Introduction to Statistical Learning Theory

David S. Rosenberg

New York University

January 23, 2018

Decision Theory: High Level View

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 [image captioning, speech recognition, machine translation]
- What's an action for predicting where a storm will be in 3 hours?
- What's an action for a self-driving car?

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?
- How far is the storm from the prediction location? [for point prediction]
- How likely is the storm's location under the prediction? [for density prediction]

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.
- 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]

Examples of Inputs

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

“Outcomes” or “Output” or “Label”

Inputs often paired with *outputs* or *outcomes or labels*

Examples of outcomes/outputs/labels

- 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 (via a **loss function**$\ell(a, y)$)

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
- $\mathcal{Y}$: outcome space
- $\mathcal{A}$: action space

Prediction Function (or "decision function")

A **prediction function** (or **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 outcome y.

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

Evaluating a Prediction Function

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

Statistical Learning Theory

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
  - see literature on **reinforcement learning**

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 prediction function $f(x)$ that generally "does well on average":

$$\ell(f(x), y) \text{ is usually small, in some sense}$$

- How can we formalize this?

The Risk Functional

Definition

The **risk** of a prediction 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 on a new example (x, y) drawn randomly from $P_{\mathcal{X}\times\mathcal{Y}}$.

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 Prediction Function

Definition

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

$$f^* \in \underset{f}{\arg\min}\, R(f),$$

where the minimum is taken over all functions from $\mathcal{X}$ to $\mathcal{A}$.

- The risk of a Bayes prediction function is called the **Bayes risk**.
- A Bayes prediction function is often called the "**target function**", since it's the best prediction function we can possibly produce.

Example: Least Squares Regression

- Spaces: $\mathcal{A} = \mathcal{Y} = \mathrm{R}$
- Square loss:

$$\ell(a, y) = (a - y)^2$$

- Risk:

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

- So Bayes prediction function is

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

Example 2: Multiclass Classification

- Spaces: $\mathcal{A} = \mathcal{Y} = \{1, \ldots, k\}$
- 0-1 loss:

$$\ell(a, y) = 1(a \neq y) := \begin{cases} 1 & \text{if } a \neq y \\ 0 & \text{otherwise.} \end{cases}$$

- Risk:

$$\begin{aligned} R(f) &= \mathbb{E}\left[1(f(x) \neq y)\right] \quad = 0 \cdot \mathbb{P}(f(x) = y) + 1 \cdot \mathbb{P}(f(x) \neq y) \\ &= \mathbb{P}(f(x) \neq y), \end{aligned}$$

which is just the misclassification error rate.

- Bayes prediction function is just the assignment to the most likely class:

$$f^*(x) \in \underset{1 \leqslant c \leqslant k}{\arg\max}\, \mathbb{P}(y = c \mid x)$$

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 $P_{\mathcal{X}\times\mathcal{Y}}$.

- 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

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

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.
- But we want to find the f that **minimizes** $R(f)$ - will minimizing $\hat{R}_n(f)$ be good enough?

Empirical Risk Minimization

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

Definition

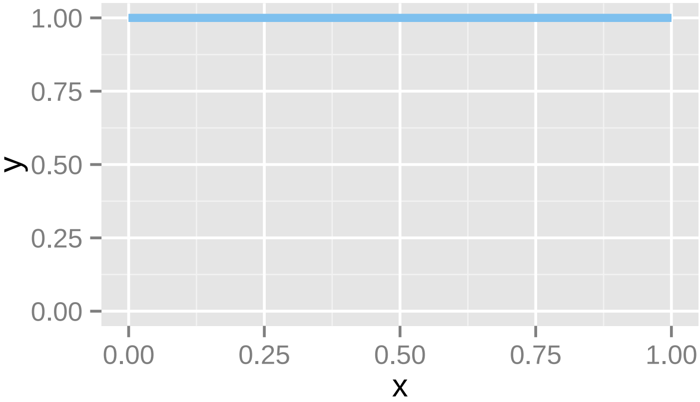
A function $\hat{f}$ is an **empirical risk minimizer** if

$$\hat{f} \in \operatorname*{arg\,min}_{f} \hat{R}_n(f),$$

where the minimum is taken over all functions.

Empirical Risk Minimization

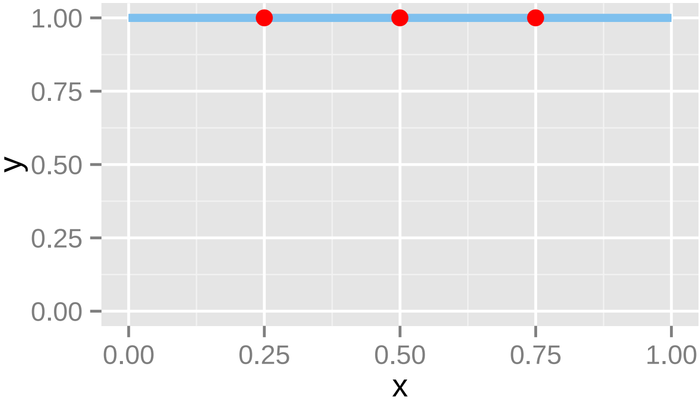
$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).



$P_{\mathcal{X} \times \mathcal{Y}}$.

Empirical Risk Minimization

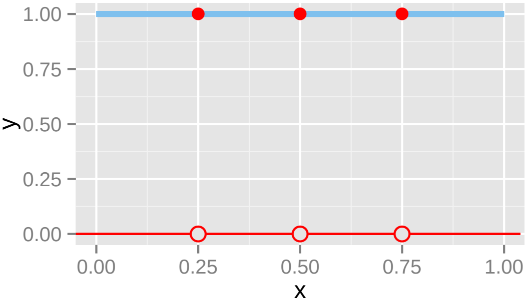
$P_{\mathcal{X}} = \text{Uniform}[0,1]$, $Y \equiv 1$ (i.e. Y is always 1).



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

Empirical Risk Minimization

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

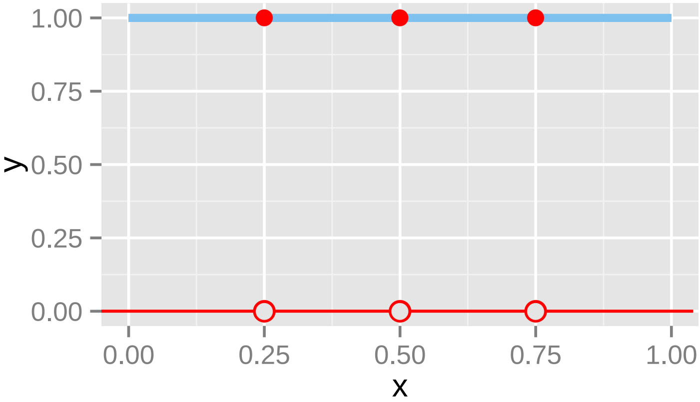


A proposed prediction function:

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

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 prediction 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 prediction functions we are choosing from.

Want Hypothesis Space that...

- Includes only those functions that have desired "regularity"
  - e.g. smoothness, simplicity
- Easy to work with

Example hypothesis spaces?

Constrained Empirical Risk Minimization

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

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

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

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