Python 3 Laboratories
Principal Components Analysis

Francesco Della Santa

December 11, 2019

Abstract

In this lesson, we will learn to implement the Principal Component Analysis.

1 Introduction

Open the Python project created during previous laboratories and the corresponding virtual environment (VE). Then do the following operations:

- download from the web page of the course the python module *mypca.py* and copy it in the project folder;
- download from the web page of the course the python module *mypca_examples.py* and copy it in the project folder;
- download from the web page of the course the folder *mypca_datasets* and copy it in the project folder;

2 Exercises

2.1 Principal Component Analysis (PCA) Implementation

Before starting with the exercises, you can read again the main characteristics of the PCA and learn the notation of this document in section 3.

Exercise 1. In the module *mypca.py*, complete the code of the following two functions for the implementation of the PCA (*without* variance normalization):

my_pca: a function that takes as input a matrix of N samples $X \in \mathbb{R}^{N\times n}$ and returns as outputs:

- the sample covariance matrix $S \in \mathbb{R}^{n\times n}$;
- the matrix $W \in \mathbb{R}^{N\times n}$ (X written w.r.t. basis $\mathcal{U}$);
- the matrix $U = [\boldsymbol{u}_1\,, \dots, \boldsymbol{u}_n] \in \mathbb{R}^{n\times n}$ with columns given by the principal components;
- the row-vector $\boldsymbol{\lambda}^\top = [\lambda_1\,, \dots, \lambda_n]$;
- the row-vector $\boldsymbol{\mu}^\top$.

pc_approx: a function that takes as inputs:

- the matrix $W \in \mathbb{R}^{N\times n}$ (samples with respect to $\mathcal{U}$);
- the matrix U;
- $m \in \mathbb{N}$;

- the row-vector $\boldsymbol{\mu}^\top$.

And returns as output the matrix $\tilde{X} \in \mathbb{R}^{N \times n}$ that approximates X with respect to the first m principal components (see (8)).

Exercise 2. In the module *mypca.py*, create the following two functions for the implementation of the PCA *with* variance normalization:

my_pca_varnorm: a function that takes as input a matrix of N samples $X \in \mathbb{R}^{N \times n}$ and returns as outputs:

- same outputs of *my_pca*;
- the row-vector $\boldsymbol{\sigma}^\top$.

pc_approx_varnorm: a function that takes as inputs:

- same inputs of *pc_approx*;
- the row-vector $\boldsymbol{\sigma}^\top$.

And returns as output the matrix $\tilde{X} \in \mathbb{R}^{N \times n}$ that approximates X with respect to the first m principal components (update (8) considering also $\boldsymbol{\sigma}$).

Exercise 3. Run the script *mypca_examples.py* in your python shell. In this script the PCA, without and with variance normalization, is applied to a dataset of *iris* flowers and a dataset of *wines*, respectively.

Iris dataset: dataset of 150 iris flowers belonging to 3 species. Each sample is described by 4 features: *Sepal Length* (Cm), *Sepal Width* (Cm), *Petal Length* (Cm), *Petal Width* (Cm).

Wines dataset: dataset of 178 wines belonging to 3 categories. Each sample is described by 13 features.

What can you say about the results?

3 PCA (Recap)

Let $X \in \mathbb{R}^{N\times n}$ be a matrix of N samples $\boldsymbol{x}_i \in \mathbb{R}^n$ (assuming $N > n$) such that

$$X = \begin{bmatrix} \boldsymbol{x}_1^\top \\ \vdots \\ \boldsymbol{x}_N^\top \end{bmatrix} = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{Nn} \end{bmatrix} \tag{1}$$

and let $\boldsymbol{\mu} \in \mathbb{R}^n$ be the mean vector of all the N individuals in X, i.e.:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}\,, \quad \forall\, j = 1, \dots, n\,. \tag{2}$$

Attention: usually in the theory the samples $\boldsymbol{x}_i$ are stored in X as column vectors; however, most of the dataset used in the practice store the samples as row vectors. Therefore we decided to use this notation in this document and in the exercises.

The *principal components* of samples X are the *orthonormal* eigenvectors $\boldsymbol{u}_1, \dots .\boldsymbol{u}_n \in \mathbb{R}^n$ (with corresponding eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$) of the *sample covariance matrix*

$$S = \frac{1}{N-1} B^\top B \in \mathbb{R}^{n\times n}\,, \tag{3}$$

where B is the *re-centered matrix* of samples X, i.e.:

$$B = X - \begin{bmatrix} \boldsymbol{\mu}^\top \\ \vdots \\ \boldsymbol{\mu}^\top \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_1^\top - \boldsymbol{\mu}^\top \\ \vdots \\ \boldsymbol{x}_N^\top - \boldsymbol{\mu}^\top \end{bmatrix} \in \mathbb{R}^{n\times n}\,. \tag{4}$$

Changing coordinates (basis): let $U = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_n] \in \mathbb{R}^{n\times n}$ the matrix with columns given by the principal components and let $\mathcal{U} = \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_n\}$ be the orthonormal base of $\mathbb{R}^n$ given by the principal components. Then, for each re-centered sample $\boldsymbol{b} \in \mathbb{R}^n$ (column vector), its representation $\boldsymbol{w} = [w_1, \dots, w_n]^\top$ with respect to the base $\mathcal{U}$ is such that

$$\boldsymbol{b} = w_1\,\boldsymbol{u}_1 + \cdots + w_n\,\boldsymbol{u}_n = \sum_{i=1}^{n} w_i\,\boldsymbol{u}_i \in \mathbb{R}^n\,; \tag{5}$$

Then

$$\boldsymbol{w} = U^\top(\boldsymbol{x} - \boldsymbol{\mu}) \quad (\text{ i.e. } \boldsymbol{w}^\top = (\boldsymbol{x}^\top - \boldsymbol{\mu}^\top)\,U\ ) \tag{6}$$

and

$$\boldsymbol{x} = U\,\boldsymbol{w} + \boldsymbol{\mu} \quad (\text{ i.e. } \boldsymbol{x}^\top = \boldsymbol{w}^\top U^\top + \boldsymbol{\mu}^\top\ )\,. \tag{7}$$

Approximation with Principal Components: the approximation $\tilde{\boldsymbol{x}} \in \mathbb{R}^n$ of a sample $\boldsymbol{x} \in \mathbb{R}^n$ with respect to the first $m \leq n$ principal components is given by

$$\tilde{\boldsymbol{x}} \approx U|_m\,\boldsymbol{w}|_m + \boldsymbol{\mu}\,, \tag{8}$$

where $U|_m = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_m] \in \mathbb{R}^{n\times m}$ and $\boldsymbol{w}|_m = [w_1, \dots, w_m]^\top \in \mathbb{R}^m$.

Variance and PCA: The *total variance* of X is always fixed (also when samples are represented with respect to basis $\mathcal{U}$) and it is

$$\Lambda = \operatorname{tr}(S) = \sum_{i=1}^{n} \lambda_i\,. \tag{9}$$

Let $W \in \mathbb{R}^{N\times n}$ be the representation of samples in X with respect to $\mathcal{U}$. Then, the variance with respect to the j-th column of W (i.e. with respect to the p.c. $\boldsymbol{u}_j$) is λ_j and the ratio

$$\frac{\lambda_j}{\Lambda} \tag{10}$$

explains in percentages how much $\boldsymbol{u}_j$ "explains" the total variance Λ.

3.1 Normalization of the Variance

In many applications, samples $\boldsymbol{x} \in \mathbb{R}^n$ can be characterized by *features* (elements of the vector) with order of magnitude very different. In this situations, is suggested to use in (3) not the matrix B but the matrix $\widehat{B}$ with normalized variance, i.e.:

$$\widehat{B} = \begin{bmatrix} (\boldsymbol{x}_1^\top - \boldsymbol{\mu}^\top) \div \boldsymbol{\sigma}^\top \\ \vdots \\ (\boldsymbol{x}_N^\top - \boldsymbol{\mu}^\top) \div \boldsymbol{\sigma}^\top \end{bmatrix}, \tag{11}$$

where $\div$ is the element-wise division and $\boldsymbol{\sigma}$ is the sample standard deviation vector, i.e.:

$$\sigma_j = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_{ij}-\mu_j)^2}\,. \tag{12}$$

The usage of $\widehat{B}$ is important because it modifies all the *features* such that they have the same "unit measure".

Obviously in this case it holds

$$\boldsymbol{w} = U^\top\ ((\boldsymbol{x}-\boldsymbol{\mu}) \div \boldsymbol{\sigma}) \quad (\text{ i.e. } \boldsymbol{w}^\top = ((\boldsymbol{x}-\boldsymbol{\mu}) \div \boldsymbol{\sigma})^\top\ U\ ) \tag{13}$$

and

$$\boldsymbol{x} = (U\,\boldsymbol{w}) \odot \boldsymbol{\sigma} + \boldsymbol{\mu} \quad (\text{ i.e. } \boldsymbol{x}^\top = \left(\boldsymbol{w}^\top\, U^\top\right) \odot \boldsymbol{\sigma}^\top + \boldsymbol{\mu}^\top\ )\,, \tag{14}$$

where $\odot$ is the element-wise product.