

---

# **Pattern Classification**

## **06. Feature Selection & Extraction**

---

AbdElMoniem Bayoumi, PhD

Fall 2021

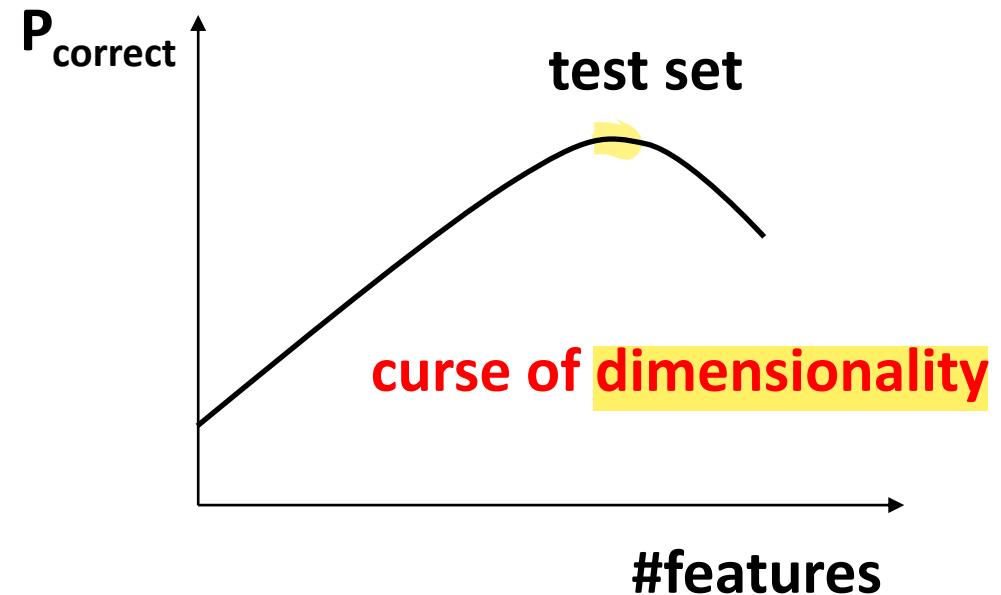
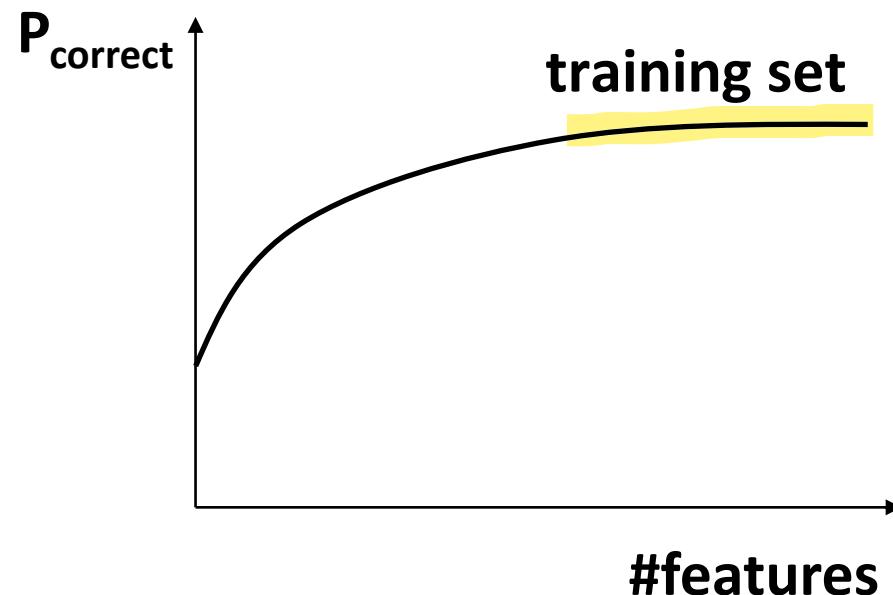
# Recap: Feature Vector

- Let  $\underline{X}(m) = \begin{bmatrix} X_1(m) \\ X_2(m) \\ \vdots \\ X_N(m) \end{bmatrix}$  be the feature vector of the  $m^{\text{th}}$  training pattern
- $N$  is the number of features, i.e., the dimension of  $\underline{X}(m)$
- $M$  is the number of the training patterns

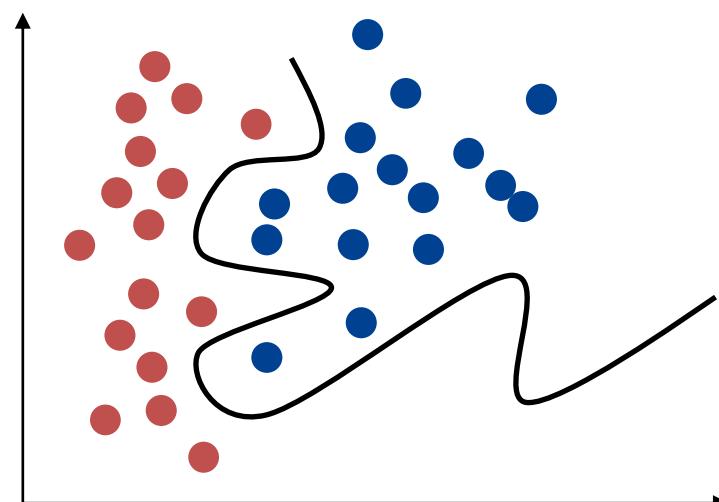
# Feature Selection Problem

- Choosing good features affects the classification performance
- In many applications, we can think of a large number of possible features
- But if we use too many features we will suffer from curse of dimensionality

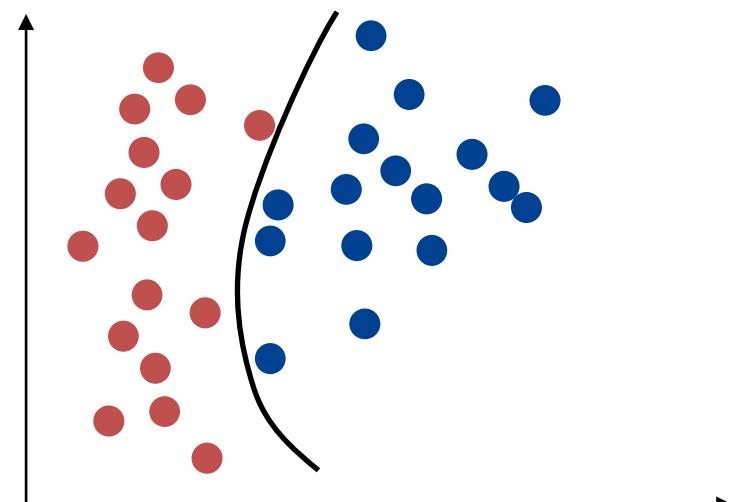
# Feature Selection Problem



Example:



A bad classifier with bad generalization ability



A good classifier with good generalization ability

# Feature Selection Problem

- The generalization ability is the ability of the classifier to generalize well for data it had not seen before, i.e., test data

# Curse of Dimensionality

- The parameters' estimates, e.g., mean vector and covariance matrix, will not be very accurate
- The data points will become scattered and clustering of classes will not be clear
- It is therefore best to reduce the no. of features and only retain the few best ones

# Feature Selection & Extraction

- **Feature Selection:** From the available N features choose a subset of L ( $\ll N$ )
- **Feature Extraction:** Transform the available N features into a smaller no. of L features through certain transformations

$$u_1 = f_1(x_1, x_2 \dots x_N) \quad \underline{X} \equiv \text{original feature vector}$$

$$u_L = f_L(x_1, x_2 \dots x_N) \quad \underline{U} \equiv \text{transformed feature vector}$$

usually  $f_1, f_2 \dots f_L$  are linear fn's, e.g.,

$$u_j = \sum_{i=1}^N V_{ij} X_i \quad , V_{ij} \equiv \text{constant}$$

- Transform original features to a more compact set of dimensions, while retaining as much information as possible.
- Transformed features may have no physical meaning → explaining the model is problematic
- May not be suitable to every domain

# Types of Features to be Removed

- **Irrelevant** Features:
  - Features that do not help in **discriminating** between the classes
- **Correlated** Features:
  - Features that **vary very closely** with other features (i.e., redundant features)
  - Hence can be removed without **much loss of information**
  - E.g., #corners & #lines

## Sequential Forward Selection Algorithm (**SFS**)

1. Examine each feature  $x_i$ , taken alone, design the classifier & select the best feature
2. From the remaining features, select the one that together with the selected group, gives best performance
3. Continue in this manner until you have selected L features

## Example on Sequential Forward Selection (SFS)

1. Take  $x_1 \rightarrow$  design the classifier only on  $x_1$   
Get  $P(\text{correct}) \equiv P_1$   
Take  $x_2 \rightarrow$  design the classifier only on  $x_2 \rightarrow P_2$   
 $\vdots$   
Take  $x_N \rightarrow$  design the classifier only on  $x_N \rightarrow P_N$

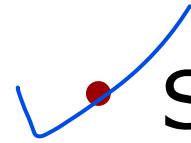
Find highest  $P_i$ . Assume  $P_5$  was highest  $\rightarrow$  choose  $X_5$  and fix it among the **chosen features** in next iterations

2. Test  $X_1, X_5 \rightarrow P_1$   
Test  $X_2, X_5 \rightarrow P_2$   
 $\vdots$   
Test  $X_N, X_5 \rightarrow P_N$   
Find highest  $P_i$ . Assume  $P_2$  was highest

3. Test  $X_1, X_2, X_5 \rightarrow P_1$   
 $\vdots$

# Sequential Backward Selection Algorithm

SBS



Similar to SFS, but done backward:

Top down

bngeb el classification lw khadna kol el features, w b3den nfdl nshel wahed wara wahed w nshof ahsan combination hatb2a anhy.

1. Start with all features selected. Remove one feature at a time, design the classifier and compute  $P_c$
2. The feature we end up removing is the one that results in the least reduction of  $P_c$
3. Keep removing till reaching the no. L selected features

usually SBS and SFS remove the same features, but it is not a must.

## Example on Sequential Backward Selection

- $X_1, X_2, X_3, X_4 \rightarrow P_c = 0.87$  *Be > T*
  - $X_2, X_3, X_4 \rightarrow P_c = 0.86$
  - $X_1, X_3, X_4 \rightarrow P_c = 0.82$
  - $X_1, X_2, X_4 \rightarrow P_c = 0.81$
  - $X_1, X_2, X_3 \rightarrow P_c = 0.84$

Remove  $X_1$

- $X_2, X_3, X_4 \rightarrow P_c = 0.86$ 
  - $X_2, X_3 \rightarrow P_c = 0.79$
  - $X_2, X_4 \rightarrow P_c = 0.80$
  - $X_3, X_4 \rightarrow P_c = 0.84$

Remove  $X_2$

Assuming  $L = 2$  then stop  
& Select  $X_3, X_4$

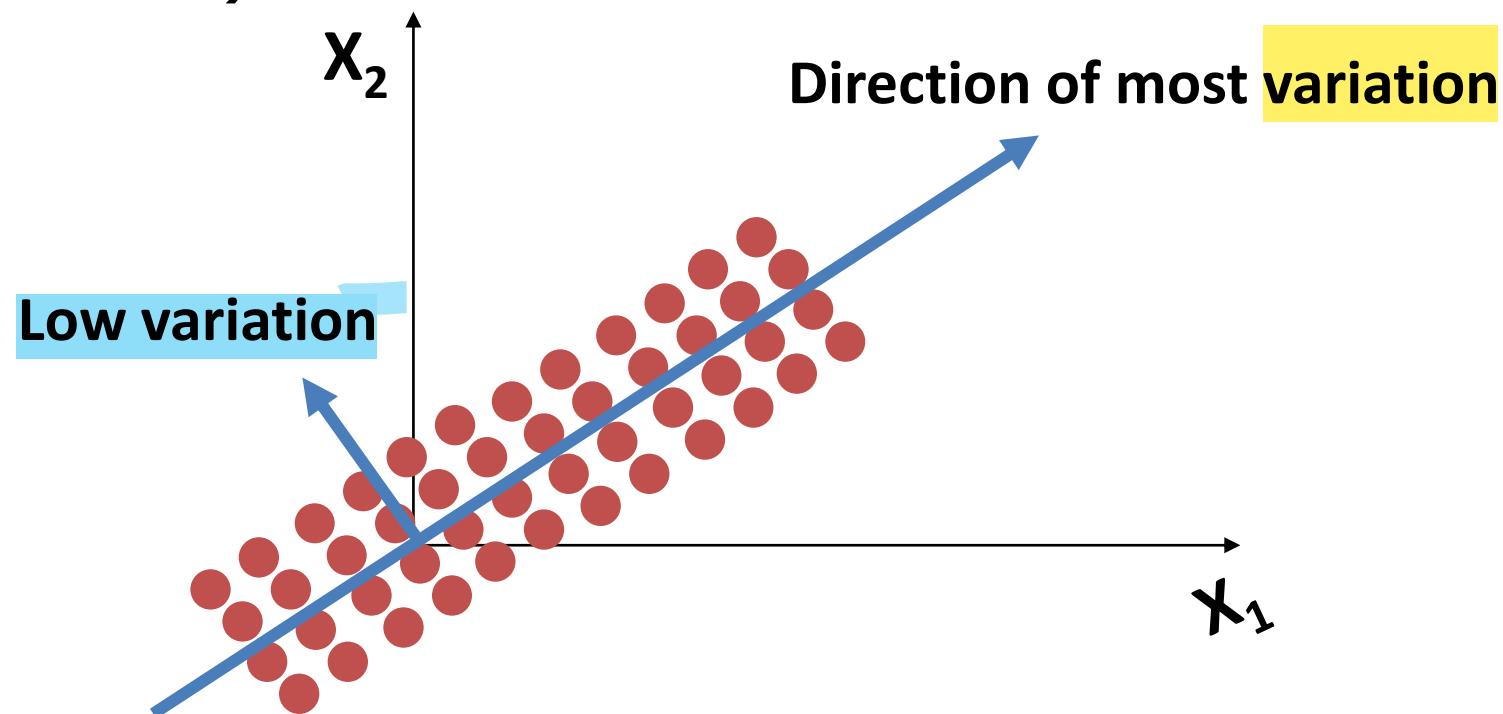
# Methods of Feature Selection

- Methods of feature selection can be categorized into:

- ↳ – **Filter type:** select features without looking into the classifier you are going to use, e.g., based on correlations of the features, distances between class means, ... etc
  - Adv: fast execution, generality
  - Disadv: tendency to select large subsets
- ↳ – **Wrapper type:** select features by taking into consideration the classifier you will use, e.g. SFS, SBS
  - Adv: accuracy
  - Disadv: slow execution, lack of generality

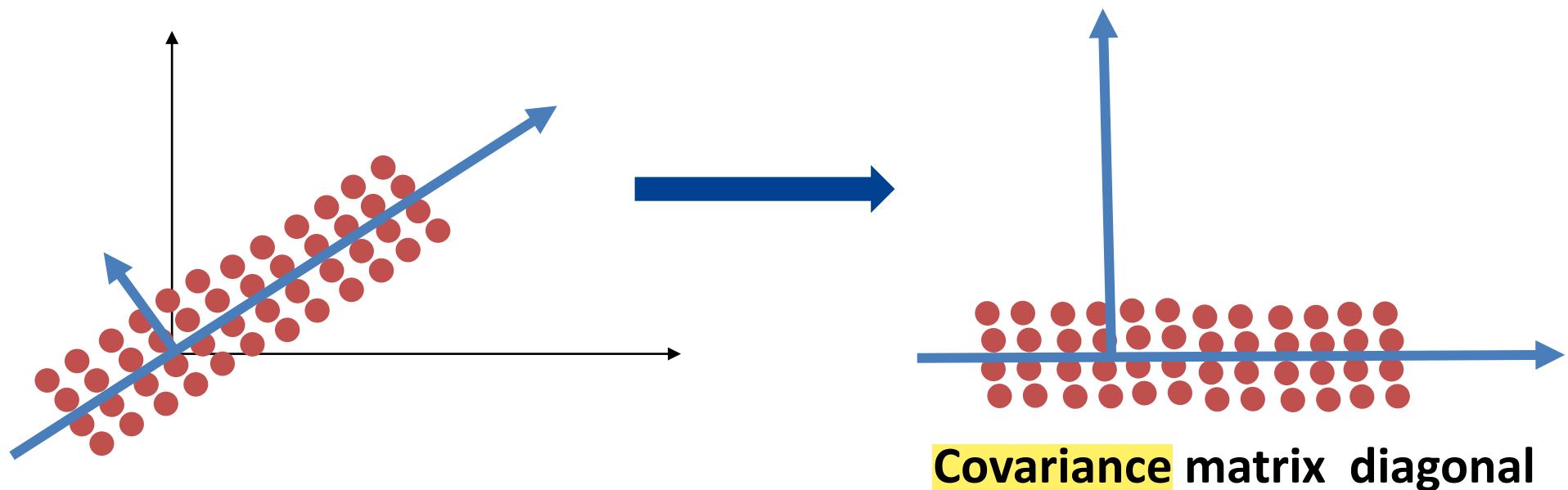
# Feature Extraction

- Principal Component Analysis (PCA)
- In PCA, the most interesting directions are those having large variations (large variance)



# Principal Component Analysis (PCA)

- Consider the most interesting directions, i.e., transformed features show an impact on data variations



# PCA Summary Steps

- Transform data to zero mean, i.e.,  $\underline{Y} = \underline{X} - \underline{\mu}$
- Estimate covariance matrix from data, i.e.,

$$\hat{\Sigma} = \frac{1}{M} \sum_{m=1}^M \underline{Y}(m) \underline{Y}(m)^T$$

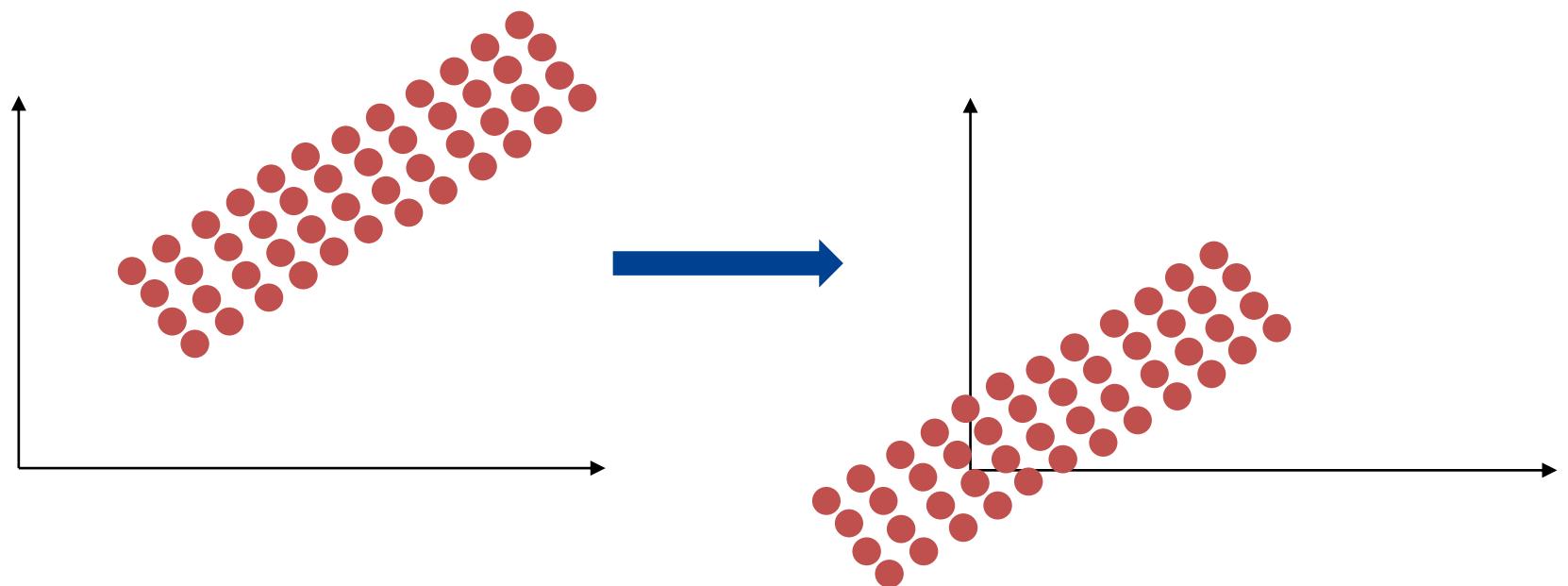
- Compute eigen values and vectors of  $\hat{\Sigma}$
- Choose the eigen vectors corresponding to big eigen values
- Perform transformation of data to a new space

$$\underline{Z} = \begin{bmatrix} \underline{u}_1^T \\ \underline{u}_2^T \\ \vdots \\ \underline{u}_L^T \end{bmatrix} \underline{Y}$$

# Principal Component Analysis (PCA)

- First, transform the problem to zero mean

$$\underline{Y} = \underline{X} - \underline{\hat{\mu}}$$



# Principal Component Analysis (PCA)

- Consider the covariance matrix:

$$\Sigma = E[\underline{Y}\underline{Y}^T]$$

or its **estimate** from the data:

$$\hat{\Sigma} = \frac{1}{M-1} \sum_{m=1}^M \underline{Y}(m)\underline{Y}(m)^T$$

where

$$\underline{Y}(m) = \underline{X}(m) - \underline{\hat{\mu}}$$

# Principal Component Analysis (PCA)

- Linear Transformation (i.e., Rotations):

$$\underline{Z} = \underline{AY}$$

- So covariance matrix of  $\underline{Z}$ :

hwa da el covariance el geded el ana 3auzo

$$\begin{aligned} cov(\underline{Z}) &= E[(\underline{AY})(\underline{AY})^T] \\ &= E[\underline{AY}\underline{Y}^T\underline{A}^T] \\ &= \underline{A}E[\underline{Y}\underline{Y}^T]\underline{A}^T \\ &= \underline{A}\Sigma\underline{A}^T \end{aligned}$$

$$(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$$

A is constant matrix

# Eigenvectors & Eigenvalues

- An eigenvector  $\underline{u}$  of a linear transformation matrix is a non-zero vector
- The eigenvector changes by only a scalar factor (called eigenvalue) when that linear transformation is applied to it:

$$B\underline{u} = \lambda \underline{u}$$

# Eigenvectors & Eigenvalues

$$B\mathbf{\underline{u}} = \lambda\mathbf{\underline{u}}$$

- There are generally  $N$  eigenvectors for a  $\mathbf{B}$  ( $N \times N$  matrix) if  $\mathbf{B}$  is diagonalizable
- Symmetric matrix is always diagonalizable

$$\begin{aligned} B[\mathbf{\underline{u}}_1 | \mathbf{\underline{u}}_2 | \cdots | \mathbf{\underline{u}}_N] &= [\lambda_1 \mathbf{\underline{u}}_1 | \lambda_2 \mathbf{\underline{u}}_2 | \cdots | \lambda_N \mathbf{\underline{u}}_N] \\ &= [\mathbf{\underline{u}}_1 | \mathbf{\underline{u}}_2 | \cdots | \mathbf{\underline{u}}_N] \begin{bmatrix} \lambda_1 & & & & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \ddots & & \lambda_N \end{bmatrix} \end{aligned}$$

Normalized eigen vectors

$$\mathbf{B}\mathbf{U} = \mathbf{U}\Omega$$

# Eigenvectors & Eigenvalues

- Eigenvectors of symmetric matrix is guaranteed to be orthogonal (i.e., perpendicular to each other)
- An orthogonal matrix is a square matrix whose columns and rows are orthogonal unit vectors (i.e., orthonormal vectors)

$$Q^T Q = I$$

$$Q^T = Q^{-1}$$

# Principal Component Analysis (PCA)

to get omega only, we need to multiply by  $U^{-1}$   
but using the property that if the eigenvectors where  
orthonormal we can say that  
 $U^{-1} = U^T$   
then after we substitute we can get  
this equation

$$BU = U\Omega$$

Take  $B = \Sigma$

$$\Sigma U = U\Omega$$

$$U^T \Sigma U = \Omega \quad \text{omega is the selection criteria}$$

- Suppose we take  $A = U^T$  to be the transformation, so:

ehna msh 3arfén A, fa hanftrd enha  $U^T$   
w han7sb aknna 3arfénha, fa hytl3lna el omega  
fa keda el denya zabatet.

$$Z = U^T Y$$

Recall:  $\text{cov}(Z) = A\Sigma A^T$

$$\begin{aligned}\text{cov}(Z) &= A\Sigma A^T \\ &= U^T \Sigma U \\ &= \Omega\end{aligned}$$

covariance matrix is diagonal !  
So we find which transformed  
features that cause difference

$$\begin{bmatrix} \underline{u}_1^T \\ \underline{u}_2^T \\ \vdots \\ \underline{u}_N^T \end{bmatrix} \text{ where } \underline{u}_i \text{ is}$$

- We took the transformation matrix A as the eigenvector of  $\Sigma$  corresponding to eigenvalue  $\lambda_i$

# Principal Component Analysis (PCA)

- Assume we reordered variables such that  $\lambda_1 \gg \lambda_2 \gg \dots \gg \lambda_N$
- We take only  $L \ll N$  directions.
- These are the directions corresponding to the largest  $L$  eigenvalues of  $\Sigma$
- Since the eigenvalue corresponds to the variance of the transformed variable  $Z_i$ , this means we took  $L$  directions with largest variance & thrown out the rest of directions

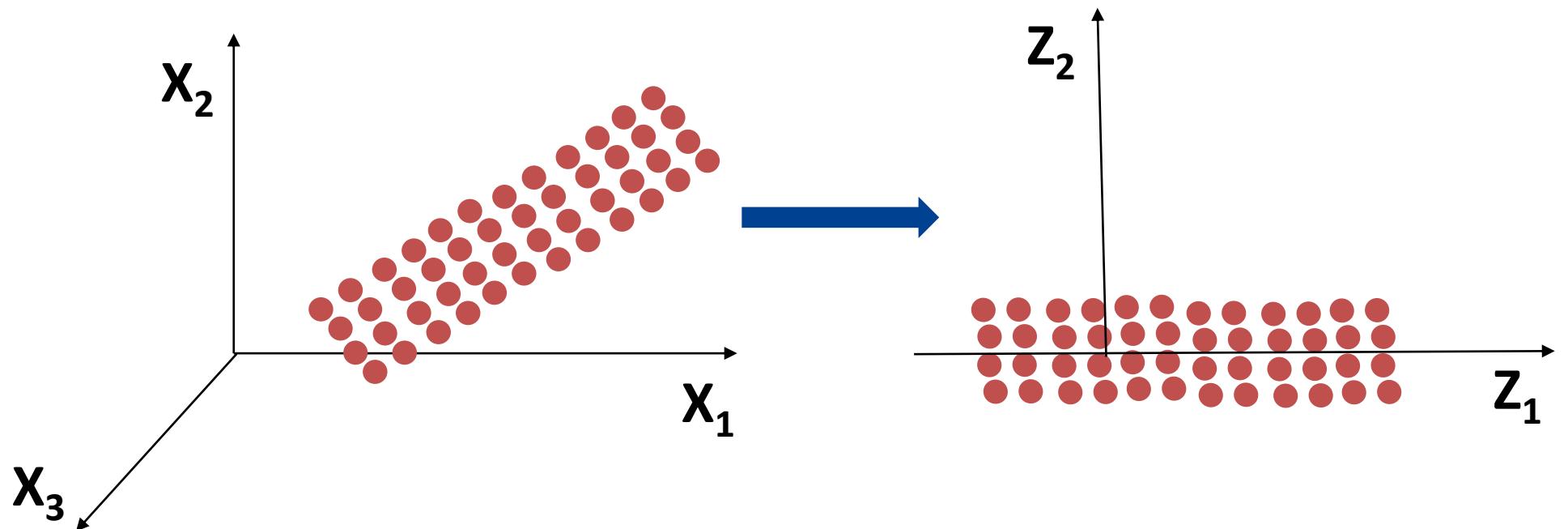
# Principal Component Analysis (PCA)

- The transformation can be re-written as follows:

$$\underline{Z} = \begin{bmatrix} \underline{u}_1^T \\ \underline{u}_2^T \\ \vdots \\ \underline{u}_L^T \end{bmatrix} \underline{Y}$$

$\underline{Z}$  is the extracted (transformed) feature vector

# Principal Component Analysis (PCA)



# Principal Component Analysis (PCA)

- Example:
  - 20 features problem
  - Eigen values: 12

5

2

0.3

0.01

0.002

0.0005

⋮

Take 4 PC's, it explains:

Note: scale affects PCA!

$$\%var = \frac{12 + 5 + 2 + 0.3}{12 + 5 + 2 + 0.3 + 0.01 + 0.002 + \dots}$$
$$\simeq 99.9\%$$

# PCA Summary Steps

- Transform data to zero mean, i.e.,  $\underline{Y} = \underline{X} - \hat{\mu}$
- Estimate covariance matrix from data, i.e.,

$$\hat{\Sigma} = \frac{1}{M-1} \sum_{m=1}^M \underline{Y}(m) \underline{Y}(m)^T$$

- Compute eigen values and vectors of  $\hat{\Sigma}$
- Choose the eigen vectors corresponding to big eigen values
- Perform transformation of data to a new space

$$\underline{Z} = \begin{bmatrix} \underline{u}_1^T \\ \underline{u}_2^T \\ \vdots \\ \underline{u}_L^T \end{bmatrix} \underline{Y}$$

very interesting lecture :))))

# Acknowledgment

- These slides have been created relying on lecture notes of Prof. Dr. Amir Atiya