Overview of statistical learning theory

Daniel Hsu

Columbia TRIPODS Bootcamp

Statistical model for machine learning

Basic goal of machine learning

Goal: Predict outcome y from set of possible outcomes $\mathcal{Y}$, on the basis of observation x from feature space $\mathcal{X}$.

- *Examples*:
  1. $x =$ email message, $y =$ spam or ham
  2. $x =$ image of handwritten digit, $y =$ digit
  3. $x =$ medical test results, $y =$ disease status

Basic goal of machine learning

Goal: Predict outcome y from set of possible outcomes $\mathcal{Y}$, on the basis of observation x from feature space $\mathcal{X}$.

- *Examples*:
  1. $x =$ email message, $y =$ spam or ham
  2. $x =$ image of handwritten digit, $y =$ digit
  3. $x =$ medical test results, $y =$ disease status

Learning algorithm:

- Receives *training data*

  $$(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$

  and returns a prediction function

  $$\hat{f} \colon \mathcal{X} \to \mathcal{Y}.$$

- On (new) *test example* (x, y), predict $\hat{f}(x)$.

Assessing the quality of predictions

Loss function: $\ell \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

- Prediction is $\hat{y}$, true outcome is y.
- Loss $\ell(\hat{y}, y)$ measures how bad $\hat{y}$ is as a prediction of y.

Assessing the quality of predictions

Loss function: $\ell\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

- Prediction is $\hat{y}$, true outcome is y.
- Loss $\ell(\hat{y}, y)$ measures how bad $\hat{y}$ is as a prediction of y.

Examples:

1. Zero-one loss:

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

2. Squared loss (for $\mathcal{Y} \subseteq \mathbb{R}$):

$$\ell(\hat{y}, y) = (\hat{y} - y)^2.$$

Why is this possible?

- Only input provided to learning algorithm is training data

$$(x_1, y_1), \ldots, (x_n, y_n).$$

- To be useful, training data must be related to test example

$$(x, y).$$

How can we formalize this?

Basic statistical model for data

IID model of data

Regard training data and test example as *independent and identically distributed* $(\mathcal{X} \times \mathcal{Y})$-valued random variables:

$$(X_1, Y_1), \ldots, (X_n, Y_n), (X, Y) \sim_{\mathrm{iid}} P.$$

Can use tools from probability to study behavior of learning algorithms under this model.

Risk

Loss $\ell(f(X), Y)$ is random, so study average-case performance.

Risk of a prediction function f, defined by

$$\mathcal{R}(f) = \mathbb{E}[\ell(f(X), Y)],$$

where expectation is taken with respect to test example (X, Y).

Risk

Loss $\ell(f(X), Y)$ is random, so study average-case performance.

Risk of a prediction function f, defined by

$$\mathcal{R}(f) = \mathbb{E}[\ell(f(X), Y)],$$

where expectation is taken with respect to test example (X, Y).

Examples:

1. *Mean squared error*: $\ell =$ squared loss,

$$\mathcal{R}(f) = \mathbb{E}[(f(X) - Y)^2].$$

2. *Error rate*: $\ell =$ zero-one loss,

$$\mathcal{R}(f) = \mathbb{P}(f(X) \neq Y).$$

Comparison to classical statistics

How (classical) learning theory differs from classical statistics:

- Typically, data distribution P is allowed to be arbitrary.
  - E.g., not from a parametric family $\{P_\theta : \theta \in \Theta\}$.
- Focus on prediction rather than general estimation of P.

Now: Much overlap between machine learning and statistics.

Inductive bias

Is predictability enough?

Requirements for learning:

- Relationship between training data and test example
  - Formalized by iid model for data.
- Relationship between Y and X.
  - Example: X and Y are non-trivially correlated.

Is this enough?

No free lunch

For any $n \leq \frac{|\mathcal{X}|}{2}$ and any learning algorithm, there is a distribution, from which the n training data and test example are drawn iid, s.t.:

1. There is a function $f^*\colon \mathcal{X} \to \mathcal{Y}$ with

$$\mathbb{P}(f^*(X) \neq Y) = 0.$$

2. The learning algorithm returns a function $\hat{f}\colon \mathcal{X} \to \mathcal{Y}$ with

$$\mathbb{P}(\hat{f}(X) \neq Y) \geq \frac{1}{4}.$$

How to pay for lunch

Must make *some* assumption about learning problem in order for learning algorithm to work well.

- Called *inductive bias* of the learning algorithm.

How to pay for lunch

Must make *some* assumption about learning problem in order for learning algorithm to work well.

- Called *inductive bias* of the learning algorithm.

Common approach:

- Assume there is a good prediction function in a restricted function class $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$.
- Goal: find $\hat{f} \colon \mathcal{X} \to \mathcal{Y}$ with small *excess risk*

$$\mathcal{R}(\hat{f}) - \min_{f \in \mathcal{F}} \mathcal{R}(f)$$

  either in expectation or with high probability over random draw of training data.

Examples

Example #1: Threshold functions

$\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0, 1\}$.

- *Threshold functions*

$$\mathcal{F} = \{f_\theta : \theta \in \mathbb{R}\}$$

where f_θ is defined by

$$f_\theta(x) = \mathbb{1}\{x > \theta\} = \begin{cases} 0 & \text{if } x \leq \theta, \\ 1 & \text{if } x > \theta. \end{cases}$$

Example #1: Threshold functions

$\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = \{0, 1\}$.

- *Threshold functions*

$$\mathcal{F} = \{f_\theta : \theta \in \mathbb{R}\}$$

where f_θ is defined by

$$f_\theta(x) = \mathbb{1}\{x > \theta\} = \begin{cases} 0 & \text{if } x \leq \theta, \\ 1 & \text{if } x > \theta. \end{cases}$$

- Learning algorithm:
  1. Sort training examples by x_i-value.
  2. Consider candidate threshold values that are (i) equal to x_i-values, (ii) equal to values midway between consecutive but non-equal x_i-values, and (iii) a value smaller than all x_i-values.
  3. Among candidate thresholds, pick $\hat{\theta}$ such that $f_{\hat{\theta}}$ incorrectly classifies the smallest number of examples in training data.

Example #2: Linear functions

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$, $\ell =$ squared loss.

- *Linear functions*
$$\mathcal{F} = \{f_w : w \in \mathbb{R}^d\}$$
where f_w is defined by
$$f_w(x) = w^\mathsf{T} x.$$

Example #2: Linear functions

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$, $\ell =$ squared loss.

- *Linear functions*

  $$\mathcal{F} = \{f_w : w \in \mathbb{R}^d\}$$

  where f_w is defined by

  $$f_w(x) = w^\mathsf{T} x.$$

- Learning algorithm ("Ordinary Least Squares"):
  - Return a solution $\hat{w}$ to system of linear equations given by

  $$\left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^\mathsf{T}\right) w = \frac{1}{n}\sum_{i=1}^{n} y_i x_i.$$

Example #3: Linear classifiers

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, +1\}$.

- *Linear classifiers*

$$\mathcal{F} = \{f_w : w \in \mathbb{R}^d\}$$

where f_w is defined by

$$f_w(x) = \operatorname{sign}(w^\mathsf{T} x) = \begin{cases} -1 & \text{if } w^\mathsf{T} x \leq 0, \\ +1 & \text{if } w^\mathsf{T} x > 0. \end{cases}$$

Example #3: Linear classifiers

$\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, +1\}$.

- *Linear classifiers*

$$\mathcal{F} = \{f_w : w \in \mathbb{R}^d\}$$

where f_w is defined by

$$f_w(x) = \operatorname{sign}(w^\intercal x) = \begin{cases} -1 & \text{if } w^\intercal x \leq 0, \\ +1 & \text{if } w^\intercal x > 0. \end{cases}$$

- Learning algorithm ("Support Vector Machine"):
  - Return solution $\hat{w}$ to following optimization problem:

$$\min_{w \in \mathbb{R}^d} \frac{\lambda}{2} \|w\|_2^2 + \frac{1}{n} \sum_{i=1}^n [1 - y_i w^\intercal x_i]_+ .$$

Over-fitting and generalization

Over-fitting

Over-fitting:

Phenomenon where learning algorithm returns $\hat{f}$ that “fits” training data well, but does not give accurate predictions on test examples.

Over-fitting

Over-fitting:

Phenomenon where learning algorithm returns $\hat{f}$ that "fits" training data well, but does not give accurate predictions on test examples.

- *Empirical risk* of f (on training data $(X_1, Y_1), \ldots, (X_n, Y_n)$):

$$\mathcal{R}_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i).$$

- *Over-fitting*: $\mathcal{R}_n(\hat{f})$ small, but $\mathcal{R}(\hat{f})$ large.

Generalization

How to avoid over-fitting

"Theorem": $\mathcal{R}(\hat{f}) - \mathcal{R}_n(\hat{f})$ is likely to be small, if learning algorithm chooses $\hat{f}$ from $\mathcal{F}$ that is "not too rich" relative to n.

- $\Rightarrow$ Observed performance on training data (i.e., empirical risk) *generalizes* to expected performance on test example (i.e., risk).
- Justifies learning algorithms based on minimizing empirical risk.

Other issues

Risk decomposition

$$
\begin{aligned}
\mathcal{R}(\hat{f}) = {} & \inf_{g\colon \mathcal{X}\to\mathcal{Y}} \mathcal{R}(g) && \text{(inherent unpredictability)} \\
& + \inf_{f\in\mathcal{F}} \mathcal{R}(f) - \inf_{g\colon \mathcal{X}\to\mathcal{Y}} \mathcal{R}(g) && \text{(approximation gap)} \\
& + \inf_{f\in\mathcal{F}} \mathcal{R}_n(f) - \inf_{f\in\mathcal{F}} \mathcal{R}(f) && \text{(estimation gap)} \\
& + \mathcal{R}_n(\hat{f}) - \inf_{f\in\mathcal{F}} \mathcal{R}_n(f) && \text{(optimization gap)} \\
& + \mathcal{R}(\hat{f}) - \mathcal{R}_n(\hat{f}). && \text{(more estimation gap)}
\end{aligned}
$$

Risk decomposition

$$\begin{aligned}
\mathcal{R}(\hat{f}) = &\inf_{g\colon \mathcal{X}\to\mathcal{Y}} \mathcal{R}(g) && \text{(inherent unpredictability)} \\
&+ \inf_{f\in\mathcal{F}} \mathcal{R}(f) - \inf_{g\colon \mathcal{X}\to\mathcal{Y}} \mathcal{R}(g) && \text{(approximation gap)} \\
&+ \inf_{f\in\mathcal{F}} \mathcal{R}_n(f) - \inf_{f\in\mathcal{F}} \mathcal{R}(f) && \text{(estimation gap)} \\
&+ \mathcal{R}_n(\hat{f}) - \inf_{f\in\mathcal{F}} \mathcal{R}_n(f) && \text{(optimization gap)} \\
&+ \mathcal{R}(\hat{f}) - \mathcal{R}_n(\hat{f}). && \text{(more estimation gap)}
\end{aligned}$$

- **Approximation**:
  - Which function classes $\mathcal{F}$ are "rich enough" for a broad class of learning problems?
  - E.g., neural networks, Reproducing Kernel Hilbert Spaces.
- **Optimization**:
  - Often finding minimizer of $\mathcal{R}_n$ is computationally hard.
  - What can we do instead?

Alternative model: online learning

Alternative to iid model for data:

- Examples arrive in a stream, one at at time.
- At time t:
  - Nature reveals x_t.
  - Learner makes prediction $\hat{y}_t$.
  - Nature reveals y_t.
  - Learner incurs loss $\ell(\hat{y}_t, y_t)$.

Alternative model: online learning

Alternative to iid model for data:

- Examples arrive in a stream, one at at time.
- At time t:
  - Nature reveals x_t.
  - Learner makes prediction $\hat{y}_t$.
  - Nature reveals y_t.
  - Learner incurs loss $\ell(\hat{y}_t, y_t)$.

Relationship between past and future:

- No statistical assumption on data.
- Just assume there exists $f^* \in \mathcal{F}$ with small (empirical) risk

$$\frac{1}{n} \sum_{t=1}^{n} \ell(f^*(x_t), y_t).$$