* Google's deep mind (paper).
→ Weights & Biases (WANDB)

→ 10 class classification problem (single label)
→ Throw away ground truth
→ Assign random (labels)

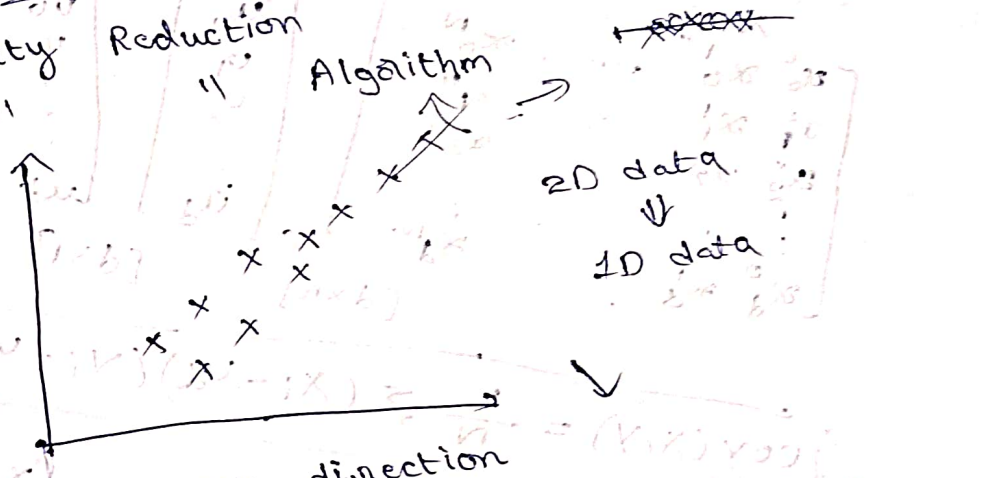
When we train, what we will get?
For validation → 10%
For training → (100%)

→ If we mislabel only 50% of labels:-
For validation - (70%)
→ Never look at Accuracy in train dataset.

21/02/25

Principle Component Analysis (PCA)

→ Dimensionality Reduction
→ Linear " Algorithm



→ project into one direction

$U^T_{d \times d} X_{dXN}$

$U^T_{1X2} X_{2XN}$

$U^T_{10X1000} X_{1000XN}$

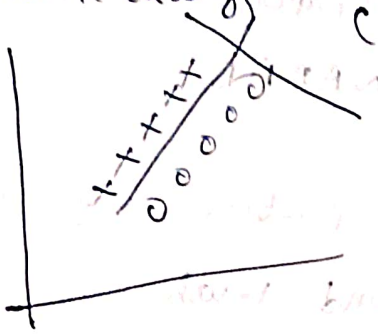
→ We need to project to vector U, after computing variance should be maximized.

$\max_{U_1} (\text{var}(U_1^T X))$

(Maximize variance of projected data)

→ PCA doesn't work always (XOR)

Ex:-



→ If the many features are linearly dependent, PCA will make it to one vector.

$$\text{Variance}(v) = E\left((\vec{x} - \vec{u})(\vec{x} - \vec{u})^T\right)$$

$$[x_1 - u \quad x_2 - u \ldots x_n - u] \begin{bmatrix} x_1 - u \\ x_2 - u \\ \vdots \\ x_n - u \end{bmatrix}$$

$$\text{Covariance}(S) = E\left[(X - M)_{d \times N} (X - M)^T_{N \times d}\right]$$

$$\begin{bmatrix} x_0^0 & x_0^1 & \cdots & x_0^N \\ x_1^0 & x_1^1 & & x_1^N \\ x_2^0 & x_2^1 & & x_2^N \\ \vdots & \vdots & & \vdots \\ x_d^0 & x_d^1 & & x_d^N \end{bmatrix}_{[d \times N]} \quad \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_d \end{bmatrix}_{[d \times 1]} \quad \begin{bmatrix} u_0 & u_0 & \cdots & u_0 \\ u_1 & u_1 & \cdots & u_1 \\ \vdots & & & \\ u_d & u_d & \cdots & u_d \end{bmatrix}_{[d \times N]}$$

$$\boxed{\text{cov}(X,Y) = \frac{1}{N} \sum (X_i - u_x)(Y_i - u_y)}$$

$$X - M \qquad [X - M]^T$$

$$\begin{bmatrix} x_0^0 - u_0 & x_0^1 - u_0 & \cdots & x_0^N - u_0 \\ x_1^0 - u_1 & x_1^1 - u_1 & \cdots & x_1^N - u_1 \\ \vdots & & & \\ x_d^0 - u_d & x_d^1 - u_d & \cdots & x_d^N - u_d \end{bmatrix}_{d \times N} \quad \begin{bmatrix} x_0^0 - u_0 & x_1^0 - u_1 & \cdots \\ x_0^1 - u_0 & x_1^1 - u_1 & \cdots \\ \vdots & & \\ x_0^N - u_0 & x_1^N - u_1 & \cdots \end{bmatrix}_{N \times d}$$

$$= \begin{bmatrix} \sigma & & \\ & \ddots & \\ & & \sigma \end{bmatrix}_{d \times d}$$

solve

student	Phy	chem
A	50	45
B	40	85
C	90	80
	Mean = 60	70

$$\begin{bmatrix} & \\ & \end{bmatrix}$$

$$\max_{u_1} (\text{var}(u_1^T X))$$

$$\text{var}(u_1^T X) = E\left[[u_1^T x - E(u_1^T x)][u_1^T x - E(u_1^T x)]^T\right]$$

$$= E\left[(u_1^T x - u_1^T u)(u_1^T x - u_1^T u)^T\right]$$

($\because E[x] = u$, $u_1^T \rightarrow$ scalar)

$$= u_1^T E[(x-u)(x-u)^T] u$$

$$= u_1^T S u \quad \text{(quadratic function in } u\text{)}$$

↓

Can we maximize this?

$u_1 \rightarrow$ is direction, unit vector } we can get infinite val

We need to put a constraint

$$\text{var}(u_1^T x) = \max_{u_1} (u_1^T S u_1) \quad \text{s.t } u_1^T u_1 = 1$$

Lagrange Multipliers

maximizing this equals to,

$$\max_{u_1} (u_1^T S u_1 - \lambda(u_1^T u_1 - 1))$$

Now differentiate with respect to u_1

$$S u_1 - \lambda u_1 = 0$$

$$S u_1 = \lambda u_1$$

We want to maximize $u_1^T S u_1$

maximize $u_1^T \lambda u_1$

maximize (λ) $\quad u_1^T u_1 = 1$

→ compute covariance matrix S of given data

→ " all eigen values & vectors of S
(singular value decomposition)

for $d \times d$ → we will get d eigen values & d eigen vectors

from $d \to 1$ take first largest eigen value and pick corresponding eigen vector

$d \to 10$ take first 10 " "

$U^T \times$
10×1000 $1000 \times N$

→ First eigen value first → principle component vector

→

Drawback

① → Linear algorithm

To how many dimensions should we reduce?

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{d} \lambda_i}$$

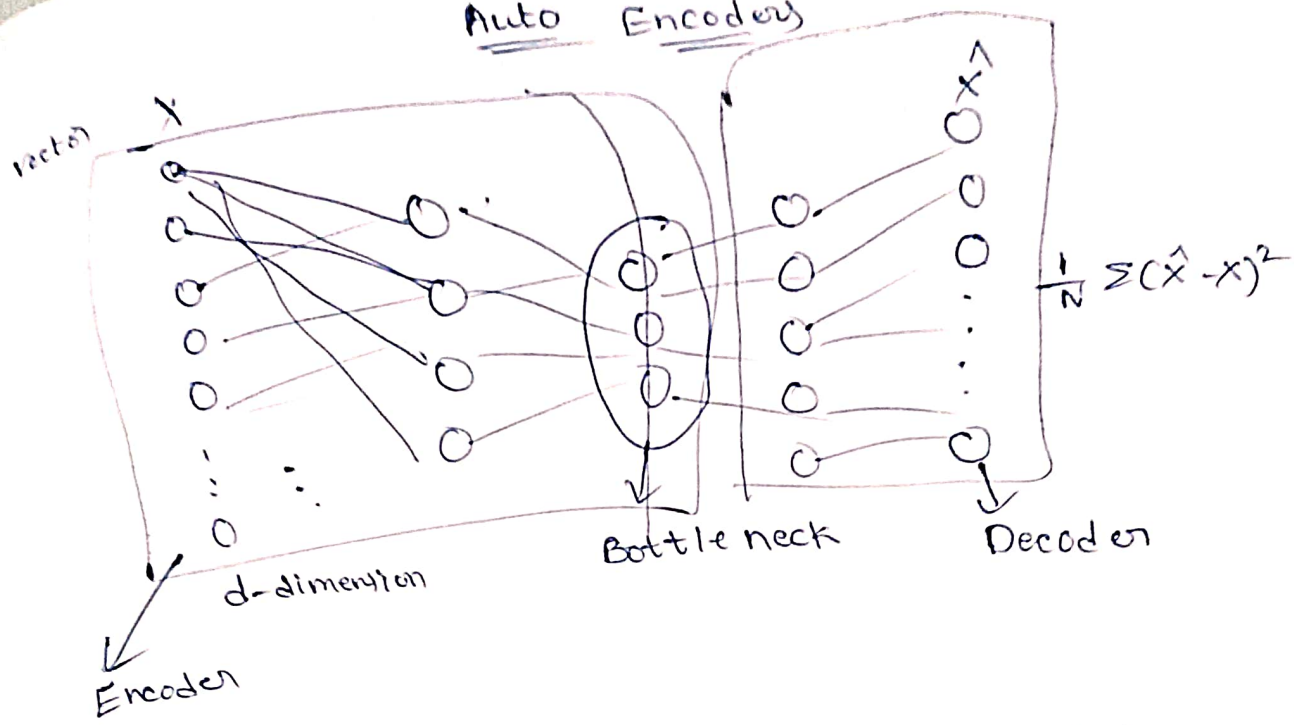
Pick first k which gives 95% ratio
(Heuristic)

② Task dependent

~~Auto~~ ~~encoders~~

t-sne : To use only if we want visualize the data and not for dimensionality reduction.

Auto Encoders



Applications

- Dimensionality reduction (For non-linear) also
- Representation (For nearest neighbour)
- Encryption
- MAST Auto Encoders are scalable vision learners
- Denoising Auto Encoders
- Predicting Missing data
  RGB data
- Translation network