Chair of Decision Sciences and Systems
TUM School of Computation, Information and Technology
Technical University of Munich

Business Analytics & Machine Learning
Tutorial sheet 9: Principle Component Analysis

Prof. Dr. Martin Bichler, Prof. Dr. Jalal Etesami
Julius Durmann, Markus Ewert, Johannes Knörr, Yutong Chao

Exercise T9.1 *Principle component analysis*

Given the following dataset $D = \{(-3,-1,-1),(0,-1,0),(-2,-1,2),(1,-1,3)\}$, compute its principal components by following the PCA algorithm introduced in class and generate the transformed data.
Each tuple of the set D represents an observation or row vector.

a) Calculate the zero-mean dataset X from the given dataset D. Note down the means.

b) Calculate the 3 x 3 covariance matrix Σ_X using the following formulas. What can you infer from it?
$var(x_j) = \frac{1}{N-1}\sum_{i=1}^{N}(x_{ij} - \overline{x}_j)^2$
$cov(x_j, x_k) = \frac{1}{N-1}\sum_{i=1}^{N}(x_{ij} - \overline{x}_j)(x_{ik} - \overline{x}_k)$
Reminder: Since the matrix X is centered, you can use the following formulas:
$var(x_j) = \frac{1}{N-1}\sum_{i=1}^{N} x_{ij}^2$
$cov(x_j, x_k) = \frac{1}{N-1}\sum_{i=1}^{N} x_{ij}x_{ik}$

c) Find the eigenvalues for the covariance matrix by solving the equation: $|\Sigma_X - \lambda I_3| = 0$. You can use the *Laplace expansion* to calculate the determinant.

d) Find the corresponding eigenvectors and order them by significance. How is the variance distributed among them?
Hint: Solving the equation $(\Sigma_X - \lambda I_3)v = 0$ gives you the corresponding eigenvectors.

e) Compute a one-dimensional PCA projection of the dataset.

f) Compute a two-dimensional PCA projection of the dataset.
Hint for e.) and f.): The general formula for projections is: $Z = X\Phi$

Exercise T9.2 *PCA Reconstruction*

Making use of the PCA projections computed in Exercise 10.1, restore the original dataset using the formula: $D \approx Z\Phi^T +$ means

a) Reverse the one-dimensional PCA projection to restore the original data. How would the data look when plotted into the original coordinate system?

b) What result do you expect when reconstructing the original data from the two-dimensional PCA projection? What is the information loss?

Exercise T9.3 *PCA Understanding*

You are given the following dataset:

$$D = \{(1, -4), (-2, 2), (0, -2), (-1, -1), (2, -3)\}$$

a) Apply principal component analysis and determine the two principal component vectors

b) Draw the data points, the principal components, and the data set recovered from the 1D projection in a single plot.

c) Now another point (a, b) is added to the dataset. As a results, the principal components do not change. Determine a possible point (a, b).

d) We now adjust the original dataset in the following three ways

1) Multiply both coordinates of the first point by some factor $k \in \mathbb{R}$.
2) Multiply all coordinates of all points by the same factor $k \in \mathbb{R}$.
3) Flip all coordinates of all points, e.g. $(1, -4) \rightarrow (-4, 1)$.

In each case, will the eigenvalues and/or principal components change? Discuss your reasoning.

Exercise T9.4 *PCA for image compression*

The goal of this exercise is to use Principal Component Analysis for image compression in Python.

a) Load an image of the famous painting "The Starry Night" by Vincent Van Gogh and store red, green, and blue values as color matrices. You can access the painting here. Use the provided template *image_compression_PCA_template.py* to load the painting, store color values, and to plot it.

b) Apply a PCA to each color matrix to identify principal color vectors. Choose $n = 5$ as the number of components. Plot the resulting compressed image. The following code snippet may be helpful:

```
for i in range(image.shape[2]):
  # get channel image, normalize data to [0, 1] before applying PCA
  channel_data = img[:, :, i] / 255.0

  # perform pca
  pca = PCA(n_components=number_of_components)
  pca.fit(channel_data)
  compressed_channel_data = pca.inverse_transform(pca.transform(channel_data))
```

c) Apply PCA for $n = \{1, 2, 5, 10, 20\}$ (if your computer permits, you can increase n even further). Plot and save the compressed image in each iteration. Observe the size of the image files. Looking at the images, at what point do you clearly identify the painting?

d) Determine a reasonable number of clusters using the "elbow criterion". For this purpose, create a scree plot that plots the explained variance of each component against the number of components, e.g. for $n \in [1, 10]$. Does the elbow point correspond to your visual impression in d)?

e) Compare the image compression using PCA with the image compression using clustering from last week.