CH- Dimensionality Reduction and visualization

* Dimensionality reduction:- We learned that we can use scatter plot for 2-D, 3-D, for 4-D, 5-D, we can use pair plot, But for n-D, we try to reduce the Dimensionality of the data.

→ Row vector and column vector :-

$x_i \in R^d$ → d-dim, column vector

↳ Real Number

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix}$$

→ Dataset :- $D = \{ x_i, y_i \}_{i=1}^{n}$ $\quad x_i \in R^d$, $y_i \in$ {setosa, versicol, verginica}

x_i → data points, y_i → class label, n → data set contain n data points

x_i is d dim vector of Real values.

e.g. In iris data set $x_i \in R^4$

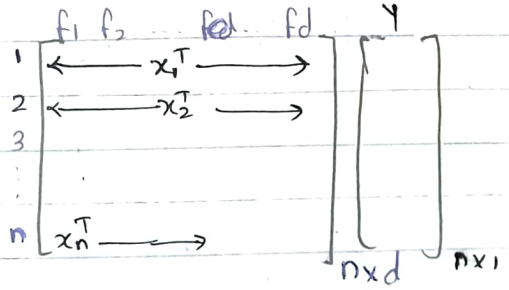
→ Data set as a data-matrix

$D = \{ x_i, y_i \}_{i=1}^{n}$

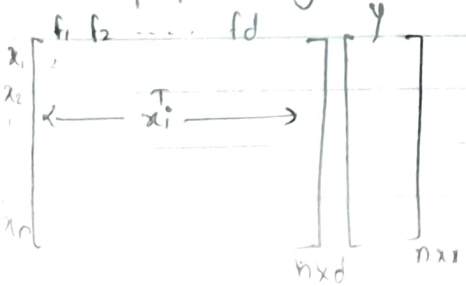
$x_i \in R^d$

$y_i \in$ {Set, Veg, Ver}

$\Rightarrow$

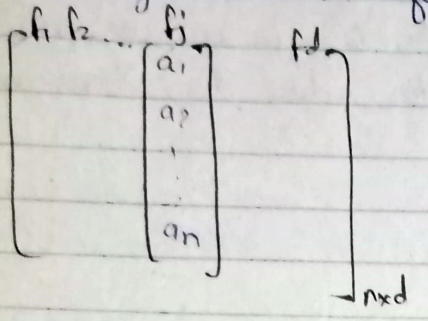


→ Data preprocessing : column-normalization



preprocessing refers some type of mathematical operation performed on data before doing real task. Colum-normalization is one of the preprocessing technique.

Colomen normalization:- We know each colomn represent feat
, take a colomn, and take all the value of that
change then as follows.



take max value of ai in fj

$f_j = [a_1\ a_2 \dots a_n]$ (minvalue, max value)

a_{min} a_{max}

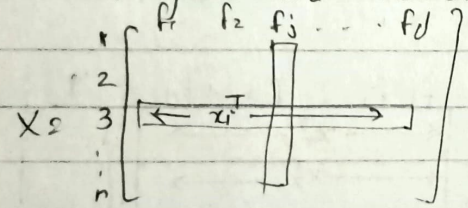
$$a_i' = \frac{a_i - a_{min}}{a_{max} - a_{min}}$$

by doing this we are ensuring that $a_i' \in [0$

Do this for each colomn.

But why are we doing this? By doing this we are getting rid
scale, every thing comes in [0,1] scale.

★ Mean of Data Matrix:-



$x_i \in R^d$

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

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

$x_1 + x_2 = [3.4, 7.4]$

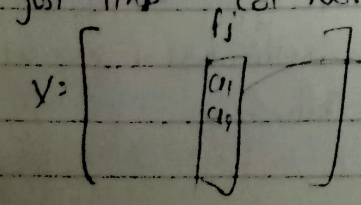
$\bar{x} = \left[\frac{3.4}{2}, \frac{7.4}{2}\right]$

$$\therefore \bar{x} = \frac{1}{n}(x_1 + x_2 + \dots + x_n)$$

↳ Each $x_i \in R^d$

★ Data preprocessing: Colomen Standardisation.

We already seen a preprocessing technique i.e. colomen stand
col-stad is more often used than normalisatio
just like Col Nor. we scale all colom one by on



$\rightarrow [a_1\ a_2 \dots a_n]$

↓ col-std

$[a_1'\ a_2' \dots a_n']$ ← mean of $\{a_i\}_1^n$

$\sigma = 1$

In column standardization we convert the column such that the mean of converted data = 0 & $\sigma = 1$

Let's see how to do it

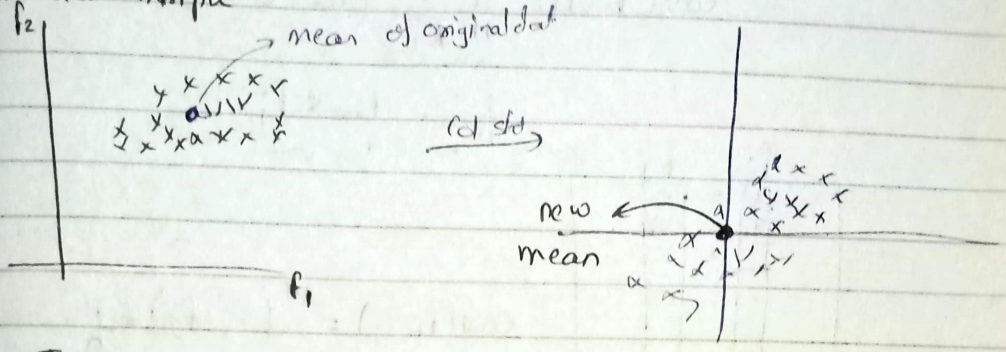
Let $\bar{a} = \text{mean}\{a_i\}_{i=1}^{n}$ ← Sample mean

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

$$a_i' = \frac{a_i - \bar{a}}{s}$$

→ This way we are guaranteeing that mean of a_i' = 0 and $\sigma = 1$

Geometrical Intuition



The mean moves to origin and σ of any axis becomes 1.

* Covariance of Data matrix

Let our Data matrix

$$D = \begin{bmatrix} f_1 & f_2 & \cdots & \cdots & f_d \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}_{n \times d}$$

for this Data matrix we have a covariance matrix of size $d \times d$

$$\xrightarrow{\text{cov}} S = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}_{d \times d}$$ → square matrix

$S_{ij} = \text{cov}(f_i, f_j)$ → S_{ij} = covariance of feature i & feature j ?

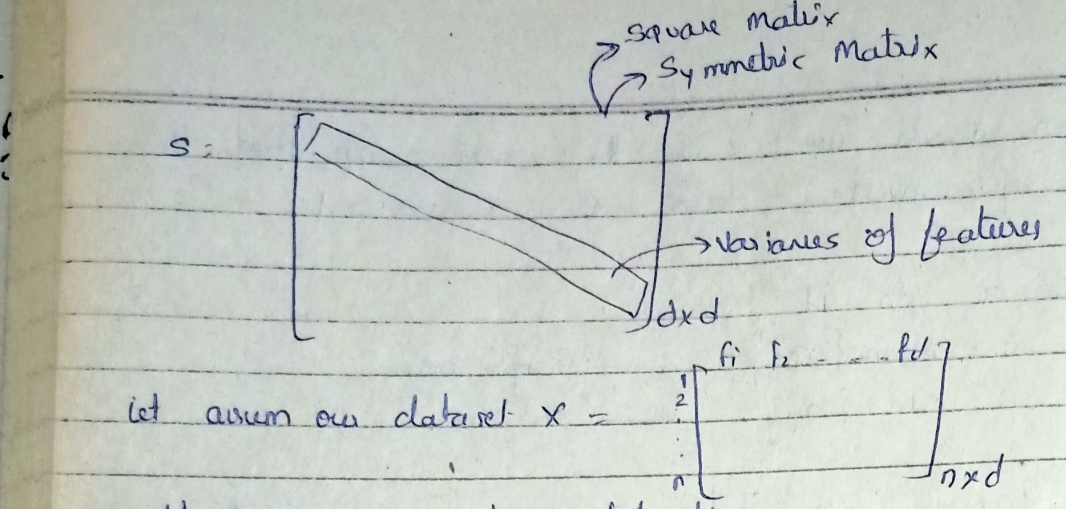
$i: 1 \to d$

$j: 1 \to d$

We know that $\text{cov}(X, Y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)$

$\text{cov}(X, X) = \text{var}(X)$

$\text{cov}(f_i, f_j) = \text{cov}(f_j, f_i)$.



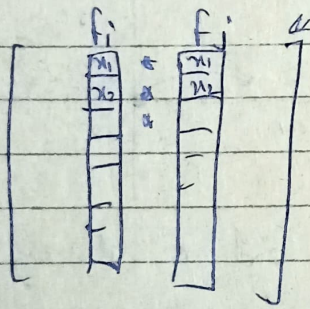
let assum our dataset X =

let suppose x is column standardize, that mean mean of each col = 0 & $\sigma = 1$

hence

$$Cov(f_i, f_j) = \frac{1}{n}\sum_{k=1}^{n} (f_{ik} - 0)(f_{jk} - 0)$$

$$= \frac{1}{n}\sum_{k=1}^{n} f_{ik} * f_{jk}$$

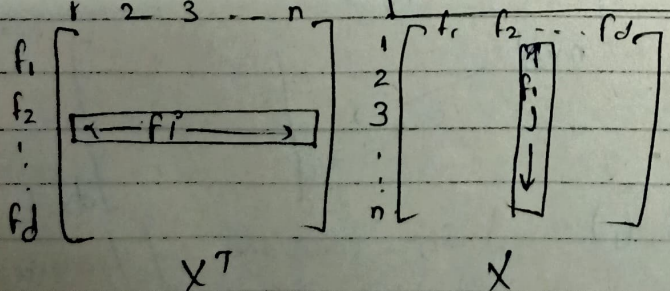


$$cov(f_i, f_j) = (f_i^T f_j) * \frac{1}{n}$$

if f_i & f_j have been standardi then:

$$cov(f_i, f_j) = \frac{f_i^T f_j}{n}$$

Hence If data set is column stardize then covarian matrix, $S_{dxd} = \frac{1}{n}(X^T)X$



X^T X

$$X^T * X = \begin{bmatrix} f_1*f_1 & f_1*f_2 & f_1*f_d \\ f_2*f_1 & f_2*f_2 & f_2*f_d \\ \vdots & & \end{bmatrix}$$

This is not But S

Colah.github.io

Note: $X = \begin{bmatrix} & f_1 & f_2 & f_3 & \dots & f_d \\ 1 & & & & & \\ 2 & & & & & \\ 3 & & & & & \\ \vdots & & & & & \\ n & & & & & \end{bmatrix}$

$S = \text{cov\_matrix} = \begin{bmatrix} & f_1 & f_2 & \dots & f_d \\ f_1 & & & & \\ f_2 & & & & \\ \vdots & & & & \\ f_d & & & & \end{bmatrix}$ $d \times d$

→ Square sym matrix

→ variances of features

If X is not column standardized

$$S_{ij} = \frac{1}{n} \sum_{k=1}^{n} (f_{ik} - \text{mean}(f_i))(f_{jk} - \text{mean}(f_j))$$

If X is column standardized

$$S = \frac{1}{n}(X^T)(X)$$

★ MNIST dataset

- Dataset of hand written digits.
- 28 x 28 pixel images.
- 60 K training data points
- 10 K test data points

$D = \{x_i, y_i\}_{i=1}^{60K}$ $x_i = \begin{bmatrix} & \\ & \end{bmatrix}_{28 \times 28}$ $y_i = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

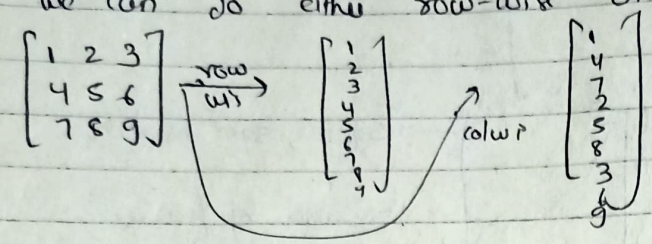
Task: given a new 28 x 28 pixel image classify that in one of the (0-9) character.

We learned the x_i should be vector, but we have x_i is 28 x 28 pixel. So how do we convert x_i to a 1D col vector.

we do flattening $\begin{bmatrix} 1 & 2 & 3 & \dots & 28 \\ 1 & & & & \\ 2 & & & & \\ 3 & & & & \\ \vdots & & & & \\ 28 & & & & \end{bmatrix} \rightarrow \begin{bmatrix} [\text{row1}] \\ [\text{row2}] \\ \vdots \end{bmatrix}$

Ex: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \end{bmatrix}$

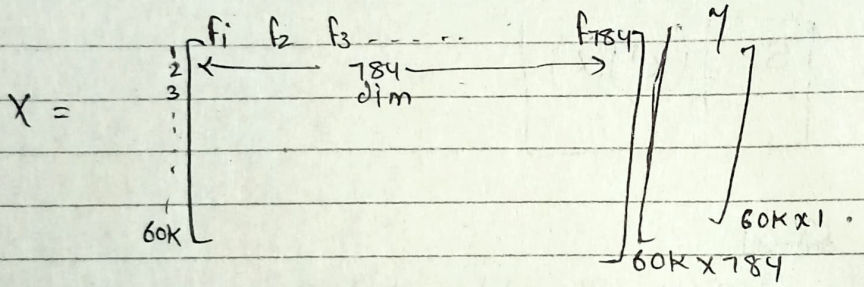
We can do either row-wise or colum wise.



In our MNIST data Now our $[\ ]_{28\times 28}$

row flatten we get $[\ ]_{784\times 1}$ size colum vector.

Now over all data set looks like.



⭐ PCA (Principal Component analysis (PCA))

PCA is basically used for many other thing but one of its application is Dimensionality Reduction

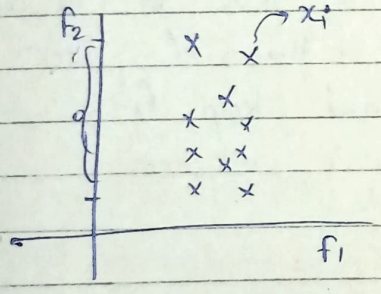
Data given d-dim $\longrightarrow$ d'-dim
s.t $d' < d$

for Ex:- MNIST $\longrightarrow$ 2D
784

we will also see other applications of dimensionality reduction, for now we see only visualization point of you.

→ Geometric interpretation of PCA:-

Let for example 2-D → 1-D



$$X = \begin{matrix} \\ 1 \\ 2 \\ 3 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_1 & f_2 \\ & \\ & \\ & \\ & \\ & \end{bmatrix}$$

Let f_1 = Hairness of hair $\in R$
f_2 = height of people

Indian data

One thing we can note is height have larger spread. while f_1 have small variance.

Now, given this 2D dataset, we want to convert it to 1D. As spread of f_1 is less, we can skip the f_1 and make another 1D dataset of only f_2

$$X = \begin{matrix} \\ 1 \\ 2 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_1 & f_2 \\ & \\ & \\ & \end{bmatrix} \longrightarrow X' = \begin{matrix} \\ 1 \\ 2 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_2 \\ \\ \\ \end{bmatrix}$$

Now suppose another example:

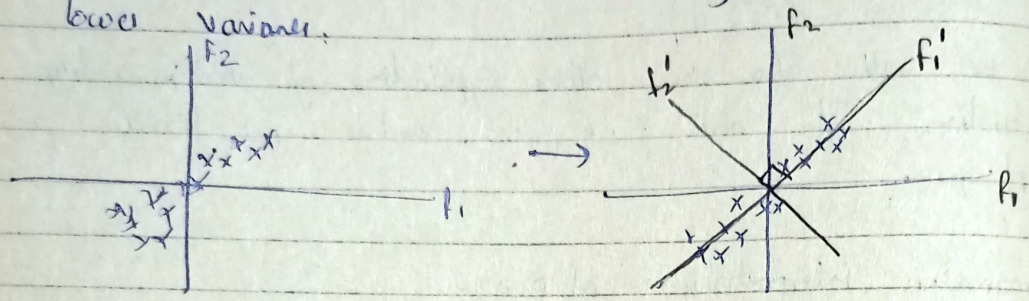
X = 2-dim colum standize dataset

$$X = \begin{bmatrix} f_1 & f_2 \\ & \\ & \end{bmatrix}$$

$$\text{mean}\{f_1\} = \text{mean}\{f_2\} = 0$$

$$\sigma\{f_1\} = \sigma\{f_2\} = 0$$

So, in this case we cannot simply drop one with lower variance.



Let's draw another set of axes like above

Now f_2' << variance than f_1' so we can simply drop f_2' and keep f_1'

step1: find $f_1' \perp f_2'$

② drop f_2'

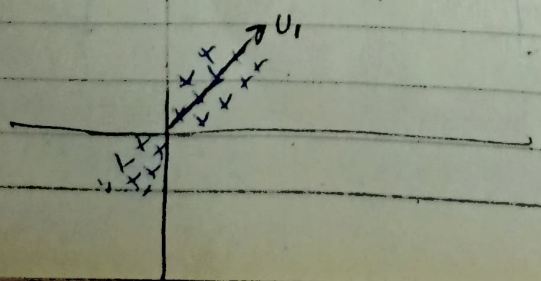
③ project x_i's into f_1' then 2D → 1D

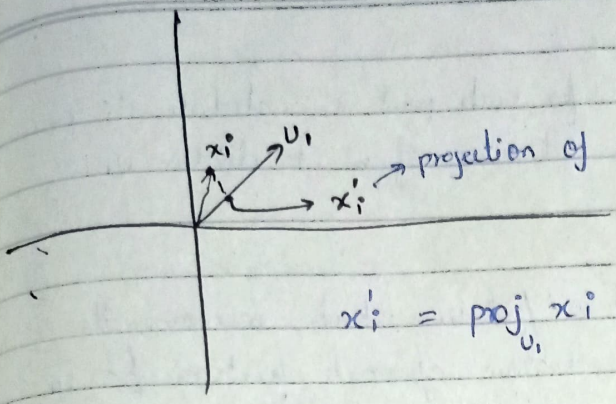
→ we want to find a direction f_i' s.t variance of x_is projected onto f_i's is maximal.

Now, let's try to write it in mathematical form.

Let $f' = u_1$

Second thing is we really don't need the whole f' line we just need direction or we can say a unit vector.





$$x_i' = \text{proj}_{u_1} x_i = \frac{u_1 \cdot x_i}{\|u_1\|^2}$$

$$= u_1 \cdot x_i \quad \{ \because \|u_1\| = 1 \}$$ (unit vector)

$$= u_1^T x_i$$

so each of $x_i' = u_1^T x_i$

$$\bar{x}' = u_1^T \bar{x}$$

($\bar{x}'$: mean of x_i'; $\bar{x}$: mean of x_i)

So, find u_1 s.t the variance of points x_i projected on u_1 is maximal.

$$\text{Var}\{\text{proj}_{u_1} x_i\}_{i=1}^n \Rightarrow \text{Var}\{u_1^T x_i\}_{i=1}^n$$

$$\Downarrow$$

$$\frac{1}{n}\sum_{i=1}^{n} \left(u_1^T x_i - u_1^T \bar{x}_i\right)^2$$

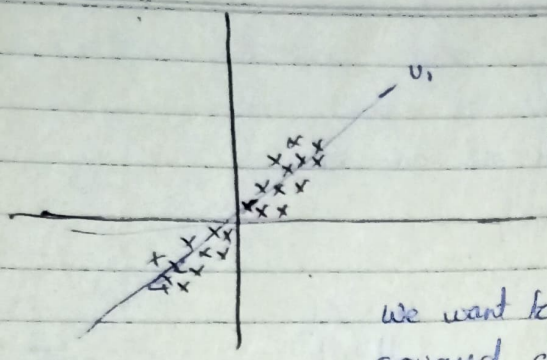
$\rightarrow 0$ {as X is a std. (zero mean $\rightarrow 0$)}

$$\Downarrow$$

We want to maximize $\rightarrow \frac{1}{n}\sum_{i=1}^{n} (u_1^T x_i)^2$ s.t u_1 is unit vector

So, we want to find u_1 s.t $\frac{1}{n}\sum_{i=1}^{n} (u_1^T x_i)^2$ is max and u_1 is unit vector.
This is an optimization problem.

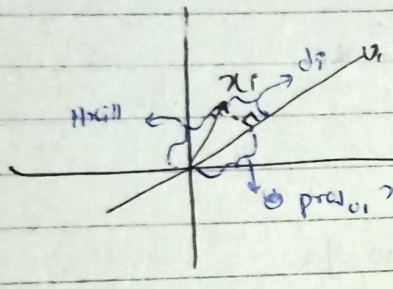
- Alternative formulation of PCA : dist - minimization.
  We have already seen a formulation of PCA i.e. finding a unit vector u_1 s.t. $\frac{1}{n}\sum (u_1^T x_i)^2$ is max.
  There is another formulation.



for each point x_i calculate d_i i.e. distance of x_i to direction u_1

we want to find u_1 which minimize the squared distance of each point x_i to u_1

$$\min \sum_{i=1}^{n} d_i^2$$



$$d_i^2 = \|x_i\|^2 - (u_1^T x_i)^2$$
$$= (x_i^T x_i) - (u_1^T x_i)^2$$

- we want $\min \sum_{i=1}^{n} (x_i^T x_i - (u_1^T x_i)^2)$

So we get two type of formulation

$$\max_{u_1} \frac{1}{n} \sum_{i=1}^{n} (u_1^T x_i)^2$$

- variance maximization PCA

$$\min_{u_1} \sum_{i=1}^{n} (x_i^T x_i - (u_1^T x_i)^2)$$

dist min PCA

Solution to our optimization problem:
Here we only see solution, we don't see how we reach to sol^n.

$$X = \begin{matrix} \\ 1 \\ 2 \\ 3 \\ \vdots \\ n \end{matrix} \begin{bmatrix} f_1 & f_2 & \cdots & f_d \\ & & & \\ & & & \\ & & & \end{bmatrix}_{n \times d}$$

$$S = \frac{1}{n} X^T (X) \quad \{\because X \text{ is column standardize.}$$

S ↓ covariance matrix

Column standardize ($\mu = 0, \sigma = 1$)

If we compute the eigen values of S = $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \ldots, \lambda_d$
Eigen vectors of S = $v_1, v_2, v_3, \ldots, v_d$

Let $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4 \geq \ldots \geq \lambda_d$

λ_1 ← eigen value.

$u_1 = v_1 = \frac{}{\lambda_1}$ = eigen vector of S corresponding to largest eigen value

So, final steps are.

(i) Column-standardize X

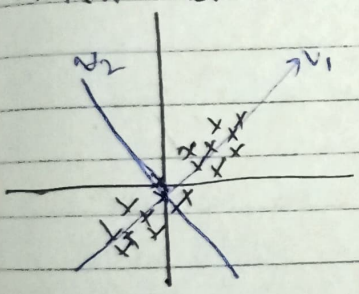
(ii) Compute covariance matrix S

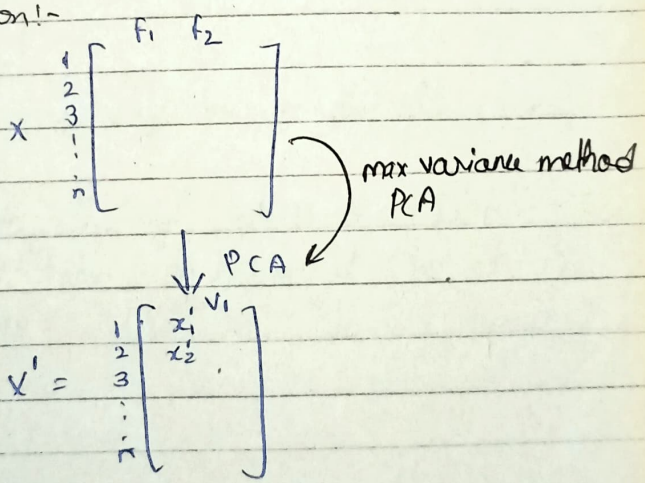
(iii) Compute eigen value & eigen vector

(iv) u_1 = eigen vector corresponding to largest eigen value.

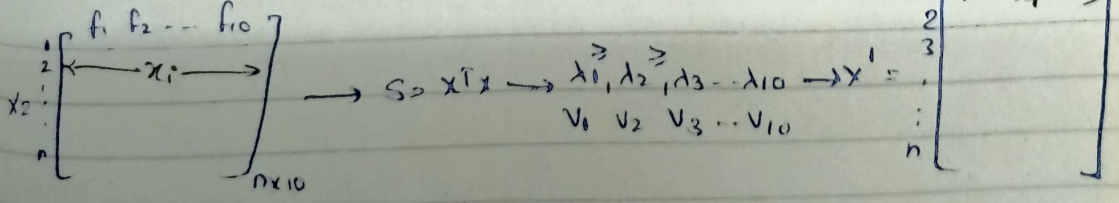
So, eigen vector is the vector in direction where we get maximal variance. But what does eigen value signify. λ tells us the percentage of variance on that eigen vector.

→ PCA for dim-reduction:-





Let's take a bigger example:



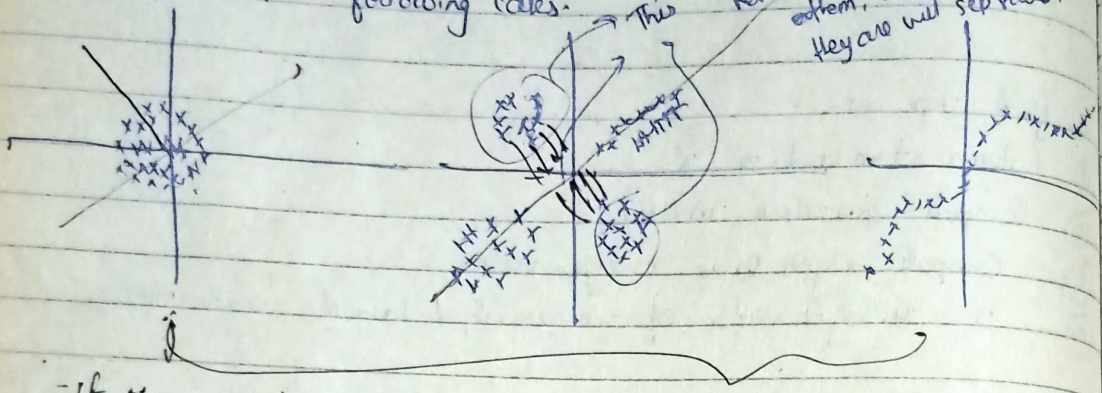
$x_i = [x_{i1}, x_{i2} \ldots x_{i10}] \rightarrow x_i' = [x_i^T v_1, x_i^T v_2]$

So, someone ask reduce dimension of d-dim data to d'-dim s.t we preserve 95% of data.

In that case as we know λ tells us the % of variance preserved by corresponding eigen vector. We will check upto which λ we get 95%. Include that much of eigen vectors.

→ Limitations of PCA :-

suppose the following cases.



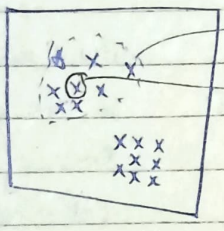
- If we apply PCA to chang 2D→1D we can see that ~~each~~ we can lost various information.

t-SNE (t-distributed, stochastic Neighbourhood Embedding)

This is one of the best sate of art dimensionality reduction technique specially for visualization.

There are also other techniques of Dimensionality Reduction like, Multi-dime mapping, sammon mapping etc.

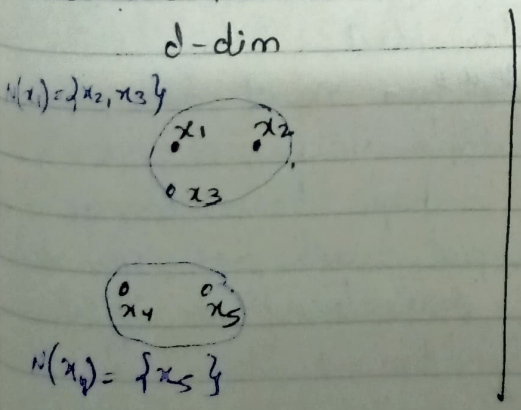
→ Neighbourhood, Embedding:
imagine we have a d-dim space.



$N(x_i) = \{x_j, \text{ s.t } x_i \& x_j \text{ are geometrically close}\}$

Suppose we want to change d-D → 2-D
Embedding means: take each point in d-D find its corresponding value in 2-D space or lower-dim space.

→ Geometric Intuition



$d(x_i', x_4')$ is not needed to be same as original.

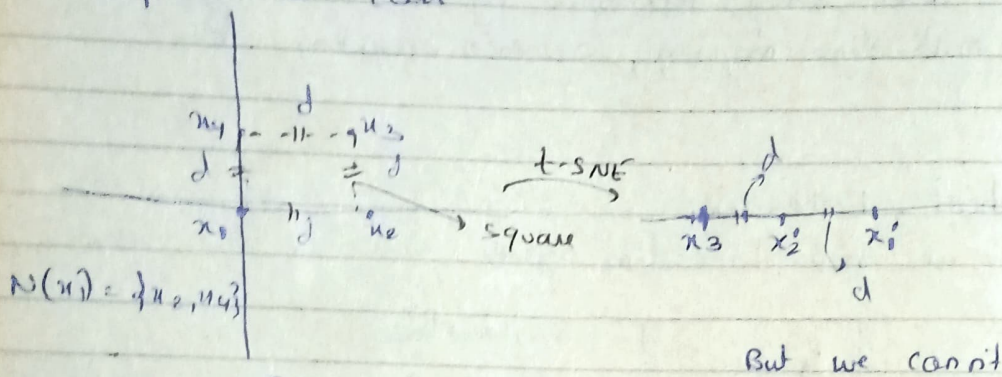
t-SNE only try to preserve distance of nodes with same neighbourhood.

distill.pub

If two nodes are not in neighbor then t-SNE don't care about their distance.

* Crowding problem.

We said t-SNE preserve dist in a Neighbor. Supose a case:



But we cannot place x_4 if we place it any where one of the Neighbor dist will not preserved.

Some time it is impossible to preserve distance in all the neighbourhood such a problem is called crowding problem.

* t-SNE:-

2 important parameter

1. steps (No. of iteration)
2. perplexity / Neighbourhood size (Not exactly but similar idea)

→ perplexity should be less than no of data points.
→ Run t-SNE for multiple value of perplexity.
→ Run t-SNE until the shape does not change.
↳ You might get different plot each time you run the t-SNE with same parameter. Hence t-SNE is not a deterministic algorithm.
→ t-SNE expands the denser Neighbourhood and shrinks less denser Neighbour. Hence, cluster sizes change after t-SNE.