Risk bounds for classification and regression rules that interpolate

Daniel Hsu

Computer Science Department & Data Science Institute

Columbia University

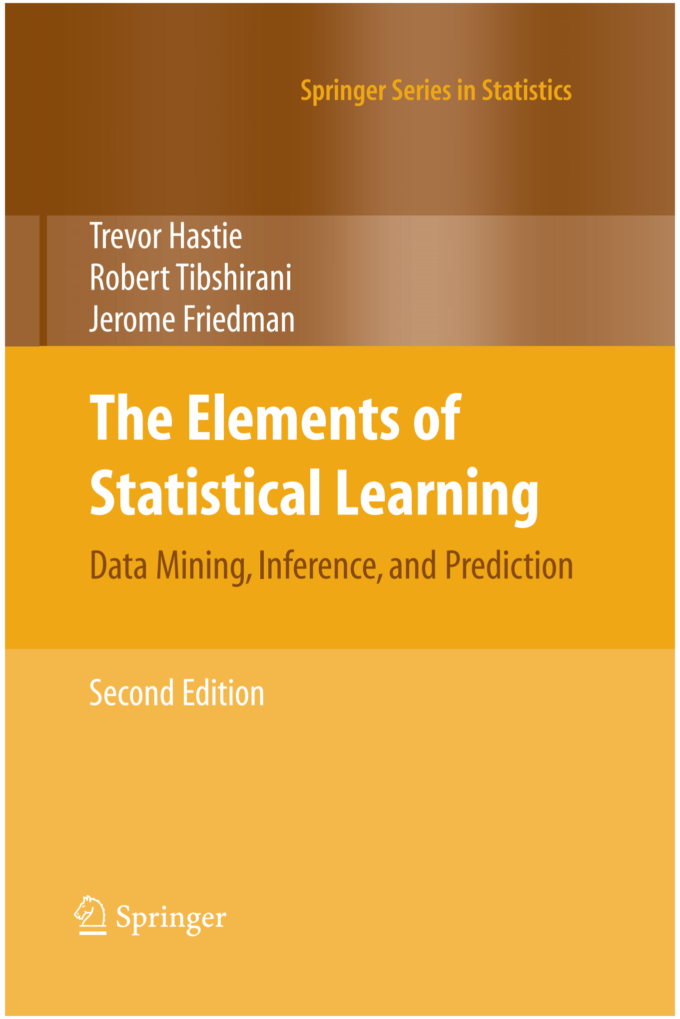
Google Research, 2019 Feb 20

Spoilers

"A model with zero training error is overfit to the training data and will typically generalize poorly."

– Hastie, Tibshirani, & Friedman,
The Elements of Statistical Learning

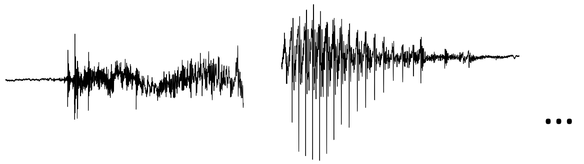
We'll give empirical and theoretical evidence against this conventional wisdom, at least in "modern" settings of machine learning.



Outline

1. Statistical learning setup
2. Empirical observations against the conventional wisdom
3. Risk bounds for rules that interpolate
    - Simplicial interpolation
    - Weighted interpolated nearest neighbor (if time permits)

Supervised learning



/k/ /a/

Training data (labeled examples)
$(x_1, y_1), \ldots, (x_n, y_n)$ from $\mathcal{X} \times \mathcal{Y}$

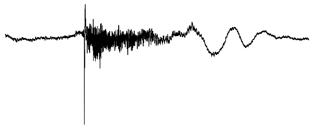
(IID from P)

$w \leftarrow w - \eta \nabla \hat{\mathcal{R}}(w)$

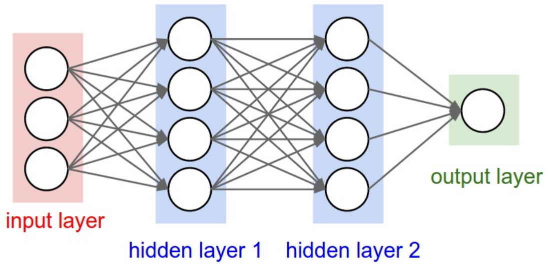
Learning algorithm

Risk: $\mathcal{R}(f) \coloneqq \mathbb{E}[\ell(f(x'), y')]$
where $(x', y') \sim P$

Test point
$x' \in \mathcal{X}$



Prediction function
$\hat{f}: \mathcal{X} \to \mathcal{Y}$



Predicted label
$\hat{f}(x') \in \mathcal{Y}$

/t/

Modern machine learning algorithms

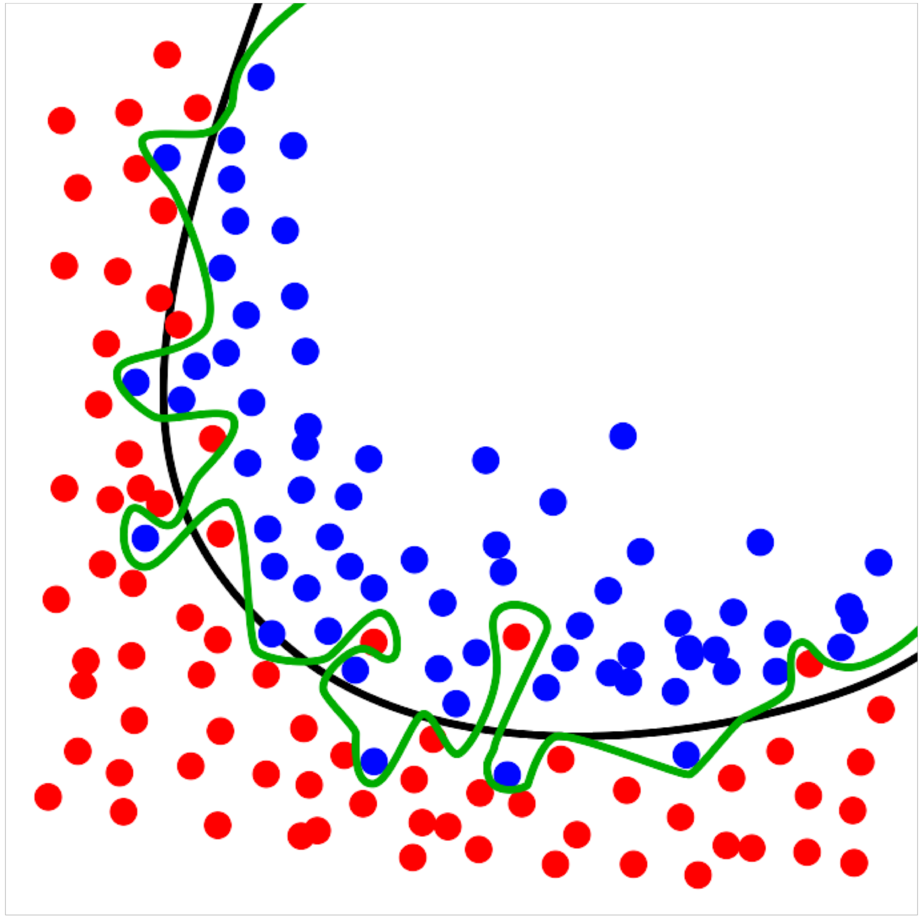
- Choose (parameterized) **function class** $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$
  - E.g., linear functions, polynomials, neural networks with certain architecture
- Use optimization algorithm to (attempt to) minimize **empirical risk**

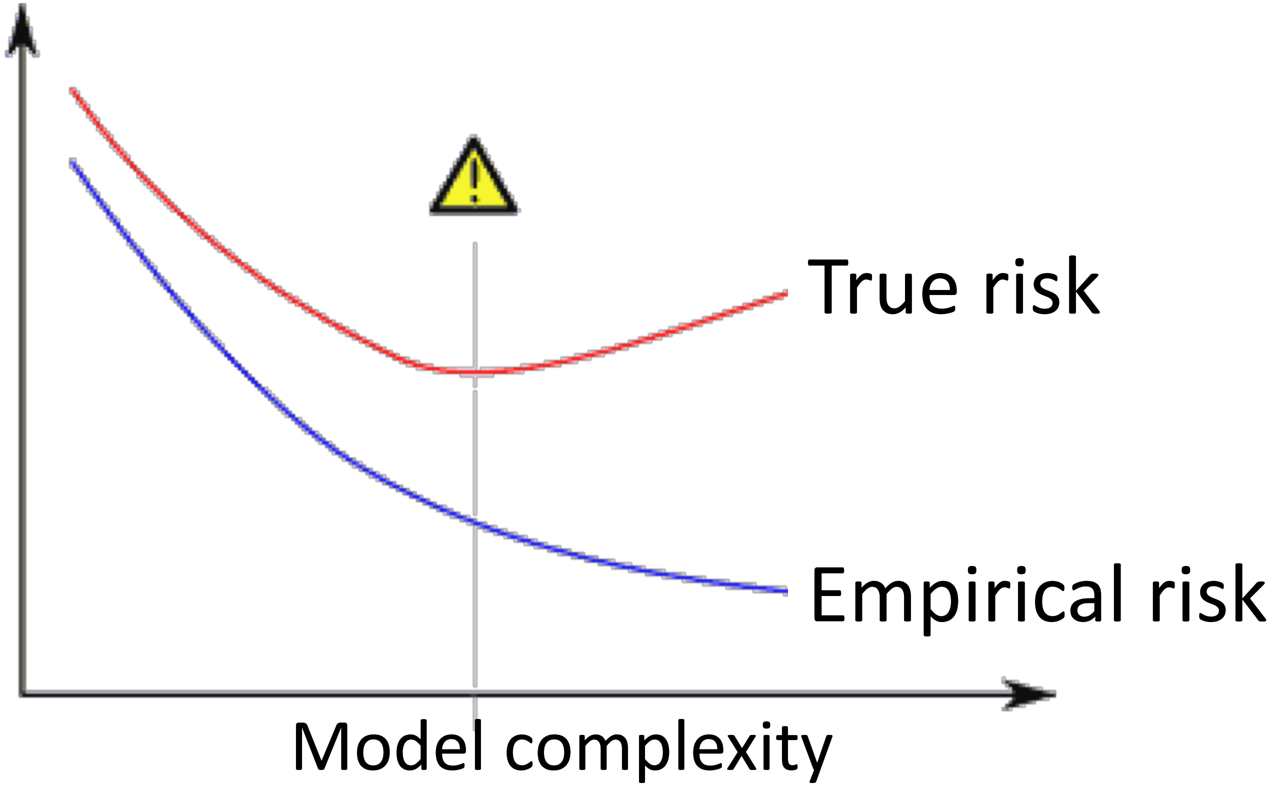
$$\hat{\mathcal{R}}(f) \coloneqq \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$$

(a.k.a. **training error**).

- **But how "big" or "complex" should this function class be?**
(Degree of polynomial, size of neural network architecture, …)

Overfitting





Generalization theory

- **Generalization theory** explains how overfitting can be avoided
- Most basic form:

$$\mathbb{E}\left[\max_{f\in\mathcal{F}} \mathcal{R}(f) - \hat{\mathcal{R}}(f)\right] \lesssim \sqrt{\frac{\text{Complexity}(\mathcal{F})}{n}}$$

- **Complexity of $\boldsymbol{\mathcal{F}}$** can be measured in many ways:
  - Combinatorial parameter (e.g., Vapnik-Chervonenkis dimension)
  - Log-covering number in $L^2(P)$ metric
  - Rademacher complexity (supremum of Rademacher process)
  - Functional / parameter norms (e.g., Reproducing Kernel Hilbert Space norm)
  - …

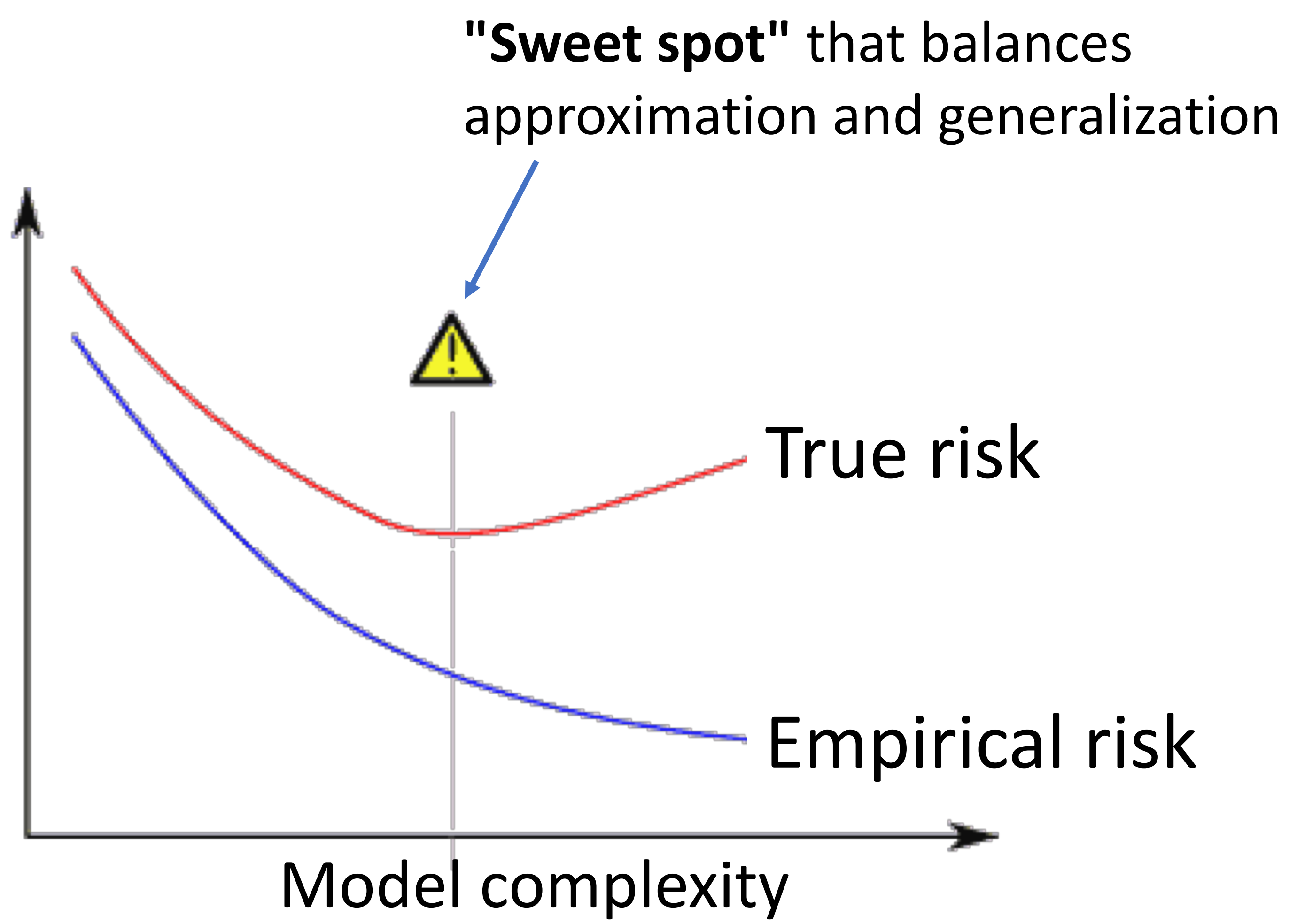
"Classical" risk decomposition

- Let $g^* \in \arg\min_{g:\mathcal{X}\to\mathcal{Y}} \mathcal{R}(g)$ be measurable function of smallest risk
- Let $f^* \in \arg\min_{f\in\mathcal{F}} \mathcal{R}(f)$ be function in $\mathcal{F}$ of smallest risk
- Then:

$$
\begin{aligned}
\mathcal{R}(\hat{f}) = \mathcal{R}(g^*) &+ [\mathcal{R}(f^*) - \mathcal{R}(g^*)] && \text{Approximation} \\
&+ [\hat{\mathcal{R}}(f^*) - \mathcal{R}(f^*)] && \text{Sampling} \\
&+ [\hat{\mathcal{R}}(\hat{f}) - \hat{\mathcal{R}}(f^*)] && \text{Optimization} \\
&+ [\mathcal{R}(\hat{f}) - \hat{\mathcal{R}}(\hat{f})] && \text{Generalization}
\end{aligned}
$$

- Smaller $\mathcal{F}$: larger Approximation term, smaller Generalization term
- Larger $\mathcal{F}$: smaller Approximation term, larger Generalization term

Balancing the two terms...



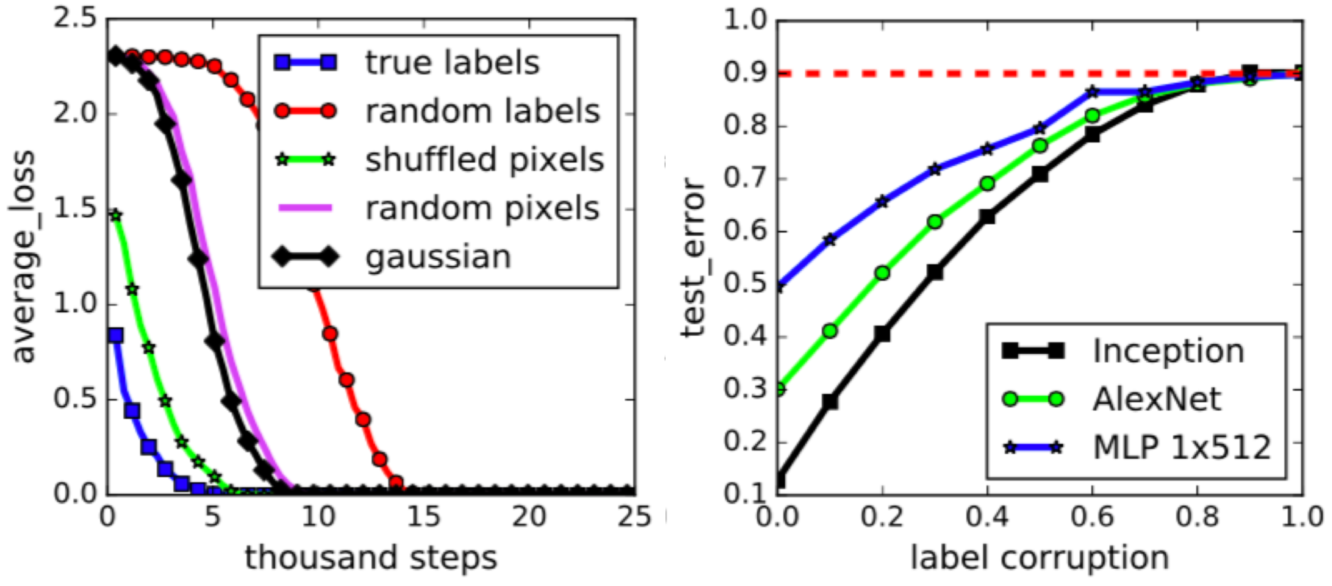
The plot thickens…

Empirical observations raise new questions

Some observations from the field

(Zhang, Bengio, Hardt, Recht, & Vinyals, 2017)

Deep neural networks:
- Can fit any training data.
- Can generalize even when training data has substantial amount of label noise.



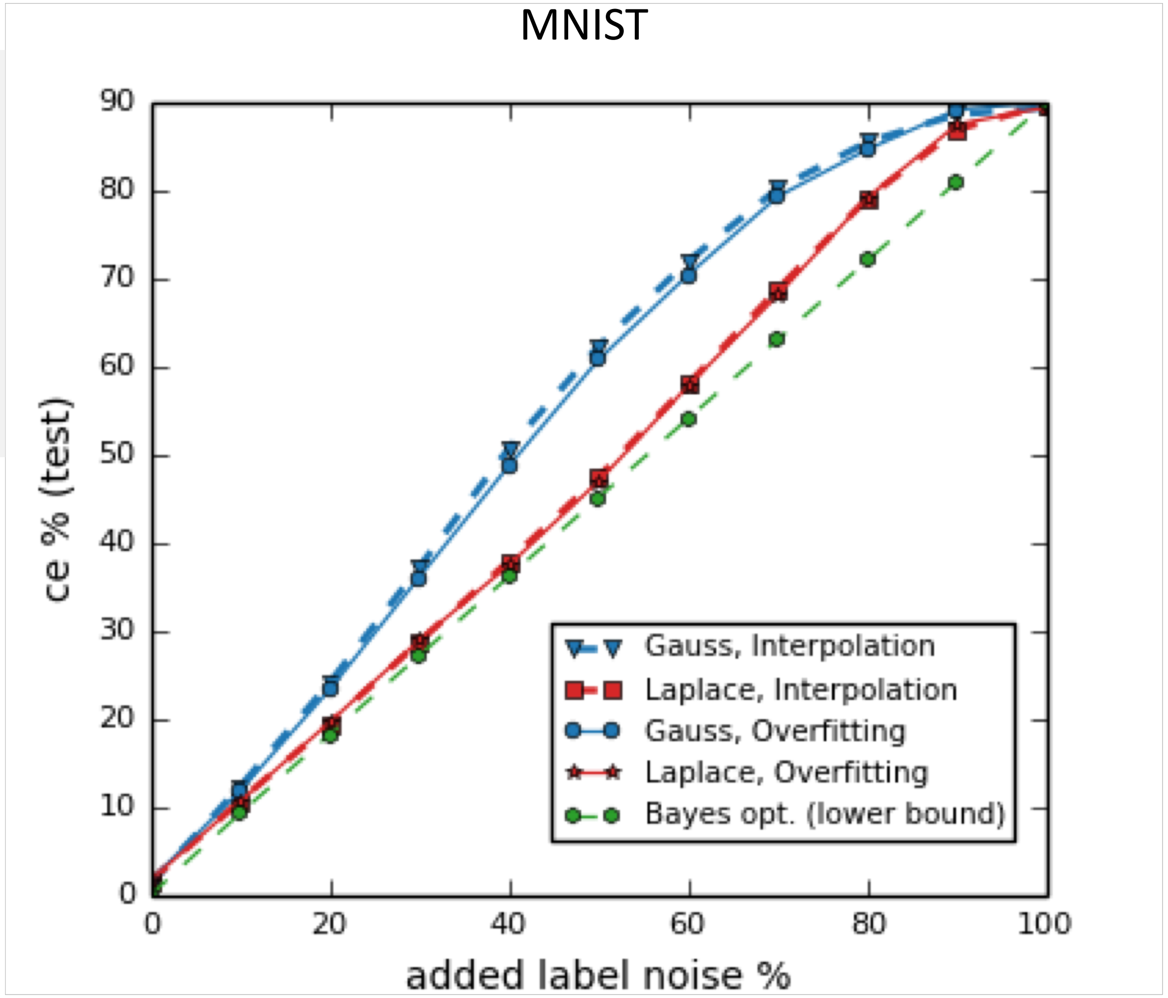
(a) learning curves (c) generalization error growth

More observations from the field

(Belkin, Ma, & Mandal, 2018)

Kernel machines:

- Can fit any training data, given enough time and rich enough feature space.
- Can generalize even when training data has substantial amount of label noise.



Overfitting or perfect fitting?

- Training produces a function $\hat{f}$ that perfectly fits **noisy** training data.
  - $\hat{f}$ is likely a very complex function!
- Yet, test error of $\hat{f}$ is non-trivial: e.g., noise rate + 5%.

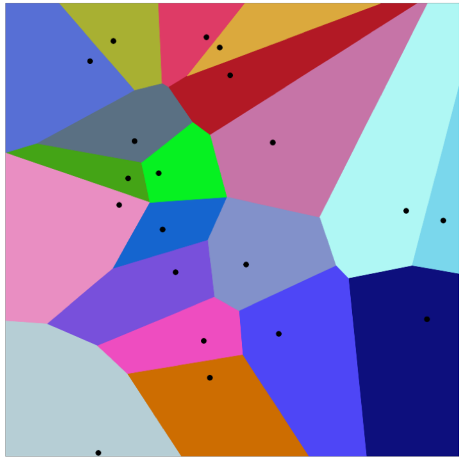
Existing generalization bounds are uninformative for function classes that can interpolate noisy data.

- $\hat{f}$ chosen from class rich enough to express all possible ways to label $\Omega(n)$ training examples.
- Bound **must** exploit specific properties of how $\hat{f}$ is chosen.

Existing theory about local interpolation

Nearest neighbor (Cover & Hart, 1967)

- Predict with label of nearest training example
- Interpolates training data
- Risk $\to 2 \cdot \mathcal{R}(g^*)$ (sort of)



Hilbert kernel (Devroye, Györfi, & Krzyżak, 1998)

- Special kind of smoothing kernel regression (like Shepard's method)
- Interpolates training data
- Consistent, but no convergence rates

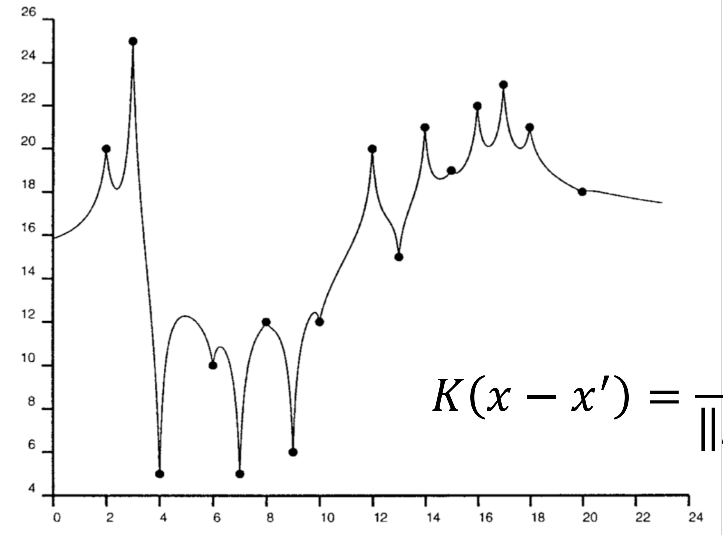


FIG. 2. The Hilbert kernel regression estimate with $a = 1$.

$$K(x - x') = \frac{1}{\|x - x'\|^d}$$

Our goals

- **Counter the "conventional wisdom" re: interpolation**
  Show interpolation methods can be consistent (or almost consistent) for classification & regression problems
- Identify some **useful properties of certain local prediction** methods
- Suggest **connections to practical methods**

New theoretical results

Theoretical analyses of two new interpolation schemes

1. **Simplicial interpolation**
   - Natural linear interpolation based on multivariate triangulation
   - Asymptotic advantages compared to nearest neighbor rule
2. **Weighted & interpolated nearest neighbor (wiNN) method**
   - Consistency + non-asymptotic convergence rates

Joint work with **Misha Belkin** (Ohio State Univ.) & **Partha Mitra** (Cold Spring Harbor Lab.)

Simplicial interpolation

Basic idea

- Construct estimate $\hat{\eta}$ of the **regression function**
$$\eta(x) = \mathbb{E}[\, y' \mid x' = x \,]$$
- Regression function η is minimizer of risk for squared loss
$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

- For **binary classification** $\mathcal{Y} = \{0,1\}$:
  - $\eta(x) = \Pr(y' = 1 \mid x' = x)$
  - **Optimal classifier**: $g^*(x) = \mathbb{I}_{\eta(x) > \frac{1}{2}}$
  - We'll construct **plug-in classifier** $\hat{f}(x) = \mathbb{I}_{\hat{\eta}(x) > \frac{1}{2}}$ based on $\hat{\eta}$

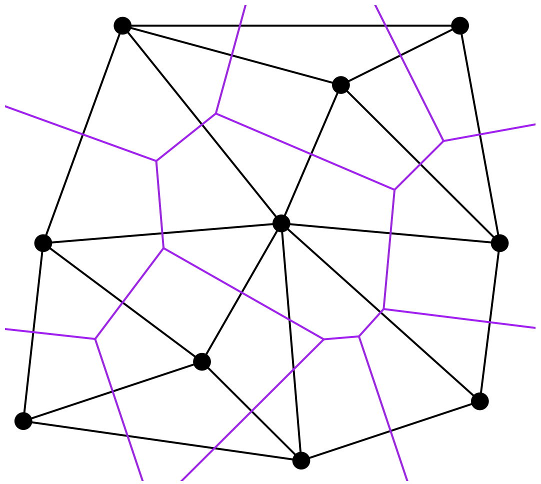
Consistency and convergence rates

Questions of interest:

- What is the (expected) risk of $\hat{f}$ as $n \to \infty$? Is it near optimal ($\mathcal{R}(g^*)$)?
- What what rate (as function of n) does $\mathbb{E}\left[\mathcal{R}\left(\hat{f}\right)\right]$ approach $\mathcal{R}(g^*)$?

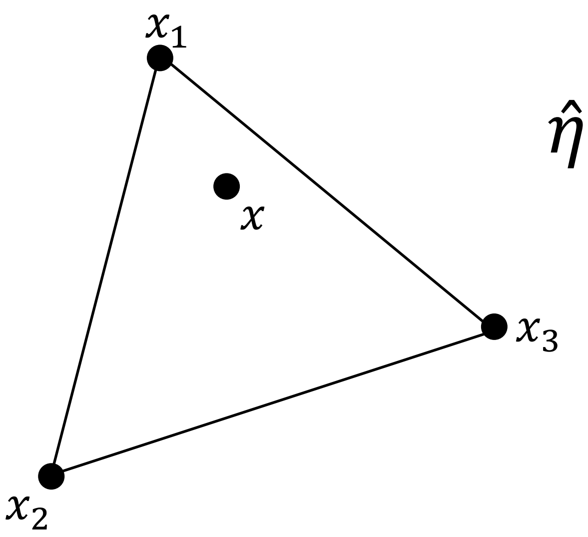
Interpolation via multivariate triangulation

- IID training examples $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^d \times [0,1]$
  - Partition $C := \operatorname{conv}(x_1, \ldots, x_n)$ into simplices with x_i as vertices via Delaunay.
  - Define $\hat{\eta}(x)$ on each simplex by affine interpolation of vertices' labels.
  - Result is piecewise linear on C. (Punt on what happens outside of C.)
- For classification ($y \in \{0,1\}$), let $\hat{f}$ be plug-in classifier based on $\hat{\eta}$.



What happens on a single simplex

- Simplex on $x_1, \dots, x_{d+1}$ with corresponding labels $y_1, \dots, y_{d+1}$
- Test point x in simplex, with barycentric coordinates $(w_1, \dots, w_{d+1})$.
- Linear interpolation at x (i.e., least squares fit, evaluated at x):

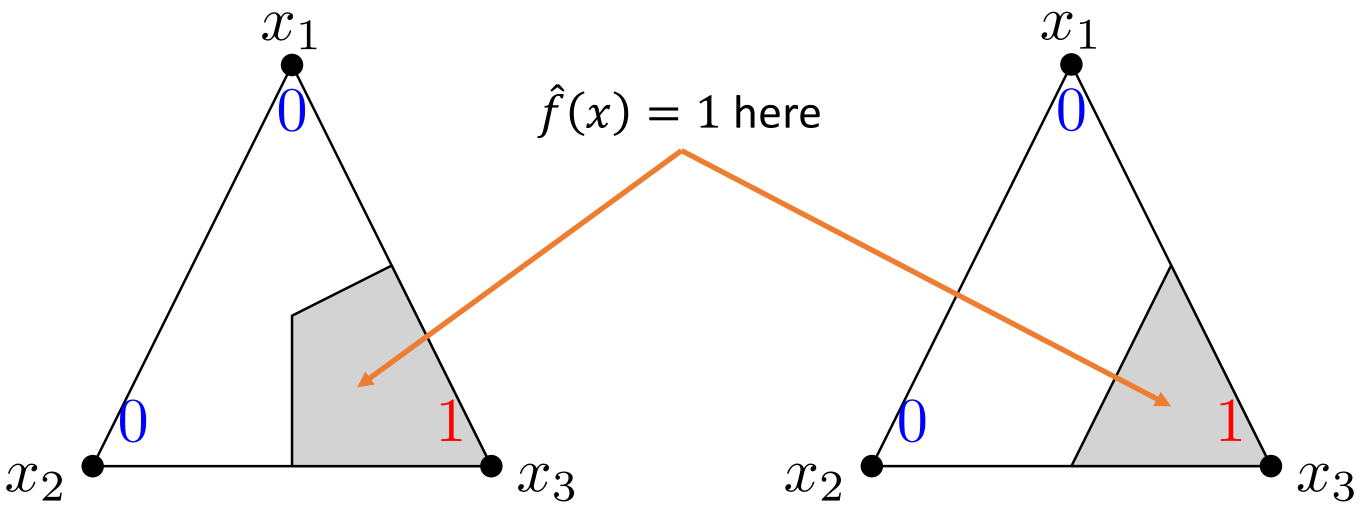


$$\hat{\eta}(x) = \sum_{i=1}^{d+1} w_i y_i$$

Key idea: aggregates information from all vertices to make prediction. (C.f. nearest neighbor rule.)

Comparison to nearest neighbor rule

- Suppose $\eta(x) = \Pr(y = 1 \mid x) < 1/2$ for all points in a simplex
  - Optimal prediction of g^* is 0 for all points in simplex.
- Suppose $y_1 = \cdots = y_d = 0$, but $y_{d+1} = 1$ (due to "label noise")



Nearest neighbor rule

Simplicial interpolation

Effect is exponentially more pronounced in high dimensions!

Asymptotic risk (binary classification)

Theorem: Assume distribution of x' is uniform on some convex set, and η is bounded away from $1/2$. Then simplicial interpolation's plug-in classifier $\hat{f}$ satisfies

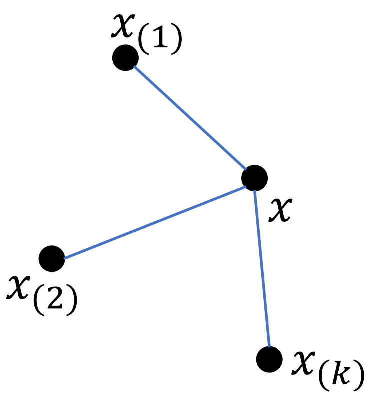
$$\limsup_{n} \mathbb{E}\left[\mathcal{R}(\hat{f})\right] \leq \left(1 + e^{-\Omega(d)}\right) \cdot \mathcal{R}(g^*)$$

- **Near-consistency in high-dimension**
- **C.f. nearest neighbor classifier**: $\limsup_{n} \mathbb{E}\left[\mathcal{R}(\hat{f})\right] \approx 2 \cdot \mathcal{R}(g^*)$
- "Blessing" of dimensionality (with caveat about convergence rate).
- Also have analysis for regression + classification w/o condition on η

Weighted & interpolated NN

Weighted & interpolated NN (wiNN) scheme

- For given test point x, let $x_{(1)}, \dots, x_{(k)}$ be k nearest neighbors in training data, and let $y_{(1)}, \dots, y_{(k)}$ be corresponding labels.



Define

$$\hat{\eta}(x) = \frac{\sum_{i=1}^{k} w(x, x_{(i)})\, y_{(i)}}{\sum_{i=1}^{k} w(x, x_{(i)})}$$

where

$$w(x, x_{(i)}) = \left\| x - x_{(i)} \right\|^{-\delta}, \qquad \delta > 0$$

Interpolation: $\hat{\eta}(x) \to y_i$ as $x \to x_i$

Comparison to Hilbert kernel estimate

Weighted & interpolated NN

$$\hat{\eta}(x) = \frac{\sum_{i=1}^{k} w(x, x_{(i)})\, y_{(i)}}{\sum_{i=1}^{k} w(x, x_{(i)})}$$

$$w(x, x_{(i)}) = \|x - x_{(i)}\|^{-\delta}$$

Our analysis needs $0 < \delta < d/2$

Hilbert kernel (Devroye, Györfi, & Krzyżak, 1998)

$$\hat{\eta}(x) = \frac{\sum_{i=1}^{n} w(x, x_i)\, y_i}{\sum_{i=1}^{n} w(x, x_i)}$$

$$w(x, x_i) = \|x - x_i\|^{-\delta}$$

MUST have $\delta = d$ for consistency

Localization makes it possible to prove non-asymptotic rate.

Convergence rates (regression)

Theorem: Assume distribution of x' is uniform on some compact set satisfying regularity condition, and η is α-Holder smooth.

For appropriate setting of k, wiNN estimate $\hat{\eta}$ satisfies

$$\mathbb{E}[\mathcal{R}(\hat{\eta})] \leq \mathcal{R}(\eta) + O\left(n^{-2\alpha/(2\alpha+d)}\right)$$

- Consistency + optimal rates of convergence for interpolating method.
- Also get consistency and rates for classification.

Conclusions and open problems

1. Interpolation is compatible with good statistical properties
2. Need **good inductive bias**:
   E.g., functions that do local averaging in high-dimensions.

Open problems

- Formally characterize inductive bias of interpolation with **existing methods** (e.g., neural nets, kernel machines, random forests)
  - Srebro: Simplicial interpolation = GD on infinite width ReLU network (d=1)
- Benefits of interpolation?

Acknowledgements

- Collaborators: Misha Belkin and Partha Mitra
- National Science Foundation
- Sloan Foundation
- Simons Institute for the Theory of Computing

Thank you!

arxiv.org/abs/1806.05161