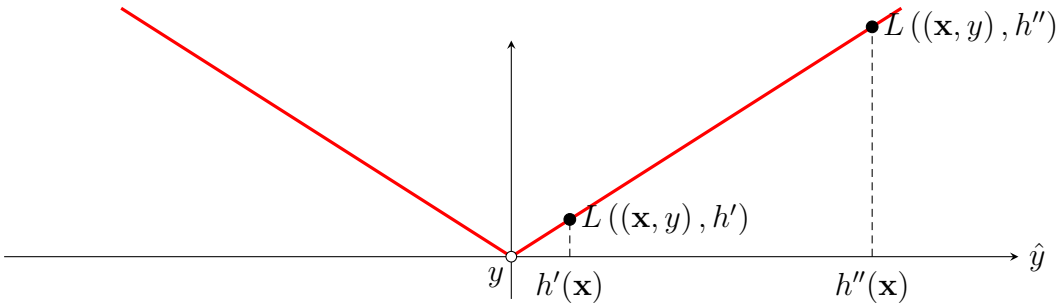


Fig. 2. For a data point with numeric nimiö $y \in \mathbb{R}$, the absolute error $|y - h(\mathbf{x})|$ can be used as a loss function to guide the learning of a hypothesis h.

See also: data point, piirre, nimiö, loss, ERM, subgradient descent, least absolute deviation regression.

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) aktivointifunktio to a weighted sum of its inputs. See also: ANN, deep net.

adaptive boosting (AdaBoost) AdaBoost is a specific boosting algoritmi that combines base learners sequentially [**?**], [**?**], [**?**]. The core idea of AdaBoost is to use the ennuste errors of the current base learner for sample weighting in the next base learner. In particular, the tth base learner learns a hypothesis $\hat{h}^{(t)}$ by weighted ERM with ?? weights $q^{(r)}$. The ennuste errors of $\hat{h}^{(t)}$ are then used to update the ?? weights by

increasing the weights of data points that have been predicted poorly (i.e., with large loss) by $\hat{h}^{(t)}$. The updated ?? weights are then used in the next base learner to learn $\hat{h}^{(t+1)}$. The ultimate hypothesis $\widetilde{h}^{(R)}$ delivered after R iterations is a linear combination of the hypotheses $\hat{h}^{(1)}, \ldots, \hat{h}^{(R)}$. AdaBoost can be interpreted as a generalized ?? [**?**, Ch. 10.4]

$$\widetilde{h}^{(t)} = \widetilde{h}^{(t-1)} + \eta^{(t)} \hat{h}^{(t)}.$$

This generalized ?? involves a oppimisnopeus $\eta^{(t)}$, which controls the amount of modification of the current hypothesis.

See also: boosting, sample weighting, ERM.

aktivointifunktio Each artificial neuron within an ANN is assigned an activation ?? $\sigma(\cdot)$ 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, rectified linear unit (ReLU).

algoritmi An algorithm is a precise, step-by-step specification for producing an output from a given input within a finite number of well-defined computational steps [**?**]. For example, a gradient-based method for lineaarinen regressio is an algorithm that explicitly describes how to map a given training set into model parameters through a sequence of ??. The precise form of an algorithm depends on the available computational infrastructure. For example, if this infrastructure allows us to compute an ??, then we can define a lineaarinen regressio algorithm using the normal equations. In contrast, if the computational infrastructure only allows

basic arithmetic (i.e., multiplication and addition), the normal equations need to be somehow translated into a sequence of arithmetic operations (e.g., as in gradient-based methods). 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: lineaarinen regressio, training set, model parameter, ??, output, model, ??.

alisovittaminen Consider an koneoppiminen method applying ERM to learn a hypothesis that minimizes the empiirinen riski on a given training set. The method is said to underfit if it fails to achieve a sufficiently low empiirinen riski on the training set. Underfitting typically occurs when the chosen model is too simple to capture the underlying relationship between piirteet and nimiöt.

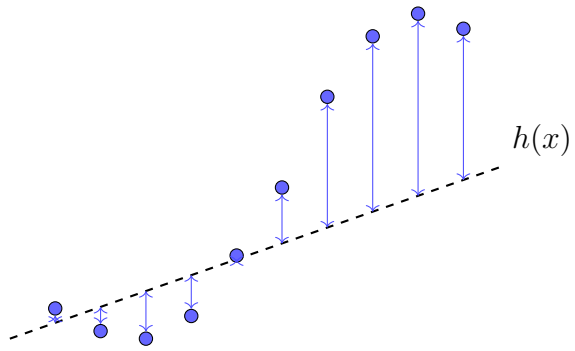


Fig. 3. No linear hypothesis h can capture the relationship between piirteet and nimiöt for the depicted training set. Thus, any method that uses a lineaarinen malli will underfit this training set.

For example, an koneoppiminen method using a lineaarinen malli on data with a highly nonlinear relationship between piirteet and nimiö will not be able to learn a hypothesis with small average loss on the training set, let alone a low riski.

See also: training set, model, riski, ylisovittaminen.

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 model. Users—whether humans or machines—can submit the piirrevektori of a data point and receive a corresponding ennuste. Suppose a trained koneoppiminen model 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 model or its training. In practice, the model

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 model, while another team handles integration and user interaction. Publishing a trained model via an API also offers practical advantages. For instance, the server can centralize computational resources that are required to compute ennusteet. Furthermore, the internal structure of the model remains hidden—which is useful for protecting intellectual property or trade secrets. However, APIs are not without riski. Techniques such as model inversion can potentially reconstruct a model from its ennusteet using carefully selected piirrevektorit.

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

area under the curve (AUC) The AUC is a quantitative ?? of the usefulness of a binary luokitin [**?**]. It is defined (using the natural ?? of the Euclidean space $\mathbb{R}^2$) as the area under the receiver operating characteristic (ROC) curve.

See also: luokitin, Euclidean space, ROC.

artificial intelligence (AI) 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 model to predict optimal actions. These ennusteet are computed from observations about the state of the environment. The choice of loss function sets AI applications apart from

more basic koneoppiminen applications. AI systems rarely have access to a labeled training set that allows the average loss to be measured for any possible choice of model parameters. Instead, AI systems use observed reward signals to estimate the loss incurred by the current choice of model parameters.

See also: koneoppiminen, ??.

artificial neural network (ANN) An artificial neural network (ANN) is a graphical (signal-flow) representation of a hypothesis that maps piirteet of a data point at its input to a ennuste for the corresponding nimiö at its output. The fundamental computational unit of an ANN is the artificial neuron, which applies an aktivointifunktio $\sigma(\cdot)$ to the sum of its inputs. The output of a neuron can be used either as the final output of the ANN or as an input to other neurons. A key design parameter of an ANN is its connectivity structure (or architecture), i.e., which neuron outputs are connected to which neuron inputs. As illustrated in Fig. **??**, we can represent an ANN as an ??. One widely used type of ANN are deep nets where neurons form consecutive layers. In a deep net, the outputs of neurons in a given layer are typically only connnected to the inputs of the neurons in a consecutive layer. Sometimes it is useful to add shortcut or skip connections that directly connect the outputs of neurons in one layer to the inputs of neurons in a non-consecutive layer [**?**, **?**].

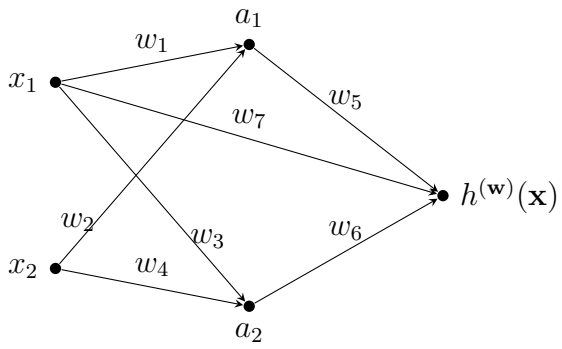


Fig. 4. An ANN can be represented as a weighted ?? with nodes representing neurons or piirteet of a data point. Piirteet can be viewed as trivial neurons without input and fixed ouput given by the piirre value. The weighted directed edges indicate how neuron outputs are used as inputs to other neurons. The edge weights are tunable model parameters and are used to scale the inputs to the neurons. The output of some neurons are used as the ennuste $h^{(\mathbf{w})}(\mathbf{x})$.

See also: nimiö, aktivointifunktio, layer, deep net.

attention Some koneoppiminen applications involve data points composed of smaller units, referred to as tokens. For example, a sentence consists of words, an image of pixel patches, and a network of nodes. In general, the tokens that constitute a single data point are not independent of one another. Instead, each token of a data point depends on (or pays attention to) specific other tokens. ?? provide a principled framework for representing and analyzing such dependencies [**?**]. Attention mechanisms use a more direct approach without explicit reference to a ??. The idea is to represent the relationship between two tokens i and i' using a parameterized ?? $f^{(\mathbf{w})}(i, i')$, where the parameters $\mathbf{w}$ are learned via a

variant of ERM. Practical attention mechanisms differ in their precise choice of attention model $f^{(\mathbf{w})}(i, i')$ 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 ?? associated with each token i, i.e., a query ?? $\mathbf{q}^{(i)}$ and a key ?? $\mathbf{k}^{(i')}$. For a given token i with query $\mathbf{q}^{(i)}$, and another token i' with key $\mathbf{k}^{(i')}$, the quantity $\left(\mathbf{q}^{(i)}\right)^{\top}\mathbf{k}^{(i')}$ indicates the extent to which token i attends to (or depends on) token i' (see Fig. **??**).

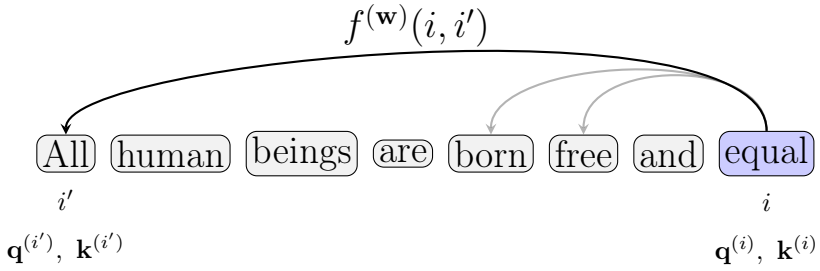


Fig. 5. Attention mechanisms learn a parameterized ?? $f^{(\mathbf{w})}(i, i')$ to measure how much token i attends to token i'. One widely used construction of $f^{(\mathbf{w})}(i, i')$ uses query and key ??, denoted by $\mathbf{q}^{(i)}$ and $\mathbf{k}^{(i)}$, assigned to each token i [**?**].

See also: token, ??, natural language processing (NLP).

autoenkoodaaja An autoencoder is an koneoppiminen method that simultaneously learns an encoder ?? $h \in \mathcal{H}$ and a decoder ?? $h^* \in \mathcal{H}^*$. Different autoencoders use different models $\mathcal{H}, \mathcal{H}^*$, e.g., ANNs with different architectures. The special case of an autoencoder using (??-valued) lineaariset mallit for $\mathcal{H}, \mathcal{H}^*$ results in principal component analysis (PCA).

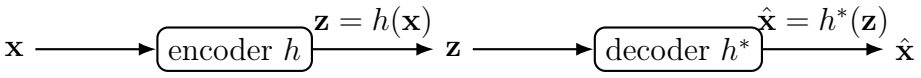


Fig. 6. Autoencoder with an encoder h mapping $\mathbf{x} \mapsto \mathbf{z}$ and a decoder h^* mapping $\mathbf{z} \mapsto \hat{\mathbf{x}}$.

The training of the encoder and decoder can be implemented via ERM using a loss that measures the deviation of the reconstructed piirrevektori $h^*\big(h\big(\mathbf{x}\big)\big)$ from the original piirrevektori $\mathbf{x}$.

See also: encoder, piirreoptimointi, ulottuvuuksien vähentäminen.

backdoor A backdoor hyökkäys refers to the intentional manipulation of an koneoppiminen training process. The attacker might perturb the training set (i.e., through data poisoning) or the ?? used by an ERM-based method. The goal of a backdoor hyökkäys is to nudge the learned hypothesis $\hat{h}$ toward specific ennusteet for a certain subset $\mathcal{T} \subset \mathcal{X}$ of the feature space. Any piirrevektori $\mathbf{x} \in \mathcal{T}$ serves as a key (or trigger) to unlock a backdoor in the sense of delivering anomalous ennusteet. The trigger pattern $\mathcal{T}$ and corresponding anomalous ennuste $\hat{h}(\mathbf{x})$, for $\mathbf{x} \in \mathcal{T}$, are only known to the attacker.

See also: hyökkäys, data poisoning.

backpropagation Backpropagation is an algoritmi for computing the ?? $\nabla_{\mathbf{w}} f(\mathbf{w})$ of an kohdefunktio $f(\mathbf{w})$ that depends on the model parameters $\mathbf{w}$ of an ANN. One example of such an kohdefunktio is the average loss incurred by the ANN on a batch of data points. This algoritmi is a direct application of the chain rule from calculus to efficiently

compute ?? of the loss function with respect to the model parameters. Backpropagation consists of two consecutive phases, also illustrated in Fig. **??**. 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 loss function, 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 ?? with respect to the ANN parameters $w_1, \ldots, w_d$ constitute the ?? $\nabla f(\mathbf{w})$, which can be used, in turn, to implement a ??.

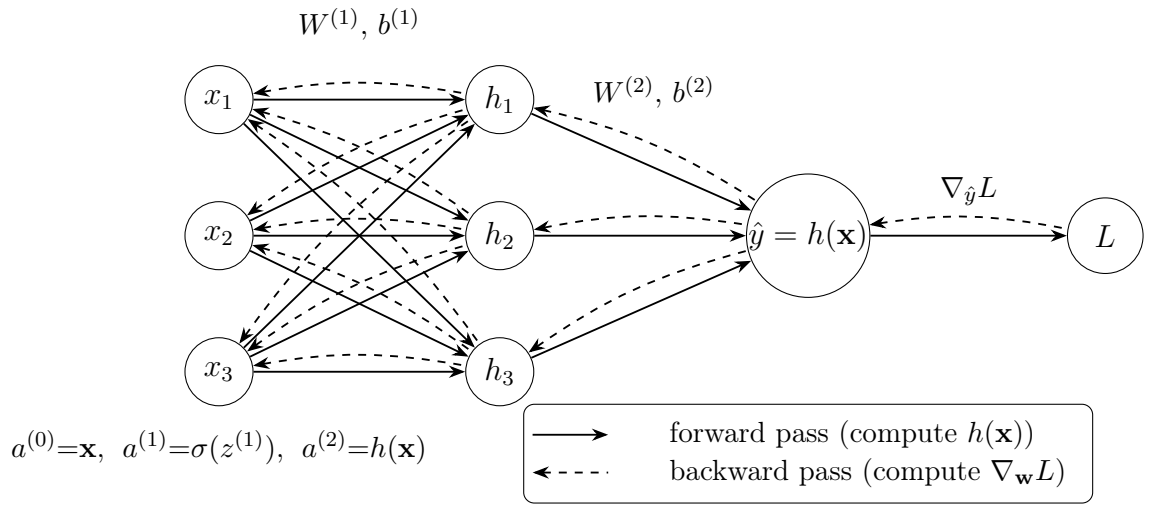


Fig. 7. Solid arrows show the forward pass (i.e., data flow and loss calculation), while dashed arrows show the ?? correction flow during the backward pass for updating the parameters $W^{(x)}$, $b^{(x)}$.

See also: ANN, loss function, gradient descent (GD), ??.

bagging Bagging is an ensemble technique where base learners use perturbed copies

$$\widetilde{\mathcal{D}}^{(1)}, \ldots, \widetilde{\mathcal{D}}^{(M)}$$

of the original training set $\mathcal{D}$ [?]. Each base learner delivers a potentially different hypothesis,

$$\hat{h}^{(1)}, \ldots, \hat{h}^{(M)}.$$

The hypothesis delivered by the overall method is obtained by aggregating the hypotheses $\hat{h}^{(1)}, \ldots, \hat{h}^{(M)}$ using some aggregation rule. For luokittelu methods, the rule is typically a majority vote, while for regressio methods, it amounts to averaging.

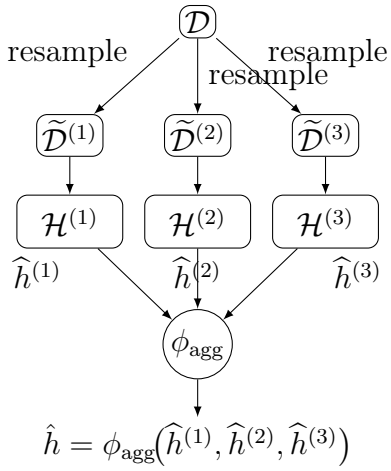


Fig. 8. An example of bagging where three base learners use perturbations $\widetilde{\mathcal{D}}^{(1)}, \ldots, \widetilde{\mathcal{D}}^{(3)}$ of the original training set $\mathcal{D}$ to learn the hypotheses $\widehat{h}^{(1)}, \ldots, \widehat{h}^{(3)}$. The final hypothesis is obtained by aggregating these individual hypotheses via some aggregation rule ϕ_{agg}.

See also: ensemble, vakaus, uusio-otanta.

base learner A base learner is an koneoppiminen method that is part of an ensemble method.

See also: ensemble, bagging, stacking, boosting.

batch In the context of stochastic gradient descent (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 ?? of opetusvirhe and, in turn, to update the model parameters.

See also: SGD, training set, data point, ??, opetusvirhe, model parameter.

batch learning In batch learning (also known as offline learning), the koneoppiminen model 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 algoritmi, resulting in a model that can make ennusteet. Since these tietoaineistot tend to be large, training is computationally expensive and time-consuming, so it is typically performed offline. After learning, the model will be static and will not adapt to new data automatically. Updating the model with new information requires retraining the model entirely. Once the model has been trained, it is launched into production where it cannot be updated. Training a model can take many hours, so many models 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 model 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, model, tietoaineisto, online learning.

Bayes estimator Consider a ?? with a joint ?? $p(\mathbf{x}, y)$ over the piirteet $\mathbf{x}$ and the nimiö y of a data point. For a given loss function $L(\cdot, \cdot)$, we refer to a hypothesis h as a Bayes estimator if its riski $\mathbb{E}\{L((\mathbf{x}, y), h)\}$ is the ?? achievable riski [**?**]. Note that whether a hypothesis qualifies as a Bayes estimator depends on the underlying ?? and the choice for the loss function $L(\cdot, \cdot)$.

See also: ??, hypothesis, riski.

Bayes risk Consider a ?? for a koneoppiminen application where each data point is interpreted as a ??. The ?? includes a ?? for the piirteet $\mathbf{x}$ and nimiö y of a data point. The Bayes riski is the ?? possible riski that can be achieved by any hypothesis $h : \mathcal{X} \to \mathcal{Y}$. Any hypothesis that achieves the Bayes riski is referred to as a Bayes estimator [**?**].

See also: ??, riski, Bayes estimator.

binary classification Binary luokittelu is luokittelu with two nimiöt. The nimiöt are usually defined as $\{-1, 1\}$ or $\{0, 1\}$.

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

binary cross-entropy (BCE) The BCE loss is the special case of cross-entropy loss for a binary luokittelu problem.

See also: cross-entropy, luokittelu.

binääritappio The 0/1 loss $L^{(0/1)}\left((\mathbf{x}, y), h\right)$ measures the quality of a luokitin $h(\mathbf{x})$ that delivers a ennuste $\hat{y}$ (e.g., via thresholding (**??**)) for the nimiö y of a data point with piirteet $\mathbf{x}$. 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: loss, luokitin, ennuste, tarkkuus, nimiö, data point, piirre.

boosting Boosting is an iterative ?? to learn an accurate hypothesis ?? (or strong learner) by sequentially combining less accurate base learners (referred to as weak learners) [**?**], [**?**], [**?**], [**?**, Ch. 10]. Boosting can be understood as a yleistys of gradient-based methods for ERM using parametric models and ?? loss functiont [**?**]. In particular, starting from an initialization $\widetilde{h}$, boosting methods construct a ?? of hypotheses $\widetilde{h}^{(t)}$, $t = 1, \ldots,$ via a generalized ??

$$\widetilde{h}^{(t)} = \widetilde{h}^{(t-1)} + \eta^{(t)} \hat{h}^{(t)}.$$

Here, $\eta^{(t)}$ denotes a oppimisnopeus and $\hat{h}^{(t)}$ is provided by the tth base learner. Comparing the above update with the plain ?? suggests that we view $\hat{h}^{(t)}$ as a (negative) generalized ??. Boosting methods differ in their choice of base learners for computing the generalized ?? $\hat{h}^{(t)}$.

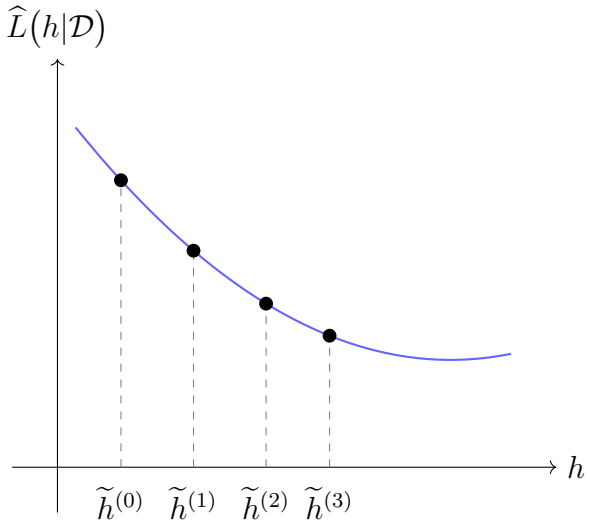


Fig. 9. Boosting methods construct a ?? of hypothesis ?? via a generalized ??. This generalized ?? uses the output of base learners.

See also: ensemble, adaptive boosting (AdaBoost), gradient boosting.

bootstrap aggregation See bagging.

cluster 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 ?? of similarity between data points is a design choice. If data points are characterized by Euclidean piirrevektorit $\mathbf{x} \in \mathbb{R}^d$, we can define the similarity between two data points via the Euclidean distance between their piirrevektorit. An example of such clusters is shown in Fig. **??**.

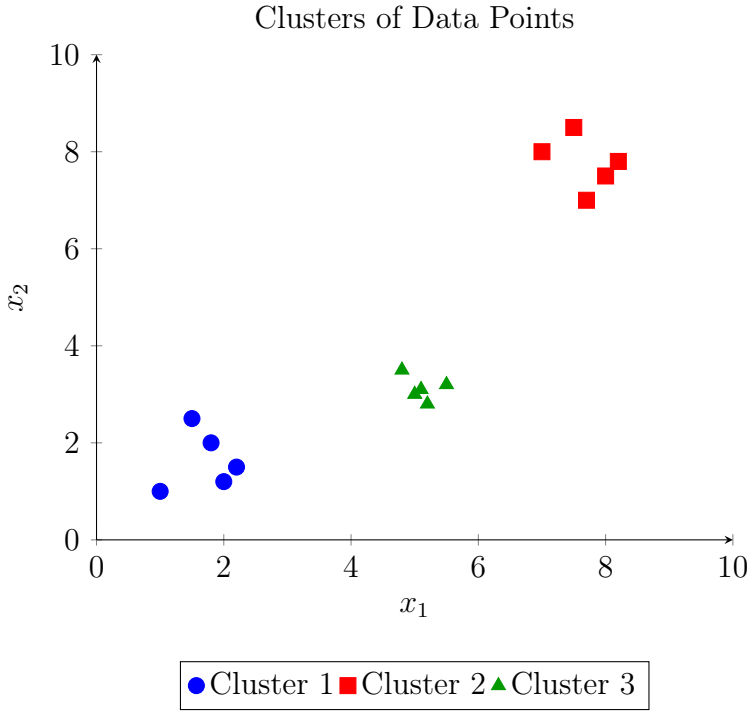


Fig. 10. Illustration of three clusters in a 2-D feature space. Each cluster groups data points that are more similar to each other than to those in other clusters, based on the Euclidean distance.

See also: data point, piirrevektori, Euclidean distance, feature space.

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

cluster centroid.

See also: clustering, piirrevektori, 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 clusters. Two devices in the same cluster generate local datasets with similar statistical properties. CFL pools the local datasets of devices in the same cluster to obtain a training set for a cluster-specific model. 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, cluster, graph clustering.

clustering Clustering methods decompose a given set of data points into a few subsets, which are referred to as clusters. Each cluster consists of data points that are more similar to each other than to data points outside the cluster. Different clustering methods use different ?? for the similarity between data points and different forms of cluster representations. The clustering method k-means uses the average piirrevektori of a cluster (i.e., the cluster ??) as its representative. A popular soft clustering method based on Gaussin sekoitemalli represents a cluster by a ??.

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

clustering assumption The clustering assumption postulates that data points in a tietoaineisto form a (small) number of groups or clusters.

Fig. 11. For data point with numeric piirrevektorit, we can use the average squared Euclidean distance to the nearest cluster centroids as a ?? of the clustering error.

Data points in the same cluster are more similar to each other than those outside the cluster [**?**]. We obtain different clustering methods by using different notions of similarity between data points.
See also: clustering, data point, tietoaineisto, cluster.

clustering error Consider a clustering method that decomposes a given tietoaineisto into clusters. The clustering error is a quantitative ?? of the usefulness of the clusters. Different clustering methods use different choices for the clustering error. For example, the hard clustering method k-means measures the clustering error via the average squared Euclidean distance between the piirrevektori of a data point and the nearest cluster centroid (see Figure **??**). Another construction for the clustering error can be based on a ?? such as the Gaussin sekoitemalli where the cluster centroids are parameters of the underlying ??.
See also: Gaussin sekoitemalli, k-means, ??, suurimman uskottavuuden menetelmä.

concept activation vector (CAV) Consider a deep net, consisting of se-

veral hidden layers, trained to predict the nimiö of a data point from its piirrevektori. One way to explain the behavior of the trained deep net is by using the activations of a hidden layer as a new piirrevektori $\mathbf{z}$. 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 ?? whose normal ?? is the CAV [**?**] for the concept $\mathcal{C}$.

See also: deep net, lineaarinen malli, ??, tulkittavuus, ??.

convex clustering Consider a tietoaineisto $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. ?? clustering learns ?? $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(m)}$ 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-?? (for $p \geq 1$). It turns out that many of the optimal ?? $\widehat{\mathbf{w}}^{(1)}, \ldots, \widehat{\mathbf{w}}^{(m)}$ coincide. A cluster then consists of those data points $r \in \{1, \ldots, m\}$ with identical $\widehat{\mathbf{w}}^{(r)}$ [**?**], [**?**].

See also: tietoaineisto, ??, clustering, ??, ??, cluster, data point.

coreset A coreset is a small subset of a larger tietoaineisto that approximates certain properties of the original tietoaineisto [**?**]. The construction of a coreset typically involves selecting representative data points and assigning them weights to reflect their importance in the original tietoaineisto (Fig. **??**).

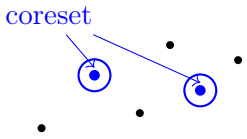


Fig. 12. A coreset (highlighted in blue) is a small subset of a larger tietoaineisto.

Coresets are particularly useful for koneoppiminen applications (such as clustering) involving large tietoaineistot, as they allow for efficient computation while preserving the essential characteristics of the data [**?**]. See also: tietoaineisto, data point, clustering.

cross-entropy Consider a multi-class luokittelu problem with a feature space $\mathcal{X}$ and a finite label space $\mathcal{Y} = \{1, \ldots, k\}$. A data point with a piirrevektori $\mathbf{x}$ is represented as a ?? $\mathbf{y} = (y_1, \ldots, y_k)^T$ over $\mathcal{Y}$, where y_c denotes the todennäköisyys that a randomly chosen data point with piirrevektori $\mathbf{x}$ has nimiö c. A hypothesis $h(\mathbf{x})$ outputs a predicted ?? $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_k)^T$. The associated cross-entropy loss is [**?**]

$$L\left((\mathbf{x}, \hat{\mathbf{y}}), h\right) := -\sum_{c=1}^{k} y_c \log \hat{y}_c.$$

The cross-entropy loss quantifies the dissimilarity between the true ?? $\mathbf{y}$ and the predicted ?? $\hat{\mathbf{y}}$. It is also a ?? of the expected number of bits required to encode nimiöt drawn from the true ?? $\mathbf{y}$ when using a coding scheme optimized for the predicted ?? $\hat{\mathbf{y}}$ [**?**].

Note that, for binary luokittelu (with $k = 2$), the cross-entropy loss reduces to the logistic loss when employing a parametric model with

model parameters $\mathbf{w}$ such that $\hat{y}_2/\hat{y}_1 = \exp(\mathbf{w}^T\mathbf{x})$. Moreover, the representation (**??**) of logistic loss requires encoding the label space $\{1, 2\}$ using the values -1 and 1.
See also: luokittelu, ??, logistic loss.

data In the context of koneoppiminen, the term data is often used as a synonym for tietoaineisto [**?**], [**?**]. The ISO/IEC 2382:2015 standard defines data as a "reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or processing" [**?**].
See also: tietoaineisto, data point, ??.

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 piirrevektorit (obtained by stacking pixel color intensities) are very different (see Fig. **??**). Data augmentation can be an efficient form of regularisointi.

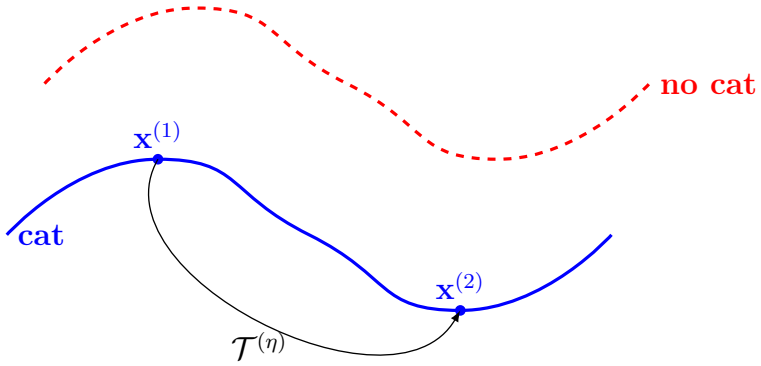


Fig. 13. Data augmentation exploits intrinsic symmetries of data points in some feature space $\mathcal{X}$. We can represent a symmetry by an ?? $\mathcal{T}^{(\eta)} : \mathcal{X} \to \mathcal{X}$, parameterized by some number $\eta \in \mathbb{R}$. For example, $\mathcal{T}^{(\eta)}$ might represent the effect of rotating a cat image by η degrees. A data point with piirrevektori $\mathbf{x}^{(2)} = \mathcal{T}^{(\eta)}\big(\mathbf{x}^{(1)}\big)$ must have the same nimiö $y^{(2)} = y^{(1)}$ as a data point with piirrevektori $\mathbf{x}^{(1)}$.

See also: data, data point, nimiö, piirrevektori, stacking, regularisointi, feature space.

data imputation See missing data.

data matrix The term data ?? is sometimes used as a synonym for the feature matrix $\mathbf{X} \in \mathbb{R}^{m \times d}$ of a tietoaineisto containing m data points, each characterized by a piirrevektori $\mathbf{x} \in \mathbb{R}^d$ [**?**]. In particular, when no nimiö information is available, the term data ?? highlights that the feature matrix fully characterizes the tietoaineisto.

See also: feature matrix, tietoaineisto, data point, piirrevektori.

data minimization principle European data protection regulation inclu-

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

See also: data.

data normalization Data normalization refers to transformations applied to the piirrevektorit of data points to improve the koneoppiminen method's laskennallinen ominaisuus or laskennallinen ominaisuus. For example, in lineaarinen regressio with gradient-based methods using a fixed oppimisnopeus, ?? depends on controlling the ?? of piirrevektorit in the training set. A common approach is to normalize piirrevektorit such that their ?? does not exceed one [**?**, Ch. 5].

See also: gradient-based method, oppimisnopeus, feature map.

data parallelism Data parallelism refers to a widely used class of distributed ?? for training an koneoppiminen model. Here, each participating device stores a full replica of the model parameters but processes a disjoint subset of the global tietoaineisto [**?**]. Compared to model parallelism, which distributes the model parameters across devices, data parallelism distributes the computational workload associated with large tietoaineistot. It is especially effective when the model fits into the memory of a single device, but the tietoaineisto or training complexity would be prohibitive without parallel processing.

See also: model parallelism, horizontal federated learning (HFL).

data point A data point is any object that conveys information [**?**]. Examples include students, radio signals, trees, images, ??, real numbers, or proteins. We describe data points of the same type by two categories of properties. The first category includes piirteet that are ?? 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 that typically require human expertise or domain knowledge to determine rather than being directly ??—associated with the data point. For a data point that represents a patient, a cancer diagnosis provided by a physician would serve as the nimiö. Fig. **??** depicts an image as an example of a data point along with its piirteet 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.

Piirteet:

- $x_1, \ldots, x_d$: 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. 14. Illustration of a data point consisting of an image. We can use different properties of the image as piirteet and higher level facts about the image as nimiöt.

The distinction between piirteet 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 piirteet.

See also: data, piirre, nimiö, tietoaineisto.

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

See also: hyökkäys, backdoor, denial-of-service attack, ??.

decision tree A decision tree is a flowchart-like representation of a hypothesis ?? h. More formally, a decision tree is a directed ?? containing a root node that reads in the piirrevektori $\mathbf{x}$ 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 piirteet $\mathbf{x}$. 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. **??** for visual illustrations.

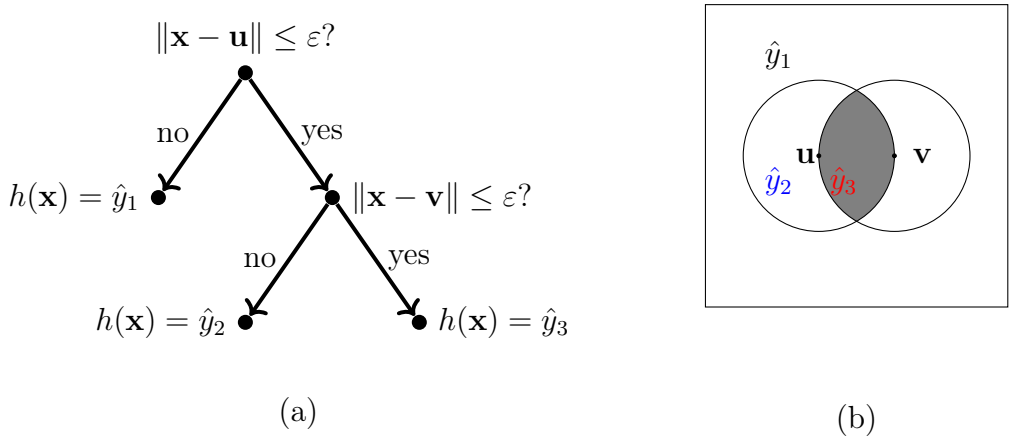


Fig. 15. (a) A decision tree is a flowchart-like representation of a piecewise constant hypothesis $h : \mathcal{X} \to \mathbb{R}$. Each piece is a päätösalue $\mathcal{R}_{\hat{y}} := \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = \hat{y}\}$. The depicted decision tree can be applied to numeric piirrevektorit, i.e., $\mathcal{X} \subseteq \mathbb{R}^d$. It is parameterized by the threshold $\varepsilon > 0$ and the ?? $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$. (b) A decision tree partitions the feature space $\mathcal{X}$ into päätösalueet. Each päätösalue $\mathcal{R}_{\hat{y}} \subseteq \mathcal{X}$ corresponds to a specific leaf node in the decision tree.

See also: päätösalue.

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 model [**?**].

See also: ANN, layer, model, large language model (LLM).

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

See also: data point, cluster, soft clustering, hard clustering.

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

See also: hyökkäys, data poisoning, model, data point.

density-based spatial clustering of applications with noise (DBSCAN) DBSCAN refers to a clustering algoritmi for data points that are characterized by numeric piirrevektorit. Like k-means and soft clustering via Gaussin sekoitemalli, DBSCAN also uses the Euclidean distances between piirrevektorit to determine the clusters. 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 piirrevektorit have a large Euclidean distance.

See also: clustering, k-means, Gaussin sekoitemalli, cluster, ??.

design matrix The term design ?? is a synonym for the feature matrix, particularly used in statistics [**?**], [**?**]. It collects the piirrevektorit of the data points in a tietoaineisto that is used for model training or validointi.

See also: feature matrix, piirrevektorit, data point, tietoaineisto.

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 model [**?**].

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

diagnosis Consider an ERM-based method that resulted in a trained model (or learned hypothesis) $\hat{h} \in \mathcal{H}$. We can diagnose the method by comparing the opetusvirhe E_t with the validointivirhe E_v incurred by $\hat{h}$ on the training set and the validation set.

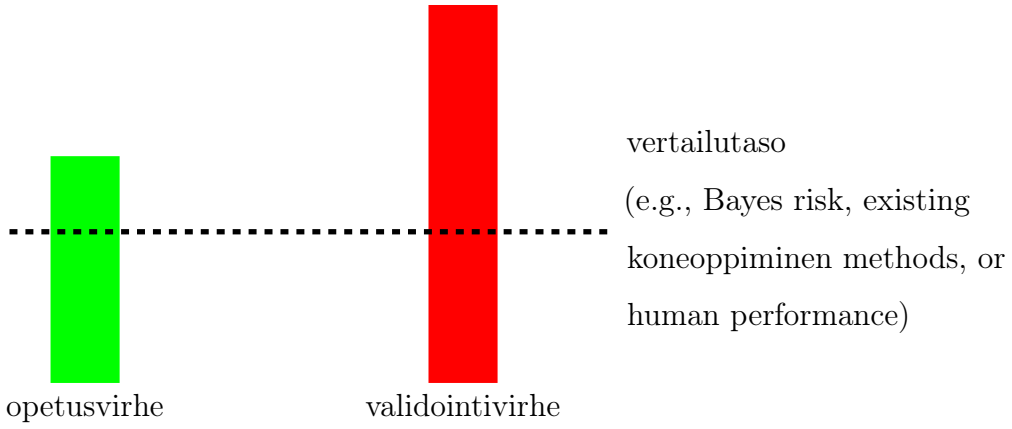


Fig. 16. We can diagnose an ERM-based koneoppiminen method by comparing its opetusvirhe with its validointivirhe. Ideally, both are on the same level as a vertailutaso.

See also: vertailutaso, validointi, k-fold cross-validation (k-fold CV), yleistys.

differentiaalinen yksityisyys 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 ennusteet for specific data points. DP is a precise ?? of privacy leakage incurred by revealing the output. Roughly speaking, an koneoppiminen method is differentially private if the ?? of the output $\mathcal{A}(\mathcal{D})$ remains largely unchanged when the sensitive attribute of one data point in the training set is changed. Note that DP builds on a ?? for an koneoppiminen method, i.e., we interpret its output $\mathcal{A}(\mathcal{D})$ as the realization of an ??. The randomness in the output

can be ensured by intentionally adding the realization of an auxiliary ?? (i.e., adding noise) to the output of the koneoppiminen method.
See also: output, privacy leakage, sensitive attribute, yksityisyyshyökkäys, privacy funnel.

diffusion method A diffusion method is a generative koneoppiminen method that trains a model to approximate the reversal of a gradual ?? corruption process [**?**], [**?**]. Diffusion methods train models using a combination of a forward (diffusion) process and a learned reverse ??. In the forward process, random noise is incrementally added to clean representations of a data point, such as an image, an audio recording, or other types of data. Similar to a ??, the trained model can be applied to a noisy representation of a data point. The resulting ennuste is a denoised representation. What sets diffusion methods apart from generic ?? is the gradual nature of the corruption process used for model training.
See also: ??, training.

early stopping Consider an ERM-based method that uses an iterative ?? (such as GD) to learn model parameters by minimizing the empiirinen riski on a given training set. Early stopping refers to terminating the iterations even if they still substantially decrease the empiirinen riski on the training set. Instead of monitoring the kohdefunktio (which is the empiirinen riski on the training set), early stopping monitors the validointivirhe incurred by the model parameters in each iteration. Early stopping can be interpreted as an implementation of regularisointi via model pruning. Indeed, terminating an iterative ?? after a small number

of iterations restricts the set of model parameters that can be reached from the initialization (see Fig. **??**).

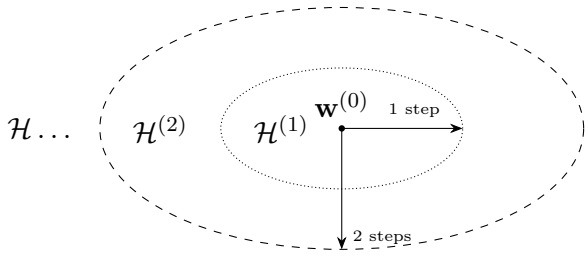


Fig. 17. A gradient-based method for ERM using a hypothesis space $\mathcal{H}$ defines a nested ?? of effective hypothesis spaces $\mathcal{H}^{(1)} \subseteq \mathcal{H}^{(2)} \subseteq \ldots \subseteq \mathcal{H}$. The effective hypothesis space $\mathcal{H}^{(t)}$ is determined by all model parameters that can be reached from the initialization $\mathbf{w}^{(0)}$ within t ??.

See also: regularisointi, gradient-based method, ylisovittaminen.

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

See also: hypothesis space, model parameter, ANN.

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

See also: riski, hypothesis, tietoaineisto, loss, data point.

empirical risk minimization (ERM) ERM is the ?? of selecting a hypothesis $\hat{h} \in \mathcal{H}$ that minimizes the average loss (or empiirinen riski) on a training set $\mathcal{D}$. The hypothesis is chosen from a hypothesis space (or model) $\mathcal{H}$. 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, model, and loss [**?**, Ch. 3]. Fig. **??** illustrates ERM for a lineaarinen malli and data points that are characterized by a single piirre x and a nimiö y. The hypothesis h is a ?? that predicts the nimiö of a data point as a linear ?? 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 loss (or empiirinen riski) incurred by the hypothesis h on the training set $\mathcal{D}$.

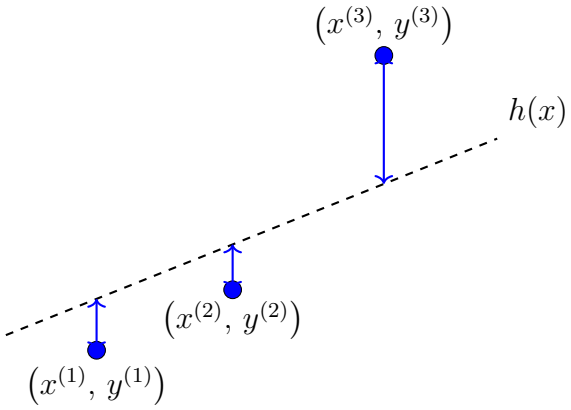


Fig. 18. ERM learns a hypothesis $h \in \mathcal{H}$ out of a model $\mathcal{H}$ by minimizing the average loss (or empiirinen riski) $1/m \sum_{r=1}^{m} L\left(\left(\mathbf{x}^{(r)}, y^{(r)}\right), h\right)$ incurred on a training set $\mathcal{D}$.

See also: ??, empiirinen riski, training set, loss, ??.

encoder See autoenkoodaaja.

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

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

ennustin A predictor is a real-valued hypothesis ??. Given a data point with piirteet $\mathbf{x}$, 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, ??, data point, piirre, ennuste, nimiö.

ensemble An ensemble method combines multiple koneoppiminen methods, each of those referred to as a base learner, to improve overall performance. The base learners can be ERM-based using different choices for the loss, model, and training set. By aggregating the ennusteet of base learners, ensemble methods can often achieve better performance than any single base learner. The aggregation can amount to averaging the ennusteet of base learners (in regressio) or using a majority vote (for luokittelu methods).

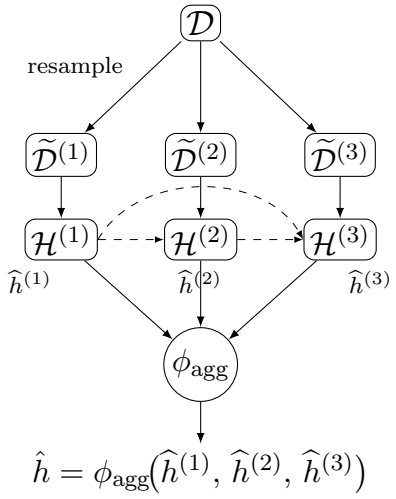


Fig. 19. A generic ensemble with three base learners, each using ERM to learn $\mathcal{H}^{(j)} \in \mathcal{H}^{(j)}$ based on the training set $\widetilde{\mathcal{D}}^{(j)}$. A base learner might also use the output of other base learners. The final hypothesis $\hat{h}$ is obtained by aggregating the hypotheses generated by the base learners.

Different ensemble methods use different constructions for the base learners. For example, bagging methods (such as a random forest) use

random sampling to construct slightly different training sets for each base learner. On the other hand, boosting methods run the base learners sequentially, i.e., each base learner tries to correct the ennuste 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 different models.

See also: bagging, boosting, stacking.

epoch An epoch represents one complete pass of the entire training set through some learning algoritmi. It refers to the point at which a model has processed every data point in the training set once. Training a model usually requires multiple epochs, since each iteration allows the model to refine the parameters and improve ennusteet. The number of epochs is something predefined by the user, and thus a hyperparameter, which plays a crucial role in determining how the model will generalize to unseen data. Too few epochs will result in alisovittaminen, while too many epochs can result in ylisovittaminen.

See also: training set, algoritmi, model, data point, parameter, ennuste, alisovittaminen, ylisovittaminen.

epävarmuus In the context of koneoppiminen, uncertainty refers to the presence of multiple plausible ?? or selityst based on available data. For example, the ennuste $\hat{h}(\mathbf{x})$ produced by a trained koneoppiminen model $\hat{h}$ 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. Todennäköisyys theory allows us to represent, quantify, and

reason about uncertainty in a mathematically rigorous manner.

See also: ??, riski, ??, ??.

estimation error Consider data points, each with piirrevektori $\mathbf{x}$ and nimiö y. In some applications, we can model the relation between the piirrevektori 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 piirrevektori. For a parametric hypothesis space, which consists of hypothesis ?? 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, piirrevektori, nimiö, hypothesis, koneoppiminen, ERM, hypothesis space, ??, model parameter.

Euclidean distance The term Euclidean distance is used as a synonym for the Euclidean norm.

See also: ??, Euclidean space.

Euclidean norm The Euclidean ?? $\|\mathbf{w}\|_2$ of a ??

$$\mathbf{w} = \big(w_1, \ldots, w_d\big) \in \mathbb{R}^d$$

is defined as

$$\|\mathbf{w}\|_2 := \sqrt{\sum_{j=1}^{d} w_j^2}.$$

The Euclidean ?? is distinct among all ?? on $\mathbb{R}^d$ in the sense that it is induced by the inner product $\mathbf{w}^T\mathbf{v}$ [**?**], [**?**], [**?**]. In other words,

$\|\mathbf{w}\|_2 = \sqrt{\mathbf{w}^T\mathbf{w}}$.

See also: ??, ??, Euclidean space.

Euclidean space The Euclidean space $\mathbb{R}^d$ of dimension $d \in \mathbb{N}$ consists of ?? $\mathbf{x} = \big(x_1, \ldots, x_d\big)$, 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 ?? $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ [?].

See also: ??.

expectation Consider a numeric piirrevektori $\mathbf{x} \in \mathbb{R}^d$ that we interpret as the realization of an ?? with ?? $p(\mathbf{x})$. 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 ?? is integrable [?], [?], [?]. Fig. ?? illustrates the expectation of a scalar ?? x that takes on values from a finite set only.

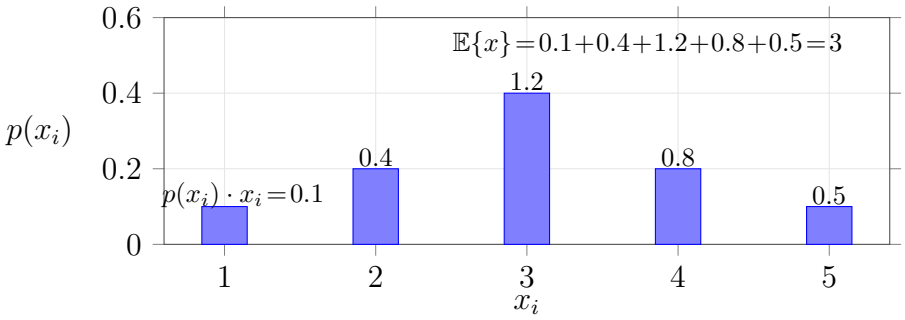


Fig. 20. The expectation of a ?? x is obtained by summing its possible values x_i, weighted by the corresponding todennäköisyys $p(x_i) = \mathbb{P}\left(x = x_i\right)$.

See also: piirrevektori, realization, ??, ??, todennäköisyys.

expert koneoppiminen aims to learn a hypothesis h that accurately predicts the nimiö of a data point based on its piirteet. We measure the ennuste error using some loss function. Ideally, we want to find a hypothesis that incurs minimal loss on any data point. We can make this informal goal precise via the independent and identically distributed assumption (i.i.d. assumption) and by using the Bayes risk as the vertailutaso for the (average) loss 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 [**?**]. ?? minimization methods learn a hypothesis that incurs a loss comparable to the best expert [**?**], [**?**].

See also: loss function, vertailutaso, ??.

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

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

feature map A piirre ?? refers to a ??

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

that transforms a piirrevektori $\mathbf{x} \in \mathcal{X}$ of a data point into a new piirrevektori $\mathbf{x}' \in \mathcal{X}'$, 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 lineaarinen malli to $\mathbf{x}'$. This idea is central to the design of kernel methodt [**?**]. Other benefits of using a piirre ?? include reducing ylisovittaminen and improving tulkittavuus [**?**]. A common use case is data visualization, where a piirre ?? with two output dimensions allows the representation of data points in a 2-D scatterplot. Some koneoppiminen methods employ trainable piirre ??, whose parameters are learned from data. An example is the use of hidden layers in a deep net, which act as successive piirre ?? [**?**]. A principled way to train a piirre ?? is through ERM, using a loss function that measures reconstruction quality, e.g., $L = \|\mathbf{x} - r(\mathbf{x}')\|^2$, where $r(\cdot)$ is a trainable ?? that attempts to reconstruct $\mathbf{x}$ from the transformed piirrevektori $\mathbf{x}'$.

See also: piirre, ??, kernel method, piirreoptimointi, PCA.

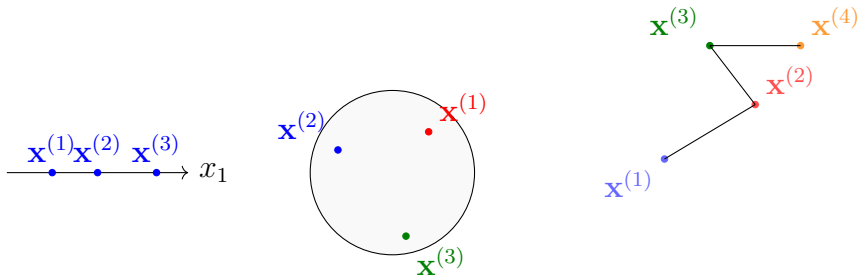
feature matrix Consider a tietoaineisto $\mathcal{D}$ with m data points with piirrevektorit $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. It is convenient to collect the individual piirrevektorit into the piirre ??

$$\mathbf{X} := \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^T = \begin{pmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(m)} & x_2^{(m)} & \cdots & x_d^{(m)} \end{pmatrix} \in \mathbb{R}^{m \times d}.$$

Note that the feature matrix is of size $m \times d$, i.e., it has m rows and d columns.

See also: tietoaineisto, data point, piirrevektori, piirre, ??.

feature space The piirre space of a given koneoppiminen application or method is constituted by all potential values that the piirrevektori of a data point can take on. For data points described by a fixed number d of numerical piirteet, a common choice for the piirre space is the Euclidean space $\mathbb{R}^d$. However, the mere presence of d numeric piirteet does not imply that $\mathbb{R}^d$ is the most appropriate representation of the piirre space. Indeed, the numerical piirteet 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. Piirreoptimointi methods try to learn a transformation of the original (potentially non-numeric) piirteet to ensure a more meaningful arrangement of data points in $\mathbb{R}^d$. Three examples of piirre spaces are shown in Fig. **??**.



$\mathcal{X}^{(1)}$ $\mathcal{X}^{(2)}$ $\mathcal{X}^{(3)}$

(a) (b) (c)

Fig. 21. Three different piirre spaces. (a) A linear space $\mathcal{X}^{(1)} = \mathbb{R}$. (b) A bounded ?? set $\mathcal{X}^{(2)} \subseteq \mathbb{R}^2$. (c) A discrete space $\mathcal{X}^{(3)}$ whose elements are nodes of an undirected ??.

See also: piirrevektori, 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 t starts from the current model parameters $\mathbf{w}^{(t)}$ 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. **??** illustrates the execution of a single iteration of FedAvg.

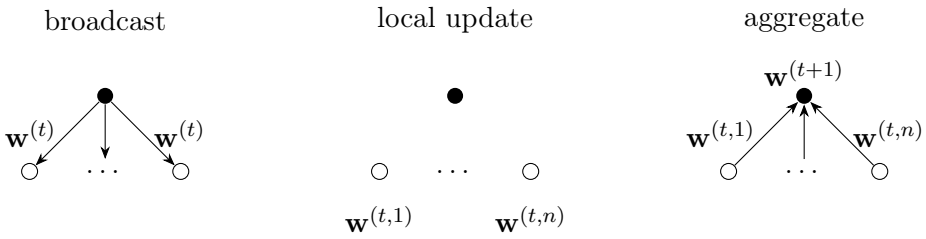


Fig. 22. Illustration of a single iteration of FedAvg, which consists of broadcasting model parameters by the server, performing local updates at clients, and aggregating the updates by the server.

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

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

See also: federoitu oppiminen, hajautettu algoritmi, FL network, ??, gradient-based method.

federated learning network (FL network) An federoitu oppiminen network consists of an undirected weighted ?? $\mathcal{G}$. 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 loss incurred on local datasets with their poikkeama across the edges of $\mathcal{G}$.

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

federated proximal (FedProx) FedProx refers to an iterative federoitu oppiminen algoritmi 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 ?? for the training [**?**].

See also: federoitu oppiminen, algoritmi, local model, model parameter, FedAvg, SGD, ??.

federated relaxed (FedRelax) An federoitu oppiminen hajautettu algoritmi.

See also: federoitu oppiminen, hajautettu algoritmi.

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

See also: federoitu oppiminen, hajautettu algoritmi, FL network, ??, gradient-based method, SGD.

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

See also: koneoppiminen, model, data.

flow-based clustering Flow-based clustering groups the nodes of an undirected ?? by applying k-means clustering to nodewise piirrevektorit. These piirrevektorit are built from network flows between carefully selected sources and destination nodes [**?**].

See also: clustering, ??, k-means, piirrevektori.

Gaussin sekoitemalli A GMM is a particular type of ?? for data points characterized by a numeric piirrevektori $\mathbf{x}$. Within a GMM, the piirrevektori $\mathbf{x}$ is drawn from a randomly selected ?? $p^{(c)} = \mathcal{N}\left(\boldsymbol{\mu}^{(c)}, \mathbf{C}^{(c)}\right)$ with $c = I$. The index $I \in \{1, \ldots, k\}$ is a ?? with todennäköisyydet $\mathbb{P}\left(I = c\right) = p_c$. A GMM is parameterized, for each $c = 1, \ldots, k$, by the todennäköisyys p_c, the ?? ?? $\boldsymbol{\mu}^{(c)}$, and the ?? $\mathbf{C}^{(c)}$.

See also: ??, ??, clustering.

generalization gap Yleistys gap is the difference between the performance of a hypothesis $h \in \mathcal{H}$ 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 ?? that allows us to compute the riski (or expected loss) $\bar{L}\big(\hat{h}\big)$ of a hypothesis h.

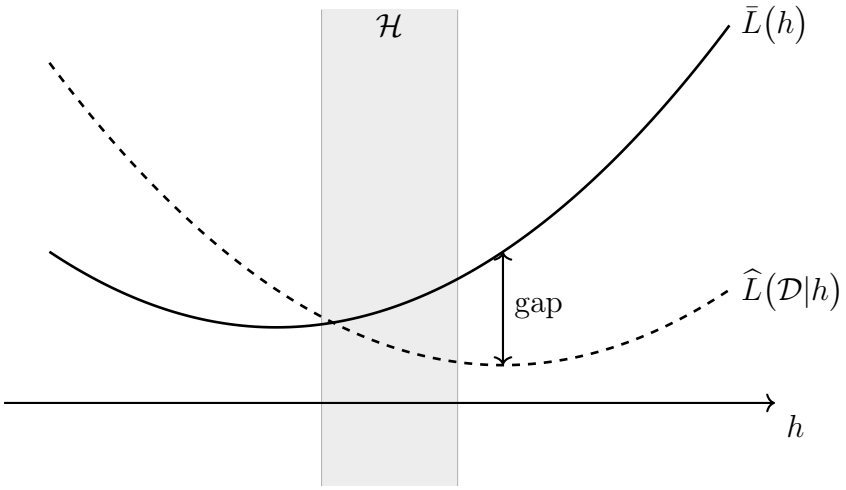


Fig. 23. The yleistys gap can be defined as the difference between the riski $\bar{L}(h)$ and the average loss (or empiirinen riski) $\widehat{L}(h|\mathcal{D}^{(\text{train})})$ computed on a training set.

In practice, the ?? underlying this expectation is unknown. Thus, we need to estimate the expectation based on observed data points. 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, loss function.

generalized total variation (GTV) GTV is a ?? 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 ?? $\mathcal{G}$ with edges $\mathcal{E}$. Given a ?? $d^{(h,h')}$ for the poikkeama between hypothesis ?? 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 parameter, ??, poikkeama, hypothesis, ??.

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

See also: RERM, GTV, regularisoija.

geometrinen mediaani 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 ?? [**?**] such that

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

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

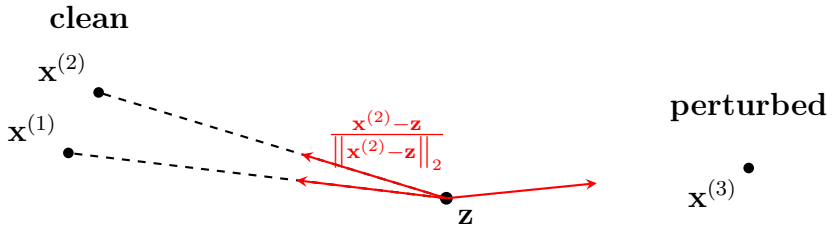


Fig. 24. Consider a solution $\mathbf{z}$ of (**??**) that does not coincide with any of the input vectors. The optimality condition for (**??**) requires that the unit ?? from $\mathbf{z}$ to the input vectors sum to zero.

See also: ??, ??.

gradient boosting ?? boosting is a boosting algoritmi that learns a hypothesis $\widetilde{h}$ by sequentially combining the hypotheses $\hat{h}^{(t)}$ [**?**, Algorithm 10.3], [**?**]. Similar to AdaBoost, ?? boosting uses a generalized ?? to combine the results of the base learners

$$\widetilde{h}^{(t)} = \widetilde{h}^{(t-1)} - \eta^{(t)} \hat{h}^{(t)}$$

where the generalized ?? $\hat{h}t$ is constructed from the tth base learner. The difference between AdaBoost and ?? boosting is in the construction of $\hat{h}t$. While AdaBoost uses weighted ERM for this construction, ?? boosting uses ERM on a modified training set. This modification is obtained by leaving the piirrevektorit untouched but replacing the nimiöt with the ?? of the loss function with respect to the ennusteet of the previous base learner.

See also: boosting, AdaBoost, GD.

gradient descent (GD) GD is an iterative method for finding the ?? of a ?? ?? $f : \mathbb{R}^d \to \mathbb{R}$. GD generates a sequence of estimates $\mathbf{w}^{(0)}$, $\mathbf{w}^{(1)}$, $\mathbf{w}^{(2)}$, ... that (ideally) converge to a ?? 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 ?? $\nabla f(\mathbf{w}^{(k)})$ of the ?? f at the current estimate $\mathbf{w}^{(k)}$. The resulting update rule is given by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \nabla f(\mathbf{w}^{(k)}) \tag{2}$$

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

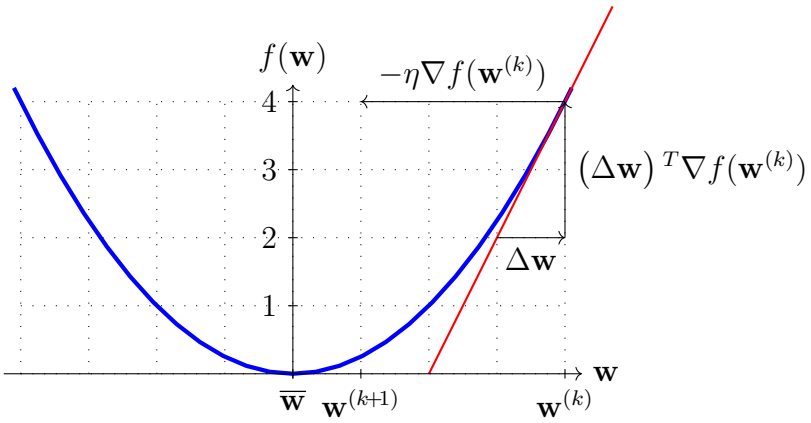


Fig. 25. A single ?? (**??**) toward the minimizer $\overline{\mathbf{w}}$ of $f(\mathbf{w})$.

See also: ??, ??, ??, step size, ??.

gradient-based method A ??-based method is an iterative technique for finding the ?? (or ??) of a ?? kohdefunktio $f(\mathbf{w})$ of the model parameters

w. Such a method constructs a sequence of approximations to an optimal choice for **w**. As the name indicates, a ??-based method uses the ?? of the kohdefunktio evaluated during previous iterations to construct new, (hopefully) improved model parameters. One important example of a ??-based method is GD.

See also: ??, ??, kohdefunktio, ??, GD.

graph clustering ?? clustering aims to cluster data points that are represented as the nodes of a ?? $\mathcal{G}$. 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: ??, clustering, data point, edge weight.

hajautettu algoritmi A distributed algoritmi is an algoritmi 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 algoritmi, which is implemented on a single device, a distributed algoritmi is executed concurrently on multiple devices with computational capabilities. Similar to a classical algoritmi, a distributed algoritmi can be modeled as a set of potential executions. However, each execution in the distributed setting involves both local computations

and message-passing ??. 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 $s_t^{(i)}$ at discrete-time instants $t = 1, \ldots, T_i$. 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 model for each device.

See also: algoritmi, device, ??, federoitu oppiminen, model.

hard clustering Hard clustering refers to the task of partitioning a given set of data points into (a few) non-overlapping clusters. This requirement allows to represent a cluster by a subset of data points, i.e., precisely those belonging to the cluster. In contrast to hard clustering, soft clustering methods allow for overlapping clusters and specify, for each data point, a numeric degree of belonging to each cluster. Hard clustering is an extreme case of soft clustering where the degrees of belonging take only two values, indicating either no belonging or full belonging. For data points characterized by numeric piirrevektorit $\mathbf{x} \in \mathbb{R}^d$, a widely used hard clustering method is k-means. Any hard clustering method for numeric piirrevektorit $\mathbf{x} \in \mathbb{R}^d$ can be adapted for non-numerical data using piirreoptimointi methods. One important example of this approach

is spectral clustering, where data points have a similarity structure in the form of an undirected ??. The nodes of this ?? represent data points while undirected (possibly weighted) edges represent similarities (and their extend) between data points. We can then use the entries of the ?? of the Laplacian matrix as numeric piirteet for each data point.

See also: clustering, data point, cluster, k-means.

harha Consider an ERM-based koneoppiminen method that learns a hypothesis $\hat{h} \in \mathcal{H}$ from a given training set. The analysis of the koneoppiminen method is often based on a ?? (such as the i.i.d. assumption) for the data generation. Here, data points and, in turn, the learned hypothesis $\hat{h}$ are viewed as (realizations of) ??. Any property $\theta\big(\hat{h}\big)$ of $\hat{h}$, such as specific model parameters in a parametric model or the ennuste error $y - \hat{h}(\mathbf{x})$ for a fixed data point, then also becomes an ??. The squared bias of a numeric property $\theta\big(\hat{h}\big) \in \mathbb{R}^r$ is [**?**], [**?**]

$$B^2 := \big\| \mathbb{E}\big\{\theta\big(\hat{h}\big)\big\} - \theta\big(\bar{h}\big) \big\|_2^2.$$

Here, $\bar{h}$ is a reference hypothesis, which could be defined by $\bar{h}(\mathbf{x}) = y$ for a fixed test data point with piirrevektori $\mathbf{x}$ and nimiö y.

See also: ??, ??, ??, 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 piirrevektori. 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)

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

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 piirteet d exceeds the number of data points m in the training set. For the training of a lineaarinen malli, adding $\alpha\|\mathbf{w}\|_2^2$ to the average neliövirhehäviö is equivalent to computing the average neliövirhehäviö on an augmented training set.

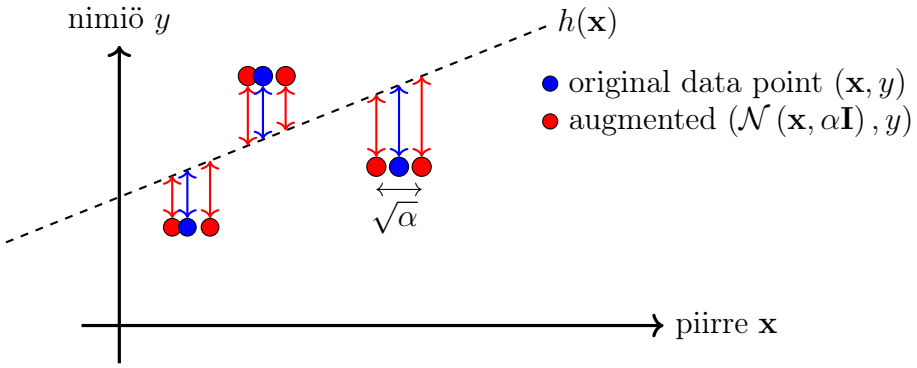


Fig. 26. For a lineaarinen malli, adding the penalty term $\alpha\|\mathbf{w}\|_2^2$ to the kohdefunktio in ERM is equivalent to ERM on an augmented tietoaineisto

This augmented training set is obtained by replacing each data point $\left(\mathbf{x}^{(r)}, y^{(r)}\right)$ in the original training set by the realization of infinitely many ?? ?? whose ?? is centered around $\left(\mathbf{x}^{(r)}, y^{(r)}\right)$.

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

high-dimensional regime The high-dimensional regime of ERM is characterized by the effective dimension of the model being larger than the sample size, i.e., the number of (labeled) data points in the trai-

ning set. For example, lineaarinen regressio methods operate in the high-dimensional regime whenever the number d of piirteet 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 ANNs, 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 todennäköisyys theory that studies the behavior of koneoppiminen methods in the high-dimensional regime [**?**], [**?**].

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

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

$$L\left((\mathbf{x}, y), h\right) := \max\{0, 1 - yh(\mathbf{x})\}. \tag{3}$$

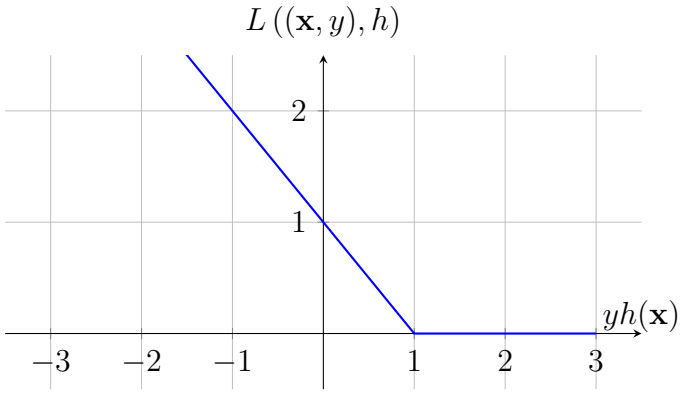


Fig. 27. The hinge loss incurred by the ennuste $h(\mathbf{x}) \in \mathbb{R}$ for a data point with nimiö $y \in \{-1, 1\}$. A regularized variant of the hinge loss is used by the support vector machine (SVM) [**?**].

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. **??**.

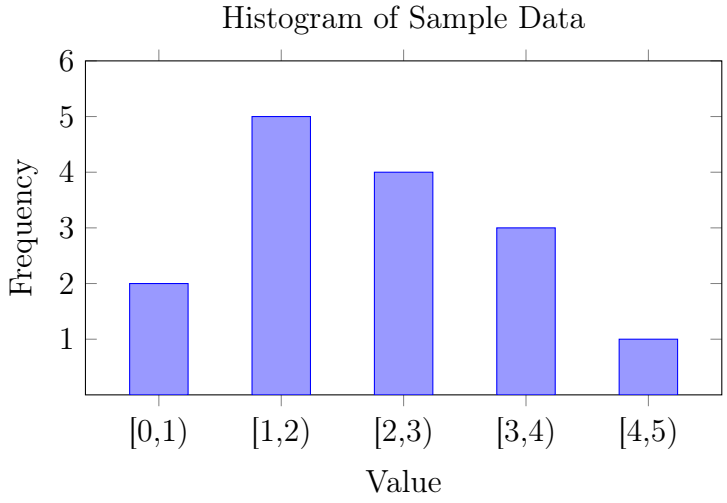


Fig. 28. A histogram consists of the fractions of data points that fall within different value ranges (i.e., bins). Each bar height shows the count of ?? in the corresponding interval.

See also: tietoaineisto, data point, ??.

horizontal federated learning (HFL) HFL uses local datasets constituted by different data points but uses the same piirteet 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 piirteet of different spatiotemporal regions. Each spatiotemporal region represents an individual data point, each characterized by the same piirteet (e.g., daily temperature or air pressure).

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

federated learning (VFL).

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

See also: loss, neliövirhehäviö, absolute error loss.

Huber regression Huber regressio [**?**] refers to ERM-based methods that use the Huber loss as a ?? 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 vakaus of the absolute error loss against the computational benefits of the ?? neliövirhehäviö.

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

hyperparameter A hyperparameter associated with a koneoppiminen method is a quantity that is used to select among a family of models. Typical examples include the oppimisnopeus used in a gradient-based method, the number of piirteet used in a lineaarinen malli, or the maximum depth of a decision tree. The usefulness of a specific choice of hyperparameter can be assessed via validointi. Similar to learning (or tuning) model parameters by ERM on a training set, we an learn (or tune) hyperparameters via minimizating the validointivirhe. Thus, in a sense hyperparameters are higher-level model parameters that are learned via a higher-level form of ERM: minimize the validointivirhe obtained for the trained model with a given hyperparameter value.

See also: model parameter, model, validointi.

hypothesis A hypothesis refers to a ?? (or ??) $h : \mathcal{X} \to \mathcal{Y}$ from the feature space $\mathcal{X}$ to the label space $\mathcal{Y}$. Given a data point with piirteet $\mathbf{x}$, we use a hypothesis ?? h to estimate (or approximate) the nimiö y using the ennuste $\hat{y} = h(\mathbf{x})$. Koneoppiminen is all about learning (or finding)

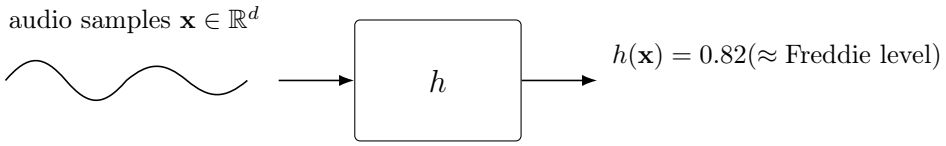


Fig. 29. A hypothesis $h : \mathcal{X} \to \mathcal{Y}$ maps the piirteet $\mathbf{x} \in \mathcal{X}$ of a data point to a ennuste $h(\mathbf{x}) \in \mathcal{Y}$ of the nimiö. For example, the koneoppiminen application `https://freddiemeter.withyoutube.com/` uses the samples of an audio recording as piirteet predict how closely a person's singing resembles that of Freddie Mercury.

a hypothesis ?? h such that $y \approx h(\mathbf{x})$ for any data point (with piirteet $\mathbf{x}$ 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 model underlying the method.

See also: ??, ??, ennuste, model.

hypothesis space A hypothesis space is a mathematical model that characterizes the learning capacity of an koneoppiminen method. The goal of such a method is to learn a hypothesis ?? that maps piirteet 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 ?? 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. **??**). 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 ?? the method can learn is also a valid hypothesis space.

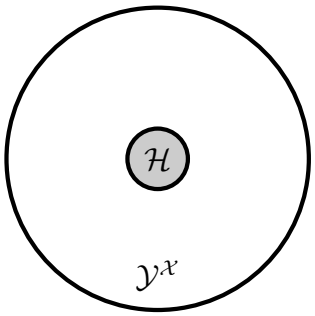


Fig. 30. The hypothesis space $\mathcal{H}$ of an koneoppiminen method is a (typically very small) subset of the (typically very large) set $\mathcal{Y}^{\mathcal{X}}$ of all possible ?? from the feature space $\mathcal{X}$ into the label space $\mathcal{Y}$.

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 laskennallinen ominaisuus. For instance, if efficient ?? operations are feasible and a roughly linear relation exists between piirteet and nimiöt, a lineaarinen malli can be a useful choice for $\mathcal{H}$.

See also: hypothesis, model, ??, lineaarinen malli.

hyökkäys 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 yksityisyyshyökkäykset, aim to infer sensitive attributes without modifying the system. Depending on their goal, we distinguish among denial-of-service attacks, backdoor attacks, and yksityisyyshyökkäykset.
See also: data poisoning, yksityisyyshyökkäys, sensitive attribute, denial-of-service attack, backdoor.

Ilmatieteen laitos The FMI is a government agency responsible for gathering and reporting weather data in Finland.
See also: data.

image segmentation Image segmentation refers to the task of clustering the pixels of an image into few segments.
See also: clustering.

independent and identically distributed assumption (i.i.d. assumption) The ?? assumption is a widely used ?? for the generation of data points. In particular, data points are represented as ?? ??.
See also: ??, ??, data point, ??.

input vector The term input ?? is often used as a synonym for the piirrevektori of a data point. In settings where data points arise from a dynamical system observed over time, piirteet are obtained from measuring input variables. These input variables are then used by koneoppiminen methods to predict the system's output (which is a nimiö in koneoppiminen

terminology).

See also: ??, piirrevektori, data point, output.

iteration The elementary computational step during the execution of an algoritmi is referred to as iteration [**?**], [**?**]. For example, the elementary computational step of gradient-based methods is a ??. More generally, the elementary computational step of a ?? is the evaluation of an underlying ?? $\mathcal{F}$, which might vary across iterations (see Fig. **??**). Many important koneoppiminen algoritmit, including Lloyd's algorithm and GD, are ??.

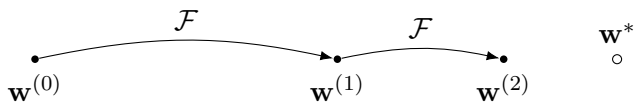


Fig. 31. A ?? consists of the repeated application of an ?? $\mathcal{F}$ with some fixed point $\mathbf{w}^*$, i.e., $\mathcal{F}\mathbf{w}^* = \mathbf{w}^*$.

See also: algoritmi, ??, GD.

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

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

Note that all entries $x_1^{(k)}, \ldots, x_d^{(k)}$ are updated simultaneously. The above iteration converges to a solution, i.e., $\lim_{t\to\infty} \mathbf{x}^{(t)} = \mathbf{x}$, under certain conditions on the ?? $\mathbf{A}$, e.g., being strictly diagonally dominant or symmetric positive definite [?], [?], [?]. Jacobi-type methods are appealing for large linear systems due to their parallelizable structure [?]. We can interpret the Jacobi method as a ??. 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 $\mathbf{x}^{(t+1)} = \mathbf{D}^{-1}(\mathbf{b} - \mathbf{R}\mathbf{x}^{(t)})$.
As an example, for the linear equation $\mathbf{Ax} = \mathbf{b}$, where

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

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: algoritmi, ??, ??, ??.

k-fold cross-validation (k-fold CV) A k-fold CV is a method for evaluating the generalization gap of an ERM-based koneoppiminen method.

The idea is to divide a tietoaineisto $\mathcal{D}$ evenly into k subsets (or folds) $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)}$.

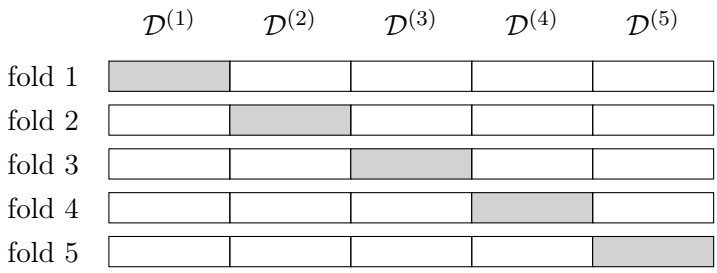


Fig. 32. In k-fold CV, the available tietoaineisto $\mathcal{D}$ is evenly divided into k folds $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)}$. Each fold is used once as a validation set, while the remaining $k-1$ folds form the training set.

For each fold $b = 1, \ldots, k$, we train the model on the union of all folds except $\mathcal{D}^{(b)}$ and validate it on $\mathcal{D}^{(b)}$. The overall performance is obtained by averaging the validointi results across all k folds.
See also: validointi, validointivirhe.

k-means The k-?? principle is an optimization-based approach to the clustering of data points with a numeric piirrevektori [**?**, Ch. 8]. As a hard clustering approach, k-?? partitions a tietoaineisto into k disjoint subsets (or clusters), which are indexed by $c = 1, \ldots, k$. Each cluster $\mathcal{C}$ is characterized by the average piirrevektori of data points that belong to it. This average (or ??) piirrevektori is referred to as the cluster centroid $\mathbf{w}^{(c)}$. A visual illustration is provided in Fig. **??**.

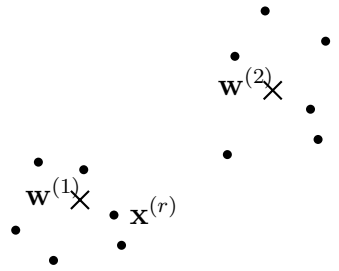


Fig. 33. A scatterplot of data points, indexed by $r = 1, \ldots, m$ and characterized by piirrevektorit $\mathbf{x}^{(r)} \in \mathbb{R}^2$. The scatterplot also includes two cluster centroids $\mathbf{w}^{(1)}, \mathbf{w}^{(2)} \in \mathbb{R}^2$.

In general, solving the k-?? ?? exactly is challenging (or NP-hard) [**?**]. However, there are simple iterative methods for finding approximately optimal cluster centroids. One such method is referred to as Lloyd's algorithm.

See also: hard clustering, cluster, Lloyd's algorithm.

k-means++ TBC.

See also: k-means.

kernel method A ydinfunktio method is an koneoppiminen method that uses a ydinfunktio K to map the original (i.e., raw) piirrevektori $\mathbf{x}$ of a data point to a new (transformed) piirrevektori $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$ [**?**], [**?**]. The motivation for transforming the piirrevektorit 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 piirrevektorit $\mathbf{z}$ might allow us to use lineaariset mallit, even if the data points are not linearly separable in the original

feature space (see Fig. **??**).

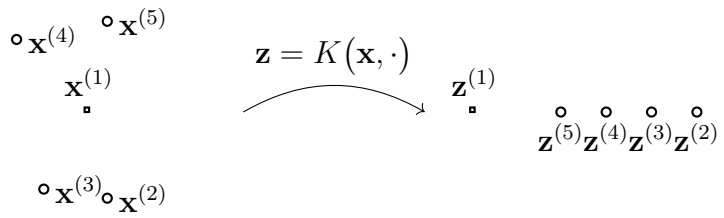


Fig. 34. Five data points characterized by piirrevektorit $\mathbf{x}^{(r)}$ and nimiöt $y^{(r)} \in \{\circ, \square\}$, for $r = 1, \ldots, 5$. With these piirrevektorit, there is no way to separate the two classes by a straight line (representing the päätöspinta of a lineaarinen luokitin). In contrast, the transformed piirrevektorit $\mathbf{z}^{(r)} = K\big(\mathbf{x}^{(r)}, \cdot\big)$ allow us to separate the data points using a lineaarinen luokitin.

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

kohdefunktio An objective ?? is a ?? that assigns a numeric objective value $f(\mathbf{w})$ to each choice $\mathbf{w}$ of some variable that we want to optimize (see Fig. **??**). In the context of koneoppiminen, the optimization variable could be the model parameters of a hypothesis $h^{(\mathbf{w})}$. Common objective ?? include the riski (i.e., expected loss) or the empiirinen riski (i.e., average loss 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 ?? or the ??) of the objective ??.

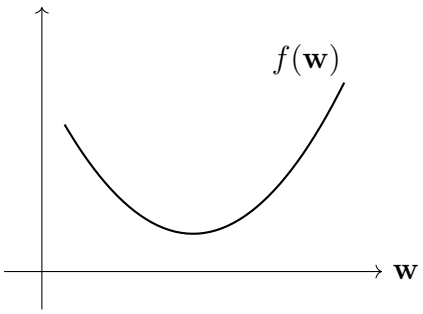


Fig. 35. An objective ?? maps each possible value $\mathbf{w}$ of an optimization variable, such as the model parameters of an koneoppiminen model, to a value $f(\mathbf{w})$ that measures the usefulness of $\mathbf{w}$.

See also: loss, empiirinen riski, ERM, ??.

koneoppiminen ML methods aim to learn (or find) a useful hypothesis ?? $\hat{h} \in \mathcal{H}$ out of a model $\mathcal{H}$. The learned $\hat{h}$ is used to compute a ennuste $\hat{y} = \hat{h}(\mathbf{x})$ for the nimiö y of a data point. The learning process is guided by a quantitative ?? of the loss incurred when the ennusteet obtained from the learned hypothesis differ from the actual nimiö y. Different ML methods use different design choices for this quantitative ?? (or loss function) as well as different choices for the model and the data points (i.e., their piirteet and nimiöt) [**?**, Ch. 3].

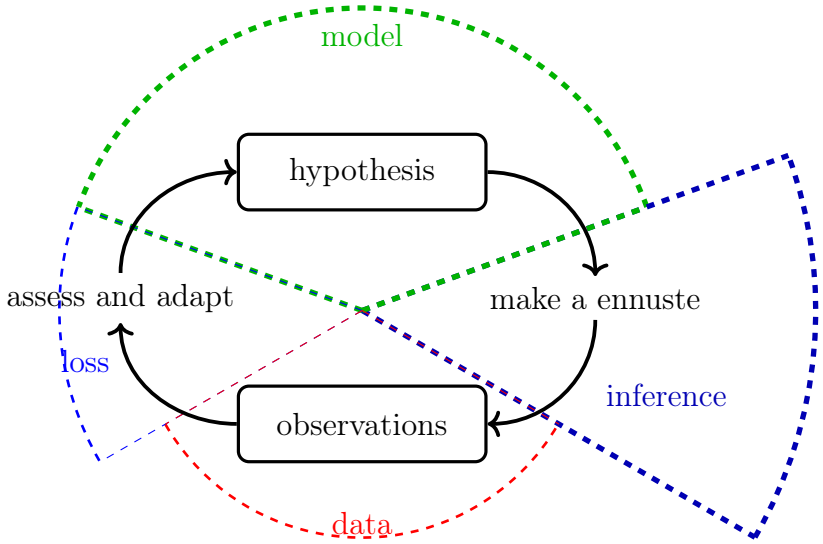


Fig. 36. ML learns a hypothesis out of a model (or hypothesis space) by trying to minimize the loss incurred by the ennusteet for the nimiöt of data points. The ennusteet are computed solely from the piirteet of the data points.

Another distinction between ML methods is how they access data points during learning. For example, some methods have access to a complete tietoaineisto during training, which allows them to use ERM [?], [?]. In contrast, online learning methods access data sequentially and, in turn, update the learned hypothesis whenever a new data point arrives [?], [?], [?].

See also: model, data, loss, ERM, online learning.

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

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

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

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

See also: ??, koneoppiminen, model, ??.

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

See also: ??.

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

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

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

See also: ??, ??, ??.

label space In an koneoppiminen application, each data point is described by a set of piirteet 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. **??** shows some

examples of nimiö spaces that are commonly used in koneoppiminen applications.

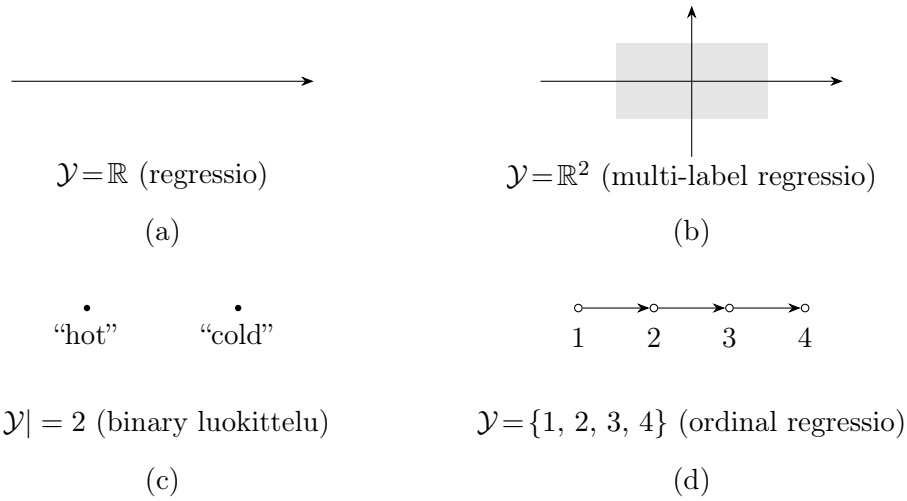


Fig. 37. Examples of nimiö spaces and the corresponding types of koneoppiminen. (a) Regressio. (b) Multi-label regressio. (c) Binary luokittelu. (d) Ordinal regressio.

The choice of the nimiö space $\mathcal{Y}$ determines the type 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.

label vector Given a tietoaineisto of m labeled data points

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

it is convenient to collect the corresponding nimiöt into a single nimiö ?? $\mathbf{y} := \big(y_1, \ldots, y_m\big)^T$ [?], [?].

See also: tietoaineisto, labeled data point, nimiö, data point.

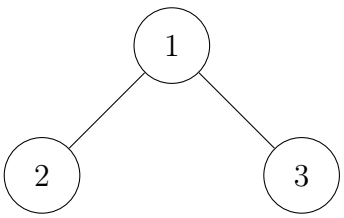
labeled data point A data point whose nimiö is known or has been determined by some means that might require human labor.

See also: data point, nimiö.

Laplacian matrix The structure of a ?? $\mathcal{G}$, with nodes $i = 1, \ldots, n$, can be analyzed using the properties of special ?? that are associated with $\mathcal{G}$. One such ?? is the ?? Laplacian ?? $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{n \times n}$, which is defined for an undirected and weighted ?? [?], [?]. It is defined elementwise as (see Fig. **??**)

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

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

See also: ??, ??, edge weight.

large language model (LLM) An LLM is an umbrella term for koneoppiminen methods that use high-dimensional koneoppiminen models (with billions of model parameters) trained on large collections of text data. LLMs are used to analyze or generate ?? of tokens that constitute text data. Many current LLMs use some variant of a transformer that is trained via self-supervised learning, i.e., the training is 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 piirteet of data points. This construction requires very little human supervision and allows for generating sufficiently large training sets for LLMs.

See also: token, transformer, NLP.

laskennalliset ominaisuudet 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 ??); 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, ??, model parameter, GD.

layer A deep net is an ANN that consists of consecutive layers, indexed by $\ell =$

1, 2, ..., 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 aktivointifunktio 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 piirteet 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.

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.

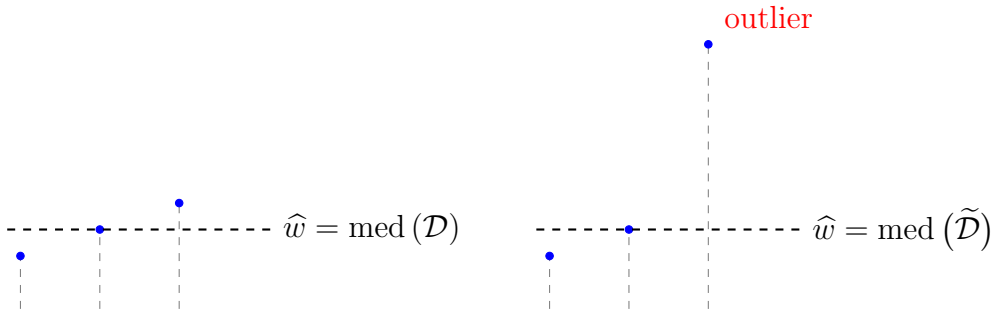


Fig. 39. For the simple parametric model $h^{(w)}(\mathbf{x}) = w$, ERM with absolute error loss amounts to computing the ??. (a) Original tietoaineisto $\mathcal{D}$. (b) Noisy tietoaineisto $\widetilde{\mathcal{D}}$ including an outlier.

For the parametric model $h^{(w)}(\mathbf{x}) = w$, ERM with absolute error loss is solved by the ??. Using neliövirhehäviö instead for the same parametric model, makes ERM computing the ??.

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 ?? $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ from a training set. Lasso is obtained from lineaarinen regressio by adding the scaled ℓ_1-?? $\alpha\|\mathbf{w}\|_1$ to the average neliövirhehäviö incurred on the training set.

See also: SRM, weights, ??, training set, lineaarinen regressio, ??, neliövirhehäviö.

least squares Least squares refers to ERM-based methods that use the average neliövirhehäviö

$$\frac{1}{m}\sum_{r=1}^{m}\left(y^{(r)} - h(\mathbf{x}^{(r)})\right)^2$$

on a training set $\mathcal{D} = \left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right) \right\}$ to measure the quality of a hypothesis ?? $h \in \mathcal{H}$. We obtain different least squares methods by using different models in ERM. For example, the least squares variant of a lineaarinen malli is a least squares method that uses a lineaarinen malli.

See also: ERM, neliövirhehäviö, lineaarinen malli, lineaarinen regressio.

leave-one-out (LOO) cross-validation A special case of the k-fold CV where the validation set is size one, i.e. the validation set is only a single

data point.

See also: k-fold CV, validointi, validointivirhe.

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

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

lineaarinen malli Consider an koneoppiminen application involving data points, each represented by a numeric piirrevektori $\mathbf{x} \in \mathbb{R}^d$. A linear model defines a hypothesis space consisting of all real-valued ?? 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 piirteet used to compute the ennuste $h(\mathbf{x})$. The choice of d is often guided not only by laskennallinen ominaisuus (e.g., fewer features reduce computation) and laskennallinen ominaisuus (e.g., more features typically reduce harha and riski), but also by tulkittavuus. A linear model using a small number of well-chosen piirteet is generally considered more interpretable [**?**], [**?**]. The linear model is attractive because it can typically be trained using scalable ?? ?? [**?**], [**?**]. Moreover, linear models often permit rigorous statistical analysis, including fundamental limits on the ?? achievable riski [**?**]. They are also useful for analyzing more complex nonlinear models such as ANNs. 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 model in the output layer. Similarly, a decision tree can be interpreted as applying a one-hot-encoded feature map based on päätösalueet, followed by a linear model that assigns a ennuste to each region. More generally, any trained model $\hat{h} \in \mathcal{H}$ that is ?? at some $\mathbf{x}'$ can be locally approximated by a ?? $g(\mathbf{x})$. Fig. **??** illustrates such a local linear approximation, defined by the ?? $\nabla \hat{h}(\mathbf{x}')$. Note that the ?? is only defined where $\hat{h}$ is ??. To ensure vakaus in the context of ??, one may prefer models whose associated ?? $\hat{h}$ is Lipschitz ??. A classic result in mathematical analysis—Rademacher's Theorem—states that if $\hat{h}$ is Lipschitz ?? with some constant L over an open set $\Omega \subseteq \mathbb{R}^d$, then $\hat{h}$ is ?? almost everywhere in Ω [**?**, Th. 3.1].

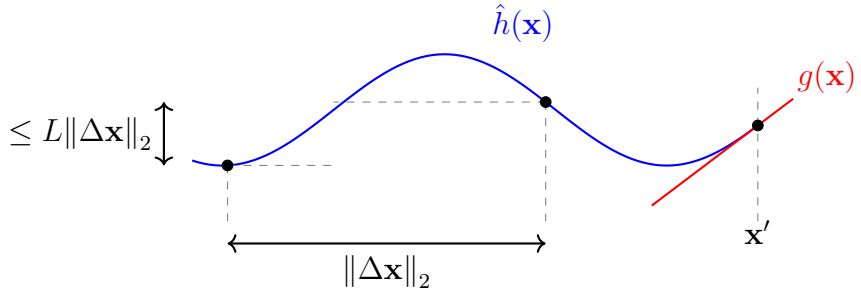


Fig. 40. A trained model $\hat{h}(\mathbf{x})$ that is ?? at a point $\mathbf{x}'$ can be locally approximated by a ?? $g \in \mathcal{H}^{(d)}$. This local approximation is determined by the ?? $\nabla \hat{h}(\mathbf{x}')$.

See also: model, hypothesis space, ??, tulkittavuus, LIME.

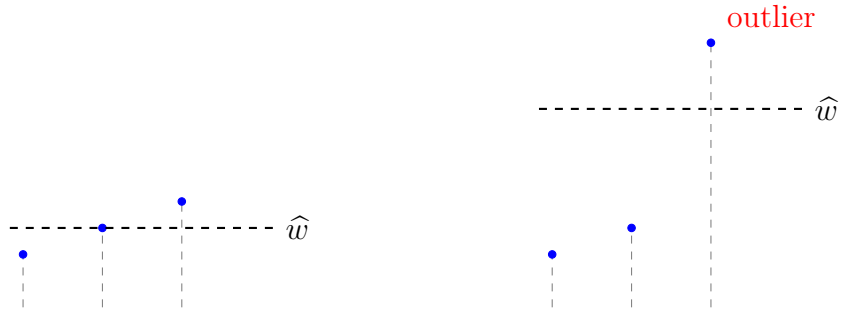
lineaarinen regressio Linear regressio methods learn a linear hypothesis ?? $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ which is used to predict the numeric nimiö $y \in \mathbb{R}$ of a data point based on its numeric piirrevektorit $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T \in \mathbb{R}^d$.

The least-squares variant of linear regressio measures the quality of a linear hypothesis ?? via the average neliövirhehäviö incurred on a training set

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

As an instance of ERM, linear (least-squares) regressio learns the model parameters $\mathbf{w}$ by solving the ??

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{m} \sum_{r=1}^{m} \left(y^{(r)} - \mathbf{w}^T \mathbf{x}^{(r)}\right)^2.$$



(a) (b)

Fig. 41. For a lineaarinen malli with $d = 1$ and using the trivial piirre $x = 1$ for any data point, linear regressio reduces to computing the average $\widehat{w} = (1/m) \sum_{r=1}^{m} y^{(r)}$. (a) A clean training set and resulting parameter (given by the average). (b) A perturbed tietoaineisto (including an outlier) and the resulting parameter.

We can rewrite the above ?? more compactly using the feature matrix $\mathbf{X} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^T \in \mathbb{R}^{m \times d}$ and the nimiö ?? $\mathbf{y} = \left(y^{(1)}, \ldots, y^{(m)}\right)^T \in$

$\mathbb{R}^m$. This allows to rewrite the above ?? as

$$\min_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{m}\|\mathbf{y}-\mathbf{X}\mathbf{w}\|_2^2.$$

By the zero-gradient condition, a necessary and sufficient condition for a vector $\widehat{\mathbf{w}}$ to be a solution to the above ?? is the linear system of equations [**?**]

$$\mathbf{X}^T\mathbf{X}\widehat{\mathbf{w}} = \mathbf{X}^T\mathbf{y}. \tag{4}$$

Instead of solving (**??**) directly (via computing the ?? or ??), many koneoppiminen mehtods use variants of GD to construct a sequence $\mathbf{w}^{(0)}, \mathbf{w}^{(1)}, \ldots,$ of increasingly accurate approximations of a solution $\widehat{\mathbf{w}}$ to (**??**). These gradient-based methods can be interpreted as a ?? for the following re-formulation of (**??**),

$$\big(\mathbf{I}-\mathbf{A}\mathbf{X}^T\mathbf{X}\big)\widehat{\mathbf{w}} + \mathbf{X}^T\mathbf{y} = \widehat{\mathbf{w}}, \text{ with some invertible ?? } \mathbf{A}.$$

This equation is solved by a ?? $\widehat{\mathbf{w}}$ if and only if this ?? also solves (**??**). The optimality condition (**??**) is also useful for the study of the stability of linear regressio. Ideally, we would like the solutions of (**??**) to be insensitive to small perturbations of the training set. We can capture these perturbations via a perturbed feature matrix $\widetilde{\mathbf{X}} = \mathbf{X} + \Delta\mathbf{X}$ and perturbed nimiö ?? $\widetilde{\mathbf{y}} = \mathbf{y} + \Delta\mathbf{y}$. Here, $\Delta\mathbf{X}$ and $\Delta\mathbf{y}$ represent small perturbations to the piirrevektorit and nimiöt of the data points in the original training set. ?? perturbation theory allows to evaluate how much the solutions of the perturbed linear regressio problem [**?**, Sec. 2.6]

$$\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}}\widetilde{\mathbf{w}} = \widetilde{\mathbf{X}}^T\widetilde{\mathbf{y}}$$

deviate from the solutions $\widetilde{\mathbf{w}}$ of the original linear regressio problem.
See also: regressio, lineaarinen malli, ERM.

linear discriminant analysis (LDA) LDA is a classical piirreoptimointi method [**?**], [**?**]. In the context of binary luokittelu problems, LDA seeks a linear feature map $\phi^{(\mathbf{w})} : \mathbb{R}^d \to \mathbb{R} : \mathbf{x} \mapsto \mathbf{w}^T\mathbf{x}$ such that the new piirre $\phi^{(\mathbf{w})}(\mathbf{x})$ optimally allows us to predict the nimiö of a data point.
See also: piirreoptimointi, ulottuvuuksien vähentäminen, self-supervised learning.

linear least squares Linear least squares refers to the variant of lineaarinen regressio that uses the neliövirhehäviö to measure the quality of a linear hypothesis ??. Conversely, it can also be viewed as the variant of least squares that restricts the hypothesis to a lineaarinen malli. In particular, linear least squares learns the parameters $\mathbf{w}$ of a linear hypothesis $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ by solving

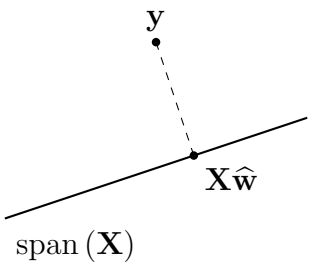
$$\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2. \tag{5}$$

Here, the feature matrix is $\mathbf{X} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)$ and the label vector is $\mathbf{y} = \left(y^{(1)}, \ldots, y^{(m)}\right)^T$. Both are constructed from the training set

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

The ?? in (**??**) admits a clear geometric interpretation, i.e., we seek the ?? $\mathbf{X}\widehat{\mathbf{w}}$ in the ?? of $\mathbf{X}$ that is closest to the label vector $\mathbf{y}$ (see Fig. **??**) [**?**, Ch. 8]. A necessary and sufficient condition for $\widehat{\mathbf{w}}$ to minimize (**??**) is the normal equations

$$\mathbf{X}^T\mathbf{X}\widehat{\mathbf{w}} = \mathbf{X}^T\mathbf{y}.$$



(a)

normal equations

$$\mathbf{X}^T\mathbf{X}\,\widehat{\mathbf{w}} \;=\; \mathbf{X}^T\mathbf{y}$$

(b)

Fig. 42. Linear least squares has both geometric and algebraic interpretations. (a) Geometrically, it finds the orthogonal ?? of the label vector $\mathbf{y}$ onto the ?? of the feature matrix $\mathbf{X}$ [?, Ch. 8]. (b) Algebraically, it solves a linear system known as normal equations.

See also: least squares, lineaarinen regressio, neliövirhehäviö, lineaarinen malli, ERM.

Lloyd's algorithm Lloyd's algoritmi [?] is an iterative ?? for finding cluster centroids that are approximately optimal for the k-means kohdefunktio. Lloyd's algoritmi alternates between: 1) updating the cluster assignment of each data point based on the nearest current cluster centroid; and 2) recalculating the cluster centroids given the updated cluster assignments [?].

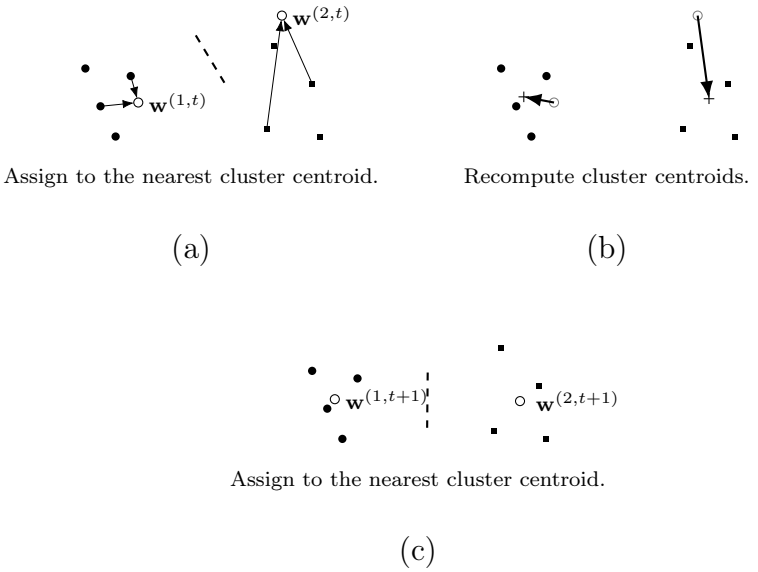


Fig. 43. Lloyd's algoritmi alternates between (a) and (c) assigning data points to the nearest cluster centroid and, in turn, (b) recomputing the cluster centroids based on the new cluster assignments.

See also: cluster centroid, k-means, clustering.

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 piirteet 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 ?? or communication infrastructure and are encoded in the edges of an FL network.

See also: tietoaineisto, data point, piirre, nimiö, koneoppiminen, ??, FL

network.

local interpretable model-agnostic explanations (LIME) Consider a trained model (or learned hypothesis) $\widehat{h} \in \mathcal{H}$, which maps the piirrevektori 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 piirrevektori $\mathbf{x}^{(0)}$ [**?**]. The selitys is given in the form of a local approximation $g \in \mathcal{H}'$ of $\widehat{h}$ (see Fig. **??**). 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 piirrevektorit centered around $\mathbf{x}^{(0)}$ and the (pseudo-)nimiö $\widehat{h}(\mathbf{x})$. Note that we can use a different model $\mathcal{H}'$ for the approximation from the original model $\mathcal{H}$. For example, we can use a decision tree to locally approximate a deep net. Another widely used choice for $\mathcal{H}'$ is the lineaarinen malli.

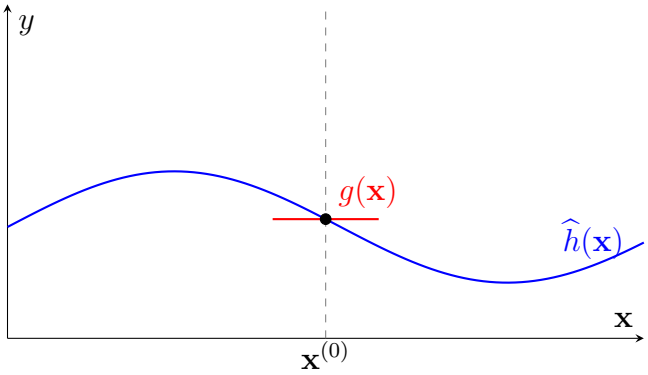


Fig. 44. To explain a trained model $\widehat{h} \in \mathcal{H}$, around a given piirrevektori $\mathbf{x}^{(0)}$, we can use a local approximation $g \in \mathcal{H}'$.

See also: model, selitys, ERM, training set, nimiö, decision tree, deep

net, lineaarinen malli.

local model Consider a collection of devices that are represented as nodes $\mathcal{V}$ of an FL network. A local model $\mathcal{H}^{(i)}$ 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, model, hypothesis space.

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

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

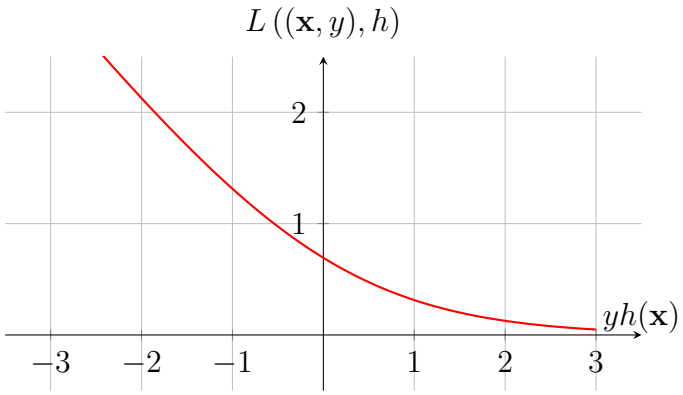


Fig. 45. The logistic loss incurred by the ennuste $h(\mathbf{x}) \in \mathbb{R}$ for a data point with nimiö $y \in \{-1, 1\}$.

Note that the expression (**??**) for the logistic loss applies only for the

label space $\mathcal{Y} = \{-1, 1\}$ and when using the thresholding rule (**??**).

See also: loss, luokittelu, luokitin, lineaarinen malli.

logistic regression Logistic regressio learns a linear hypothesis ?? (or luokitin) $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ to predict a binary nimiö y based on the numeric piirrevektori $\mathbf{x}$ of a data point [**?**], [**?**]. The quality of a linear hypothesis ?? is measured by the average logistic loss on some labeled data points (i.e., the training set).

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

loss koneoppiminen methods use a loss function $L\left(\mathbf{z}, h\right)$ 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 loss function L itself and the specific value $L\left(\mathbf{z}, h\right)$, for a data point $\mathbf{z}$ and hypothesis h.

See also: loss function, empiirinen riski.

loss function A loss ?? is a ??

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

It assigns a nonnegative real number (i.e., the loss) $L\left((\mathbf{x}, y), h\right)$ to a pair that consists of a data point, with piirteet $\mathbf{x}$ 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. **??** depicts a loss ?? for a given data point, with piirteet $\mathbf{x}$ and nimiö y, as a ?? of the hypothesis $h \in \mathcal{H}$.

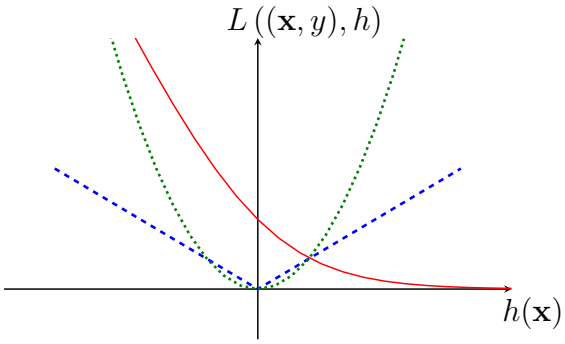


Fig. 46. Some loss ?? $L\left((\mathbf{x}, y), h\right)$ for a fixed data point, with piirrevektori $\mathbf{x}$ and nimiö y, and a varying hypothesis h. koneoppiminen methods try to find (or learn) a hypothesis that incurs minimal loss.

See also: loss, nimiö, piirrevektori, ERM.

luokitin A classifier is a hypothesis (i.e., a ??) $h(\mathbf{x})$ used to predict a nimiö taking on values from a finite label space. We might use the ?? value $h(\mathbf{x})$ itself as a ennuste $\hat{y}$ for the nimiö. However, it is customary to use a ?? $h(\cdot)$ 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 ?? $h(\mathbf{x}) \in \mathbb{R}$ as a classifier. A ennuste $\hat{y}$ can then be obtained via thresholding,

$$\hat{y} = 1 \text{ for } h(\mathbf{x}) \geq 0 \text{ and } \hat{y} = -1 \text{ otherwise.} \tag{7}$$

We can characterize a classifier by its päätösalueet $\mathcal{R}_a$, 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 piirteet $\mathbf{x}$. 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.

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

machine unlearning Consider an koneoppiminen method that learns a hypothesis $\hat{h}$ via ERM on a training set $\mathcal{D}$. The learned hypothesis can reveal information about $\mathcal{D}$, which is exploited by yksityisyyshyökkäykset such as model inversion. Machine unlearning refers to techniques that modify $\hat{h}$, so that it is harder to infer properties of individual data points in $\mathcal{D}$ [**?**]. Machine unlearning helps to meet legal requirements for yksityisyyden suoja in AI systems [**?**].
See also: model inversion, yksityisyyden suoja, ??.

mallin valinta In koneoppiminen, model selection refers to the process of choosing between different candidate models. In its most basic form, model selection amounts to: 1) training each candidate model; 2) computing the validointivirhe for each trained model; and 3) choosing the

model with the smallest validointivirhe [**?**, Ch. 6].

See also: koneoppiminen, model, training, validointivirhe.

mean absolute error (MAE) The MAE of a hypothesis is the average absolute error loss computed over a given tietoaineisto. In theoretical analyses, MAE also denotes the expected absolute error loss, i.e., the corresponding riski.

See also: absolute error loss, riski.

mean squared error (MSE) The MSE of a hypothesis is the average neliövirhehäviö computed over a given tietoaineisto. In theoretical analyses, MSE also denotes the expected neliövirhehäviö, i.e., the corresponding riski.

See also: neliövirhehäviö, riski.

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

See also: ??, estimation error, ??, neliövirhehäviö.

membership inference attack Consider an koneoppiminen method that learns a hypothesis via ERM on a training set. Membership inference hyökkäys is a form of yksityisyyshyökkäys where an adversary tries to determine whether a particular data point was part of the training set. The

attacker typically queries $\hat{h}$ with candidate piirrevektorit $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(B)}$, and infers the membership status of a given data point based on the ennusteet $\hat{h}\big(\mathbf{x}^{(1)}\big), \ldots, \hat{h}\big(\mathbf{x}^{(B)}\big)$ [**?**].

See also: hyökkäys, yksityisyyshyökkäys.

metric A metric is a quantitative ?? used to compare objects. In mathematics, a metric measures the distance between two points in a space 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 [**?**]. In the context of koneoppiminen, the term metric refers to a quantitative ?? of how well a model performs (somewhat similar to a loss function). Examples include tarkkuus, precision, and the average binääritappio on a test set [**?**], [**?**]. The term loss function is typically used in the context of model training, while the term metric is used in the context of model validointi.

See also: loss, validointi, tarkkuus.

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

mathematical model for koneoppiminen is a hypothesis space. A hypothesis space consists of hypothesis ?? that are used by an koneoppiminen method to predict nimiöt from the piirteet of data points. Another important type of mathematical model is a ??, which consists of ?? 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 ?? in Fig. **??**.

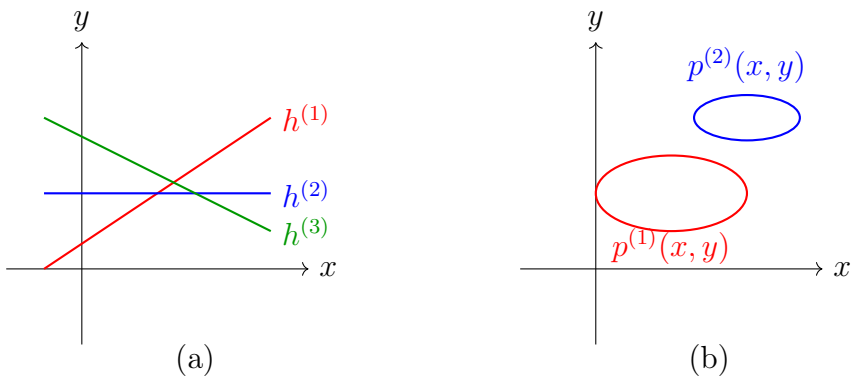


Fig. 47. Two types of mathematical models used in koneoppiminen. (a) A hypothesis space consisting of three ??. (b) A ?? consisting of ?? over the plane spanned by the piirre and nimiö values of a data point.

See also: hypothesis space, ??, ??.

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

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

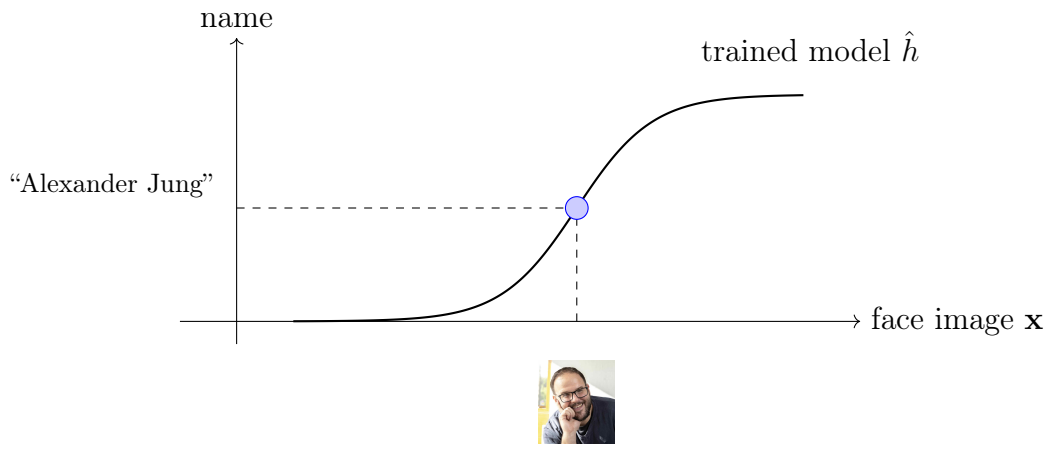


Fig. 48. Model inversion techniques implemented in the context of facial image classification.

See also: model, yksityisyyshyökkäys, koneoppiminen, sensitive attribute, data point, ennuste, luokittelu, ??, ??, yksityisyyden suoja.

model parallelism Model parallelism refers to a particular class of distributed ?? used to train an koneoppiminen model. Here, different devices store and process disjoint subsets of the model parameters. This approach contrasts with data parallelism, where each device maintains a full replica of the model parameters while processing disjoint subsets of the global tietoaineisto. Model parallelism allows us to train an koneoppiminen model whose parameters cannot fit into the memory of a

single device. One key application of model parallelism is the training of extremely large ANNs, such as transformer models with billions of model parameters.

See also: VFL.

model parameter The elements of a parametric model are specified by quantities that are referred to as model parameters. In the context of koneoppiminen, a parametric model consists of hypothesis maps that are specified by a list of model parameters $w_1, w_2, \ldots, w_d$. It is often convenient to stack these model parameters into a ?? $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T \in \mathbb{R}^d$.

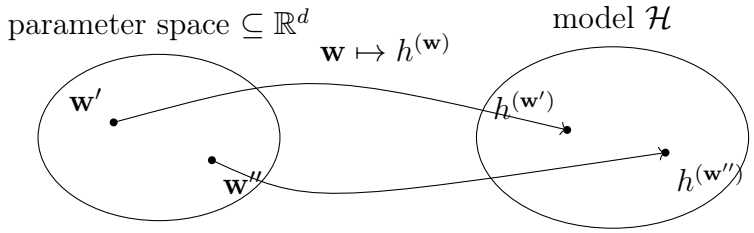


Fig. 49. The model parameters $\mathbf{w}$ select a well-defined hypothesis $h^{(\mathbf{w})}$ out of the model $\mathcal{H}$.

We can think of model parameters as an identifier for a hypothesis ??, similar to how a social security number identifies a person.

See also: model, parameter, hypothesis, ??.

monitehtäväoppiminen Multitask learning aims to leverage relations between different oppimistehtävät. Consider two oppimistehtävät 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: oppimistehtävä, tietoaineisto, deep net, weights, layer.

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.

mutual information (MI) The MI $I(\mathbf{x}; y)$ between two ?? $\mathbf{x}$, y defined on the same ?? 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 ?? 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: ??, ??, ennuste, hypothesis, ERM, koneoppiminen.

naapuri A neighbor of a node $i \in \mathcal{V}$ within an ?? is a node $i' \in \mathcal{V} \setminus \{i\}$ that is ?? via an edge to node i.

See also: ??, ??.

naapurusto Consider some ?? $\mathcal{X}$ 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 ??, which is a special case of a ??, the neighborhood of a node $i \in \mathcal{V}$ is the set of its naapurit.

See also: metric, naapuri.

natural language processing (NLP) NLP studies koneoppiminen methods for the the analysis and generation of human language. Typical NLP tasks include text luokittelu, machine translation, sentiment analysis, and question answering. Modern NLP systems represent language as ?? of tokens and train models that capture contextual dependencies, such as attention-based methods.

See also: token, attention.

neliövirhehäviö The squared error loss measures the ennuste error of a hypothesis h when predicting a numeric nimiö $y \in \mathbb{R}$ from the piirteet $\mathbf{x}$ 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: loss, ennuste, hypothesis, nimiö, piirre, data point.

nested cross-validation Nested cross-validation is a method of extending the k-fold CV from training and validation sets to also cover the test set. Instead of simply choosing a single test set from the data, two loops are run: the outer and the inner loop. The outer loop uses k-fold CV to

separate a test set from the data and the inner loop again uses k-fold CV to separate the rest of the data into a training and validation sets. Doing this decreases the ?? and harha in the results.

See also: k-fold CV, leave-one-out (LOO) cross-validation, validointi, validointivirhe.

networked data Networked data consist of local datasets that are related by some notion of pairwise similarity. We can represent networked data using a ?? 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, ??, 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 parameter, GTV.

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

See also: model, local dataset, federoitu oppiminen.

networked model A networked model 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: model, FL network, local model, hypothesis space.

nimiö A label is 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. Synonyms for label, commonly used in specific domains, include "response variable," "output variable," and "target" [?], [?], [?].

See also: data point, label space.

normal equations The optimality condition for the model parameters in linear least squares are often referred to as normal equations.

See also: model parameter, linear least squares.

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 (generation of) data points $\mathbf{z}^{(t)}$ as ?? ?? with a common ?? $\mathbb{P}^{(\mathbf{z})}$. The riski $\mathbb{E}\{L\left(\mathbf{z}, \mathbf{w}\right)\}$ 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 kohdefunktio for learning the model parameters $\mathbf{w}$. Unfortunately, the above limit can only be evaluated if we wait infinitely long in order to collect all data points. However, many koneoppiminen applications require methods that learn online: 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 riski. As its name suggests, online GD updates $\mathbf{w}^{(t)}$ via a (projected) ?? such that

$$\mathbf{w}^{(t+1)} := P_{\mathcal{W}}\big(\mathbf{w}^{(t)} - \eta_t \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)\big). \tag{8}$$

Note that (**??**) is a ?? for the current component $L\left(\mathbf{z}^{(t)}, \cdot\right)$ of the riski. The update (**??**) 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 loss $\sum_{t'=1}^{t-1} L\left(\mathbf{z}^{(t')}, \cdot\right)$. 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 [**?**], [**?**].

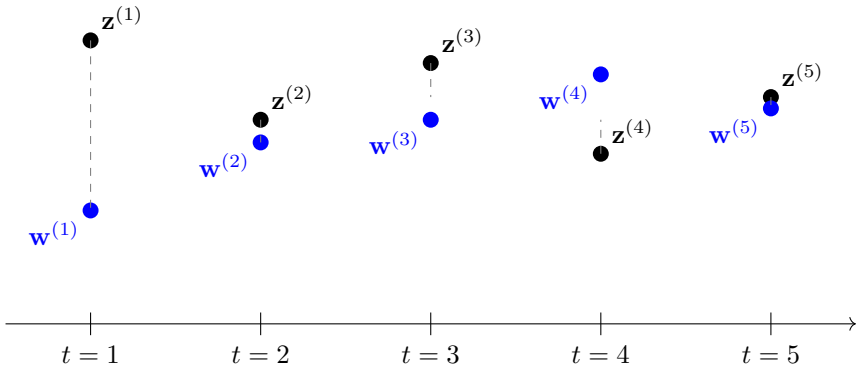


Fig. 50. An instance of online GD that updates the model parameters $\mathbf{w}^{(t)}$ using the data point $\mathbf{z}^{(t)} = x^{(t)}$ arriving at time t. This instance uses the neliövirhehäviö $L\left(\mathbf{z}^{(t)}, w\right) = (x^{(t)} - w)^2$.

See also: kohdefunktio, GD, ??, 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 ?? and ?? temperatures recorded by an Ilmatieteen laitos 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 algoritmi 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 algoritmi, which has the entire input available from the start, an online algoritmi must handle epävarmuus about future inputs and cannot revise past decisions. Similar to an offline algoritmi, we represent an online algoritmi formally as a collection of possible executions. However, the execution sequence for an online algoritmi 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 t, the algoritmi performs a computational step s_t, generates an output out_t, and then subsequently receives the next input (data point) in_{t+1}. A notable example of an online algoritmi in koneoppiminen is online GD, which incrementally updates model parameters as new data points arrive.

See also: algoritmi, data, data point, epävarmuus, koneoppiminen, online GD, model parameter, online learning.

opetusvirhe The average loss 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 loss that is achieved by a solution of ERM.

See also: loss, 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. 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. Consider, for example, the ?? [**?**, Ch. 5]

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}) \tag{9}$$

of a gradient-based method for ERM, where the kohdefunktio $f(\mathbf{w})$ is the empiirinen riski incurred by $h^{(\mathbf{w})}$ on a training set. Given the current model parameters $\mathbf{w}^{(t)}$ at iteration t, the ?? produces updated model parameters $\mathbf{w}^{(t+1)}$ by moving in the opposite direction of the ?? $\nabla f(\mathbf{w}^{(t)})$.

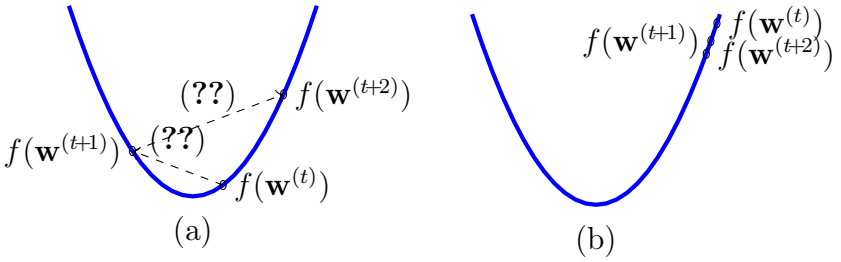


Fig. 51. Effect of using an inadequate learning rate η in the ?? (**??**). (a) If η is too large, the ?? can "overshoot" such that the iterates $\mathbf{w}^{(t)}$ diverge away from the optimum, i.e., $f(\mathbf{w}^{(t+1)}) > f(\mathbf{w}^{(t)})$. (b) If η is too small, the ?? make too little progress towards the optimum within the available number of iterations (due to limited computational budget).

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

oppimistehtävä 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 piirteet and nimiöt. Given a choice of model $\mathcal{H}$ and loss function, a learning task leads to an instance of ERM and can thus be represented by the associated kohdefunktio $\widehat{L}\big(h|\mathcal{D}\big)$ for $h \in \mathcal{H}$. Importantly, multiple distinct learning tasks can be constructed from the same tietoaineisto by selecting different sets of piirteet and nimiöt (see Fig. **??**).

An image showing cows grazing in the Austrian countryside.

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

Task 2 (luokittelu):
Piirteet 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. 52. 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 ?? 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 monitehtäväoppiminen methods, tends to be more effective than solving them independently (thereby ignoring dependencies among learning tasks) [**?**], [**?**], [**?**].

See also: monitehtäväoppiminen, label space.

optimism in the face of uncertainty koneoppiminen methods learn model parameters $\mathbf{w}$ according to some performance criterion $\bar{f}(\mathbf{w})$. However, 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 loss on a given tietoaineisto (i.e., the training set) as an estimate for the riski of a hypothesis. Using a ??, 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 ?? 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 todennäköisyys. 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. **??**). Two examples of koneoppiminen methods that use such an optimistic construction of an kohdefunktio are SRM [**?**, Ch. 11] and upper confidence bound (UCB) methods for sequential decision making [**?**, Sec. 2.2].

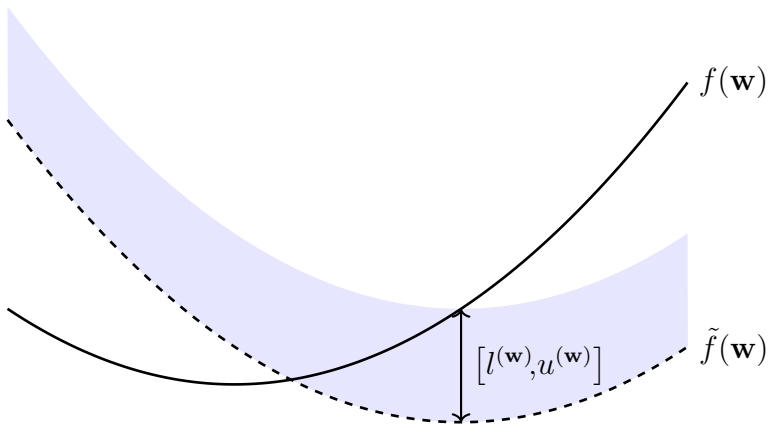


Fig. 53. koneoppiminen methods learn model parameters $\mathbf{w}$ by using some estimate of $f(\mathbf{w})$ for the ultimate performance criterion $\bar{f}(\mathbf{w})$. Using a ??, one can use $f(\mathbf{w})$ to construct confidence intervals $\left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$, which contain $\bar{f}(\mathbf{w})$ with high probability. The best plausible performance ?? for a specific choice $\mathbf{w}$ of model parameters is $\tilde{f}(\mathbf{w}) := l^{(\mathbf{w})}$.

See also: kohdefunktio, ??, gradient-based method, UCB.

outlier Many koneoppiminen methods are motivated by the i.i.d. assumption, which interprets data points as realizations of ?? ?? with a common ??. 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 statis-

tical properties of most data points as an outlier. Different methods for outlier detection use different ?? of this deviation. Statistical learning theory studies fundamental limits on the ability to mitigate outliers reliably [**?**], [**?**].

See also: vakaus, stability, Huber regression, ??.

output The term output is sometimes used as a synonym for the nimiö of a data point [**?**].

See also: nimiö, data point.

output vector The term output ?? is used as a synonym for the label vector of a tietoaineisto [**?**].

See also: output, label vector, tietoaineisto.

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

See also: koneoppiminen, model, hypothesis, ??, lineaarinen malli, weights, ANN.

parameter space The parameter space $\mathcal{W}$ of an koneoppiminen model $\mathcal{H}$ is the set of all feasible choices for the model parameters (see Fig. **??**). Many important koneoppiminen methods use a model that is parameterized by

?? of the Euclidean space $\mathbb{R}^d$. Two widely used examples of parameterized models are lineaariset mallit and deep nets. The parameter space is then often a subset $\mathcal{W} \subseteq \mathbb{R}^d$, e.g., all ?? $\mathbf{w} \in \mathbb{R}^d$ with a ?? smaller than one.

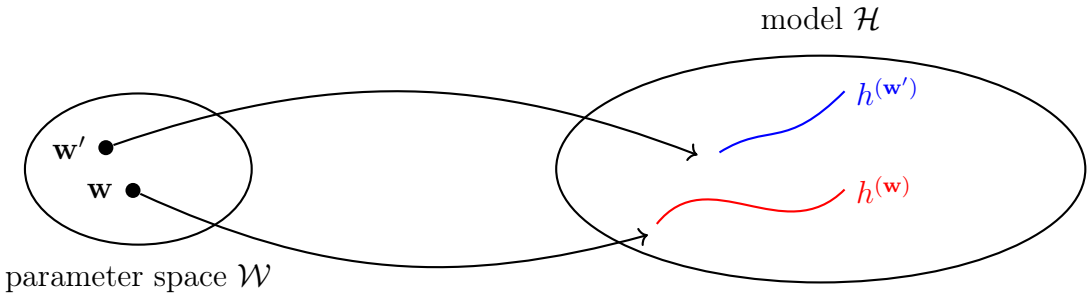


Fig. 54. The parameter space $\mathcal{W}$ of an koneoppiminen model $\mathcal{H}$ consists of all feasible choices for the model parameters. Each choice $\mathbf{w}$ for the model parameters selects a hypothesis ?? $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: parameter, model, model parameter.

parametric model A parametric model is a mathematical model characterized by a finite set of variable quantities called parameters. An important example is the ?? consisting, for a given ?? d, of all ?? (on sample space $\mathbb{R}^d$) with some ?? ?? $\boldsymbol{\mu} \in \mathbb{R}^d$ and ?? $\mathbf{C} \in \mathbb{R}^{d \times d}$. In the context of koneoppiminen, a parametric model defines a hypothesis space $\mathcal{H}$ parameterized by a finite number of model parameters. Each hypothesis $h \in \mathcal{H}$ is uniquely identified by a list of model parameters $w_1, w_2, \ldots, w_d$ (see Fig. **??**). We can stack these parameters into a ?? $\mathbf{w} \in \mathbb{R}^d$. Two widely used examples of parametric models are the lineaarinen malli

and the ANN. The corresponding parameter space is typically a subset of $\mathbb{R}^d$.

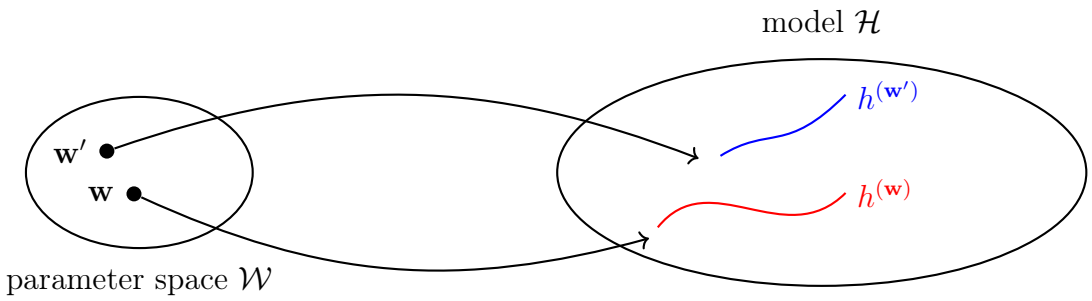


Fig. 55. The parameter space $\mathcal{W}$ of an koneoppiminen model $\mathcal{H}$ consists of all feasible choices for the model parameters. Each choice $\mathbf{w}$ for the model parameters selects a hypothesis ?? $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: parameter space, model, model parameter.

penalty term Consider an ERM-based koneoppiminen method that learns model parameters $\mathbf{w}$ by minimizing the average loss (or empiirinen riski) $\widehat{L}\big(\mathbf{w}|\mathcal{D}\big)$ on a training set $\mathcal{D}$. To avoid ylisovittaminen, and control the generalization gap, it is common to augment the kohdefunktio $\widehat{L}\big(\mathbf{w}|\mathcal{D}\big)$ with a penalty term $\alpha\mathcal{R}\big\{\mathbf{w}\big\}$. We refer to the resulting modified ERM as RERM.

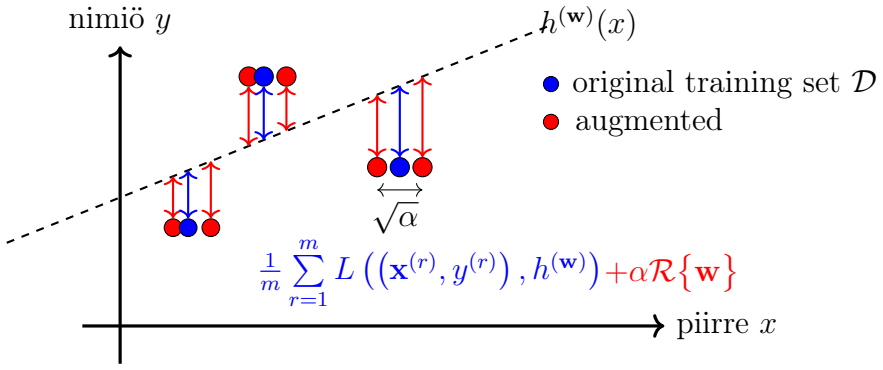


Fig. 56. Adding a penalty term $\alpha\mathcal{R}\{\mathbf{w}\}$ to the kohdefunktio in ERM is equivalent to including perturbations of the training set $\mathcal{D}$ during training. The regularisointi parameter α controls the extend of the perturbations.

The penalty term depends only on the model parameters but not on the data points in the training set. For some combinations of model and loss function, the penalty term can be obtained as the average loss incurred on (an infinite number of) perturbed copies of the training set. In other words, adding a penalty term in ERM can be viewed as a form of data augmentation.

See also: ylisovittaminen, RERM, regularisointi, data augmentation.

perceptron algorithm The perceptron algoritmi is one of the oldest koneoppiminen algoritmit; it was developed by Frank Rosenblatt in 1957 [**?**]. The perceptron algoritmi works on data with numeric piirteet $\mathbf{x}_r \in \mathbb{R}^d$ and binary nimiöt $y_r \in \{-1, 1\}$ and returns a lineaarinen luokitin. It is guaranteed to find such a lineaarinen luokitin if the classes are linearly separable. As the algoritmi is trying to find a ?? $g(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ separating

the classes, i.e., $\text{sgn}(\mathbf{w}^T\mathbf{x}_r) = y_r \; \forall r \in \{1, \ldots, m\}$, at each iteration, a sample r' that is wrongly classified, i.e., $\text{sgn}(\mathbf{w}^T\mathbf{x}'_r) \neq y_{r'}$. Then, the weights $\mathbf{w}$ are updated by $\mathbf{w} \leftarrow \mathbf{w} + y_{r'}\mathbf{x}_{r'}$. While this does not guarantee that the sample is now correctly classified, the error margin has decreased.

See also: binary classification, lineaarinen luokitin, ??.

perplexity The perplexity of an ?? x is defined as e^{H_x}.

See also: ??.

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. **??**, where the the signal samples of a finite-duration audio signal are used as its features.

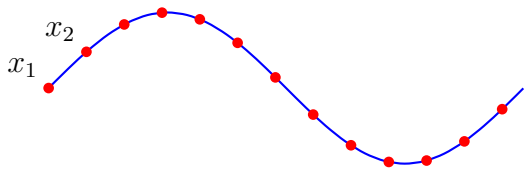


Fig. 57. An audio signal (blue waveform) and its discretized signal samples (red dots) that can be used as its features $x_1, \ldots, x_d$.

Domain-specific synonyms for the term feature are "covariate," "explanatory variable," "independent variable," "input (variable)," "predictor (variable)," or "regressor" [**?**], [**?**], [**?**].

See also: data point.

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

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

that reads in the piirteet $\mathbf{x} \in \mathcal{X}$ of a data point and delivers new piirteet $\mathbf{x}' \in \mathcal{X}'$ 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 ?? $\mathbf{\Phi}$, and for a quantitative ?? of the usefulness of a specific $\mathbf{\Phi} \in \mathcal{H}$. For example, PCA uses $\mathcal{X} := \mathbb{R}^d$, $\mathcal{X}' := \mathbb{R}^{d'}$ with $d' < d$, and a hypothesis space

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

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

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

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

piirrevektori Piirre ?? refers to a ?? $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ whose entries are individual piirteet $x_1, \ldots, x_d$. Many koneoppiminen methods use piirre ?? 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 ?? that belong to an infinite-dimensional ?? (e.g., see kernel method).

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

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

See also: federoitu oppiminen, FL network, local model.

polynomial regression Polynomial regressio is an instance of ERM that learns a polynomial hypothesis ?? to predict a numeric nimiö based on the numeric piirteet 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 ?? 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.

precision Precision is a metric commonly used in binary classification. It measures the proportion of true positives (correctly predicted positive data points) in all the positively predicted data points. I.e., it is the tarkkuus of the positively predicted data points. Precision can also be extended to multiclass luokittelu.

Precision is rarely used in isolation, as it encourages selecting only the most likely data points. Still, unlike tarkkuus, precision remains a valid metric even with imbalanced tietoaineistot.

See also: sekaannusmatriisi, metric, recall, luokittelu.

principal component analysis (PCA) Consider a tietoaineisto

$$\mathcal{D} = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right\}$$

constituted by data points characterized by piirrevektorit $\mathbf{x}^{(r)} \in \mathbb{R}^d$, for $r = 1, \ldots, m$. PCA determines, for a given number $d' < d$ a linear feature map

$$\mathbf{\Phi}^{(\mathbf{W})} : \mathbb{R}^d \to \mathbb{R}^{d'} : \mathbf{x} \mapsto \mathbf{W}\mathbf{x}$$

such that the new piirrevektorit $\mathbf{z}^{(r)} = \mathbf{W}\mathbf{x}^{(r)}$ allow us to reconstruct the original piirteet with ?? linear reconstruction error [**?**, **?**, **?**]

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

We can view PCA as a form of ERM using the loss function $L\left(\mathbf{x}, \mathbf{W}\right) = \left\| \mathbf{x} - \widehat{\mathbf{R}}\mathbf{W}\mathbf{x} \right\|_2^2$ with a reconstruction ?? $\widehat{\mathbf{R}}$ that achieves the above minimum reconstruction error. It turns out that this ERM problem can be solved by a ?? $\mathbf{W} = \left(\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(d')}\right)^T$ whose rows are given by d' ?? corresponding to the d' largest ?? of the ??

$$\widehat{\mathbf{Q}} = (1/m) \sum_{r=1}^{m} \mathbf{x}^{(r)} \left(\mathbf{x}^{(r)}\right)^T = \mathbf{X}^T\mathbf{X}.$$

Note that $\widehat{\mathbf{Q}}$ coincides with the ?? of $\mathcal{D}$ if its sample mean vanishes. The ?? ?? $\widehat{\mathbf{Q}}$ allows for an ?? of the form [**?**, **?**]

$$\widehat{\mathbf{Q}} = \sum_{j=1}^{d} \lambda_j \mathbf{u}^{(d)} \left(\mathbf{u}^{(d)}\right)^T = \begin{pmatrix} \mathbf{u}^{(1)} & \cdots & \mathbf{u}^{(d)} \end{pmatrix} \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_d \end{pmatrix} \begin{pmatrix} \left(\mathbf{u}^{(1)}\right)^T \\ \vdots \\ \left(\mathbf{u}^{(d)}\right)^T \end{pmatrix}.$$

This decomposition consists of decreasing nonnegative ?? $\lambda_1 \geq \lambda_2 \geq \ldots, \geq \lambda_d \geq 0$ and corresponding ?? $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(d)}$ that form an orthonormal basis of $\mathbb{R}^d$.

See also: feature map, piirreoptimointi, ulottuvuuksien vähentäminen, ??.

privacy funnel The privacy funnel is a method for learning a feature map that provides privacy-friendly piirteet for a data point [**?**].

See also: data point, piirre, ??, differentiaalinen yksityisyys.

privacy leakage Consider an koneoppiminen application that processes a tietoaineisto $\mathcal{D}$ and delivers some output, such as the ennusteet 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 ?? for the data generation, we can measure the privacy leakage via the MI between the output and the sensitive piirre. Another quantitative ?? of privacy leakage is differentiaalinen yksityisyys. The relations between different ?? of privacy leakage have been studied in the literature (see [**?**]).

See also: MI, differentiaalinen yksityisyys, yksityisyyshyökkäys, ??.

päätösalue Consider a hypothesis ?? 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, ??, nimiö, piirre.

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

with different ?? values.

See also: hypothesis, ??, piirrevektori, ??, päätösalue, naapurusto, ??.

Q-learning Q-learning is a popular ?? algoritmi that learns an optimal policy by estimating the optimal action-value function (or Q-function) [**?**].

See also: ??, ??.

Rademacher complexity Similar to the VC dimension, the Rademacher complexity [**?**] is a quantitative ?? of the size of a hypothesis space $\mathcal{H}$. It is based on the empirical Rademacher complexity, which is defined for a given tietoaineisto $\mathcal{D}$ as

$$\mathcal{R}_{\mathcal{D}}(\mathcal{H}) = \mathbb{E}_{\varepsilon_1, \ldots, \varepsilon_m} \sup_{h \in \mathcal{H}} \frac{1}{m} \sum_{r=1}^{m} \varepsilon_r h\big(\mathbf{x}^{(r)}\big).$$

Here, the expectation is taken with respect to the ?? $\varepsilon_1, \ldots, \varepsilon_m$, which are ?? and take values in $\{-1, +1\}$ with equal todennäköisyys $1/2$. The Rademacher complexity of $\mathcal{H}$ is then defined as the expectation of the empirical Rademacher complexity of a random tietoaineisto $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ that consists of m ?? ?? $\mathbf{x}^{(r)} \in \mathcal{X}$, for $r = 1, \ldots, m$.

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

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

See also: decision tree, tietoaineisto.

random projection A random ?? uses a random ?? $\mathbf{A} \in \mathbb{R}^{d' \times d}$, with $d' < d$, to map a piirrevektori $\mathbf{x} \in \mathbb{R}^d$ to a shorter piirrevektori $\mathbf{A}\mathbf{x} \in \mathbb{R}^{d'}$. It is a basic method for piirreoptimointi and ulottuvuuksien vähentäminen. The ?? ?? $\mathbf{A}$ is typically generated entry-wise by ?? ?? with a common ?? $\mathbb{P}$. For a broad class of such ??, a random ?? approximately preserves pairwise Euclidean distances between piirrevektorit of a given finite tietoaineisto. The celebrated ?? guarantees the existence of such a distance-preserving ulottuvuuksien vähentäminen ?? but does not itself involve randomness. Random ?? provide a probabilistic construction that realizes this guarantee with high todennäköisyys. Roughly speaking, for many relevant applications, random ?? preserve the most relevant information contained in the original (typically very long) piirrevektori. Fig. **??** illustrates this behavior for an RGB image. The left panel shows the original image. The middle panel shows a masked image where a randomly selected five percent of the original pixels are kept, and the remaining pixels are set to a fixed light-gray color. The right panel shows the result of a simple reconstruction based on repeated averaging of nearby retained pixels.

original 95 % masked reconstructed

(a) (b) (c)

Fig. 58. Illustration of a random ?? in the form of removing (or masking) all image pixels, except those in a small random subset. (a) The left panel shows the original RGB image. (b) The middle panel shows a version with only a random five percent subset of pixels retained. (c) The right panel shows a simple convolution-based reconstruction that diffuses information from the known pixels into masked regions.

See also: piirreoptimointi, ulottuvuuksien vähentäminen, ??.

Python demo: click me

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

See also: ??, ??, ??, ??.

recall Recall is a metric commonly used in binary classification. It is the ratio of true positives (correctly predicted positive data points) to all actual positives (data points with positive true nimiö). I.e., it tells the proportion of true positives correctly labelled. Recall can also be extended to multiclass luokittelu.

Recall is not usually used alone, as perfect recall can be achieved by labelling all points as positive. Still, it is robust against imbalanced tietoaineistot unlike, e.g., tarkkuus.

See also: sekaannusmatriisi, metric, precision, luokittelu.

receiver operating characteristic (ROC) Consider a label space $\mathcal{Y} = \{-1, 1\}$ and a luokitin that uses a real-valued hypothesis $h(\mathbf{x})$. For a given threshold $\eta \in \mathbb{R}$, the ultimate ennuste is $\hat{y} = 1$ if $h(\mathbf{x}) \geq \eta$ and $\hat{y} = -1$ otherwise. On a test set $\mathcal{D}^{(\text{test})}$, we compute, for each value of η, the following two quantities: 1) true positive rate $\text{TPR}^{(\eta)}$; and 2) false positive rate $\text{FPR}^{(\eta)}$. The ROC curve is the ?? [**?**]

$$\text{ROC} : \mathbb{R} \to \mathbb{R}^2 : \eta \mapsto \left(\text{TPR}^{(\eta)}, \text{FPR}^{(\eta)}\right).$$

One important characteristic of the ROC curve is the area under the curve (AUC).

See also: luokitin, AUC.

rectified linear unit (ReLU) The ReLU is a popular choice for the aktivointifunktio 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: aktivointifunktio, ANN.

recurrent neural network (RNN) An RNN is a specific type of ANN that is designed for processing data that consist of a ?? of tokens. An RNN maintains an internal hidden state that is updated recurrently as new tokens are processed. This recurrent dependence allows information to propagate across time steps, making RNNs suitable for tasks such as

speech recognition, language modeling, or time series ennuste. However, their inherently sequential computation limits parallelization and is challenging for gradient-based methods. Variants like the long short-term memory (LSTM) and gated recurrent unit (GRU) mitigate these problems.

See also: ANN, token.

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

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

regularisoija A regularizer assigns each hypothesis h from a hypothesis space $\mathcal{H}$ a quantitative ?? $\mathcal{R}\big\{h\big\}$ 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 ?? $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [**?**, Ch. 3]. Lasso uses the regularizer $\mathcal{R}\big\{h\big\} := \|\mathbf{w}\|_1$ for linear hypothesis ?? $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [**?**, Ch. 3].

See also: harjaregressio, Lasso, loss, kohdefunktio.

regularisointi A key challenge of modern koneoppiminen applications is that they often use large models, which have an effective dimension in the order of billions. Training a high-dimensional model 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) Model pruning: We prune the original model $\mathcal{H}$ to obtain a smaller model $\mathcal{H}'$. For a parametric model, 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) Loss penalization: We modify the kohdefunktio of ERM by adding a penalty term to the opetusvirhe. The penalty term estimates how much higher the expected loss (or riski) is compared with the average loss 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 a ?? to the piirrevektori of a data point.

Fig. **??** illustrates the above three routes to regularization. These routes are closely related and sometimes fully equivalent. Data augmentation using ?? to perturb the piirrevektorit 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 model pruning.

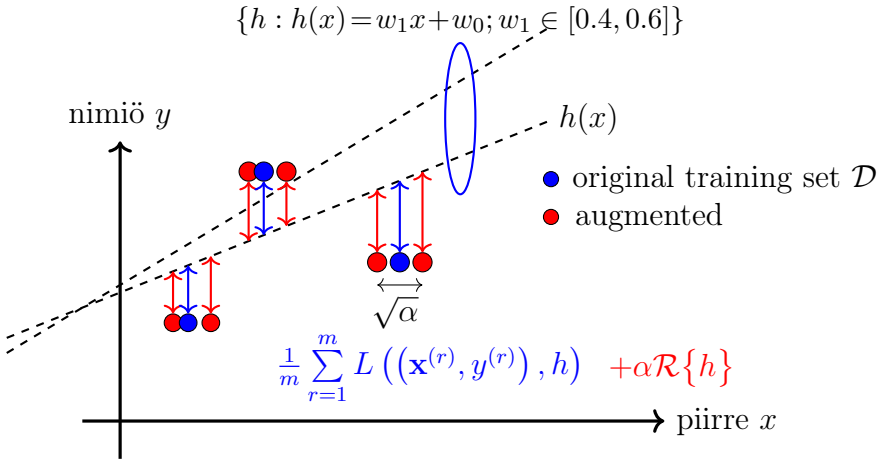


Fig. 59. Three approaches to regularization: 1) data augmentation; 2) loss penalization; and 3) model pruning (via constraints on model parameters).

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

regularized empirical risk minimization (RERM) Basic ERM learns a hypothesis (or trains a model) $h \in \mathcal{H}$ based solely on the empiirinen riski $\widehat{L}\big(h|\mathcal{D}\big)$ 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 kohdefunktio of (**??**) can be intuitively understood as a surrogate for the increased average loss 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 lineaarinen malli 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 loss caused by adding ?? to the piirrevektorit in the training set [**?**, 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, loss, SRM.

regularized loss minimization (RLM) See RERM.

response The term response is sometimes used as a synonym for the nimiö of a data point [**?**].

See also: nimiö, data point, target.

response vector The term response ?? is used as a synonym for the label vector of a tietoaineisto [**?**].

See also: response, label vector, tietoaineisto, target vector.

reward A reward refers to some observed (or measured) quantity that allows us to estimate the loss 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: loss, ??, ??.

risk stratification Risk stratification assigns data points to risk groups (or strata) by quantizing the ennuste $\hat{h}(\mathbf{x})$ obtained from a trained risk prognosis model. Typical choices for these strata include low, intermediate, and high risk [**?**], [**?**].

See also: stratification, ennuste, clustering.

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

See also: ?? ??, i.i.d. assumption, loss, ??.

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

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

See also: ??, ??, tietoaineisto, piirrevektori.

sample size The number of individual data points contained in a ?? or tietoaineisto. Consider a ERM-based method that uses a training set $\mathcal{D}$ with sample size m and a model $\mathcal{H}$ with effective dimension $d_{\text{eff}}(\mathcal{H})$. If the training set can be well-approximated by the i.i.d. assumption, then the ratio between m and $d_{\text{eff}}(\mathcal{H})$ can be a useful indicator for the occurence of ylisovittaminen [**?**, Ch. 6].

See also: data point, tietoaineisto.

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

See also: ??.

sample weighting Consider an ERM-based method that learns a hypothesis by minimizing the average loss on a training set. In its basic form, ERM treats all data points equally important. However, in some applications, it can be useful to put different emphasis on the ennuste errors obtained for different data points. For example, if a data point is considered an outlier we should reduce its influence on the learned hypothesis. We can implement this idea by assigning a nonnegative weight $q^{(r)}$ to each data point $\left(\mathbf{x}^{(r)}, y^{(r)}\right)$ in the training set. This results in the weighted ERM principle

$$\min_{h \in \mathcal{H}} \sum_{r=1}^{m} q^{(r)} \, L\left(\left(\mathbf{x}^{(r)}, y^{(r)}\right), h\right). \tag{11}$$

Fig. **??** illustrates the concept of a training set of three data points that contribute unequally to the empiirinen riski.

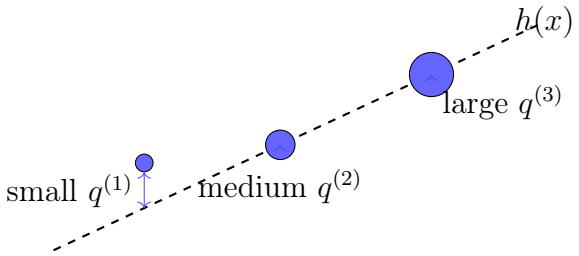


Fig. 60. Sample weighting assigns each data point of a training set a weight $q^{(r)}$. Assigning a small weight (such as $q^{(1)}$ in this example) to a data point decreases its influence on the hypothesis learned via solving (**??**).

See also: ERM, outlier, AdaBoost.

satunnaisalgoritmi A ?? algoritmi uses a random mechanism during its execution. For example, SGD uses a randomly selected subset of data points to compute an approximation for the ?? of an kohdefunktio. We can represent a ?? algoritmi by a ??, i.e., the possible execution sequence is the possible ?? of a ?? [**?**], [**?**], [**?**].

See also: ??, algoritmi, SGD, data point, ??, kohdefunktio, ??, ??, ??, gradient-based method.

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

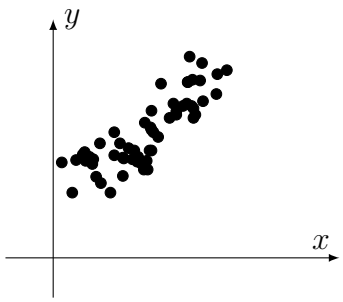


Fig. 61. A scatterplot with circle markers, where the data points represent daily weather conditions in Finland. Each data point is characterized by its ?? daytime temperature x as the piirre and its ?? daytime temperature y as the nimiö. The temperatures have been measured at the Ilmatieteen laitos 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 piirrevektorit in high-dimensional spaces.
See also: data point, ??, piirre, ??, nimiö, Ilmatieteen laitos, piirrevektori, ulottuvuuksien vähentäminen.

sekaannusmatriisi Consider a finite tietoaineisto with m data points, each characterized by a piirrevektori $\mathbf{x}$ and a nimiö $y \in \mathcal{Y}$ with a finite label space $\mathcal{Y} = \{1, \ldots, k\}$. For a given hypothesis h, the confusion ?? is a $k \times k$?? where each row corresponds to a specific value of the true nimiö $y \in \mathcal{Y}$ and each column to a specific value of the ennuste $h(\mathbf{x}) \in \mathcal{Y}$. The ?? entry in the cth row and c'th column is the number of data points with the true nimiö $y = c$ that are predicted as $h(\mathbf{x}) = c'$. The sum of the main diagonal entries is the number of correctly classified data points, i.e., those for which $y = h(\mathbf{x})$. Summing the off-diagonal entries

resuls in the total number of data points that are misclassified by h.

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

self-supervised learning Self-supervised learning uses some of the piirteet of a data point as its nimiö. For example, if a data point consists of a sentence within a text document, we can use the last word of the sentence as the nimiö that is to be predicted from all the previous words, which form the piirteet of the data point. A main application of self-supervised learning is in NLP for the training of LLMs from large collections of text data.

See also: piirre, nimiö, LLM.

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

See also: ??, regularisointi.

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

See also: ennuste, selitys, koneoppiminen, model.

selitys One approach to enhance the ?? of an koneoppiminen method for its human user is to provide an explanation alongside the ennusteet 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 piirteet of a given data point [**?**]. Alternatively, explanations can be visual—for example, intensity ?? that highlight image regions that drive the ennuste [**?**]. Fig. **??** illustrates two types of explanations. The first is a local linear approximation $g(\mathbf{x})$ of a nonlinear trained model $\hat{h}(\mathbf{x})$ around a specific piirrevektori $\mathbf{x}'$, as used in the method LIME. The second form of explanation depicted in the figure is a sparse set of ennusteet $\hat{h}(\mathbf{x}^{(1)}), \hat{h}(\mathbf{x}^{(2)}), \hat{h}(\mathbf{x}^{(3)})$ at selected piirrevektorit, offering concrete reference points for the user.

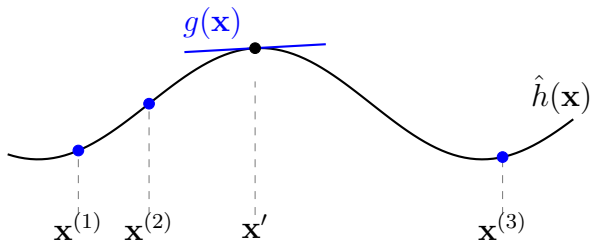


Fig. 62. A trained model $\hat{h}(\mathbf{x})$ can be explained locally at some point $\mathbf{x}'$ by a linear approximation $g(\mathbf{x})$. For a ?? $\hat{h}(\mathbf{x})$, this approximation is determined by the ?? $\nabla\hat{h}(\mathbf{x}')$. Another form of explanation could be the ?? values $\hat{h}\big(\mathbf{x}^{(r)}\big)$ for $r = 1, 2, 3$.

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

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 ?? that allows us to predict the nimiö of a data point from its piirteet. 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, ??, 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 ?? $\mathcal{G} = \big(\mathcal{V} := \{1, \ldots, m\}, \mathcal{E}\big)$. 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, ??.

skip connection Consider a deep net with neurons that are organized in consecutive layers. A skip connection links the output of a neuron in some layer to the input of a neuron in a non-consecutive layer [**?**].

See also: deep net, ??.

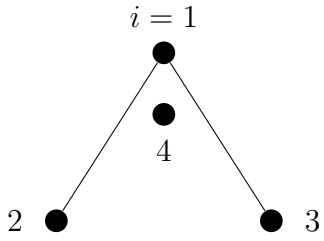
soft clustering Soft clustering refers to the task of partitioning a given set of data points into (a few) overlapping clusters. Each data point is assigned to several different clusters with varying degrees of belonging. Soft clustering methods determine the degree of belonging (or soft cluster assignment) for each data point and each cluster. A principled approach to soft clustering for data points characterized by numerical piirrevektorit is via a ?? such as the Gaussin sekoitemalli. The conditional todennäköisyys of a data point belonging to a specific mixture component is then a natural choice for the degree of belonging. Gaussin sekoitemalli soft clustering methods can be applied to non-numeric data by using piirreoptimointi methods to provide numerical piirteet (such as in spectral clustering).

See also: clustering, cluster, degree of belonging, Gaussin sekoitemalli, spectral clustering.

soft label Consider a luokittelu problem where data points are characterized by piirteet $\mathbf{x}$ and nimiöt from a finite label space $\mathcal{Y} = \{1, \ldots, k\}$. Some koneoppiminen applications involve data points that have almost identical piirteet but different nimiöt. In such cases, instead of assigning to each data point a single nimiö $y \in \mathcal{Y}$, it can be more useful to assign an entire ?? over the label space. This ?? can be represented as a ?? $\mathbf{y} = \big(y_1, \ldots, y_k\big)^T \in \Delta^k$. We can view the entries y_c, for $c = 1, \ldots, k$, as soft nimiöt of a data point. Mathematically, the soft nimiö y_c is the todennäköisyys that a randomly chosen data point with piirrevektori $\mathbf{x}$ has nimiö c.

See also: luokittelu, ??, cross-entropy.

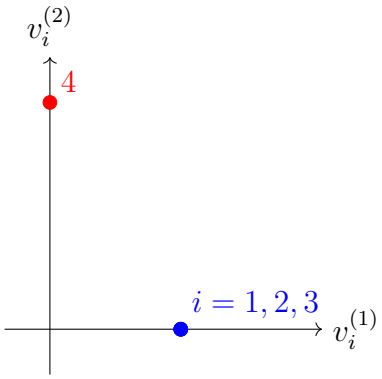
spectral clustering Spectral clustering is a particular instance of graph clustering, i.e., it clusters data points represented as the nodes $i = 1, \ldots, n$ of a ?? $\mathcal{G}$. Spectral clustering uses the ?? of the Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$ to construct piirrevektorit $\mathbf{x}^{(i)}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



(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)

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

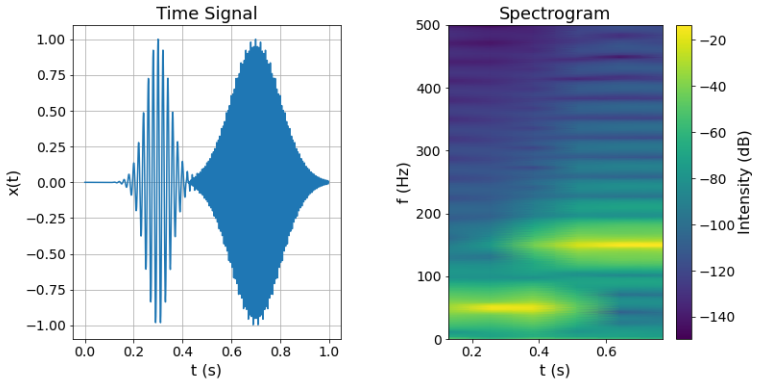
$$\mathbf{v}^{(1)} = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \; \mathbf{v}^{(2)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

(d)

Fig. 63. (a) An undirected ?? $\mathcal{G}$ 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 ??. (c) A scatterplot of data points using the piirrevektorit $\mathbf{x}^{(i)} = \left(v_i^{(1)}, v_i^{(2)}\right)^T$. (d) Two ?? $\mathbf{v}^{(1)}, \mathbf{v}^{(2)} \in \mathbb{R}^d$ corresponding to the ?? $\lambda = 0$ of the Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$.

See also: clustering, graph clustering, Laplacian matrix, ??.

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. **??** depicts a time signal along with its spectrogram.



(a) (b)

Fig. 64. (a) A time signal consisting of two modulated ?? 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 Mathematically, an koneoppiminen method is a ?? $\mathcal{A}$ from a given tietoaineisto $\mathcal{D}$ to an output $\mathcal{A}(\mathcal{D})$. As a case in point, consider an ERM-based koneoppiminen method that maps a training set $\mathcal{D}$ to the learned model parameters $\mathcal{A}(\mathcal{D}) = \widehat{\mathbf{w}}$, which achieve the ?? average loss on the training set. Instead of the learned model parameters, the output $\mathcal{A}(\mathcal{D})$ could also be the ennusteet obtained from the trained model. Stability refers to the desirable property of $\mathcal{A}$ that small changes in the input tietoaineisto $\mathcal{D}$ result in small changes in the output $\mathcal{A}(\mathcal{D})$. The notion of stability is intimately related to the notion of yleistys. In particular, there are formal notions of stability that allow us to bound the generalization gap (see [**?**, Ch. 13]). To build intuition, consider the three tietoaineistot depicted in Fig. **??**, each of which is equally likely under the same data-generating ??. Since the optimal model parameters are determined by this underlying ??, 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 ?? realizations from the same ??, i.e., it must be stable.

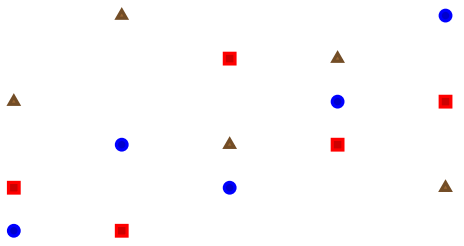


Fig. 65. Three tietoaineistot $\mathcal{D}^{(*)}$, $\mathcal{D}^{(\square)}$, and $\mathcal{D}^{(\triangle)}$, each sampled independently from the same data-generating ??. A stable koneoppiminen method should return similar outputs when trained on any of these tietoaineistot.

See also: yleistys, vakaus.

stacking Stacking is one of the main types of ensemble methods. In stacking, a finite number M of base learners are trained on the same tietoaineisto but with different models $\mathcal{H}^{(j)}$ or loss functiont $L^{(j)}$, for $j = 1, \ldots, M$ [**?**, Ch. 8.8], [**?**], [**?**]. The jth base learner delivers a learned hypothesis $\widehat{h}^{(j)} \in \mathcal{H}^{(j)}$. The final ennuste for a data point is obtained by aggregating the ennusteet of the base learners via an aggregation rule $\phi_{\rm agg}$, such as majority voting for luokittelu or averaging for regressio. We can interpret stacking as a form of piirreoptimointi, where each base learner extracts a new piirre. The aggregation rule can be obtained by another instance of ERM that learns a hypothesis ?? $\phi_{\rm agg} \in \mathcal{H}$ from a meta-model $\mathcal{H}$. The hypothesis $\phi_{\rm agg}$ is applied to the transformed piirrevektori

$$\mathbf{\Phi}(\mathbf{x}) = \big(\hat{h}^{(1)}(\mathbf{x}), \ldots, \hat{h}^{(M)}(\mathbf{x})\big)^{T}.$$

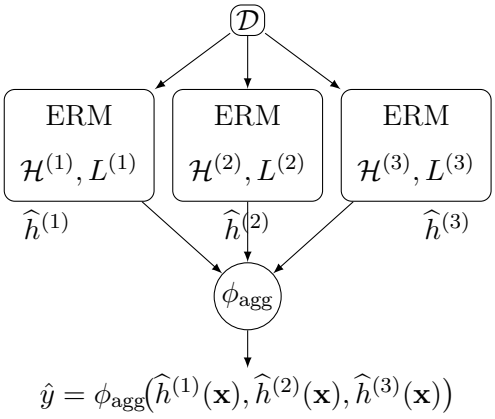


Fig. 66. Three base learners using ERM with different models and loss functiont to obtain learned hypotheses $\widehat{h}^{(1)}$, $\widehat{h}^{(2)}$, $\widehat{h}^{(3)}$. For a data point with piirrevektori $\mathbf{x}$, each base learner delivers a ennuste $\widehat{h}^{(j)}(\mathbf{x})$, for $j = 1, 2, 3$. These ennusteet are then used as new piirteet for an aggregation rule ϕ_{agg} that delivers the overall ennuste $\hat{y}$. The aggregation rule can be obtained by training a meta-model $\mathcal{H}$.

See also: ensemble, bagging.

step size See oppimisnopeus.

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

See also: model, ??, cluster, todennäköisyys, nimiö.

stochastic gradient descent (SGD) SGD is obtained from GD by replacing the ?? of the kohdefunktio with a ?? approximation. A main application of SGD is to train a parameterized model 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 ?? of the empiirinen riski (as a ?? 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 ?? approximation to the ?? 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. **??**). 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 [**?**].

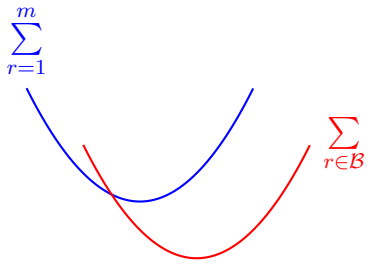


Fig. 67. SGD for ERM approximates the ?? by replacing the sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over all data points in the training set (indexed by $r = 1, \ldots, m$) with a sum over a randomly chosen subset $\mathcal{B} \subseteq \{1, \ldots, m\}$.

See also: GD, ??, kohdefunktio, ??, model, ERM, training set, data

point, empiirinen riski, ??, model parameter, 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 model, such as a lineaarinen malli 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: algoritmi, gradient-based method.

stratification The process of splitting a tietoaineisto into subsets, so called strata, according to some key attribute is called stratification [**?**], [**?**], [**?**]. The goal is to ensure that an koneoppiminen method performs well for each stratum defined by these attributes. For example, in a medical tietoaineisto, we may want to stratify a patient tietoaineisto by age groups to ensure that an koneoppiminen model performs well across all age groups.

When splitting a tietoaineisto into a training set and a validation set, stratification ensures that both sets have similar distributions of the key attribute. Without stratification, using a small validation set may underrepresent or even completely miss data points with a rare attribute, leading to misleading performance estimates. See Fig. **??** for a visual illustration.

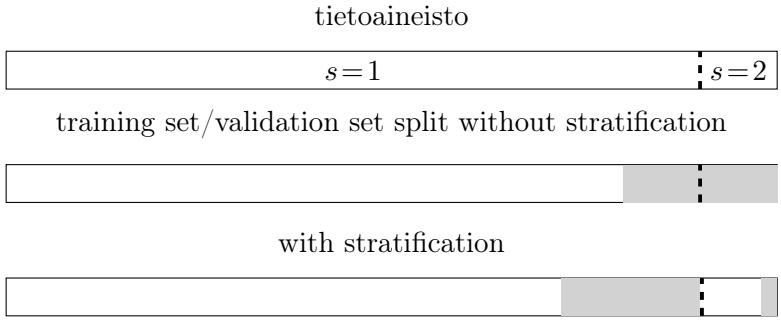


Fig. 68. Stratification ensures that both the training set and the validation set (shaded grey) have similar distributions of a binary key attribute s. In other words, with stratification, both strata (i.e., $s=1$ and $s=2$ based on the key attribute) allocate the same validation proportion—20% of their own width.

See also: stratum, validointi, k-fold CV.

stratum A stratum is a subset of data points that all share a common property (which could be a piirre or a nimiö). For example, in a weather tietoaineisto, all measurements from the same Ilmatieteen laitos weather station form one stratum.

Example (CSV snippet):

```
time, station, value, unit
2023-06-01 12:00, Helsinki, 18.2, degree Celsius
2023-06-01 13:00, Helsinki, 18.5, degree Celsius
2023-06-01 14:00, Helsinki, 19.0, degree Celsius
  2023-06-01 12:00, Oulu, 12.1, degree Celsius
  2023-06-01 13:00, Oulu, 12.4, degree Celsius
```

```
2023-06-01 14:00, Oulu, 12.7, degree Celsius
2023-06-01 12:00, Tampere, 15.3, degree Celsius
2023-06-01 13:00, Tampere, 15.6, degree Celsius
2023-06-01 14:00, Tampere, 16.0, degree Celsius
```

Here, the rows for each station (i.e., `Helsinki`, `Oulu`, `Tampere`) represent different strata.

See also: data point, tietoaineisto, stratification.

structural risk minimization (SRM) SRM is an instance of RERM, with which the model $\mathcal{H}$ can be expressed as a ?? 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, model, yleistys, ERM, regularisoija, riski.

subgradient descent ?? descent is a yleistys of GD that does not require differentiability of the ?? to be minimized. This generalization is obtained by replacing the concept of a ?? with that of a ??. Similar to ??, ?? allow us to construct local approximations of an kohdefunktio. The kohdefunktio might be the empiirinen riski $\widehat{L}\big(h^{(\mathbf{w})}\big|\mathcal{D}\big)$ viewed as a ?? of the model parameters $\mathbf{w}$ that select a hypothesis $h^{(\mathbf{w})} \in \mathcal{H}$.

See also: ??, yleistys, GD, ??, ??, kohdefunktio, empiirinen riski, model parameter, hypothesis.

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

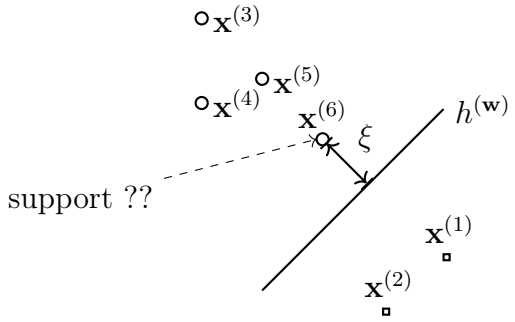


Fig. 69. The SVM learns a hypothesis (or luokitin) $h^{(\mathbf{w})}$ with minimal average soft-margin hinge loss. Minimizing this loss is equivalent to maximizing the margin ξ between the päätöspinta of $h^{(\mathbf{w})}$ and each class of the training set.

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, lineaarinen malli, luokitin, hinge loss.

suurimman uskottavuuden menetelmä Consider data points $\mathcal{D} = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$

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

See also: ??, ??, ??.

target The term target is sometimes used as a synonym for the nimiö of a data point [**?**], [**?**].

See also: nimiö, data point.

target vector The term target ?? is used as a synonym for the label vector of a tietoaineisto [**?**], [**?**].

See also: target, label vector, tietoaineisto.

tarkkuus Consider data points characterized by piirteet $\mathbf{x} \in \mathcal{X}$ 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 binääritappio $L^{(0/1)}\left(\cdot, \cdot\right)$.

See also: binääritappio, loss, metric.

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

See also: data point, model, ERM, validation set.

tietoaineisto A dataset is a set of distinct data points. Strictly speaking, a dataset is an unordered collection of data points that does not contain any repetitions. However, in koneoppiminen literature, the term dataset is often used as a synonym for ??, i.e., a ?? (or finite list) of data points that may contain repetitions. koneoppiminen methods use datasets for model training and validointi. The notion of a dataset is broad, i.e., data points may represent concrete physical entities (such as humans or animals) or abstract objects (such as numbers). For illustration, Fig. **??** 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. **??** would require visiting the cow herd. In practice, we work with a more convenient representation (or approximation) of the dataset. Various mathematical models have been developed for this purpose [**?**], [**?**], [**?**], [**?**]. One of the most widely used is the relational model, which organizes data as a table (or relation) [**?**], [**?**]. A table consists of rows and columns, where each row corresponds to a single data point, and each column represents a specific attribute of a data point. koneoppiminen methods

typically interpret these attributes as piirteet or as a nimiö of a data point. As an illustration, Table **??** shows a relational representation of the dataset from Fig. **??**. In the relational model, the order of rows is immaterial, and each attribute (i.e., 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. **??**

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

While the relational model is useful for the study of many koneoppiminen applications, it may be insufficient regarding the requirements for ??. 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, feature space, label space.

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

See also: koneoppiminen, ??, ??, data.

todennäköisyys We assign a probability value, typically chosen in the interval $[0, 1]$, to each ?? that can occur in a ?? [**?**], [**?**], [**?**], [**?**].
See also: ??, ??.

token A token is a basic unit of information obtained by splitting a ?? of symbols, such as a text string, into smaller parts. In NLP, tokens often correspond to words, subwords, or characters that form the piirteet of a data point. Tokenization transforms raw text (e.g., "The cat sleeps") into a ?? of tokens (e.g., ["The", "cat", "sleeps"]), which can then be mapped to numerical piirrevektorit.
See also: ??, piirrevektori.

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 model $\mathcal{H}$. The training of a model $\mathcal{H}$ is guided by the loss 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 parameters $\mathbf{w}$. A widely-used approach to training is ERM, which learns a hypothesis by minimizing the average loss incurred on a training set. One of the main challenges in koneoppiminen is to control the poikkeama between the loss incurred on the training set and the loss incurred on other (unseen) data points.
See also: model, loss, ERM.

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 loss 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) [**?**, Sec. 6.6].

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

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

See also: oppimistehtävä, monitehtäväoppiminen

transformer In the context of koneoppiminen, the term transformer refers to an ANN that uses some form of attention mechanism to capture dependencies among tokens [**?**]. The attention mechansim is what sets transformers apart from previous models used for sequential data such as recurrent neural networks (RNNs). A transformer ANN often combines several attention layers via more traditional layer architectures.

See also: attention, NLP.

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 model behavior [**?**], [**?**], [**?**], [**?**], [**?**]. The idea is as follows: If a human user understands an koneoppiminen method, then they should be able

to anticipate its ennusteet on a test set. We illustrate such a test set in Fig. **??**, 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 ??. Since $\hat{h}$ corresponds to a ??, the user can anticipate the ennusteet 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.

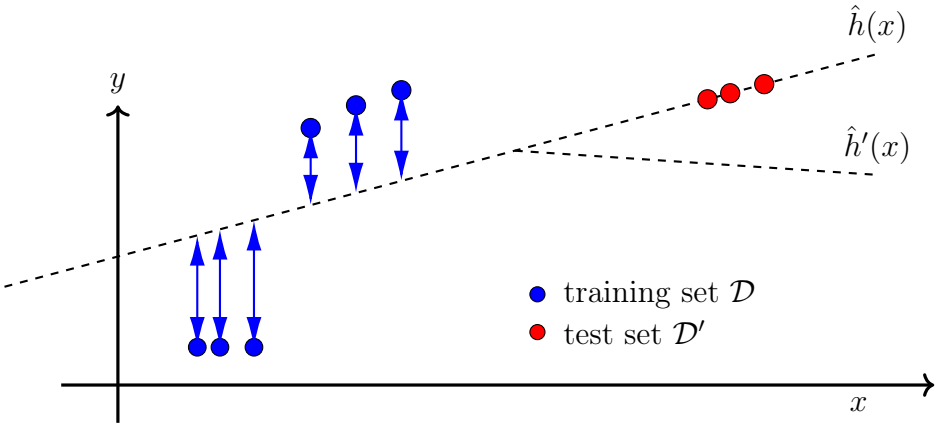


Fig. 71. We can assess the interpretability of trained koneoppiminen models $\hat{h}$ and $\hat{h}'$ by comparing their ennusteet to pseudo-nimiöt generated by a human user for $\mathcal{D}'$.

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. **??**, interpretability of an koneoppiminen method $\hat{h}$ requires that the human user can anticipate its ennusteet on an arbitrary test set. This contrasts with selitettävyys, where the

user is supported by external selitteet—such as saliency ?? or reference examples from the training set—to understand the ennusteet of $\hat{h}$ on a specific test set $\mathcal{D}'$.

See also: selitettävyys, ??, regularisointi, LIME.

ulottuvuuksien vähentäminen Dimensionality reduction refers to methods that learn a transformation $h : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ of a (typically large) set of raw piirteet $x_1, \ldots, x_d$ into a smaller set of informative piirteet $z_1, \ldots, z_{d'}$. Using a smaller set of piirteet is beneficial in several ways:

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

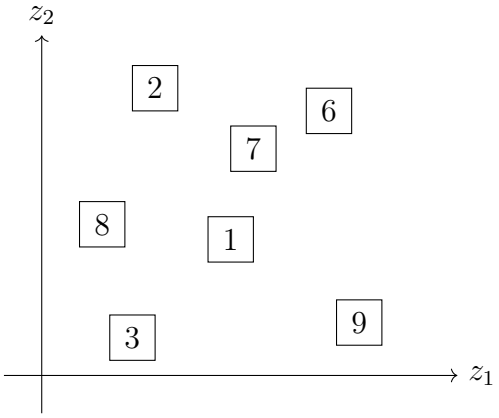


Fig. 72. Example of dimensionality reduction: High-dimensional image data (e.g., high-resolution images of handwritten digits) embedded into 2-D using learned piirteet (z_1, z_2) and visualized in a scatterplot.

See also: ylisovittaminen, autoenkoodaaja, random projection, PCA, ??.

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

their estimated ?? 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 ?? bounds, are established in [**?**]. See also: optimism in the face of uncertainty, reward, ??, epävarmuus, ??.

uusio-otanta For the analysis of koneoppiminen methods, it is often useful to interpret a given tietoaineisto $\mathcal{D} = \left\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\right\}$ as a collection of (realizations of) ?? ?? with common ?? $\mathbb{P}$. In practice, the ?? $\mathbb{P}$ is unknown and must be estimated from $\mathcal{D}$. The idea of the bootstrap method is to use the empirical distribution $\mathbb{P}^{(\mathcal{D})}$ of $\mathcal{D}$ as an estimator for $\mathbb{P}$ [**?**],

$$(1/m)\left|r : \mathbf{z}^{(r)} \in \mathcal{A}\right| \approx \mathbb{P}\left(\mathcal{A}\right).$$

Repeatedly sampling from the empirical distribution, which is equivalent to sampling with replacement from $\mathcal{D}$ [**?**], results in new tietoaineistot $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(B)}$, each containing m data points. We then use each of those tietoaineistot for model training (e.g., via ERM), resulting in the learned hypotheses $\widehat{h}^{(1)}, \ldots, \hat{h}^{(B)}$. We can use these learned hypotheses to estimate important characteristics of an koneoppiminen method such as harha, ??, or generalization gap [**?**].

See also: ??, ??, ??, histogrammi.

vakaus Robustness is a key requirement for ??. 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 piirteet of a data point in order to manipulate the ennuste delivered by a trained koneoppiminen model. Robustness also includes the stability of ERM-based methods against perturbations of the training set. Such perturbations can occur within data poisoning hyökkäykset.

See also: ??, stability, data poisoning, hyökkäys.

validation set A set of data points used to estimate the riski of a hypothesis $\hat{h}$ that has been learned by some koneoppiminen method (e.g., solving ERM). The average loss of $\hat{h}$ on the validointi set is referred to as the validointivirhe and can be used to diagnose an koneoppiminen method (see [**?**, 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, riski, hypothesis, koneoppiminen, ERM, loss, validointi, validointivirhe, opetusvirhe, hypothesis space, k-fold CV, LOO CV.

validointi Consider a hypothesis $\hat{h}$ that has been learned via some koneoppiminen method, e.g., by solving ERM on a training set $\mathcal{D}$. Validation refers to the process of evaluating the loss 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 loss of $\hat{h}$ on the validation set is referred to as the validointivirhe. An example of validation is shown in Fig. **??**.

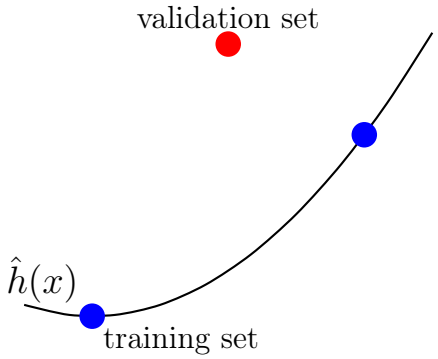


Fig. 73. Illustration of validation. The blue points represent the data points in the training set, while the red point represents a data point in the validation set. The hypothesis $\hat{h}$ (black curve) fits the data points in the training set perfectly, but incurs a large loss on the data point in the validation set.

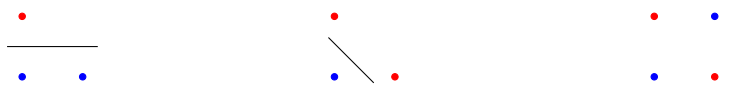
See also: training set, validation set, validointivirhe, ylisovittaminen, yleistys, k-fold CV, LOO CV.

validointivirhe Consider a hypothesis $\hat{h}$ that is obtained by some koneoppiminen method, e.g., using ERM on a training set. The average loss 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, loss, validation set, validointi.

Vapnik–Chervonenkis dimension (VC dimension) The statistical properties of an ERM-based method depend critically on the expressive capacity of its hypothesis space (or model) $\mathcal{H}$. A standard ?? of this capacity is the VC ?? $\mathrm{VCdim}\,(\mathcal{H})$ [**?**]. 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 $h \in \mathcal{H}$. Formally, this means that for every one of the 2^m possible assignments of binary nimiöt to each piirrevektori in $\mathcal{D}$, there exists some hypothesis $h \in \mathcal{H}$ that realizes this labeling. Intuitively, the VC ?? 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. **??** illustrates the definition of the VC ?? for a lineaarinen malli $\mathcal{H}^{(2)}$ with $d = 2$ piirteet. Fig. **??**(a) and **??**(b) show the same set of three noncollinear piirrevektorit 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 piirrevektorit, the set is shattered. Fig. **??**(c) depicts four piirrevektorit with a specific labeling. No linear separator can correctly classify all data points in this case. Thus, VCdim $\left(\mathcal{H}^{(2)}\right) = 3$.



(a) (b) (c)

Fig. 74. Illustration of the VC ?? for a lineaarinen malli $\mathcal{H}^{(2)}$ that is used to learn a lineaarinen luokitin in the feature space $\mathbb{R}^2$.

More generally, for a lineaarinen malli $\mathcal{H}^{(d)}$, the VC ?? equals $d + 1$. In other words, for lineaariset mallit, the VC ?? essentially matches the

?? of the underlying parameter space $\mathbb{R}^d$. For more complex hypothesis spaces, such as decision trees or ANNs, the relation between VC ?? and the ?? of the feature space is far less direct. In these cases, alternative complexity ??, such as the Rademacher complexity, can be more useful for analyzing ERM-based methods.

See also: hypothesis space, Rademacher complexity, yleistys, koneoppiminen, effective dimension.

vertailutaso Consider some koneoppiminen method that produces a learned hypothesis (or trained model) $\hat{h} \in \mathcal{H}$. We evaluate the quality of a trained model by computing the average loss 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 model 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 model.

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 ??. In many cases, given a ?? $p(\mathbf{x}, y)$, we can precisely determine the ?? achievable riski

among any hypotheses (not even required to belong to the hypothesis space $\mathcal{H}$) [**?**].

This ?? achievable riski (referred to as the Bayes risk) is the riski of the Bayes estimator for the nimiö y of a data point, given its piirteet $\mathbf{x}$. Note that, for a given choice of loss function, the Bayes estimator (if it exists) is completely determined by the ?? $\mathbb{P}$ [**?**, Ch. 4]. However, computing the Bayes estimator and Bayes risk presents two main challenges. First, the ?? $\mathbb{P}$ is unknown and must be estimated from observed data. Second, even if $\mathbb{P}$ were known, computing the Bayes risk exactly may be computationally infeasible [**?**]. A widely used ?? is the ?? $(\mathbf{x}, y) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for data points characterized by numeric piirteet and nimiöt. Here, for the neliövirhehäviö, the Bayes estimator is given by the ?? ?? $\mu_{y|\mathbf{x}}$ of the nimiö y, given the piirteet $\mathbf{x}$ [**?**], [**?**]. The corresponding Bayes risk is given by the ?? ?? $\sigma^2_{y|\mathbf{x}}$ (see Fig. **??**).

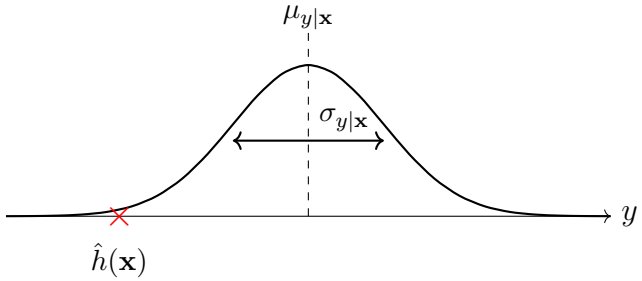


Fig. 75. If the piirteet and the nimiö of a data point are drawn from a ??, we can achieve the ?? riski (under neliövirhehäviö) by using the Bayes estimator $\mu_{y|\mathbf{x}}$ to predict the nimiö y of a data point with piirteet $\mathbf{x}$. The corresponding ?? riski is given by the ?? ?? $\sigma^2_{y|\mathbf{x}}$. We can use this quantity as a baseline for the average loss of a trained model $\hat{h}$.

See also: Bayes risk, Bayes estimator.

vertical federated learning (VFL) In VFL, different devices have access to different piirteet 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 piirrevektorit for the data points. Each device $i \in \mathcal{V}$ observes only a subset $\mathcal{F}^{(i)} \subseteq \{1, \ldots, d'\}$ of piirteet, resulting in a local dataset $\mathcal{D}^{(i)}$ with piirrevektorit

$$\mathbf{x}^{(i,r)} = \left(x_{j_1}^{(r)}, \ldots, x_{j_d}^{(r)}\right)^T.$$

Some of the devices may also have access to the nimiöt $y^{(r)}$, for $r = 1, \ldots, m$, of the global tietoaineisto (see Fig. **??**).

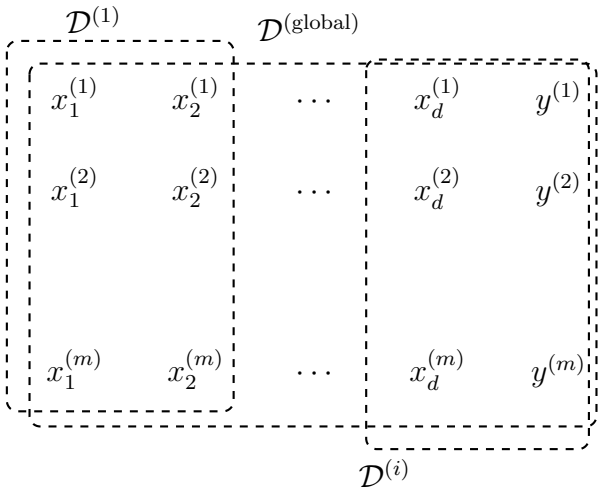


Fig. 76. VFL uses local datasets that are derived from the data points of a common global tietoaineisto. The local datasets differ in the choice of piirteet used to characterize the data points.

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 yksityisyyden suoja. We can view VFL as a specific form of model parallelism.

See also: yksityisyyden suoja, federoitu oppiminen, model parallelism.

weight Consider a parameterized hypothesis space $\mathcal{H}$. We use the term

weights for numeric model parameters that are used to scale piirteet or their transformations in order to compute $h^{(\mathbf{w})} \in \mathcal{H}$. A lineaarinen malli 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 ANNs to form linear combinations of piirteet or the outputs of neurons in hidden layers (see Fig. **??**).

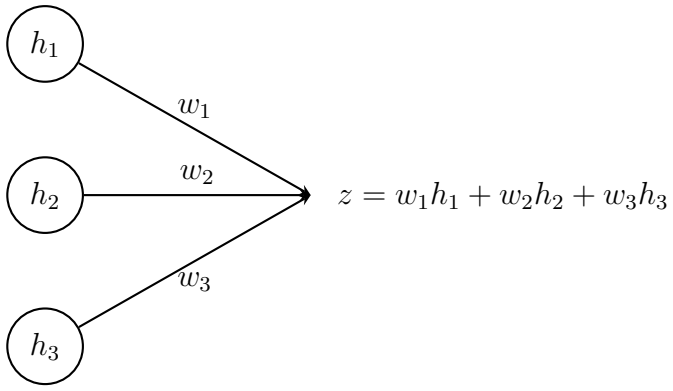


Fig. 77. A section of an ANN that contains a hidden layer with outputs (or activations) h_1,h_2, and h_3. These outputs are combined linearly to compute z, which can be used either as output of the ANN or as input to another layer.

See also: hypothesis space, model parameters, piirre, lineaarinen malli, ANN, layer, activation.

weighted least squares Weighted least squares refers to ERM-based methods that use the weighted average neliövirhehäviö

$$\frac{1}{m}\sum_{r=1}^{m} q^{(r)}\big(y^{(r)} - h(\mathbf{x}^{(r)})\big)^2$$

on a training set $\mathcal{D} = \left\{ \left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right) \right\}$ to measure the quality of a hypothesis ?? $h \in \mathcal{H}$. The weights $q^{(1)}, \ldots, q^{(m)} \in \mathbb{R}_+$ allow us to emphasize or de-emphasize the contribution of individual data points in the training set. Ideally, we assign a small weight $q^{(r)}$ to the rth data point if it is an outlier (see Fig. **??**). We obtain different weighted least squares methods by using different models in ERM.

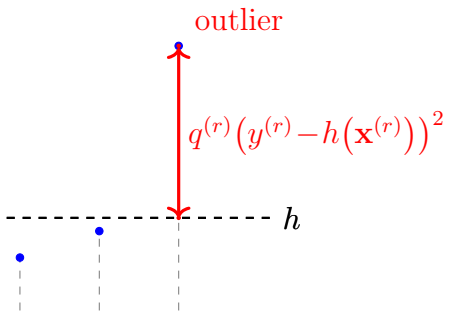


Fig. 78. Weighted least squares can be used to mitigate the effect of outlier data points in a training set.

See also: ERM, neliövirhehäviö, lineaarinen regressio, lineaarinen malli.

ydinfunktio Consider a set of data points, each represented by a piirrevektori $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ denotes the feature space. A (real-valued) kernel is a ?? $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that assigns to every pair of piirrevektorit $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ a real number $K\left(\mathbf{x}, \mathbf{x}'\right)$. This value is typically interpreted as a similarity ?? between $\mathbf{x}$ and $\mathbf{x}'$. The defining property of a kernel is that it is symmetric, i.e., $K\left(\mathbf{x}, \mathbf{x}'\right) = K\left(\mathbf{x}', \mathbf{x}\right)$, and that for any finite set of

piirrevektorit $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$, the ??

$$\mathbf{K} = \begin{pmatrix} K\big(\mathbf{x}_1, \mathbf{x}_1\big) & K\big(\mathbf{x}_1, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_1, \mathbf{x}_n\big) \\ K\big(\mathbf{x}_2, \mathbf{x}_1\big) & K\big(\mathbf{x}_2, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_2, \mathbf{x}_n\big) \\ \vdots & \vdots & \ddots & \vdots \\ K\big(\mathbf{x}_n, \mathbf{x}_1\big) & K\big(\mathbf{x}_n, \mathbf{x}_2\big) & \ldots & K\big(\mathbf{x}_n, \mathbf{x}_n\big) \end{pmatrix} \in \mathbb{R}^{n \times n}$$

is ??. A kernel naturally defines a transformation of a piirrevektori $\mathbf{x}$ into a ?? $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$. The ?? $\mathbf{z}$ maps an input $\mathbf{x}' \in \mathcal{X}$ to the value $K\big(\mathbf{x}, \mathbf{x}'\big)$. We can view the ?? $\mathbf{z}$ as a new piirrevektori 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 ?? (RKHS) [**?**], [**?**]. Since $\mathbf{z}$ belongs to a RKHS, which is a ??, we can interpret it as a generalized piirrevektori. Note that a finite-length piirrevektori $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T \in \mathbb{R}^d$ can be viewed as a ?? $\mathbf{x} : \{1, \ldots, d\} \rightarrow \mathbb{R}$ that assigns a real value to each index $j \in \{1, \ldots, d\}$.

See also: piirrevektori, feature space, ??, kernel method.

yksityisyyden suoja 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 piirteet $\mathbf{x}$. Many important koneoppiminen applications involve data points representing humans. Each data point is characterized by piirteet $\mathbf{x}$, 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 ?? $\mathcal{A}(\mathcal{D})$. 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 parameter, ennuste, data point, piirre, nimiö, sensitive attribute, ??.

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

See also: hyökkäys, sensitive attribute, model inversion, ??, ??.

yleistys Generalization refers to the ability of a model trained on a training set to make accurate ennusteet 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 loss 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 ?? for data generation, such as the i.i.d. assumption. Here, we interpret data points as independent ?? with an identical ?? $p(\mathbf{z})$. This ??, which is assumed fixed but unknown, allows us to define the riski of a trained model $\hat{h}$ as

the expected loss

$$\bar{L}\big(\hat{h}\big) = \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\big\{L(\hat{h},\mathbf{z})\big\}.$$

The difference between riski $\bar{L}\big(\hat{h}\big)$ and empiirinen riski $\widehat{L}\big(\hat{h}|\mathcal{D}^{(\text{train})}\big)$ is known as the generalization gap. Tools from todennäköisyys theory, such as ?? and uniform ??, allow us to bound this gap under certain conditions [**?**].

Generalization without todennäköisyys: Todennäköisyys theory is one way to study how well a model 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 model $\hat{h}$ should be robust, i.e., its ennuste $\hat{h}(\mathbf{x})$ should not change much if we slightly change the piirteet $\mathbf{x}$ 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. **??** for a visual illustration.

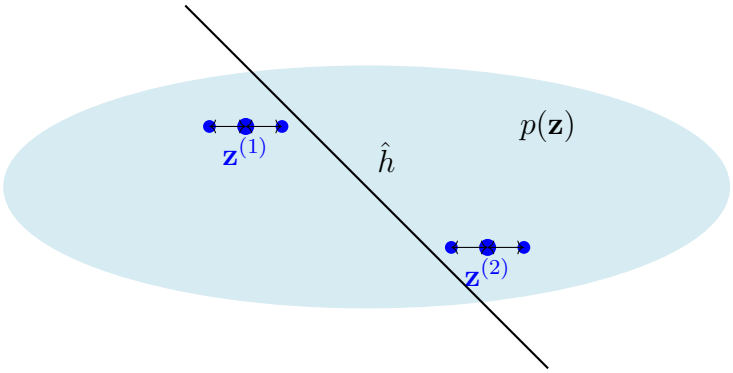


Fig. 79. Two data points $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}$ that are used as a training set to learn a hypothesis $\hat{h}$ via ERM. We can evaluate $\hat{h}$ outside $\mathcal{D}^{(\text{train})}$ either by an i.i.d. assumption with some underlying ?? $p(\mathbf{z})$ or by perturbing the data points.

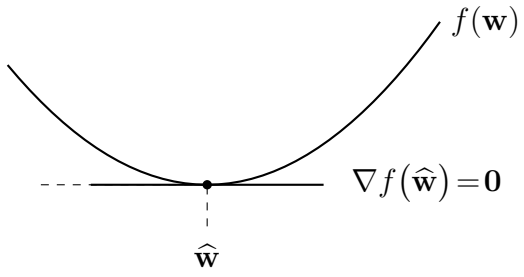
See also: ERM, i.i.d. assumption, ylisovittaminen, validointi.

ylisovittaminen Consider an koneoppiminen method that uses ERM to learn a hypothesis with the ?? empiirinen riski on a given training set. Such a method is overfitting the training set if it learns a hypothesis with a empiirinen riski on the training set that a significantly smaller than the average loss outside the training set. In other words, if a koneoppiminen method overfits it has a large generalization gap.

See also: ERM, yleistys, validointi, generalization gap.

zero-gradient condition Consider the unconstrained ?? $\min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w})$ with a ?? and ?? kohdefunktio $f(\mathbf{w})$. A necessary and sufficient condition for a ?? $\widehat{\mathbf{w}} \in \mathbb{R}^d$ to solve this problem is that the ?? $\nabla f\big(\widehat{\mathbf{w}}\big)$ is the zero ?? $\nabla f\big(\widehat{\mathbf{w}}\big) = \mathbf{0}$. In other words [**?**, p. 140],

$$\nabla f\big(\widehat{\mathbf{w}}\big) = \mathbf{0} \Leftrightarrow f\big(\widehat{\mathbf{w}}\big) = \min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w}).$$



By defining the ?? ?? $\nabla f : \mathbb{R}^d \to \mathbb{R}^d$, we can rewrite the zero-gradient condition as a fixed-point equation

$$\big(\mathcal{I} - \alpha \nabla f\big)\widehat{\mathbf{w}} = \widehat{\mathbf{w}}.$$

Here, $\mathcal{I}$ denotes the identity ?? (i.e., $\mathcal{I}(\mathbf{w}) = \mathbf{w}$) and α is an arbitrary positive number.

See also: ??, ??, ??, kohdefunktio, ??, ??.

Hakemisto