For e-vals/vecs

$$Xv_i = \lambda_i v_i \qquad X = QDQ^T \qquad XQ = QD$$

For singular vals/vecs

$$Xv_i = \sigma_i u_i$$

$$X = UDV^T \quad (m\times m,\ m\times n,\ n\times n)$$

$$XV = UD \quad (m\times n)$$

Ex.

$$X = \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 2 & -2 \end{bmatrix}$$

Let's get the SVD.

XX^T is 3×3

X^TX is 2×2 $\rightsquigarrow$ get V

$$X^TX = \begin{bmatrix} 17 & 8 \\ 8 & 17 \end{bmatrix} \qquad \begin{bmatrix} a & b \\ b & a \end{bmatrix}$$

(

↳ get e-vals/vecs

$v_1 = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\1\end{pmatrix}$ assoc. $\lambda_1 = 25 \rightarrow \sigma_1 = 5$

$v_2 = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix}$ assoc. $\lambda_2 = 9 \rightarrow \sigma_2 = 3$

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

3×2

Know: $Xv_i = \sigma_i u_i$

then $u_i = \frac{Xv_i}{\sigma_i}$

$$u_1 = \frac{Xv_1}{\sigma_1} = \frac{1}{5}\begin{bmatrix}3 & 2\\ 2 & 3\\ 2 & -2\end{bmatrix}\begin{pmatrix}1\\1\end{pmatrix}/\sqrt{2} = \boxed{\frac{1}{5\sqrt{2}}\begin{bmatrix}5\\5\\0\end{bmatrix}}$$

$$u_2 = \frac{Xv_2}{\sigma_2} = \frac{1}{3}\begin{bmatrix}3 & 2\\ 2 & 3\\ 2 & -2\end{bmatrix}\begin{pmatrix}1\\-1\end{pmatrix}/\sqrt{2} = \boxed{\frac{1}{3\sqrt{2}}\begin{bmatrix}1\\-1\\4\end{bmatrix}}$$

$$u_2 = \frac{Xv_2}{\sigma_2} = \frac{1}{3}\begin{bmatrix} 2 & 2 \\ 2 & 2 \\ 2 & -2 \end{bmatrix}\begin{pmatrix} 1 \\ -1 \end{pmatrix}/\sqrt{2} = \frac{1}{3\sqrt{2}}\begin{bmatrix} -1 \\ -1 \\ 4 \end{bmatrix}$$

u_3 is whatever is orthog. to u_1 and u_2.

$$u_3 = \frac{1}{\sqrt{2+\frac{1}{4}}}\begin{bmatrix} 1 \\ -1 \\ -1/2 \end{bmatrix}$$

Verify at home: $X = UDV^T$.

Statistical Machine Learning

Building/analyzing data analysis procs
using probabilistic thinking
(where helpful)

Broadly two main cats of ML probs

Broadly two main cats of ML probs

→ (I) supervised learning

supervised means we have "examples" to train the ML method

idea: want to predict some Y from $\underset{\sim}{X}$ and we have examples (traing data): $(x_n, y_n)_{n=1}^{N}$

w/in supervised two main types

(1) regression problems: Y is a continuas /real-valued var.

(2) classification problems: Y is discrete valued / finite (categorical)

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

- predict stock market each day

- predict stock market perf. from economic indicaters

$\underset{\sim}{X}$ y

- predicting adult height from childhood nutrition

$\underset{\sim}{X}$

Ex. classication: Y is categorical

- predict if individual will default on loan given credit score

$Y \in \{0, 1\}$

$\underset{\sim}{X}$

binary classification problem

- predict plant species from photo

Y X

$Y \in \{tree, fern, bush, \ldots\}$ pixels

(II) Unsupervised Problem

No clear prediction problem.

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

(just have some data)

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

Math setup for supervised learning

We have some var Y we want to predict from

predict from

$$\underset{\sim}{X} = (X_1, X_2, \ldots, X_p)$$

↑ p = num of input vars.

We're going to come up w/ a function $\hat{f}$ so that

$$Y \approx \hat{f}(\underset{\sim}{X}) = \hat{f}(X_1, X_2, \ldots, X_p)$$

Course goals: talk about 2 things

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

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

For super problems we assume we have training data to construct $\hat{f}$:

(x_n, y_n) ← N = size of

$(\underset{\sim}{x}_n, y_n)_{n=1}^{N}$ ← N = size of training data

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

$y_n \in \mathbb{R}$ for regr

$y_n \in \{c_1, c_2, \ldots, c_K\}$ for classr.

K ← # classes

Linear Regression (regression problems)

Why? (1) classic method - well studied

(2) simple (good)

(3) powerful

(4) basis for more complex method

Setup: For lin. reg. assume we have some input vars (design):

Some input vars (design):

$$\underset{\sim}{X} = (X_1, \ldots, X_p) \in \mathbb{R}^p$$

and assoc. coef.

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

then lin. reg. assumes a model of the form

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

$$= \sum_{j=1}^{p} \beta_j X_j$$

to learn $\hat{f}$ we will "learn" some good values for β, call them $\hat{\beta}$ and then form $\hat{f}$ as

$$\hat{f}(\underset{\sim}{X}) = \underset{\sim}{X}^T \hat{\beta} = \sum_{j=1}^{p} \hat{\beta}_j X_j .$$

Where is the intercept?

Why not

$$\hat{f}(\underset{\sim}{X}) = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j X_j$$

Can hide the intercept in the design

In this course we use $\underset{\sim}{X}$ in two ways

(1) vars. we measure

(2) the input to our ML algo. (design)

this more

Need not be the same, can do any pre-proc. on (1) to get (2)

vars I measure : $X_1, X_2, \ldots, X_p$

input to ML : X_1^2, $\log(X_2)$, $\sin(X_3)$...

Can always get intercept by setting

$X_1 = 1$ always,

i.e. $\underset{\sim}{X} = (1, \text{other vars})$

If I do that then

$$\underset{\sim}{X}^T \beta = (1, X, Y, Z)^T (\beta_1, \beta_2, \beta_3, \beta_4)$$

$$= \beta_1 + \beta_2 X + \beta_3 Y + \beta_4 Z + \cdots$$

How do I determine good values for $\hat{\beta}$?

I want to choose $\hat{\beta}$ so that

$$\underset{\sim}{Y} \simeq \hat{f}(\underset{\sim}{X}) = \underset{\sim}{X}^T \hat{\beta}$$

Many ways to determine if $Y \approx \hat{f}(x)$

Simplest way: Least-Squares regression

Choose $\hat{\beta}$ as value that minimizes the squared training error (loss)

$$L(\beta) = \sum_{n=1}^{N} (y_n - x_n^T \beta)^2$$

y_n: true target in training

$x_n^T \beta$: predicted val.