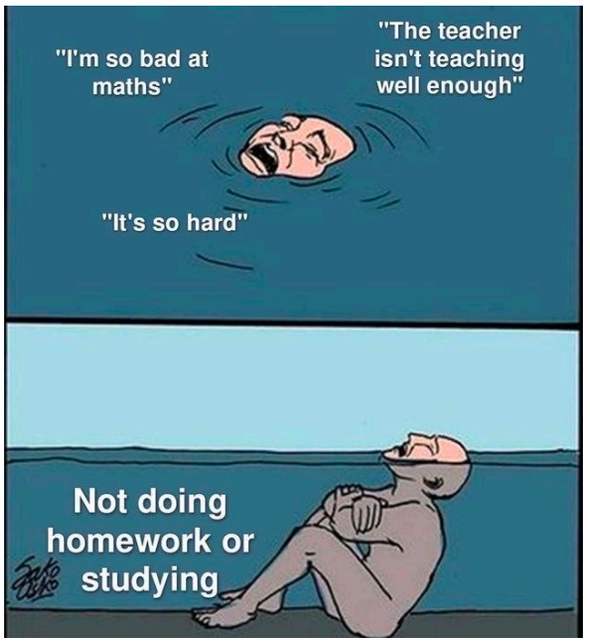
March 11, 2023

DSML : Math for ML.

Optimization 5: Principal Component Analysis.



Recap:

(a) Classification - choosing $\vec{w}$ and w_0.

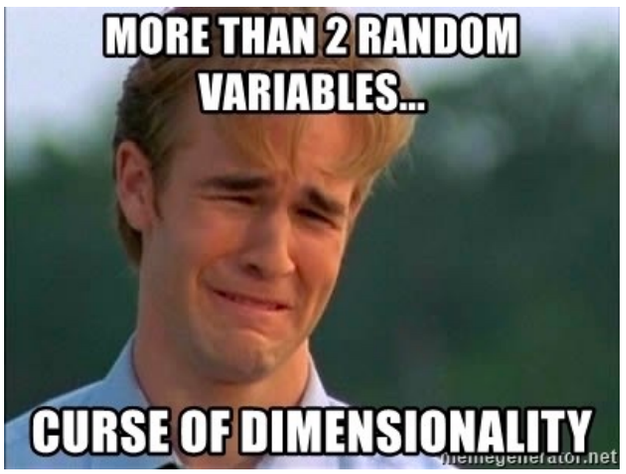
(b) Brute force: very inefficient.

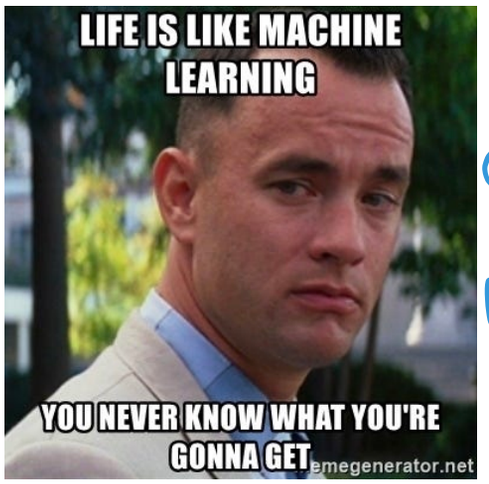
(c) Alternative: Gradient descent

(d) Functions, limits, derivatives.

(e) Partial derivatives and gradients.

(f) Gradient Descent in action.



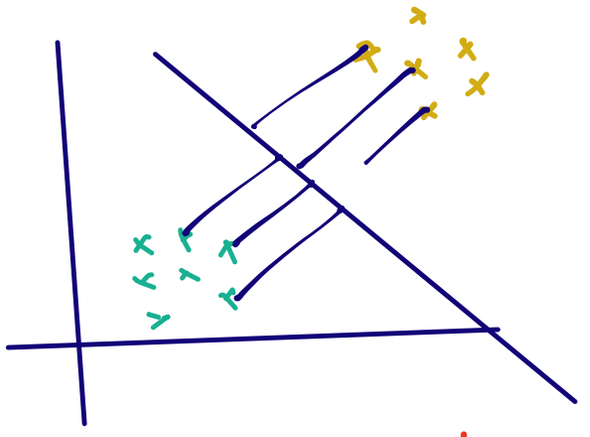


Today:

(a) Variants of Gradient Descent.

(b) Dimensionality Reduction - PCA.

Recap:



* Regularizer.

$$\mathcal{L}(D; \bar{w}, w_0) = -\left(\sum_{i=1}^{n} (\bar{w}^T \bar{x}_i + w_0)\, y_i\right) + \lambda \underbrace{(\|\bar{w}\| - 1)}_{\text{controls the importance}}.$$

$$\bar{w}^{(t+1)} = \bar{w}^{(t)} - \eta \nabla_{\bar{w}} \mathcal{L}(D; \bar{w}, w_0)$$

$$= \bar{w}^{(t)} - \eta \left(- \sum_{i=1}^{n} y_i \bar{x}_i + \lambda \cdot \frac{\bar{w}}{\|\bar{w}\|} \right)$$

$$w_0^{(t+1)} = w_0^{(t)} - \eta \cdot \left(- \sum_{i=1}^{n} y_i \right)$$

Variants of Gradient Descent.

Vanilla G.D.:

$$\bar{w}^{(t+1)} = \bar{w}^{(t)} - \eta \left(- \sum_{i=1}^{n} y_i \bar{x}_i + \lambda \cdot \frac{\bar{w}}{\|\bar{w}\|} \right)$$

$$w_0^{(t+1)} = w_0^{(t)} - \eta \cdot \left(- \sum_{i=1}^{n} y_i \right)$$

$n \rightarrow$ Size of the dataset. $O(n)$.

$$D = \{ (\bar{x}_i, y_i) \}_{i=1}^{n}$$

Affects training time greatly.

$10^7 \rightarrow$ 10 million images.

"Stochastic Gradient Descent"

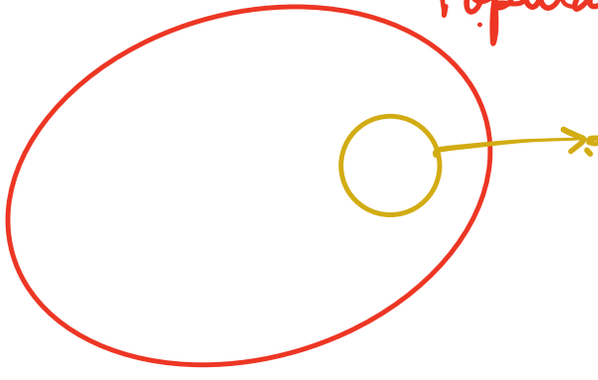
"Stochastic Gradient Descent".

* Pick one $(\bar{x}_i, y_i)$ randomly from our dataset. Use that to do the following update:

$$\bar{w}^{(t+1)} = \bar{w}^{(t)} - \eta \left(- y_i \bar{x}_i + \lambda \cdot \frac{\bar{w}}{\|\bar{w}\|} \right)$$

$$w_0^{(t+1)} = w_0^{(t)} - \eta \cdot (- y_i)$$

$\leftarrow O(1)$.



→ sample, calculate "approximate" mean & confidence intervals.

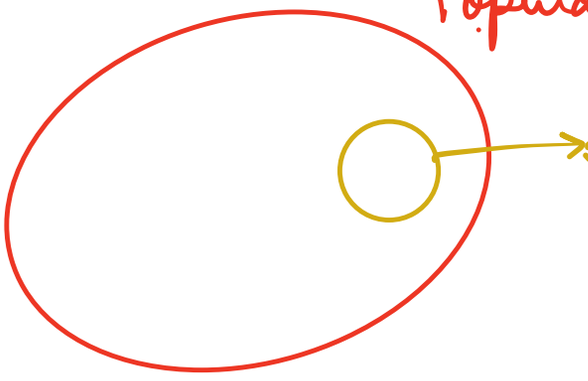
"Batch - Gr. D."

* Pick any k $(\bar{x}_i, y_i)$ randomly from our dataset. Use that to do the following update:

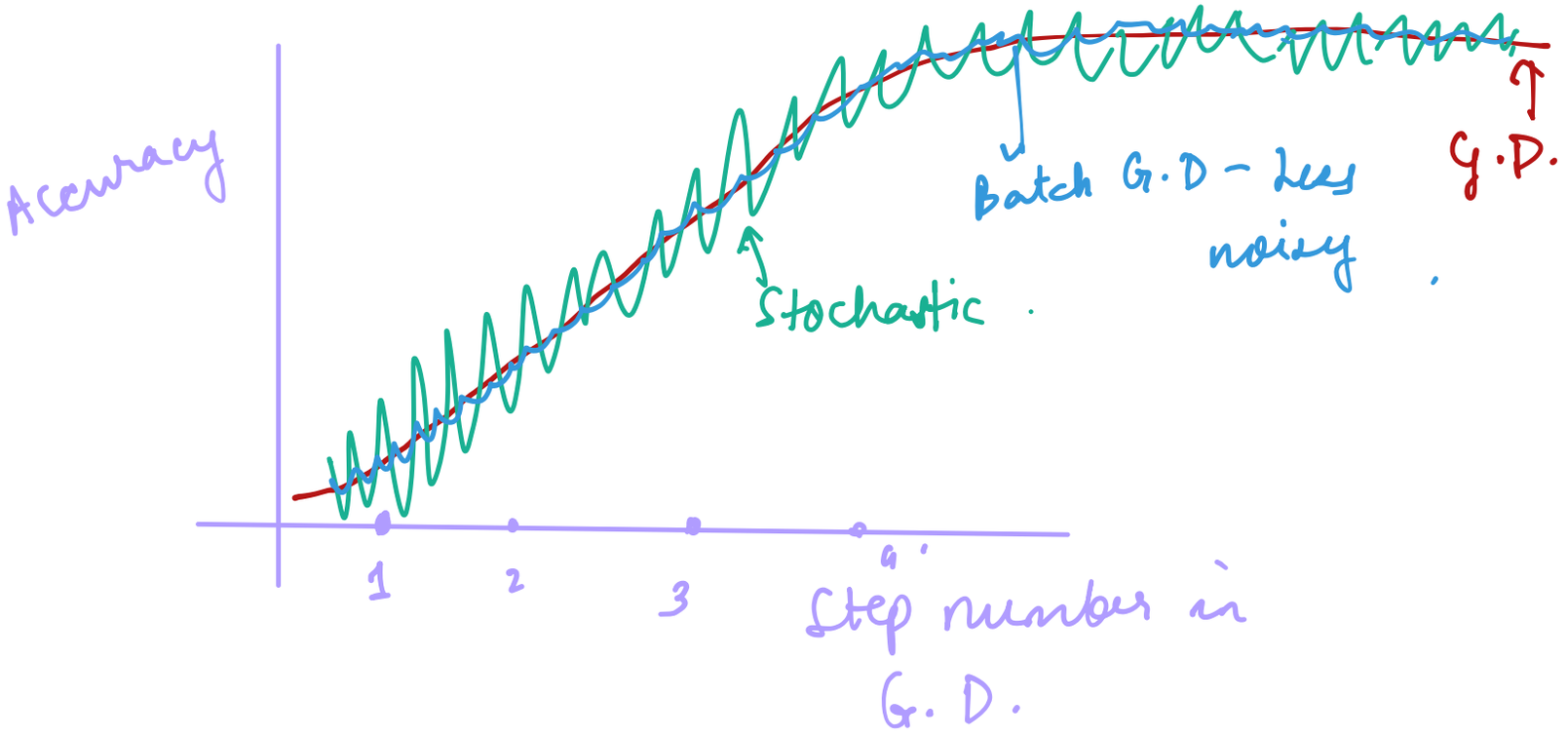
$$\bar{w}^{(t+1)} = \bar{w}^{(t)} - \eta \left(-\sum_{i=1}^{k} y_i \bar{x}_i + \lambda \cdot \frac{\bar{w}}{\|\bar{w}\|} \right)$$

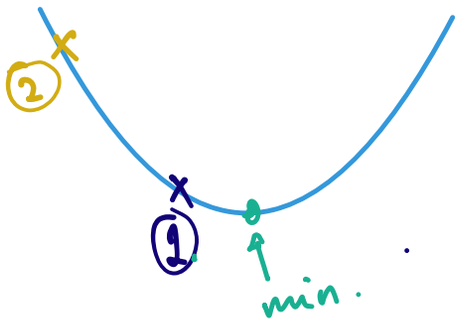
$$w_0^{(t+1)} = w_0^{(t)} - \eta \cdot \left(-\sum_{i=1}^{k} y_i \right)$$

k ≈ 32 to 256.

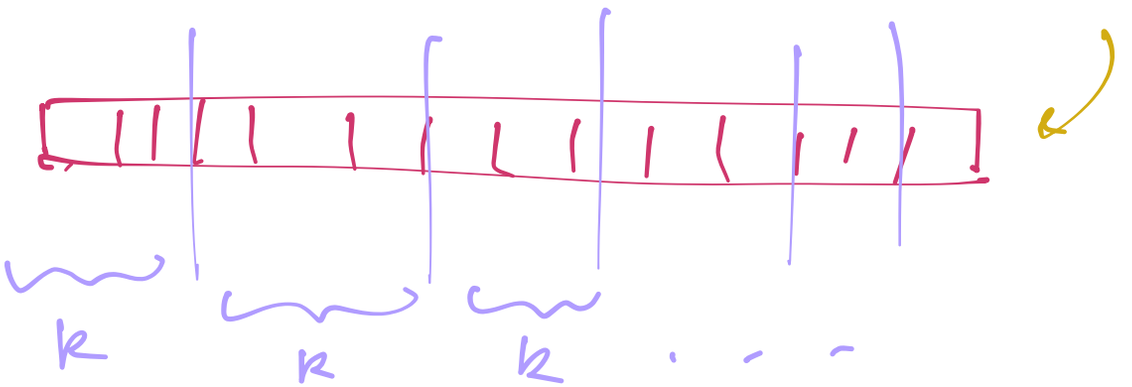


sample, calculate "approximate" mean & confidence intervals.

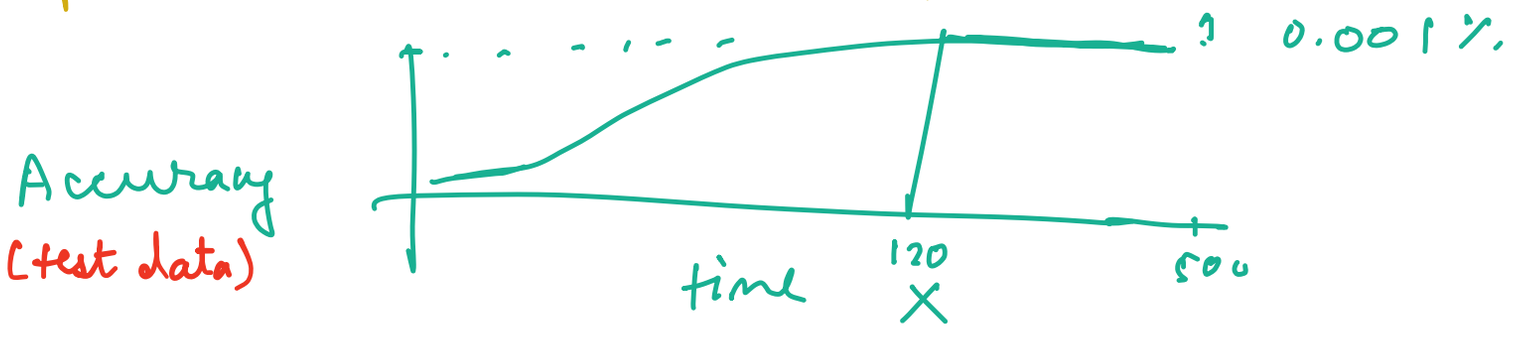

Accuracy
Batch G.D - less noisy
G.D.
Stochastic
1
2
3
4
Step number in G.D.


2
1
min.

* Random permutation of the data is used to implement Batch G.D.



* Using the whole dataset once to update our model → 1 "epoch".



Dimensionality Reduction.

$$D = \{ [\bar{x}_i, y_i) : \bar{x}_i \in \mathbb{R}^{d} \}_{i=1}^{n}$$

d.

Examples:
1. Flatten.
2. Feature Engineering.
3. t-SNE.
4. Autoencoders.

$d \gg 2 \rightarrow$ Very difficult to visualize!!

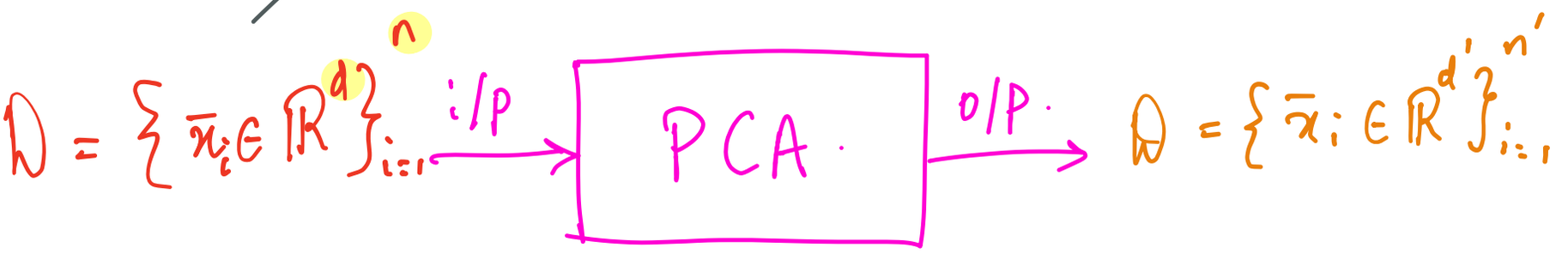
Problems of high dimensional data:

1. Visualization is tough.
2. Training time increases.
3. Computational costs (time/memory).
4. Difficult to work with (Math is harder).

Dimensionality Reduction algorithms help us deal with these issues. Eg: Principal Component Analysis (PCA).

Input / Output Relationship.

unlabelled data.



$>, <, =$

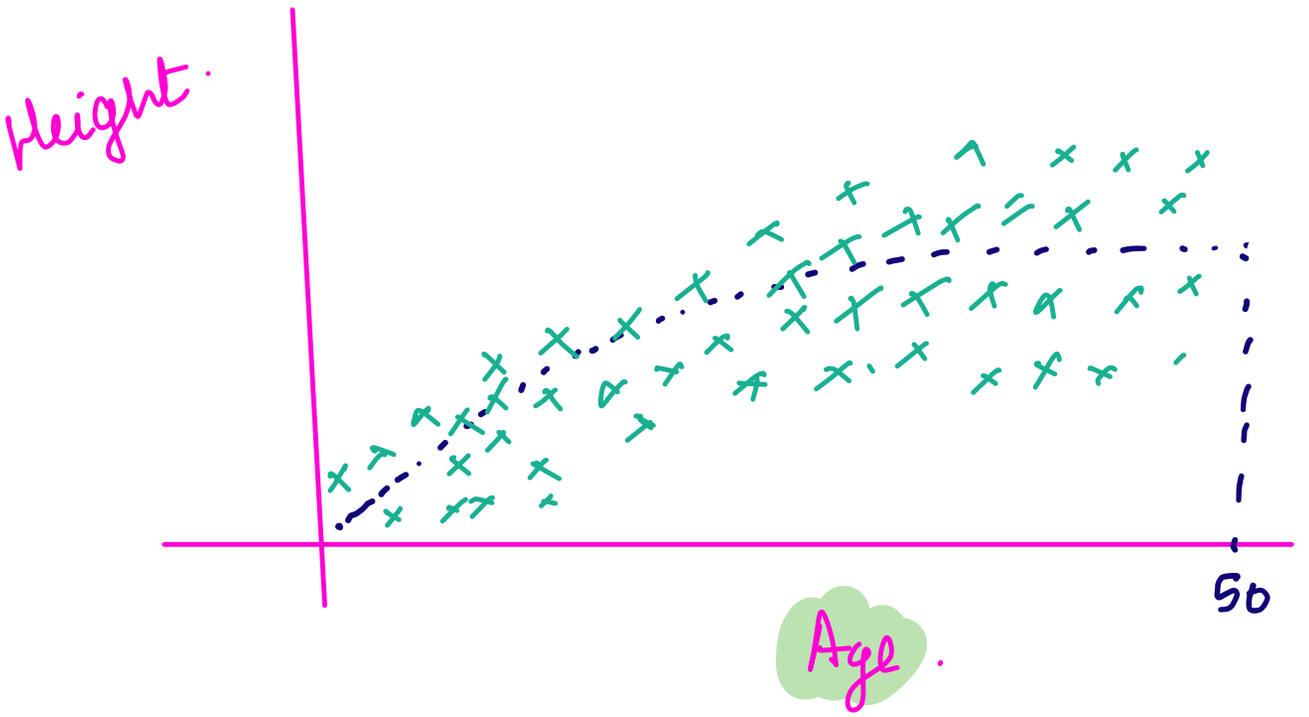
$d > d'$

$n = n'$

This is important because we don't want to lose essential information.

Example 2: Height vs. Age.

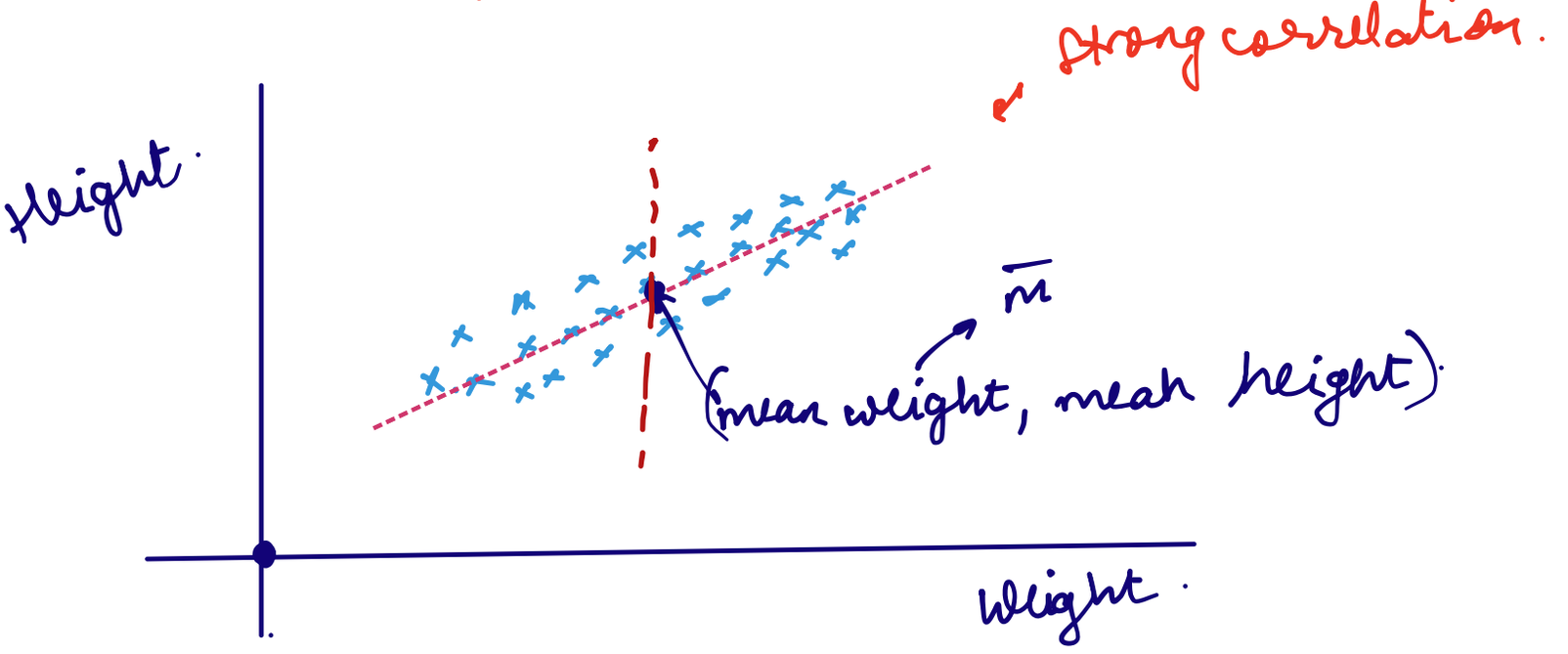
Strong correlation ⇒ magnitude of C.C. > 0.5.



There isn't a strong correlation between age & height.

→ In such cases, it is not possible to do D.R without losing information.

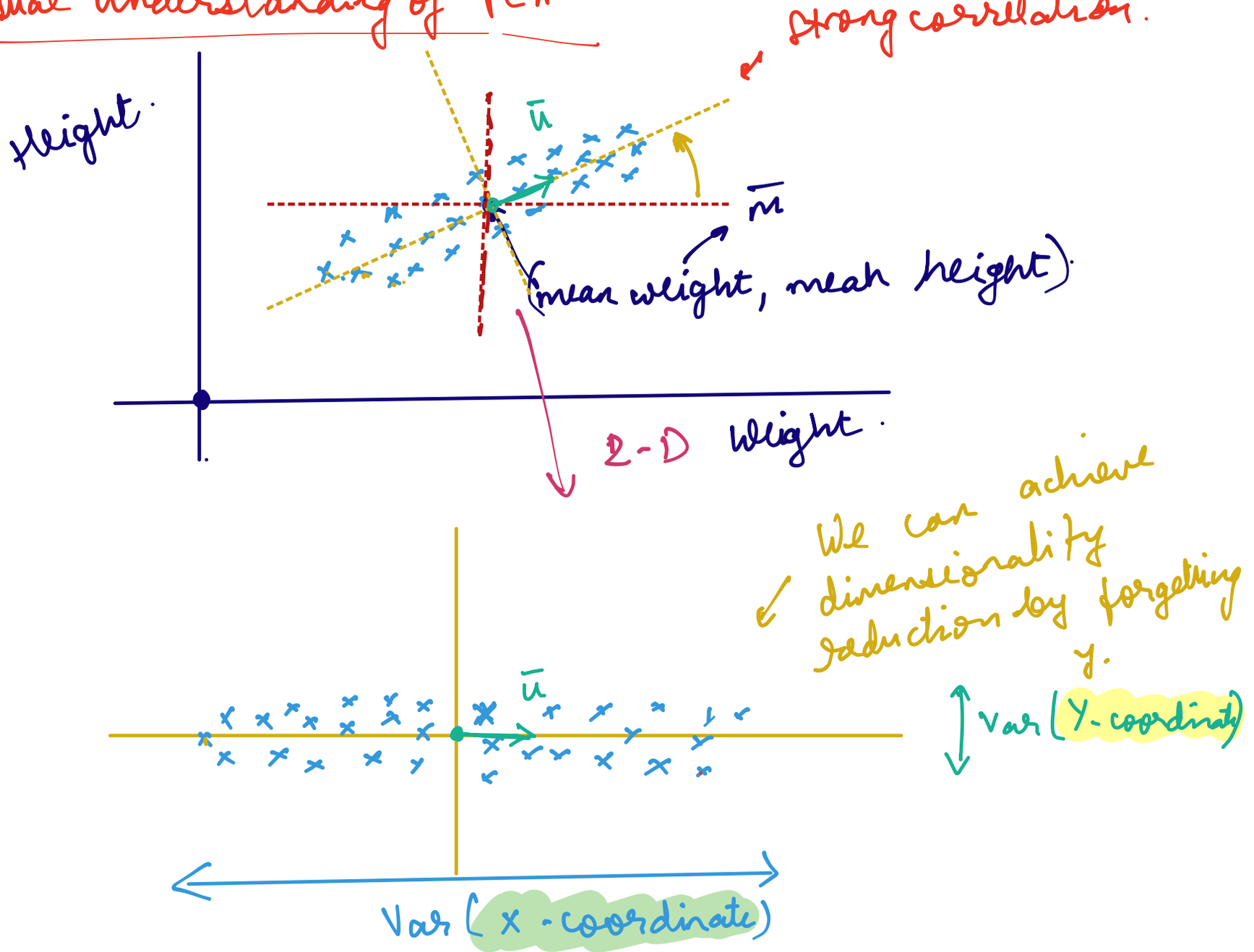
Example : Height vs. Weight.



→ In such cases, Dimensionality reduction is possible.

$D = \{ \bar{x}_i \in \mathbb{R}^d \}_{i=1}^{n}$ $\quad D' = \{ (\bar{x}_i - \bar{m}) \}$

Visual understanding of PCA.

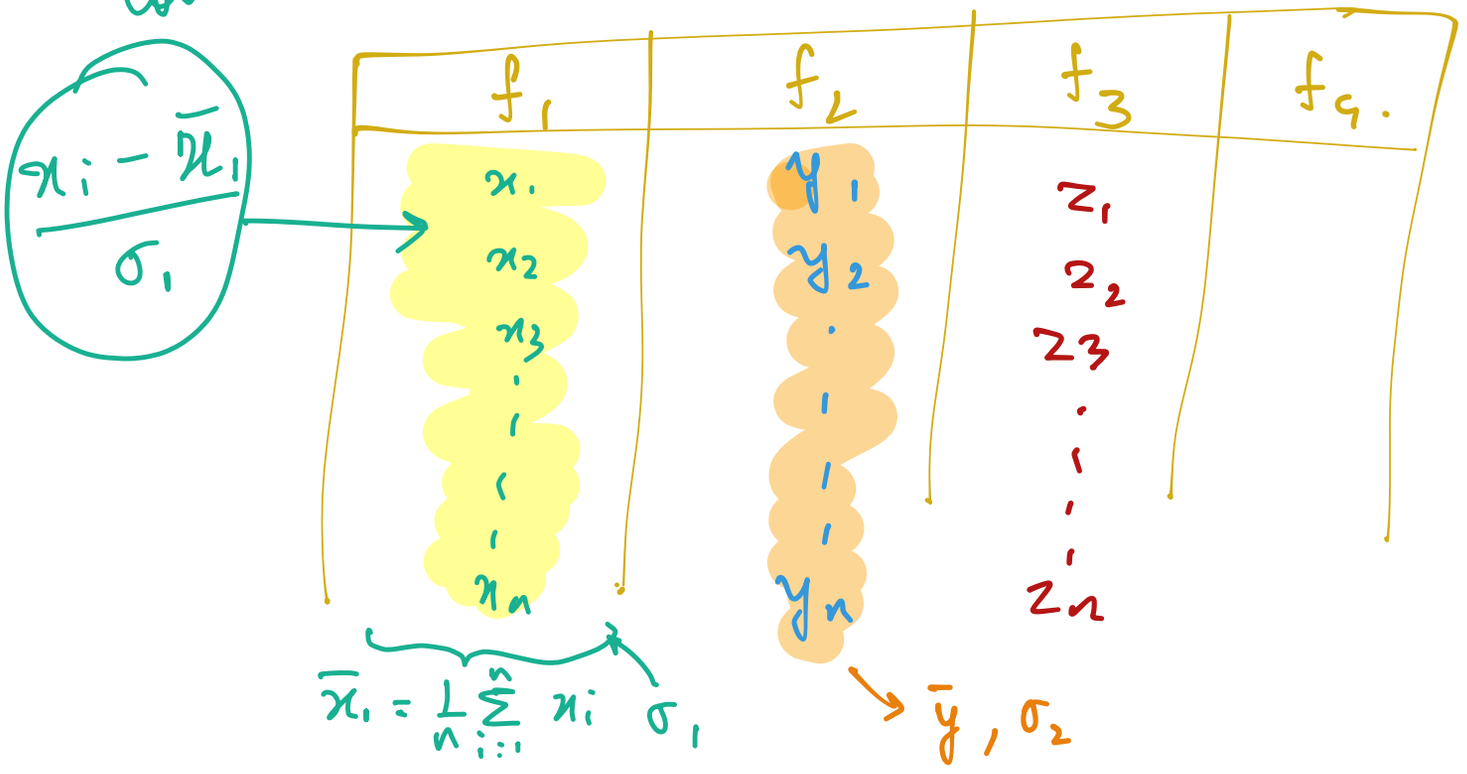


Implementing PCA:

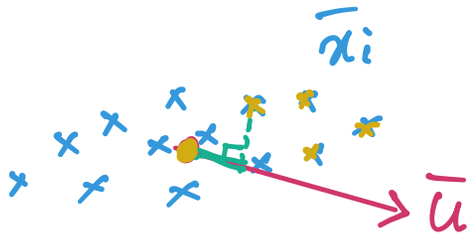
Step 1 : Column standardization.

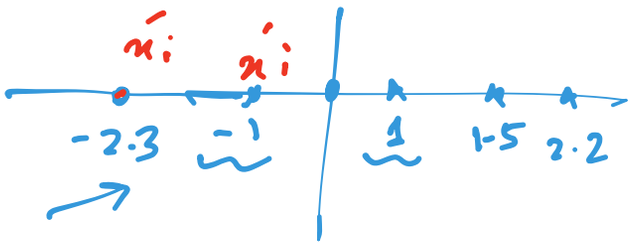
$\frac{x_i - \bar{x}}{\sigma}$

Column standardization - $\frac{y_i - \bar{y}}{\sigma_2}$



Step 2: Find a vector $\bar{u}$ such that the variance of $\bar{x}$ along $\bar{u}$ is maximized.





Projection of $\bar{x}_i$ on $\bar{u}$: $\dfrac{\bar{x}_i^T \bar{u}}{\|\bar{u}\|}$

$$\text{Var}(x_i') = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\bar{x}_i^T \bar{u}}{\|\bar{u}\|} \right)^2$$

x_1' x_2' x_3' x_4' x_5'

$[-2.2, \quad 1.3, \quad 1.7, \quad 0.86, \quad -3.4]$

↓

add. → not going to help.

var? $\frac{1}{n}\sum_{i=1}^{n}(x_i' - \bar{\mu})^2$

$$\Rightarrow \quad \bar{u}^* = \underset{\bar{u}}{\operatorname{argmax}} \; \frac{1}{n} \sum_{i=1}^{n} (\bar{x}_i^T \bar{u})^2$$

$$\text{s.t.} \quad \|\bar{u}\|^2 = 1$$

$\downarrow$ Lagrange multipliers

$$\bar{u}^* = \underset{\bar{u}}{\operatorname{argmax}} \; \frac{1}{n} \sum_{i=1}^{n} (\bar{x}_i^T \bar{u})^2 + \lambda (\|\bar{u}\|^2 - 1)$$

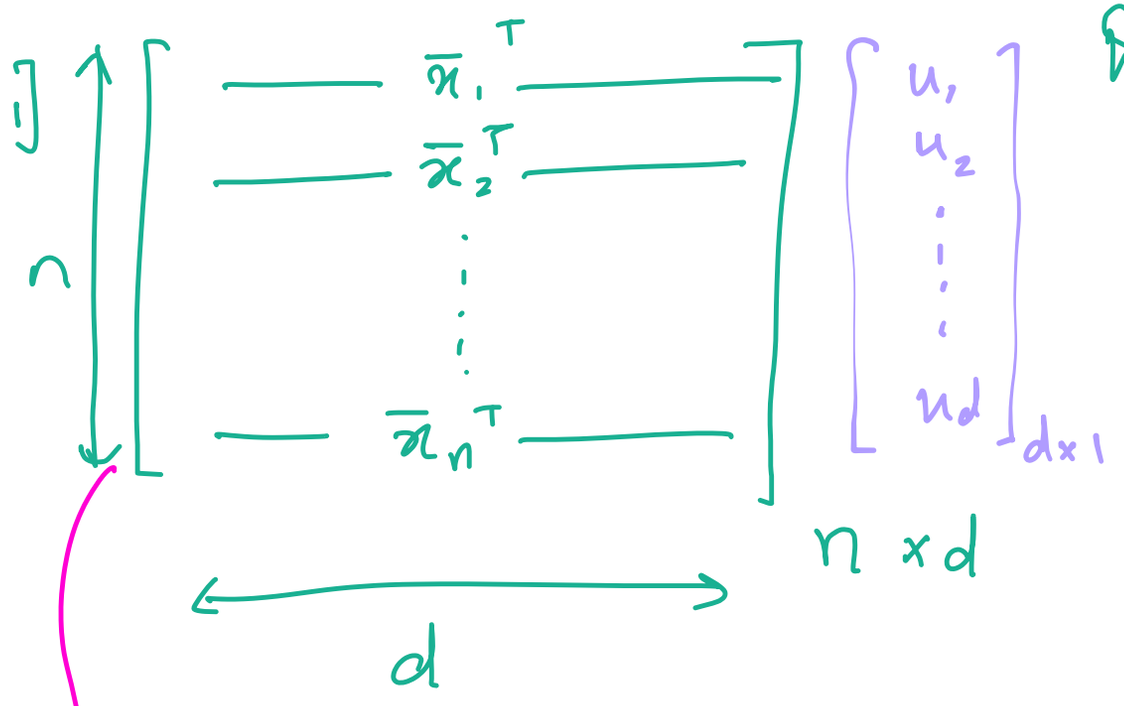
↳ We can stop here and apply Gradient Ascent to find $\bar{u}$.

But, finding a math solution is more efficient.

$$\bar{u}^* = \underset{\bar{u}}{\operatorname{argmax}} \; \frac{1}{n} \sum_{i=1}^{n} (\bar{x}_i^T \bar{u})^2 + \lambda (\|\bar{u}\|^2 - 1)$$

- change how we are seeing this.

$$\mathcal{D} = \{ \bar{x}_i \in \mathbb{R}^a \}_{i=1}^{n}$$



$$X\bar{u} = \bar{v}$$

$$\begin{bmatrix} \bar{x}_1^T \bar{u} \\ \bar{x}_2^T \bar{u} \\ \bar{x}_3^T \bar{u} \\ \vdots \\ \bar{x}_n^T \bar{u} \end{bmatrix} \rightarrow \|\bar{v}\|^2$$

This matrix is called the data matrix.
or X.

$$\bar{u}^* = \underset{\bar{u}}{\operatorname{argmax}} \underbrace{\frac{1}{n} \cdot \bar{u}^T X^T X \bar{u} \cdot + \lambda \left(\|\bar{u}\|^2 - 1 \right).}_{f(\bar{u})}$$

$$\bar{v} = X \bar{u}$$

$$\bar{v}^T \bar{v} = (X\bar{u})^T (X\bar{u})$$

$$= \underbrace{\bar{u}^T X^T X \bar{u}}$$

Why did we do this? to make taking the gradient easier!!

→ Fact 1: $f(\bar{u}) = \bar{u}^T S \bar{u}$ $S \rightarrow$ any matrix

then $\nabla_{\bar{u}} f(\bar{u}) = (S + S^T)\bar{u}$

$$f(\bar{u}) = \frac{1}{n}\,\bar{u}^T (X^T X)\,\bar{u} + \lambda(\bar{u}^T\bar{u} - 1)$$

Fact 1:

$$\nabla_{\bar{u}} f(\bar{u}) = \frac{1}{n}\left((X^T X) + (X^T X)^T\right)\bar{u} + \lambda(2\bar{u})$$

$$\rightarrow \text{Since } (X^T X)^T = X^T X \cdot + \lambda(2\bar{u})$$

$$= \frac{2}{n} X^T X\,\bar{u} + \lambda(2\bar{u})$$

$$\nabla_{\bar{u}}\, \bar{u}^T\bar{u} = 2\bar{u}$$

$$\frac{\partial x^2}{\partial x} = 2x$$

$$\nabla_{\bar{u}} f(\bar{u}) = 2\left(\frac{1}{n} X^T X \bar{u} + \lambda \bar{u}\right) = 0 .$$

$$\Rightarrow \frac{1}{n} X^T X \bar{u} = -\lambda \bar{u}$$

$$\Rightarrow X^T X \bar{u} = \underbrace{-(n\lambda)}\ \bar{u}$$

$$\boxed{(X^T X)\bar{u} = \lambda' \bar{u} .} \quad \textcircled{1}$$

Eigenvalue, Eigenvector expression!!

The vector $\bar{u}$ which maximizes $f(\bar{u})$ is the one which satisfies ①.

By observing ①, we notice that
$\bar{u}$ is nothing but the eigenvector
of the matrix $X^T X$.

What is an Eigenvector?

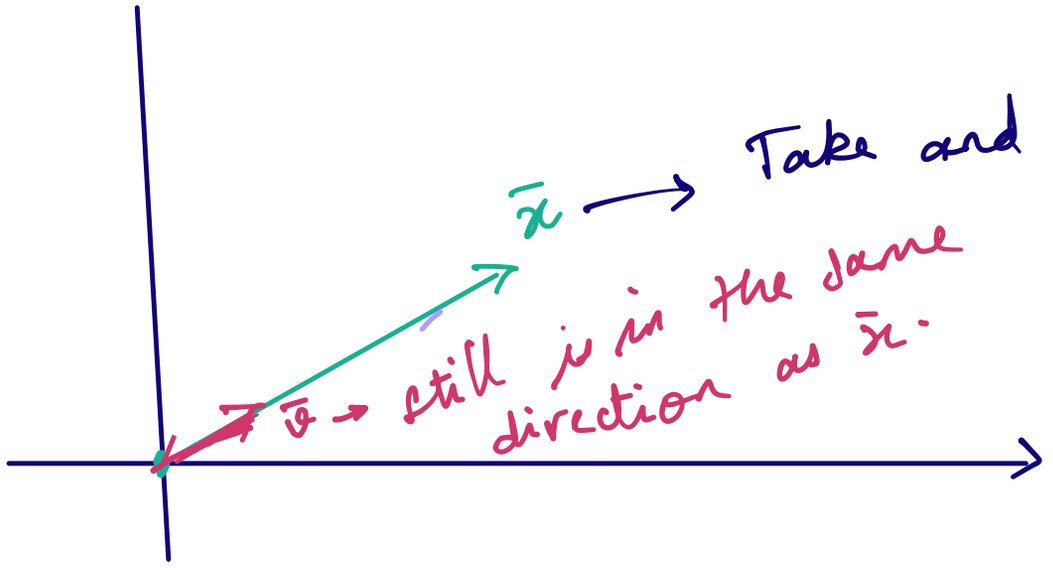
"eigen" → German word ≈ identity. ↓ same direction

For any square matrix S,
↳ no. of rows = no. of cols.

if we find an $\bar{x}$ such that

$$S\bar{x} = \lambda \bar{x}$$

then $\bar{x}$ is said to be an "Eigenvector".



$$\underline{S_{d\times d}} \quad \underline{\bar{x}_{d\times 1}} = \underline{\bar{v}_{d\times 1}}$$

Result (of step 2): The vector $\bar{u}$ which captures max. variance is the eigenvector of the matrix $X^T X$.

How to find this? → numpy

Each matrix has many eigenvectors,
Each eigenvector has a corresponding eigenvalue.

So, we pick the eigenvector corresponding to the maximum eigenvalue for our $\bar{u}$.

Step 3: Other directions?

If we want to reduce

$$d \longrightarrow d'$$

we simply pick the first d' eigenvectors with the largest corresponding eigenvalues.

$3 \times 3 \rightarrow (\bar{x}_1, \lambda_1), (\bar{x}_2, \lambda_2), (\bar{x}_3, \lambda_3)$

$\lambda_1 = 0.5$, $\lambda_2 = 0.3$, $\lambda_3 = 0.2$.

$\downarrow$

2

Pick $(\bar{x}_1, \bar{x}_2)$, $(0.5, 0.3)$.

These are the new axes.

Step 4: How to convert $\bar{x}_i \in \mathbb{R}^d$ to $\bar{x}_i' \in \mathbb{R}^{d'}$:

$$\frac{\bar{x}_i^T \bar{u}_1}{\|\bar{u}_1\|} + \frac{\bar{x}_i^T \bar{u}_2}{\|\bar{u}_2\|} \cdots + \frac{\bar{x}_i^T \bar{u}_{d'}}{\|\bar{u}_{d'}\|}.$$

$\downarrow$ coordinate along $\bar{u}_1$

coordinate along $\bar{u}_2$ ...

Step 5: How to calculate the information loss?

$$\frac{(\lambda_1 + \lambda_2 + \cdots + \lambda_{d'})}{\sum_{i=1}^{d} \lambda_i} \rightarrow \text{tells us the fraction of info-retained.}$$