CPSC 532D — 9. NONUNIFORM LEARNING

Danica J. Sutherland
University of British Columbia, Vancouver
Fall 2024

Recall the decomposition of error we made back in Section 1.4:

$$\underbrace{\mathrm{L}_{\mathcal{D}}(\hat{h}_{\mathrm{S}}) - \mathrm{L}_{bayes}}_{\text{excess error}} = \underbrace{\mathrm{L}_{\mathcal{D}}(\hat{h}_{\mathrm{S}}) - \inf_{h \in \mathcal{H}} \mathrm{L}_{\mathcal{D}}(h)}_{\text{estimation error}} + \underbrace{\inf_{h \in \mathcal{H}} \mathrm{L}_{\mathcal{D}}(h) - \mathrm{L}_{bayes}}_{\text{approximation error}}.$$

We've talked a lot about the estimation error of ERM, bounding it in terms of Rademacher complexity or (when applicable) VC dimension. What we haven't really talked about yet is the approximation error. We drew some examples with polynomials in Figure 1.1, but if we don't know what the optimal predictor looks like... what should we do?

There are some particular cases where we can analyze this approximation error gap mathematically, if we assume things about the form of $\mathcal{D}$. But those assumptions usually rely on constants that are hard to know for any specific problem, and there's not usually a clear way to estimate them (or the Bayes error) from data, either.

The practical solution is generally to just try a bunch of different $\mathcal{H}$ and/or a bunch of different learning algorithms, then pick the best based on a validation set V. This is a good idea in practice, and we can make some theoretical guarantees on its generalization based on L_{V} being close to $\mathrm{L}_{\mathcal{D}}$. But it's still hard to use that approach to say anything with confidence about the approximation error.

9.1 STRUCTURAL RISK MINIMIZATION

SRM says: let's use a *huge* $\mathcal{H}$, one where the approximation error is going to be small, maybe even *zero* if $\mathcal{H}$ is what's called *universal* (coming up soon!). This will probably mean $\mathcal{H}$ has infinite VC dimension, large Rademacher complexity, etc. But let's decompose

$$\mathcal{H} = \mathcal{H}_1 \cup \mathcal{H}_2 \cup \cdots = \bigcup_{k \in \mathbb{N}} \mathcal{H}_k.$$

For instance, we might have $\mathcal{H}_k$ the set of decision trees of depth k, the set of degree-k polynomials, or the set of linear classifiers with $\|w\| \le 2^k$. We're going to assume that *each* $\mathcal{H}_k$ has uniform convergence:

$$\forall k \in \mathbb{N}. \quad \Pr_{\mathrm{S} \sim \mathcal{D}^m}\left(\sup_{h \in \mathcal{H}_k} \mathrm{L}_{\mathcal{D}}(h) - \mathrm{L}_{\mathrm{S}}(h) \le \varepsilon_k(m, \delta)\right) \ge 1 - \delta \tag{9.1}$$

for functions ε_k satisfying that for all k and all $\delta \in (0, 1)$, $\lim_{m \to \infty} \varepsilon_k(m, \delta) = 0$.

We'll also need a set of weights $w_k \ge 0$ such that $\sum_{k=1}^{\infty} w_k \le 1$; a typical choice is $6/(\pi^2 k^2) \approx 0.61/k^2$, since $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$.

This is the problem that made Euler famous.

For more, visit https://cs.ubc.ca/~dsuth/532D/24w1/.

Proposition 9.1. *Let $\mathcal{H} = \mathcal{H}_1 \cup \mathcal{H}_2 \cup \dots$ satisfy (9.1), and let $w_k \geq 0$ have $\sum_{k=1}^{\infty} w_k \leq 1$. Then for any $\mathcal{D}$, with probability at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$, we have*

$$\forall h \in \mathcal{H}. \quad L_{\mathcal{D}}(h) \leq L_S(h) + \min_{k: h \in \mathcal{H}_k} \varepsilon_k(m, \delta w_k).$$

Proof. We do a union bound over the $\mathcal{H}_k$, allocating δw_1 probability that anything in $\mathcal{H}_1$ violates the bound, δw_2 that anything in $\mathcal{H}_2$ does, and so on. Thus the total probability anything in $\mathcal{H}$ violates it is at most $\sum_k \delta w_k \leq \delta$. □

SRM is then the algorithm that minimizes this upper bound on $L_{\mathcal{D}}(h)$:

Definition 9.2. Given bounds on a decomposition of $\mathcal{H}$ as in (9.1), and weights $w_k \geq 0$ with $\sum w_k \leq 1$ and $\bigcup_{k: w_k > 0} \mathcal{H}_k = \mathcal{H}$, *structural risk minimization* is given by

$$\mathrm{SRM}_{\mathcal{H},\delta}(S) \in \arg\min_{h \in \mathcal{H}} \left[L_S(h) + \varepsilon_{k_h}(m, \delta w_{k_h}) \right] \qquad \text{where } k_h \in \arg\min_{k: h \in \mathcal{H}_k} \varepsilon_k(m, w_k \delta).$$

Typically, $k_h = \min\{k : h \in \mathcal{H}_k\}$.

We can implement this minimization by a finite number of calls to an "ERM oracle", as long as our loss is lower-bounded by $a \leq \ell(h, z)$, e.g. $a = 0$:

function $\mathrm{SRM}_{\mathcal{H},\delta}(S)$
  `best` $\leftarrow \infty$
  for $k = 1, 2, \dots$ **do**
    $h_k \leftarrow \mathrm{ERM}_{\mathcal{H}_k}(S)$
    `cand_loss` $\leftarrow L_S(h_k) + \varepsilon_k(m, w_k \delta)$
    if `cand` < `best` **then**
      $\hat{h} \leftarrow h_k$
      `best` $\leftarrow$ `cand`
    if $\min_{k' > k} a + \varepsilon_{k'}(m, w_{k'} \delta) >$ `best` **then**
      break
  return $\hat{h}$

Note that if we "decompose" as $\mathcal{H}_1 = \mathcal{H}$, then SRM becomes just $\mathrm{ERM}_{\mathcal{H}}$.

Theorem 9.3. *Let $h^* \in \mathcal{H}$ be any fixed hypothesis in the setup of Definition 9.2, and let $a \leq \ell(h, z) \leq b$ for all $h \in \mathcal{H}$, $z \in \mathcal{Z}$. Then, with probability at least $1 - \delta - \delta'$ over the choice of random samples $S \sim \mathcal{D}^m$, SRM satisfies*

$$L_{\mathcal{D}}(\mathrm{SRM}_{\mathcal{H},\delta}(S)) \leq L_{\mathcal{D}}(h^*) + \varepsilon_{k_{h^*}}\left(m, w_{k_{h^*}} \delta\right) + (b - a)\sqrt{\tfrac{1}{2m} \log \tfrac{1}{\delta'}}.$$

Proof. Let $\hat{h}_S = \mathrm{SRM}_{\mathcal{H}}(S)$. We have that

$$\begin{aligned} L_{\mathcal{D}}(\hat{h}_S) &\leq L_S(\hat{h}_S) + \varepsilon_{k_{\hat{h}_S}}(m, w_{k_{\hat{h}_S}} \delta) && \text{by Proposition 9.1, prob} \geq 1 - \delta \\ &\leq L_S(h^*) + \varepsilon_{k_{h^*}}(m, w_{k_{h^*}} \delta) && \text{by def of SRM;} \end{aligned}$$

the conclusion follows by applying Hoeffding's inequality with probability δ' to upper-bound $L_S(h^*)$. □

Compare this to ERM that just knows in advance which $\mathcal{H}_{k_{h^*}}$ to pick; with probability

at least $1 - 2\delta$, that would have performance

$$L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}}(S)) \le L_{\mathcal{D}}(h^*) + \varepsilon_{k_{h^*}}(m, \delta) + (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{\delta}}.$$

How much worse this is depends on how much worse $\varepsilon_{k_{h^*}}(m, w_{k_{h^*}}\delta)$ is than $\varepsilon_{k_{h^*}}(m, \delta)$.

9.1.1 *With Rademacher bounds*

Since this is a little abstract, let's see what happens if we plug in the Rademacher bound of Theorem 5.7: let $\mathcal{R}_{k,m} = \mathbb{E}_{S\sim\mathcal{D}^m}\mathrm{Rad}((\ell\circ\mathcal{H}_k)|_S)$, assume $a \le \ell(h,z) \le b$, and for simplicity assume that $\mathcal{R}_{k+1,m} \ge \mathcal{R}_{k,m}$ for all k. Then

$$\varepsilon_k(m,\delta) = 2\mathcal{R}_{k,m} + (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{\delta}}.$$

Let's also plug in $w_k = 6/(\pi^2 k^2)$. Then Proposition 9.1 becomes that

$$\Pr\left(\forall h\in\mathcal{H}.\quad L_{\mathcal{D}}(h) \le L_S(h) + 2\mathcal{R}_{k_h,m} + (b-a)\sqrt{\frac{1}{2m}\log\frac{\pi^2 k_h^2}{6\delta}}\right) \ge 1-\delta, \tag{9.2}$$

where $k_h = \min\{k : h\in\mathcal{H}_k\}$. Using this bound to define an SRM algorithm gives

$$\mathrm{SRM}_{\mathcal{H},\delta}(S) \in \arg\min_{h\in\mathcal{H}}\left[L_S(h) + 2\mathcal{R}_{k_h,m} + (b-a)\sqrt{\frac{1}{2m}\log\frac{\pi^2 k_h^2}{6\delta}}\right]. \tag{9.3}$$

Theorem 9.3 gives that with probability at least $1 - (1+\frac{6}{\pi^2})\delta$,

$$\begin{aligned} L_{\mathcal{D}}(\mathrm{SRM}_{\mathcal{H},\delta}(S)) &\le L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_h + \frac{1}{2}\log\frac{\pi^2}{6\delta}} + \sqrt{\frac{1}{2}\log\frac{\pi^2}{6\delta}}\right] \\ &\le L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_h} + \sqrt{2\log\frac{\pi^2}{6\delta}}\right]. \end{aligned}$$

Letting $\delta' = \frac{\pi^2+6}{\pi^2}\delta$ so that $\frac{\pi^2}{6\delta} = \frac{\pi^2}{6}\frac{\pi^2}{\pi^2+6}\frac{1}{\delta} < \frac{1.03}{\delta}$, this means that with probability at least $1-\delta'$ we have

$$L_{\mathcal{D}}\left(\mathrm{SRM}_{\mathcal{H},\frac{\pi^2}{\pi^2+6}\delta'}(S)\right) \le L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_{h^*}} + \sqrt{2\log\frac{1.03}{\delta'}}\right]. \tag{9.4}$$

Compare to ERM with $\mathcal{H}_{k_{h^*}}$: with probability at least $1-\delta'$,

$$L_{\mathcal{D}}(\mathrm{ERM}_{\mathcal{H}_{k_{h^*}}}(S)) \le L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\sqrt{2\log\frac{2}{\delta'}}.$$

So, as long as we have a reasonable number of samples compared to the complexity of h^* – that is, $m \gg \log k_{h^*}$ – we pay essentially no penalty for not knowing the correct $\mathcal{H}_k$ in advance!

9.1.2 *Problems with bound minimization*

Concentration inequalities are usually pretty conservative, since they hold for *all* distributions subject to some mild constraints (e.g. sub-Gaussianity). Symmetrization is also often a bit loose; it introduces a factor of 2 that might not be needed, e.g. in

equation (11) / Appendix E.4 of [Zho+22] we established that this 2 can (basically) be a 1 for Gaussian-data ℓ_1-loss regression.

So, if we minimize a potentially loose bound, then we might get bad results: because our bound is too conservative, we'll have too much bias towards a simple solution. (If the problem turns out to be realizable, but we didn't assume that from the outset, then we can't adapt to the fast $1/m$ rate; we'll operate assuming the slow $1/\sqrt{m}$ rate.) Fundamentally, this means the performance of our algorithm is based on how good at theoretical analysis we are; we'd usually rather have an algorithm that works well whether we're smart or not.

It's also kind of weird for us to have to pre-commit to a certain failure probability δ; that's not usually how we think about things. That in particular, though, we'll be able to avoid.

9.1.3 *Aside: Avoiding the δ dependence*

It's pretty annoying that the algorithm depends on a specific choice of δ; that "feels like" an analysis parameter, not an algorithm one. We can do this by defining a slight variant of the algorithm; notice that (9.2) implies

$$\Pr\left(\forall h \in \mathcal{H}.\ \ L_{\mathcal{D}}(h) \le L_S(h) + 2\mathcal{R}_{k_h,m} + (b-a)\sqrt{\frac{1}{m}\log k_h} + (b-a)\sqrt{\frac{1}{2m}\log\frac{\pi^2}{6\delta}}\right) \ge 1-\delta,$$

since we only made the upper bound looser with $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ for nonnegative a, b. But when minimizing *this* upper bound, the $(b-a)\sqrt{\frac{1}{2m}\log\frac{\pi^2}{6\delta}}$ term doesn't depend on h at all, and so we can just ignore it;

$$\mathrm{SRM}_{\mathcal{H}}(S) \in \arg\min_{h\in\mathcal{H}}\left[L_S(h) + 2\mathcal{R}_{k_h,m} + (b-a)\sqrt{\frac{1}{m}\log k_h}\right].$$

A slight variant of Theorem 9.3 still applies; we just have to use the ε_k that splits the two square root terms up, giving for this variant that with probability at least $1-\delta$,

w/ prob at least $1-\frac{\pi^2}{6+\pi^2}\delta$
$$L_{\mathcal{D}}(\mathrm{SRM}_{\mathcal{H}}(S)) \le L_S(\hat{h}_S) + 2\mathcal{R}_{k_{\hat{h}_S},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_{\hat{h}_S}} + \sqrt{\frac{1}{2}\log\frac{6+\pi^2}{6\delta}}\right]$$

by def of SRM
$$\le L_S(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_{h^*}} + \sqrt{\frac{1}{2}\log\frac{6+\pi^2}{6\delta}}\right]$$

w/ prob at least $1-\frac{6}{6+\pi^2}\delta$
$$\le L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_{h^*}} + \sqrt{\frac{1}{2}\log\frac{6+\pi^2}{6\delta}} + \sqrt{\frac{1}{2}\log\frac{6+\pi^2}{6\delta}}\right]$$

$$= L_{\mathcal{D}}(h^*) + 2\mathcal{R}_{k_{h^*},m} + \frac{b-a}{\sqrt{m}}\left[\sqrt{\log k_{h^*}} + \sqrt{2\log\frac{1+\pi^2/6}{\delta}}\right].$$

Note that $1 + \pi^2/6 < 2.7$. This gets essentially the same result as (9.4), without requiring committing to a δ in the algorithm.

Note that this was only possible because $[a, b]$ didn't depend on $\mathcal{H}_k$. This isn't always true; for example, our analysis of logistic regression with $\|x\| \le C$ and $\mathcal{H}_B = \{x \mapsto w \cdot x : \|w\| \le B\}$ used (4.4) to get that $b - a = BC$. In these cases, if we want to use our exact SRM analysis, as far as I know we have to incorporate δ into

the algorithm itself.

9.1.4 *Relationship to regularization*

Think about using SRM with $\mathcal{H} = \{x \mapsto w \cdot x : w \in \mathbb{R}^d\}$ with $\mathcal{H}_k = \{x \mapsto w \cdot x : \|w\| \leq r2^{k-1}\}$ for some $r > 0$; this should be chosen in advance of seeing the data, e.g. just picking $r = 1$. Consider logistic loss, and assume $\|x\| \leq C$ almost surely.

Suppose that h corresponds to a vector w. If $\|w\| \geq r$, we have

$$B_{k_h-1} = \frac{1}{2}B_{k_h} = r2^{k_h-2} < \|w\| \leq r2^{k_h-1} = B_{k_h},$$

implying $B_{k_h} < 2\|w\|$ and $k_h < 2 + \log_2 \frac{\|w\|}{r}$. Thus, in general, $B_{k_h} < \max(2\|w\|, r)$ and $k_h < 2 + \max\left(0, \log_2 \frac{\|w\|}{r}\right) = \max\left(2, \log_2 \frac{4\|w\|}{r}\right)$. Thus, recalling Section 5.2.2 and Equation (4.4), we can use (9.3) to construct an instance of SRM as

$$\arg\min_{w\in\mathbb{R}^d} L_S(x \mapsto w\cdot x) + \frac{C\max(2\|w\|, r)}{\sqrt{m}}\left[2 + \sqrt{\log\left(\max\left(2, \log_2 \frac{4\|w\|}{r}\right)\right) + \frac{1}{2}\log\frac{\pi^2}{6\delta}}\right].$$

Now, let's squint a bit, and assume that we chose an r such that the w with $\|w\|$ significantly smaller than r aren't relevant to the optimization – they're not confident enough to achieve a small L_S – but that getting a low L_S doesn't require a $\|w\|$ *so* big that $\log\log_2 \frac{4\|w\|}{r}$ is meaningfully more than "constant." Then, this optimization problem looks a lot like

$$\arg\min_{w\in\mathbb{R}^d} L_S(x \mapsto w \cdot x) + \frac{\lambda}{\sqrt{m}}\|w\|$$

for some $\lambda > 0$. This is pretty close to the "default" regularized logistic regression, which would use $\|w\|^2$. (It also probably wouldn't have an explicit m in the equation, but if you're tuning λ for a fixed particular problem, that doesn't matter, and indeed the total amount of regularization should often scale with m according to $\sqrt{m}$, as we'll see a little later in the course.)

In fact, the optimization problems with $\|w\|$ and with $\|w\|^2$ are themselves equivalent: if you consider the curve of possible solutions as you vary λ (the "regularization path"), you would get the exact same set of solutions. So, SRM can be seen as motivation for standard regularization techniques.

9.2 NONUNIFORM LEARNABILITY

The sample complexity for SRM to learn a hypothesis h^* depends on the particular h^*, not just on $\mathcal{H}$. This motivates a weaker definition of learning than PAC learning, called *nonuniform learning*.

Definition 9.4. An algorithm $\mathcal{A}(S)$ (ε, δ)-*competes with* a hypothesis h if it satisfies $\Pr_{S\sim\mathcal{D}^m}(L_{\mathcal{D}}(\mathcal{A}(S)) \leq L_{\mathcal{D}}(h) + \varepsilon) \geq 1 - \delta$.

Definition 9.5. An algorithm $\mathcal{A}$ *nonuniformly learns* $\mathcal{H}$ there is a finite sample complexity function $m(\varepsilon, \delta, h)$ such that for all $\varepsilon, \delta \in (0, 1)$ and $h \in \mathcal{H}$, given $m \geq m(\varepsilon, \delta, h)$ iid samples from any $\mathcal{D}$, $\mathcal{A}(S)$ (ε, δ)-competes with h.

Definition 9.6. A hypothesis class $\mathcal{H}$ is *nonuniformly learnable* if there exists an algorithm $\mathcal{A}$ which nonuniformly learns $\mathcal{H}$.

Theorem 9.3 establishes that SRM nonuniformly learns any $\mathcal{H}$ which we can decompose into a countable union of $\mathcal{H}_k$ which each allow for uniform convergence.

In fact, for binary classifiers with 0-1 loss, SRM nonuniformly learns any $\mathcal{H}$ which is nonuniformly learnable:

Proposition 9.7. *If $\mathcal{H}$ of binary classifiers is nonuniformly learnable under the 0-1 loss, it can be written as a countable union of $\mathcal{H}_k$ with finite VC dimension.*

Proof. Define

$$\mathcal{H}_k = \left\{h \in \mathcal{H} : m\left(\tfrac{1}{8}, \tfrac{1}{7}, h\right) \leq k\right\},$$

where $m(\varepsilon, \delta, h)$ is the sample complexity function of an algorithm $\mathcal{A}$ that nonuniformly learns $\mathcal{H}$. Then $\mathcal{H} = \bigcup_{k \geq 1} \mathcal{H}_k$.

For any k, consider $\mathcal{H}_k$. Let $\mathcal{D}$ be any distribution realizable by $\mathcal{H}_k$, i.e. there is some $h^* \in \mathcal{H}_k$ with $L_{\mathcal{D}}(h^*) = 0$. Since $\mathcal{A}(S)$ competes with that h^*, $\Pr_{S \sim \mathcal{D}^m}(L_{\mathcal{D}}(A(S)) \leq \frac{1}{8}) \geq \frac{6}{7}$. This means that we can (roughly) learn *any* realizable distribution. But our No Free Lunch theorem, specifically Corollary 8.2, implied that, if $\text{VCdim}(\mathcal{H}_k) = \infty$, then there would be some realizable $\mathcal{D}$ that we can't learn to this (ε, δ). Thus $\text{VCdim}(\mathcal{H}_k)$ can't be infinite. □

9.3 MINIMUM DESCRIPTION LENGTH

9.3.1 *Singleton Classes*

Suppose we have a countable $\mathcal{H} = \{h_1, h_2, \dots\}$. Then we could partition it into *singleton* sub-classes, $\mathcal{H}_k = \{h_k\}$. Denoting the weight for the class $\{h\}$ by w_h, each of these $\mathcal{H}_k$ have "uniform convergence" via a simple Hoeffding bound with

$$\varepsilon_k(m, w_h\delta) \leq (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{w_h\delta}} \leq (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{w_h}} + (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{\delta}},$$

splitting out the dependence on δ for simplicity as in Section 9.1.3. SRM then becomes

$$\text{SRM}_{\mathcal{H}}(S) \in \arg\min_{h \in \mathcal{H}} L_S(h) + \sqrt{\frac{1}{2m}\log\frac{1}{w_h}},$$

and this has the guarantee by Theorem 9.3 that

$$L_{\mathcal{D}}(\text{SRM}_{\mathcal{H}}(S)) \leq L_{\mathcal{D}}(h^*) + (b-a)\sqrt{\frac{1}{2m}\log\frac{1}{w_{h^*}}} + (b-a)\sqrt{\frac{2}{m}\log\frac{2}{\delta}}.$$

But... how should we set w_h? There's no "smaller" h; what order should we use?

9.3.2 *Minimum Description Length*

One popular way to decide on weights is based on choosing some *prefix-free binary language* to determine the hypotheses: for example, the binary representation of a `gz`iped Python program implementing that hypothesis. Then we can choose a weight according to the following result:

Proposition 9.8 (Kraft's inequality)**.** *If $\mathcal{S} \subseteq \{0,1\}^*$ is prefix-free (there are no $s \neq s' \in \mathcal{S}$*

such that s is a prefix of s'), then

$$\sum_{s \in \mathcal{S}} 2^{-|s|} \leq 1.$$

Proof. Define the following random process: starting with the empty string, add either a 0 or a 1 with equal probability. If the current string is in $\mathcal{S}$, terminate; if no element of $\mathcal{S}$ begins with the current string, also terminate; otherwise, repeat. Since $\mathcal{S}$ is prefix-free, this process hits any string $s \in \mathcal{S}$ with probability $2^{-|s|}$; these probabilities must sum to at most one. □

Thus, we can choose a representation for $\mathcal{H}$ so that h has description length $|h|$, and assign $w_h = 2^{-|h|}$. This gives

$$\mathrm{MDL}_{\mathcal{H}}(S) \in \arg\min_{h \in \mathcal{H}} L_S(h) + \sqrt{\frac{\log 2}{2m} |h|}$$

$$L_{\mathcal{D}}(\mathrm{MDL}_{\mathcal{H}}(S)) \leq L_{\mathcal{D}}(h^*) + (b-a)\sqrt{\frac{\log 2}{2m} |h^*|} + (b-a)\sqrt{\frac{2}{m} \log \frac{2}{\delta}}.$$

This is one formalization of Occam's razor: if there are multiple explanations of the data ($L_S(h_1) = 0 = L_S(h_2)$), prefer the simplest one (the one with shortest explanation).

But we need to *pre-commit* to a notion of description length before seeing the data. A nice analogy: `codegolf.stackexchange.com`, a site where people compete to find the shortest implementation of a program doing some task, prohibits by default any language written after the contest was started.

If we choose $|h|$ to be the length of shortest possible implementation of h in some programming language, this is known as the *Kolmogorov complexity*. This version of the MDL principle is then to regularize by the Kolmogorov complexity. If you're familiar with Bayesian learning, this would be something like *maximum a posteriori* (MAP) inference with a Kolmogorov complexity prior. The "free lunch" algorithm outlined by Nakkiran [Nak21] is closely related to this where $\mathcal{H}$ is just the set of all Turing machines. The fully-Bayesian analogue is (basically) something called *Solomonoff induction*. For fuller introductions to these concepts, there are various relevant textbooks [LV19; Hut05; HQC24].

It's not quite *the same; MAP wouldn't have the square root.*

REFERENCES

[HQC24] Marcus Hutter, David Quarel, and Elliot Catt. *An Introduction to Universal Artificial Intelligence*. 2024.

[Hut05] Marcus Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. 2005.

[LV19] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. 4th edition. 2019.

[Nak21] Preetum Nakkiran. *Turing-Universal Learners with Optimal Scaling Laws*. 2021. arXiv: 2111.05321.

[Zho+22] Lijia Zhou, Frederic Koehler, Pragya Sur, Danica J. Sutherland, and Nathan Srebro. "A Non-Asymptotic Moreau Envelope Theory for High-Dimensional Generalized Linear Models". *NeurIPS*. 2022. arXiv: 2210.12082.