Lecture 14 — Learning Theory (Part 1 of 2)

Alex Schwing and Matus Telgarsky

March 1, 2018

Theory

CS Theory.

- Design and analysis of algorithms.
- Time complexity, space complexity, ...
- Often worst-case.

Theory

CS Theory.

- Design and analysis of algorithms.
- Time complexity, space complexity, ...
- Often worst-case.

ML Theory.

- Design and analysis of *ML* algorithms.
- Time complexity, space complexity,
  sample complexity, *label complexity*, ...
- Often *average-case*.

Standard ML setup.

- Want to do well on some *task*; have some input/output pairs.
- We choose a performance criterion and a family of models.
- We pick a good model wrt the criterion on the data.

Standard ML setup.

- Want to do well on some *task*; have some input/output pairs.
- We choose a performance criterion and a family of models.
- We pick a good model wrt the criterion on the data.

- **Examples**: linear regression, logistic regression, SVM, ...
  sorting, protein folding, ... !
- **Counterexamples** (for now)**:** k-nn, ...

Standard ML setup.

- Want to do well on some *task*; have some input/output pairs.
- We choose a performance criterion and a family of models.
- We pick a good model wrt the criterion on the data.

- **Examples**: linear regression, logistic regression, SVM, ... sorting, protein folding, ... !
- **Counterexamples** (for now)**:** k-nn, ...

Formal questions.

- **Representation/Approximation.** The limitations of our model choice.
- **Optimization.** Searching for the best model.
- **Generalization.** Gap between training and testing errors.

Why?

Why?

- Helps us reason about machine learning.

Why?

- Helps us reason about machine learning.
- Helps us improve/make algorithms.

Why?

- Helps us reason about machine learning.
- Helps us improve/make algorithms.
- We are curious.

Why?

- Helps us reason about machine learning.
- Helps us improve/make algorithms.
- We are curious.
- Math is comfy.

Representation/Approximation.

[Questions so far?]

Representation/Approximation.

[Questions so far?]

Representation failures.

- Linear functions can fail on even some simple problems.

Representation successes.

- Polynomial Kernel SVM, RBF Kernel SVM, 2-layer neural nets can fit any continuous function.

Linear functions do not suffice.

Theorem (Minsky-Papert, '69)**.** Consider the 4-point dataset where the corners of the square $\{\pm 1, \pm 1\}$ are labeled with their product. On this data, every linear classifier makes at least 1 error.

Linear functions do not suffice.

Theorem (Minsky-Papert, '69)**.** Consider the 4-point dataset where the corners of the square $\{\pm 1, \pm 1\}$ are labeled with their product. On this data, every linear classifier makes at least 1 error.

- **Picture "proof".** [Must pass between two positives and one negative, fail on other negative.]

Linear functions do not suffice.

Theorem (Minsky-Papert, '69)**.** Consider the 4-point dataset where the corners of the square $\{\pm 1, \pm 1\}$ are labeled with their product. On this data, every linear classifier makes at least 1 error.

- **Picture "proof".** [Must pass between two positives and one negative, fail on other negative.]
- **Algebraic proof.**
  Consider any halfspace $H := \{x \in R^2 : a^\top x + b \geq 0\}$.
  - If $\{u, -u\} \in H$ for some u, then $b \geq 0$:

    $$\begin{aligned} a^\top u + b \geq 0 \quad \wedge \quad a^\top(-u) + b \geq 0 \quad &\Longrightarrow \quad a^\top(u - u) + 2b \geq 0 \\ &\Longrightarrow \quad b \geq 0. \end{aligned}$$

  - If some $v \notin H$ and $b \geq 0$, then $-v \in H$:

    $$a^\top v + b < 0 \Longrightarrow a^\top(-v) - b > 0 \Longrightarrow a^\top(-v) + b > 2b \geq 0.$$

  So for any (a, b), at least one of the two plusses are wrong, or one of the minuses are wrong.

3-layer networks approximate continuous functions.

Theorem. For any continuous $f : [0,1]^d \to \mathbb{R}$ and any $\epsilon > 0$, there exists a 3-layer network g with

$$\int_{[0,1]^d} |f(x) - g(x)| \, \mathrm{d}x \leq \epsilon.$$

3-layer networks approximate continuous functions.

Theorem. For any continuous $f : [0,1]^d \to \mathbb{R}$ and any $\epsilon > 0$, there exists a 3-layer network g with

$$\int_{[0,1]^d} |f(x) - g(x)| \, \mathrm{d}x \leq \epsilon.$$

- **Proof** (sketch)**.** First approximate f with a step (piecewise constant) function; then approximate each step function with a 2-layer network (details in lecture).

Optimization

[The second of three analysis topics; questions so far?]

Optimization

[The second of three analysis topics; questions so far?]

- **The optimization question:**
  How do we choose a function which fits the data?

Optimization

[The second of three analysis topics; questions so far?]

- **The optimization question:**
  How do we choose a function which fits the data?

- In more detail, **(Regularized) Emprical Risk Minimization (ERM)**: We have data $((x_i, y_i))_{i=1}^n$,
  predictors $\mathcal{F}$,
  a regularizer Reg,
  and a performance criterion ℓ;
  We seek to optimize

$$\arg\min_{f\in\mathcal{F}} \widehat{\text{Risk}}(f) + \text{Reg}(f) \qquad \text{where} \quad \widehat{\text{Risk}}(f) := \frac{1}{n}\sum_{i=1}^{n} \ell(f, x_i, y_i).$$

Optimization

[The second of three analysis topics; questions so far?]

- **The optimization question:**
  How do we choose a function which fits the data?

- In more detail, **(Regularized) Emprical Risk Minimization (ERM)**: We have data $((x_i, y_i))_{i=1}^n$,
  predictors $\mathcal{F}$,
  a regularizer Reg,
  and a performance criterion ℓ;
  We seek to optimize

$$\arg\min_{f \in \mathcal{F}} \widehat{\text{Risk}}(f) + \text{Reg}(f) \qquad \text{where} \quad \widehat{\text{Risk}}(f) := \frac{1}{n} \sum_{i=1}^{n} \ell(f, x_i, y_i).$$

- **Example:** Ridge regression; $f_w(x) := w^\top x$ for some w, and

$$\ell(f_w, x, y) = (w^\top x - y)^2/2, \qquad \text{Reg}(f) := \frac{\lambda}{2} \|w\|^2.$$

  We can find w with gradient descent (or "closed form" $(X^\top X + \lambda I)^{-1} X^\top y$).

Ridge regression in more detail.

Recall the *Ridge Regression Estimator* in matrix/vector form:

$$\hat{w} := \arg\min_{w \in \mathbb{R}^d} \frac{1}{2n}\|Xw - y\|_2^2 + \frac{\lambda}{2}\|w\|_2^2.$$

Let's consider our three analysis questions.
Also: what is the role of λ?

Ridge regression in more detail.

Recall the *Ridge Regression Estimator* in matrix/vector form:

$$\hat{w} := \arg\min_{w \in \mathbb{R}^d} \frac{1}{2n}\|Xw - y\|_2^2 + \frac{\lambda}{2}\|w\|_2^2.$$

Let's consider our three analysis questions.
Also: what is the role of λ?

- **Representation:** not just a linear predictor, moreover $\left\{x \mapsto w^\top x : \|w\| \leq \sqrt{1/\lambda}\right\}$ since

$$\|\hat{w}\|^2 \leq \frac{1}{n\lambda}\|X\hat{w} - y\|^2 + \|\hat{w}\|^2 \leq \frac{1}{n\lambda}\|X0 - y\|^2 + \|0\|^2 \leq \frac{1}{\lambda}.$$

Ridge regression in more detail.

Recall the *Ridge Regression Estimator* in matrix/vector form:

$$\hat{w} := \arg\min_{w \in \mathbb{R}^d} \frac{1}{2n} \|Xw - y\|_2^2 + \frac{\lambda}{2} \|w\|_2^2.$$

Let's consider our three analysis questions.
Also: what is the role of λ?

- **Representation:** not just a linear predictor, moreover $\left\{ x \mapsto w^\top x : \|w\| \le \sqrt{1/\lambda} \right\}$ since

$$\|\hat{w}\|^2 \le \frac{1}{n\lambda} \|X\hat{w} - y\|^2 + \|\hat{w}\|^2 \le \frac{1}{n\lambda} \|X0 - y\|^2 + \|0\|^2 \le \frac{1}{\lambda}.$$

- **Optimization:** as discussed in class, to achieve accuracy ϵ, gradient descent needs $\frac{\sigma_{\max}(X)+\lambda}{\sigma_{\min}(X)+\lambda} \ln(1/\epsilon)$ iterations.

Ridge regression in more detail.

Recall the *Ridge Regression Estimator* in matrix/vector form:

$$\hat{w} := \arg\min_{w \in \mathbb{R}^d} \frac{1}{2n}\|Xw - y\|_2^2 + \frac{\lambda}{2}\|w\|_2^2.$$

Let's consider our three analysis questions.
Also: what is the role of λ?

- **Representation:** not just a linear predictor, moreover $\left\{x \mapsto w^\top x : \|w\| \le \sqrt{1/\lambda}\right\}$ since

$$\|\hat{w}\|^2 \le \frac{1}{n\lambda}\|X\hat{w} - y\|^2 + \|\hat{w}\|^2 \le \frac{1}{n\lambda}\|X0 - y\|^2 + \|0\|^2 \le \frac{1}{\lambda}.$$

- **Optimization:** as discussed in class, to achieve accuracy ϵ, gradient descent needs $\frac{\sigma_{\max}(X)+\lambda}{\sigma_{\min}(X)+\lambda}\ln(1/\epsilon)$ iterations.

- **Generalization:** *coming up next!*

Overfitting and generalization.

[Final analysis topic; questions so far?]

Overfitting and generalization.

[Final analysis topic; questions so far?]

We train on some data because that is what we have.

But what we *want* is good *future* performance.

- **Generalizing** means similar past and future performance.
- **Overfitting** means (vastly) better past performance.

Overfitting and generalization.

[Final analysis topic; questions so far?]
We train on some data because that is what we have.
But what we *want* is good *future* performance.

- **Generalizing** means similar past and future performance.
- **Overfitting** means (vastly) better past performance.

Example: Suppose data $((x_i, y_i))_{i=1}^n$, with x_i random, and

$$y_i := \bar{w}^\top x_i + \xi_i$$

with independent and zero mean $(\xi_i)_{i=1}^n$.

- Algo 1: ordinary least squares.
- Algo 2: fit an n-degree polynomial.

Overfitting and generalization.

[Final analysis topic; questions so far?]
We train on some data because that is what we have.
But what we *want* is good *future* performance.

- **Generalizing** means similar past and future performance.
- **Overfitting** means (vastly) better past performance.

Example: Suppose data $((x_i, y_i))_{i=1}^n$, with x_i random, and

$$y_i := \bar{w}^\top x_i + \xi_i$$

with independent and zero mean $(\xi_i)_{i=1}^n$.

- Algo 1: ordinary least squares.
- Algo 2: fit an n-degree polynomial.

Second should *overfit*.
To formalize this, need a *model* for unseen data.

Models for data.

Learning theory provides *many* candidate models.

- **Statistical** setting: training data and future data drawn IID (idependent and identically distributed) from some distribution. (An *average case* setting.)
- **Online** setting: an adversary constructs examples with full knowledge of what we are doing. (A *worst case* setting.)

Which one is more realistic?

Models for data.

Learning theory provides *many* candidate models.

- **Statistical** setting: training data and future data drawn IID (idependent and identically distributed) from some distribution. (An *average case* setting.)
- **Online** setting: an adversary constructs examples with full knowledge of what we are doing. (A *worst case* setting.)

Which one is more realistic?

Example: spam filtering.

- On the one hand, spammers observe what google does, try to break its detection.
- On the other hand, they don't know exactly what google is doing.

Models for data.

Learning theory provides *many* candidate models.

- **Statistical** setting: training data and future data drawn IID (idependent and identically distributed) from some distribution. (An *average case* setting.)
- **Online** setting: an adversary constructs examples with full knowledge of what we are doing. (A *worst case* setting.)

Which one is more realistic?

Example: spam filtering.

- On the one hand, spammers observe what google does, try to break its detection.
- On the other hand, they don't know exactly what google is doing.

If each morning google re-trains on the last 30 days of data, assuming it is IID, is it so bad? (Mixture of both settings.)

The *Statistical Learning Theory* setting.

- We receive n examples IID from some underlying distribution.
- We would like to do well according to some performance criterion *in expectation*.

Example. Least squares: we receive $((x_i, y_i))_{i=1}^n$, we select $\hat{w}$, and the quantity we want to minimize is

$$\mathbb{E}(\hat{w}^\top x - y)^2,$$

but we can't compute this!
(We only have a finite sample.)

Empirical Risk Minimization and generalization.

We have advocated

(Regularized) Empirical Risk Minimization (ERM):

We choose a function class $\mathcal{F}$, a loss function ℓ, a regularization scheme Reg, and approximately optimize

$$\operatorname{argmin}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f, x_i, y_i) + \operatorname{Reg}(f).$$

Why should this work?

Empirical Risk Minimization and generalization.

We have advocated
(Regularized) Empirical Risk Minimization (ERM):
We choose a function class $\mathcal{F}$, a loss function ℓ, a regularization scheme Reg, and approximately optimize

$$\text{argmin}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f, x_i, y_i) + \text{Reg}(f).$$

Why should this work?

- Vague intuition: law of large numbers (LLN).
- Rigorous version: suppose $\hat{f}$ selected *without* using $((x_i, y_i))_{i=1}^{n}$. If $((x_i, y_i))_{i=1}^{n}$ are IID, *then so are* $(\ell(\hat{f}, x_i, y_i))_{i=1}^{n}$, and by LLN

$$\frac{1}{n} \sum_{i=1}^{n} \ell(\hat{f}, x_i, y_i) \to \mathbb{E}(\ell(\hat{f}, X, Y)) \qquad \text{as } n \to \infty.$$

Empirical Risk Minimization and generalization.

We have advocated
(Regularized) Empirical Risk Minimization (ERM):
We choose a function class $\mathcal{F}$, a loss function ℓ, a regularization scheme Reg, and approximately optimize

$$\operatorname{argmin}_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(f, x_i, y_i) + \operatorname{Reg}(f).$$

Why should this work?

- Vague intuition: law of large numbers (LLN).
- Rigorous version: suppose $\hat{f}$ selected *without* using $((x_i, y_i))_{i=1}^n$. If $((x_i, y_i))_{i=1}^n$ are IID, *then so are* $(\ell(\hat{f}, x_i, y_i))_{i=1}^n$, and by LLN

$$\frac{1}{n}\sum_{i=1}^{n} \ell(\hat{f}, x_i, y_i) \to \mathbb{E}(\ell(\hat{f}, X, Y)) \qquad \text{as } n \to \infty.$$

Two issues.

- We want a bound for finite n.
- We used $((x_i, y_i))_{i=1}^n$ to select $\hat{f}$!

Issue #1: bounds for finite samples.

The easiest tool here is **Hoeffding's inequality**.

Theorem (Hoeffding's inequality)**.** Suppose each draw from the distribution lies in the interval $[a, b]$. With probability at least $1 - \delta$ over an iid draw of $(z_i)_{i=1}^n$,

$$\mathbb{E}Z \leq \frac{1}{n}\sum_{i=1}^{n} z_i + (b-a)\sqrt{\frac{\ln(1/\delta)}{2n}}.$$

- **Remark** (on terminology)**.** This is sometimes called a *concentration inequality*, or a *deviation bound*.

Issue #1: bounds for finite samples.

The easiest tool here is **Hoeffding's inequality**.

Theorem (Hoeffding's inequality)**.** Suppose each draw from the distribution lies in the interval $[a, b]$. With probability at least $1 - \delta$ over an iid draw of $(z_i)_{i=1}^n$,

$$\mathbb{E}Z \le \frac{1}{n}\sum_{i=1}^{n} z_i + (b-a)\sqrt{\frac{\ln(1/\delta)}{2n}}.$$

- **Remark** (on terminology)**.** This is sometimes called a *concentration inequality*, or a *deviation bound*.
- **Example.** Consider *fixed* $\hat{f}$, and binary loss $\ell(f, x, y) := \mathbf{1}[f(x) \neq y] \in [0, 1]$. Then, with probability at least $(1 - \delta)$ over an IID draw $((x_i, y_i))_{i=1}^n$,

$$\Pr\left[\hat{f}(X) \neq Y\right] = \mathbb{E}\mathbb{1}\left[\hat{f}(X) \neq Y\right] \le \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left[\hat{f}(x_i) \neq y_i\right] + \sqrt{\frac{\ln(1/\delta)}{2n}}$$

- **Remark** (scaling)**:** Randomly receive n \$$d$ bills; confidence interval scales with $d\sqrt{\ln(1/\delta)/2n}$.

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Intuition: larger function classes overfit more.

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Intuition: larger function classes overfit more.

Rigorous bound: Suppose we select f from functions $\mathcal{F}$ using data $((x_i, y_i))_{i=1}^n$. Then (with probability at least $1 - \delta$),

$$\mathbb{E}\ell(f, X, Y) \leq \frac{1}{n}\sum_{i=1}^{n}\ell(f, x_i, y_i) + \widetilde{\mathcal{O}}\left(\sqrt{\frac{\text{Complexity}(\mathcal{F}) + \ln(1/\delta)}{n}}\right).$$

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Intuition: larger function classes overfit more.

Rigorous bound: Suppose we select f from functions $\mathcal{F}$ using data $((x_i, y_i))_{i=1}^n$. Then (with probability at least $1 - \delta$),

$$\mathbb{E}\ell(f, X, Y) \le \frac{1}{n}\sum_{i=1}^n \ell(f, x_i, y_i) + \widetilde{\mathcal{O}}\left(\sqrt{\frac{\text{Complexity}(\mathcal{F}) + \ln(1/\delta)}{n}}\right).$$

Example.

- Linear classification $w \mapsto \text{sgn}(w^\top x)$ with $w \in R^d$ has

$$\begin{aligned}\Pr\left[\text{sgn}(w^\top x) \neq y)\right] &\le \frac{1}{n}\sum_{i=1}^n \mathbb{1}\left[\text{sgn}(w^\top x_i) \neq y_i)\right] \\ &\quad + \widetilde{\mathcal{O}}\left(\sqrt{d + \ln(1/\delta)}n\right).\end{aligned}$$

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Intuition: larger function classes overfit more.

Rigorous bound: Suppose we select f from functions $\mathcal{F}$ using data $((x_i, y_i))_{i=1}^n$. Then (with probability at least $1 - \delta$),

$$\mathbb{E}\ell(f, X, Y) \leq \frac{1}{n}\sum_{i=1}^n \ell(f, x_i, y_i) + \widetilde{\mathcal{O}}\left(\sqrt{\frac{\text{Complexity}(\mathcal{F}) + \ln(1/\delta)}{n}}\right).$$

Example.

- Linear classification $w \mapsto \text{sgn}(w^\top x)$ with $w \in R^d$ has

$$\Pr\left[\text{sgn}(w^\top x) \neq y)\right] \leq \frac{1}{n}\sum_{i=1}^n \mathbb{1}\left[\text{sgn}(w^\top x_i) \neq y_i)\right] + \widetilde{\mathcal{O}}\left(\sqrt{d + \ln(1/\delta)}n\right).$$

  - **Remark:** How to minimize rhs?

Issue #2: *uniform* deviations.

Previous bounds were for a *fixed function* $\hat{f}$.

Let's take a step back and use our intuition.

Intuition: larger function classes overfit more.

Rigorous bound: Suppose we select f from functions $\mathcal{F}$ using data $((x_i, y_i))_{i=1}^n$. Then (with probability at least $1 - \delta$),

$$\mathbb{E}\ell(f, X, Y) \leq \frac{1}{n}\sum_{i=1}^{n} \ell(f, x_i, y_i) + \widetilde{\mathcal{O}}\left(\sqrt{\frac{\text{Complexity}(\mathcal{F}) + \ln(1/\delta)}{n}}\right).$$

Example.

- Linear classification $w \mapsto \text{sgn}(w^\top x)$ with $w \in R^d$ has

$$\Pr\left[\text{sgn}(w^\top x) \neq y)\right] \leq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\left[\text{sgn}(w^\top x_i) \neq y_i)\right] + \widetilde{\mathcal{O}}\left(\sqrt{d + \ln(1/\delta)}n\right).$$

  - **Remark:** How to minimize rhs?
    Logistic upper bounds...

Simple example: finite classes

When $|\mathcal{F}| < \infty$, can do Complexity$(\mathcal{F}) \leq \ln|\mathcal{F}|$.

Theorem. With probability at least $1 - \delta$,

$$\Pr[f(X) \neq Y] \leq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[f(x_i) \neq y_i] + \sqrt{\frac{\ln(|\mathcal{F}|) + \ln(1/\delta)}{2n}}.$$

Simple example: finite classes

When $|\mathcal{F}| < \infty$, can do Complexity$(\mathcal{F}) \leq \ln |\mathcal{F}|$.
Theorem. With probability at least $1 - \delta$,

$$\Pr[f(X) \neq Y] \leq \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[f(x_i) \neq y_i] + \sqrt{\frac{\ln(|\mathcal{F}|) + \ln(1/\delta)}{2n}}.$$

- **Proof.** Define $\epsilon := \sqrt{\ln |F|/\delta / 2n}$ and events

$$E_j := \left[\Pr[f_j(X) \neq Y] > \epsilon + \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[f_j(x_i) \neq y_i] \right].$$

By Hoeffding, $\Pr[E_j] \leq \delta/|F|$, and by union bound

$$\Pr[\cup_j E_j] \leq \sum_j \Pr[E_j] \leq \delta.$$

Where to go from here?

Have much more sophisticated bounds of the form

$$\text{Risk}(f) \leq \widehat{\text{Risk}}(f) + \widetilde{\mathcal{O}}\left(\sqrt{\frac{\text{Complexity}(\mathcal{F}) + \ln(1/\delta)}{n}}\right),$$

where:

- $\text{Risk}(f) = \Pr[f(X) \neq Y]$, $\mathcal{F}$ is ReLU networks with p parameters and L layers, $\text{Complexity}(\mathcal{F}) = \tilde{\mathcal{O}}(pL)$.
- $\text{Risk}(f)$ is least squares risk, $\mathcal{F} := \left\{w \in \mathbb{R}^d : \|w\| \leq \sqrt{1/\lambda}\right\}$ (as in Ridge regression), $\text{Complexity}(\mathcal{F}) = 1/\lambda$.

Summary

- **ML Theory** is design and analysis of ML algorithms.

Summary

- **ML Theory** is design and analysis of ML algorithms.
- **Standard setup:** we fit a model to data according to some performance criterion.
  - **Representation:** is our model powerful enough to capture this phenomenon?
  - **Optimization:** can we (efficiently?) fit our model to data?
  - **Generalization:** does our model perform well on unseen data?

Summary

- **ML Theory** is design and analysis of ML algorithms.
- **Standard setup:** we fit a model to data according to some performance criterion.
  - **Representation:** is our model powerful enough to capture this phenomenon?
  - **Optimization:** can we (efficiently?) fit our model to data?
  - **Generalization:** does our model perform well on unseen data?
- **Generalization/overfitting** requires a model of *unseen* data.
  - **Today** we sketched the *statistical learning theory setting*.
  - **Next time** we'll go into it in more detail.