Introduction to Statistical Learning Theory

David Rosenberg

New York University

October 29, 2016

What types of problems are we solving?

- In data science problems, we generally need to:
  - Make a decision
  - Take an action
  - Produce some output
- Have some evaluation criterion

Actions

Definition

An *action* is the generic term for what is produced by our system.

Examples of Actions

- Produce a 0/1 classification [classical ML]
- Reject hypothesis that $\theta = 0$ [classical Statistics]
- Written English text [speech recognition]
- Probability that a picture contains an animal [computer vision]
- What's an action for predicting where a storm will be in 3 hours?
- What's an action for automated driving?

Evaluation Criterion

Decision theory is about finding "optimal" actions, under various definitions of optimality.

Examples of Evaluation Criteria

- Is classification correct?
- Does text transcription exactly match the spoken words?
    - Should we give partial credit? How?
- Is probability "well-calibrated"?

Real Life: Formalizing a Business Problem

- First two steps to formalizing a problem:
    1. Define the *action space* (i.e. the set of possible actions)
    2. Specify the evaluation criterion.
- Finding *the right formalization* can be an interesting challenge
- Formalization may evolve gradually, as you understand the problem better

Inputs

Most problems have an extra piece, going by various names:

- Inputs [ML]
- Covariates [Statistics]
- Side Information [Various settings]

Examples of Inputs

- A picture
- A storm's historical location and other weather data
- A search query

Output / Outcomes

Inputs often paired with *outputs* or *outcomes*

Examples of outputs / outcomes

- Whether or not the picture actually contains an animal
- The storm's location one hour after query
- Which, if any, of suggested the URLs were selected

Typical Sequence of Events

Many problem domains can be formalized as follows:

1. Observe input x.
2. Take action a.
3. Observe outcome y.
4. Evaluate action in relation to the outcome: $\ell(a, y)$.

Note

- Outcome y is often **independent** of action a
- But this is **not always the case**:
  - search result ranking
  - automated driving

Formalization: The Spaces

The Three Spaces:

- Input space: $\mathcal{X}$
- Action space: $\mathcal{A}$
- Outcome space: $\mathcal{Y}$

Concept check:

- What are the spaces for linear regression?
- What are the spaces for logistic regression?
- What are the spaces for a support vector machine?

Some Formalization

The Spaces

- $\mathcal{X}$: input space
- $\mathcal{Y}$: output space
- $\mathcal{A}$: action space

Decision Function

A **decision function** gets input $x \in \mathcal{X}$ and produces an action $a \in \mathcal{A}$:

$$\begin{aligned} f: \quad \mathcal{X} &\to \mathcal{A} \\ x &\mapsto f(x) \end{aligned}$$

Loss Function

A **loss function** evaluates an action in the context of the output y.

$$\begin{aligned} \ell: \quad \mathcal{A} \times \mathcal{Y} &\to \mathbf{R} \\ (a, y) &\mapsto \ell(a, y) \end{aligned}$$

Real Life: Formalizing a "Data Science" Problem

- First two steps to formalizing a problem:
  1. Define the *action space* (i.e. the set of possible actions)
  2. Specify the evaluation criterion.
- When a "stakeholder" asks the data scientist to solve a problem, she
  - may have an opinion on what the action space should be, and
  - hopefully has an opinion on the evaluation criterion, but
  - she really cares about your **producing a "good" decision function.**
- Typical sequence:
  1. Stakeholder presents problem to data scientist
  2. Data scientist produces decision function
  3. Engineer deploys "industrial strength" version of decision function

Evaluating a Decision Function

- Loss function ℓ evaluates a single action
- How to evaluate the decision function as a whole?
- We will use the standard **statistical learning theory** framework.

A Simplifying Assumption

- Assume action has no effect on the output
  - includes all traditional prediction problems
  - what about stock market prediction?
  - what about stock market investing?
- What about fancier problems where this does not hold?
  - often can be reformulated or "reduced" to problems where it does hold
  - e.g. see literature on **contextual bandit problems**

Setup for Statistical Learning Theory

- Assume there is a **data generating distribution** $P_{\mathcal{X}\times\mathcal{Y}}$.
- All input/output pairs (x,y) are generated i.i.d. from $P_{\mathcal{X}\times\mathcal{Y}}$.
- Want decision function $f(x)$ that generally "does well" (small loss):

$$\ell(f(x),y) \text{ is small}$$

- How can we formalize this?

The Risk Functional

Definition

The **risk** of a decision function $f : \mathcal{X} \to \mathcal{A}$ is

$$R(f) = \mathbb{E}\ell(f(x), y).$$

In words, it's the **expected loss** of f, where the expectation is over $(x, y) \sim P_{\mathcal{X} \times \mathcal{Y}}$.

Risk function cannot be computed

Since we don't know $P_{\mathcal{X} \times \mathcal{Y}}$, we cannot compute the expectation.
But we can estimate it...

The Bayes Decision Function

Definition

A **Bayes decision function** $f^*: \mathcal{X} \to \mathcal{A}$ is a function that achieves the *minimal risk* among all possible functions:

$$f^* = \arg\min_f R(f),$$

where the minimum is taken over all measurable functions from $\mathcal{X}$ to $\mathcal{A}$. The risk of a Bayes decision function is called the **Bayes risk**.

- A Bayes decision function is often called the "**target function**", since it's what we would ultimately like to produce as our decision function.

Example 1: Least Squares Regression

- spaces: $\mathcal{A} = \mathcal{Y} = \mathbf{R}$
- square loss:

$$\ell(a, y) = \frac{1}{2}(a - y)^2$$

- mean square risk:

$$\begin{aligned} R(f) &= \frac{1}{2}\mathbb{E}\left[(f(x) - y)^2\right] \\ (\text{homework} \implies) &= \frac{1}{2}\mathbb{E}\left[(f(x) - \mathbb{E}[y|x])^2\right] + \frac{1}{2}\mathbb{E}\left[(y - \mathbb{E}[y|x])^2\right] \end{aligned}$$

- target function:

$$f^*(x) = \mathbb{E}[y|x]$$

Example 2: Multiclass Classification

- spaces: $\mathcal{A} = \mathcal{Y} = \{0, 1, \ldots, K-1\}$
- 0-1 loss:
$$\ell(a, y) = 1(a \neq y) := \begin{cases} 1 & \text{if } a \neq y \\ 0 & \text{otherwise.} \end{cases}$$
- risk is misclassification error rate
$$\begin{aligned} R(f) &= \mathbb{E}\left[1(f(x) \neq y)\right] \\ &= \mathbb{P}\left(f(x) \neq y\right) \end{aligned}$$
- target function is the assignment to the most likely class
$$f^*(x) = \underset{1 \leqslant k \leqslant K}{\arg\max}\, \mathbb{P}(y = k \mid x)$$

But we can't compute the risk!

- Can't compute $R(f) = \mathbb{E}\ell(f(x), y)$ because we **don't know** $P_{\mathcal{X}\times\mathcal{Y}}$.
- One thing we can do in ML/statistics/data science is

assume we have sample data.

Let $\mathcal{D}_n = ((x_1, y_1), \ldots, (x_n, y_n))$ be drawn i.i.d. from $\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$.

- Let's draw some inspiration from the Strong Law of Large Numbers: If $z, z_1, \ldots, z_n$ are i.i.d. with expected value $\mathbb{E}z$, then

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} z_i = \mathbb{E}z,$$

with probability 1.

The Empirical Risk Functional

Let $\mathcal{D}_n = ((x_1, y_1), \ldots, (x_n, y_n))$ be drawn i.i.d. from $\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$.

Definition

The **empirical risk** of $f : \mathcal{X} \to \mathcal{A}$ with respect to $\mathcal{D}_n$ is

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

By the Strong Law of Large Numbers,

$$\lim_{n\to\infty} \hat{R}_n(f) = R(f),$$

almost surely.

That's a start...

Empirical Risk Minimization

We want risk minimizer, is empirical risk minimizer close enough?

Definition

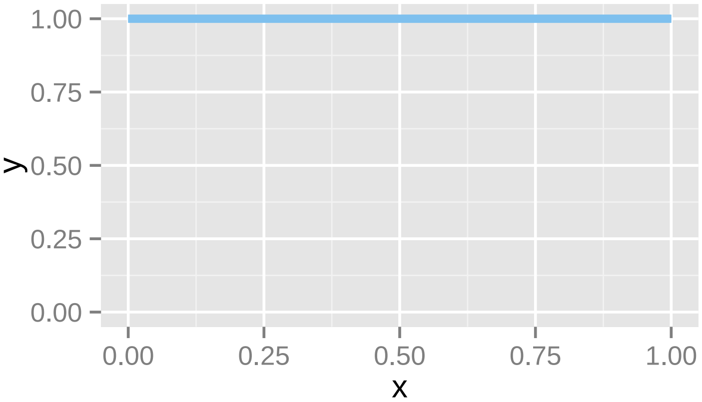
A function $\hat{f}$ is an **empirical risk minimizer** if

$$\hat{f} = \arg\min_{f} \hat{R}_n(f),$$

where the minimum is taken over all [measurable] functions.

Empirical Risk Minimization

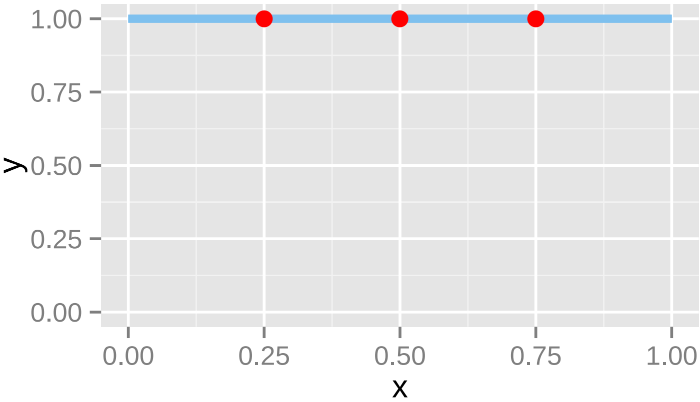
$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).



$\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$.

Empirical Risk Minimization

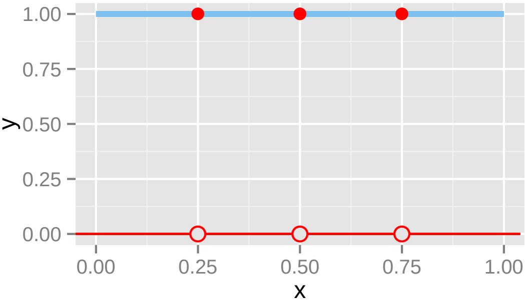
$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).



A sample of size 3 from $\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$.

Empirical Risk Minimization

$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).

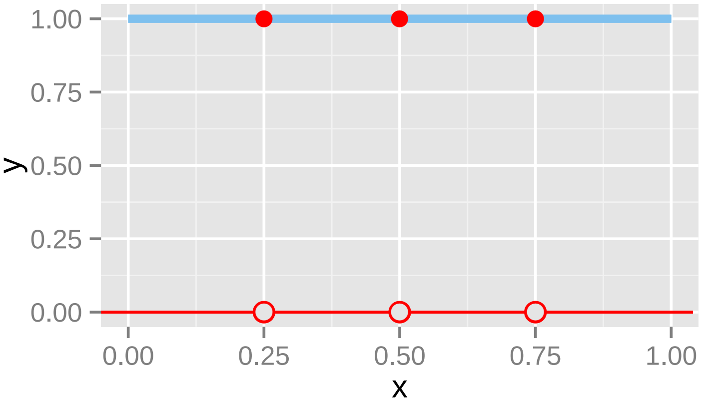


A proposed decision function:

$$\hat{f}(x) = 1(x \in \{0.25, 0.5, 0.75\}) = \begin{cases} 1 & \text{if } x \in \{0.25, .5, .75\} \\ 0 & \text{otherwise} \end{cases}$$

Empirical Risk Minimization

$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).



Under square loss or 0/1 loss: $\hat{f}$ has Empirical Risk = 0 and Risk = 1.

Empirical Risk Minimization

- ERM led to a function f that just memorized the data.
- How to spread information or "**generalize**" from training inputs to new inputs?
- Need to smooth things out somehow...
    - A lot of modeling is about spreading and extrapolating information from one part of the input space $\mathcal{X}$ into unobserved parts of the space.
- One approach: "Constrained ERM"
    - Instead of minimizing empirical risk over all decision functions,
    - constrain to a particular subset, called a **hypothesis space**.

Hypothesis Spaces

Definition

A **hypothesis space** $\mathcal{F}$ is a set of functions mapping $\mathcal{X} \to \mathcal{A}$.

- It is the collection of decision functions we are considering.

Usually $\mathcal{F}$ is a much smaller than the set of all possible functions $\mathcal{X} \to \mathcal{A}$.

Want Hypothesis Space that...

- Includes only those functions that have desired "smoothness"
- Easy to work with

Example hypothesis spaces?

Math tip: $\mathcal{A}^{\mathcal{X}}$ is common shorthand for the set of all functions mapping $\mathcal{X} \to \mathcal{A}$.

Constrained Empirical Risk Minimization

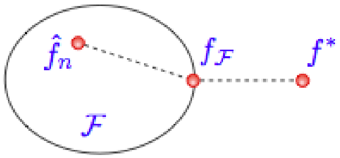
- Hypothesis space $\mathcal{F}$, a set of functions mapping $\mathcal{X} \to \mathcal{A}$
- **Empirical risk minimizer** (ERM) in $\mathcal{F}$ is

$$\hat{f}_n = \underset{f \in \mathcal{F}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i).$$

- **Risk minimizer** in $\mathcal{F}$ is $f_{\mathcal{F}}^* \in \mathcal{F}$, where

$$f_{\mathcal{F}}^* = \underset{f \in \mathcal{F}}{\arg\min} \mathbb{E}\ell(f(x), y).$$

Error Decomposition



$$
\begin{aligned}
f^* &= \arg\min_f \mathbb{E}\ell(f(x),y) \\
f_{\mathcal{F}} &= \arg\min_{f\in\mathcal{F}} \mathbb{E}\ell(f(y),y)) \\
\hat{f}_n &= \arg\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n}\ell(f(x_i),y_i)
\end{aligned}
$$

- **Approximation Error** (of $\mathcal{F}$) $= R(f_{\mathcal{F}}) - R(f^*)$
- **Estimation error** (of $\hat{f}_n$ in $\mathcal{F}$) $= R(\hat{f}_n) - R(f_{\mathcal{F}})$

Figure from Sasha Rakhlin's MLSS Lectures (2012): http://yosinski.com/mlss12/MLSS-2012-Rakhlin-Statistical-Learning-Theory/

Error Decomposition

Definition

The **excess risk** of f is the amount by which the risk of f exceeds the Bayes risk

$$\textbf{Excess Risk}(\hat{f}_n) = R(\hat{f}_n) - R(f^*) = \underbrace{R(\hat{f}_n) - R(f_{\mathcal{F}})}_{\text{estimation error}} + \underbrace{R(f_{\mathcal{F}}) - R(f^*)}_{\text{approximation error}} .$$

This is a more general expression of the bias/variance tradeoff for mean squared error:

- Approximation error = "bias"
- Estimation error = "variance"

Approximation Error

- Approximation error is a property of the class $\mathcal{F}$
- It's our penalty for restricting to $\mathcal{F}$ rather than considering all measurable functions
  - Approximation error is the minimum risk possible with $\mathcal{F}$ (even with infinite training data)
- *Bigger* $\mathcal{F}$ mean *smaller* approximation error.

Estimation Error

- *Estimation error*: The performance hit for choosing f using finite training data
  - *Equivalently*: It's the hit for not knowing the true risk, but only the empirical risk.
- *Smaller* $\mathcal{F}$ means *smaller* estimation error.
- *Under typical conditions:* "With infinite training data, estimation error goes to zero."
  - Infinite training data solves the *statistical* problem, which is not knowing the true risk.]

ERM Overview

- Given a loss function $\ell : \mathcal{A} \times \mathcal{Y} \to \mathbf{R}$.
- Choose hypothesis space $\mathcal{F}$.
- Use an algorithm (an optimization method) to find $\hat{f}_n \in \mathcal{F}$ minimizing the empirical risk:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i).$$

- (So, $\hat{R}(\hat{f}) = \min_{f \in \mathcal{F}} \hat{R}(f)$).
- Data scientist's job: choose $\mathcal{F}$ to optimally balance between approximation and estimation error.

Optimization Error

- Does unlimited data solve our problems?
- There's still the *algorithmic* problem of *finding* $\hat{f}_n \in \mathcal{F}$.
- For nice choices of loss functions and classes $\mathcal{F}$, the algorithmic problem can be solved (to any desired accuracy).
    - Takes time! Is it worth it?
- **Optimization error:** If $\tilde{f}_n$ is the function our optimization method returns, and $\hat{f}_n$ is the empirical risk minimizer, then the optimization error is $R(\tilde{f}_n) - R(\hat{f}_n)$
- NOTE: May have $R(\tilde{f}_n) < R(\hat{f}_n)$, since $\hat{f}_n$ may overfit more than $\tilde{f}_n$!