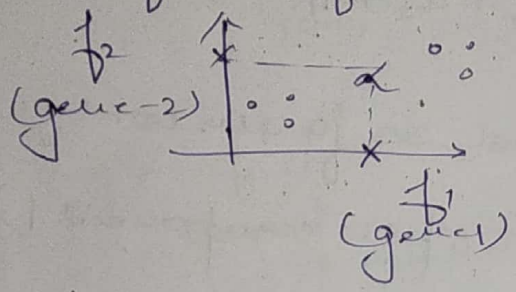
Principal component Analysis (PCA)

(Unsupervised learning problem)

m → TE
n → features

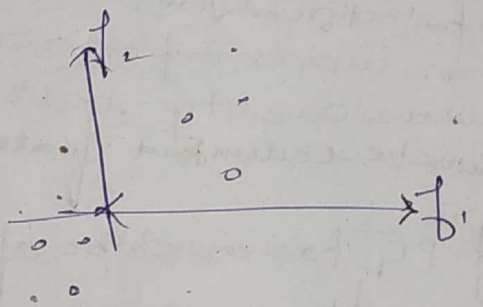
PCA - find the axis with max^m variance

e.g. for 2 features

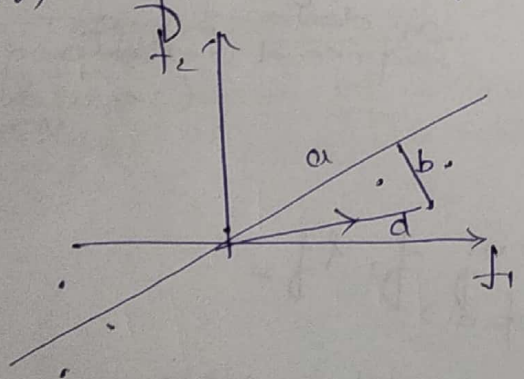


Step-1
- take the mean value wrt (f_1 and f_2) denoted by μ
find μ.

Step-2
Shift the origin to μ.



Step-3
Try to fit a line passing through origin



distances fixed for candidate line

minimize Σb
or maximize Σa

$d^2 = a^2 + b^2$

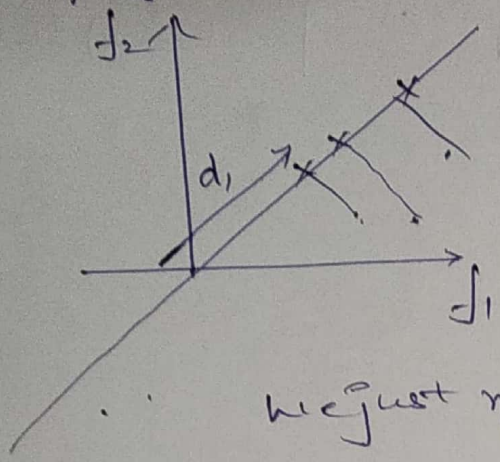
by pythagoras a & b are oppositely related

and will give the axis with max^m variance

Scanned by CamScanner

Step 4

Project all those distances so that line



SS = EV (Eigen Value)

SS = Sum of squares

$= d_1^2 + d_2^2 + d_3^2 + d_4^2 + d_5^2 + d_6^2$

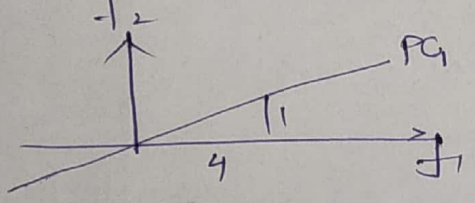
∴ We just need the largest SS for all obtaining line.

That line is called PC1 (principal component 1).

Advantages —

i) Avoid overfitting / curse
ii) computational cost
iii) gives more important features → i.e. on what features are more contributing to PC1 (or can be called the slope of PC1.
iv) dimensionality red^n.
v) Eliminate redundant features

e.g. if PC_1 has a slope of 0.25.



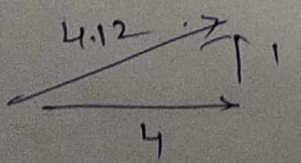
So, f_1 is more contributing features than f_2

Or data is more spread out along f_1 (i.e. more variance)

So, simply

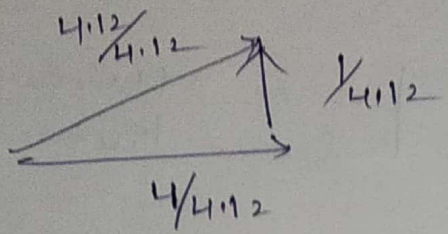
PC_1 is a linear comb^n of f_1 & f_2 (cocktail recipe)

PC_1 is a linear comb^n of variables



Scanned by CamScanner

When we do PCA with SVD, we scale the length 4.12 to 1



So, now f_1 contributes 0.97

and $f_2 = 0.242$

But our ratio is the same.

Eigen Vector / Singular Vector for PC1

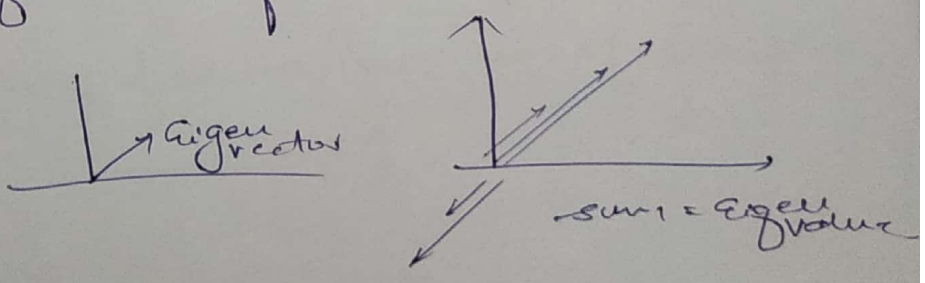
The unit vector that we just found along PC1 is known as Eigen / Singular Vector for PC1.

Loading Scores

The proportion of each features in Eigen comp are called loading scores.

Eigen value of PC1 = SS

The sum of square of dist of proj of points on PC1 (SS) = eigen value of PC1



Singular value of PC1

Singular value of PC1 = $\sqrt{\text{Eigen value for PC1}}$

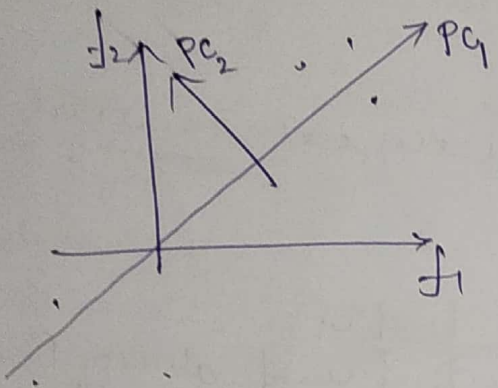
$= \sqrt{SS}$

Note that SS/n is the variance along the PC1 and we are maximizing it. i.e we are maximising EV and variance

Scanned by CamScanner

→ PC_2 will be directly the line to the PC passing through the origin without any further optimize.

Note:
Simply remember PC and PC_2 passes through the mean of all the data points & are linear combination of f_1 and f_2



We know, how to get the vector → loading scores
so, already explained contribution of f_1 and f_2 in that for PC_1 & make any one of them -ve.

i.e. for PC_1
if $f_1 \rightarrow 0.97$
$f_2 \rightarrow 0.242$

for PC_2
$f_1 \rightarrow -0.242$
$f_2 \rightarrow 0.97$
} loading scores of PC_2

Scanned by CamScanner

To draw the final PCA plot
- Mean on the origin
- Rotate to make PC1 as x-axis.

Thatle how PCA is done using SVD (Singular Value Decomposition)

Note:-
- for Eigenvalues distances are along the axes.

Variance : since it is along the axis.

Variation = Eigenvalue / (samplesize - 1) = Eigenvalue / (n-1) = SS / (n-1)

total variation = Sum of var along both the PCs
= $\lambda_1 + \lambda_2$.

Scree plot.
The graphical repⁿ of the % of variation that each PC accounts for.

* Similar is the PCA for 3 variable f_1, f_2, f_3 Lie gd PCA1, PCA2, PCA3, then-

No. of PCs = min(no. of features/var, no. of samples)

Scanned by CamScanner

Maths

Variance of a matrix

$$\sigma^2 = \frac{\Sigma (x_i - \bar{x})^2}{N}$$

N = sample example

$$= \frac{1}{N} \Sigma x_i^2 - \bar{x}^2$$

{ popul^n variance }

already $s^2 = \frac{\Sigma (x_i - \bar{x})^2}{n-1}$ → sample variance

$n-1$ → degree of freedom

or $\text{Var}(X) = E\left[(X - E(X))^2 \right]$

$= E[X^2 - 2X E[X] + E[X]^2]$

$= E[X^2] - 2E[X]E[X] + E[X]^2$

$= E[X^2] - E[X]^2$

Covariance of a matrix

→ gives only the dirn i.e how the variables are related to each other

+vely
or
-vely
} no. sig. of its value.

not related ⇒ value very close to 0

$$\text{cov}(x,y) = \frac{\Sigma (x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

$$= \frac{\Sigma x_i y_i}{N-1} - \bar{x}\bar{y}$$

Scanned by CamScanner

Note:-

If we consider a matrix as a transformation then in simple terms Eigen value is the strength of that transformation in a particular direction known as Eigen vector.

(see method to find Eigen value & Eigen vector)

Correlation [-1 to 1]

(effect of covariance)

gives the direction as well as mag. of how the variables are related

$$r, \text{ correl}^n = \frac{\text{cov}(x,y)}{\sigma_x \sigma_y}$$

In layman's term,

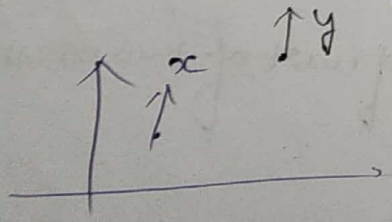
cov matrix → Physically/data structure view

→ its elements (i,j) are covariance b/w i^{th} & j^{th} element of (random) two vectors.

How similar the variances of features are.

Mathematically

Linear transform theory



strongly correlated (only one dirn) while its correl^n mag. also matters.

gives dirn of max^m variation.

PCA Physically.

projecting data onto Eigenvectors of covariance matrix

Mathematically PCs → Eigen vectors of (normalized empirical) covariance matrix.

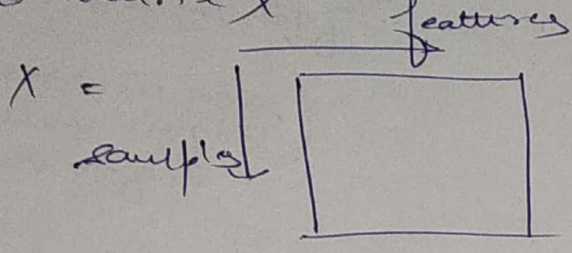
(Eigenvector & Eigen value of covariance matrix.

Scanned by CamScanner

Creation of covariance Matrix

PCA is simply eigenvector of empirical covariance matrix

given a matrix X



Step 1

do $x_{ij} - \bar{x}_j$ column wise

→ so gives relation b/w a feature and a sample

X = [ ]

Step-2

$X^T X$

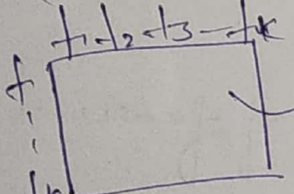
Now

$$\Sigma = \text{cov}(X) = \frac{(X^T X)}{n-1}$$ → for sample

n for popul^n

$n \approx (n-1)$ for very large value of n.

→ features in covariance & corre^n matrix



$$\rightarrow \frac{\Sigma(x_i-\bar{x})(y_i-\bar{y})}{N}$$

or $$\frac{\Sigma(f_{pi}-\bar{f}_p)(f_{qi}-\bar{f}_q)}{N-1}$$

Unit of covariance → product of unit of two variables

unit of correl^n → unitless

* correl^n & covariance matrix are symmetric & each element in the matrix represent correl^n & covariance b/w these var i and var j — diagonal = 1 in correl^n matrix

which are values of different data points for the feature values for the column feature

for 2d matrix I have 2 features n = 2 — features

features [ ] n is the datapoints for each column

$$\frac{\Sigma(x_i-\bar{x})(y_i-\bar{y})}{N}$$

Eigen Value

$A = \begin{bmatrix} \ \end{bmatrix}$ → So, $Ax = \lambda x$

(λ ↑ Eigen value; x ↓ Eigen vector)

$\Rightarrow (A - \lambda I)x = 0$

$\Rightarrow |A - \lambda I| = 0$ → characteristic Eqⁿ

$\begin{vmatrix} -\lambda & - & - \\ - & -\lambda & - \\ - & - & -\lambda \end{vmatrix} = 0$ and get the λ.

$\lambda = -, -$ ↳ equal to no. of dimension

Eigenspace

↳ is the eigenvectors that correspond to some Eigen vec value

↳ or the set of vectors that satisfy this Eqⁿ

$(A - \lambda I)\vec{x} = 0$

the Eigen-space is corresponding to each for $\lambda = \lambda_1, \lambda_2, \lambda_3$

$(A - \lambda_1 I)x = 0$

$\begin{bmatrix} - \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \end{bmatrix} = 0$ get the first Eigenvector

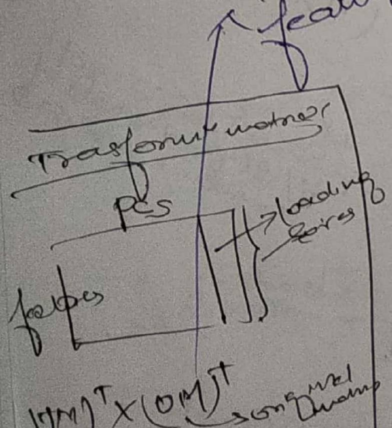
$x_1 - \frac{1}{2}x_2 = 0$

$x_1 = \frac{1}{2}x_2$

$\begin{bmatrix} 1/2 \\ 1 \end{bmatrix} v$

Solving this we will always get eqⁿ

PC as a linear function (combⁿ) of features



$(OM)^T \times (OM)^T$ ↳ original data

↓

↳ $T_x R^T$ → ans.

Eigen Vectors of the covariance matrix are actually the dir^ns where there is the most variance. (most inform^n)

Eigen values are simply the coefficient attached to Eigen Vectors (amt of variance carried by PCs)

* By ranking the Eigen vectors from highest to lowest based on eigen value we get most significant PCs {in scree plot may be}

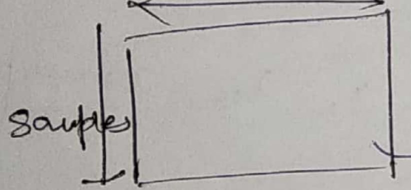
% Calcul^n

% of contribut^n calcul^n → $\frac{\text{E Val}_i}{\text{Sum of E Val}} * 100\%$ or $\frac{\text{Variance}}{\text{total variance}} * 100\%$

no. of discarding depends on computational cost & accuracy.

Feature Vector → Values of each E corresponding to all the features for all TE

choose PCs



→ each row ⊕ set for cyclone ⊕ set for HAR

in SVM 2D features

in cyclone sample 4D excluding time & 5D including time

* But even now the i/p dataset (sample values are of original axes).

so recast it on feature vector by $(m \times n)^T$

significant
we get PCs as a linear comp^n → $(n \times m)$

⇓

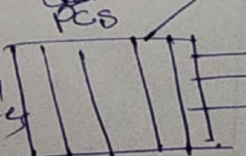
final dataset = $(\text{feature vector})^T * (\text{standardized original dataset})^T$

↓

Transform^n matrix

$(n \times d)^T$

$(d \times n)$



** Each PCs we get of by eigenvectors of covariance matrix

left with values only on PCs $(m \times d)^T$ $(d \times m)^T$ $(d \times m)$

$(n \times d)^T \times (m \times n)^T$

finally take one more transform^n to get $(m \times d)$

Scanned by CamScanner

Covariance v/s Correl^n matrix for PCA

creation

The covariance matrix

$\because cov(x,y) = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y}_i)}{N}$

correl^n matrix — standardized version of covar matrix

$\because correl^n(x,y) = \frac{cov(x,y)}{\sigma_x \sigma_y}$

$= \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{N \sigma_x \sigma_y}$ (same)

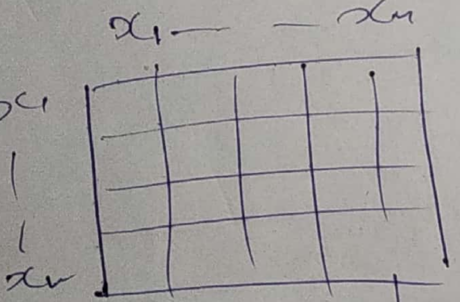
$= \frac{\Sigma(x_i - k)(y_i - k)}{N \sigma_x^2}$

If $x_i = y_i$, corr^n = 1

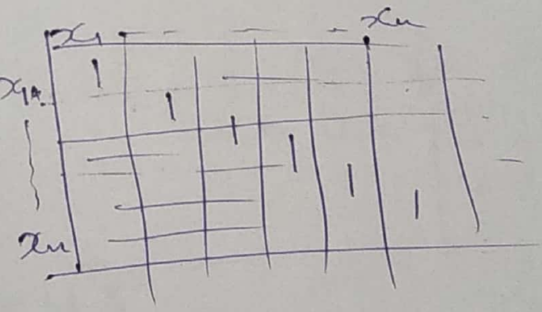
* covariance matrix are used when the features have same scale.
  correl^n when scale diff. like pressure & temperature.
  depends on application

* generally correl^n matrix contribut. by PC1 & PC2 is less say 62% and 24%. while in covariance matrix 96%, 2% etc.

Covariance matrix (symmetric)



correl^n matrix (symmetric)



Standardized original i/p data

is EV will give a different direction with max^m variances

The more is EV for height EV roughly

each cell in cov. mat can't be compared but in correl^n can — very clearly or very separated, more clearly highest variance

PCA

$$\Sigma = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}$$

for a d-dimensional summary of the data that captures as much variance as possible.

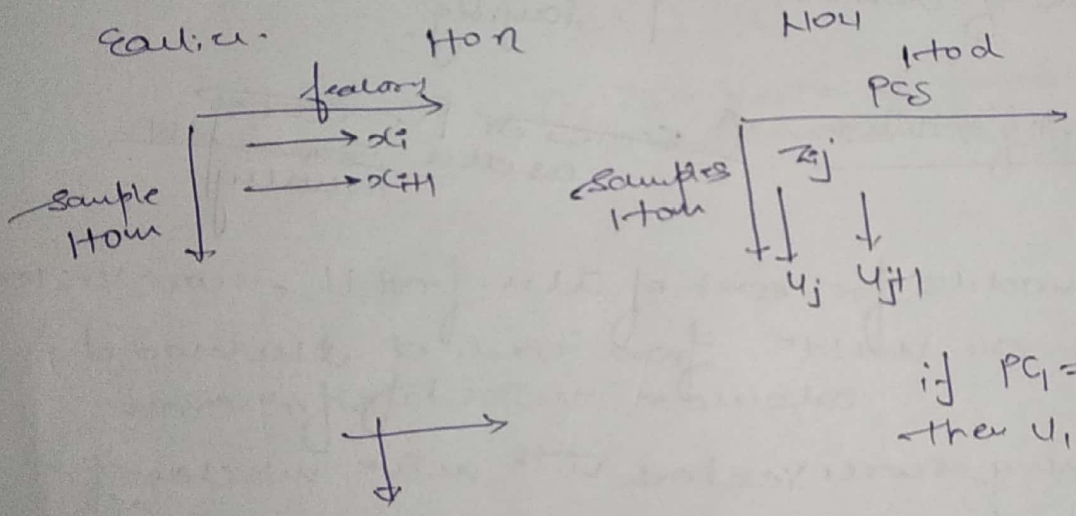
We represent data in the basis of first d PCs.

$\{u_1, u_2, \ldots, u_d\}$

and compute

$$z_{ij} = u_j^T x^{(i)} \quad \text{for } j = 1 \text{ to } d, \; i = 1 \text{ to } m$$

i.e. conversion to PC axis

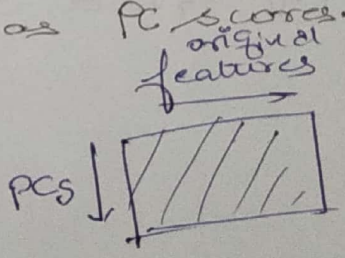


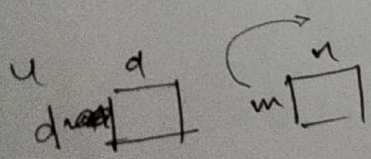
if $PC_1 = f_1$ say

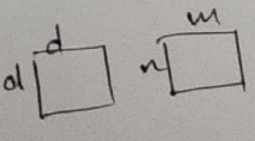
then $u_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$.

So, $\boxed{z_{ij} = u_j^T x^{(i)}}$ or $Z = U^T X^T$ simply

These entries are referred to as PC scores.







Scanned by CamScanner

Remember:

PCs are obtained only after centering and scaling the original features.

Note:

Even if there are multiple PCs we will get some general classification by plotting first & second PCs any method features.

(it have most variance contribution & diff. dirn)

Since we use the one along which most of the variance change occur.

like

Walking
Walking upstairs
Walking downstairs

|

standing
sitting
laying.

Classification Algorithms

~~scores~~ or PC ~~scores~~ as the ~~score~~ plot actually.

steps →

(1) Build models for each of the first H score vectors.
(2) Predict on validation from each of the model & calculate misclassification error

cross validation can be directly seen by estimate.

Select q score vectors with min misclassification error.

(3) Use the q score vectors to build lpf training data

(4) Predict on test data misclassification error

Scanned by CamScanner

KNN
K Nearest Neighbour

$$P(y^{(i)}=j \mid x=x^{(i)}) = \frac{1}{K} \sum_{q \in N} 1\{y^{(q)}=j\}$$

Here we "set" to simultaneously select the no. of PCs or KPCs & the optimal value of K.
We consider potential $K \in \{1,3,5,7,9\}$

5. Softmax Regression

$$P(y^{(i)}=j \mid x=x^{(i)}) = \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{L} e^{\theta_l^T x^{(i)}}}$$

$j \in \{1,2,\ldots,L\}$

θ_j is a vector of parameters associated with the single class j.

Scanned by CamScanner