LECTURE 24

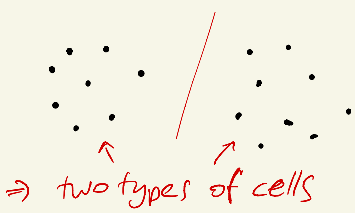
Machine Learning

- What is learning, understanding, attention, experience?
- How do we make technology achieve that?
- Math. models? Based on h.d. probability.

① Unsupervised learning – from own experience (infant). Supervised – from a teacher.

Examples we have seen before:

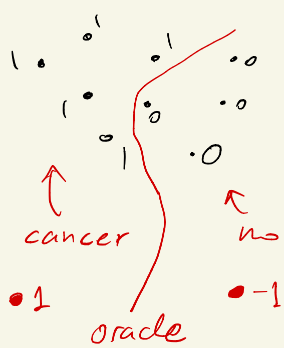
(a) Unsupervised learning: clustering



Unlabeled data $x_1, \dots, x_n \in \mathbb{R}^d$

e.g. n cells, d genes

(b) Superised learning: classification



Labeled data $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{R}^d \times \{0,1\}$

e.g. n people, d symptoms cancer/no "Training data"

Want to build an "oracle"

that makes a diagnosis

for a new patient: $x_{n+1} \longmapsto y_{n+1}$

Supervised ML: a general framework

- A pair of random variables (or vectors) $(X, Y) \in \mathcal{X} \times \mathcal{Y}$.

  label ↑ ↑↑ sets

Ex. $(X, Y) \in \mathbb{R}^d \times \{0, 1\}$ as above. Objective reality.

symptoms, cancer/no — X, Y are correlated, ideally strongly.

- The joint distribution of (X, Y) is unknown. We only see:
- Training data $(X_1, Y_1), \dots, (X_n, Y_n)$: iid copies of (X, Y).
- Goal: predict Y from X as best as we can.

$\Rightarrow$ We want to construct an oracle

$$h : \mathcal{X} \to \mathcal{Y} : \qquad h(X) \approx Y \qquad (*)$$

to make predictions for new, unseen data: $h(X_{n+1}) = Y_{n+1}$

input ↑ ↑ output

Q How do we quantify the "goodness of fit" $(*)$?

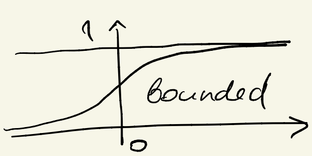
- We fix a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, e.g. $\ell(t-s) = (t-s)^2$, and define the risk (a.k.a. test error)

$$\boxed{R(h) := \mathbb{E}\, \ell(h(X), Y)} = \mathbb{E}\left(h(X_{n+1}), Y_{n+1}\right)$$

Examples:

(a) quadratic loss $\ell(t, s) = (t-s)^2 \Rightarrow R(h) = \mathbb{E}\left(h(X) - Y\right)^2$

(b) logistic loss $\rightarrow$



(c) hinge loss (svm)

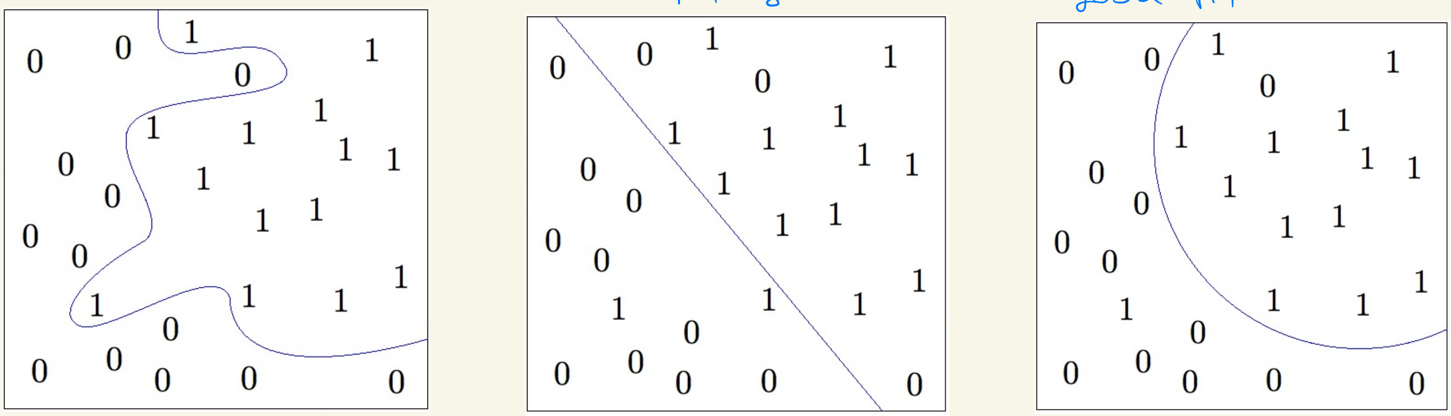
...

Q How do we construct an oracle h?

(1) h too complex: overfitting

(2) h too simple: underfitting

(3) h is OK: good fit



OUR STRATEGY:

1. Fix some collection of functions $\mathcal{H}$, called a hypothesis class.
2. Select $h \in \mathcal{H}$ that best fits the training data.

Examples:

(a) $\mathcal{H} = \{$all functions $h : \mathcal{X} \to \mathcal{Y}\}$ $\Rightarrow$ overfitting (1)

(b) All linear functions: $\mathcal{H} = \{ h(x) = \langle w, x \rangle + b : w \in \mathbb{R}^d, b \in \mathbb{R} \}$.

$\Rightarrow$ linear regression

(c) $\mathcal{H} = \{ h(x) = \text{sign}(\langle w, x \rangle + b) : w \in \mathbb{R}^d, b \in \mathbb{R} \}$ $\Rightarrow$ SVM (2)

(c) $\mathcal{H} = \{$all polynomials $p(x)$ of degree $\leq 2\}$. (3)

(d) $\mathcal{H} = \{$all functions realized by a given neural network architecture$\}$

(e) $\mathcal{H} = \{ h_1, h_2 \}$ $\Rightarrow$ hypothesis testing

No systematic way to choose $\mathcal{H}$. ("Model selection")

- The best $h \in \mathcal{H}$ is the one that minimizes the risk

$$R(h) = \mathbb{E}\,\ell(h(X), Y).$$

$$h^* := \underset{h \in \mathcal{H}}{\operatorname{argmin}}\ R(h).$$

- But $R(h)$ can't be computed (can't take $\mathbb{E}$ over the population)

empirical risk (a.k.a. training error)

$$R_n(h) := \frac{1}{n}\sum_{i=1}^{n} \ell(h(X_i), Y_i).$$

$$h_n^* := \underset{h \in \mathcal{H}}{\operatorname{argmax}}\ R_n(h)$$

can be computed from training data (can be NP hard)

Ex (Binary classification), quadratic loss ⇒

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n} \underbrace{(h(X_i) - Y_i)^2}_{= \begin{cases} 1 & \text{if } h(X_i) \neq Y_i \\ 0 & \text{otherwise} \end{cases}} = \% \text{ of misclassified training data.}$$

Empirical Risk Minimization (ERM) Algorithm

① Training: for input data $(X_1, Y_1), \dots, (X_n, Y_n)$; compute h_n^*.

② Prediction: on query X, output $h_n^*(X)$ "oracle"

How do we measure the quality?

- **Generalization error** $:= R(h_n^*)$

measures how well the algorithm generalizes to unseen data.

- Examples:

(a) $\mathcal{H} = \{\text{all functions}\}$, $Y = f(X)$.

$\exists$ a perfect fit to the training data: $h_n^*(x) := \begin{cases} Y_i & \text{if } x = X_i \\ 0 & \text{elsewhere.} \end{cases}$

training error $R_n(h_n^*) = 0$. (Overfitting)

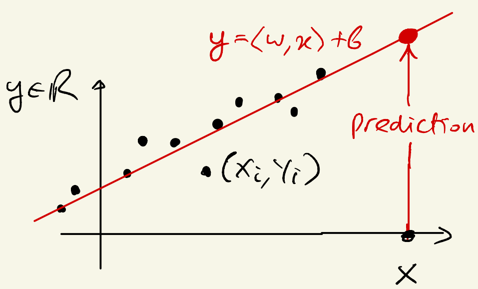
BUT does NOT generalize well:

$R(h_n^*)$ is large. Memorizes, not generalizes.

(b) $\mathcal{H} = \{\text{all linear functions}\}$, quadratic loss $\Rightarrow$

$$w_n^* = \arg\min_{w \in \mathbb{R}^d,\, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \left(\langle w, X_i \rangle + b - Y_i \right)^2$$

= linear regression. OK.



Our goal: Bound the generalization error.

How does it depend on the "complexity" of $\mathcal{H}$?