1 Principal Component Analysis (PCA)

Introduction

StatQuest: Principal Component Analysis (PCA), Step-by-Step
Visualization of structured and unstructured data

1.1 What is PCA?

Principal Component Analysis (PCA) is a linear dimensionality reduction technique (technika liniowej redukcji wymiarowości) that can be utilized for extracting information from a high-dimensional space by projecting (rzutowanie) it into a lower-dimensional sub-space. It tries to preserve (zachować) the essential (istotne) parts that have more variation of the data (większą zminoość) and remove the non-essential parts with fewer (mniejszą) variation.

One important thing to note about PCA is that it is an **Unsupervised** dimensionality reduction technique, you can cluster the similar data points based on the feature correlation between them without any supervision (nadzoru) (or labels).

PCA is a statistical procedure that uses an orthogonal transformation (przekształcenia ortogonalne) to convert a set of observations of possibly (potencjalnie) correlated variables (entities (jednostek) each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components. Features, Dimensions, and Variables (zmienne) are all referring to the same thing in this tutorial.

1.2 What is dimentions?

Dimensions are nothing (nic innego) but features that represent the data. For example, a 28×28 image has 784 picture elements (pixels) that are the dimensions or features which together represent that image.

2 What is main usage PCA (główne zastosowanie)?

2.1 Data Visualization

When working on any data related (związany) problem, extensive (rozległa) data exploration like finding out (odkrywanie) how the variables are correlated or understanding the distribution of a few variables is crucial (kluczowa). Considering (biorąc pod uwagę) that there are a large number of variables or dimensions along which (wzdłuż których) the data is distributed (rozproszone), visualization can be a challenge and almost impossible. Using dimensionality reduction, data can be projected into a lower dimension, thereby (dzięki czemu) allowing you to visualize the data in a 2D or 3D space.

2.2 Speeding Machine Learning Algorithm

Since (ponieważ) PCA's main idea (główna idea) is dimensionality reduction, you can leverage (wykorzystać) that to speed up your machine learning algorithm's training and testing time considering your data has a lot of features, and the ML algorithm's learning is too slow.

3 What is mathematical Foundation of PCA?

From a mathematical perspective, PCA can be viewed as (postrzegany jako) an eigendecomposition (rozkład na wartości własne) of the data covariance matrix or as a singular (osobliwych) value decomposition (SVD) of the data matrix.

3.1 Problem Formulation

Given a dataset $\mathbf{X} \in \mathbb{R}^{n \times p}$ with n observations and p features, our goal is to find a linear transformation that maps this data to a lower-dimensional space while maximizing variance.

Let $\mathbf{X}$ be a centered matrix (meaning each feature has zero mean). The sample covariance matrix is:

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^\top \mathbf{X}$$

$$\text{cov}_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

gdzie: x,y to dwie zmienne - cechy

PCA aims to find a set of orthogonal vectors $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k$ (where $k < p$) such that the projection of the data onto these vectors has maximum variance.

3.2 Discuss the optimization problem in PCA.

For the first principal component, we aim to find:

$$\mathbf{w}_1 = \arg\max_{\|\mathbf{w}\|=1} \mathbf{w}^\top \mathbf{C}\mathbf{w}$$

Geometric Perspective:
Projecting the data matrix $\mathbf{X}$ onto the direction $\mathbf{w}$:

$$\mathbf{t} = \mathbf{X}\mathbf{w}$$

This yields a vector $\mathbf{t} \in \mathbb{R}^n$ – a new coordinate for each observation after projection.

Statistical Perspective:
The variance of these values (i.e., how much the data spreads along the direction $\mathbf{w}$) is:

$$\text{Var}(\mathbf{t}) = \text{Var}(\mathbf{X}\mathbf{w}) = \mathbf{w}^\top \mathbf{C}\mathbf{w}$$

This objective function (funkcja celu) represents the variance of the data when projected onto $\mathbf{w}$. The constraint (ograniczenie) $\|\mathbf{w}\| = 1$ ensures (zapewnia) that $\mathbf{w}$ is a unit vector (wektor jednostkowy).

The solution to this optimization problem is the eigenvector (wektor własny) corresponding (odpowiadający) to the largest eigenvalue (największej wartości własnej) of $\mathbf{C}$. Similarly, the j-th principal component corresponds to the eigenvector of the j-th largest eigenvalue.

The covariance matrix $\mathbf{C} = \frac{1}{n-1}\mathbf{X}^\top\mathbf{X}$ always has:

- eigenvalues $\lambda_i \geq 0$
- because it is **symmetric** and **positive semi-definite**

3.3 Eigendecomposition Approach

Let's decompose the covariance matrix $\mathbf{C}$ into its eigenvalues and eigenvectors:

$$\mathbf{C} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$$

where:

- $\mathbf{V}$ is a matrix with eigenvectors as its columns
- $\mathbf{\Lambda}$ is a diagonal matrix with eigenvalues in descending order

The principal components are the columns of $\mathbf{V}$, with the first column corresponding to the largest eigenvalue, representing the direction of maximum variance.

3.4 SVD Approach

Alternatively, we can use Singular Value Decomposition (SVD) directly (bezpośrednio) on the centered (scentralizowanje) data matrix:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

where:

- $\mathbf{U}$ is an $n \times n$ orthogonal matrix
- $\mathbf{\Sigma}$ is an $n \times p$ diagonal matrix with non-negative singular values
- $\mathbf{V}$ is a $p \times p$ orthogonal matrix

The principal components are the columns of $\mathbf{V}$, and the principal scores (projections of data onto principal components) are given by $\mathbf{XV}$.

3.5 Data Projection

Once we have found the principal components, we can project the original data onto the lower-dimensional space:

$$\mathbf{T} = \mathbf{X}\mathbf{W}$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k]$ contains the first k principal components.
The reconstructed data in the original space can be obtained by:

$$\mathbf{X}_{\text{rec}} = \mathbf{T}\mathbf{W}^\top$$

The error between the original and reconstructed data is minimized in the least-squares sense. (w sensie najmniejszych kwadratów)

3.6 Conceptual Understanding

Dimensions are simply features that represent the data. For example, a 28×28 image has 784 pixels, each being a dimension or feature that collectively (wspólnie) represents the image.

One important characteristic of PCA is that it's an unsupervised dimensionality reduction technique. This means it can cluster similar data points based on feature correlations without any supervision (labels).

PCA is essentially (w istocie) a statistical procedure that applies (stosuje) an orthogonal transformation to convert a set of possibly correlated variables into a set of linearly uncorrelated variables called principal components. Throughout (przez cały czas) this tutorial, the terms *features*, *dimensions*, and *variables* are used interchangeably (zamiennie).

4 What is the kernel used for in PCA?

Kernel PCA extends (rozszerza) traditional PCA to handle (umożliwiając) nonlinear dimensionality reduction by implicitly (niejawne) mapping (odwzorowanie) data into a higher-dimensional feature space.

While standard PCA finds principal components in the original feature space, Kernel PCA operates in a transformed space $\mathcal{F}$ using a nonlinear mapping function $\phi : \mathbb{R}^p \to \mathcal{F}$.

Instead of computing this mapping explicitly (jawnie), Kernel PCA uses the "kernel trick" by defining (definiując) a kernel function:

5 Mathematical Formulation

We define a kernel function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Common kernel functions include:

- Polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + c)^d$
- RBF/Gaussian: $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$
- Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha \mathbf{x}_i^T \mathbf{x}_j + c)$

6 Where did these specific functions come from?

- Mathematicians and researchers discovered that **certain functions** (e.g., RBF, polynomial, sigmoid) satisfy the so-called (tak zwany) **Mercer's condition**, which means they can be interpreted as inner products (iloczyn skalarny) in *some (possibly infinite-dimensional (nieskończonej)) space*.
- This allows them to be used as **kernel functions**.

For example:

- **RBF kernel:**

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$$

behaves as if it were:

$$\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

in some **infinite-dimensional feature space**. And **you don't need to know** what that space is—you just use the ready-made (gotową) formula.

7 Key Differences (kluczowe różnice) from PCA

1. **Nonlinearity**: While PCA can only identify (zidentyfikować) linear patterns, Kernel PCA can capture nonlinear relationships in the data.
2. **Implicit Feature Space (niejawna przestrzeń cech)**: Kernel PCA operates in a potentially infinite-dimensional feature space without explicitly computing the transformation.

3. **Computational Approach (podejście obliczeniowe)**: Instead of eigendecomposition (rozkładu wartości własnych) of the covariance matrix, Kernel PCA solves the eigenvalue problem for the kernel matrix $\mathbf{K}$, where $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$.

4. **Hyperparameter Selection**: Kernel PCA requires selecting an appropriate kernel function and its parameters, making it more flexible but also more complex to optimize.

5. **Pre-image Problem (problem odwzorowania odwrotnego)**: Unlike (w przeciwieństwie do) PCA, reconstructing original data from Kernel PCA projections (the "pre-image problem") is non-trivial and often approximated.

	Classic PCA	**Kernel PCA**
Input	$n \times m$	$n \times m$
Matrix analyzed	Covariance $C = m \times m$	Kernel $K = n \times n$
What is compared?	Features with each other	Samples with each other
Space	Original (feature space)	Hidden (samples in higher dimension)

Table 1: Comparison of Classic PCA and Kernel PCA

8 When to Use Kernel PCA?

Kernel PCA is particularly useful (przydatny) when:

- The data has nonlinear patterns that linear PCA cannot capture.
- Visualization of complex, high-dimensional data is needed.
- Pre-processing for nonlinear classification problems.