Introduction to Machine Learning (67577)
Lecture 5

Shai Shalev-Shwartz

School of CS and Engineering,
The Hebrew University of Jerusalem

Nonuniform learning, MDL, SRM, Decision Trees, Nearest Neighbor

Outline

How to Express Prior Knowledge

- So far, learner expresses prior knowledge by specifying the hypothesis class $\mathcal{H}$

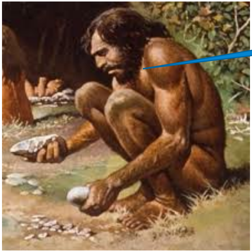
Other Ways to Express Prior Knowledge

Occam's Razor: "A short explanation is preferred over a longer one"

Other Ways to Express Prior Knowledge

Occam's Razor: "A short explanation is preferred over a longer one"

"Things that look alike must be alike"



Outline

Bias to Shorter Description

- Let $\mathcal{H}$ be a countable hypothesis class
- Let $w : \mathcal{H} \to \mathbb{R}$ be such that $\sum_{h \in \mathcal{H}} w(h) \leq 1$
- The function w reflects prior knowledge on how important $w(h)$ is

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs
- Suppose that the description language is prefix-free, namely, for every $h \neq h'$, $d(h)$ is not a prefix of $d(h')$
  (Always achievable by including an "end-of-word" symbol)

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs
- Suppose that the description language is prefix-free, namely, for every $h \neq h'$, $d(h)$ is not a prefix of $d(h')$
  (Always achievable by including an "end-of-word" symbol)
- Let $|h|$ be the length of $d(h)$

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs
- Suppose that the description language is prefix-free, namely, for every $h \neq h'$, $d(h)$ is not a prefix of $d(h')$
  (Always achievable by including an "end-of-word" symbol)
- Let $|h|$ be the length of $d(h)$
- Then, set $w(h) = 2^{-|h|}$

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs
- Suppose that the description language is prefix-free, namely, for every $h \neq h'$, $d(h)$ is not a prefix of $d(h')$
  (Always achievable by including an "end-of-word" symbol)
- Let $|h|$ be the length of $d(h)$
- Then, set $w(h) = 2^{-|h|}$
- Kraft's inequality implies that $\sum_h w(h) \leq 1$

Example: Description Length

- Suppose that each $h \in \mathcal{H}$ is described by some word $d(h) \in \{0,1\}^*$
  E.g.: $\mathcal{H}$ is the class of all python programs
- Suppose that the description language is prefix-free, namely, for every $h \neq h'$, $d(h)$ is not a prefix of $d(h')$
  (Always achievable by including an "end-of-word" symbol)
- Let $|h|$ be the length of $d(h)$
- Then, set $w(h) = 2^{-|h|}$
- Kraft's inequality implies that $\sum_h w(h) \le 1$
  - Proof: define probability over words in $d(\mathcal{H})$ as follows: repeatedly toss an unbiased coin, until the sequence of outcomes is a member of $d(\mathcal{H})$, and then stop. Since $d(\mathcal{H})$ is prefix-free, this is a valid probability over $d(\mathcal{H})$, and the probability to get $d(h)$ is $w(h)$.

Bias to Shorter Description

Theorem (Minimum Description Length (MDL) bound)

Let $w : \mathcal{H} \to \mathbb{R}$ be such that $\sum_{h \in \mathcal{H}} w(h) \le 1$. Then, with probability of at least $1 - \delta$ over $S \sim \mathcal{D}^m$ we have:

$$\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$$

Bias to Shorter Description

Theorem (Minimum Description Length (MDL) bound)

Let $w : \mathcal{H} \to \mathbb{R}$ be such that $\sum_{h \in \mathcal{H}} w(h) \le 1$. Then, with probability of at least $1 - \delta$ over $S \sim \mathcal{D}^m$ we have:

$$\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$$

Compare to VC bound:

$$\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$$

Proof

- For every h, define $\delta_h = w(h) \cdot \delta$

Proof

- For every h, define $\delta_h = w(h) \cdot \delta$
- By Hoeffding's bound, for every h,

$$\mathcal{D}^m\left(\left\{S : L_{\mathcal{D}}(h) > L_S(h) + \sqrt{\frac{\log(2/\delta_h)}{2m}}\right\}\right) \leq \delta_h$$

Proof

- For every h, define $\delta_h = w(h) \cdot \delta$
- By Hoeffding's bound, for every h,

$$\mathcal{D}^m \left(\left\{ S : L_{\mathcal{D}}(h) > L_S(h) + \sqrt{\frac{\log(2/\delta_h)}{2m}} \right\} \right) \le \delta_h$$

- Applying the union bound,

$$\mathcal{D}^m \left(\left\{ S : \exists h \in \mathcal{H}, L_{\mathcal{D}}(h) > L_S(h) + \sqrt{\frac{\log(2/\delta_h)}{2m}} \right\} \right) =$$

$$\mathcal{D}^m \left(\cup_{h \in \mathcal{H}} \left\{ S : L_{\mathcal{D}}(h) > L_S(h) + \sqrt{\frac{\log(2/\delta_h)}{2m}} \right\} \right) \le$$

$$\sum_{h \in \mathcal{H}} \delta_h \le \delta .$$

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$
- Recall that our goal is to minimize $L_{\mathcal{D}}(h)$ over $h \in \mathcal{H}$

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$

- Recall that our goal is to minimize $L_{\mathcal{D}}(h)$ over $h \in \mathcal{H}$
- Minimizing the VC bound leads to the ERM rule

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\; L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\; L_D(h) \le L_S(h) + C\,\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$

- Recall that our goal is to minimize $L_{\mathcal{D}}(h)$ over $h \in \mathcal{H}$
- Minimizing the VC bound leads to the ERM rule
- Minimizing the MDL bound leads to the MDL rule:

$$\mathrm{MDL}(S) \;\in\; \operatorname*{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}} \right]$$

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$

- Recall that our goal is to minimize $L_{\mathcal{D}}(h)$ over $h \in \mathcal{H}$
- Minimizing the VC bound leads to the ERM rule
- Minimizing the MDL bound leads to the MDL rule:

$$\mathrm{MDL}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}} \right]$$

- When $w(h) = 2^{-|h|}$ we obtain $-\log(w(h)) = |h|\log(2)$

Bound Minimization

- MDL bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}}$
- VC bound: $\forall h \in \mathcal{H},\ L_D(h) \le L_S(h) + C\sqrt{\frac{\mathrm{VCdim}(\mathcal{H}) + \log(2/\delta)}{2m}}$

- Recall that our goal is to minimize $L_{\mathcal{D}}(h)$ over $h \in \mathcal{H}$
- Minimizing the VC bound leads to the ERM rule
- Minimizing the MDL bound leads to the MDL rule:

$$\mathrm{MDL}(S) \ \in \ \operatorname*{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \sqrt{\frac{-\log(w(h)) + \log(2/\delta)}{2m}} \right]$$

- When $w(h) = 2^{-|h|}$ we obtain $-\log(w(h)) = |h|\log(2)$
- Explicit tradeoff between bias (small $L_S(h)$) and complexity (small $|h|$)

MDL guarantee

Theorem

For every $h^* \in \mathcal{H}$*, w.p.* $\geq 1 - \delta$ *over* $S \sim \mathcal{D}^m$ *we have:*

$$L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{-\log(w(h^*)) + \log(2/\delta)}{2m}}$$

MDL guarantee

Theorem

For every $h^* \in \mathcal{H}$, w.p. $\geq 1-\delta$ over $S \sim \mathcal{D}^m$ we have:

$$L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{-\log(w(h^*)) + \log(2/\delta)}{2m}}$$

- Example: Take $\mathcal{H}$ to be the class of all python programs, with $|h|$ be the code length (in bits)

MDL guarantee

Theorem

For every $h^* \in \mathcal{H}$, w.p. $\geq 1 - \delta$ over $S \sim \mathcal{D}^m$ we have:

$$L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{-\log(w(h^*)) + \log(2/\delta)}{2m}}$$

- Example: Take $\mathcal{H}$ to be the class of all python programs, with $|h|$ be the code length (in bits)
- Assume $\exists h^* \in \mathcal{H}$ with $L_{\mathcal{D}}(h^*) = 0$. Then, for every ϵ, δ, exists sample size m s.t. $\mathcal{D}^m(\{S : L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq \epsilon\}) \geq 1 - \delta$

MDL guarantee

Theorem

For every $h^* \in \mathcal{H}$, w.p. $\geq 1-\delta$ over $S \sim \mathcal{D}^m$ we have:

$$L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{-\log(w(h^*)) + \log(2/\delta)}{2m}}$$

- Example: Take $\mathcal{H}$ to be the class of all python programs, with $|h|$ be the code length (in bits)
- Assume $\exists h^* \in \mathcal{H}$ with $L_{\mathcal{D}}(h^*) = 0$. Then, for every ϵ, δ, exists sample size m s.t. $\mathcal{D}^m(\{S : L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq \epsilon\}) \geq 1-\delta$
- MDL is a Universal Learner

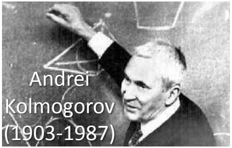
MDL guarantee

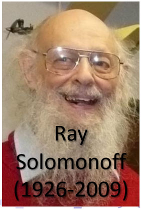
Theorem

For every $h^* \in \mathcal{H}$, w.p. $\geq 1-\delta$ over $S \sim \mathcal{D}^m$ we have:

$$L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{-\log(w(h^*)) + \log(2/\delta)}{2m}}$$

- Example: Take $\mathcal{H}$ to be the class of all python programs, with $|h|$ be the code length (in bits)
- Assume $\exists h^* \in \mathcal{H}$ with $L_{\mathcal{D}}(h^*) = 0$. Then, for every ϵ, δ, exists sample size m s.t. $\mathcal{D}^m(\{S : L_{\mathcal{D}}(\mathrm{MDL}(S)) \leq \epsilon\}) \geq 1-\delta$
- MDL is a Universal Learner





Contradiction to the fundamental theorem of learning ?

- Take again $\mathcal{H}$ to be all python programs
- Note that $\mathrm{VCdim}(\mathcal{H}) = \infty$
- The No-Free-Lunch theorem we can't learn $\mathcal{H}$
- So how come we can learn $\mathcal{H}$ using MDL ???

Outline

Non-uniform learning

Definition (Non-uniformly learnable)

$\mathcal{H}$ is *non-uniformly learnable* if $\exists A$ and $m_{\mathcal{H}}^{\text{NUL}} : (0,1)^2 \times \mathcal{H} \to \mathbb{N}$ s.t.,
$\forall \epsilon, \delta \in (0,1), \forall h \in \mathcal{H}$, if $m \geq m_{\mathcal{H}}^{\text{NUL}}(\epsilon, \delta, h)$ then $\forall \mathcal{D}$,

$$\mathcal{D}^m\left(\{S : L_{\mathcal{D}}(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon\}\right) \geq 1 - \delta \ .$$

- Number of required examples depends on ϵ, δ, and h

Definition (Agnostic PAC learnable)

$\mathcal{H}$ is agnostically PAC learnable if $\exists A$ and $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ s.t.
$\forall \epsilon, \delta \in (0,1)$, if $m \geq m_{\mathcal{H}}(\epsilon, \delta)$, then $\forall \mathcal{D}$ and $\forall h \in \mathcal{H}$,

$$\mathcal{D}^m\left(\{S : L_{\mathcal{D}}(A(S)) \leq L_{\mathcal{D}}(h) + \epsilon\}\right) \geq 1 - \delta \ .$$

- Number of required examples depends only on ϵ, δ

Non-uniform learning vs. PAC learning

Corollary

Let $\mathcal{H}$ be the class of all computable functions

- *$\mathcal{H}$ is non-uniform learnable, with sample complexity,*

$$m_{\mathcal{H}}^{NUL}(\epsilon, \delta, h) \;\le\; \frac{-\log(w(h)) + \log(2/\delta)}{2\epsilon^2}$$

- *$\mathcal{H}$ is not PAC learnable.*

Non-uniform learning vs. PAC learning

Corollary

Let $\mathcal{H}$ be the class of all computable functions

- *$\mathcal{H}$ is non-uniform learnable, with sample complexity,*

$$m_{\mathcal{H}}^{\mathit{NUL}}(\epsilon, \delta, h) \;\leq\; \frac{-\log(w(h)) + \log(2/\delta)}{2\epsilon^2}$$

- *$\mathcal{H}$ is not PAC learnable.*

- We saw that the VC dimension characterizes PAC learnability
- What characterizes non-uniform learnability ?

Characterizing Non-uniform Learnability

Theorem

A class $\mathcal{H} \subset \{0,1\}^{\mathcal{X}}$ is non-uniform learnable if and only if it is a countable union of PAC learnable hypothesis classes.

Proof (Non-uniform learnable $\Rightarrow$ countable union)

- Assume that $\mathcal{H}$ is non-uniform learnable using A with sample complexity $m_{\mathcal{H}}^{\text{NUL}}$

Proof (Non-uniform learnable $\Rightarrow$ countable union)

- Assume that $\mathcal{H}$ is non-uniform learnable using A with sample complexity $m_{\mathcal{H}}^{\mathsf{NUL}}$
- For every $n \in \mathbb{N}$, let $\mathcal{H}_n = \{h \in \mathcal{H} : m_{\mathcal{H}}^{\mathsf{NUL}}(1/8, 1/7, h) \le n\}$

Proof (Non-uniform learnable $\Rightarrow$ countable union)

- Assume that $\mathcal{H}$ is non-uniform learnable using A with sample complexity $m^{\mathsf{NUL}}_{\mathcal{H}}$
- For every $n \in \mathbb{N}$, let $\mathcal{H}_n = \{h \in \mathcal{H} : m^{\mathsf{NUL}}_{\mathcal{H}}(1/8, 1/7, h) \le n\}$
- Clearly, $\mathcal{H} = \cup_{n\in\mathbb{N}}\mathcal{H}_n$.

Proof (Non-uniform learnable $\Rightarrow$ countable union)

- Assume that $\mathcal{H}$ is non-uniform learnable using A with sample complexity $m_{\mathcal{H}}^{\mathsf{NUL}}$
- For every $n \in \mathbb{N}$, let $\mathcal{H}_n = \{h \in \mathcal{H} : m_{\mathcal{H}}^{\mathsf{NUL}}(1/8, 1/7, h) \le n\}$
- Clearly, $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$.
- For every $\mathcal{D}$ s.t. $\exists h \in \mathcal{H}_n$ with $L_{\mathcal{D}}(h) = 0$ we have that $\mathcal{D}^n(\{S : L_{\mathcal{D}}(A(S)) \le 1/8\}) \ge 6/7$

Proof (Non-uniform learnable $\Rightarrow$ countable union)

- Assume that $\mathcal{H}$ is non-uniform learnable using A with sample complexity $m_{\mathcal{H}}^{\text{NUL}}$
- For every $n \in \mathbb{N}$, let $\mathcal{H}_n = \{h \in \mathcal{H} : m_{\mathcal{H}}^{\text{NUL}}(1/8, 1/7, h) \le n\}$
- Clearly, $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$.
- For every $\mathcal{D}$ s.t. $\exists h \in \mathcal{H}_n$ with $L_{\mathcal{D}}(h) = 0$ we have that $\mathcal{D}^n(\{S : L_{\mathcal{D}}(A(S)) \le 1/8\}) \ge 6/7$
- The fundamental theorem of statistical learning implies that $\mathrm{VCdim}(\mathcal{H}_n) < \infty$, and therefore $\mathcal{H}_n$ is agnostic PAC learnable

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n\in\mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$
- Choose $w : \mathbb{N} \to [0,1]$ s.t. $\sum_n w(n) \leq 1$. E.g. $w(n) = \frac{6}{\pi^2 n^2}$

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$
- Choose $w : \mathbb{N} \to [0,1]$ s.t. $\sum_n w(n) \le 1$. E.g. $w(n) = \frac{6}{\pi^2 n^2}$
- Choose $\delta_n = \delta \cdot w(n)$ and $\epsilon_n = \sqrt{C \frac{d_n + \log(1/\delta_n)}{m}}$

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$
- Choose $w : \mathbb{N} \to [0,1]$ s.t. $\sum_n w(n) \leq 1$. E.g. $w(n) = \frac{6}{\pi^2 n^2}$
- Choose $\delta_n = \delta \cdot w(n)$ and $\epsilon_n = \sqrt{C \frac{d_n + \log(1/\delta_n)}{m}}$
- By the fundamental theorem, for every n,

$$\mathcal{D}^m(\{S : \exists h \in \mathcal{H}_n, L_{\mathcal{D}}(h) > L_S(h) + \epsilon_n\}) \leq \delta_n \ .$$

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$
- Choose $w : \mathbb{N} \to [0,1]$ s.t. $\sum_n w(n) \le 1$. E.g. $w(n) = \frac{6}{\pi^2 n^2}$
- Choose $\delta_n = \delta \cdot w(n)$ and $\epsilon_n = \sqrt{C \frac{d_n + \log(1/\delta_n)}{m}}$
- By the fundamental theorem, for every n,

$$\mathcal{D}^m(\{S : \exists h \in \mathcal{H}_n, L_{\mathcal{D}}(h) > L_S(h) + \epsilon_n\}) \le \delta_n \ .$$

- Applying the union bound over n we obtain

$$\mathcal{D}^m(\{S : \exists n, h \in \mathcal{H}_n, L_{\mathcal{D}}(h) > L_S(h) + \epsilon_n\}) \le \sum_n \delta_n \le \delta \ .$$

Proof (Countable union $\Rightarrow$ non-uniform learnable)

- Assume $\mathcal{H} = \cup_{n \in \mathbb{N}} \mathcal{H}_n$, and $\mathrm{VCdim}(\mathcal{H}_n) = d_n < \infty$
- Choose $w : \mathbb{N} \to [0,1]$ s.t. $\sum_n w(n) \le 1$. E.g. $w(n) = \frac{6}{\pi^2 n^2}$
- Choose $\delta_n = \delta \cdot w(n)$ and $\epsilon_n = \sqrt{C \frac{d_n + \log(1/\delta_n)}{m}}$
- By the fundamental theorem, for every n,

$$\mathcal{D}^m(\{S : \exists h \in \mathcal{H}_n, L_{\mathcal{D}}(h) > L_S(h) + \epsilon_n\}) \le \delta_n \ .$$

- Applying the union bound over n we obtain

$$\mathcal{D}^m(\{S : \exists n, h \in \mathcal{H}_n, L_{\mathcal{D}}(h) > L_S(h) + \epsilon_n\}) \le \sum_n \delta_n \le \delta \ .$$

- This yields a generic non-uniform learning rule

Outline

Structural Risk Minimization (SRM)

$$\mathrm{SRM}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \min_{n : h \in \mathcal{H}_n} \sqrt{C \frac{d_n - \log(w(n)) + \log(1/\delta)}{m}} \right]$$

Structural Risk Minimization (SRM)

$$\mathrm{SRM}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} \left[L_S(h) + \min_{n : h \in \mathcal{H}_n} \sqrt{C \frac{d_n - \log(w(n)) + \log(1/\delta)}{m}} \right]$$

- As in the analysis of MDL, it is easy to show that for every $h \in \mathcal{H}$,

$$L_{\mathcal{D}}(\mathrm{SRM}(S)) \;\leq\; L_S(h) + \min_{n : h \in \mathcal{H}_n} \sqrt{C \frac{d_n - \log(w(n)) + \log(1/\delta)}{m}}$$

Structural Risk Minimization (SRM)

$$\mathrm{SRM}(S) \in \operatorname*{argmin}_{h\in\mathcal{H}} \left[L_S(h) + \min_{n:h\in\mathcal{H}_n} \sqrt{C\frac{d_n - \log(w(n)) + \log(1/\delta)}{m}} \right]$$

- As in the analysis of MDL, it is easy to show that for every $h \in \mathcal{H}$,

$$L_{\mathcal{D}}(\mathrm{SRM}(S)) \;\le\; L_S(h) + \min_{n:h\in\mathcal{H}_n} \sqrt{C\frac{d_n - \log(w(n)) + \log(1/\delta)}{m}}$$

- Hence, SRM is a generic non-uniform learner with sample complexity

$$m_{\mathcal{H}}^{\text{NUL}}(\epsilon,\delta,h) \le \min_{n:h\in\mathcal{H}_n} C\,\frac{d_n - \log(w(n)) + \log(1/\delta)}{\epsilon^2}$$

No-free-lunch for non-uniform learnability

- Claim: For any infinite domain set, $\mathcal{X}$, the class $\mathcal{H} = \{0,1\}^{\mathcal{X}}$ is not a countable union of classes of finite VC-dimension.
- Hence, such classes $\mathcal{H}$ are not non-uniformly learnable

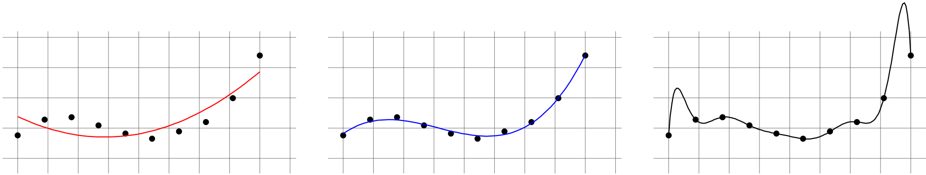
The cost of weaker prior knowledge

- Suppose $\mathcal{H} = \cup_n \mathcal{H}_n$, where $\mathrm{VCdim}(\mathcal{H}_n) = n$
- Suppose that some $h^* \in \mathcal{H}_n$ has $L_{\mathcal{D}}(h^*) = 0$
- With this prior knowledge, we can apply ERM on $\mathcal{H}_n$, and the sample complexity is $C\,\frac{n+\log(1/\delta)}{\epsilon^2}$
- Without this prior knowledge, SRM will need $C\,\frac{n+\log(\pi^2 n^2/6)+\log(1/\delta)}{\epsilon^2}$ examples
- That is, we pay order of $\log(n)/\epsilon^2$ more examples for not knowing n in advanced

The cost of weaker prior knowledge

- Suppose $\mathcal{H} = \cup_n \mathcal{H}_n$, where $\mathrm{VCdim}(\mathcal{H}_n) = n$
- Suppose that some $h^* \in \mathcal{H}_n$ has $L_{\mathcal{D}}(h^*) = 0$
- With this prior knowledge, we can apply ERM on $\mathcal{H}_n$, and the sample complexity is $C\,\frac{n+\log(1/\delta)}{\epsilon^2}$
- Without this prior knowledge, SRM will need $C\,\frac{n+\log(\pi^2 n^2/6)+\log(1/\delta)}{\epsilon^2}$ examples
- That is, we pay order of $\log(n)/\epsilon^2$ more examples for not knowing n in advanced

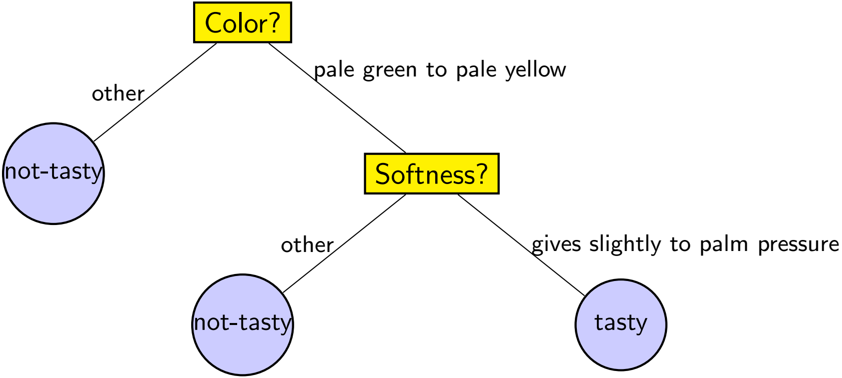
SRM for model selection:



Outline

1. Minimum Description Length
2. Non-uniform learnability
3. Structural Risk Minimization
4. Decision Trees
5. Nearest Neighbor and Consistency

Decision Trees



VC dimension of Decision Trees

- Claim: Consider the class of decision trees over $\mathcal{X}$ with k leaves. Then, the VC dimension of this class is k
- Proof: A set of k instances that arrive to the different leaves can be shattered. A set of $k+1$ instances can't be shattered since 2 instances must arrive to the same leaf

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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)

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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 '$\mathbb{1}_{[x_i=1]}$' for some $i \in [d]$

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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 '$\mathbb{1}_{[x_i=1]}$' for some $i \in [d]$
  - A leaf whose value is 1

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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 '$\mathbb{1}_{[x_i=1]}$' for some $i \in [d]$
  - A leaf whose value is 1
  - A leaf whose value is 0

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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 '$\mathbb{1}_{[x_i=1]}$' for some $i \in [d]$
  - A leaf whose value is 1
  - A leaf whose value is 0
  - End of the code

Description Language for Decision Trees

- Suppose $\mathcal{X} = \{0,1\}^d$ and splitting rules are according to $\mathbb{1}_{[x_i=1]}$ for some $i \in [d]$
- Consider the class of all such decision trees over $\mathcal{X}$
- Claim: This class contains $\{0,1\}^{\mathcal{X}}$ and hence its VC dimension is $|\mathcal{X}| = 2^d$
- But, we can bias to "small trees"
- 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 '$\mathbb{1}_{[x_i=1]}$' for some $i \in [d]$
  - A leaf whose value is 1
  - A leaf whose value is 0
  - End of the code
- Can apply MDL learning rule: search tree with n nodes that minimizes

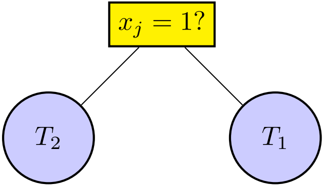
$$L_S(h) + \sqrt{\frac{(n+1)\log_2(d+3) + \log(2/\delta)}{2m}}$$

Decision Tree Algorithms

- NP hard problem ...
- Greedy approach: 'Iterative Dichotomizer 3'
- Following the MDL principle, attempts to create a small tree with low train error
- Proposed by Ross Quinlan

$\texttt{ID3}(S, A)$

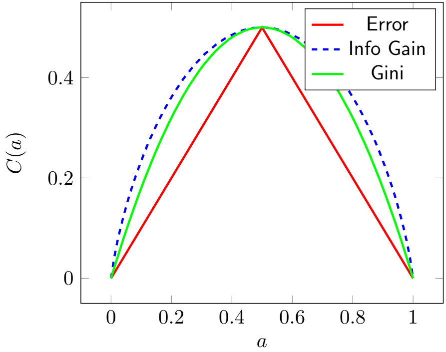
- INPUT: training set S, feature subset $A \subseteq [d]$
- **if** all examples in S are labeled by 1, return a leaf 1
- **if** all examples in S are labeled by 0, return a leaf 0
- **if** $A = \emptyset$, return a leaf whose value = majority of labels in S. **else** :
  - Let $j = \text{argmax}_{i \in A} \texttt{Gain}(S, i)$
  - **if** all examples in S have the same label
    Return a leaf whose value = majority of labels in S
  - **else**
    Let T_1 be the tree returned by $\texttt{ID3}(\{(\mathbf{x}, y) \in S : x_j = 1\}, A \setminus \{j\})$.
    Let T_2 be the tree returned by $\texttt{ID3}(\{(\mathbf{x}, y) \in S : x_j = 0\}, A \setminus \{j\})$.
    Return the tree:



Gain Measures

$$\texttt{Gain}(S,i) \;=\; C(\underset{S}{\mathbb{P}}[y]) - \left(\underset{S}{\mathbb{P}}[x_i]\,C(\underset{S}{\mathbb{P}}[y|x_i]) + \underset{S}{\mathbb{P}}[\neg x_i]C(\underset{S}{\mathbb{P}}[y|\neg x_i])\right).$$

- Train error: $C(a) = \min\{a, 1-a\}$
- Information gain: $C(a) = -a\log(a) - (1-a)\log(1-a)$
- Gini index: $C(a) = 2a(1-a)$



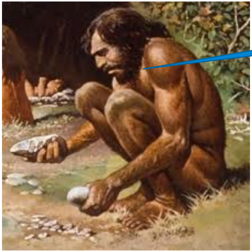
Pruning, Random Forests,...

In the exercise you'll learn about additional practical variants:

- Pruning the tree
- Random Forests
- Dealing with real valued features

Outline

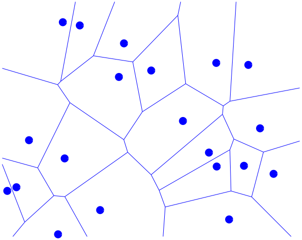
Nearest Neighbor



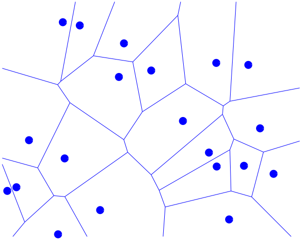
"Things that look alike must be alike"

- Memorize the training set $S = (x_1, y_1), \ldots, (x_m, y_m)$
- Given new x, find the k closest points in S and return majority vote among their labels

1-Nearest Neighbor: Voronoi Tessellation



1-Nearest Neighbor: Voronoi Tessellation



- Unlike ERM,SRM,MDL, etc., there's no $\mathcal{H}$
- At training time: "do nothing"
- At test time: search S for the nearest neighbors

Analysis of k-NN

- $\mathcal{X} = [0,1]^d, Y = \{0,1\}$, $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$, $\mathcal{D}_{\mathcal{X}}$ is the marginal distribution over $\mathcal{X}$, and $\eta : \mathbb{R}^d \to \mathbb{R}$ is the conditional probability over the labels, that is, $\eta(\mathbf{x}) = \mathbb{P}[y = 1|\mathbf{x}]$.

Analysis of k-NN

- $\mathcal{X} = [0,1]^d, Y = \{0,1\}$, $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$, $\mathcal{D}_\mathcal{X}$ is the marginal distribution over $\mathcal{X}$, and $\eta : \mathbb{R}^d \to \mathbb{R}$ is the conditional probability over the labels, that is, $\eta(\mathbf{x}) = \mathbb{P}[y = 1|\mathbf{x}]$.
- Recall: the Bayes optimal rule (that is, the hypothesis that minimizes $L_\mathcal{D}(h)$ over all functions) is

$$h^\star(\mathbf{x}) = \mathbb{1}_{[\eta(\mathbf{x})>1/2]} \ .$$

Analysis of k-NN

- $\mathcal{X} = [0,1]^d, Y = \{0,1\}$, $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$, $\mathcal{D}_{\mathcal{X}}$ is the marginal distribution over $\mathcal{X}$, and $\eta : \mathbb{R}^d \to \mathbb{R}$ is the conditional probability over the labels, that is, $\eta(\mathbf{x}) = \mathbb{P}[y = 1|\mathbf{x}]$.
- Recall: the Bayes optimal rule (that is, the hypothesis that minimizes $L_{\mathcal{D}}(h)$ over all functions) is

$$h^\star(\mathbf{x}) = \mathbb{1}_{[\eta(\mathbf{x})>1/2]} \ .$$

- Prior knowledge: η is c-Lipschitz. Namely, for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}, \quad |\eta(\mathbf{x}) - \eta(\mathbf{x}')| \leq c \, \|\mathbf{x} - \mathbf{x}'\|$

Analysis of k-NN

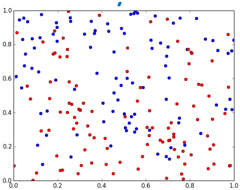
- $\mathcal{X} = [0,1]^d, Y = \{0,1\}$, $\mathcal{D}$ is a distribution over $\mathcal{X} \times \mathcal{Y}$, $\mathcal{D}_{\mathcal{X}}$ is the marginal distribution over $\mathcal{X}$, and $\eta : \mathbb{R}^d \to \mathbb{R}$ is the conditional probability over the labels, that is, $\eta(\mathbf{x}) = \mathbb{P}[y = 1|\mathbf{x}]$.
- Recall: the Bayes optimal rule (that is, the hypothesis that minimizes $L_{\mathcal{D}}(h)$ over all functions) is

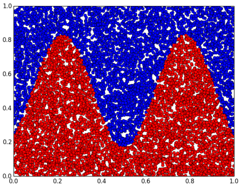
$$h^\star(\mathbf{x}) = \mathbb{1}_{[\eta(\mathbf{x})>1/2]} \ .$$

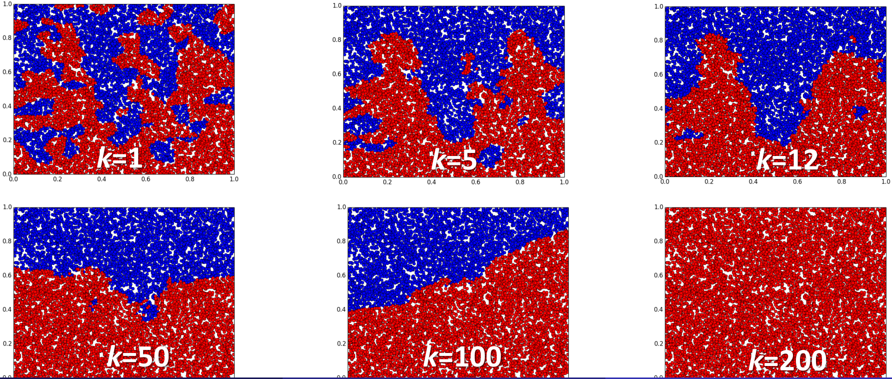
- Prior knowledge: η is c-Lipschitz. Namely, for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}, \quad |\eta(\mathbf{x}) - \eta(\mathbf{x}')| \le c \, \|\mathbf{x} - \mathbf{x}'\|$
- Theorem: Let h_S be the k-NN rule, then,

$$\mathop{\mathbb{E}}_{S \sim \mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \ \le \ \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^\star) + \left(6\,c\,\sqrt{d} + k\right) m^{-1/(d+1)} \ .$$

k-Nearest Neighbor: Bias-Complexity Tradeoff

$S=$ 

$h^*=$ 



Curse of Dimensionality

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \;\leq\; \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^\star) + \left(6\,c\,\sqrt{d}+k\right) m^{-1/(d+1)} \;.$$

- Suppose $L_{\mathcal{D}}(h^\star) = 0$. Then, to have error $\leq \epsilon$ we need $m \geq (4\,c\,\sqrt{d}/\epsilon)^{d+1}$.
- Number of examples grows exponentially with the dimension
- This is not an artifact of the analysis

Curse of Dimensionality

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \;\le\; \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^\star) + \left(6\,c\,\sqrt{d}+k\right) m^{-1/(d+1)} \;.$$

- Suppose $L_{\mathcal{D}}(h^\star) = 0$. Then, to have error $\le \epsilon$ we need $m \ge (4\,c\,\sqrt{d}/\epsilon)^{d+1}$.
- Number of examples grows exponentially with the dimension
- This is not an artifact of the analysis

Theorem

For any $c > 1$, and every learner, there exists a distribution over $[0,1]^d \times \{0,1\}$, such that $\eta(\mathbf{x})$ is c-Lipschitz, the Bayes error of the distribution is 0, but for sample sizes $m \le (c+1)^d/2$, the true error of the learner is greater than $1/4$.

Contradicting the No-Free-Lunch?

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \;\le\; \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^\star) + \left(6\,c\,\sqrt{d}+k\right) m^{-1/(d+1)} \;.$$

- Seemingly, we learn the class of all functions over $[0,1]^d$
- But this class is not learnable even in the non-uniform model ...

Contradicting the No-Free-Lunch?

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}(h_S)] \;\leq\; \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(h^\star) + \left(6\,c\,\sqrt{d}+k\right) m^{-1/(d+1)} \,.$$

- Seemingly, we learn the class of all functions over $[0,1]^d$
- But this class is not learnable even in the non-uniform model ...
- There's no contradiction: The number of required examples depends on the Lipschitzness of η (the parameter c), which depends on $\mathcal{D}$.
  - PAC: $m(\epsilon,\delta)$
  - non-uniform: $m(\epsilon,\delta,h)$
  - consistency: $m(\epsilon,\delta,h,\mathcal{D})$

Issues with Nearest Neighbor

- Need to store entire training set
  "Replace intelligence with fast memory"
- Curse of dimensionality
  We'll later learn dimensionality reduction methods
- Computational problem of finding nearest neighbor
- What is the "correct" metric between objects ?
  Success depends on Lipschitzness of η, which depends on the right metric

Summary

- Expressing prior knowledge: Hypothesis class, weighting hypotheses, metric
- Weaker notions of learnability:
  "PAC" stronger than "non-uniform" stronger than "consistency"
- Learning rules: ERM, MDL, SRM
- Decision trees
- Nearest Neighbor