Learning Kernel-Based Halfspaces with the Zero-One Loss

Shai Shalev-Shwartz[1], **Ohad Shamir**[1]
and Karthik Sridharan[2]

[1]The Hebrew University

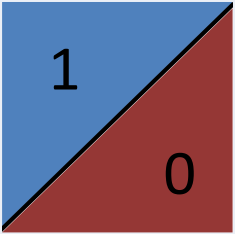
[2]TTI Chicago

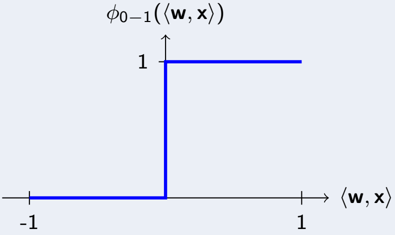
COLT, June 2010

Halfspaces

Hypothesis Class

$$\{\mathbf{x} \mapsto \phi_{0-1}(\langle \mathbf{w}, \mathbf{x} \rangle)\}$$



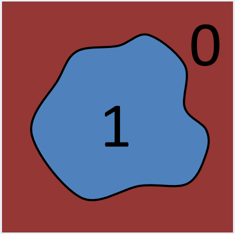


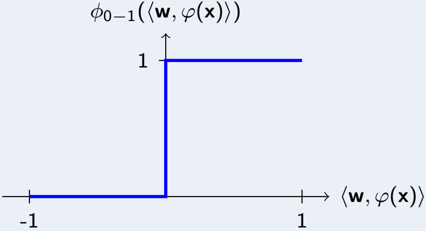
Sample Complexity: $\mathcal{O}(\mathbf{d}/\epsilon^{\mathbf{2}})$

Kernel-Based Halfspaces

Hypothesis Class

$$\{\mathbf{x} \mapsto \phi_{0-1}(\langle \mathbf{w}, \varphi(\mathbf{x}) \rangle)\}$$



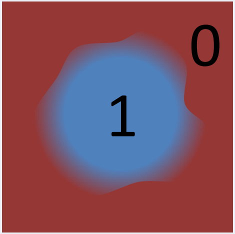


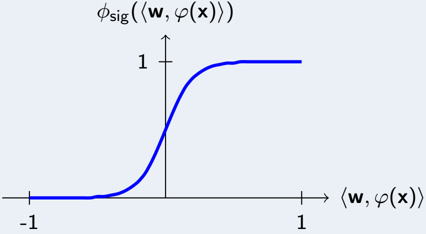
Sample Complexity: ∞

Fuzzy Kernel-Based Halfspaces

Hypothesis Class

$$\{\mathbf{x} \mapsto \phi_{\mathrm{sig}}(\langle \mathbf{w}, \varphi(\mathbf{x})\rangle)\}$$



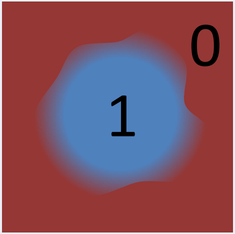


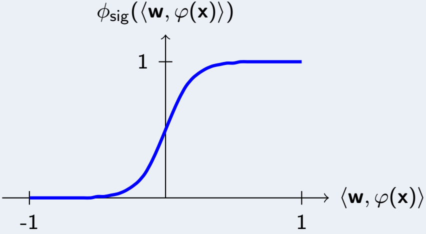
Sample Complexity: $\mathcal{O}(\mathbf{L}^2/\epsilon^2)$

Fuzzy Kernel-Based Halfspaces

Hypothesis Class

$$\{\mathbf{x} \mapsto \phi_{\text{sig}}(\langle \mathbf{w}, \varphi(\mathbf{x}) \rangle)\}$$





Sample Complexity: $\mathcal{O}(\mathbf{L}^2/\epsilon^2)$
Time Complexity: ??

Formal Results

Time complexity of learning Fuzzy Halfspaces

- **Positive Result**: can be done in poly$(1/\epsilon)$ for any fixed L (*worst case*)
  - Do convex optimization, just use a different kernel...
- **Negative Result**: can't be done in poly$(L, 1/\epsilon)$ time

Related Work: Surrogates to $0-1$ loss

- Popular fix: replace $0-1$ loss with convex loss (e.g., hinge loss)
  - No finite-sample approximation guarantees!
  - Asymptotic guarantees exist (Zhang 2004; Bartlett, Jordan, McAuliffe 2006)

Related Work: Surrogates to $0-1$ loss

- Popular fix: replace $0-1$ loss with convex loss (e.g., hinge loss)
  - No finite-sample approximation guarantees!
  - Asymptotic guarantees exist (Zhang 2004; Bartlett, Jordan, McAuliffe 2006)
- Ben-David & Simon 2000: By a covering technique, can learn fuzzy halfspaces in $\exp(\mathcal{O}(L^2/\epsilon^2))$ time
  - Worst case = best case
  - Exponentially worse than our bound (however, requires exponentially less examples)

Related Work: Directly for $0 - 1$ loss

- Agnostically learning halfspaces in poly(d^{1/ϵ^4}) time (Kalai, Klivans, Mansour, Servedio 2005; Blais, O'Donell, Wimmer 2008)
  - But only under distributional assumptions.
  - Dimension-dependent (problematic for kernels)

Technique Idea

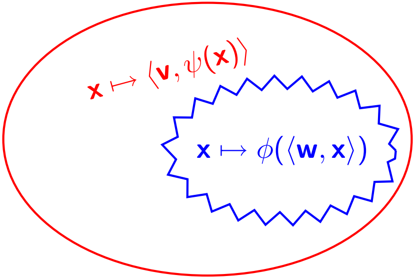
- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| = 1\}$
- Loss function: $\mathbb{E}_{\hat{y} \sim \phi(\langle \mathbf{w}, \mathbf{x} \rangle)} \mathbf{1}_{\hat{y} = y}$

Technique Idea

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| = 1\}$
- Loss function: $\mathbb{E}_{\hat{y} \sim \phi(\langle \mathbf{w}, \mathbf{x} \rangle)} \mathbf{1}_{\hat{y}=y} = |\phi(\langle \mathbf{w}, \mathbf{x} \rangle) - y|$

Technique Idea

- Original class: $\mathcal{H} = \{\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| = 1\}$
- Loss function: $\mathbb{E}_{\hat{y} \sim \phi(\langle \mathbf{w}, \mathbf{x} \rangle)} \mathbf{1}_{\hat{y} = y} = |\phi(\langle \mathbf{w}, \mathbf{x} \rangle) - y|$
- Problem: Loss is non-convex w.r.t. $\mathbf{w}$
- The main idea: Work with a larger hypothesis class for which the loss becomes convex



Technique Idea

- Assume $\|\mathbf{x}\| \leq 1$, and suppose that $\phi(a)$ is a polynomial $\sum_{j=0}^{\infty} \beta_j a^j$
- Then

$$\phi(\langle \mathbf{w}, \mathbf{x} \rangle) = \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j$$

Technique Idea

- Assume $\|\mathbf{x}\| \leq 1$, and suppose that $\phi(a)$ is a polynomial $\sum_{j=0}^{\infty} \beta_j a^j$
- Then

$$
\begin{aligned}
\phi(\langle \mathbf{w}, \mathbf{x} \rangle) &= \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j \\
&= \sum_{j=0}^{\infty} \sum_{k_1,\ldots,k_j} (2^{j/2} \beta_j w_{k_1} \cdots w_{k_j})(2^{-j/2} x_{k_1} \cdots x_{k_j})
\end{aligned}
$$

Technique Idea

- Assume $\|\mathbf{x}\| \leq 1$, and suppose that $\phi(a)$ is a polynomial $\sum_{j=0}^{\infty} \beta_j a^j$
- Then

$$\begin{aligned} \phi(\langle \mathbf{w}, \mathbf{x} \rangle) &= \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j \\ &= \sum_{j=0}^{\infty} \sum_{k_1,\ldots,k_j} (2^{j/2} \beta_j w_{k_1} \cdots w_{k_j})(2^{-j/2} x_{k_1} \cdots x_{k_j}) \\ &= \langle \mathbf{v}_{\mathbf{w}}, \Psi(\mathbf{x}) \rangle \end{aligned}$$

Technique Idea

- Assume $\|\mathbf{x}\| \leq 1$, and suppose that $\phi(a)$ is a polynomial $\sum_{j=0}^{\infty} \beta_j a^j$
- Then

$$\begin{aligned}
\phi(\langle \mathbf{w}, \mathbf{x} \rangle) &= \sum_{j=0}^{\infty} \beta_j (\langle \mathbf{w}, \mathbf{x} \rangle)^j \\
&= \sum_{j=0}^{\infty} \sum_{k_1, \ldots, k_j} (2^{j/2} \beta_j w_{k_1} \cdots w_{k_j})(2^{-j/2} x_{k_1} \cdots x_{k_j}) \\
&= \langle \mathbf{v}_{\mathbf{w}}, \Psi(\mathbf{x}) \rangle
\end{aligned}$$

- Ψ is the feature mapping of the RKHS corresponding to the infinite-dimensional polynomial kernel

$$k(\mathbf{x}, \mathbf{x}') = \frac{1}{1 - \frac{1}{2} \langle \mathbf{x}, \mathbf{x}' \rangle}$$

Technique Idea

Therefore, given sample $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$,

$$\min_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} |\phi(\langle \mathbf{w}, \mathbf{x}_i \rangle) - y_i|$$

equivalent to

$$\min_{\mathbf{v}_\mathbf{w}:\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} |\langle \mathbf{v}_\mathbf{w}, \Psi(\mathbf{x}_i) \rangle - y_i|$$

Technique Idea

Therefore, given sample $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$,

$$\min_{\mathbf{w}:\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} |\phi(\langle \mathbf{w}, \mathbf{x}_i \rangle) - y_i|$$

equivalent to

$$\min_{\mathbf{v}_{\mathbf{w}}:\|\mathbf{w}\|=1} \frac{1}{m} \sum_{i=1}^{m} |\langle \mathbf{v}_{\mathbf{w}}, \Psi(\mathbf{x}_i) \rangle - y_i|$$

Algorithm

$$\arg \min_{\mathbf{v}:\|\mathbf{v}\|\leq B} \frac{1}{m} \sum_{i=1}^{m} |\langle \mathbf{v}, \Psi(\mathbf{x}_i) \rangle - y_i|,$$

using the infinite-dimensional polynomial kernel

Technique Idea

Theorem

Let H_B consist of all predictors of the form $\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle)$, where

- $\phi(a) = \sum_{j=0}^{\infty} \beta_j a^j$
- $\sum_{j=0}^{\infty} 2^j \beta_j^2 \leq B$

With $\mathcal{O}(B/\epsilon^2)$ examples, returned predictor $\hat{\mathbf{v}}$ satisfies w.h.p.

$$err_{\mathcal{D}}(\hat{\mathbf{v}}) \leq \min_{\mathbf{v} \in H_B} err_{\mathcal{D}}(\mathbf{v}) + \epsilon$$

Technique Idea

Algorithm

$$\arg \min_{\mathbf{v}:\|\mathbf{v}\|\leq B} \frac{1}{m} \sum_{i=1}^{m} |\langle \mathbf{v}, \Psi(\mathbf{x}_i) \rangle - y_i|,$$

using the infinite-dimensional polynomial kernel

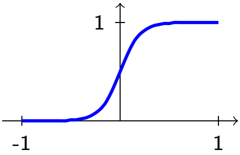
Technique Idea

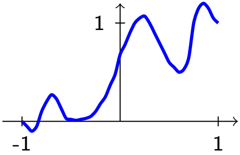
Algorithm

$$\arg\min_{\mathbf{v}:\|\mathbf{v}\|\leq B} \frac{1}{m}\sum_{i=1}^{m} |\langle \mathbf{v}, \Psi(\mathbf{x}_i)\rangle - y_i|,$$

using the infinite-dimensional polynomial kernel

- **Same** algorithm competitive against **all** ϕ with coefficient bound B - including optimal one for data distribution





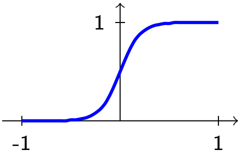
Technique Idea

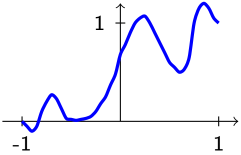
Algorithm

$$\arg \min_{\mathbf{v}:\|\mathbf{v}\|\leq B} \frac{1}{m} \sum_{i=1}^{m} |\langle \mathbf{v}, \Psi(\mathbf{x}_i)\rangle - y_i|,$$

using the infinite-dimensional polynomial kernel

- **Same** algorithm competitive against **all** ϕ with coefficient bound B - including optimal one for data distribution

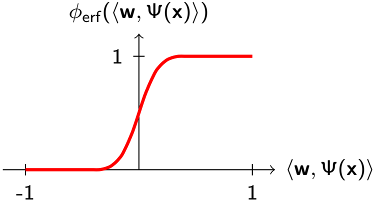




- In practice, parameter B chosen by cross validation. Algorithm can work much faster depending on distribution

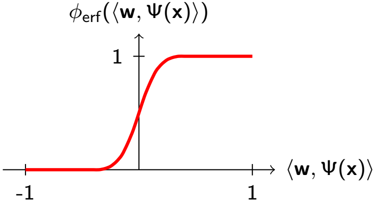
Example - Error Function

$$\phi_{\text{erf}}(\langle \mathbf{w}, \mathbf{x} \rangle) = \frac{1 + \text{erf}(\sqrt{\pi} L \langle \mathbf{w}, \mathbf{x} \rangle)}{2}$$

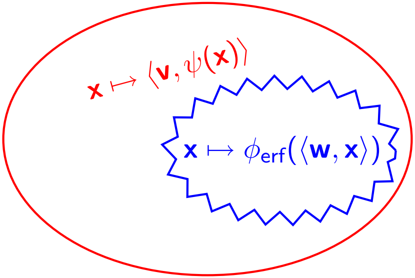


Example - Error Function

$$\phi_{\text{erf}}(\langle \mathbf{w}, \mathbf{x} \rangle) = \frac{1 + \text{erf}(\sqrt{\pi} L \langle \mathbf{w}, \mathbf{x} \rangle)}{2}$$

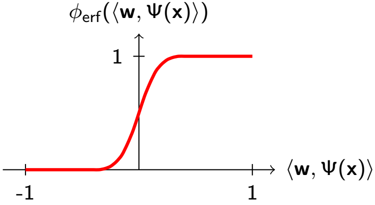


ϕ_{erf} can be written as an infinite-degree polynomial

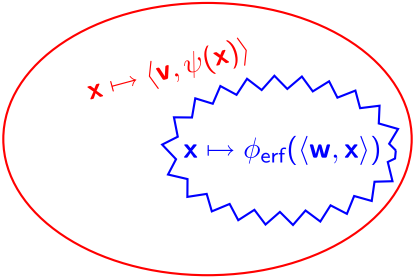


Example - Error Function

$$\phi_{\text{erf}}(\langle \mathbf{w}, \mathbf{x} \rangle) = \frac{1 + \text{erf}(\sqrt{\pi} L \langle \mathbf{w}, \mathbf{x} \rangle)}{2}$$



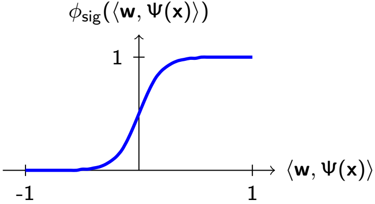
ϕ_{erf} can be written as an infinite-degree polynomial



Unfortunately, bad dependence on L. Can we get a better bound?

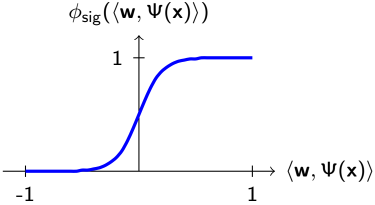
Sigmoid Function

$$\phi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle) = \frac{1}{1 + \exp(-4L \langle \mathbf{w}, \mathbf{x} \rangle)}$$



Sigmoid Function

$$\phi_{\text{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle) = \frac{1}{1 + \exp(-4L \langle \mathbf{w}, \mathbf{x} \rangle)}$$



- ϕ_{sig} is not a polynomial
- However, can be ϵ-approximated by a polynomial with coefficient bound $B \leq \mathcal{O}\left(\exp\left(7L \log\left(\frac{L}{\epsilon}\right)\right)\right)$
  - We use a truncated sum of Chebyshev polynomials
  - Closed-form coefficient bound via tools from complex analysis

Sigmoid Function

Worst-Case Guarantee

Can learn fuzzy halfspace class $\{\mathbf{x} \mapsto \phi_{\mathsf{sig}}(\langle \mathbf{w}, \mathbf{x} \rangle) : \|\mathbf{w}\| = 1\}$ in time/sample complexity $\mathcal{O}(\exp(7L \log(L/\epsilon)))$

Picking ϕ_{sig} is just for the analysis - algorithm is oblivious to ϕ used

Hardness Result

- Better bound? Maybe with some other L-Lipschitz ϕ?
- Proper learning is hard, but here we search for any predictor

Hardness Result

- Better bound? Maybe with some other L-Lipschitz ϕ?
- Proper learning is hard, but here we search for any predictor

Theorem

Can't learn Fuzzy Halfspaces with L-Lipschitz ϕ in poly$(L, 1/\epsilon)$ time.

Proof Idea

Proof by reduction:

- Cryptographic assumption: No poly-time solution to $\tilde{\mathcal{O}}(n^{1.5})$-unique-shortest-vector problem

Proof Idea

Proof by reduction:

- Cryptographic assumption: No poly-time solution to $\tilde{\mathcal{O}}(n^{1.5})$-unique-shortest-vector problem
- $\Rightarrow$ can't PAC-learn intersection of n^ρ halfspaces over $\{-1,+1\}^n$ in poly-time (Klivans and Sherstov, 2006)

Proof Idea

Proof by reduction:

- Cryptographic assumption: No poly-time solution to $\tilde{\mathcal{O}}(n^{1.5})$-unique-shortest-vector problem
- $\Rightarrow$ can't PAC-learn intersection of n^{ρ} halfspaces over $\{-1,+1\}^n$ in poly-time (Klivans and Sherstov, 2006)
- $\Rightarrow$ can't agnostic-PAC-learn single halfspaces over $\{-1,+1\}^n$ in poly-time (otherwise, can use boosting to learn intersections)

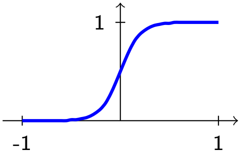
Proof Idea

Proof by reduction:

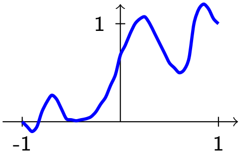
- Cryptographic assumption: No poly-time solution to $\tilde{\mathcal{O}}(n^{1.5})$-unique-shortest-vector problem
- $\Rightarrow$ can't PAC-learn intersection of n^{ρ} halfspaces over $\{-1,+1\}^n$ in poly-time (Klivans and Sherstov, 2006)
- $\Rightarrow$ can't agnostic-PAC-learn single halfspaces over $\{-1,+1\}^n$ in poly-time (otherwise, can use boosting to learn intersections)
- $\Rightarrow$ can't agnostic-PAC-learn fuzzy halfspaces over $\mathbb{R}^n$ in poly-time, when L is polynomially small

Summary

- New technique for learning predictors $\mathbf{x} \mapsto \phi(\langle \mathbf{w}, \mathbf{x} \rangle)$, ϕ possibly non-convex, with the $0 - 1$ loss



- Single algorithm, simultaneously competitive against all ϕ, including optimal one for the data distribution



- In fact, equivalent to standard SVM, but composing our kernel