

Machine learning theory

Nonuniform learnability

Hamid Beigy

Sharif university of technology

March 13, 2023





1. Introduction
2. Nonuniform learnability
3. Structural risk minimization
4. Homeworks
5. Minimum description length
6. Occam's Razor
7. Decision trees
8. Consistency
9. Nearest neighbor
10. Summary
11. Reading

Introduction



1. Let H be a hypothesis space on a domain \mathcal{X} , where \mathcal{X} is given an arbitrary probability distribution \mathcal{D} .
2. The notions of PAC learnability allow the sample sizes to depend on the accuracy and confidence parameters, but they are uniform with respect to the labeling rule and the underlying data distribution.
3. So far, learner expresses prior knowledge by specifying the hypothesis class H .
4. Consequently, classes that are learnable in that respect are **limited**, (they must have a **finite VC-dimension**).
5. There are too many hypotheses classes that have **infinite VC-dimension**. What can we talk about their learnability?
6. In this section, we consider more relaxed, **weaker notions of learnability** (nonuniform learnability).
7. **Nonuniform learnability** allows the sample size to depend on the hypothesis to which the learner is compared.
8. It can be shown that **nonuniform learnability** is a strict relaxation of agnostic PAC learnability.



1. A hypothesis h is (ϵ, δ) -competitive with another hypothesis h' if, with probability higher than $(1 - \delta)$,

$$\mathbf{R}(h) \leq \mathbf{R}(h') + \epsilon.$$

2. In agnostic PAC learning, the number of required examples depends only on ϵ and δ .

Definition (Agnostic PAC learnability)

A hypothesis class H is agnostically PAC learnable if there exist a learning algorithm, A , and a function $m_H : (0, 1)^2 \mapsto \mathbb{N}$ such that, for every $\epsilon, \delta \in (0, 1)$ and every distribution \mathcal{D} , if $m \geq m_H(\epsilon, \delta)$, then with probability of at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$ it holds that

$$\mathbf{R}(A(S)) \leq \min_{h' \in H} \mathbf{R}(h') + \epsilon.$$

Note that this implies that for every $h \in H$

$$\mathbf{R}(A(S)) \leq \mathbf{R}(h) + \epsilon.$$

3. This definition shows that the sample complexity is independent of specific h .
4. A hypothesis class H is agnostically PAC learnable if it has finite VC-dimension.

Nonuniform learnability



1. In nonuniform learnability, we allow the sample size to be of the form $m_H(\epsilon, \delta, h)$; namely, it depends also on the h with which we are competing.

Definition (Nonuniformly learnability)

A hypothesis class H is nonuniformly learnable if there exist a learning algorithm, A , and a function $m_H^{NUL} : (0, 1)^2 \times H \mapsto \mathbb{N}$ such that, for every $\epsilon, \delta \in (0, 1)$ and every distribution \mathcal{D} , if $m \geq m_H^{NUL}(\epsilon, \delta, h)$, then with probability of at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$ it holds that

$$\mathbf{R}(A(S)) \leq \mathbf{R}(h) + \epsilon.$$

2. In both types of learnability, we require that the output hypothesis will be (ϵ, δ) -competitive with every other hypothesis in the class.
3. The difference between these two notions of learnability is the question of whether the sample size m may depend on the hypothesis h to which the error of $A(S)$ is compared.



1. The nonuniform learnability is a **relaxation of agnostic PAC learnability**.
2. That is, **if a class is agnostic PAC learnable then it is also nonuniformly learnable**.
3. There is also a **second relaxation**, where the sample complexity is allowed to depend even on **the probability distribution \mathcal{D}** . This is called **consistency**, but it turns out to be too weak to be useful.

Theorem

Let H be a hypothesis class that can be written as a countable union of hypothesis classes, $H = \bigcup_{n \in \mathbb{N}} H_n$, where each H_n enjoys the uniform convergence property. Then, H is nonuniformly learnable.

Proof.

This theorem can be proved by introducing a new learning paradigm. □



Theorem (nonuniform learnability)

A hypothesis class H of binary classifiers is *nonuniformly learnable* if and only if it is a *countable union of agnostic PAC learnable hypothesis classes*.

Proof.

\implies Assume that $H = \bigcup_{n \in \mathbb{N}} H_n$, where each H_n is PAC learnable. Using the [fundamental theorem of statistical learning](#), then each H_n has the uniform convergence property. Therefore, using the above Theorem, we obtain that H is nonuniform learnable.

\Leftarrow Assume that H is nonuniform learnable using some algorithm A . For every $n \in \mathbb{N}$, let

$$H_n = \left\{ h \in H \mid m_H^{NUL} \left(\frac{1}{8}, \frac{1}{7}, h \right) \leq n \right\}.$$

1. Clearly, $H = \bigcup_{n \in \mathbb{N}} H_n$.
2. Using the definition of m_H^{NUL} , we know that for any distribution \mathcal{D} that satisfies the realizability assumption with respect to H_n , with probability of at least $\frac{6}{7}$ over $S \sim \mathcal{D}^n$ we have that $\mathbf{R}(A(S)) \leq \frac{1}{8}$.
3. Using the fundamental theorem of statistical learning, this implies that the VC-dimension of H_n must be finite, and therefore H_n is agnostic PAC learnable.

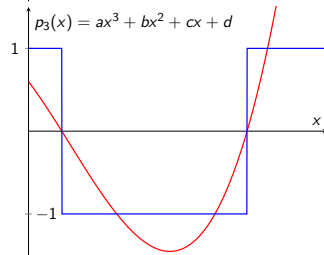
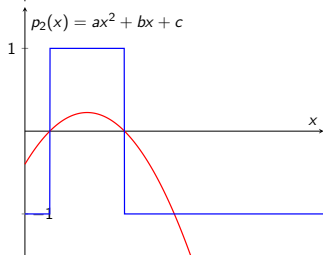
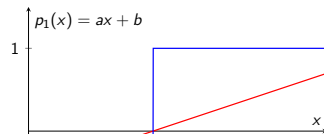
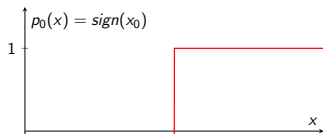
□



1. The following example shows that nonuniform learnability is a strict relaxation of agnostic PAC learnability; namely,
there are hypothesis classes that are nonuniform learnable but are not agnostic PAC learnable.

Example

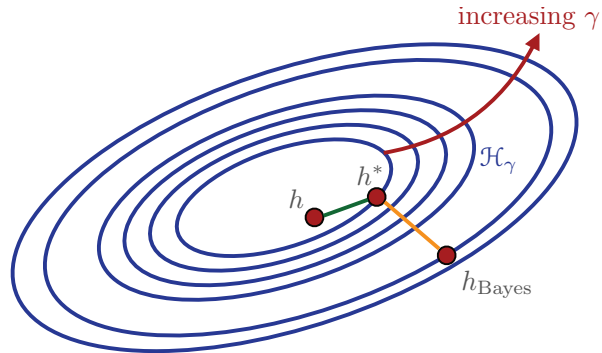
- Consider a binary classification problem with $\mathcal{X} = \mathbb{R}$.
- For every $n \in \mathbb{N}$ let H_n be the class of polynomial classifiers of degree n .
- H_n is the set of all classifiers of form $h(x) = \text{sign}(p_n(x))$ where $p_n : \mathbb{R} \mapsto \mathbb{R}$ is a polynomial of degree n .
- Let $H = \bigcup_{n \in \mathbb{N}} H_n$, then H is the class of all polynomial classifiers over \mathbb{R} .
- It is easy to verify that $VC(H) = \infty$, while $VC(H_n) = n + 1$.
- Hence, H is not PAC learnable, while on the basis of the above Theorem, H is nonuniformly learnable.



Structural risk minimization



1. Suppose we can decompose H as a union of increasingly $\bigcup_{\gamma \in \Gamma} H_\gamma$ increasing with γ for some set Γ .



2. The problem then consists of selecting the parameter $\gamma^* \in \Gamma$ and thus the hypothesis set H_{γ^*} with the most favorable **trade-off between estimation and approximation errors**.
3. For SRM, H is assumed to be decomposable into a countable set, thus, we write it as $H = \bigcup_{k \geq 1} H_k$.
4. Also, the hypothesis sets are nested, i.e. $H_k \subset H_{k+1}$ for all $k \geq 1$.
5. SRM consists of choosing the index $k^* \geq 1$ and the ERM hypothesis $h \in H_{k^*}$ that **minimize an upper bound on the excess error**.



1. The hypothesis set for SRM: $H = \bigcup_{k \geq 1} H_k$ with $H_1 \subset H_2 \subset \dots \subset H_k \subset \dots$
2. We suppose that we are given a family H_n of hypothesis classes, each of which being PAC learnable, but how do we select n ?
3. So far, we have encoded our prior knowledge by specifying a hypothesis class H , which we believe includes a good predictor for the learning task at hand.
4. Yet another way to express our prior knowledge is by specifying preferences over hypotheses within H .
5. In the Structural Risk Minimization (SRM) paradigm, we do so by
 - 5.1 first assuming that H can be written as $H = \bigcup_{n \in \mathbb{N}} H_n$ and
 - 5.2 then specifying a weight function, $w : \mathbb{N} \mapsto [0, 1]$, which assigns a weight to each hypothesis class, H_n , such that a higher weight reflects a stronger preference for the hypothesis class.
6. We will discuss how to learn with such prior knowledge.



1. Let H be a hypothesis class that can be written as $H = \bigcup_{n \in \mathbb{N}} H_n$.
2. It tries to find a hypothesis that

$$h_m^{SRM} = \arg \min_{h \in H_n, n \in \mathbb{N}} \hat{\mathbf{R}}(h) + \text{Complexity}(H_n, m)$$

3. Let also for each n , the class H_n enjoys the uniform convergence property with a sample complexity function $m_{H_n}^{UC}(\epsilon, \delta)$.
4. We suppose that we are given a family H_n of hypothesis classes, each of which being PAC learnable, but how do we select n ?
5. Let us also define the function $\epsilon_n : \mathbb{N} \times (0, 1) \mapsto (0, 1)$ by

$$\epsilon_n(m, \delta) = \min \left\{ \epsilon \mid m_{H_n}^{UC}(\epsilon, \delta) \leq m \right\}$$

6. In words, we have a fixed training size m , and we are interested in the lowest possible upper bound on the gap between empirical and true risks achievable by using a sample of m examples.
7. From the definitions of uniform convergence and ϵ_n , it follows that for every m and δ , with probability of at least δ over the choice of $S \sim \mathcal{D}^m$, for all $h \in H_n$ we have that

$$|\mathbf{R}(h) - \hat{\mathbf{R}}(h)| \leq \epsilon_n(m, \delta)$$



1. Let $w : \mathbb{N} \mapsto [0, 1]$ be **weight function** over the hypothesis classes H_1, H_2, \dots such that $\sum_{n=1}^{\infty} w(n) \leq 1$.
2. Such a weight function can be **the priori preference** or some **measure of the complexity** of different hypothesis classes.
3. When $H = H_1 \cup H_2 \cup \dots \cup H_N$ and $w(n) = \frac{1}{N}$, this corresponds to **no a priori preference to any hypothesis class**.
4. When H is a **(countable) infinite union of hypothesis classes**, a uniform weighting is not possible but we need other weighting such as $w(n) = \frac{6}{(\pi n)^2}$ or $w(n) = 2^{-n}$.
5. The SRM rule follows a **bound minimization** approach.
6. This means that the goal of the paradigm is to find a hypothesis that minimizes a certain upper bound on the true risk.



The bound that the SRM rule wishes to minimize is given in the following theorem.

Theorem

Let $w : \mathbb{N} \mapsto [0, 1]$ be a function such that $\sum_{n=1}^{\infty} w(n) \leq 1$. Let H be a hypothesis class that can be written as $H = \bigcup_{n \in \mathbb{N}} H_n$, where for each n , H_n satisfies the uniform convergence property with a sample complexity function $m_{H_n}^{UC}$. Let $\epsilon_n(m, \delta) = \min\{\epsilon \mid m_H^{UC}(\epsilon, \delta) \leq m\}$. Then, for every $\delta \in (0, 1)$ and distribution \mathcal{D} , with probability of at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$, the following bound holds (simultaneously) for every $n \in \mathbb{N}$ and $h \in H_n$.

$$|\mathbf{R}(h) - \hat{\mathbf{R}}(h)| \leq \epsilon_n(m, w(n) \times \delta)$$

Therefore, for every $\delta \in (0, 1)$ and distribution \mathcal{D} , with probability of at least $1 - \delta$ for all $h \in H$, it holds that

$$\mathbf{R}(h) \leq \hat{\mathbf{R}}(h) + \min_{h \in H_n, n \in \mathbb{N}} \epsilon_n(m, w(n) \times \delta)$$

**Proof.**

1. For each n define $\delta_n = w(n)\delta$.
2. Applying the assumption that uniform convergence holds for all n with the rate of $|\mathbf{R}(h) - \hat{\mathbf{R}}(h)| \leq \epsilon_n(m, \delta_n)$.
3. We obtain that if we fix n in advance, then with probability of at least $1 - \delta_n$ over the choice of $S \sim \mathcal{D}^m$, for all $h \in H_n$, we have

$$|\mathbf{R}(h) - \hat{\mathbf{R}}(h)| \leq \epsilon_n(m, \delta_n)$$

4. Applying the union bound over $n = 1, 2, \dots$, we obtain that with probability of at least

$$\begin{aligned} 1 - \sum_n \delta_n &= 1 - \delta \sum_n w(n) \\ &\geq 1 - \delta \end{aligned}$$

5. The preceding holds for all n .





1. Let $n(h) = \min\{n \mid h \in H_n\}$. Then the above Theorem implies that SRM searches for h that minimizes

$$\mathbf{R}(h) \leq \hat{\mathbf{R}}(h) + \epsilon_{n(h)}(m, w(n(h)) \times \delta)$$

2. The following Theorem shows that the SRM paradigm can be used for nonuniform learning of every class, which is a countable union of uniformly converging hypothesis classes. [The proof is given in page 62 of Book](#) (Shalev-Shwartz and Ben-David 2014).

Theorem

Let H be a hypothesis class such that $H = \bigcup_{n \in \mathbb{N}} H_n$, where each H_n has the uniform convergence property with sample complexity $m_{H_n}^{UC}$. Let $w : \mathbb{N} \mapsto [0, 1]$ be such that $w(n) = \frac{6}{\pi^2 n^2}$. Then, H is nonuniformly learnable using the SRM rule with rate

$$m_H^{NUL}(\epsilon, \delta, h) \leq m_{H_{n(h)}}^{UC} \left(\frac{\epsilon}{2}, \frac{6\delta}{(\pi n(h))^2} \right)$$

3. This theorem also proves the nonuniform learnability.



1. We have shown that **any countable union of classes of finite VC-dimension** is **nonuniformly learnable**.
2. It turns out that, for any infinite domain set, \mathcal{X} , the class of all binary valued functions over \mathcal{X} is **not a countable union of classes of finite VC-dimension**.
3. It follows that, in some sense, the **NFL theorem** holds for nonuniform learning as well:

NFL for nonuniform learning

When the domain is not finite, there exists no nonuniform learner with respect to the class of all deterministic binary classifiers.

4. Although for each such classifier there exists a trivial algorithm that learns it (ERM with respect to the hypothesis class that contains only this classifier).



1. The **prior knowledge** of a nonuniform learner for H is **weaker**, it is searching for a model throughout the entire class H , rather than being focused on one specific H_n .
2. The cost of this **weakening of prior knowledge** is the **increase in sample complexity** needed to compete with any specific $h \in H_n$.
3. Consider the task of binary classification with the zero-one loss and assume that for all n , we have $VC(H_n) = n$.
4. For H_n , we have $m_{H_n}^{UC}(\epsilon, \delta) = C \frac{n + \log(1/\delta)}{\epsilon^2}$, where C is a constant.
5. By using weight function $w(n) = \frac{1}{2n^2}$, we have

$$m_H^{NUL}(\epsilon, \delta, h) - m_{H_n}^{UC}(\epsilon/2, w(n)\delta) = O\left(\frac{\log n}{\epsilon^2}\right)$$

6. That is, the cost of relaxing the learner's prior knowledge from a specific H_n that contains the target h to a countable union of classes depends on the log of the index of the first class in which h resides.
7. That cost increases with the index of the class, which can be interpreted as reflecting the value of knowing a good priority order on the hypotheses in H .

Homeworks



Please send these homeworks via email. The deadline is **1402/01/20**.

1. Let $H_n = \left\{ h \in H \mid m_H^{NUL} \left(\frac{1}{8}, \frac{1}{7}, h \right) \leq n \right\}$, show that $VC(H_n)$ is finite.
2. Prove Theorem 7.2 of Book (Shalev-Shwartz and Ben-David 2014).
3. Is $\sin(\theta x)$ nonuniformly learnable?
4. What are differences between definitions of **uniform convergence property** and **agnostic PAC learnability**?
5. Let $H_n = \{ \mathbb{R} \mapsto \{0, 1\} \mid f^{-1}(\cdot) \text{ is finite} \}$. Is H agnostic PAC learnable? Is H nonuniformly learnable?

Minimum description length



1. Let H be a countable hypothesis class. Then, we can write H as a countable union of singleton classes, namely, $H = \bigcup_{n \in \mathbb{N}} \{h_n\}$
2. By Hoeffding's inequality, each singleton class has the uniform convergence property with rate $m^{UC}(\epsilon, \delta) = \frac{\log(2/\delta)}{2\epsilon^2}$.

3. Therefore, function ϵ_n becomes $\epsilon_n(m, \delta) = \sqrt{\frac{\log(2/\delta)}{2m}}$ and SRM rule becomes

$$\arg \min_{h_n \in H} \left[\hat{\mathbf{R}}(h_n) + \sqrt{\frac{\log(1/w(n)) + \log(2/\delta)}{2m}} \right]$$

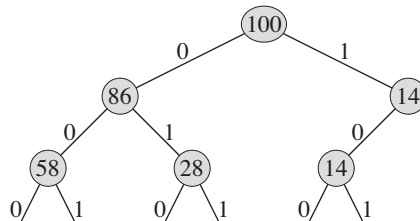
4. Equivalently, we can think of the weight function as $w : H \mapsto [0, 1]$, and then the SRM rule becomes

$$\arg \min_{h \in H} \left[\hat{\mathbf{R}}(h) + \sqrt{\frac{-\log w(h) + \log(2/\delta)}{2m}} \right]$$

5. This means, the prior knowledge is solely determined by the weight we assign to each hypothesis.
6. We assign higher weights to hypotheses that we believe are more likely to be the correct one, and in the learning algorithm we prefer hypotheses that have higher weights.



1. Having a hypothesis class, one can wonder about how represent each hypothesis in the class.
2. We naturally fix some description language such as **English**, **a programming language**, or **some set of mathematical formulas**.
3. Let H be the hypothesis class we wish to describe. Fix some finite set Σ of symbols, which we call the alphabet.
4. Let $\Sigma = \{0, 1\}$. A string is a finite sequence of symbols from Σ : for example, $\sigma = (0, 1, 1, 1, 0)$ is a string of length 5.
5. We denote by $|\sigma|$ the length of a string. The set of all finite length strings is denoted Σ^* .
6. A description language for H is a function $d : H \mapsto \Sigma^*$, mapping each member $h \in H$ to a string $d(h)$ (the description of h and its length is denoted by $|h|$).
7. We require that **description languages be prefix-free**; namely, for every distinct h and h' , $d(h)$ is not a prefix of $d(h')$.





1. Prefix-free collections of strings enjoy the following combinatorial property:

Lemma (Kraft Inequality)

If $S \subseteq \{0, 1\}^*$ is a prefix-free set of strings, then

$$\sum_{\sigma \in S} \frac{1}{2^{|\sigma|}} \leq 1.$$

Proof.

- Define a probability distribution over the members of S as follows:
- Repeatedly toss an unbiased coin, with faces labeled 0 and 1, until the sequence of outcomes is a member of S , at that point, stop.
- For each $\sigma \in S$, let $P(\sigma)$ be the probability that this process generates the string σ .
- Note that since S is prefix-free, for every $\sigma \in S$, if the coin toss outcomes follow the bits of σ then we will stop only once the sequence of outcomes equals σ .
- We therefore get that, for every $\sigma \in S$, $P(\sigma) = \frac{1}{2^{|\sigma|}}$.
- Since probabilities add up to at most 1, our proof is concluded.

□



From Kraft's inequality, any prefix-free description language of a hypothesis class, H , gives rise to a weighting function w over that hypothesis class. We will simply set $w(h) = \frac{1}{2^{|h|}}$.

Theorem

Let H be a hypothesis class and let $d : H \mapsto \Sigma^*$ be a prefix-free description language for H . Then, for every sample size, m , every confidence parameter, $\delta > 0$, and every probability distribution, \mathcal{D} , with probability greater than $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$, we have that,

$$\mathbf{R}(h) \leq \hat{\mathbf{R}}(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}}$$

where $|h|$ is the length of $d(h)$.

Proof.

1. Choose $w(h) = \frac{1}{2^{|h|}}$.
2. Note that $\ln(2^{|h|}) = |h| \ln(2) < |h|$.
3. Apply Theorem of SRM bound, with $\epsilon_n(m, \delta) = \sqrt{\frac{\log(2/\delta)}{2m}}$

□



1. This Theorem result suggests a learning paradigm for H given a training set, S , search for a hypothesis $h \in H$ that minimizes the bound $\mathbf{R}(h) \leq \hat{\mathbf{R}}(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}}$.
2. This suggests trading off empirical risk for saving description length.
3. This yields the **Minimum Description Length learning** paradigm as

$$\arg \min_{h \in H} \left[\hat{\mathbf{R}}(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}} \right]$$

Occam's Razor



1. The last Theorem suggests that, having two hypotheses sharing the same empirical risk, **the true risk of the one that has shorter description can be bounded by a lower value.**
2. Thus, this result can be viewed as conveying a philosophical message.

Occam's Razor

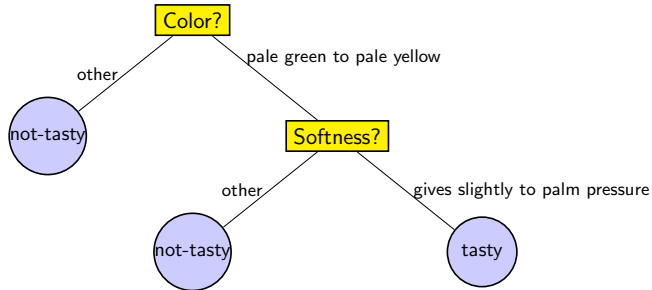
A short explanation (that is, a hypothesis that has a short length) tends to be more valid than a long explanation.

3. This Theorem shows that **the more complex a hypothesis h is**, the larger the sample size it has to fit to guarantee that it has a small true risk, $\mathbf{R}(h)$.
4. How do we choose the **description language?** (**after/before seeing data?**)
5. From the Hoeffding's bound, if we commit to any hypothesis before seeing the data, then we are guaranteed a rather small estimation error term $\mathbf{R}(h) \leq \hat{\mathbf{R}}(h) + \sqrt{\frac{|h| + \ln(2/\delta)}{2m}}$.
6. As long as **description language is chosen independently of the training sample**, the generalization bound holds.

Decision trees



A decision tree is a predictor, $h : \mathcal{X} \mapsto \mathcal{Y}$, that predicts the label associated with an instance x by traveling from a root node of a tree to a leaf.



Lemma (VC dimension of decision trees)

Let H be the class of decision trees over \mathcal{X} with k leaves. Then, $VC(H) = k$.

Proof.

1. A set of k instances that arrive to the different leaves can be shattered.
2. A set of $(k + 1)$ instances can't be shattered since 2 instances must arrive to the same leaf.

□

**Lemma (VC dimension of decision trees)**

Let $\mathcal{X} = \{0, 1\}^d$ and splitting rules are according to $\mathbb{I}[x_i = 1]$ for some $i \in \{1, \dots, d\}$. Let also H be the class of decision trees over \mathcal{X} . Then, $VC(H) = 2^d$.

Proof.

1. The height of this tree is at most $d + 1$.
2. A full binary tree with height of $d + 1$ has 2^d leaves.
3. Hence, a set of 2^d instances that arrive to the different leaves can be shattered.
4. Then, $VC(H) = 2^d$.

□

Check: Class H contains $\{0, 1\}^{\mathcal{X}}$ and hence $VC(H) = |\mathcal{X}| = 2^d$.



1. A tree with n nodes can be described as $(n + 1)$ blocks, each of size $\log_2(d + 3)$ bits, indicating (in depth-first order).
 - An internal node of the form $\mathbb{I}[x_i = 1]$ for some $i \in \{1, \dots, d\}$
 - A leaf whose value is 1
 - A leaf whose value is 0
 - End of the code
2. Let internal nodes have two children, this is a prefix-free encoding of the tree.
3. The description length of a tree with n nodes is $(n + 1) \log_2(d + 3)$.
4. Applying MDL learning rule to **search tree with n nodes that minimizes**

$$\hat{R}(h) + \sqrt{\frac{(n + 1) \log_2(d + 3) + \log(2/\delta)}{2m}}$$

5. This belongs to the class of **NP hard problems**.
6. Hence, practical decision tree learning algorithms are based on **heuristics**.
7. For example, **Iterative Dichotomizer 3** (ID3), proposed by J. Ross Quinlan, is a **greedy algorithm** (Quinlan 1986).
8. ID3 follows the **MDL principle**, attempts to create **a small tree with low train error**.

Consistency



1. The notion of learnability can be further relaxed by allowing the needed sample sizes to depend not only on ϵ , δ , and h but also on the underlying data-generating probability distribution \mathcal{D} .
2. This type of performance guarantee is captured by the notion of **consistency** of a learning rule.

Definition (Consistency)

Let \mathcal{Z} be a domain set, let \mathcal{P} be a set of probability distributions over \mathcal{Z} , and let H be a hypothesis class. A learning rule A is **consistent** with respect to H and \mathcal{P} if there exists a function $m_H^{CON} : (0, 1)^2 \times H \times \mathcal{P} \mapsto \mathbb{N}$ such that, for every $\epsilon, \delta \in (0, 1)$, every $h \in H$, and every $\mathcal{D} \in \mathcal{P}$, if $m \geq m_H^{CON}(\epsilon, \delta, h, \mathcal{D})$ then H with probability of at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$ it holds that

$$\mathbf{R}(A(S)) \leq \mathbf{R}(h) + \epsilon.$$

If \mathcal{P} is the set of all distributions, we say that A is **universally consistent with respect to H** .

3. The notion of **consistency is a relaxation** of the previous notion of **nonuniform learnability**.
4. If an **algorithm nonuniformly learns a class H** , it is also **universally consistent** for that class.
5. The **relaxation is strict** in the sense that there are consistent learning rules that are not successful nonuniform learners.



1. The following algorithm is a universally consistent for the class of all binary classifiers over \mathbb{N} this class is not nonuniformly learnable.

Example (Memorize algorithm)

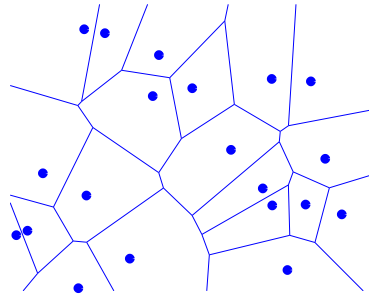
An algorithm memorizes the training examples, and, given a test point x , it predicts the majority label among all labeled instances of x that exist in the training sample and some fixed default label if no instance of x appears in the training set. Show that this algorithm is universally consistent for every countable domain \mathcal{X} and a finite label set \mathcal{Y} w.r.t. the zero-one loss.

2. Intuitively, it is not obvious that this algorithm should be viewed as a learner, since it lacks the aspect of generalization, namely, of using observed data to predict the labels of unseen examples.
3. The fact that this algorithm is a consistent algorithm for the class of all functions over any countable domain set therefore raises doubt about the usefulness of consistency guarantees.
4. May this algorithm overfit?
5. For more information regarding consistency, please read chapters 6 and 11 of (Devroye, Györfi, and Lugosi 1996).

Nearest neighbor



1. Memorize the training set $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$.
2. Given new x , find the k closest points in S and return **majority vote** among their labels.



3. Unlike ERM,SRM,MDL, there is no H .
4. At training time: **do nothing**.
5. At test time: **search S for the nearest neighbors**.



1. Let

- $\mathcal{X} = [0, 1]^n$
- $\mathcal{Y} = \{0, 1\}$
- \mathcal{D} is a distribution over $\mathcal{X} \times \mathcal{Y}$
- $\mathcal{D}_{\mathcal{X}}$ is the marginal distribution over \mathcal{X}
- $\eta : \mathbb{R}^n \mapsto \mathbb{R}$ is the conditional probability over the labels, i.e. $\eta(\mathbf{x}) = \mathbb{P}[y = 1 \mid \mathbf{x}]$

2. Bayes optimal rule

$$h^*(\mathbf{x}) = \mathbb{I} \left[\eta(\mathbf{x}) \geq \frac{1}{2} \right]$$

3. **Prior knowledge:** η is c -Lipschitz. That is, for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, we have

$$|\eta(\mathbf{x}) - \eta(\mathbf{x}')| \leq c \|\mathbf{x} - \mathbf{x}'\|$$

Theorem

Let h be the k -NN rule, then

$$\mathbb{E}_{S \sim \mathcal{D}^m} [\mathbf{R}(h)] \leq \left(1 + \sqrt{\frac{8}{k}} \right) \mathbf{R}(h^*) + (6c\sqrt{n} + k) m^{-\frac{1}{n+1}}$$

4. It seems that, **we learn the class of all functions over $[0, 1]^n$.**
5. But **this class is not learnable even in the non-uniform model.**
6. There's no contradiction: **the number of required examples depends on the Lipschitzness of η (the parameter c), which depends on \mathcal{D} .**
7. Hence, **it is learnable in consistency model.**

Summary



1. The first possible goal of deriving performance guarantees on a learning algorithm is bounding the risk of the output predictor.
2. Both PAC learning and nonuniform learning **give us an upper bound on the true risk of the learned hypothesis** based on its empirical risk.
3. **Consistency guarantees do not provide such a bound.**
4. However, it is always possible **to estimate the risk of the output predictor** using a **validation set**.



1. How many examples we need to collect in order to learn it.
 - PAC learning gives a crisp answer.
 - For NUL and consistency, we do not know the number of examples required for learning H .
 - In NUL this number depends on the best hypothesis in H .
 - In consistency this number also depends on the underlying distribution.
 - In this sense, PAC learning is the only useful definition of learnability.
2. If $R_{est}(h)$ is small, its risk may still be large if H has a large $R_{app}(h)$.
3. How many examples are required to be as good as the Bayes optimal predictor?
 - PAC guarantees do not provide us with a crisp answer.
 - This reflects the fact that the usefulness of PAC learning relies on the quality of our prior knowledge.



1. PAC guarantees also help us to understand what we should do next if our learning algorithm returns a hypothesis **with a large risk**.
2. We can bound $\mathbf{R}_{est}(h)$ and therefore know how much of the error is due to $\mathbf{R}_{app}(h)$.
3. If $\mathbf{R}_{app}(h)$ is large, we know that we should use a different hypothesis class.
4. If a NUL algorithm fails, we can consider a different weighting function.
5. When a consistent algorithm fails, we don't know the reason is $\mathbf{R}_{est}(h)$ or $\mathbf{R}_{app}(h)$.
6. If we are sure we have a problem with $\mathbf{R}_{est}(h)$, we do not know how many more examples are needed to make $\mathbf{R}_{est}(h)$ small.

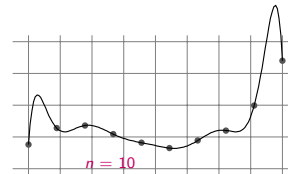
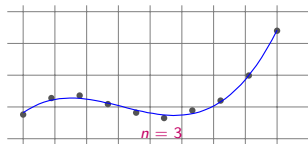
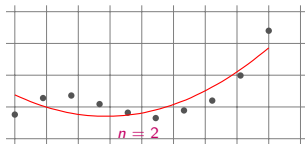


1. The most useful aspect of learning theory is providing an answer to the question of [how to learn?](#).
 - PAC learning yields the limitation of learning (via NFL theorem) and the necessity of prior knowledge.
 - PAC learning gives us a crisp way to encode prior knowledge by choosing a hypothesis class, and after that, we have a generic learning rule (ERM).
 - NUL learning also yields a crisp way to encode prior knowledge by specifying weights over (subsets of) hypotheses of H and after that we have a generic learning rule (SRM).
2. Unlike the notions of PAC learnability and nonuniform learnability, the definition of consistency does not yield a natural learning paradigm or a way to encode prior knowledge.
3. In fact, in many cases there is no need for prior knowledge at all.
4. As an example, we saw that even the [Memorize algorithm](#), which intuitively should not be called a learning algorithm, is a consistent algorithm for any class defined over a countable domain and a finite label set.
5. This hints that consistency is a very weak requirement.



1. The SRM rule is also advantageous in model selection when prior knowledge is partial.

- Consider the regression problem of learning a function, $h : \mathbb{R} \mapsto \mathbb{R}$.
- As prior knowledge we consider the hypothesis class of polynomials.
- We are uncertain regarding which degree n would give the best results for our data set.
- A small degree might not fit the data well (large $\mathbf{R}_{app}(h)$).
- A high degree might lead to overfitting (large $\mathbf{R}_{est}(h)$).



2. It is easy to see that the empirical risk decreases as we enlarge the degree.

3. if we choose $H = \{p_n(x) \mid 0 \leq n \leq 10\}$, then the ERM rule with respect to this class would output a $p_{10}(x)$ and would overfit.

4. If we choose $H = \{p_n(x) \mid 0 \leq n \leq 2\}$, then the ERM would underfit (large $\mathbf{R}_{app}(h)$).

5. We can use SRM rule on $H = \{p_n(x) \mid n \in \mathbb{N}\}$ and ordering subsets of H according to n . This yields a $h(x) = p_3(x)$ since combination of its $\hat{\mathbf{R}}(h)$ and bound on its $\mathbf{R}_{app}(h)$ is the smallest.

6. The SRM rule enables us to select the right model on the basis of the data itself.

7. The price we pay for this flexibility is that we do not know in advance the number of examples to compete with the best hypothesis in H .



1. One may argue that even though consistency is a weak requirement, it is desirable that a learning algorithm will be consistent with respect to the set of all functions from \mathcal{X} to \mathcal{Y} .
2. This gives us a guarantee that for enough training examples, we will always be as good as the Bayes optimal predictor.
3. Therefore, if we have two algorithms, where one is consistent and the other one is not consistent, we should prefer the consistent algorithm.
4. This argument is problematic for two reasons.
 - First, maybe it is the case that for most natural distributions we will observe in practice that the sample complexity of the consistent algorithm will be so large so that in every practical situation we will not obtain enough examples to enjoy this guarantee.
 - Second, it is not very hard to make any PAC or nonuniform learner consistent with respect to the class of all functions from \mathcal{X} to \mathcal{Y} .



1. Consider a countable domain, \mathcal{X} , a finite label set \mathcal{Y} , and a hypothesis class, H .
2. We can make any NUL learner for H be consistent with respect to the class of all classifiers from \mathcal{X} to \mathcal{Y} using the following simple trick.
3. Upon receiving a training set S :
 - First run the NUL learner over S , and then obtain a bound on the true risk of the learned predictor. If this bound is small enough we are done.
 - Otherwise, we revert to the [Memorize algorithm](#).
4. This simple modification [makes the algorithm consistent](#) with respect to all functions from \mathcal{X} to \mathcal{Y} .
5. Since it is easy to make any algorithm consistent, [it may not be wise to prefer one algorithm over the other just because of consistency considerations](#).



1. NFL implies that no algorithm can learn the class of all classifiers over an infinite domain.
2. However, we saw that the **Memorize algorithm is consistent** with respect to the class of all classifiers over a countable infinite domain.
3. Why these two statements do not contradict each other?

Theorem (NFL theorem)

Let \mathcal{X} be a countable infinite domain and let $\mathcal{Y} = \{-1, +1\}$. For any algorithm, A , and a training set size, m , there exist a distribution \mathcal{D} over X and a function $h^* : \mathcal{X} \mapsto \mathcal{Y}$, such that if A will get a sample $S \sim \mathcal{D}^m$, labeled by h^* , then A is likely to return a classifier with a larger error.

4. The consistency of Memorize implies the following:

Consistency of Memorize

For every distribution over \mathcal{X} and a labeling function $h^* : \mathcal{X} \mapsto \mathcal{Y}$, there exists a training set size $m(\mathcal{D}, h^*)$ such that if Memorize receives at least m examples it is likely to return a classifier with a small error.

5. In NFL, we first fix m , and then find a \mathcal{D} and a h^* that are bad for this training set size.
6. In consistency guarantees, we first fix \mathcal{D} and h^* , and then we find a m that suffices for learning this particular \mathcal{D} and h^* .



1. The classes of **infinite VC-dimension** can be learnable, in some **weaker sense of learnability**.
2. For countable hypothesis classes, we can apply the MDL scheme, where hypotheses with shorter descriptions are preferred.
3. We can implement the class of all predictors in C++, which is a powerful class of functions and probably contains all that we can hope to learn in practice.
4. Even the implementation of the ERM paradigm with respect to all C++ programs of description length at most 1000 bits requires an exhaustive search over 2^{1000} hypotheses.
5. While the sample complexity of learning this class is just $\frac{1000 + \log(2/\delta)}{\epsilon^2}$, the runtime is $\geq 2^{1000}$, which is **too computationally hard**.
6. The notions of learnability can be summarized as




	Bounds on R based on \hat{R}	Bounds on $R(A(s))$ $\inf_{h \in H} R(h)$ based on m	compared to	Encode prior knowl- edge
(Agn) PAC	✓	✓ (in advance)		✓ (specifying H)
Nonuniform	✓	✓ (depends on the best $h \in H$)		✓ (weights)
Consistent	×	×		×

Reading



1. Chapters 7 & 18 & 19 of [Understanding machine learning : From theory to algorithms](#) (Shalev-Shwartz and Ben-David 2014).



-  Devroye, Luc, Laszlo Györfi, and Gabor Lugosi (1996). *A probabilistic theory of pattern recognition*. Springer.
-  Quinlan, J. Ross (1986). "Induction of Decision Trees". In: *Machine Learning* 1.1, pp. 81–106.
-  Shalev-Shwartz, Shai and Shai Ben-David (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Questions?