Dimensionality Reduction

2-D, 3-D :- scatter plot

4-D, 5-D, 6-D :- pair plot

10-D, 100-D, 1000-D → what about them how to visualize it?

n-D → 2-D or 3-D

PCA & t-SNE

Row-vector & column-Vector

flower :- [SL, PL, SW, PW]

↳ i^{th} point :

$x_i \in R^d$ → d-dimensional column vector, containing Real values

→ Real space

$$x_i = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ x_{i_3} \\ \vdots \\ x_{i_d} \end{bmatrix}_{d \times 1}$$

for eg:-

$$f_1 = \begin{bmatrix} 2.1 \\ 3.2 \\ 1.6 \\ 1.4 \end{bmatrix} \begin{matrix} \rightarrow P.L \\ \rightarrow P.W \\ \rightarrow S.L \\ \rightarrow S.W \end{matrix}$$

Note :- The default vector is always column vector.

$x_i = [2.1, 3.2, 4.6, 1.2]_{1 \times 4}$: row-vector.

How to represent data set

$$D = \{x_i, y_i\}_{i=1}^{n}$$ → # data points

data-points ← $x_i \in R^d$; $x_i \in R^4$

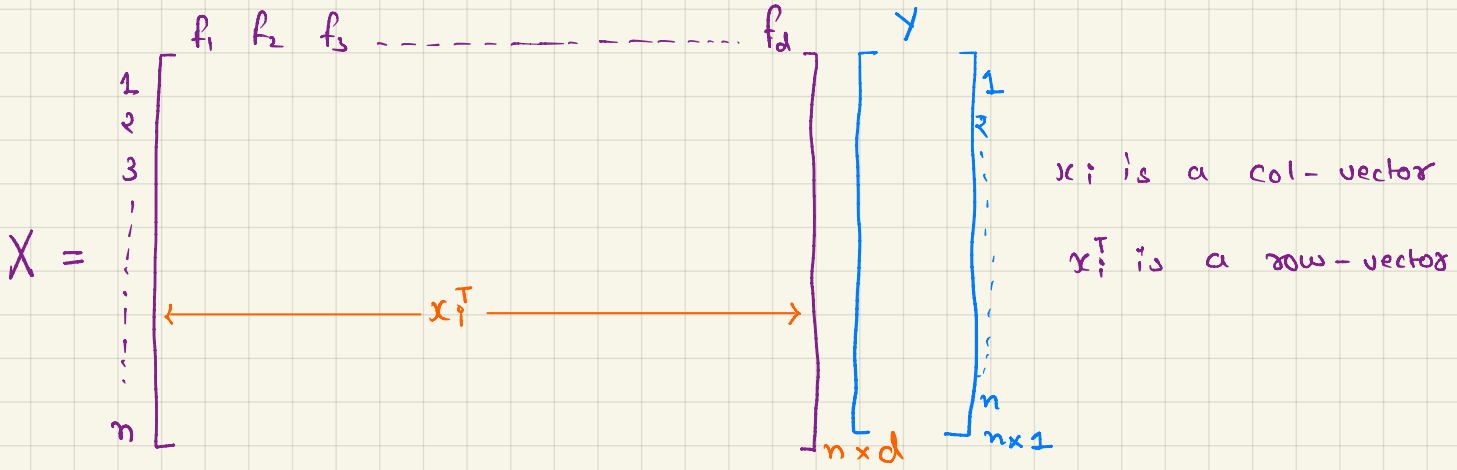
$$x_i = \begin{bmatrix} SL \\ SW \\ PL \\ PW \end{bmatrix}$$

class-labels ← $y_i \in$ {setosa, versicolor, verginica}

Dataset as a data-matrix

$$D = \{x_i, y_i\}_{i=1}^{n}, \quad x_i \in R^d, \quad y_i \in \{S, Vi, Ve\}$$

($x_i \in R^d$: d-features)



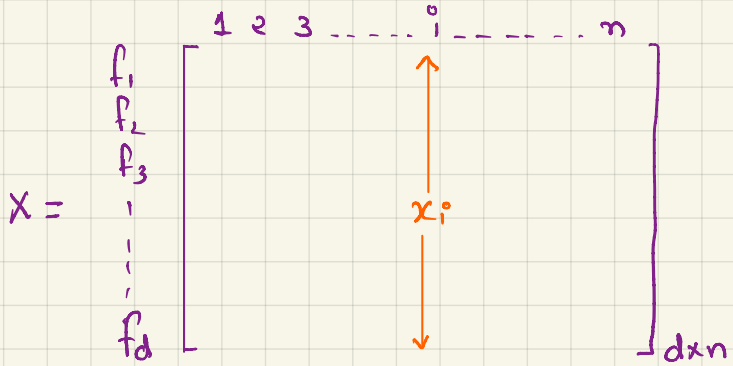
x_i is a col-vector

x_i^T is a row-vector

Each datapoint : row

Each column : feature

Another way

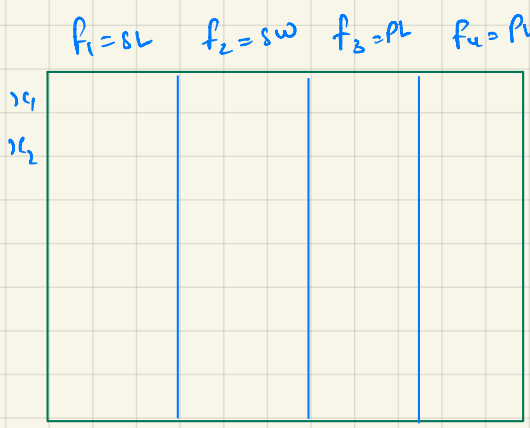


column : data-point

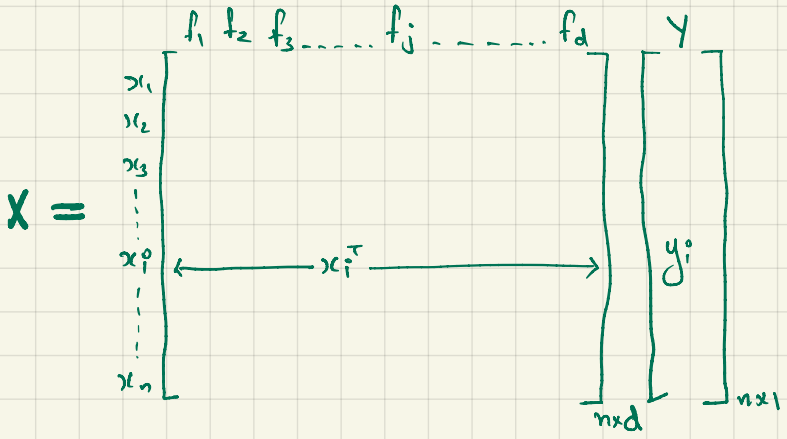
row : feature/variable

$f_1 = PL$
$f_2 = PW$
$f_3 = SL$
$f_4 = SW$

eg :-



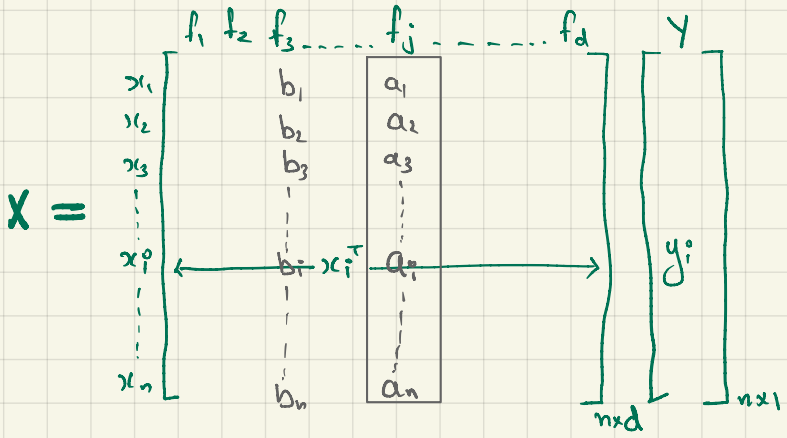
Data preprocessing : feature/column Normalization



obtain data → pre-processing → data modeling (dim - reduction)

↳ column normalization

⋮



$a_1, a_2, \ldots\ldots\ldots a_n$ = n-value of f_j

$\max(a_i) = a_{max} \geq a_i \quad (i: 1 \to n)$

$\min(a_i) = a_{min} \leq a_i \quad (i: 1 \to n)$

$a_1', a_2', a_3', \ldots\ldots a_i', \ldots a_n'$

$$a_i' = \frac{a_i - a_{min}}{a_{max} - a_{min}} \qquad a_i' \in [0, 1]$$

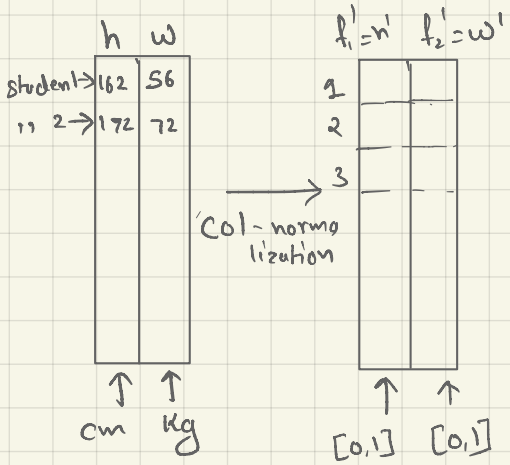
eg:- $a'_{min} = \frac{a_{min} - a_{min}}{a_{max} - a_{min}} = 0$; $a'_{max} = \frac{a_{max} - a_{min}}{a_{max} - a_{min}} = 1$

$a_1, a_2, a_3 \ldots\ldots\ldots a_i \ldots\ldots a_d$; $a_i \in R$

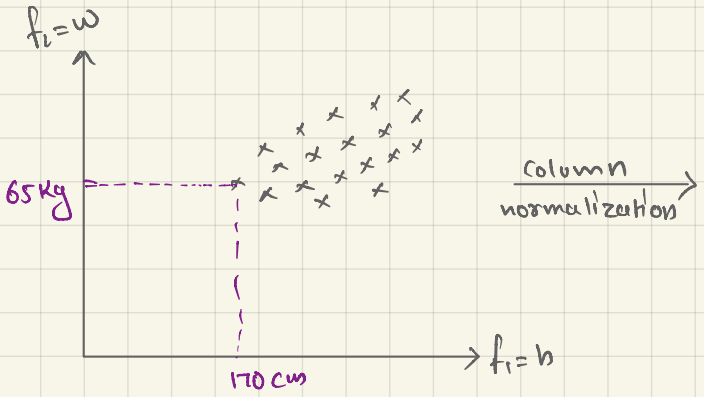
↓ column normalization

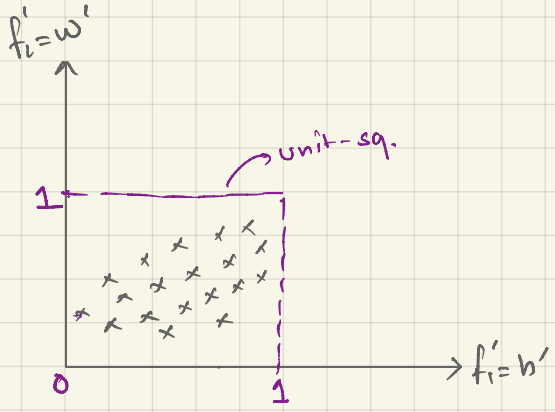
$a_1', a_2', a_3' \ldots\ldots\ldots a_i' \ldots\ldots a_d'$; s.t $a_i' \in [0,1]$

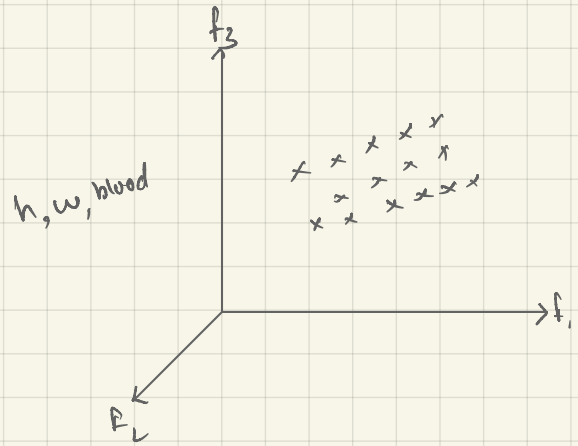
why

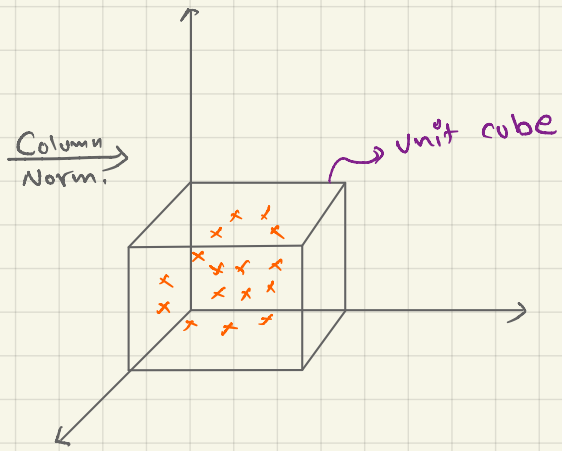


you are getting rid of scale (cm → m or feet) by doing normalization. we are putting everything in b/w of 1.





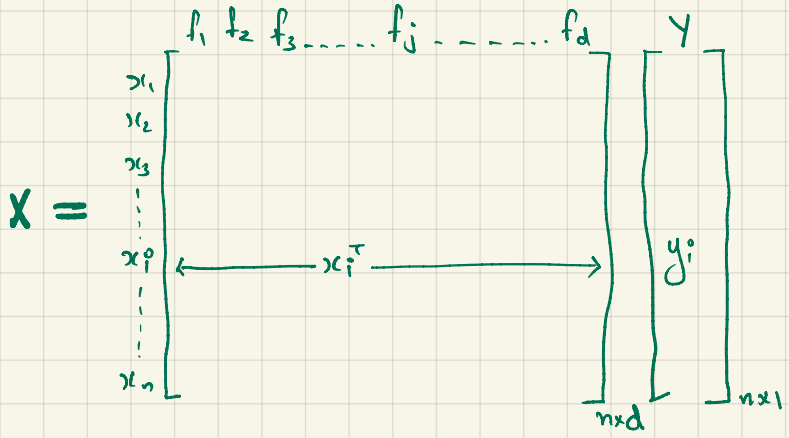




⇒ any where in n-dim. space $\xrightarrow{\text{Col. norm.}}$ unit-hyper cube in the same n-dim. space.

↳ by doing that we are getting rid of scale.

Mean - Vector :-



Let, $x_i \in R^d$

Suppose

$\vec{x_1} = [2.2 \ , \ 4.2] \in R^2$

$\vec{x_2} = [1.2 \ , \ 3.2] \in R^2$

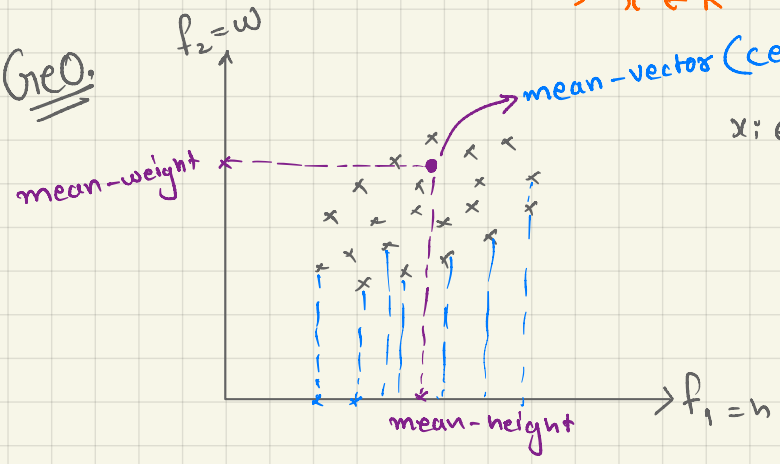
$\vec{x_1} + \vec{x_2} = [3.4 \ , \ 7.4]$

→ mean vector

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} \vec{x_i} = \frac{1}{n} (\vec{x_1} + \vec{x_2} + \vec{x_3} + \dots + \vec{x_n})$$

$\bar{x} \in R^d$

Geo:



$x_i \in R^2$

$[h, w]$

f_1, f_2

$\bar{x} = [h_{\bar{x}}, w_{\bar{x}}]$

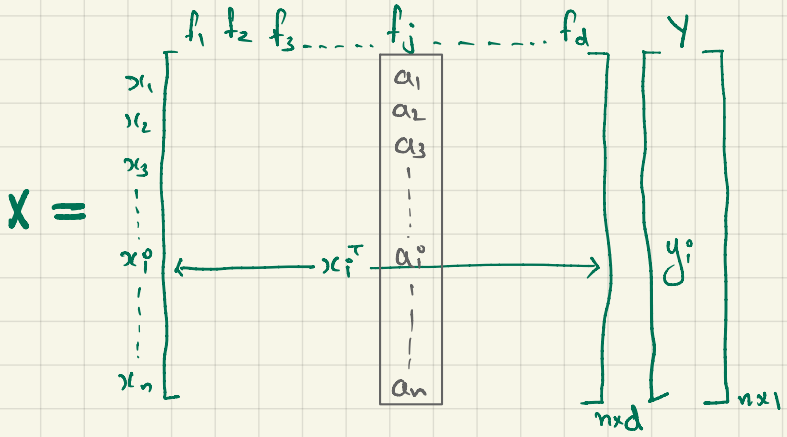
$h_{\bar{x}} = \text{mean} (h_i)_{i=1}^{n}$

$w_{\bar{x}} = \text{mean} (w_i)_{i=1}^{n}$

Data preprocessing : column - standardization

Column - normalization :- $[0,1]$ ← get rid of scales of each feature

column - standadization :- more often used in practice.



$a_1, a_2, a_3 \dots\dots a_i \dots\dots a_n$ ← n-value of f_j ← any distribution

↓ col. std.

$a_1', a_2', a_3' \dots\dots a_i' \dots\dots a_n'$ ← $\begin{cases} \text{mean}\{a_i'\}_{i=1}^n = 0 \\ \text{std-dev}\{a_i'\}_{i=1}^n = 1 \end{cases}$ standardization

→ $\bar{a} = \text{mean}\{a_i\}_{i=1}^n$ ← sample mean

$s = \text{std-dev}\{a_i\}_{i=1}^n$ ← sample std-dev

$$a_i' = \frac{a_i - \bar{a}}{s} \longrightarrow \begin{cases} \text{mean}\{a_i'\}_{i=1}^n = 0 \\ \text{std-dev}\{a_i'\}_{i=1}^n = 1 \end{cases}$$

→ similar like Std-normal variate (z)

$$z = \frac{x - \mu}{\sigma}$$

$x \sim N(\mu, \sigma)$
$z \sim N(0,1)$
→ only diff. it comes from normal distbⁿ & above formula come from any distbⁿ.