Prediction theory

COMS 4771 Fall 2025

Goals of prediction

General statistical model for prediction:

- Regard outcome that we want to predict as a random variable Y, and corresponding feature vector we observe as a random vector X
- Joint distribution P of (X, Y) is the "full population" of interest (Sometimes write as $P_{X,Y}$)

Problem: Create a program $f : \mathcal{X} \to \mathcal{Y}$ that, given X, returns a prediction of Y

Usually these programs are called predictors or prediction functions

How to measure how good/bad a prediction is?

Loss function $\mathrm{loss} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ measures how bad $\hat{y}$ is as a prediction of the outcome y

$$\mathrm{loss}(\hat{y}, y)$$

(Loss is usually non-negative, and smaller loss is better)

Example: zero-one loss (usually for classification problems)

$$\text{loss}_{0/1}(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} \neq y \\ 0 & \text{otherwise} \end{cases}$$

Example: squared error, a.k.a. square loss (for $\mathcal{Y} \subseteq \mathbb{R}$)

$$\text{loss}_{\text{sq}}(\hat{y}, y) = (\hat{y} - y)^2$$

X and Y are random variables, so $\text{loss}(f(X), Y)$ **is also a random variable**

Standard "average-case" benchmark: expected value of the loss, a.k.a. risk:

$$\text{Risk}[f] = \mathbb{E}[\text{loss}(f(X), Y)]$$

Expectation "integrates" $\text{loss}(f(x), y)$ with respect to joint distribution of (X, Y)

Standard loss functions are usually simplifications of application-specific loss

Example: spam filtering, $\mathcal{Y} = \{\text{ham}, \text{spam}\}$

- Mildly annoying if spam email is erroneous put in the inbox
- But very bad if real (important) email is put in spam folder
- Zero-one loss treats both types of mistakes equally
- Perhaps better to use $\text{loss}(\hat{y}, y)$ given by

	$y = \text{ham}$	$y = \text{spam}$
$\hat{y} = \text{ham}$	0	1
$\hat{y} = \text{spam}$	9	0

This is an example of a cost-sensitive loss function

Tricky coins

Can you predict the outcome of a coin toss?

I have 1000 different coins; heads-biases are $\theta_1, \ldots, \theta_{1000} \in [0,1]$
I pick a coin randomly and toss it; you need to guess the outcome

Optimal predictions of binary outcomes

Suppose you want to **predict binary outcome** Y where $\text{range}(Y) = \{0, 1\}$ to minimize the risk under zero-one loss (i.e., error rate)

$X =$ side-information, potentially informative about distribution of Y

Example:

- Y is outcome of coin toss in "tricky coins" scenario
- X is identity of the coin I picked

- Best prediction given $X = x$ is

$$f^\star(x) = \begin{cases} \_\_\_\_ & \text{if } \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \\ \_\_\_\_ & \text{if } \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \\ \_\_\_\_\_\_\_\_\_\_ & \text{if } \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \end{cases}$$

- $f^\star(x)$ depends on the conditional distribution of Y given $X = x$

Role of training data

Difficulty: **optimal predictions/predictors depend on distribution of** (X, Y)

- E.g., if distribution (X, Y) corresponds to entire human population, the need to poll entire human population to calculate optimal prediction / predictors

Training data can help, under certain assumptions

- **Nearest neighbor:** Assume training data is enough to "cover" most x's (w.r.t. distance function being used) and supply correct labels
- **Generative models:** Assume training data yields good estimate of $P_{X,Y}$ (via P_Y and $P_{X|Y}$)
- ...

Common assumption: **training data is "representative" sample of population**

Usual interpretation: training data $(X^{(1)}, Y^{(1)}), \dots, (X^{(n)}, Y^{(n)})$ form independent and identically distributed (i.i.d.) sample from distribution of (X, Y)

Notation:

$$((X^{(i)}, Y^{(i)}))_{i=1}^{n} \overset{\text{i.i.d.}}{\sim} (X, Y)$$

or

$$((X^{(i)}, Y^{(i)}))_{i=1}^{n} \overset{\text{i.i.d.}}{\sim} P$$

(if P is the distribution of (X, Y))

Example: suppose only one coin (or you ignore the identity of the chosen coin)

- Let $\hat{Y}$ be the majority value among $Y^{(1)}, \dots, Y^{(n)}$, i.e.,

$$\hat{Y} = \begin{cases} 0 & \text{if more 0s than 1s in } Y^{(1)}, \dots, Y^{(n)} \\ 1 & \text{if more 1s than 0s in } Y^{(1)}, \dots, Y^{(n)} \\ \text{either 0 or 1} & \text{if equal number of 0s and 1s} \end{cases}$$

- What's the probability that $\hat{Y} = y^{\star}$?

General case:

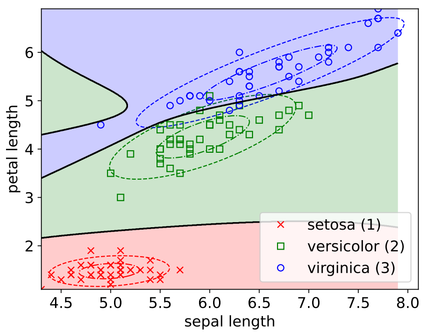
- Let $\hat{f}(x)$ be the majority value among all $Y^{(i)}$ such that $X^{(i)} = x$
  - If no such examples exist, then set $\hat{f}(x)$ arbitrarily

- Same as previous example, except with $D = |\mathrm{range}(X)|$ "coins", and as few as n/D training data pertinent to some coins

Some ways training data can help when $\mathrm{range}(X)$ **is large/infinite**

- Assume/leverage "local regularity"
  - Prediction at x "benefits" from data $(X^{(i)}, Y^{(i)})$ for which $X^{(i)}$ is nearby x

- Assume/leverage "global structure"
  - Prediction at x "benefits" from all data $(X^{(i)}, Y^{(i)})$



Why i.i.d. assumption? Consider some gross violations:

- Gross violation #1: Distribution of training data has nothing to do with distribution of (X, Y)

- Gross violation #2: Suppose $(X^{(1)}, Y^{(1)}) \sim (X, Y)$, and then we define $(X^{(i)}, Y^{(i)}) = (X^{(1)}, Y^{(1)})$ for all $i = 2, \ldots, n$

Role of test data

Assumption: test data $(\tilde{X}^{(1)}, \tilde{Y}^{(1)}), \dots, (\tilde{X}^{(m)}, \tilde{Y}^{(m)}) \overset{\text{i.i.d.}}{\sim} (X, Y)$, all independent of training data

Suppose we have created a classifier $\hat{f}\colon \mathcal{X} \to \mathcal{Y}$ using training data, and we would like to know how good it is

- (True) error rate is $\text{err}[\hat{f}] = \mathbb{E}[\text{loss}_{0/1}(\hat{f}(X), Y)]$
- To calculate $\text{err}[\hat{f}]$, we need to know the distribution of (X, Y)
- Using test data, we estimate $\text{err}[\hat{f}]$ by

$$\widetilde{\text{err}}[\hat{f}] = \frac{1}{m} \sum_{i=1}^{m} \text{loss}_{0/1}(\hat{f}(\tilde{X}^{(i)}), \tilde{Y}^{(i)})$$

  This is the test error rate

Test error rate: $\widetilde{\text{err}}[\hat{f}] = \dfrac{S}{m}$ where

$$S = \sum_{i=1}^{m} \mathbb{1}\{\hat{f}(\tilde{X}^{(i)}) \neq \tilde{Y}^{(i)}\}$$

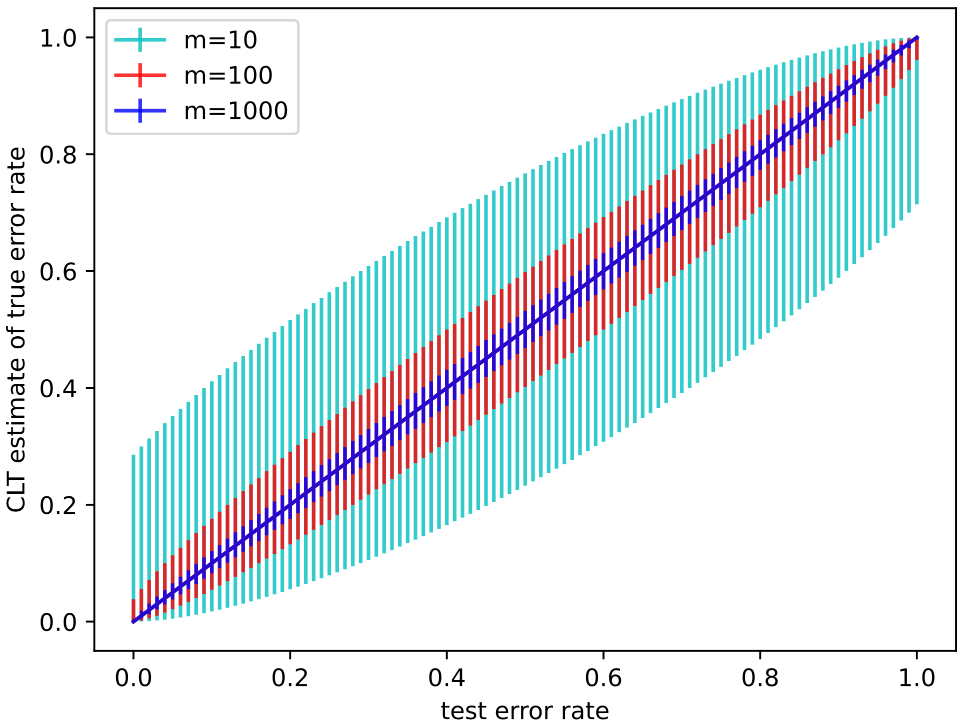
is sum of m i.i.d. $\text{Bernoulli}(\theta)$ random variables where $\theta = \text{err}[\hat{f}]$

Distribution of S is Binomial with m trials and success probability θ

- Notation: $S \sim \text{Binomial}(m, \theta)$

Facts about $S \sim \text{Binomial}(m, \theta)$

- $\mathbb{E}(S) = m\theta$
- $\text{var}(S) = m\theta(1-\theta)$
- $\dfrac{S - m\theta}{\sqrt{m\theta(1-\theta)}} \longrightarrow \text{N}(0,1)$ as $m \to \infty$ (by Central Limit Theorem)



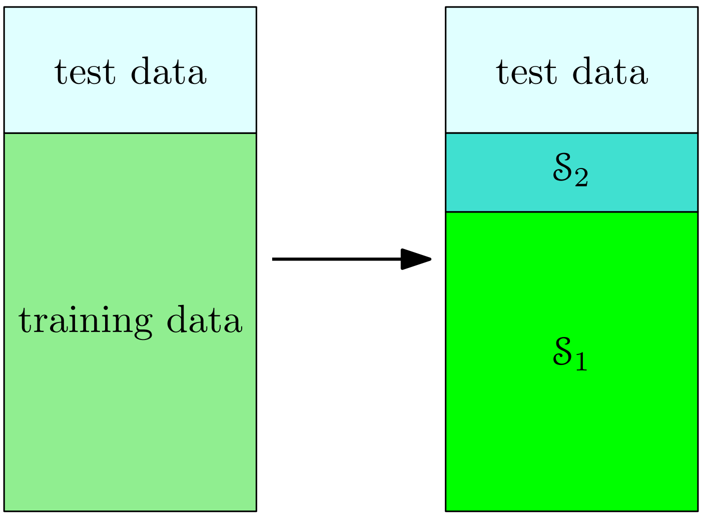
Why should test data be independent of training data?
Why doesn't previous argument apply with i.i.d. training data?

Cross validation

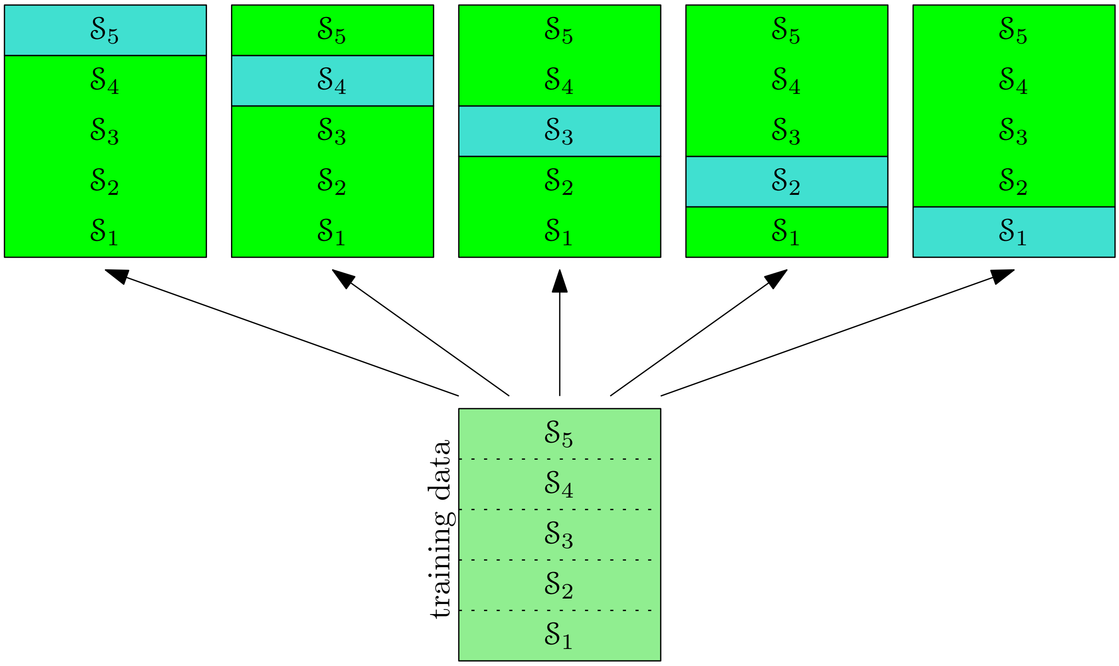
Common practice: split dataset into three parts

1. Training data: provided as input to learning algorithms
2. Validation data (a.k.a. development data, held-out data): used to evaluate experimentation with models, tweaks to learning algorithm, etc.
3. Test data: only used after you have settled on the learning algorithm/hyperparameters/etc., to evaluate the final predictor

(Hold-out) cross validation: simulate splitting dataset into training + test data
. . . all done only using training data



K-fold cross validation



Leave one out cross validation (LOOCV): K-fold cross validation with $K = n$