Can generalize EVP into SVD (Singular Value Decomp.)

For any matrix X $(m \times n)$ → XX^T is $\underbrace{m \times \underbrace{n \times n} \times m}_{m \times m}$

X^TX is $n \times n$

I can decompose it as

$$X = UDV^T$$

where ① $\underset{m \times m}{U}$ is orthog. whose cols are the e-vectors of XX^T ← symmetric

↑ left sing. vectors $(XX^T)^T = (X^T)^T X^T = XX^T$

② $\underset{n \times n}{V}$ is orthog. whose cols are the e-vects of X^TX ← symmetric

↑ right sing. vects.

③ $$D = \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0 \\ & & \sigma_r & \\ \hline & 0 & & 0 \end{array}\right]$$ is $m \times n$

rank(X)

where $\sigma_i = \sqrt{\lambda_i} = \sqrt{\text{e-vals of } XX^T \text{ or } X^TX}$

↑ singular values

Eqn for e-vects/values: $Xv_i = \lambda_i v_i$

Eqn for singular vects/values: $\underline{Xv_i} = \underline{\sigma_i u_i}$

$X = UDV^T \Rightarrow \underline{XV} = UDV^TV = \underline{UD}$

Ex.

$$X = \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}$$ let's get SVD.

$$X^T X = \begin{bmatrix} 17 & 8 \\ 8 & 17 \end{bmatrix}$$

$$v_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad ; \quad v_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$\lambda_1 = 25 \qquad \lambda_2 = 9$$

$$V = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad D = \begin{bmatrix} 5 & 0 \\ 0 & 3 \\ 0 & 0 \end{bmatrix}$$

Know: $X v_i = \sigma_i u_i \Leftrightarrow u_i = \dfrac{X v_i}{\sigma_i}$

$$u_1 = \frac{X v_1}{\sigma_1} = \frac{1}{5}\begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}/\sqrt{2} = \frac{1}{5\sqrt{2}}\begin{bmatrix} 5 \\ 5 \\ 0 \end{bmatrix}$$

$$u_2 = \frac{1}{3}\begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix}/\sqrt{2} = \frac{1}{3\sqrt{2}}\begin{bmatrix} 1 \\ -1 \\ 4 \end{bmatrix}$$

u_3 = orthog to u_1 and u_2

$$= \frac{1}{\sqrt{2 + 1/4}}\begin{bmatrix} 1 \\ -1 \\ -1/2 \end{bmatrix}$$

Statistical Learning

Trying to distinguish among

- statistics
- machine learning
- data science

is almost always wrong.

Statistical machine learning (SML)

building / analyzing data analysis approaches using probabilistic thinking.

Broadly two categories of ML problems

(I) Supervised Learning

supervised means we have examples to "train" the method

idea: want to predict some Y from $\underset{\sim}{X}$ and we have examples (traing data)

$$(y_n, \underset{\sim}{x}_n)_{n=1}^{N}$$

w/in supervised two sub-classes:

1. regression problems : Y is a continuous valued var.

2. classification problems : Y is discrete valued var.

Ex. regression: $Y \in \mathbb{R}$

- predicting $\overbrace{\text{stock market perf.}}^{Y}$ from $\underbrace{\text{economic indicators}}_{\underset{\sim}{X}}$

- predicting $\underbrace{\text{adult height}}_{Y}$ from $\overbrace{\text{childhood nutrition}}^{\underset{\sim}{X}}$

Ex. classification

$Y = \{\text{default, not}\}$

Binary classification - predict if individual will default on a loan given $\underbrace{\text{credit score}}_{\underset{\sim}{X}}$

- predict $\underbrace{\text{Bird species}}_{Y=\{\text{hawk, sparrow, ...}\}}$ from $\underbrace{\text{genomic data}}_{\underset{\sim}{X}}$

(II) Unsupervised

No clear prediction problem.

No clear distinction btwn Y and $\underset{\sim}{X}$

Just have some data.

Goal: learn / summarize important trends or info or relationships in my data.

Mathematical setup for supervised problems:

We have some variable Y we want to predict, variously called

- outcome
- predicted var
- dependent var
- ...

We're going to try to predict this Y using some other vars

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

$p = \#$ vars

Called: predictors, features, covariates, independent vars, ...

How to predict?

Come up w/ a function $\hat{f}$ so that

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

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 some training data to construct $\hat{f}$:

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

N = size of training data

where $\underset{\sim}{x}_n \in \mathbb{R}^p$ and

$y_n \in \mathbb{R}$ for regression problems

$y_n \in \{c_1, c_2, \dots, c_K\}$

K = # classes.