Title: Sample complexity

URL: https://en.wikipedia.org/wiki/Sample_complexity

PageID: 43269516

Categories: Category:Machine learning

Supervised learning

Unsupervised learning

Semi-supervised learning

Self-supervised learning

Reinforcement learning

Meta-learning

Online learning

Batch learning

Curriculum learning

Rule-based learning

Neuro-symbolic AI

Neuromorphic engineering

Quantum machine learning

Classification

Generative modeling

Regression

Clustering

Dimensionality reduction

Density estimation

Anomaly detection

Data cleaning

AutoML

Association rules

Semantic analysis

Structured prediction

Feature engineering

Feature learning

Learning to rank

Grammar induction

Ontology learning

Multimodal learning

Apprenticeship learning

Decision trees

Ensembles Bagging Boosting Random forest

Bagging

Boosting

Random forest

k -NN

Linear regression

Naive Bayes

Artificial neural networks

Logistic regression

Perceptron

Relevance vector machine (RVM)

Support vector machine (SVM)

BIRCH

CURE

Hierarchical

k -means

Fuzzy

Expectation–maximization (EM)

DBSCAN

OPTICS

Mean shift

Factor analysis

CCA

ICA

LDA

NMF

PCA

PGD

t-SNE

SDL

Graphical models Bayes net Conditional random field Hidden Markov

Bayes net

Conditional random field

Hidden Markov

RANSAC

k -NN

Local outlier factor

Isolation forest

Autoencoder

Deep learning

Feedforward neural network

Recurrent neural network LSTM GRU ESN reservoir computing

LSTM

GRU

ESN

reservoir computing

Boltzmann machine Restricted

Restricted

GAN

Diffusion model

SOM

Convolutional neural network U-Net LeNet AlexNet DeepDream

U-Net

LeNet

AlexNet

DeepDream

Neural field Neural radiance field Physics-informed neural networks

Neural radiance field

Physics-informed neural networks

Transformer Vision

Vision

Mamba

Spiking neural network

Memtransistor

Electrochemical RAM (ECRAM)

Q-learning

Policy gradient

SARSA

Temporal difference (TD)

Multi-agent Self-play

Self-play

Active learning

Crowdsourcing

Human-in-the-loop

Mechanistic interpretability

RLHF

Coefficient of determination

Confusion matrix

Learning curve

ROC curve

Kernel machines

Bias–variance tradeoff

Computational learning theory

Empirical risk minimization

Occam learning

PAC learning

Statistical learning

VC theory

Topological deep learning

AAAI

ECML PKDD

NeurIPS

ICML

ICLR

IJCAI

ML

JMLR

Glossary of artificial intelligence

List of datasets for machine-learning research List of datasets in computer vision and image processing

List of datasets in computer vision and image processing

Outline of machine learning

v

t

e

The sample complexity of a machine learning algorithm represents the number of training-samples that it needs in order to successfully learn a target function.

More precisely, the sample complexity is the number of training-samples that we need to supply to the algorithm, so that the function returned by the algorithm is within an arbitrarily small error of the best possible function, with probability arbitrarily close to 1.

There are two variants of sample complexity:

The weak variant fixes a particular input-output distribution;

The strong variant takes the worst-case sample complexity over all input-output distributions.

The No free lunch theorem , discussed below, proves that, in general, the strong sample complexity is infinite, i.e. that there is no algorithm that can learn the globally-optimal target function using a finite number of training samples.

However, if we are only interested in a particular class of target functions (e.g., only linear functions) then the sample complexity is finite, and it depends linearly on the VC dimension on the class of target functions. [1]

Definition

Let X {\displaystyle X} be a space which we call the input space, and Y {\displaystyle Y} be a space which we call the output space, and let Z {\displaystyle Z} denote the product X × Y {\displaystyle X\times Y} . For example, in the setting of binary classification, X {\displaystyle X} is typically a finite-dimensional vector space and Y {\displaystyle Y} is the set { − 1 , 1 } {\displaystyle \{-1,1\}} .

Fix a hypothesis space H {\displaystyle {\mathcal {H}}} of functions h : X → Y {\displaystyle h\colon X\to Y} . A learning algorithm over H {\displaystyle {\mathcal {H}}} is a computable map from Z {\displaystyle Z} to H {\displaystyle {\mathcal {H}}} . In other words, it is an algorithm that takes as input a finite sequence of training samples and outputs a function from X {\displaystyle X} to Y {\displaystyle Y} . Typical learning algorithms include empirical risk minimization , without or with Tikhonov regularization .

Fix a loss function L : Y × Y → R ≥ 0 {\displaystyle {\mathcal {L}}\colon Y\times Y\to \mathbb {R} _{\geq 0}} , for example, the square loss L (y , y ′) = (y − y ′) 2 {\displaystyle {\mathcal {L}}(y,y')=(y-y')^{2}} , where h (x) = y ′ {\displaystyle h(x)=y'} . For a given distribution ρ {\displaystyle \rho } on X × Y {\displaystyle X\times Y} , the expected risk of a hypothesis (a function) h ∈ H {\displaystyle h\in {\mathcal {H}}} is

In our setting, we have h = A (S n) {\displaystyle h={\mathcal {A}}(S_{n})} , where A {\displaystyle {\mathcal {A}}} is a learning algorithm and S n = ((x 1 , y 1) , … , (x n , y n)) ∼ ρ n {\displaystyle S_{n}=((x_{1},y_{1}),\ldots ,(x_{n},y_{n}))\sim \rho ^{n}} is a sequence of vectors which are all drawn independently from ρ {\displaystyle \rho } . Define the optimal risk E H ∗ = inf h ∈ H E (h) . {\displaystyle {\mathcal {E}}_{\mathcal {H}}^{*}={\underset {h\in {\mathcal {H}}}{\inf }}{\mathcal {E}}(h).} Set h n = A (S n) {\displaystyle h_{n}={\mathcal {A}}(S_{n})} , for each sample size n {\displaystyle n} . h n {\displaystyle h_{n}} is a random variable and depends on the random variable S n {\displaystyle S_{n}} , which is drawn from the distribution ρ n {\displaystyle \rho ^{n}} . The algorithm A {\displaystyle {\mathcal {A}}} is called consistent if E (h n) {\displaystyle {\mathcal {E}}(h_{n})} probabilistically converges to E H ∗ {\displaystyle {\mathcal {E}}_{\mathcal {H}}^{*}} . In other words, for all ■ , δ > 0 {\displaystyle \epsilon ,\delta >0} , there exists a positive integer N {\displaystyle N} , such that, for all sample sizes n ≥ N {\displaystyle n\geq N} , we have

Pr ρ n [E (h n) − E H ∗ ≥ ε] < δ . {\displaystyle \Pr _{\rho ^{n}}[{\mathcal {E}}(h_{n})-{\mathcal {E}}_{\mathcal {H}}^{*}\geq \varepsilon]<\delta .} The sample complexity of A {\displaystyle {\mathcal {A}}} is then the minimum N {\displaystyle N} for which this holds, as a function of ρ , ■ {\displaystyle \rho ,\epsilon } , and δ {\displaystyle \delta } . We write the sample complexity as N (ρ , ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} to emphasize that this value of N {\displaystyle N} depends on ρ , ■ {\displaystyle \rho ,\epsilon } , and δ {\displaystyle \delta } . If A {\displaystyle {\mathcal {A}}} is not consistent , then we set N (ρ , ■ , δ) = ∞ {\displaystyle N(\rho ,\epsilon ,\delta)=\infty } . If there exists an algorithm for which N (ρ , ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} is finite, then we say that the hypothesis space H {\displaystyle {\mathcal {H}}} is learnable .

In others words, the sample complexity N (ρ , ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} defines the rate of consistency of the algorithm: given a desired accuracy ■ {\displaystyle \epsilon } and confidence δ {\displaystyle \delta } , one needs to sample N (ρ , ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} data points to guarantee that the risk of the output function is within ■ {\displaystyle \epsilon } of the best possible, with probability at least 1 − δ {\displaystyle 1-\delta } . [2]

In probably approximately correct (PAC) learning , one is concerned with whether the sample complexity is polynomial , that is, whether N (ρ , ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} is bounded by a polynomial in 1 / ■ {\displaystyle 1/\epsilon } and 1 / δ {\displaystyle 1/\delta } . If N (ρ

, ■ , δ) {\displaystyle N(\rho ,\epsilon ,\delta)} is polynomial for some learning algorithm, then one says that the hypothesis space H {\displaystyle {\mathcal {H}}} is PAC-learnable . This is a stronger notion than being learnable.

Unrestricted hypothesis space: infinite sample complexity

One can ask whether there exists a learning algorithm so that the sample complexity is finite in the strong sense, that is, there is a bound on the number of samples needed so that the algorithm can learn any distribution over the input-output space with a specified target error. More formally, one asks whether there exists a learning algorithm A {\displaystyle {\mathcal {A}}} , such that, for all ■ , δ > 0 {\displaystyle \epsilon ,\delta >0} , there exists a positive integer N {\displaystyle N} such that for all n ≥ N {\displaystyle n\geq N} , we have

sup ρ (Pr ρ n [E (h n) − E H ∗ ≥ ε]) < δ , {\displaystyle \sup _{\rho }\left(\Pr _{\rho ^{n}}[{\mathcal {E}}(h_{n})-{\mathcal {E}}_{\mathcal {H}}^{*}\geq \varepsilon]\right)<\delta ,} where h n = A (S n) {\displaystyle h_{n}={\mathcal {A}}(S_{n})} , with S n = ((x 1 , y 1) , … , (x n , y n)) ∼ ρ n {\displaystyle S_{n}=((x_{1},y_{1}),\ldots ,(x_{n},y_{n}))\sim \rho ^{n}} as above. The No Free Lunch Theorem says that without restrictions on the hypothesis space H {\displaystyle {\mathcal {H}}} , this is not the case, i.e., there always exist "bad" distributions for which the sample complexity is arbitrarily large. [1]

Thus, in order to make statements about the rate of convergence of the quantity sup ρ (Pr ρ n [E (h n) − E H ∗ ≥ ε]) , {\displaystyle \sup _{\rho }\left(\Pr _{\rho ^{n}}[{\mathcal {E}}(h_{n})-{\mathcal {E}}_{\mathcal {H}}^{*}\geq \varepsilon]\right),} one must either

constrain the space of probability distributions ρ {\displaystyle \rho } , e.g. via a parametric approach, or

constrain the space of hypotheses H {\displaystyle {\mathcal {H}}} , as in distribution-free approaches.

Restricted hypothesis space: finite sample-complexity

The latter approach leads to concepts such as VC dimension and Rademacher complexity which control the complexity of the space H {\displaystyle {\mathcal {H}}} . A smaller hypothesis space introduces more bias into the inference process, meaning that E H ∗ {\displaystyle {\mathcal {E}}_{\mathcal {H}}^{*}} may be greater than the best possible risk in a larger space. However, by restricting the complexity of the hypothesis space it becomes possible for an algorithm to produce more uniformly consistent functions. This trade-off leads to the concept of regularization . [2]

It is a theorem from VC theory that the following three statements are equivalent for a hypothesis space H {\displaystyle {\mathcal {H}}} :

H {\displaystyle {\mathcal {H}}} is PAC-learnable.

The VC dimension of H {\displaystyle {\mathcal {H}}} is finite.

H {\displaystyle {\mathcal {H}}} is a uniform Glivenko-Cantelli class .

This gives a way to prove that certain hypothesis spaces are PAC learnable, and by extension, learnable.

An example of a PAC-learnable hypothesis space

X = R d , Y = { − 1 , 1 } {\displaystyle X=\mathbb {R} ^{d},Y=\{-1,1\}} , and let H {\displaystyle {\mathcal {H}}} be the space of affine functions on X {\displaystyle X} , that is, functions of the form x ■ ■ w , x ■ + b {\displaystyle x\mapsto \langle w,x\rangle +b} for some w ∈ R d , b ∈ R {\displaystyle w\in \mathbb {R} ^{d},b\in \mathbb {R} } . This is the linear classification with offset learning problem. Now, four coplanar points in a square cannot be shattered by any affine function, since no affine function can be positive on two diagonally opposite vertices and negative on the remaining two. Thus, the VC dimension of H {\displaystyle {\mathcal {H}}} is d + 1 {\displaystyle d+1} , so it is finite. It follows by the above characterization of PAC-learnable classes that H {\displaystyle {\mathcal {H}}} is PAC-learnable, and by extension, learnable.

Sample-complexity bounds

Suppose H {\displaystyle {\mathcal {H}}} is a class of binary functions (functions to { 0 , 1 } {\displaystyle \{0,1\}}). Then, H {\displaystyle {\mathcal {H}}} is (■ , δ) {\displaystyle (\epsilon ,\delta)} -PAC-learnable with a sample of size: [3] N = O (V C (H) + ln ■ 1 δ ■) {\displaystyle N=O{\bigg (}{\frac {VC({\mathcal {H}})+\ln {1 \over \delta }}{\epsilon }}{\bigg)}} where V C (H) {\displaystyle VC({\mathcal {H}})} is the VC dimension of H {\displaystyle {\mathcal {H}}} .

Moreover, any (■ , δ) {\displaystyle (\epsilon ,\delta)} -PAC-learning algorithm for H {\displaystyle {\mathcal {H}}} must have sample-complexity: [4] N = Ω (V C (H) + ln ■ 1 δ ■) {\displaystyle N=\Omega {\bigg (}{\frac {VC({\mathcal {H}})+\ln {1 \over \delta }}{\epsilon }}{\bigg)}} Thus, the sample-complexity is a linear function of the VC dimension of the hypothesis space.

Suppose H {\displaystyle {\mathcal {H}}} is a class of real-valued functions with range in [0 , T] {\displaystyle [0,T]} . Then, H {\displaystyle {\mathcal {H}}} is (■ , δ) {\displaystyle (\epsilon ,\delta)} -PAC-learnable with a sample of size: [5] [6] N = O (T 2 P D (H) ln ■ T ■ + ln ■ 1 δ ■ 2) {\displaystyle N=O{\bigg (}T^{2}{\frac {PD({\mathcal {H}})\ln {T \over \epsilon }+\ln {1 \over \delta }}{\epsilon ^{2}}}{\bigg)}} where P D (H) {\displaystyle PD({\mathcal {H}})} is Pollard's pseudo-dimension of H {\displaystyle {\mathcal {H}}} .

Other settings

In addition to the supervised learning setting, sample complexity is relevant to semi-supervised learning problems including active learning , [7] where the algorithm can ask for labels to specifically chosen inputs in order to reduce the cost of obtaining many labels. The concept of sample complexity also shows up in reinforcement learning , [8] online learning , and unsupervised algorithms, e.g. for dictionary learning . [9]

Efficiency in robotics

A high sample complexity means that many calculations are needed for running a Monte Carlo tree search . [10] It is equivalent to a model-free brute force search in the state space. In contrast, a high-efficiency algorithm has a low sample complexity. [11] Possible techniques for reducing the sample complexity are metric learning [12] and model-based reinforcement learning. [13]

See also

Active learning (machine learning)

References