Two types of learning problems:

(1) $\boxed{\text{supervised problems}}$ ← first half of course

(2) un-supervised problems

(I) Supervised learning:

→ we have some "training" examples
we to learn from them

basically a prediction problem:

Want to predict some Y from X

have examples

two flavors:

(1) "Regression": predicting a continuous outcome

Ex. predict stock price from economic indicators

Ex. predict adult height from childhood diet

(2) "Classification": predicting a categorical outcome

Ex. Will an individual default on a loan given credit score?

Outcome = { 0 = don't default

$$\text{Outcome} = \begin{cases} 0 = \text{don't default} \\ 1 = \text{will default} \end{cases}$$

Ex. Predict disease status from bloodwork

$$\text{outcome} = \begin{cases} 0 = \text{no-disease} \\ 1 = \text{have disease} \end{cases}$$

binary problems

Ex. predict race from genomic data

"Multi-class problem"

$$\text{Outcome} = \begin{cases} \vdots \end{cases}$$

(II) Unsupervised problems

Don't have "training data".

We want to learn/summarize important trends in the data.

(No specific quantity we are predicting.)

General Setup for Supervised problems

We have some variable Y

called: outcome, predicted var, dependent

the thing we want to predict

the thing we want to predict

we're going to try to predict Y using other variables $X_1, \ldots, X_p$

\# vars.

called: predictors, features, covariates, independent variables.

the things used to predict Y.

Predict?

Want to come up w/ a function $\hat{f}$ so that

$$Y \approx \underbrace{\hat{f}(X_1, \ldots, X_p)}_{\hat{Y}}$$

Goal of this course:

(methods) ① how do we construct $\hat{f}$?

(evaluation) ② how do we determine if $\hat{f}$ is good?

Way supervised problems work:

we will use training data to construct $\hat{f}$

Statistical Learning

Assume we have some training data — training data size

$$(y_n, x_n) \text{ for } n = 1, \dots, N$$

where

$y_n \in S$ and $x_n \in \mathbb{R}^p$ — measured covariates for nth obs.

e.g. $x_n = (x_{n1}, x_{n2}, \dots, x_{np}) \in \mathbb{R}^p$

$S = \mathbb{R}$ for regression problems

$S = \{c_1, c_2, c_3, \dots, c_K\}$ for classification — # classes

for Statistical learning we think of the training data $\{(y_n, x_n)\}_{n=1}^N$ as having some (maybe complicated) joint distribution (probabilistic model)

We want to construct some $\hat{f}$, to do this we first construct a measure of "wrongness" called a Loss function $L: \mathcal{F} \to \mathbb{R}$ — space of possible fs

so that

$L(f)$ ← how good a job f does i.e. how close is $f(x_1, \dots, x_p)$ to Y — some potential $\hat{f}$

ideally we would let

$$(*) \quad \hat{f} = \underset{f \in F}{\operatorname{argmin}} \; \mathbb{E}\left[L(f)\right]$$

Risk = expected loss. (Risk minimization)

we often don't know the joint dist of training data exactly. So instead we let

we only have training data

$$\rightarrow \hat{f} = \underset{f \in F}{\operatorname{argmin}} \; \frac{1}{N} \sum_{n=1}^{N} L(f)$$

(empirical risk minimization)

Linear Regression:

1. classic method (well-studied)
2. Simple (this is good)
3. can be powerful
4. basis for many other methods

We want $\hat{f}$. Regression assumes

our estimate

$$Y = f(X_1, \ldots, X_p) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$$

true underlying model

a linear relation btwn Y and Xs

If we let $\underset{\sim}{X} = (1, X_1, \ldots, X_p) \in \mathbb{R}^{p+1}$

If we let $\underset{\sim}{X} = (1, X_1, \ldots, X_p) \in \mathbb{R}^{p+1}$

and $\beta = (\beta_0, \beta_1, \ldots, \beta_p) \in \mathbb{R}^{p+1}$

the we can simply write

true βs assoc. w/ f

$$Y = \underset{\sim}{X}^T \beta$$

We say regression is linear b/c f is linear in βs (linear = inner product)

LR: makes simplifying assumption Namely: linearity

$\mathcal{F}$ could be infinite dimensional (intractable)

$$\Downarrow \text{LR}$$

$$\mathcal{F} \simeq \mathbb{R}^{p+1}$$

(we only need to learn $p+1$ βs to determine f or $\hat{f}$)

To determine $\hat{f}$ we need $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$

so we form a loss function

$$\boxed{L(f) = L(\beta) = \left(\underline{Y} - \left(\beta_0 + \sum_{j=1}^{p} \beta_j \underline{X_j}\right)\right)^2}$$

how good $f(X) \approx Y$ and then

how good $f(X) \simeq Y$ and then

$$\boxed{\hat{\beta} = \underset{\beta}{\operatorname{argmin}}\, L(\beta)}$$

Subsequently, $\hat{f}(\underset{\sim}{X}) = \underset{\sim}{X}^T \beta$

Practically: empirical risk to minimize

$$L(\beta) = \sum_{n=1}^{N} \Big(y_n - \beta_0 - \sum_{j=1}^{p} \beta_j X_{nj}\Big)^2$$

If we let X be our "design" matrix

$$X = \begin{bmatrix} 1 & X_{11} & X_{12} & \cdots & X_{1p} \\ \vdots & X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & & & & \\ 1 & X_{N1} & X_{N2} & \cdots & X_{Np} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

and $y = (y_1, \ldots, y_n) \in \mathbb{R}^N$

then our empirical risk to minimize is

$$\boxed{L(\beta) = \|y - X\beta\|^2}$$

note this takes the correct form.

turns out (Calc III from hell)

$$\boxed{\hat{\beta} = (X^T X)^{-1} X^T y}$$

$$\boxed{\hat{\beta} = (X^T X)^{-1} X^T y}$$

projection of y onto $Col(X)$.

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y$$