COSC 311: Introduction to Data Visualization and Interpretation

Principal component analysis (PCA)

Dr. Shuangquan (Peter) Wang
(spwang@salisbury.edu)

Department of Computer Science

Salisbury University

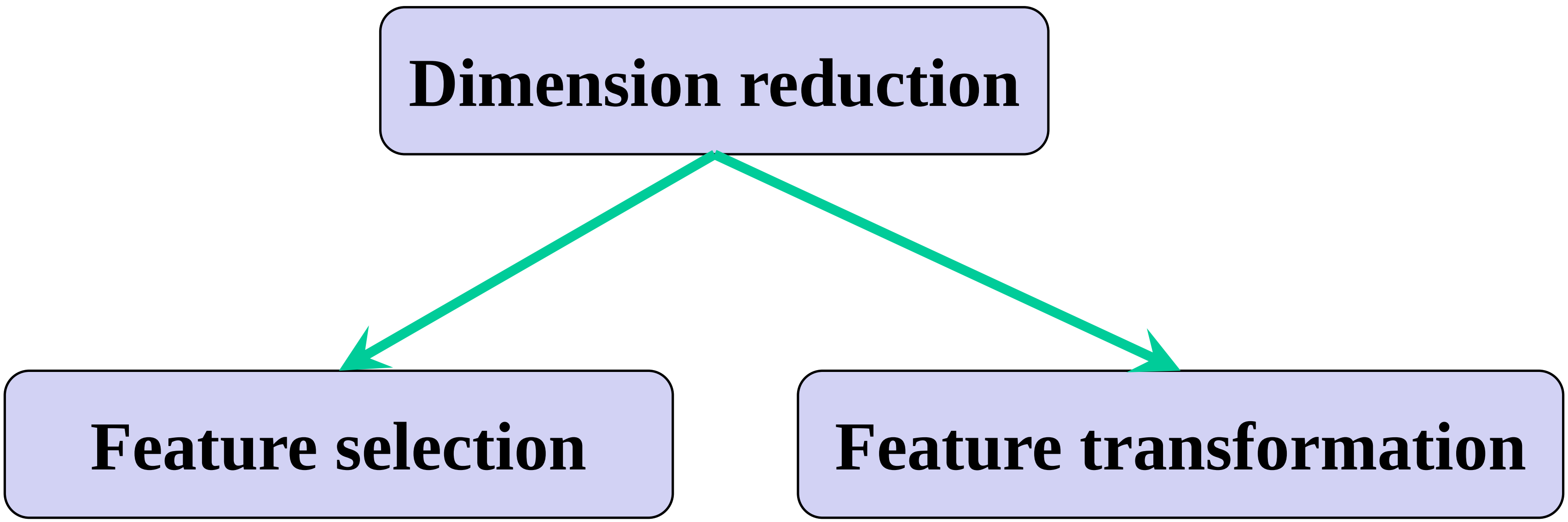
Salisbury UNIVERSITY

About this note

- The contents of this note refer to:
  - Book "Python Machine Learning"
  - Textbook "Data Science from Scratch"
  - Teaching materials at Department of Computer Science, William & Mary
  - Python tutorial: https://docs.python.org/3/tutorial/

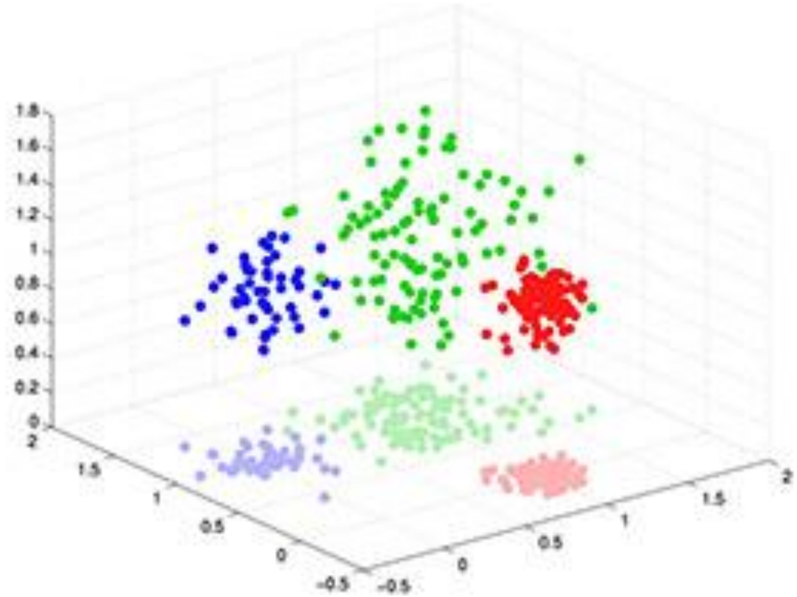
Dimension reduction

- The process of reducing the number of random variables (features) under consideration by obtaining a set of principal variables (features) [4]



Feature transformation

- Derive information from the feature set to construct a new feature subspace
- Example:



Feature transformation from 3D space to 2D space

- **Remember kernel SVM? We deal with nonlinear separable problems by projecting features onto a higher dimensional space via a mapping function.**
- **Q: Why do we need to reduce feature dimension here?**

Why do we need dimension reduction?

- Eliminate the noise features and redundant features
  - Q: what is the difference between noise features and redundant features?
- Avoid overfitting
- Reduce the complexity of the model
- Decrease the training time and (possibly) the testing time

Why not extract fewer features at the beginning?

- We do not have enough domain knowledge
  - Do not know which features are more representative
  - Do not know how many features are enough
  - Normally, we extract much more features than needed, which leads to "curse of dimensionality" problem
    - when the dimensionality increases, the volume of the space increases so fast that the available data become sparse [6] (the more features, the much more samples we need)

Cont’d

- Example:

1st feature set:

- **Use each pixel in the picture as one feature (1000 pixel * 1000 pixel = 1Million features)**

Salmon

Sea bass

2nd feature set [7]:

- **Length**
- **Lightness**
- **Width**
- **Position of mouth**

Domain knowledge: A sea bass is generally longer than a salmon [7]

- **What can we learn?**
  - **Domain knowledge is very important during we transform a real-world problem into a machine learning model!**

Where do we use dimension reduction?

Data collection from sensors (physical, virtual, etc.)

↓

Data preprocessing (missing value, outliers, etc.)

↓

Feature extraction (time & freq. domain, etc.)

↓

Dimension reduction (selection or transformation)

↓

Dataset generation (label, train & test dataset)

↓

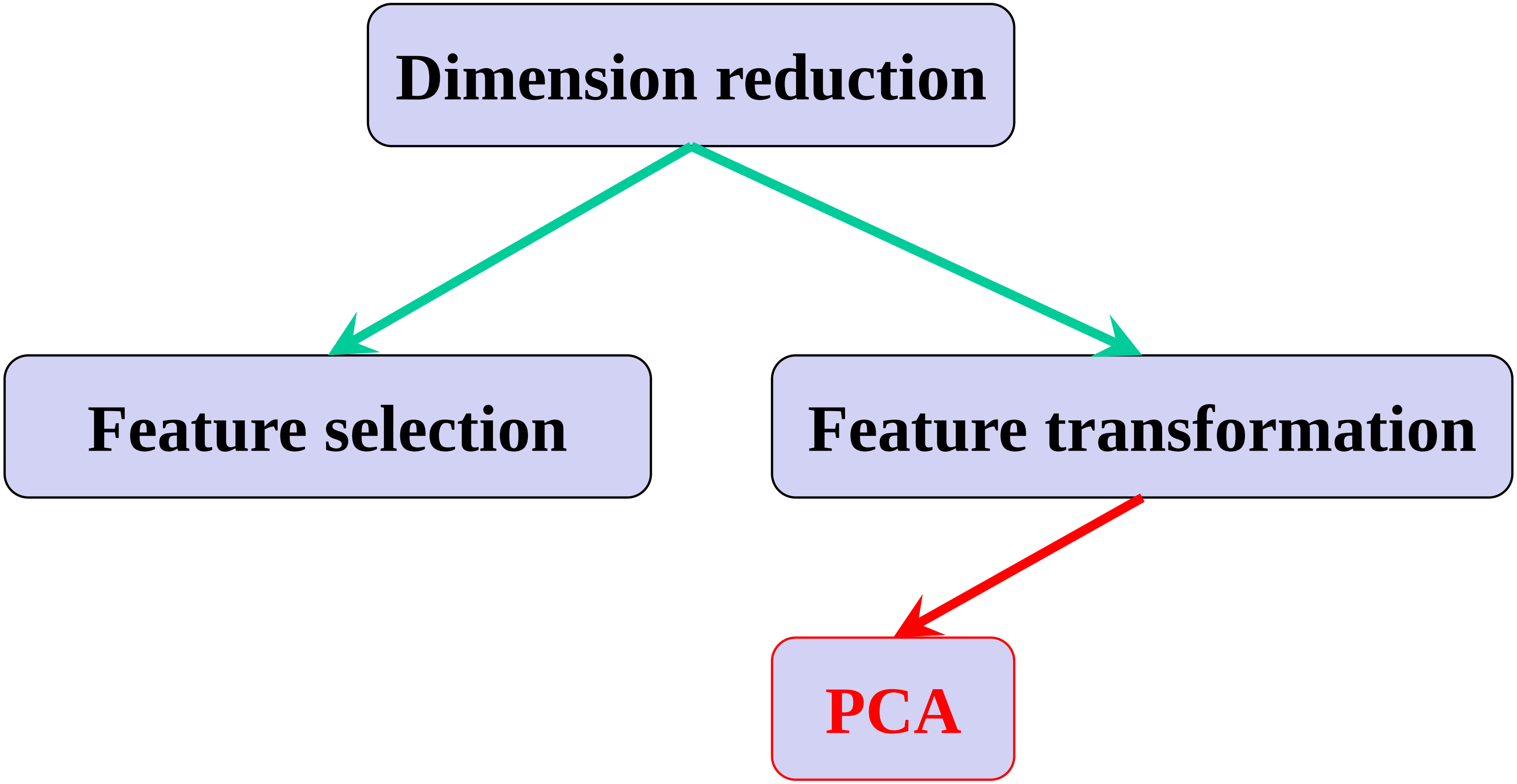
Model construction (classification, clustering, etc.)

↓

Model evaluation (accuracy, time & energy cost, etc.)

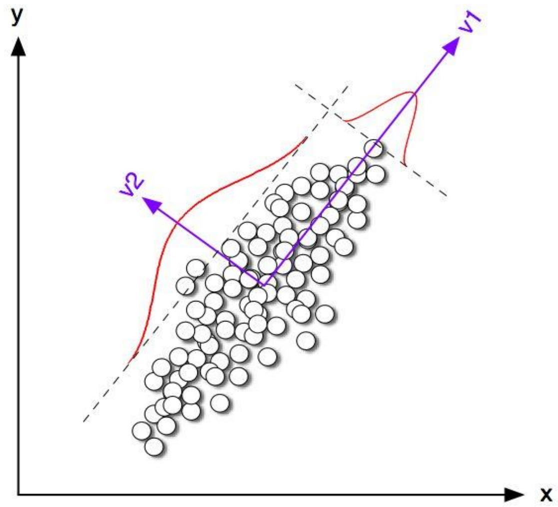
Principal component analysis (PCA)

- PCA is a feature transformation method



Basic rationale

- An example dataset in 2D space



Observation:

1. **Variable x and y are highly correlated**
2. **Correlation indicates information redundancy (Q: How to minimize it?)**
3. **We can rotate the coordinates so that:**
   - **The rotated axes are orthogonal**
   - **Data points are decentralized as much as possible**[8]

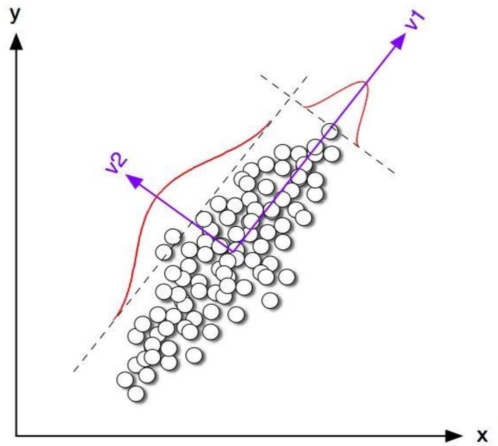
$$Cov(x, y) = 0 \; if \; x \; and \; y \; are \; orthogonal$$

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

Salisbury UNIVERSITY

Cont’d

- An example dataset in 2D space



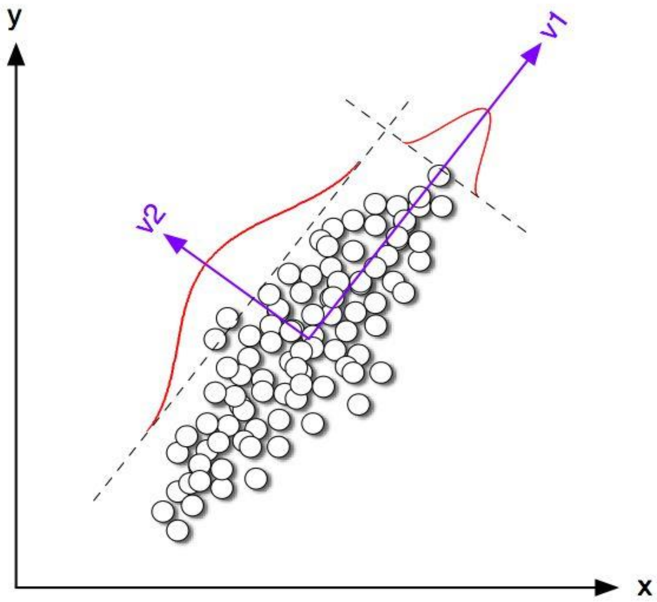
- **How to quantify the “decentralization” of data?**
  - **Map data on the axis, and maximize the variance**

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}$$

- **For rotated axes (principal components), which is more important?**
  - **The 1st PC have the largest variance (order in variance)**
  - **For unimportant axes, we can drop them (dimension reduction)**

Cont'd

- How to rotate the coordinates?



- **Rotated axis *V1* or *V2* could be formulated as a linear combination of x axis and y axis**

$$\begin{aligned} V1 &= w_1^{(1)} \cdot x + w_2^{(1)} \cdot y \\ V2 &= w_1^{(2)} \cdot x + w_2^{(2)} \cdot y \end{aligned} \Rightarrow (V1 \;\; V2) = (x \;\; y) \cdot \begin{pmatrix} w_1^{(1)} & w_1^{(2)} \\ w_2^{(1)} & w_2^{(2)} \end{pmatrix}$$

Problem statement

- Given a feature dataset $X = \{x_1, x_2, \cdots, x_n\}$, x_i is a d-dimensional feature vector
- We try to construct a $d \times k$-dimensional transformation matrix W that allows us to map a feature vector x_i onto a new k-dimensional feature subspace ($k \leq d$)
  - All axes are orthogonal
  - The variances of principal components (PC) are in descending order (1^{st} axis/PC has the biggest variance)

Mathematical expression [9]

- Suppose W is the projection matrix, the mapping of x_i in new (sub)space is $W^T x_i$
- The variance of all vectors in new space is $\sum_i W^T x_i x_i^T W$
- PCA will **maximize the variance**, i.e. $\max_W tr(W^T X X^T W)$, s.t. $W^T W = I$

$$\operatorname{tr}(A) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn}$$

Trace of an n-by-n square matrix A is defined to be the sum of the elements on the main diagonal -- Wiki

Mathematical expression [9]

- According to "Lagrange Multiplier", we have $XX^TW = \lambda W$, XX^T is the covariance matrix
- Compute the eigenvectors and eigenvalues of XX^T

A scalar λ is called an eigenvalue of the $n \times n$ matrix A if there is a nontrivial solution x of $Ax = \lambda x$. Such an x is called an eigenvector corresponding to the eigenvalue λ [3].

- Take the k largest eigenvalues, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$, and use corresponding k eigenvectors to construct $W = (w_1, \cdots, w_k)$

Six step algorithm

- PCA transformation steps [1]

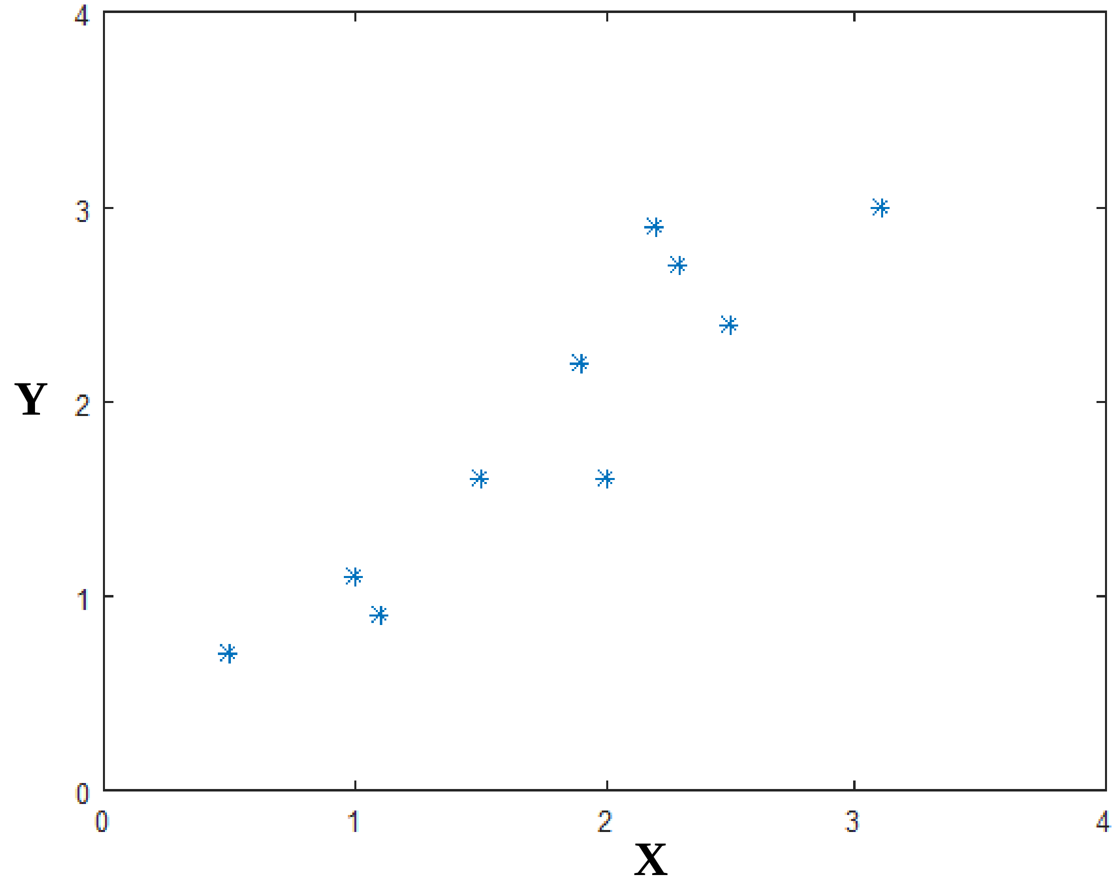
1. Standardize the d-dimensional dataset.
2. Construct the covariance matrix.
3. Decompose the covariance matrix into its eigenvectors and eigenvalues.
4. Select k eigenvectors that correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace ($k \leq d$).
5. Construct a projection matrix $\boldsymbol{W}$ from the "top" k eigenvectors.
6. Transform the d-dimensional input dataset $\boldsymbol{X}$ using the projection matrix $\boldsymbol{W}$ to obtain the new k-dimensional feature subspace.

Data preparation

- Raw data

Data =

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9



This example and the data refer to [2]

Six step algorithm

- Step 1: standardize the data
  - Subtract the mean VS z-score normalization

X	Y
0.6900	0.4900
-1.3100	-1.2100
0.3900	0.9900
0.0900	0.2900
1.2900	1.0900
0.4900	0.7900
0.1900	-0.3100
-0.8100	-0.8100
-0.3100	-0.3100
-0.7100	-1.0100

Subtract the mean

Mean = 0

X	Y
0.8787	0.5789
-1.6683	-1.4294
0.4967	1.1695
0.1146	0.3426
1.6429	1.2877
0.6240	0.9333
0.2420	-0.3662
-1.0316	-0.9569
-0.3948	-0.3662
-0.9042	-1.1932

z-score normalization

Mean = 0

Variance = 1

Six step algorithm

- Step 2: construct the covariance matrix
  - covariance matrix is a $d \times d$ symmetric matrix (d is the number of dimensions/features in the dataset)
    - *Cov(x,x) = var(x); Cov(y,y) = var(y); …*
    - *Cov(x,y)* = 0, *x* and *y* are independent
    - *Cov(x,y)* > 0, *x* and *y* move in same direction
    - *Cov(x,y)* < 0, *x* and *y* move in opposite direction

$$\begin{pmatrix} cov(x,x) & cov(x,y) \\ cov(y,x) & cov(y,y) \end{pmatrix}$$

Covariance matrix of 2D feature space

$$\begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

Covariance matrix of 3D feature space

Six step algorithm

- Step 2: construct the covariance matrix

$$CovMatrix = \begin{bmatrix} 0.6166 & 0.6154 \\ 0.6154 & 0.7166 \end{bmatrix}$$

Six step algorithm

- Step 3: obtain the eigenvalues and eigenvectors of the covariance matrix
  - The eigenvectors represent the principal components (the directions of maximum variance)
  - The corresponding eigenvalues define their magnitude

$$D = \begin{bmatrix} 0.0491 & 0.0000 \\ 0.0000 & 1.2840 \end{bmatrix} \qquad V = \begin{bmatrix} -0.7352 & 0.6779 \\ 0.6779 & 0.7352 \end{bmatrix}$$

Eigenvalues of covariance matrix

Eigenvectors of covariance matrix

Six step algorithm

- Step 4: sort the eigenvalues by decreasing order to rank the eigenvectors
  - 1.2840 > 0.0491, so after sorting, the eigenvectors are:

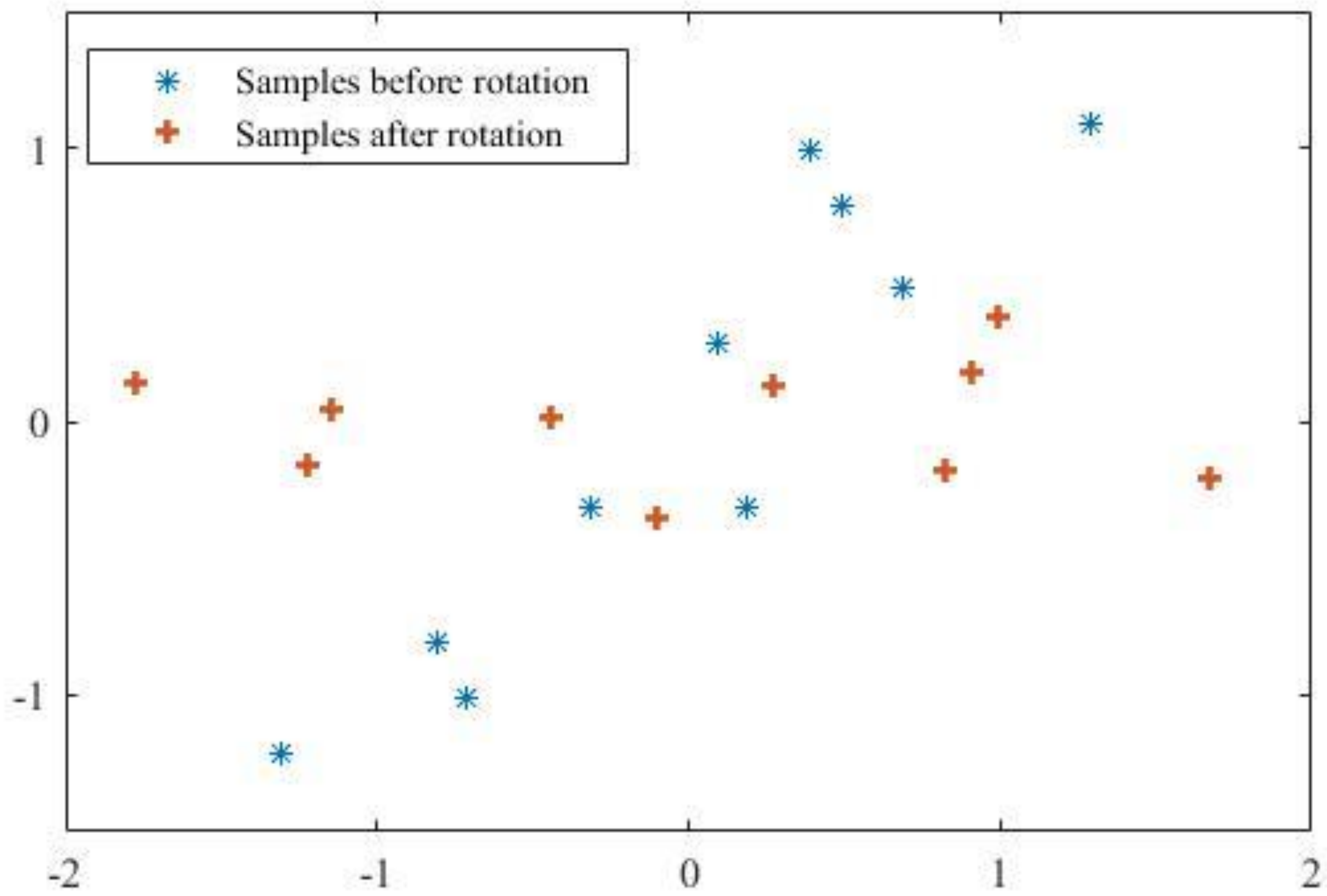
$$V = \begin{bmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{bmatrix}$$

Eigenvectors after sorting

V is the rotation matrix which rotates X-Y coordinates to V1-V2 coordinates

Six step algorithm

- The feature space before and after rotation

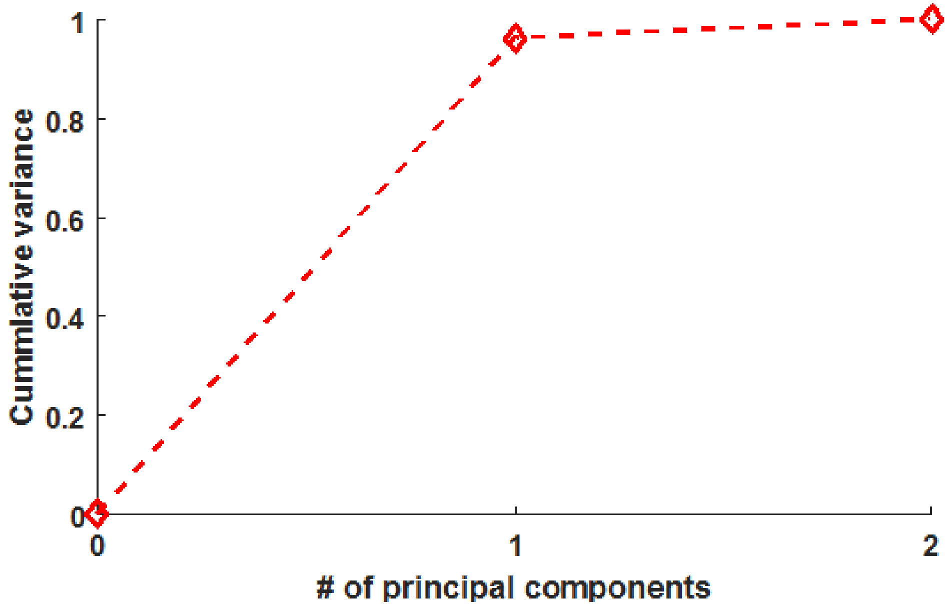


Six step algorithm

- Step 5: construct a projection matrix from the selected eigenvectors
  - Q: how many eigenvectors should we select?

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

(η is the ratio of variance loss)



The 1st PC contains more than 95% variances (information)

Six step algorithm

- Step 6: transform the data onto the lower-dimensional subspace
  - We only keep the 1st PC
  - Transform the data to 1D using projection matrix W

$A =$

X	Y
0.6900	0.4900
-1.3100	-1.2100
0.3900	0.9900
0.0900	0.2900
1.2900	1.0900
0.4900	0.7900
0.1900	-0.3100
-0.8100	-0.8100
-0.3100	-0.3100
-0.7100	-1.0100

$$W = \begin{bmatrix} 0.6779 \\ 0.7352 \end{bmatrix}$$

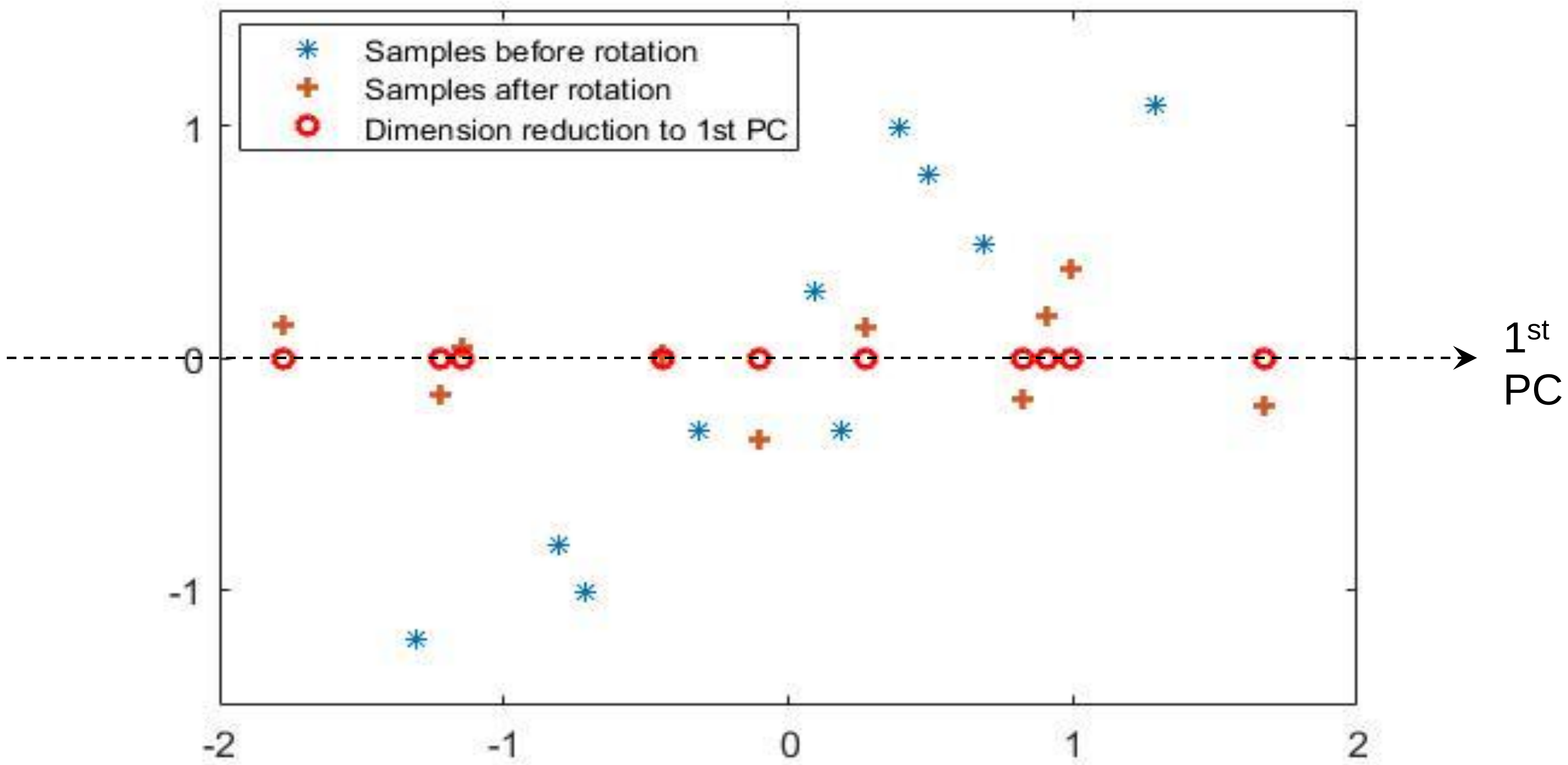
$A \times W =$

0.8280
-1.7776
0.9922
0.2742
1.6758
0.9129
-0.0991
-1.1446
-0.4380
-1.2238

Salisbury UNIVERSITY

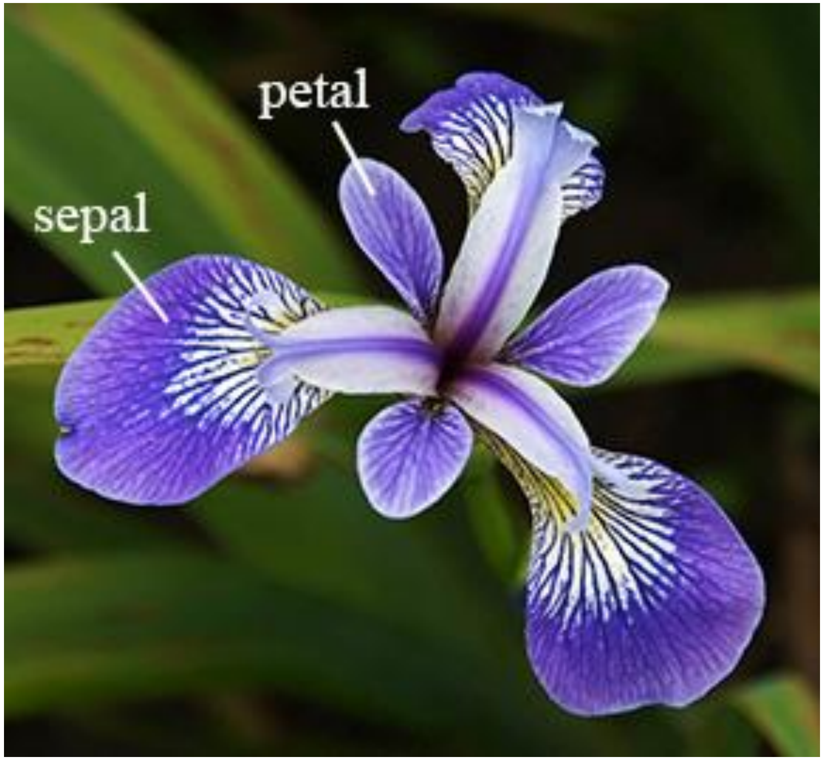
Six step algorithm

- Data points in 1st PC space



Example

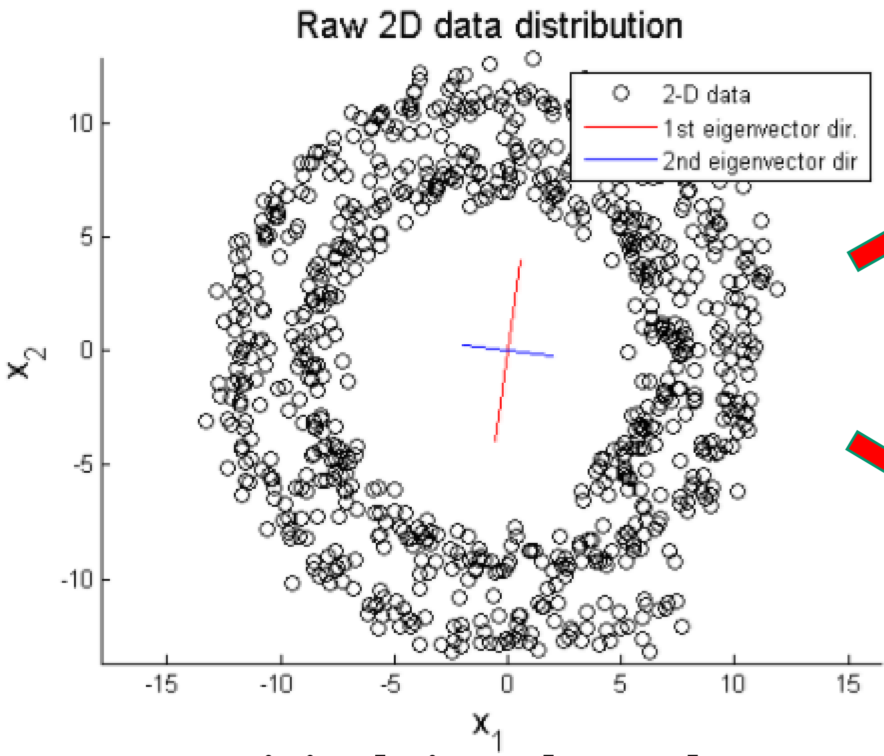
- PCA on “iris” dataset



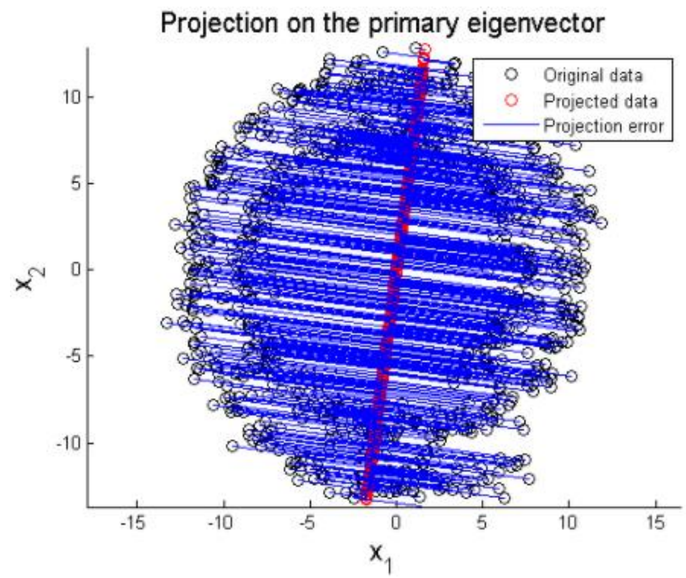
- 150 samples
- 4 features
  - sepal length
  - sepal width
  - petal length
  - petal width
- 3 classes
  - Iris Setosa
  - Iris Versicolour
  - Iris Virginica

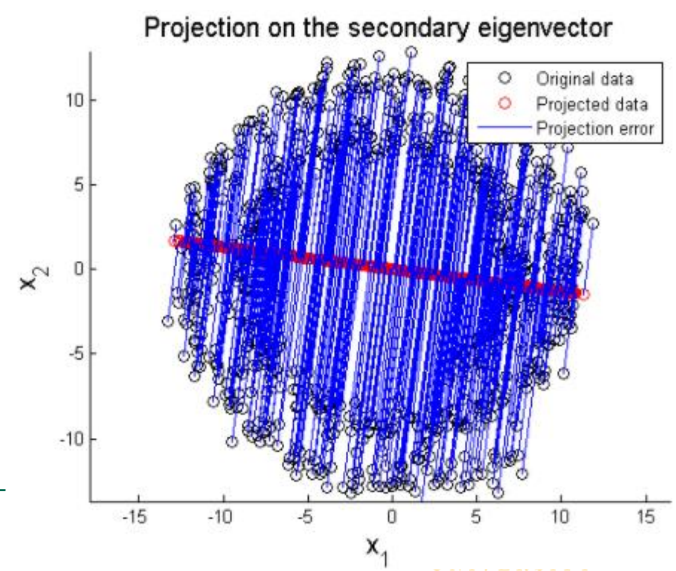
The picture refers to [10] and the dataset is from [11]

An example of failure [12]



Original circualr 2D data





Thanks

References

[1] Randal S. Olson, Python Machine Learning, 2015.

[2] http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

[3] http://www.math.harvard.edu/archive/20_spring_05/handouts/ch05_notes.pdf

[4] https://en.wikipedia.org/wiki/Dimensionality_reduction

[5] https://www.cs.waikato.ac.nz/ml/weka/

[6] https://en.wikipedia.org/wiki/Curse_of_dimensionality

[7] https://www.utdallas.edu/~bxt043000/Teaching/CS-4398/F2011/Lecture2.ppt

[8] http://matlabdatamining.blogspot.com/2010/02/principal-components-analysis.html

[9] http://blog.csdn.net/shizhixin/article/details/51181379

[10] http://terpconnect.umd.edu/~petersd/666/html/iris_pca.html

[11] http://archive.ics.uci.edu/ml/datasets/Iris

[12] https://www.projectrhea.org/rhea/index.php/PCA_Theory_Examples

[13] Isa Kemal Pakatci et al., Gene Set Analysis Using Principal Components, BCB '10.

[14] Guan-Chun Luh and Ching-Chou Hsieh, Face recognition using immune network based on principal component analysis. GEC '09.

[15] David T. Nguyen et al., A reconfigurable architecture for network intrusion detection using principal component analysis. FPGA '06.

[16] Xiaoxu Han. Nonnegative Principal Component Analysis for Cancer Molecular Pattern Discovery. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB). 2010.