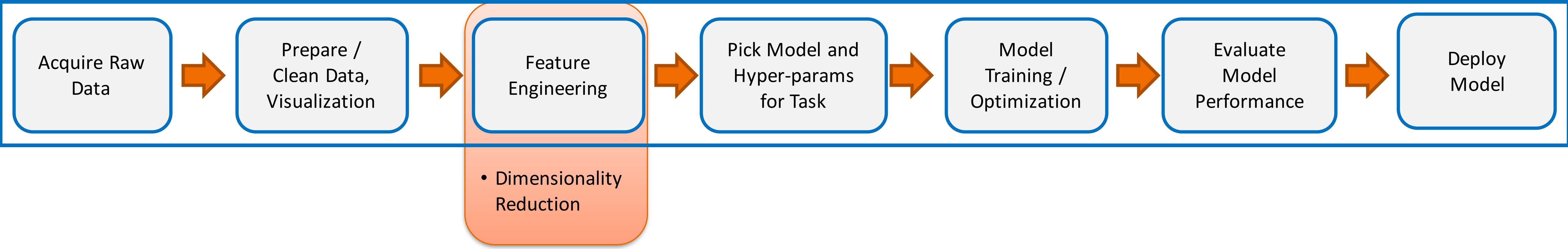
Focus for this lecture

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

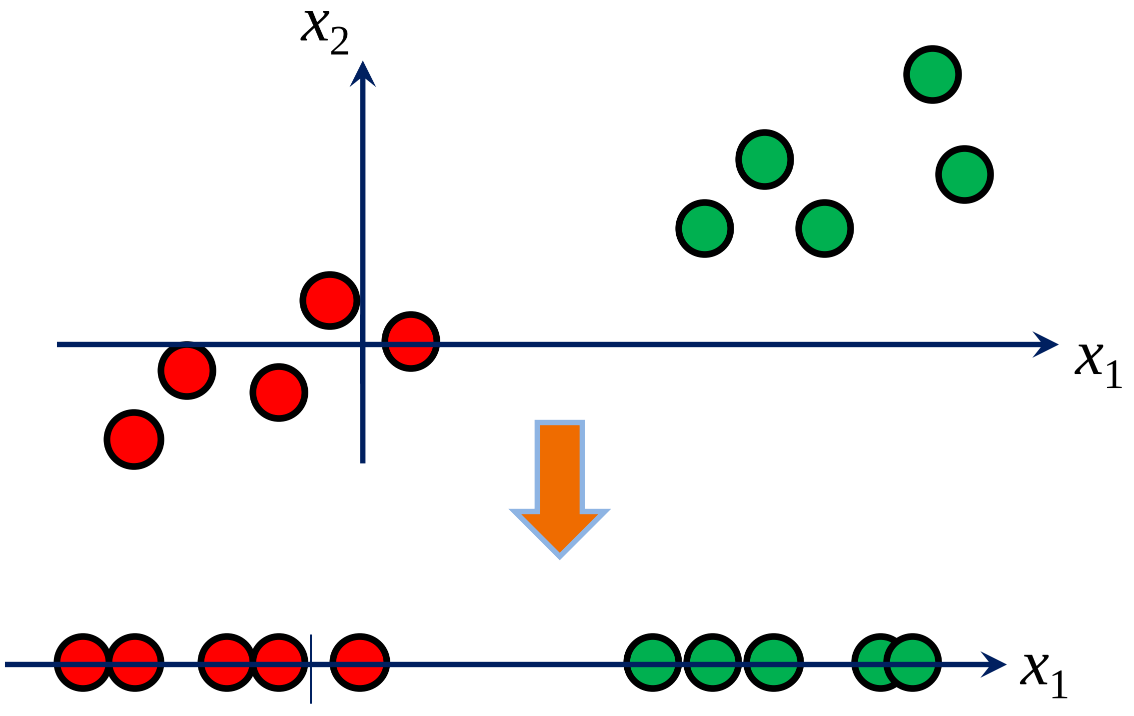
- Dimensionality Reduction
- Introduction to PCA
- PCA Algorithm
- Applications of PCA

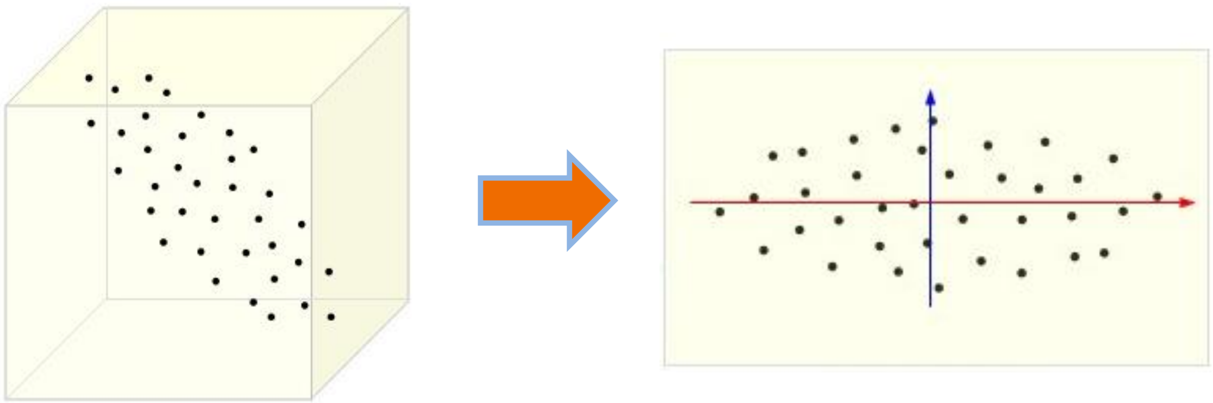


Dimensionality Reduction

Reducing Dimensions

- Feature Selection:
  - Choose the "best" features from your data
- Feature Extraction:
  - Initial set of measured data and builds derived features intended to be informative and non-redundant
- Feature Visualization:
  - How are the ‘best’ features distributed in 1D/2D/3D ?

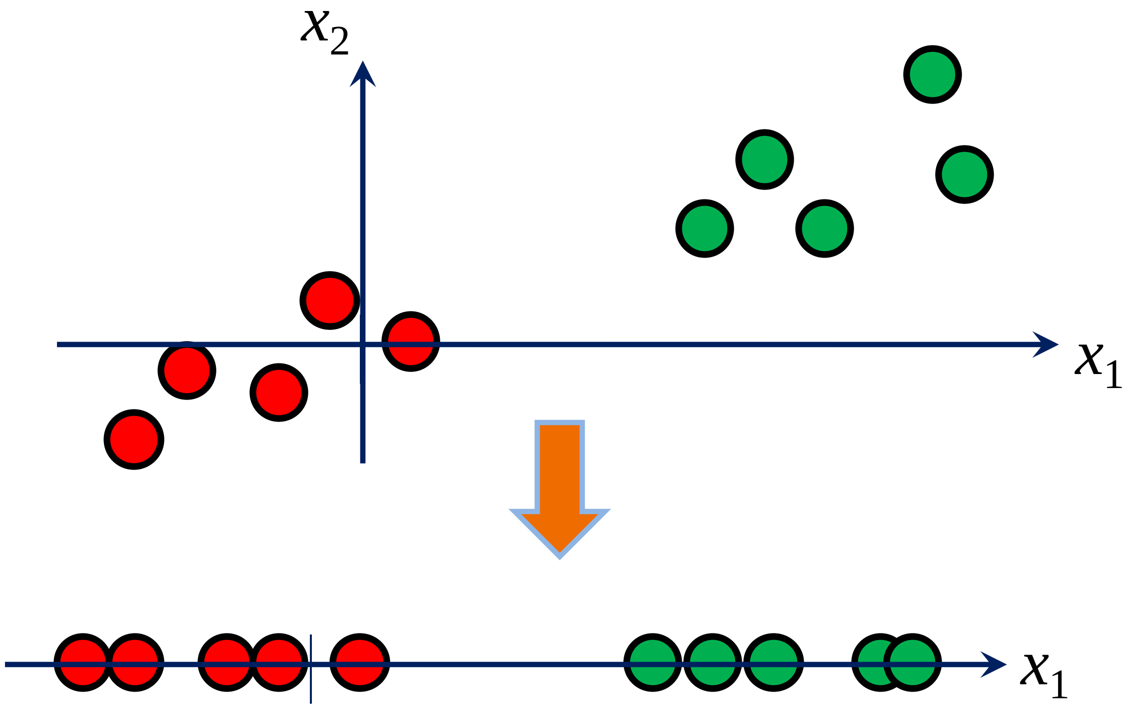




Selecting and Extracting Features

$$\begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting first and third feature



$$\begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

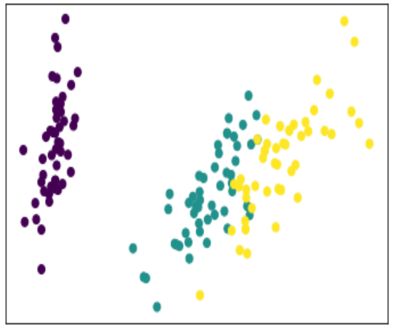
Selecting first and fourth feature

NOTE: Data samples are color-coded by their class label. But label info is <u>not used</u> for feature selection.

Selecting and Extracting Features

$$\begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting first and third feature



$$\begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting first and fourth feature

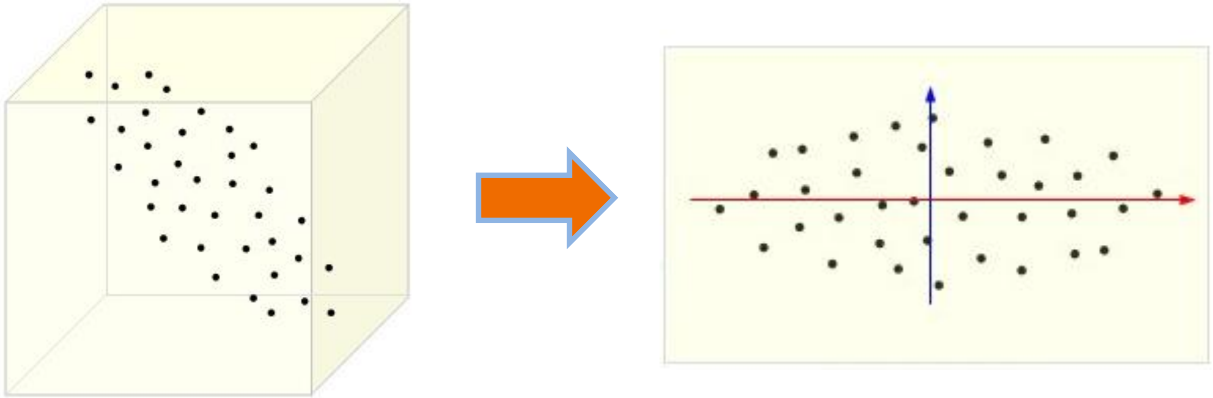
NOTE: Data samples are color-coded by their class label. But label info is <u>not used</u> for feature selection.

Selecting and Extracting Features

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.0 & 0.4 & 0.2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

New Features as linear combination of old Features

$$X' = AX$$

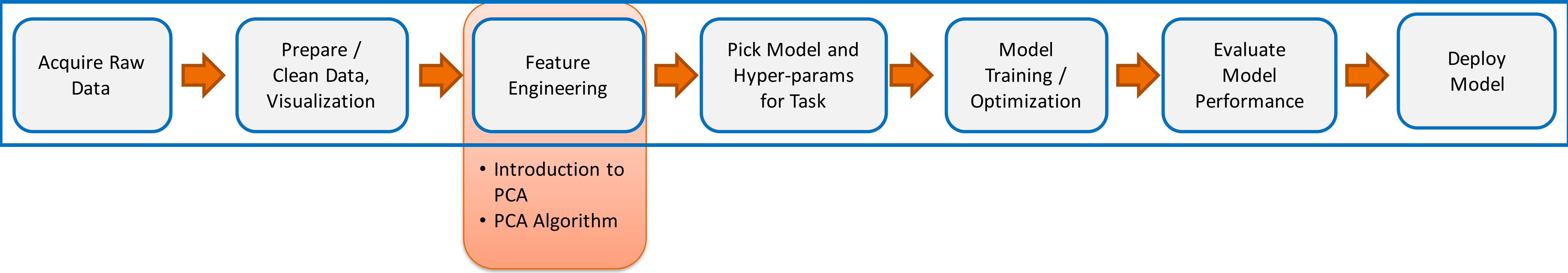


Journey so far…

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

✓ Dimensionality Reduction

7
7



Feature Extraction

Introduction to Principal Component Analysis (PCA)

PCA: A Toy Example

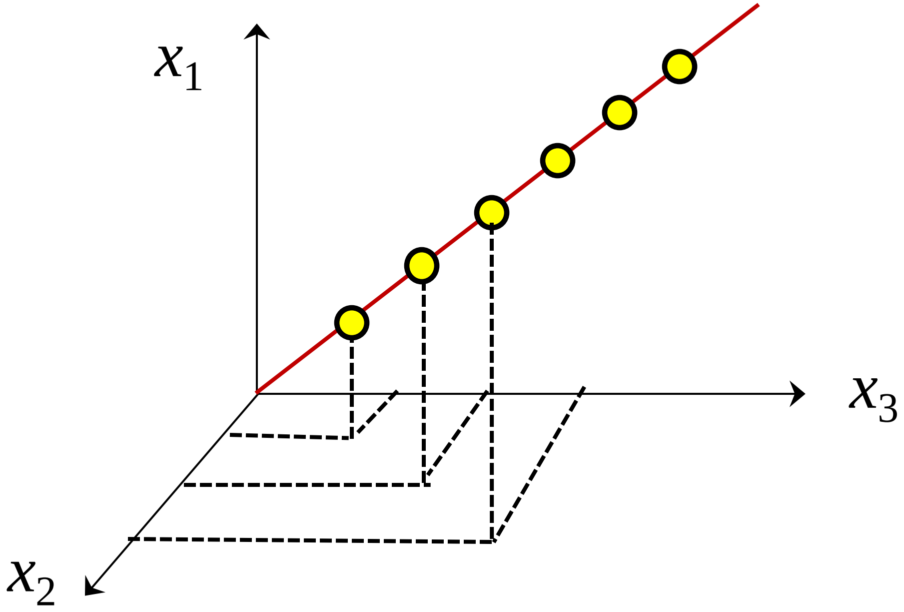
- Consider the following dataset:

1	4	3	6	5	2
2	8	6	12	10	4
3	12	9	18	15	6

PCA: A Toy Example

- All these points fall on a line: a 1-dimensional subspace of the original 3D space:

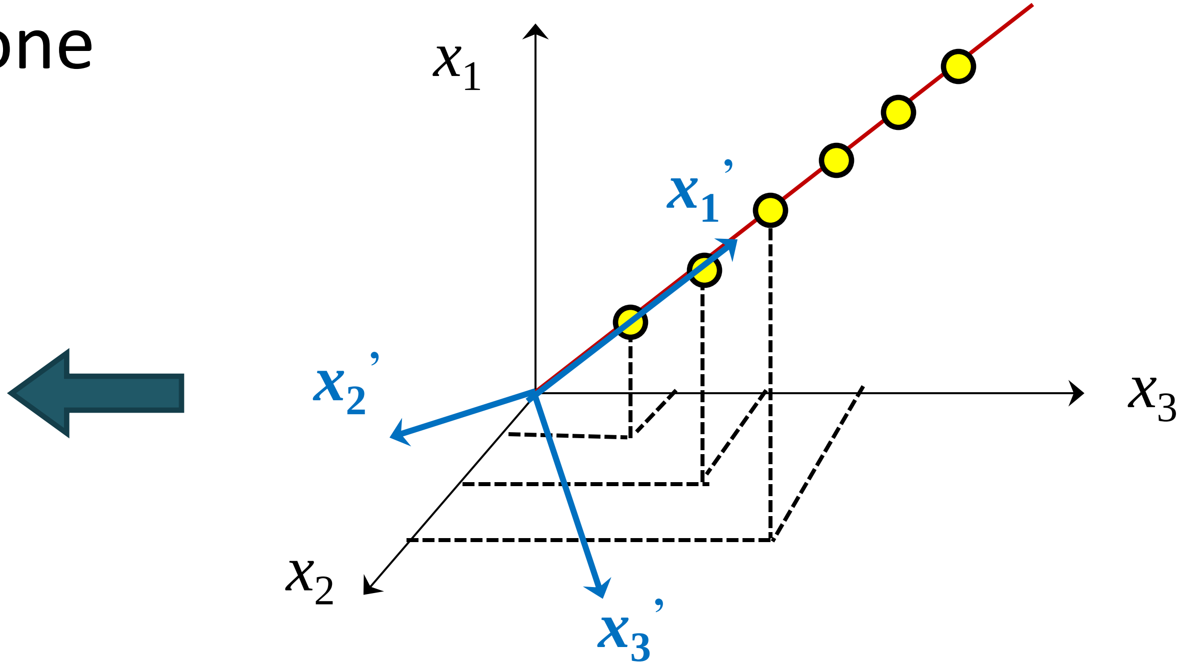
1	2	3	4	5	6
2	4	6	8	10	12
3	6	9	12	15	18



PCA: A Toy Example

- Consider a new co-ordinate system with one axis along the line
- All co-ordinates except the first one are zeros now.

3.7	7.5	11.2	15	18.7	22.4
0	0	0	0	0	0
0	0	0	0	0	0



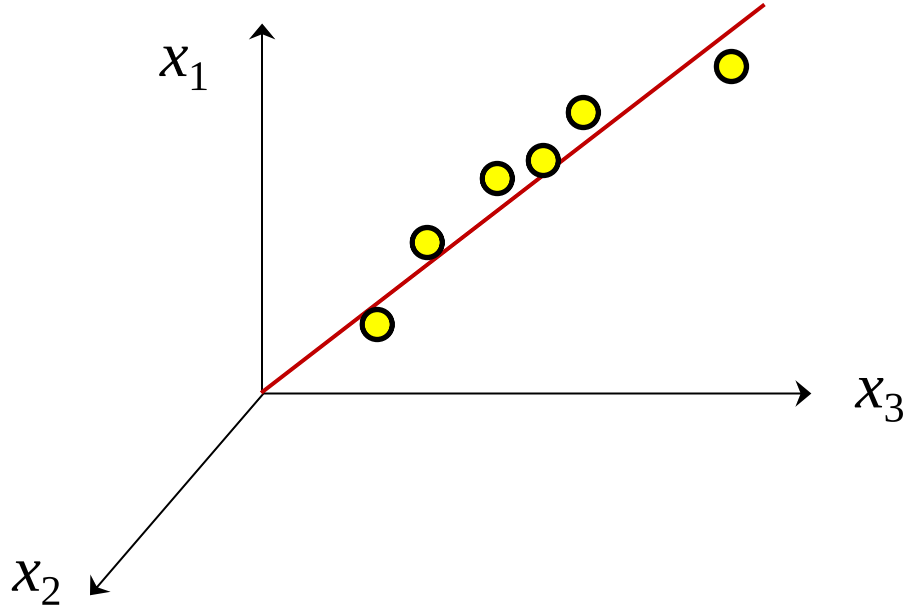
PCA: Toy Example - 2

- Consider the following dataset:

1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

PCA: Toy Example - 2

1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

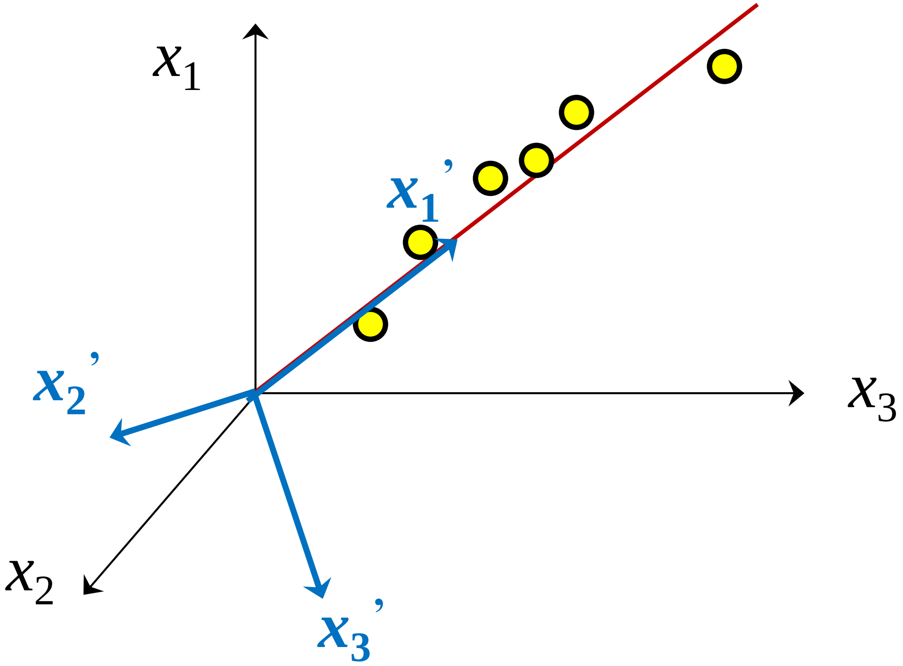


PCA: Toy Example - 2

1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

3.61	7.4	11.1	15.0	18.4	22.4
0.2	0.4	0.9	0.7	0.8	0.3
0.1	0.1	0.1	0.1	0.1	0.1

NOTE: These values are made up. Not exact.



Variance

Data values x	Mean $\bar{x}$	$x - \bar{x}$	$(x-\bar{x})^2$
7	16	−9	81
11	16	−5	25
11	16	−5	25
15	16	−1	1
20	16	4	16
20	16	4	16
28	16	12	144

Variance: s^2 $\sum(x-\bar{x})^2 = 308$

$$s^2 = \frac{\sum(x-\bar{x})^2}{n-1} = \frac{308}{7-1} = \frac{308}{6} =$$

Sample Variance:

$$s^2 = \frac{\sum(x-\bar{x})^2}{n-1}$$

Standard Deviation:

$$s = \sqrt{\frac{\sum(x-\bar{x})^2}{n-1}}$$

n = Sample size

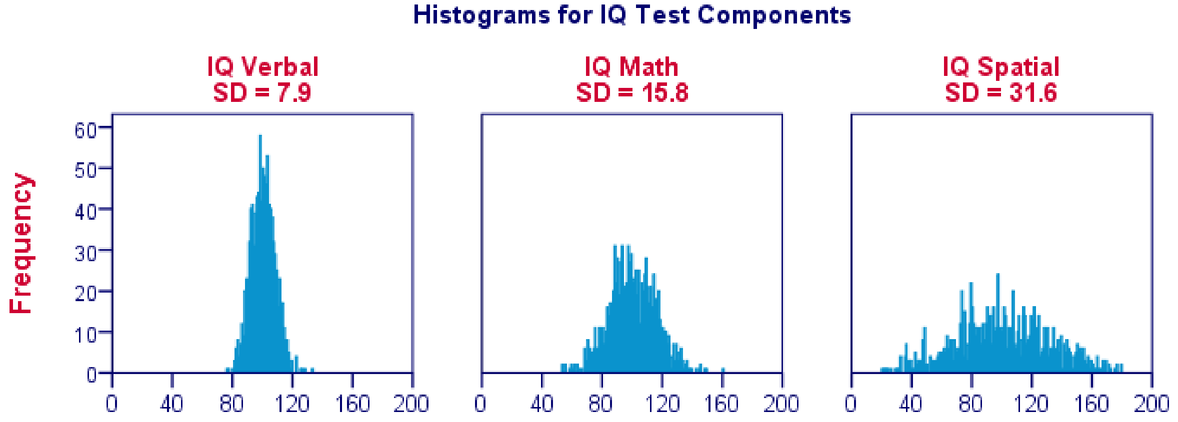
$n = 7$

$$\text{Mean} = \frac{\sum x}{n}$$

$\bar{x} = 16$

Mean = 'Average' value

S.D = Average deviation of samples from mean



Ref: https://www.spss-tutorials.com/img/standard-deviation-example-4.png

Reference: https://youtu.be/179ce7ZzFA8

Covariance : m samples, n features

Vectors 1 and 3

Cell (3, 1) or (1, 3)

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 5 & 4 & 1 \\ 3 & 8 & 6 \end{bmatrix}$$

Covariance

$$\begin{bmatrix} 2.67 & 0.67 & -2.67 \\ 0.67 & 4.67 & 2.33 \\ -2.67 & 2.33 & 4.67 \end{bmatrix}$$

Covariance Matrix

$$\begin{bmatrix} 0.39701 & 0.51117 \\ 0.55582 & 0.93003 \\ 0.59403 & 0.96645 \\ 0.51544 & 0.29759 \\ 0.85313 & 0.18118 \\ 0.88564 & 0.69114 \end{bmatrix}$$

Variance:

$$s^2 = \frac{\sum \left(\overline{X} - X_i\right)^2}{N}$$

Covariance:

$$cov(\mathbf{X},\mathbf{Y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

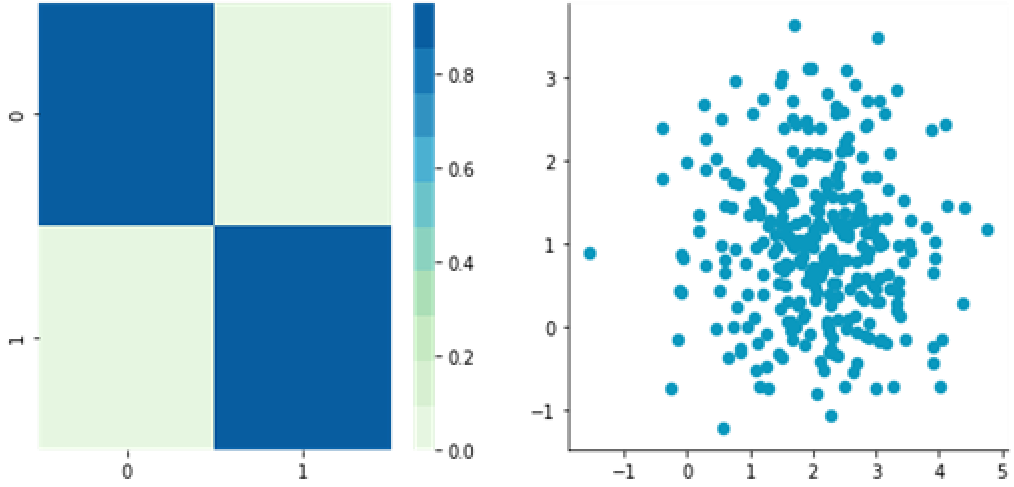
$$Cov(M_a, M_b) = \frac{1}{m}\sum_{i=1}^{m}(q_{i,a} - \overline{q_a})(q_{i,b} - \overline{q_b})$$

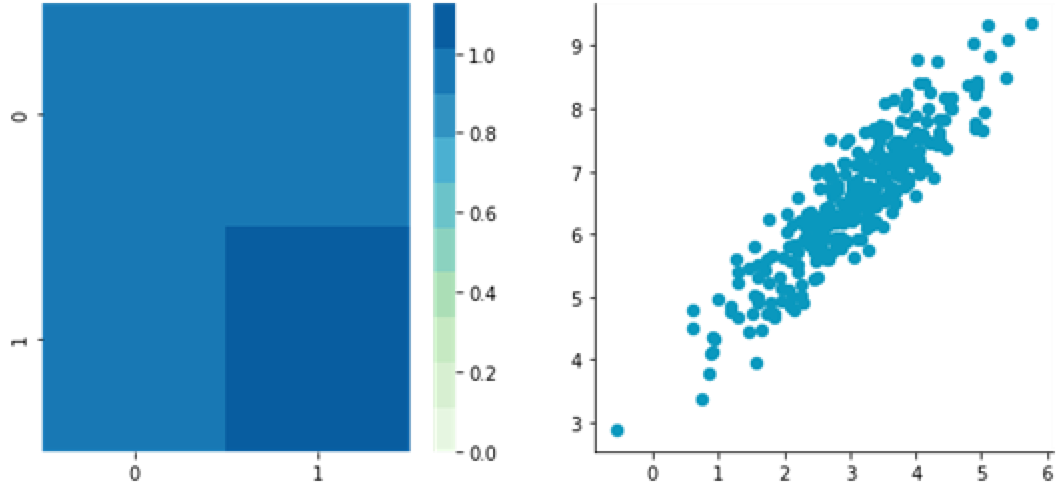
$$\begin{array}{c|cccccc} & M1 & M2 & M3 & \dots & Mn \\ \hline S1 & q_{1,1} & q_{1,2} & q_{1,3} & \dots & q_{1,n} \\ S2 & q_{2,1} & q_{2,2} & q_{2,3} & \dots & q_{2,n} \\ S3 & q_{3,1} & q_{3,2} & q_{3,3} & \dots & q_{3,n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Sm & q_{m,1} & q_{m,2} & q_{m,3} & & q_{m,n} \end{array}$$

$$C = \begin{pmatrix} \mathrm{cov}(M_1, M_1) & \mathrm{cov}(M_1, M_2) & \cdots & \mathrm{cov}(M_1, M_n) \\ \mathrm{cov}(M_2, M_1) & \mathrm{cov}(M_2, M_2) & \cdots & \mathrm{cov}(M_2, M_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}(M_n, M_1) & \mathrm{cov}(M_n, M_2) & \cdots & \mathrm{cov}(M_n, M_n) \end{pmatrix}_{n \times n}$$

N-dimensional Covariance Matrix

Covariance Matrix



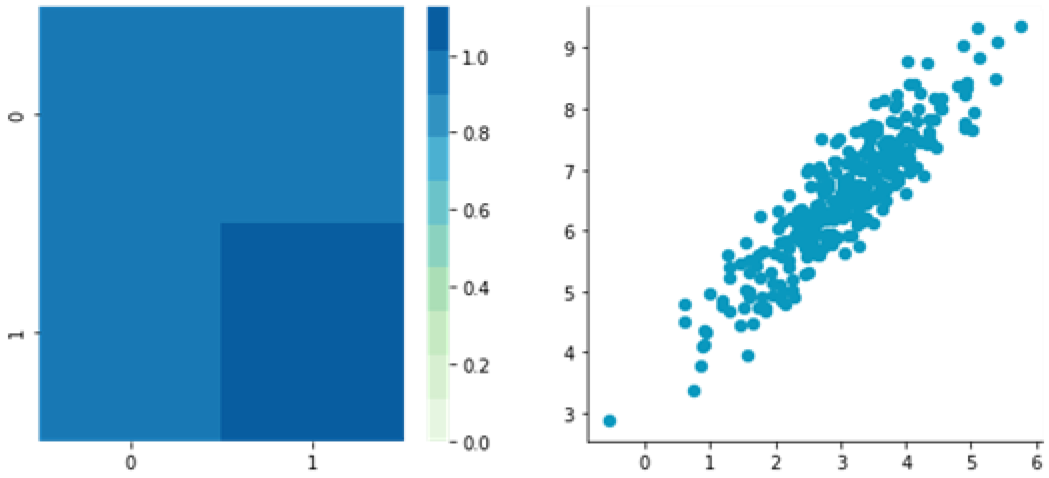


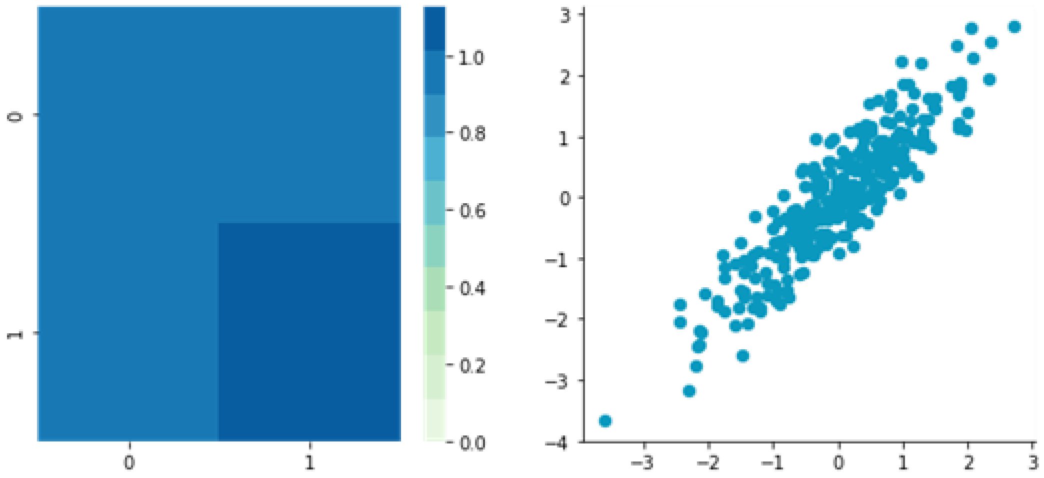
$$C = \begin{bmatrix} +0.95 & -0.04 \\ -0.04 & +0.87 \end{bmatrix}$$

$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

Mean Normalization

$$X' = X - \bar{x}$$

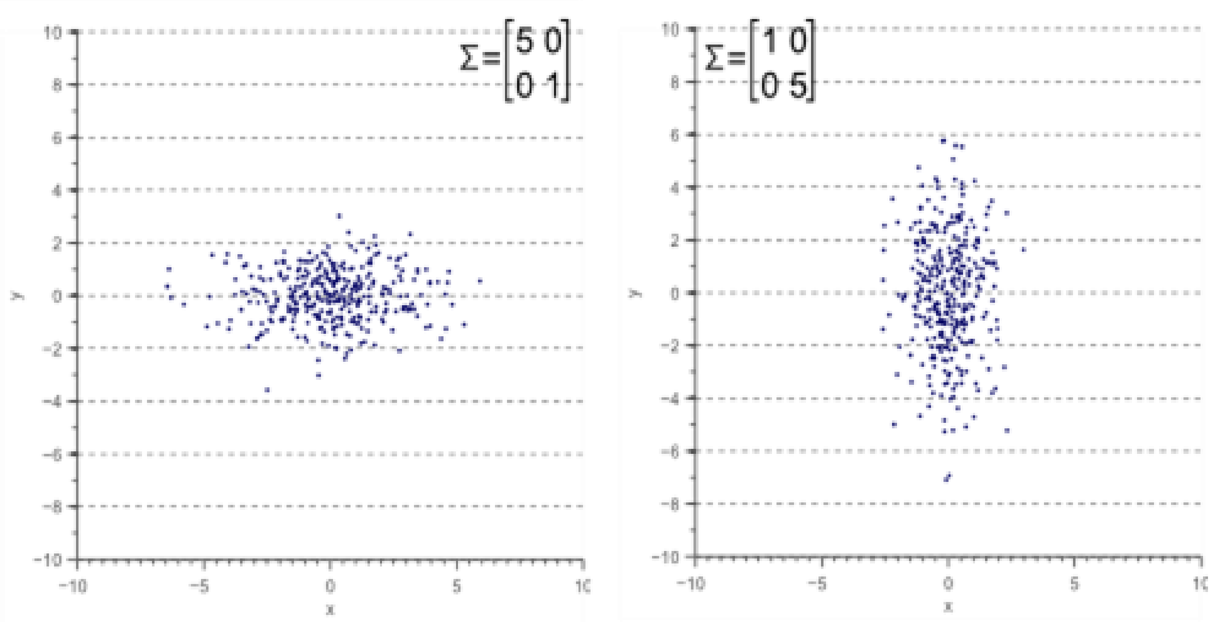




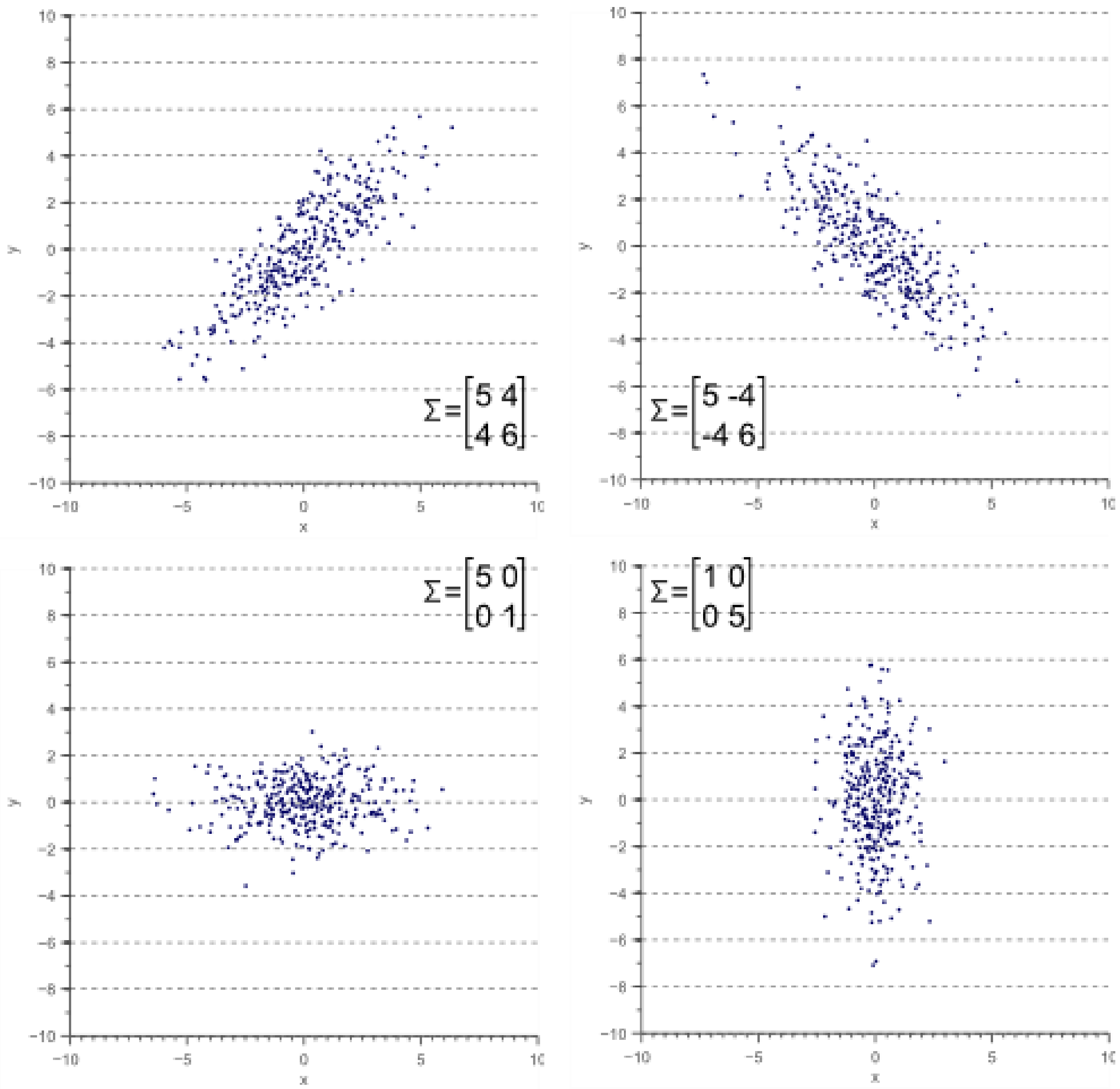
$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

$$C = \begin{bmatrix} +0.95 & +0.92 \\ +0.92 & +1.12 \end{bmatrix}$$

Covariance Matrix encodes spread and orientation of data

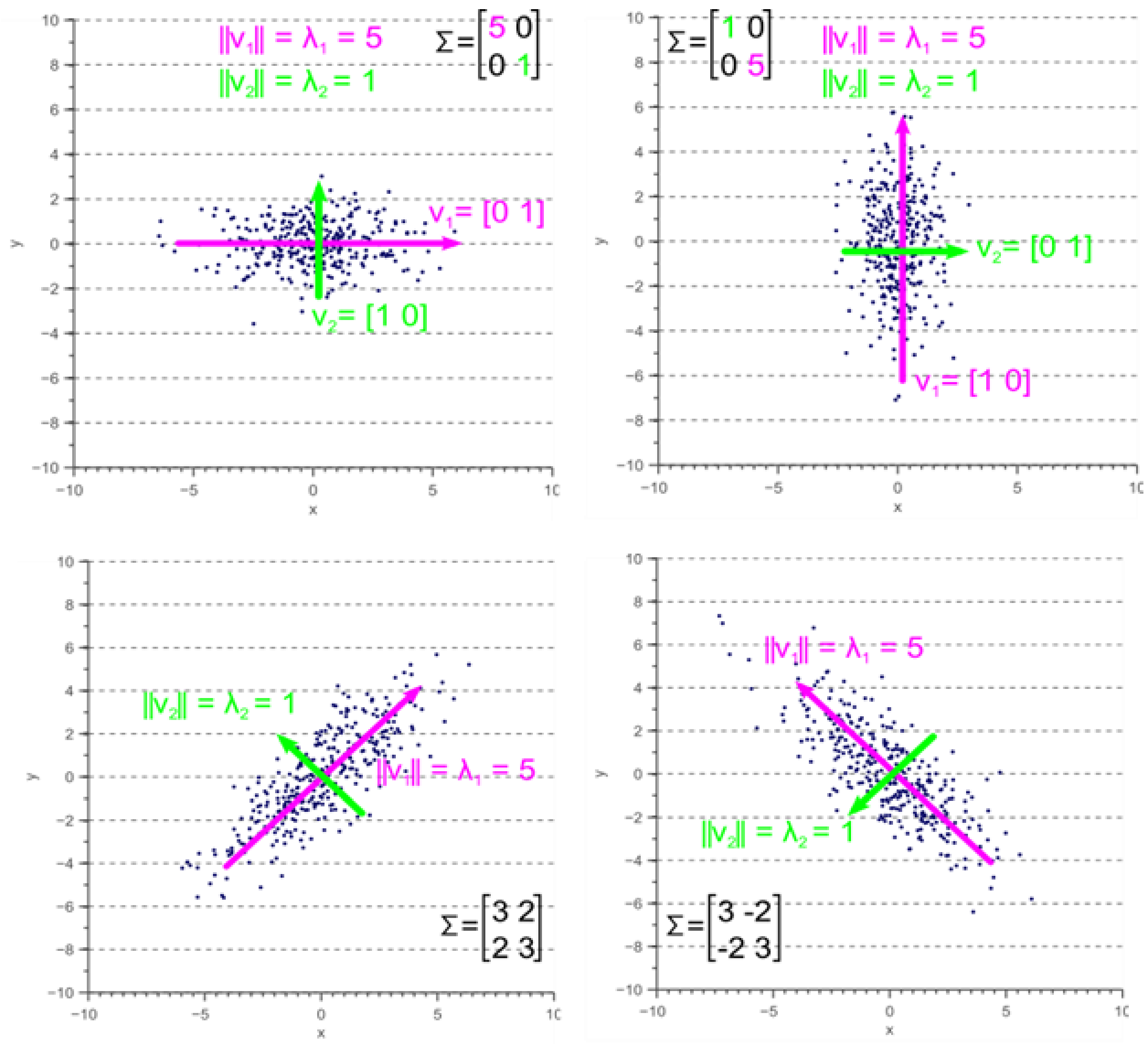


Covariance Matrix encodes spread and orientation of data



Eigen-analysis of Covariance Matrix

v_1, v_2: *Principal Components*



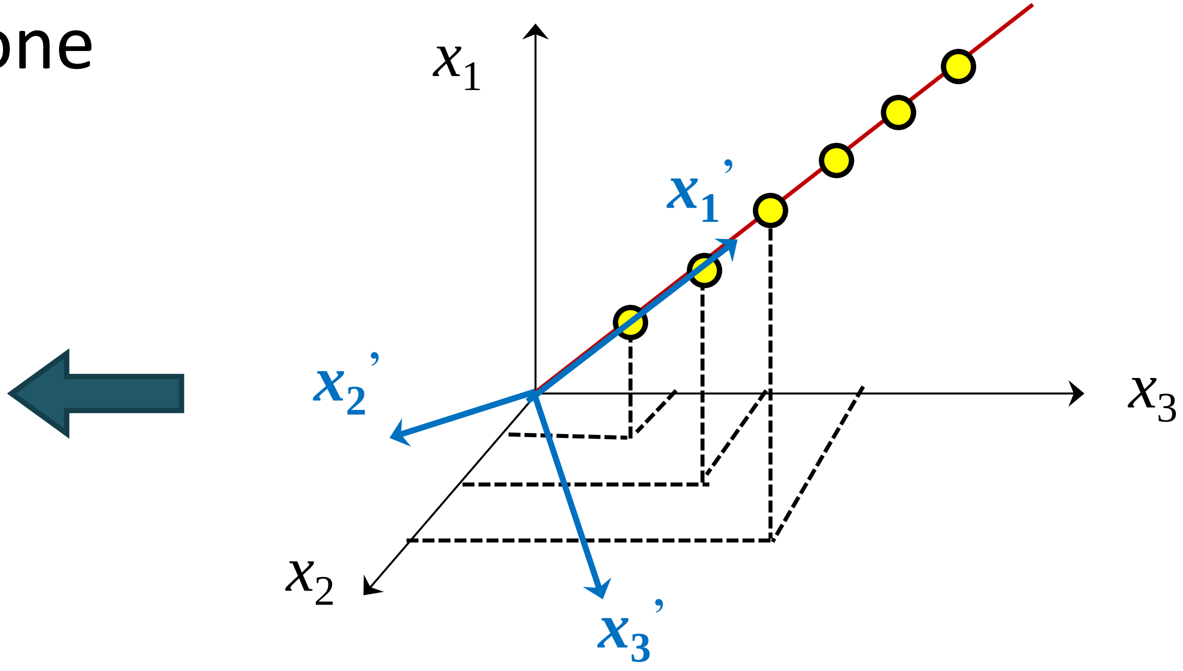
$$\Sigma \vec{v} = \lambda \vec{v}$$

Value of λ indicates `variance' (spread) in direction of eigenvector v associated with λ

PCA: A Toy Example

- Consider a new co-ordinate system with one axis along the line
- All co-ordinates except the first one are zeros now.

3.7	7.5	11.2	15	18.7	22.4
0	0	0	0	0	0
0	0	0	0	0	0

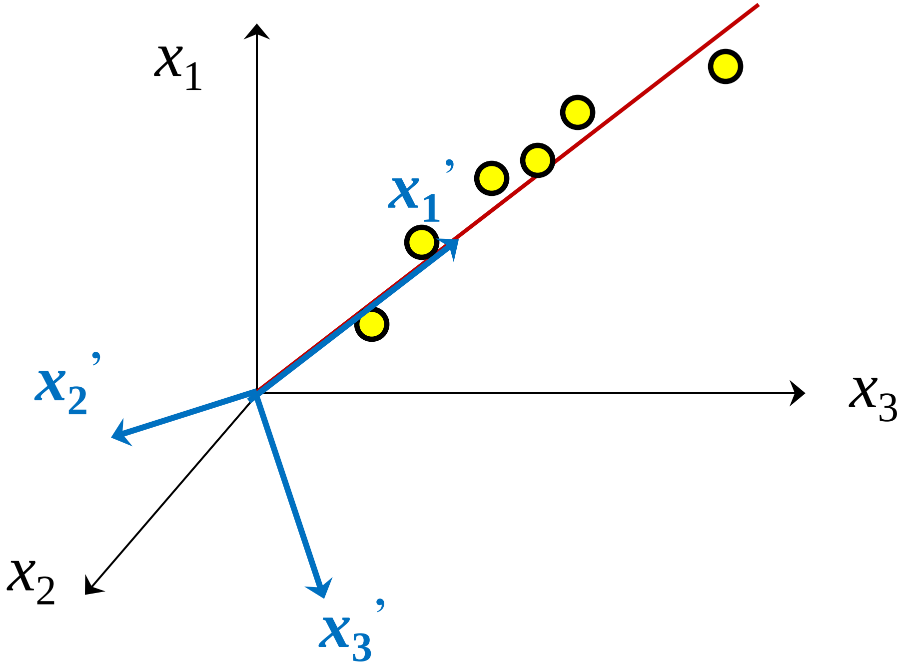


PCA: Toy Example - 2

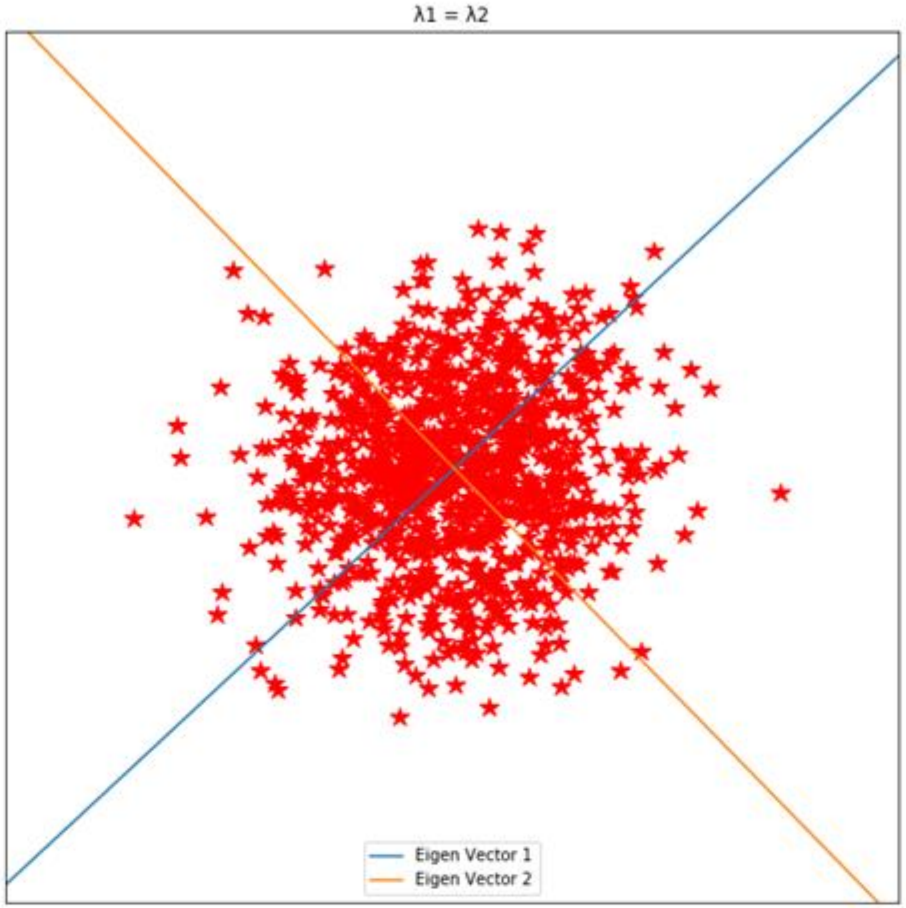
1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

3.61	7.4	11.1	15.0	18.4	22.4
0.2	0.4	0.9	0.7	0.8	0.3
0.1	0.1	0.1	0.1	0.1	0.1

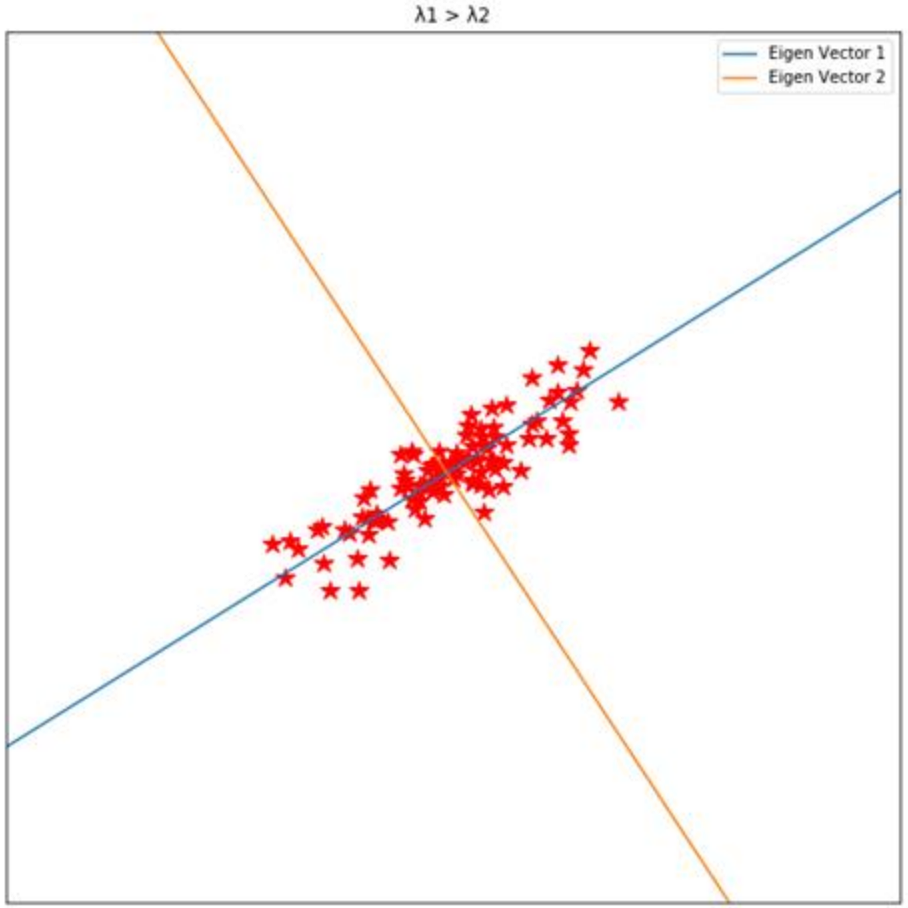
NOTE: These values are made up. Not exact.



Covariance, Eigen Values and Vectors



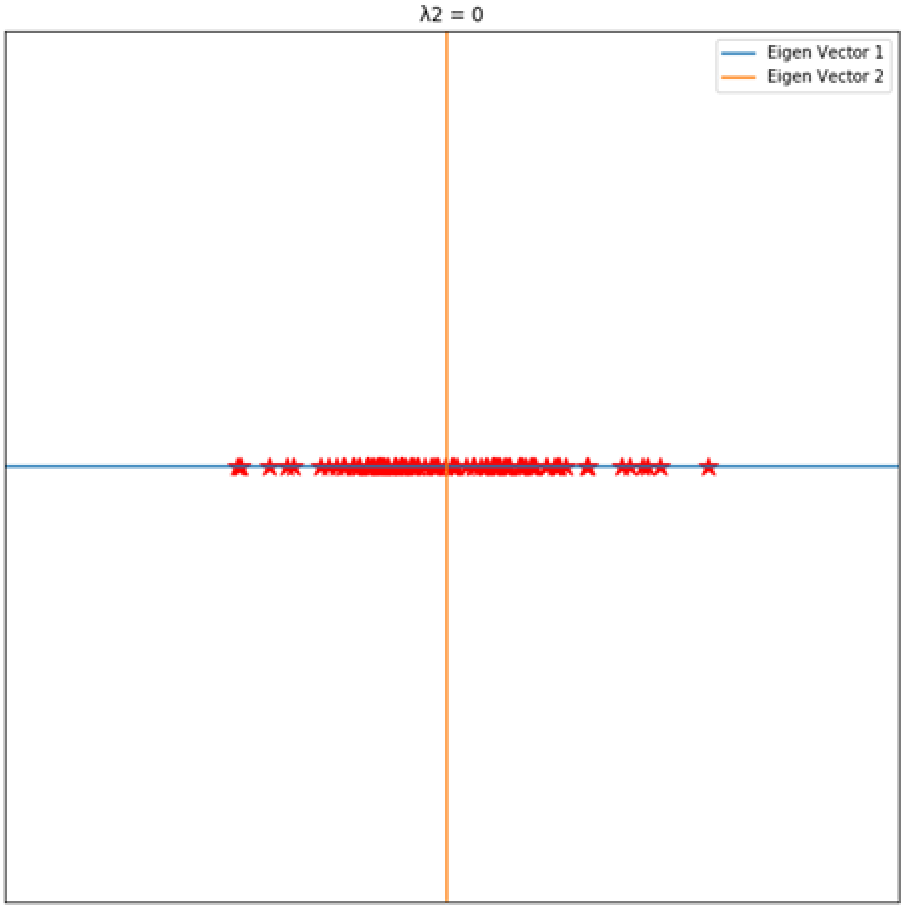
Both the Eigen values are equal
(Distribution is circular)

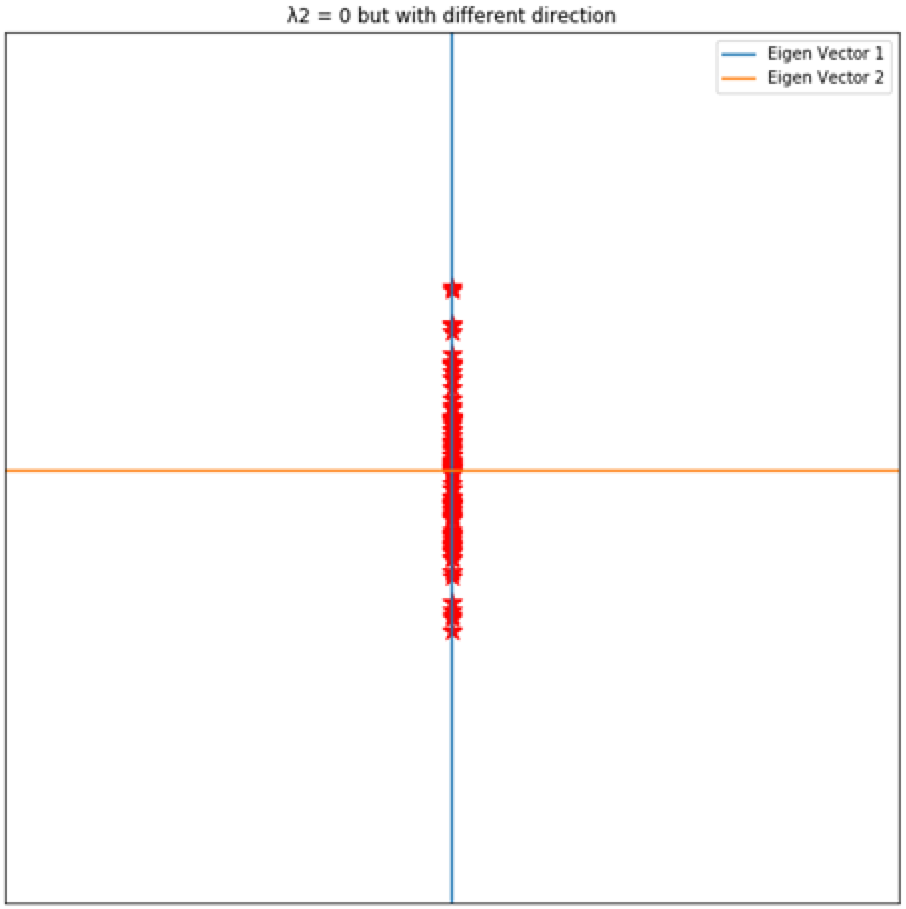


One Eigen value is greater than the other
(Distribution is elongated in the direction of that Eigen vector)

Code Available

Covariance, Eigen Values and Vectors



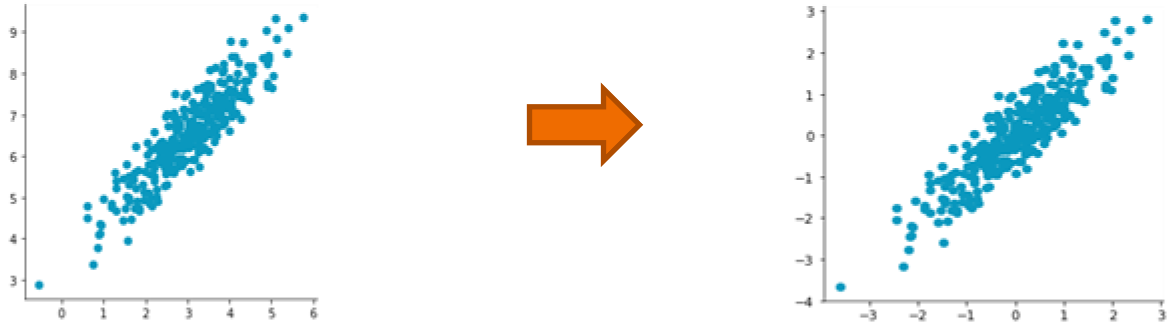


Only one Eigen value is non-zero, distribution of data will align on that Eigen vector

The PCA Recipe

$$\begin{bmatrix} 0.39701 & 0.51117 \\ 0.55582 & 0.93003 \\ 0.59403 & 0.96645 \\ 0.51544 & 0.29759 \\ 0.85313 & 0.18118 \\ 0.88564 & 0.69114 \end{bmatrix}$$

1. Center the data



$$X' = X - \bar{x}$$

2. Compute the covariance matrix

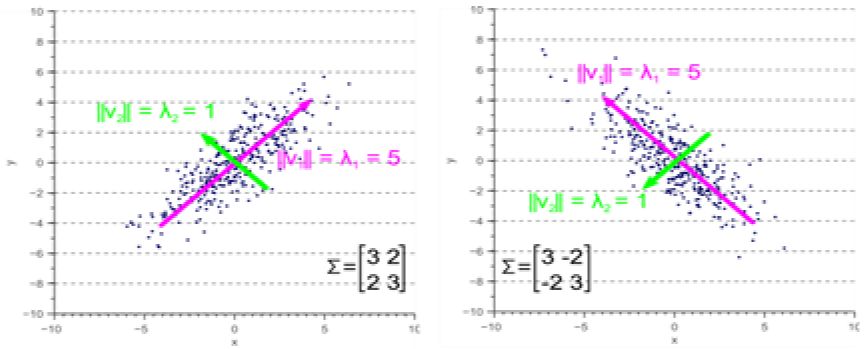
$$\left(\begin{array}{cccccc} & M1 & M2 & M3 & \dots & Mn \\ S1 & q_{1,1} & q_{1,2} & q_{1,3} & \dots & q_{1,n} \\ S2 & q_{2,1} & q_{2,2} & q_{2,3} & \dots & q_{2,n} \\ S3 & q_{3,1} & q_{3,2} & q_{3,3} & \dots & q_{3,n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ Sm & q_{m,1} & q_{m,2} & q_{m,3} & & q_{m,n} \end{array}\right) \Rightarrow C = \begin{pmatrix} \mathrm{cov}(M_1,M_1) & \mathrm{cov}(M_1,M_2) & \cdots & \mathrm{cov}(M_1,M_n) \\ \mathrm{cov}(M_2,M_1) & \mathrm{cov}(M_2,M_2) & \cdots & \mathrm{cov}(M_2,M_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}(M_n,M_1) & \mathrm{cov}(M_n,M_2) & \cdots & \mathrm{cov}(M_n,M_n) \end{pmatrix}_{n \times n}$$

N-dimensional Covariance Matrix

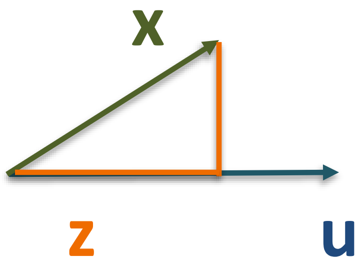
The PCA Recipe

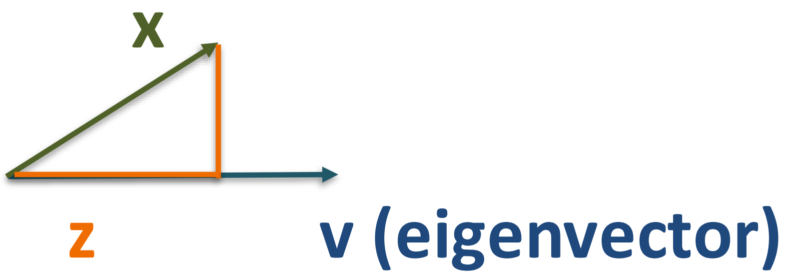
3. Compute Eigenvectors and Eigenvalues of Covariance Matrix Σ

$$\Sigma\vec{v} = \lambda\vec{v}$$



4. Project data onto eigenvectors to obtain new coordinates

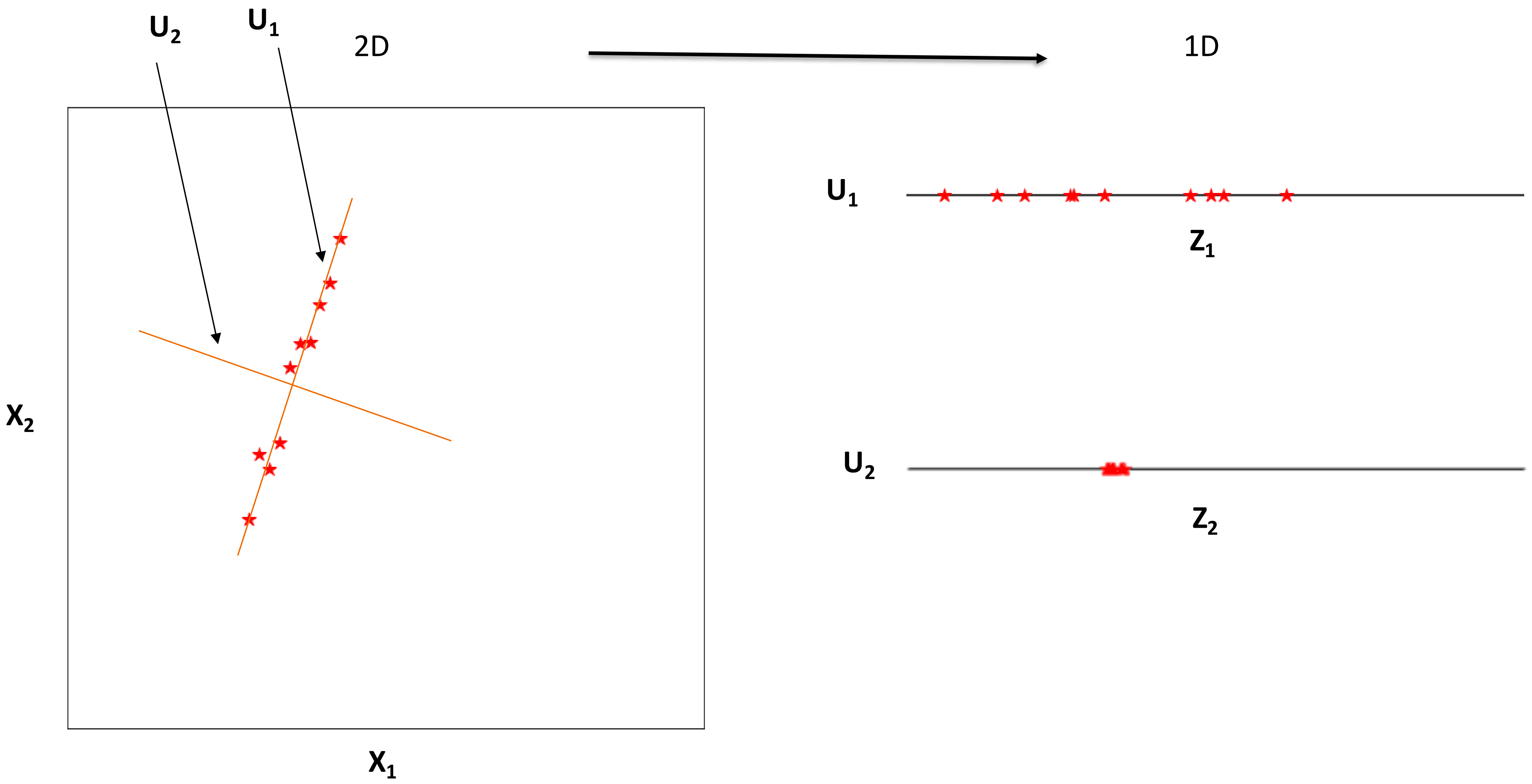




$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & v_1^T & \cdot & \cdot \\ \cdot & \cdot & v_2^T & \cdot & \cdot \\ \cdot & \cdot & v_3^T & \cdot & \cdot \\ \cdot & \cdot & v_4^T & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

New coordinates

Old coordinates

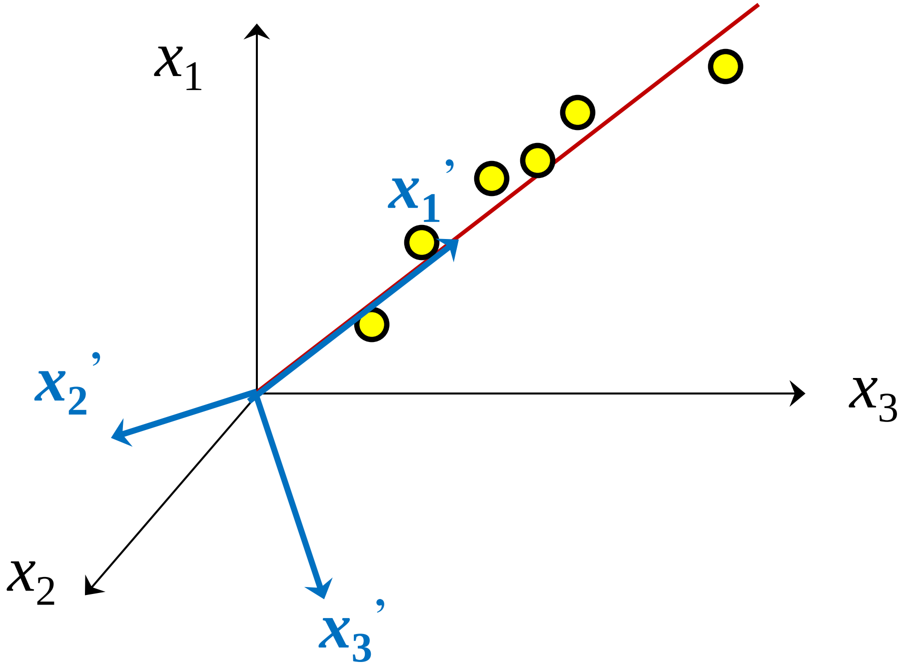

U2
U1
2D
1D
X2
X1
U1
Z1
U2
Z2

PCA: Toy Example - 2

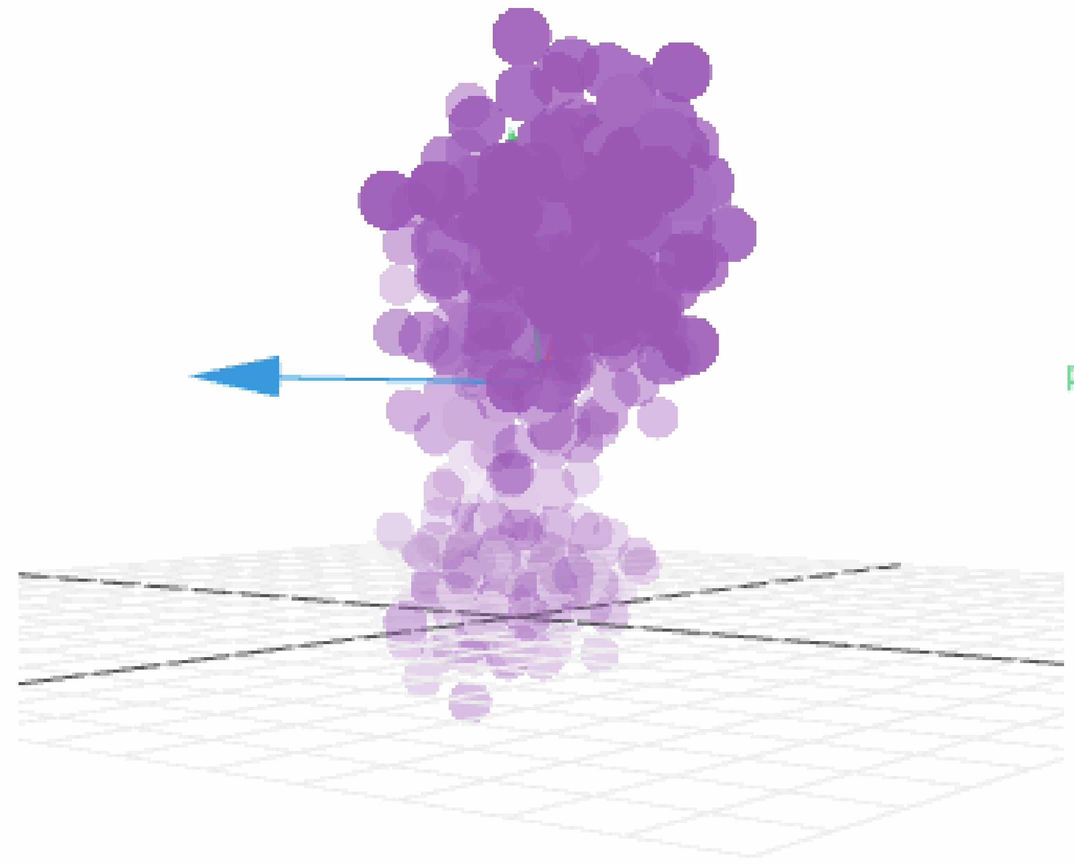
1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

3.61	7.4	11.1	15.0	18.4	22.4
0.2	0.4	0.9	0.7	0.8	0.3
0.1	0.1	0.1	0.1	0.1	0.1

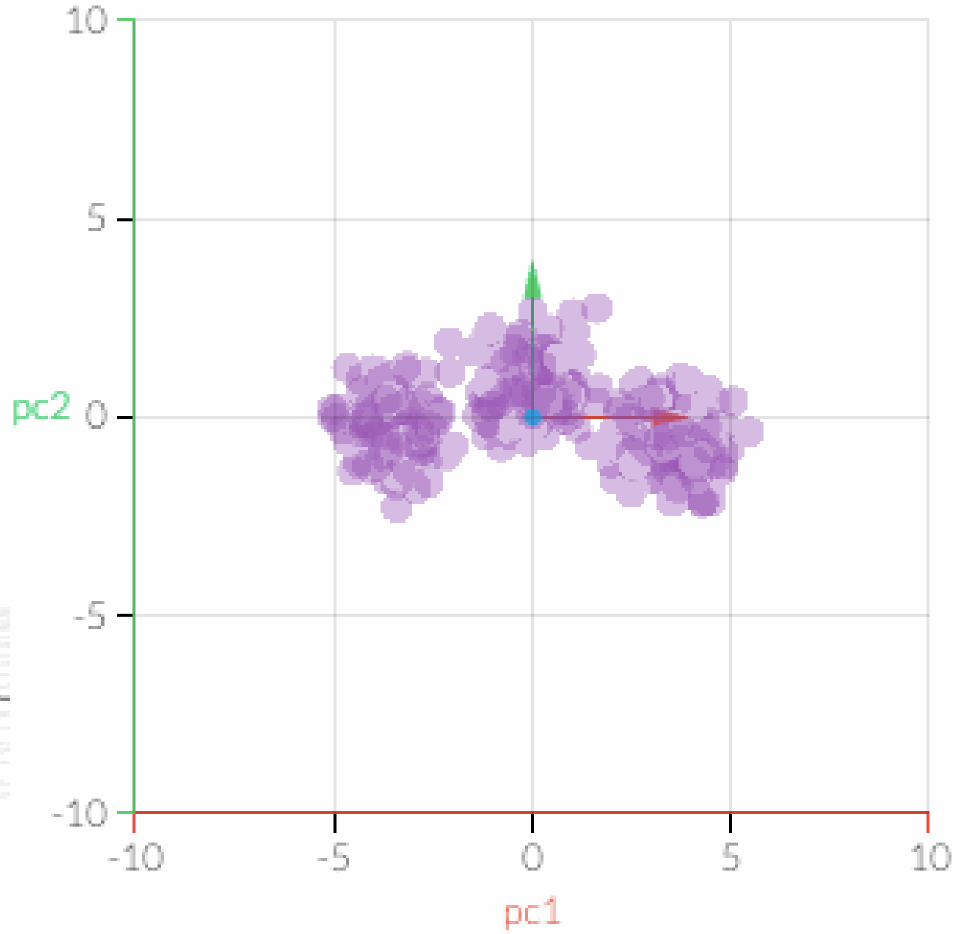
NOTE: These values are made up. Not exact.



3D to 2D



X1, X2, X3



Z1, Z2

PCA based Feature Extraction

$$\underbrace{\begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ . \\ . \\ z_r \end{bmatrix}}_{Z}^{r \times 1} = \underbrace{\begin{bmatrix} u_{11} & u_{12} & u_{13} & . & . & . & . & . & u_{1d} \\ u_{21} & u_{22} & u_{23} & . & . & . & . & . & u_{2d} \\ u_{31} & u_{32} & u_{33} & . & . & . & . & . & u_{3d} \\ . & & & & & & & & \\ u_{r1} & u_{r2} & u_{r3} & . & . & . & . & . & u_{rd} \end{bmatrix}}_{U}^{r \times d} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ . \\ . \\ . \\ . \\ x_d \end{bmatrix}}_{X}^{d \times 1}$$

Each row in U is an eigen vector of co-variance Matrix

Appreciating PCA: Two Questions

$$\text{Eg.} \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{d} \lambda_i} > 0.90$$

- How many Eigen vectors to select?
  - Ans: Eigen Vectors corresponding to the larger Eigen values
- How much information is lost? Can we recover the old data/information from the new?

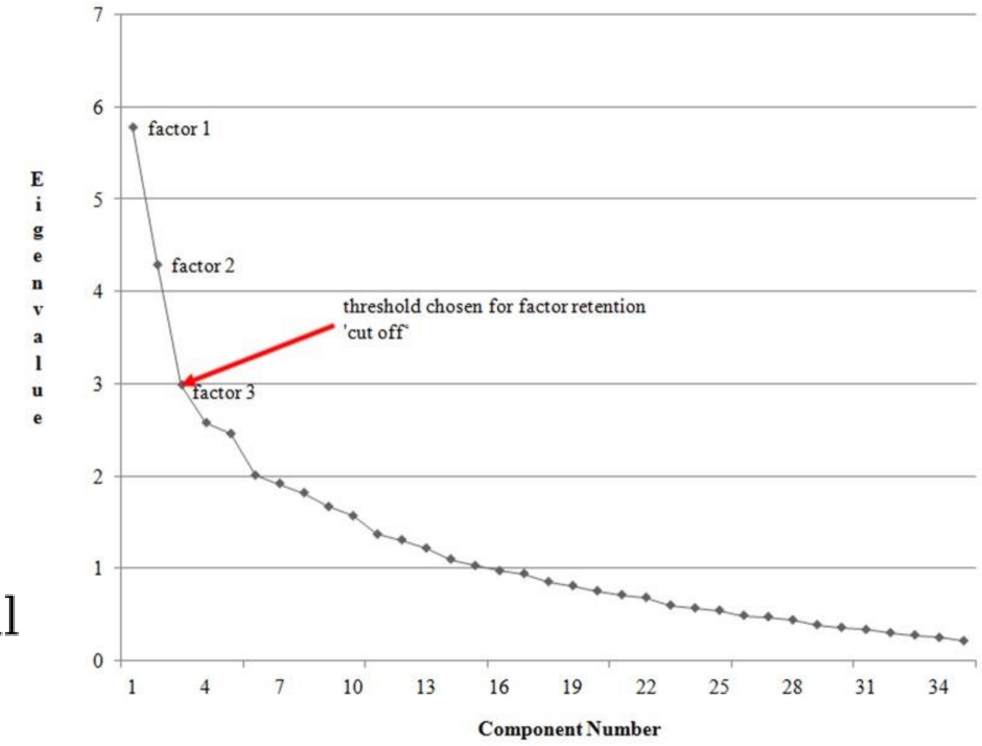
$$\mathbf{x} = z_1\mathbf{u_1} + z_2\mathbf{u_2} + z_3\mathbf{u_3} + z_4\mathbf{u_4}$$

$$\mathbf{x} = z_1\mathbf{u_1} + z_2\mathbf{u_2} + z_3\mathbf{u_3} + z_4\mathbf{u_4}$$

$$\mathbf{x}' = z_1\mathbf{u_1} + z_2\mathbf{u_2}$$

$$\text{Loss in Information} = ||\mathbf{x} - \mathbf{x}'||$$

Note: z_3 and z_4 are small and also λ_3 and λ_4 are small



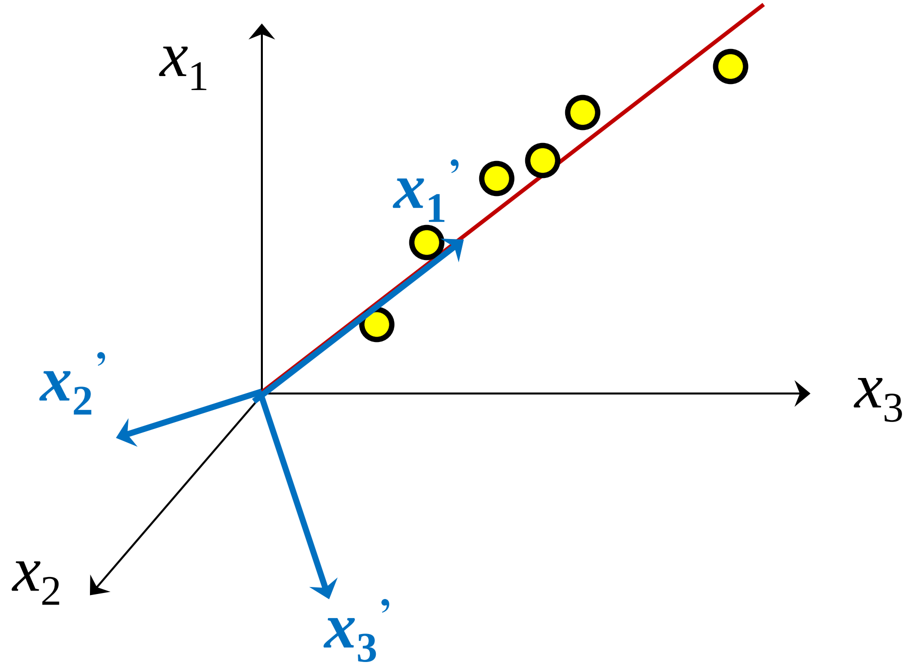
https://www.researchgate.net/profile/Collins_Ouma/publication/263548435/figure/fig1/AS:202705835433990@1425340245309/A-Scree-plot-of-eigenvalues-of-the-unrotated-factors-displaying-an-elbow-of-the-plot.png

PCA: Toy Example - 2

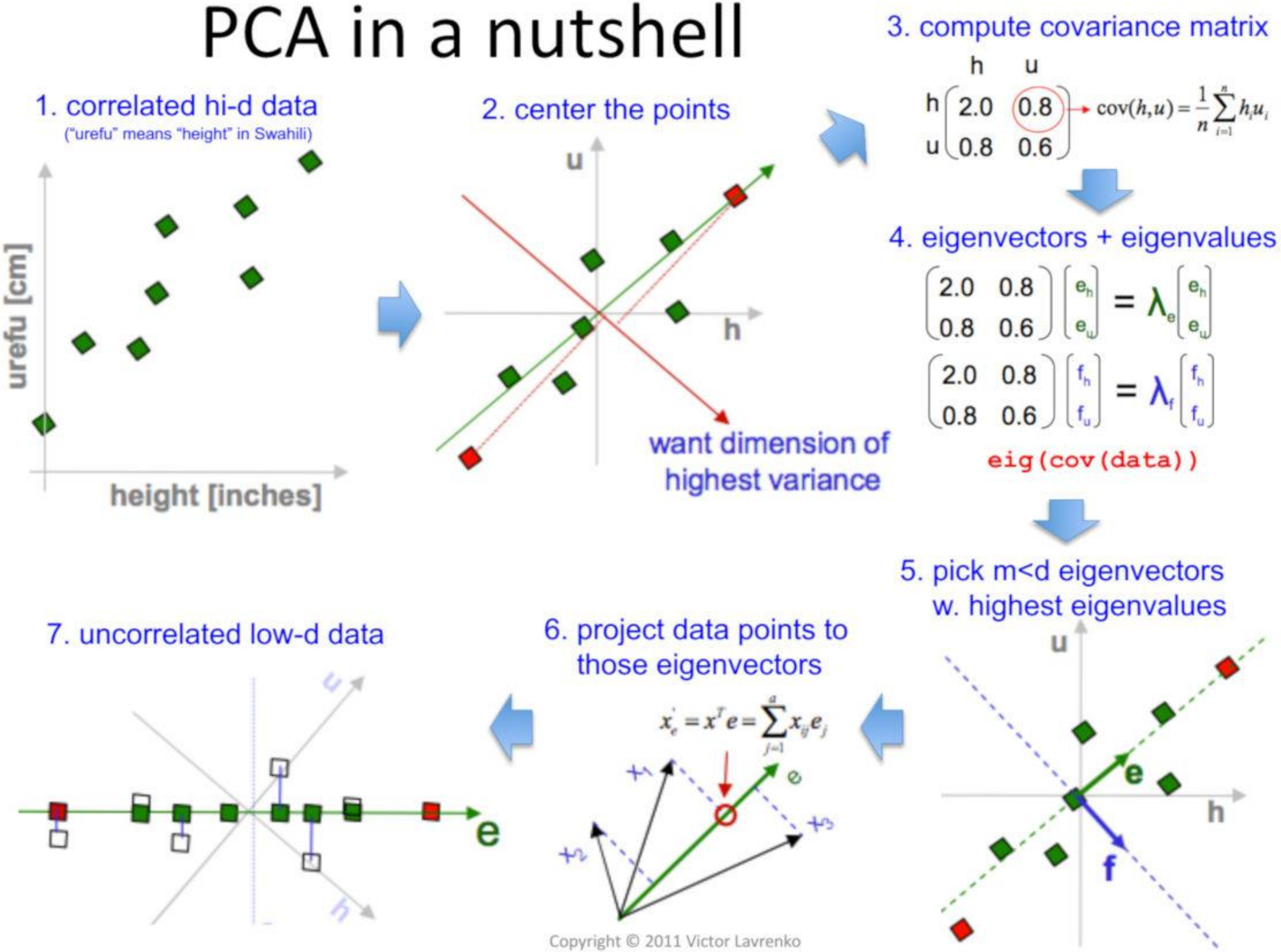
1	4	3	5.7	5.1	2.2
2	7.9	5.8	12	9.9	4.1
3.1	12	9	18	15	6.3

3.61	7.4	11.1	15.0	18.4	22.4
0.2	0.4	0.9	0.7	0.8	0.3
0.1	0.1	0.1	0.1	0.1	0.1

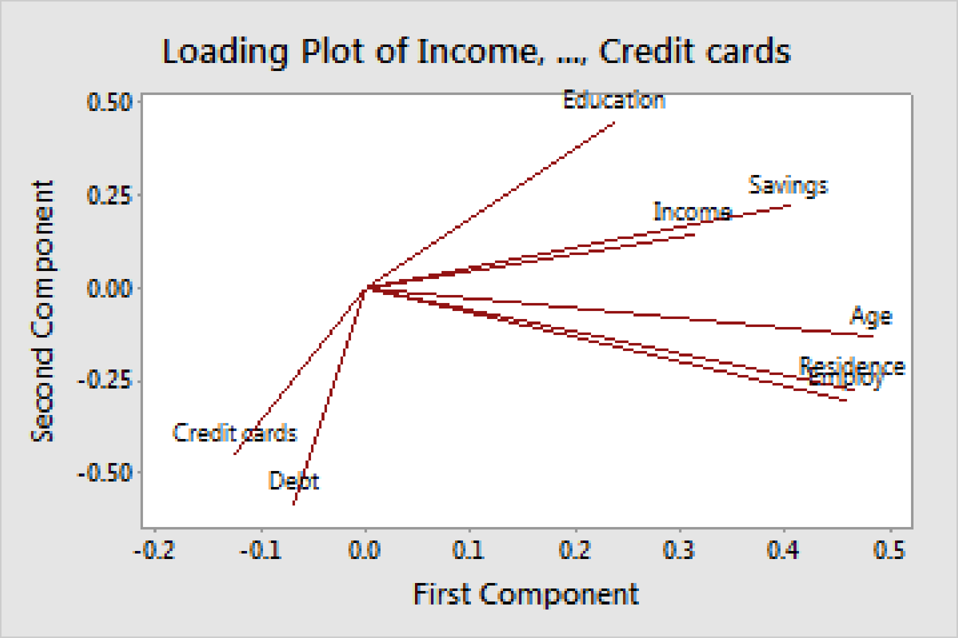
NOTE: These values are made up. Not exact.



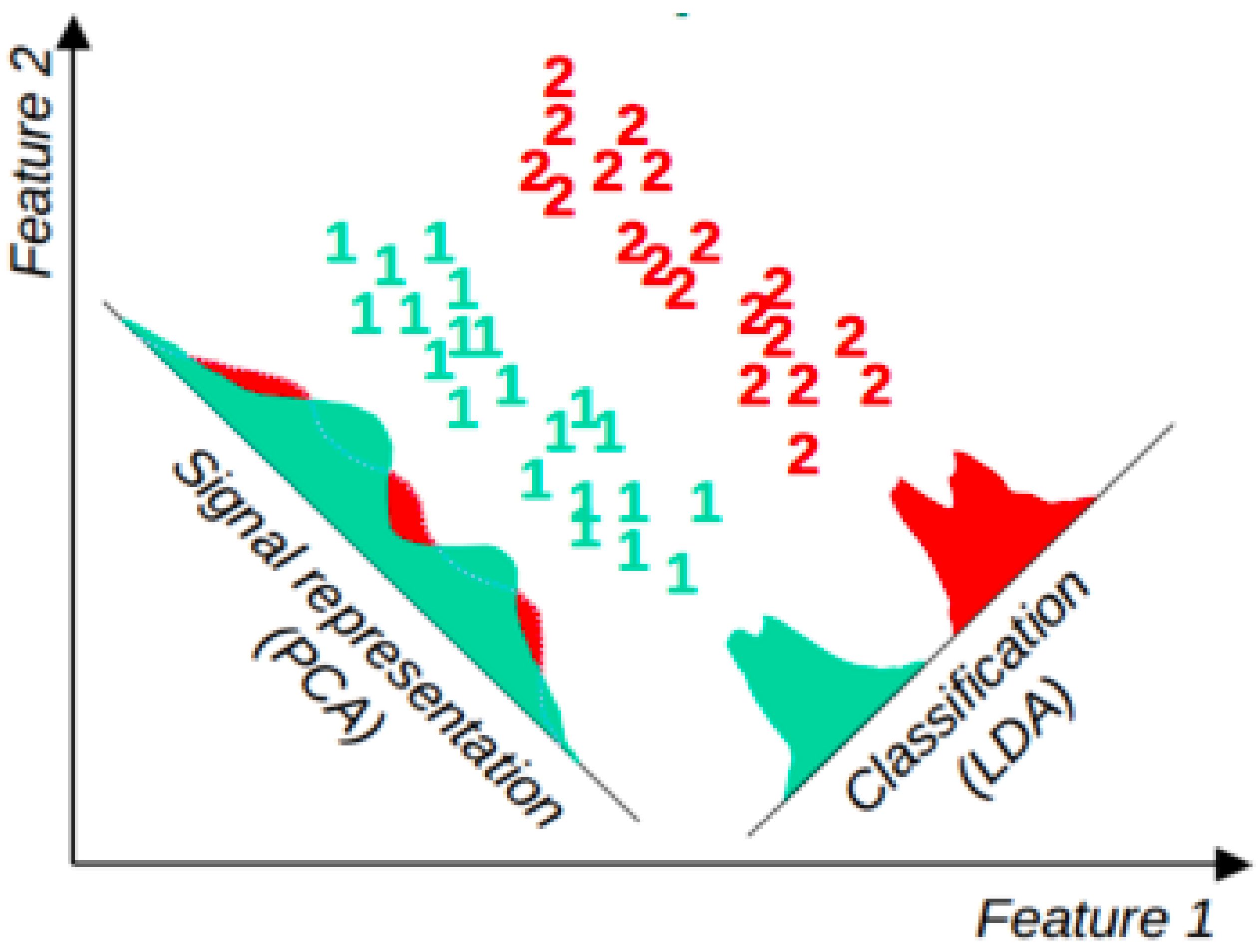
PCA in a nutshell



Analysis using ‘factor loadings’

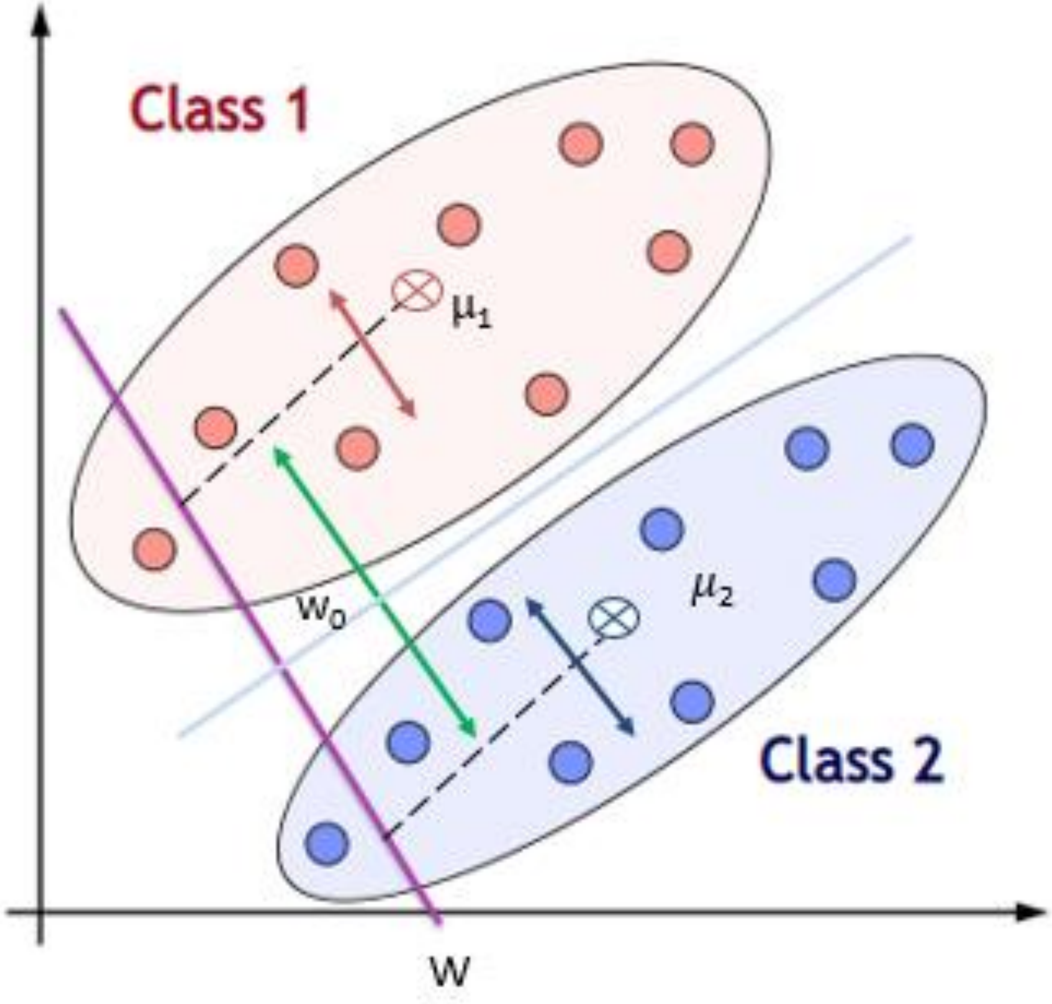


Linear Discriminant Analysis (LDA)



Source: https://www.mygreatlearning.com/blog/linear-discriminant-analysis-or-lda/

Linear Discriminant Analysis (LDA)



- Maximize distance between classes
- Minimize distance within a class
- Criterion: $J(w) = \dfrac{w^T S_b w}{w^T S_w w}$

S_b= between-class scatter matrix

S_w= within-class scatter matrix

- Vector w is a solution of generalized Eigen value problem:

$$S_b w = \lambda\, S_w w$$

- Classification function:

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \underset{\text{Class 2}}{\overset{\text{Class 1}}{\gtrless}} 0$$

Linear Discriminant Analysis (LDA)

- Also known as "Fisher Discriminant"

- Does dimensionality Reduction
  - Also use the label "y"
  - Or Supervised Dimensionality Reduction

- There are also nonlinear Dimensionality Reduction schemes

Summary

- We often get raw data/logs/measurements

- Two problems:
  - Select good ones out of all
  - Define new ones as linear combination of existing

- Dimensionality reduction for
  - Compression/compaction
  - Classification/Discrimination

Selecting and Extracting Features

$$\begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting first and third feature

$$\begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting first and fourth feature

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 \\ 0.0 & 0.4 & 0.2 & 1.7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

New Features as linear combination of old Features

$$X' = AX$$

For PCA: Rows are Eigen vectors of the covariance matrix.

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cdots & u_1^T & \cdots \\ \cdots & u_2^T & \cdots \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

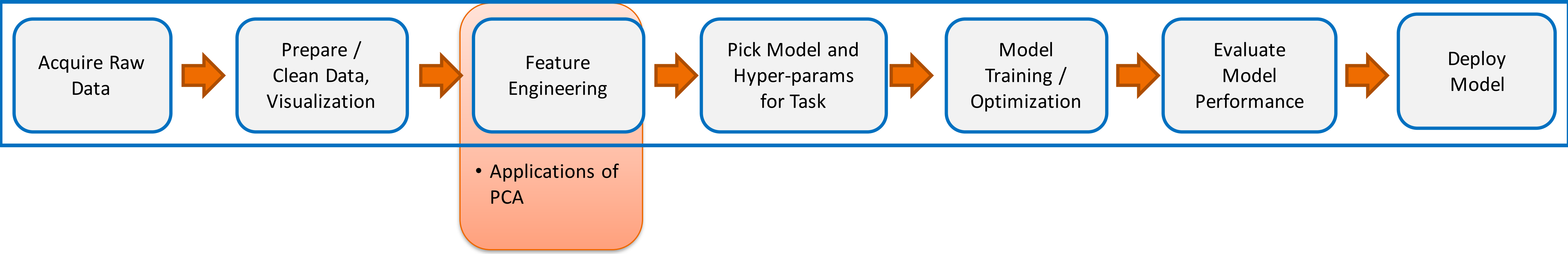
Journey so far…

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

- ✓ Dimensionality Reduction
- ✓ Introduction to PCA
- ✓ PCA Algorithm

42
42

Questions?



Dimensionality Reduction

Applications of PCA

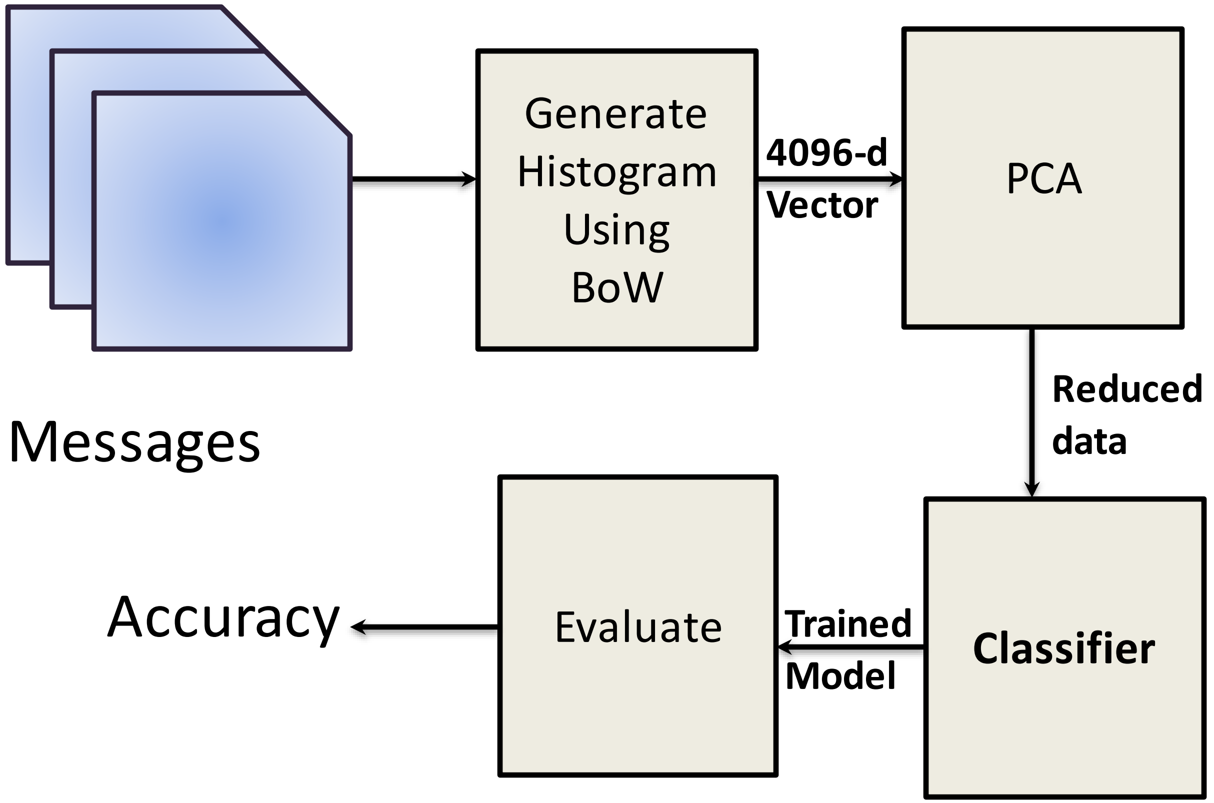
Case Study

Classification after PCA

Code Available

Case Study: PCA and Classification

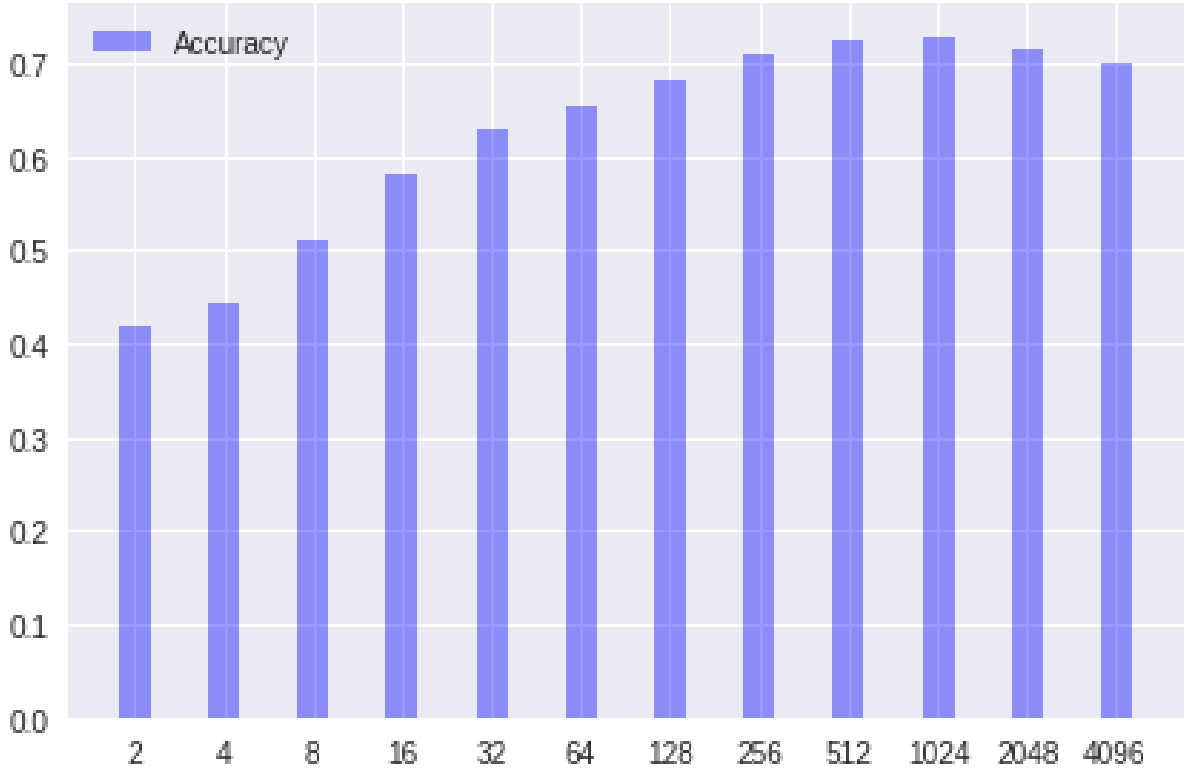
- Text data with 20 classes
- Preprocessing:
  - Find the Histograms for Each Document using Bag of words
  - Apply PCA to reduce the dimensions
- Train the classifier on the reduced data
- Find the Accuracy to Evaluate the model



Code Available

Effect of PCA on the Accuracy

- Change r (dimensions in projected space) to 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096
- With just 3% (32) of the total dimensions (4096), comparable accuracies are obtained

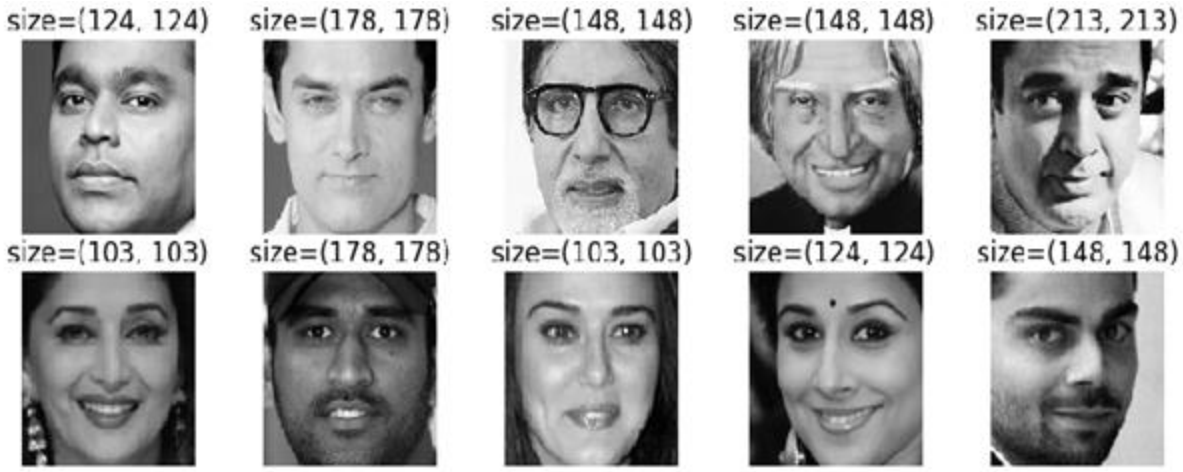


Case Study

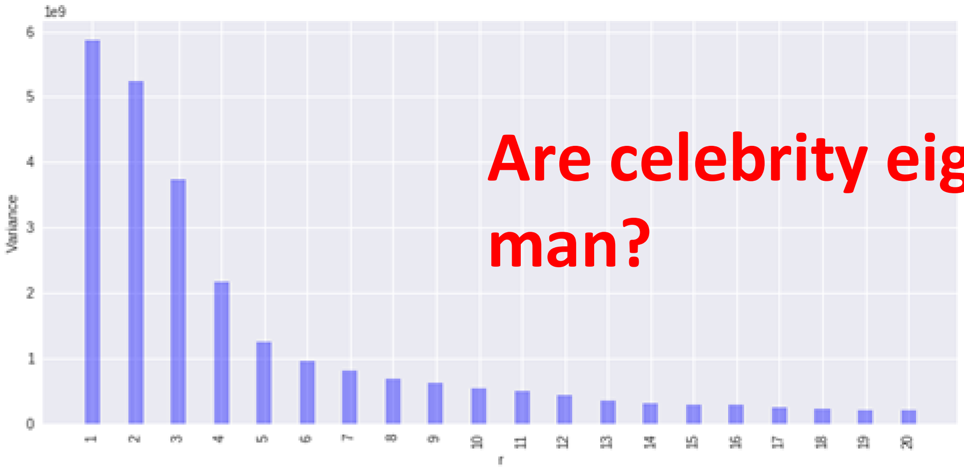
Recognition and data compression with Eigen faces

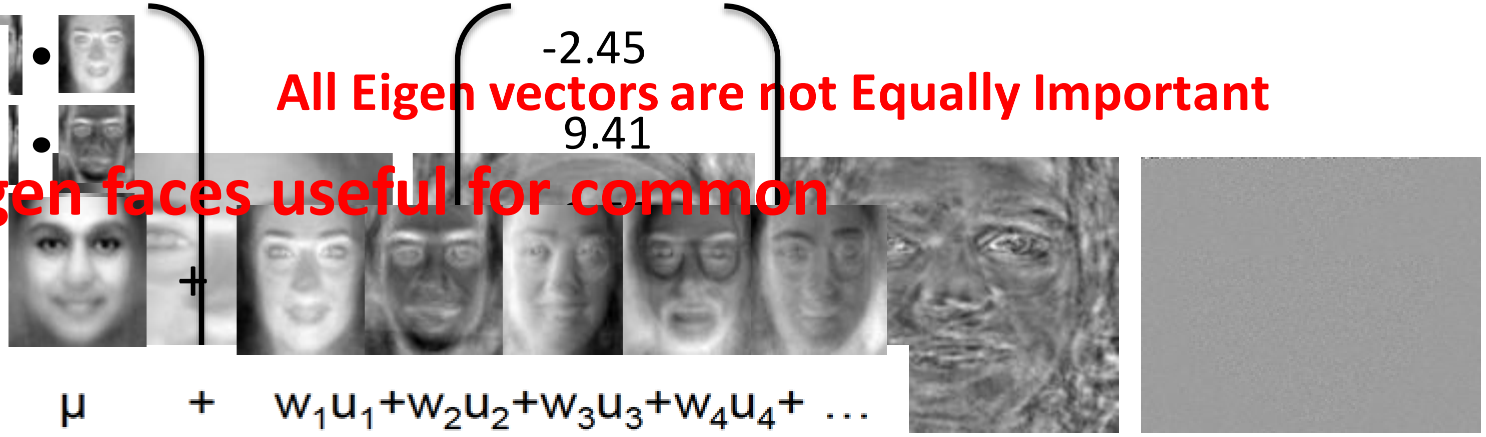
Code Available

Recognize Indian Celebrities



- Rescale and Find mean of the face
- Find the Eigen Faces
  - All Eigen faces are not equally important
- Find the weights to represent the face in Eigen space
- Reconstruct the image





-2.45

All Eigen vectors are not Equally Important

9.41

Are celebrity eigen faces useful for common man?

-0.005

$\mu + w_1u_1 + w_2u_2 + w_3u_3 + w_4u_4 + \ldots$

en Vector 1

Eigen Vector 100

Eigen Vector 150

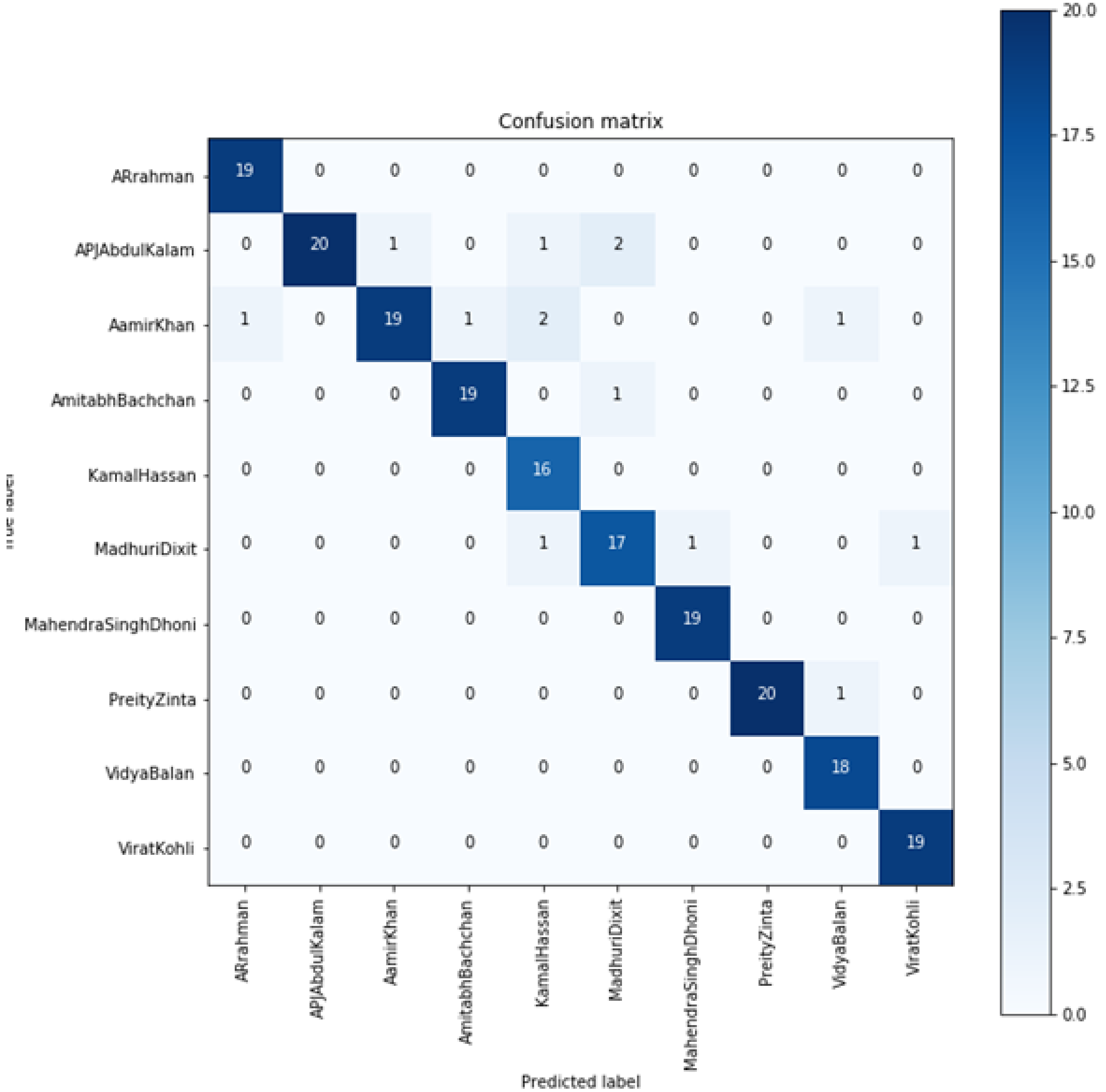
Eigen Vector 200

Code Available

Classification

Training images: 400
Test images: 200

Accuracy: 96%



Three Viewpoints

- Maximal variance on the new features.
- Data Compression and Minimal Reconstruction Error.
- Orthogonal Line Fitting.

Code Available

Application: Compression

12 X 12 Patches = 144D

(a) 144 (b) 60 (c) 16

(d) 6 (e) 3

Question/Quiz

- Q1: Dimensionality reduction: 144 to 3
  - What is the compression ratio? Is it really 3/144?
  - Do you think you can get a compression scheme for compressing a 100 X 100 color (or 3* 10000 Bytes) to 3*10000 * 3/144 Bytes?

- Q2: Then why PCA is not replacing JPEG or other similar ones?
  - Why are we stuck with these "old" standards?

Summary: PCA

- Compute Eigen values and Eigen Vectors of the covariance matrix
- Select the principal components
- Define new features.
- Will classification performance improve? Depends:
  - Do we throw away signal?
  - Do we throw away noise?

Summary

Acquire Raw Data → Prepare / Clean Data, Visualization → Feature Engineering → Pick Model and Hyper-params for Task → Model Training / Optimization → Evaluate Model Performance → Deploy Model

Feature Engineering

- ✓ Dimensionality Reduction
- ✓ Introduction to PCA
- ✓ PCA Algorithm
- ✓ Applications of PCA

Thanks!!

Questions?