Statistical Learning and Linear Regression

Almost any attempt to clearly define a difference between statistics and machine learning – it wrong.

it's wrong

Classically: $\boxed{\text{Statistics}}$ (statistical inference)
formulating probabilistic models about variables in order to infer properties/params

$\boxed{\text{machine learning}}$: using algos to predict output given input

Statistical (machine) learning (SML)

build / understand ML approaches using prob. models

Broadly there are two categories of SML problems

I Supervised Learning

supervised meaning we have some examples to "train" our approach

idea: want to predict Y from $\underset{\sim}{X}$
and we have example pairs
$(Y, \underset{\sim}{X})$

w/in supervised problems two types of problems:

with in

1. "regression" = predicting a continuous outcome
2. "classification" = predicting a categorical outcome

Ex. regression problems:

- predicting $\underbrace{\text{stock market performance}}_{}$ from $\underbrace{\text{economic indicators}}_{\underset{\sim}{X}}$ $Y \in \mathbb{R}$

- predicting $\underbrace{\text{adult height}}_{Y \in \mathbb{R}}$ from $\underbrace{\text{childhood height}}_{\underset{\sim}{X}}$

classification problems:

- predict if individual $\overbrace{\text{will default on a loan}}^{Y}$ given $\overbrace{\text{credit score}}$ $\underset{\sim}{X}$

binary classification

$$Y = \begin{cases} 0 & \text{don't default} \\ 1 & \text{do default} \end{cases}$$

- predict $\underbrace{\text{racial identity}}_{Y}$ from $\underbrace{\text{genomic data}}_{\underset{\sim}{X}}$

multi-class classification

$$Y = \begin{cases} \text{white} \\ \text{black} \\ \text{green} \end{cases}$$

begs question about what ML is

Ⅱ Unsupervised Learning

begs q about what ML is?

Don't have a clear prediction problem.

We want to learn/summarize important trends/info/rels about the vars.

In this class: first half = supervised problems
second half introduce unsupervised problems.

Mathematical Setup for supervised Learning

We have some var. Y

called: outcome, predicted var, dependent var

the thing we want to predict

We're going to try to predict Y given some other vars

$$\underset{\sim}{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(p)})$$

$p = \#$ vars

called: predictors, features, covariates, independent vars

they are the things we use to predict.

Predict?

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

$$Y \approx \hat{f}(\underset{\sim}{X}) = \hat{f}(X^{(1)}, \ldots, X^{(p)}) \overset{\text{def}}{=} \hat{Y}$$

Course goals:

1. Methods: how do we construct $\hat{f}$?
2. Evaluation: how do we determine if $\hat{f}$ is good.

For supervised problems

we assume we have <u>training data</u> to construct $\hat{f}$ of the form

$$(y_n, \underset{\sim}{x}_n) \text{ for } n=1,\ldots,N$$

Sample size

where $y_n \in S$

$S =$ space of possible y_n-s

$S = \mathbb{R}$ for regression

$S = \{c_1, c_2, \ldots, c_K\}$

S= { c_1,...,c_k } is for classification

$$\underset{\sim}{x}_n = (x_n^{(1)}, \ldots, x_n^{(p)})$$

SML

Assume that these traing data are sampled from some joint dist $p(\underset{\sim}{x}, y)$ – typically i.i.d.

what is a general way of constructing a $\hat{f}$?

We can construct a measure L (loss fn) that tells us, for some putative f, how good f does at predicting Y from $\underset{\sim}{X}$.

[Given $(\underset{\sim}{X}, Y)$ come from $p(\underset{\sim}{x}, y)$]

Idea!

$$L(Y, f(\underset{\sim}{X})) = \begin{cases} \text{large if } Y \not\approx f(\underset{\sim}{X}) \\ \text{small if } Y \approx f(\underset{\sim}{X}) \end{cases}$$

Examples: regression context

$$L(y, f(\underset{\sim}{x})) = (y - f(\underset{\sim}{x}))^2$$ squared loss

$$L(y, f(\underset{\sim}{x})) = |y - f(\underset{\sim}{x})|$$ abs. loss

classification context

$$L(y, f(\underset{\sim}{x})) = \begin{cases} 0 & y = f(\underset{\sim}{x}) \\ 1 & y \neq f(\underset{\sim}{x}) \end{cases}$$ 0-1 loss

either context: $L(y, f(\underset{\sim}{x})) = -\log p(\underset{\sim}{x}, y)$

neg. log-like. loss

negative log-likelihood loss

How do we use a loss to build a $\hat{f}$?

We'd like to choose $\hat{f}$ so that it incurs a small loss.

In particular minimize expected loss:

ideal: Risk Minimization

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \; \mathbb{E}[L(Y, f(\underset{\sim}{X}))]$$

Risk = Expected loss

$\mathcal{F}$ = hypothesis space

b/c we don't know $p(\underset{\sim}{x}, y)$ to calc $\mathbb{E}[\cdots]$ we instead approx. using a sample of training data.

Aside: $\mathbb{E}[z] \approx \bar{z} = \frac{1}{N}\sum_{n=1}^{N} z_i$ when z_i iid ...

training data

So

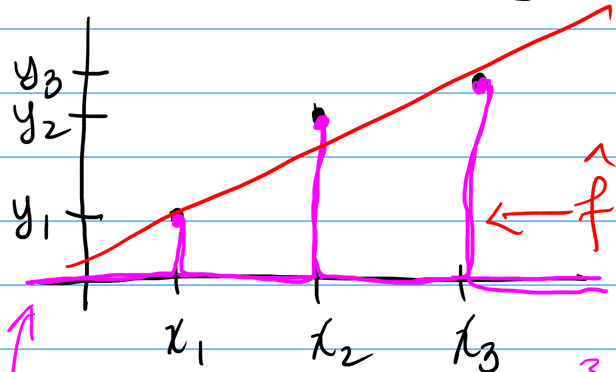
$$\hat{f} = \underset{f \in \mathcal{F}}{\arg\min} \underbrace{\frac{1}{N}\sum_{n=1}^{N} L(y_n, f(\underset{\sim}{x}_n))}_{\text{Empirical Risk}}$$

Empirical Risk $\longrightarrow \mathbb{E}[L(Y, f(\underset{\sim}{X}))]$ as $N \to \infty$.

ERM: empirical risk minimization

Have a couple choices: L and $\mathcal{F}$

One big problem: over-fitting



not penalising correctly

Consider L = sq. err.

$\mathcal{F}$ = all possible functions

too large

has $ER = 0 = \underbrace{\frac{1}{N}\sum_{n=1}^{3}(y_n - f(x_n))^2}_{0}$

this won't generalize well.

Way to avoid this:

related

1. Restrict $\mathcal{F}$ (linear, quad, smooth, diff'able)
2. Change L (penalization to avoid silly f)
3. get a better estimate of Risk. (test/validation, cross-validation)

Linear Regression

Why look at LR?

1. classic method - very well studied
2. simple (good)
3. can be very powerful
4. LR is the basis for more complex methods

Properly: $\underbrace{\text{linear}}_{\mathcal{F} = \text{lin. fns}}$ $\underbrace{\text{least-squares}}_{L(y, f(\underset{\sim}{x})) = (y - f(\underset{\sim}{x}))^2}$ $\underbrace{\text{regression}}_{\text{continuous } y}$ → OLS: ordinary least-squares (regression)

Notation: $Y, \underset{\sim}{X}$ generic/RVs

$\underset{\sim}{X} = (X^{(1)}, \ldots, X^{(P)})$

Samples: $y_n, \underset{\sim}{x}_n = (x_n^{(1)}, \ldots, x_n^{(P)})$

Setup:

$$Y = f(X^{(1)}, \ldots, X^{(P)}) = \beta^{(0)} + \sum_{j=1}^{P} \beta^{(j)} X^{(j)} \quad \text{(linear)}$$

If $\underline{X} = (1, X^{(1)}, X^{(2)}, \ldots, X^{(P)}) \in \mathbb{R}^{P+1}$ design

and $\beta = (\beta^{(0)}, \beta^{(1)}, \ldots, \beta^{(P)}) \in \mathbb{R}^{P+1}$

design matrix

then

$$Y = \underline{X}^T \beta$$

linear in β b/c its basically inner product

Idea: by assuming a linear form of $f \in \mathcal{F}$ so that

$$f(\underset{\sim}{x}) = \underline{x}^T \beta \longrightarrow \beta \in \mathbb{R}^{p+1}$$

we restrict $\mathcal{F}$ to that of linear fns.

$$\mathcal{F} \text{ (potentially infinite dim'l)} \overset{OLS}{\rightsquigarrow} \mathcal{F} = \mathbb{R}^{p+1}$$

b/c $f = f_\beta \in \mathcal{F}$