The mathematics of Principal Component Analysis

Matteo Gardini*

October 18, 2022

Abstract

This document summarize the theoretical part of the principal components analysis (PCA). We recall some useful facts from linear algebra and hence we derive the PCA algorithm formally. Finally we apply it to a data-set concerning breast cancers and we perform some dimensional reduction and classification with PCA using python.

1 Preliminaries

In this section we recall some trivial concepts regarding linear algebra. We start with the elementary notion of dot product in $\mathbb{R}^n$ and hence we talk about orthogonal matrices since they play an important role in the PCA algorithm.

Notation 1. If we consider a vector $a \in \mathbb{R}^n$ it is usual to denote its components by $(a_1, \dots, b_n)$. Nevertheless, if we have a bunch of vectors in $\mathbb{R}^n$ $\boldsymbol{X}_1, \dots, \boldsymbol{X}_p$ the components of $\boldsymbol{X}_j$ are denoted by: $\boldsymbol{X}_j = \left(X_j^1, \dots, X_j^n\right)^T$. The notation will be clear from the context.

1.1 Dot product

Given two vectors $a, b \in \mathbb{R}^n$ there are two different equivalent definitions of scalar product. From the algebraic point of view the dot product is simply the sum of the products of the element of the array. From a geometric point of view the dot product $a \cdot b$, represent the projection of a along the direction given by the vector b. We have the following proposition.

Proposition 1.1. *Consider two vectors $a, b \in \mathbb{R}^n$ of the form $a = (a_1, \dots, a_n)$ and $b = (b_1, \dots, b_n)$. We define the dot product of a and b as:*

$$a \cdot b = \sum_{i=1}^{n} a_i b_i.$$

Then

$$\sum_{i=1}^{n} a_i b_i = \|a\| b\| \cos\theta,$$

where θ is the angle between a and b and $\|\cdot\|$ is the euclidean norm.

*Department of Mathematics, University of Genoa, Via Dodecaneso 16146, Genoa, Italy, email gardini@dima.unige.it

Proof. Take $e_i = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$, $i = 1, \ldots, n$ an orthogonal base for $\mathbb{R}^n$, we have that:

$$a = \sum_{i=1}^{n} a_i e_i, \qquad b = \sum_{i=1}^{n} b_i e_i.$$

Then:

$$a \cdot b = a \cdot \sum_{i=1}^{n} b_i e_i = \sum_{i=1}^{n} b_i a \cdot e_i = \sum_{i=1}^{n} b_i \|a\| \cos\theta_i = \sum_{i=1}^{n} b_i a_i.$$

■

1.2 Orthogonal matrices

Orthogonal matrices play a very important role in the PCA. The key properties of orthogonal matrices are resumed here. Essentially, they are linear transformations from $\mathbb{R}^n$ to $\mathbb{R}^n$ preserving angles and lengths. This kind of transformation can be interpreted as a rotation of the coordinate system.

Definition 1.1. *$C \in \mathbb{R}^{n\times n}$ is said to be orthogonal if $C^T = C^{-1}$ and if we have:*

$$c_i \cdot c_j = 0, \quad \text{if } i \neq j,$$
$$c_i \cdot c_j = 1, \quad \text{if } i = j.$$

where $\cdot$ denotes the usual scalar product in $\mathbb{R}^n$ and c_i is the i-th column of C.

Hence we are requiring that the columns of C are an orthogonal base for $\mathbb{R}^n$. Remember that, given two vector $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ in $\mathbb{R}^n$, the scalar product is defined as:

$$a \cdot b = \sum_{i=1}^{n} a_i b_i.$$

Remember that the multiplication of a vector by a matrices affect usually the length of the vector and also the angle between them. This is not the case for orthogonal matrices, as the following lemma states.

Lemma 1.2. *Orthogonal transformation preserves lengths and angles between vectors.*

Proof. Let be $x \in \mathbb{R}^n$ and $C \in \mathbb{R}^{n\times n}$ be an orthogonal matrix. We have to show that:

$$\|Cx\| = \|x\|,$$

where $\|\cdot\|$ denotes the usual euclidean norm in $\mathbb{R}^n$.

$$\|Cx\|^2 = Cx \cdot Cx = (Cx)^T Cx = x^T C^T C x = x^T x = \|x\|^2.$$

■

In order to show that also angles are preserved we consider the angle θ between vectors $v, u \in \mathbb{R}^d$ and the angle θ_c between vectors Cv, Cu, where C is an orthogonal matrix. We recall that $v \cdot w = \|v\|\|u\| \cos\theta$:

$$\cos\theta_c = \frac{Cv \cdot Cw}{\|Cv\|\|Cu\|} = \frac{(Cv)^T (Cw)}{\|Cv\|\|Cu\|} = \frac{v^T C^T C w}{\|Cv\|\|Cu\|} = \frac{v^T w}{\|v\|\|u\|} = \frac{v \cdot w}{\|v\|\|u\|} = \cos\theta$$

2 Preparation for PCA

2.1 Matrices manipulations for PCA

The following results will be very useful to derive some results regarding the PCA analysis.

Consider a random vector $\boldsymbol{X} \in \mathbb{R}^d$ such that:

$$\boldsymbol{X} = \begin{bmatrix} X^1 \\ \vdots \\ X^d \end{bmatrix}$$

where $X^i,\ i = \ldots, d$ are random variables on some given probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Consider n independent copies of $\boldsymbol{X}$, $\boldsymbol{X}_i$ and define $\mathcal{X} \in \mathbb{R}^{n \times d}$ as:

$$\mathcal{X} = \begin{bmatrix} \boldsymbol{X}_1^T \\ \vdots \\ \boldsymbol{X}_n^T \end{bmatrix} = \begin{bmatrix} X_1^1 & \ldots & X_1^d \\ \vdots & \vdots & \vdots \\ X_n^1 & \ldots & X_n^d. \end{bmatrix}$$

This is a matrix where we have realizations on the rows and features on the columns. We also define the expected value of the random vector $\boldsymbol{X}$, $\mathbb{E}[\boldsymbol{X}] \in \mathbb{R}^d$ as:

$$\mathbb{E}[\boldsymbol{X}] = \begin{bmatrix} \mathbb{E}\left[X^1\right] \\ \vdots \\ \mathbb{E}\left[X^d\right] \end{bmatrix}$$

and the covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$ with entries $\sigma_{ij} = cov(X^i, X^j)$.

Lemma 2.1.

$$\Sigma = \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right] - \mathbb{E}[\boldsymbol{X}]\,\mathbb{E}[\boldsymbol{X}]^T, \tag{1}$$

and

$$\Sigma = \mathbb{E}\left[(\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}])\,(\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}])^T\right]. \tag{2}$$

Proof. First we prove the first equality. By definition we have that:

$$(\Sigma)_{ij} = \sigma_{ij} = \mathbb{E}\left[X^i X^j\right] - E\left[X^i\right] E\left[X^j\right].$$

On the other hand we have that:

$$\left(\boldsymbol{X}\boldsymbol{X}^T\right)_{ij} = \left(\begin{bmatrix} X^1 \\ \vdots \\ X^d \end{bmatrix} \begin{bmatrix} X^1, \ldots, X^d \end{bmatrix} \right) = X^i X^j,$$

and hence:

$$\left(\mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right]\right)_{ij} = \mathbb{E}\left[X^i X^j\right],$$
$$\left(\mathbb{E}[\boldsymbol{X}]\,\mathbb{E}\left[\boldsymbol{X}^T\right]\right)_{ij} = \mathbb{E}\left[X^i\right]\mathbb{E}\left[X^j\right].$$

Finally we have that:

$$\left(\mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right]\right)_{ij} - \left(\mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}^T\right]\right)_{ij} = (\Sigma)_{ij},$$

and hence (1) is proved.

In order to prove (2) we just need a computation:

$$\begin{aligned}\mathbb{E}\left[\left(\boldsymbol{X} - \mathbb{E}\left[\boldsymbol{X}\right]\right)\left(\boldsymbol{X} - \mathbb{E}\left[\boldsymbol{X}\right]\right)^T\right] &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T - \boldsymbol{X}\mathbb{E}\left[\boldsymbol{X}\right]^T - \mathbb{E}\left[\boldsymbol{X}\right]\boldsymbol{X}^T + \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T\right] \\ &= \mathbb{E}\left[\boldsymbol{X}BX^T\right] - \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T - \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}^T\right] + \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T \\ &= \mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right] - \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T = \Sigma,\end{aligned}$$

where in the last equality we have used Equation (1). ■

Lemma 2.2. *Let be $\boldsymbol{X} \in \mathbb{R}^d$ a random vector. Then $\mathbb{E}\left[\boldsymbol{X}^T\right] = \mathbb{E}\left[\boldsymbol{X}\right]^T$.*

Proof. By definition we have that:

$$\mathbb{E}\left[\boldsymbol{X}\right]^T = \left[\mathbb{E}\left[X^1\right], \ldots, \mathbb{E}\left[X^n\right]\right].$$

Moreover:

$$\mathbb{E}\left[X^T\right] = \mathbb{E}\left[\left[X^1, \ldots, X^d\right]\right] = \left[\mathbb{E}\left[X^1\right], \ldots, \mathbb{E}\left[X^n\right]\right].$$

■

2.2 Estimators

Consider $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ realizations of the random vector $\boldsymbol{X} \in \mathbb{R}^n$. We define the empirical mean $\bar{\boldsymbol{X}} \in \mathbb{R}^{d\times d}$ as:

$$\bar{\boldsymbol{X}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i, \tag{3}$$

and the sample covariance $S \in \mathbb{R}^d$ as:

$$S = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i\boldsymbol{X}_i^T - \bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^T = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{X}_i - \bar{\boldsymbol{X}}\right)\left(\boldsymbol{X}_i - \bar{\boldsymbol{X}}\right)^T.$$

Remark 1. Observe that:

$$n\bar{\boldsymbol{X}} = \sum_{i=1}^{n} X_i = \mathcal{X}^T \mathbb{1}_n.$$

We have the following lemma.

Lemma 2.3. *Let $A_i \in \mathbb{R}^{d\times n}$ and $B_i \in \mathbb{R}^{n\times d}$. Define:*

$$A = \sum_{i=1}^{n} A_i, \quad B = \sum_{j=1}^{n} B_j.$$

Then:

$$AB = \sum_{i=1}^{n}\sum_{j=1}^{n} A_iB_j.$$

Proof.

$$AB = \left(\sum_{i=1}^{n} A_i\right)\left(\sum_{j=1}^{n} B_j\right) = (A_1 + \cdots + A_n)(B_1 + \cdots + B_n)$$

$$= A_1(B_1 + \cdots + B_n) + \cdots + A_n(B_1 + \cdots + B_n) = \sum_{i=1}^{n} A_i(B_1 + \cdots + B_n)$$

$$= \sum_{i=1}^{n} A_i \left(\sum_{j=1}^{n} B_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_i B_j.$$

■

Sometimes it is useful to define $\bar{\boldsymbol{X}}$ and Σ in terms of $\mathcal{X}$. The following proposition provides such expression of $\bar{\boldsymbol{X}}$ and Σ.

Proposition 2.4. *If we define:* $\mathbf{1}_n = [1, \ldots, 1]^T \in \mathbb{R}^n$ *we have that:*

$$\bar{\boldsymbol{X}} = \frac{1}{n}\mathcal{X}^T \mathbf{1}_n.$$

Furthermore, if we define $I_n \in \mathbb{R}^{n\times n}$ the identity matrix we have that:

$$S = \frac{1}{n}\mathcal{X}^T\left[I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right]\mathcal{X}$$

Proof. The first result is easy to prove. Renenber that:

$$n\bar{\boldsymbol{X}} = \sum_{i=1}^{n} X_i,$$

and hence we have to prove that $\mathcal{X}^T 1_n = \sum_{i=1}^{n} X_i$.

$$\mathcal{X}^T\mathbf{1}_n = \begin{bmatrix} X_1^1 & \ldots & X_1^d \\ \vdots & \vdots & \vdots \\ X_n^1 & \ldots & X_n^d. \end{bmatrix}\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} X_i^1 \\ \vdots \\ \sum_{i=1}^{n} X_i^d \end{bmatrix} = \sum_{i=1}^{n}\begin{bmatrix} X_i^1 \\ \vdots \\ X_i^d \end{bmatrix} = \sum_{i=1}^{d} X_i = n\bar{\boldsymbol{X}}.$$

In order to prove the second we firstly compute $\mathcal{X}^T\mathcal{X}$. Write $\mathcal{X}^T$ as:

$$\mathcal{X}^T = \begin{bmatrix} X_1^1 & \ldots & X_n^1 \\ \vdots & \vdots & \vdots \\ X_1^d & \ldots & X_n^d \end{bmatrix} = M_1 + \cdots + M_n,$$

where M_j has the following form:

$$M_j = \begin{bmatrix} 0 & \ldots & 0 & X_1^j & 0 & \ldots & 0 \\ 0 & \ldots & 0 & X_2^j & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & 0 & X_n^j & 0 & \ldots & 0 \end{bmatrix}.$$

By using Lemma 2.3 we get that:

$$\mathcal{X}^T\mathcal{X} = \left(\sum_{i=1}^{n} M_i\right)\left(\sum_{i=1}^{n} M_i\right)^T = \left(\sum_{i=1}^{n} M_i\right)\left(\sum_{i=1}^{n} M_i^T\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} M_i M_j^T$$

Because of the structure of the matrices M_j, $j = 1, \ldots, n$ we have that $M_i M_j^T = \mathbf{0}$ for $i \neq j$, $M_i M_i^T = \boldsymbol{X}_i \boldsymbol{X}_i^T$ and hence:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} M_i M_j^T = \sum_{i=1}^{n} M_i M_i^T = \sum_{i=1}^{n} \boldsymbol{X}_i \boldsymbol{X}_i^T.$$

Recalling that $\bar{\boldsymbol{X}} = \frac{1}{n}\mathcal{X}^T\mathbf{1}_n$ we have that:

$$\begin{aligned} S &= \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{X}_i \boldsymbol{X}_i^T - \bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^T = \frac{1}{n}\mathcal{X}^T\mathcal{X}^T - \frac{1}{n}\mathcal{X}^T\mathbf{1}_n\left(\frac{1}{n}\mathcal{X}^T\mathbf{1}_n\right)^T \\ &= \frac{1}{n}\mathcal{X}^T\mathcal{X} - \frac{1}{n^2}\mathcal{X}^T\mathbf{1}_n\mathbf{1}_n^T\mathcal{X} = \frac{1}{n}\mathcal{X}I_n\mathcal{X} - \frac{1}{n^2}\mathcal{X}^T\mathbf{1}_n\mathbf{1}_n^T\mathcal{X} = \frac{1}{n}\mathcal{X}^T\left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right)\mathcal{X}, \end{aligned}$$

where I_n is the identity matrix of order n.

■

Lemma 2.5. *The operator $H : \mathbb{R}^n \to \mathbb{R}^n$ defined as $H = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ is an orthogonal projection, namely $H^2 = H$.*

Proof. First we observe that $\mathbf{1}_n^T\mathbf{1}_n = n$.

$$\begin{aligned} H^2 &= \left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right)\left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right) \\ &= I_n^2 - \frac{1}{n}I_n\mathbf{1}_n\mathbf{1}_n^T I_n + \frac{1}{n^2}\mathbf{1}_n\mathbf{1}_n^T\mathbf{1}_n\mathbf{1}_n^T \\ &= I_n - \frac{1}{n}I_n\mathbf{1}_n\mathbf{1}_n^T - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T I_n + \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T I_n = I_n - \frac{1}{n}I_n\mathbf{1}_n\mathbf{1}_n^T = H. \end{aligned}$$

■

What does H do the a vector $v \in R^n$?

$$Hv = \left(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right)v = v - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T v = v - \not\Vdash_n \bar{v},$$

where in the last equation we used:

$$\mathbf{1}_n^T v = [1, \ldots, 1]\, v = \sum_{i=1}^{n} v_i = \bar{v}n.$$

This means that the operator H takes a vector $v \in R^n$ and removes its mean from it.

Lemma 2.6. *Let $v \in \mathbb{R}^n$ and let H be defined as above. Then*

$$\bar{H}v = 0.$$

Proof.

$$\bar{H}v = \frac{1}{n}\mathbf{1}_n^T\left(v - \bar{v}\mathbf{1}\right) = \frac{1}{n}\mathbf{1}_n^T v - \frac{1}{n}\mathbf{1}_n^T\bar{v}\mathbf{1}_n = \bar{v} - \frac{\bar{v}}{n}\mathbf{1}_n^T\mathbf{1}_n = \bar{v} - \bar{v} = 0.$$

■

Lemma 2.7. *Consider $u \in R^d$ and let $\Sigma \in \mathbb{R}^{d\times d}$ be a covariance matrix. Then:*

$$u\Sigma u^T = var\left(u^T\boldsymbol{X}\right).$$

Proof.

$$\begin{aligned} u\Sigma u^T &= u\left(\mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right] - \mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T\right)u = u^T\mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^T\right]u - u^T\mathbb{E}\left[\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}\right]^T u \\ &= \mathbb{E}\left[u^T\boldsymbol{X}\boldsymbol{X}^T u\right] - \mathbb{E}\left[u^T\boldsymbol{X}\right]\mathbb{E}\left[\boldsymbol{X}^T u\right] = \mathbb{E}\left[(u\boldsymbol{X})^2\right] - \mathbb{E}\left[u\boldsymbol{X}\right]^2 = var\left[u^T\boldsymbol{X}\right] \end{aligned}$$

■

The previous lemma states that the variance of the random vector $\boldsymbol{X}$ along the direction u is given by $u\Sigma u^T$. By the way observe that dimensions are correct because $u\Sigma u^T \in \mathbb{R}$ and $var\left[u^T\boldsymbol{X}\right] \in \mathbb{R}$.

Why we say that $u^T\boldsymbol{X}$ is the projection of $\boldsymbol{X}$ along direction u? This follows from the notion of scalar (dot) product. Indeed we have:

$$u^T\boldsymbol{X} = u_1X^1 + \cdots + u_nX^n = u\cdot\boldsymbol{X} = \|u\|\|\boldsymbol{X}\|\cos\theta,$$

and the term $\|u\|\|\boldsymbol{X}\|\cos\theta$ measures the length of the projection of $\boldsymbol{X}$ along the direction given by u.

2.3 Eigenvalues and Eigenvectors

We remember the following fact.

Theorem 2.8. *Let $\Sigma \in \mathbb{R}^{d\times d}$ be a covariance matrix. Then $\exists$ and orthogonal matrix $P \in \mathbb{R}^{d\times d}$ and a diagonal matrix $D \in \mathbb{R}^{d\times d}$ such that:*

$$\Sigma = PDP^T$$

Observe that $P = [v_1, \ldots, v_d]$ where $v_i \in R^d$ and $v_i \cdot v_j = 0$ if $i \neq j$ and $v_i \cdot v_i = 1$.
Furthermore we have that:

$$\Sigma v_1 = PDP^T v_1 = v_1\lambda_1.$$

λ_1 is called eigenvalue and v_1 is the eigenvector associated to the eigenvalue λ_1.
If we now consider the variance along the direction v_1 we have that:

$$v_1^T\Sigma v_1 = \lambda_1 v_1 v_1^T = \lambda_1.$$

Hence we can conclude that the variance along the direction given by the eigenvector v_1 is given by λ_1. In other words, λ_1 represent the variance of $\boldsymbol{X}$ along the direction identified by the eigenvector v_1.

Proposition 2.9. *Consider $u \in R^d$ and the random vector $\boldsymbol{X} \in \mathbb{R}^d$. The variance of $u^T \boldsymbol{X}$ is maximized for $u = v_1$, where v_1 is the eigenvector associated to the biggest eigenvalue λ_1.*

Proof. Assume that $\mathbb{E}[\boldsymbol{X}] = \mathbf{0}$ and consider $u \in \mathbb{R}^d$ such that $\|u\| = 1$. We have that:

$$var\left(u^T \boldsymbol{X}\right) = \mathbb{E}\left[\left(u^T X\right)^2\right] = \mathbb{E}\left[u^T \boldsymbol{X} \boldsymbol{X}^T u\right] = u^T \mathbb{E}\left[\boldsymbol{X} \boldsymbol{X}^T\right] u = u^T \Sigma u.$$

Define $b = P^T u \in \mathbb{R}^d$: we have that:

$$u^T \Sigma u = u^T P D P^T u = \left(P^T u\right) D \left(P^T u\right) = b^T D b = \sum_{j=1}^{d} b_j^2 \lambda_j \leq \lambda_1 \sum_{j=1}^{n} b_j^2 = \lambda_1 \|b\|^2.$$

Observe that:

$$\lambda_1 \|b\|^2 = b^T b = \left(P^T u\right) P^T u = u^T P P^T u = u^T u = 1.$$

Hence we have that:

$$var\left(u^T \boldsymbol{X}\right) = u^T \Sigma u \leq \lambda_1,$$

which holds for every $u \in R^d$ such that $\|u\| = 1$. Hence we have that:

$$\sup_{u \in R^d,\, \|u\|=1} u^T \Sigma u \leq \lambda_1.$$

Consider now the eigenvector $v_1 \in \mathbb{R}^d$ associated to λ_1 and recall that $\|v_1\| = 1$.

$$\begin{aligned} v_1^T \Sigma v_1 = \boldsymbol{v_1}^T P D P^T v_1 &= \left(P^T v_1\right)^T D P^T v_1 \\ &= \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & \dots & \lambda_d \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \lambda_1. \end{aligned}$$

Hence, along the direction of v_1 the variance is given exactly by the eigenvalue λ_1. We conclude that the direction of v_1 is the direction which maximize the variance. ■

3 Principal Component Analysis

In order to reduce the size of the data-set we can simply chose $k \geq d$ principal components and project on a subspace of $\mathbb{R}^d$.

$$P_k = [v_1, \dots, v_k], \qquad P_k \in \mathbb{R}^{d \times k}.$$

Define:

$$\boldsymbol{Y}_i = P_k^T \boldsymbol{X}_i,$$

$P_k \in \mathbb{R}^{d \times k}$, $\boldsymbol{X}_i \in \mathbb{R}^d$ and hence $\boldsymbol{Y}_i \in R^k$. Then:

$$\mathfrak{Y} = \left[\boldsymbol{Y}_1^T, \dots, \boldsymbol{Y}_n^T\right]$$

is such that $\mathfrak{Y}$ $in \mathbb{R}^{n \times k}$. This sample data-set is such that the total variance is a fraction fo the initial variance of the data-set $\mathcal{X}$.

Theorem 3.1. *[Johnson and Wichern [1, Result 8.2]] Let $\boldsymbol{X}$ have covariance matrix Σ with eigenvalue-eigenvector pair $(\lambda_1, v_1), \ldots, (\lambda_d, v_d)$ where $\lambda_1 \geq \lambda_2 \geq \ldots, \geq \lambda_d \geq 0$. Let $Y_i = v_1 \boldsymbol{X}$ for $i = 1, \ldots, d$ be the principal components. Then:*

$$\sigma_{11} + \cdots + \sigma_{dd} = \sum_{i=1}^{p} var\,(X_i) = \lambda_1 + \cdots + \lambda_d = \sum_{i=1}^{p} var(Y_i)$$

This result states that the total population variance is given by the sum of the eigenvalue λ_i, for $i = 1, \ldots, d$. Hence the percentage of the variance explained by the principal component Y_k is given by:

$$\frac{\lambda_k}{\sum_{i=1}^{d} \lambda_i}, \quad k = 1, \ldots, d.$$

Usual the number of principal components k is chosen such that they are enough to explain a sufficiently large percentage of the variance. Usually this choice is made heuristically, namely there is not a rigorous way to chose the number k. See Johnson and Wichern [1] for details.

The principal component Y^i is given by $Y^i = v_i^T \boldsymbol{X} = v_i^1 X^1 + \cdots + v_i^d X^d$. The magnitude of v_1^j measure the importance of the j-th variable to the i-th principal component, irrespective of the other variables. In particular v_1^j is proportional to the correlation coefficient between Y^1 and X^k.

Proposition 3.2. *[Johnson and Wichern [1, Result 8.3] If $Y^1 = v_1^T \boldsymbol{X}, \ldots, Y^d = v_d^T \boldsymbol{X}$ are the principal components obtained from the covariance matrix Σ, then:*

$$\rho_{Y^i, X^k} = \frac{v_i^k \sqrt{\lambda_i}}{\sqrt{\sigma_{kk}}}, \quad i, k = 1, \ldots, d$$

are the correlation coefficient between the components Y^i and the variables X^k.

3.1 PCA Algorithm

To summarize the PCA algorithm is the followind.
Consider $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$, $\boldsymbol{X}_i \in \mathbb{R}^d$ a cloud of n points in dimension d.

- Compute the empirical covariance matrix $S \in \mathbb{R}^{d \times d}$.
- Compute the decomposition $S = PDP^T$, where $D = diag\,(\lambda_1, \ldots, \lambda_d)$, with $\lambda_1 \geq \lambda_2 \geq \ldots, \geq 0$ and $P = [v_1, \ldots, v_n]$ is an orthogonal matrix.
- Chose $k < d$ and set $P_k = [v_1, \ldots, v_k] \in \mathbb{R}^{d \times k}$.
- The output is $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n$ where $\boldsymbol{Y}_i = P_k^T \boldsymbol{X}_i \in \mathbb{R}^k$, $i = 1, \ldots, n$.

1 PCA with python

In this example we will use PCA for dimansional reduction and we show how it can be used for classification. We will use the dataset collecting the measures of some breast cancers. Some of them are benign and others are malignant,

[1]:
```
from sklearn.datasets import load_breast_cancer
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```
# Load data relative to breast cancer
breast = load_breast_cancer()
```

[3]:
```
# Get the data as numpy array fortmat
breast_data = breast.data
```

[4]:
```
# The the target: these are 0 or 1. 0 if Benign, 1 Malignant
breast_labels = breast.target
```

[5]:
```
# reshape so you can concatenate
labels = np.reshape(breast_labels,(569,1))
```

[6]:
```
# Join numpy array data with numpy array labels
final_breast_data = np.concatenate([breast_data,labels],axis=1)
```

[7]:
```
# Create a dataframe
breast_dataset = pd.DataFrame(final_breast_data)
# Get the name of the variables
features = breast.feature_names
```

[8]:
```
# Add a label colulms named label which will cointain the 0,1 for Benigng and␣
↪Malignant
features_labels = np.append(features,'label')
```

[9]:
```
# Give a name to the columns of the dataset
breast_dataset.columns = features_labels
breast_dataset.head()
```

[9]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	\
0	17.99	10.38	122.80	1001.0	0.11840	
1	20.57	17.77	132.90	1326.0	0.08474	
2	19.69	21.25	130.00	1203.0	0.10960	
3	11.42	20.38	77.58	386.1	0.14250	
4	20.29	14.34	135.10	1297.0	0.10030	

```
   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst area  \
0                 0.07871  ...          17.33           184.60      2019.0
1                 0.05667  ...          23.41           158.80      1956.0
2                 0.05999  ...          25.53           152.50      1709.0
3                 0.09744  ...          26.50            98.87       567.7
4                 0.05883  ...          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
4            0.1374             0.2050           0.4000                0.1625

   worst symmetry  worst fractal dimension  label
0          0.4601                  0.11890    0.0
1          0.2750                  0.08902    0.0
2          0.3613                  0.08758    0.0
3          0.6638                  0.17300    0.0
4          0.2364                  0.07678    0.0

[5 rows x 31 columns]
```

[10]:
```
# Replace 0 with Benign and 1 with Malignant
breast_dataset['label'].replace(0, 'Benign',inplace=True)
breast_dataset['label'].replace(1, 'Malignant',inplace=True)
```

[11]:
```
# THE PCA
# The the data in a numpy array format
x = breast_dataset.loc[:, features].values
x = StandardScaler().fit_transform(x) # normalizing the features
```

[12]:
```
# Creates numbers for features normalized. Now the have no immediate meaning
feat_cols = ['feature'+str(i) for i in range(x.shape[1])]
normalised_breast = pd.DataFrame(x,columns=feat_cols)
```

[13]:
```
# PCA with only two components
# Now comes the critical part, the next few lines of code will be projecting the␣
 ↪thirty-dimensional
```

```
# Breast Cancer data to two-dimensional principal components.
pca_breast = PCA(n_components=2)
# Call the PCA
principalComponents_breast = pca_breast.fit_transform(x)
```

[14]:
```
# Get the eigenvectors
eigenvectors = pca_breast.components_

# Check che norm of the eigenvectos
print("The norm of eigenvector v1 is %5.2f" %  np.linalg.norm(eigenvectors[0,:]))
print("The norm of eigenvector v2 is %5.2f" %  np.linalg.norm(eigenvectors[1,:]))
print("The dot products of v1 and v2 is %5.2f" %  np.dot((eigenvectors[0,:
 ↪]),(eigenvectors[1,:])))
```

```
The norm of eigenvector v1 is  1.00
The norm of eigenvector v2 is  1.00
The dot products of v1 and v2 is -0.00
```

[15]:
```
# Get the eigen values
eigenvalues = pca_breast.explained_variance_
print("The norm of eigenvalue lambda1 is %5.2f" %  eigenvalues[0])
print("The norm of eigenvalue lambda2 is %5.2f" %  eigenvalues[1])
```

```
The norm of eigenvalue lambda1 is 13.30
The norm of eigenvalue lambda2 is  5.70
```

[16]:
```
# Get the variance ration explained by the two eigenvalues
variance_ratio = pca_breast.explained_variance_ratio_
print(variance_ratio)
```

```
[0.44272026 0.18971182]
```

[17]:
```
principal_breast_Df = pd.DataFrame(data = principalComponents_breast
             , columns = ['principal component 1', 'principal component 2'])
```

[18]:
```
# Get the principal components, eg the data transformed in the new system of␣
 ↪cohordinates
principal_breast_Df.head()
```

[18]:

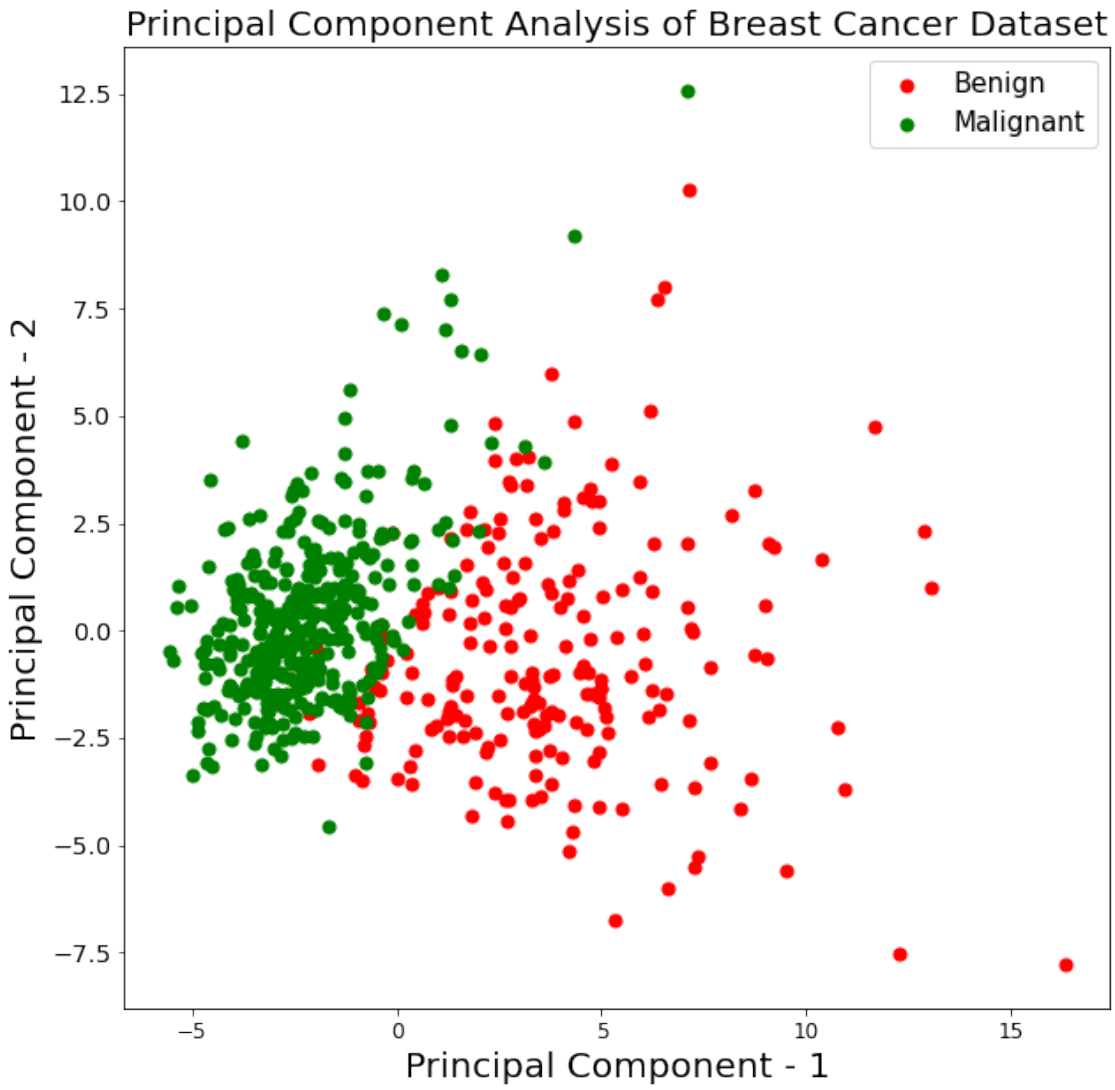
	principal component 1	principal component 2
0	9.192837	1.948583
1	2.387802	-3.768172
2	5.733896	-1.075174
3	7.122953	10.275589
4	3.935302	-1.948072

[19]:
```
print('Explained variation per principal component: {}'.format(pca_breast.
 ↪explained_variance_ratio_))
```

```
Explained variation per principal component: [0.44272026 0.18971182]
```

[20]:
```
plt.figure()
plt.figure(figsize=(10,10))
plt.xticks(fontsize=12)
plt.yticks(fontsize=14)
plt.xlabel('Principal Component - 1',fontsize=20)
plt.ylabel('Principal Component - 2',fontsize=20)
plt.title("Principal Component Analysis of Breast Cancer Dataset",fontsize=20)
targets = ['Benign', 'Malignant']
colors = ['r', 'g']
for target, color in zip(targets,colors):
    indicesToKeep = breast_dataset['label'] == target
    plt.scatter(principal_breast_Df.loc[indicesToKeep, 'principal component 1']
                , principal_breast_Df.loc[indicesToKeep, 'principal component␣
 ↪2'], c = color, s = 50)

plt.legend(targets,prop={'size': 15})
```

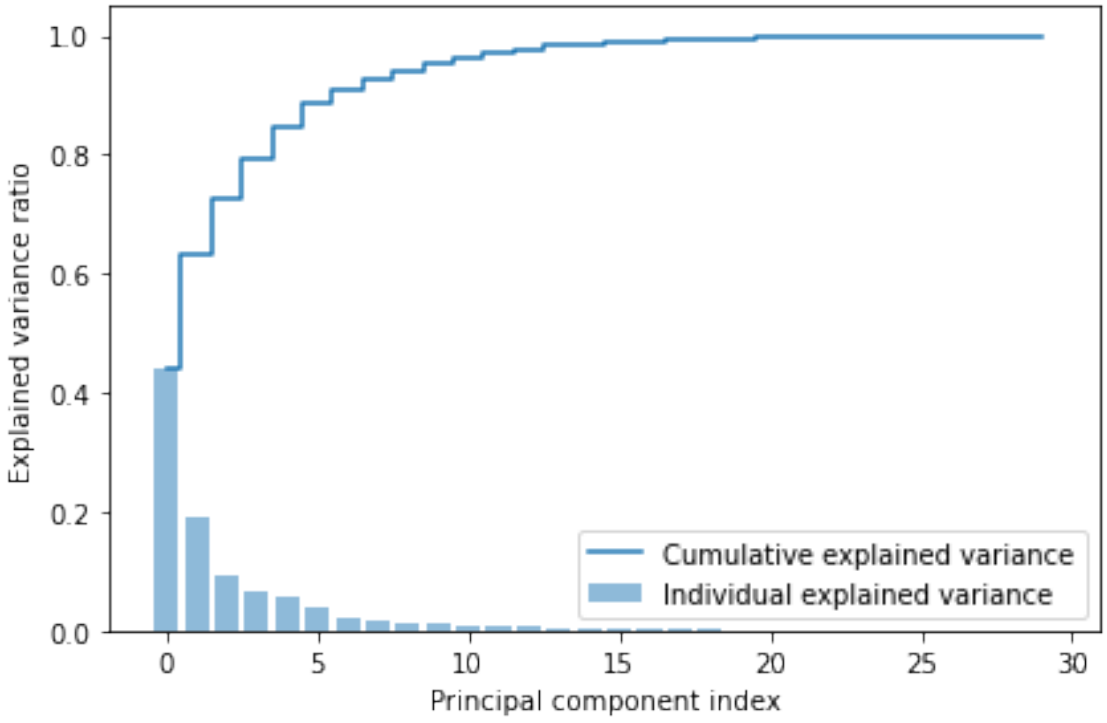
[20]:
```
<matplotlib.legend.Legend at 0x1aeb7b88490>

<Figure size 432x288 with 0 Axes>
```



Observe the previous plot. We can state that we can use PCA also to predict if a given type of cancer is Benign or Malignant simply projecting the data on a 2D plot. This holds because it appears that the values can be separated by a line dividing the plane in two halfs: one in which the brest cancer is benign and the otherone in which it is malignant

[21]:
```
# Instantiate PCA
#
pca = PCA()

# Determine transformed features
#
X_train_pca = pca.fit_transform(x)

# Determine explained variance using explained_variance_ration_ attribute
```

```
#
exp_var_pca = pca.explained_variance_ratio_

# Cumulative sum of eigenvalues; This will be used to create step plot
# for visualizing the variance explained by each principal component.
#
cum_sum_eigenvalues = np.cumsum(exp_var_pca)

# Create the visualization plot
#
plt.bar(range(0,len(exp_var_pca)), exp_var_pca, alpha=0.5, align='center',␣
 ↪label='Individual explained variance')
plt.step(range(0,len(cum_sum_eigenvalues)), cum_sum_eigenvalues,␣
 ↪where='mid',label='Cumulative explained variance')
plt.ylabel('Explained variance ratio')
plt.xlabel('Principal component index')
plt.legend(loc='best')
plt.tight_layout()
plt.show()
```



We can state that the first four principal components explains the 80% percent of the variance.

References

[1] Richard Arnold Johnson and Dean W. Wichern. *Applied multivariate statistical analysis*. Prentice Hall, Upper Saddle River, NJ, 5. ed edition, 2002. ISBN 0130925535. URL http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+330798693&sourceid=fbw_bibsonomy.