Lecture 10: Dimensionality Reduction with Principal Component Analysis

CSE4130: 기초머신러닝

Junsuk Choe (최준석)

Roadmap

(1) Problem Setting

(2) Maximum Variance Perspective

(3) Projection Perspective

(4) Eigenvector Computation and Low-Rank Approximations

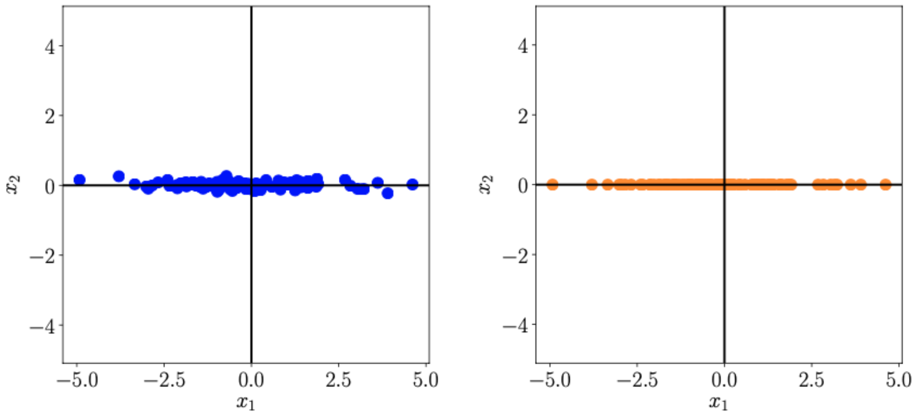
(5) PCA in High Dimensions

(6) Key Steps of PCA in Practice

(7) Latent Variable Perspective

Roadmap

Dimensionality Reduction



(a) Dataset with x_1 and x_2 coordinates. (b) Compressed dataset where only the x_1 coordinate is relevant.

- High-dimensional data
  - hard to analyze and visualize
  - Often, overcomplete and many dimensionas are redundant
- Compact data representation is always preferred just like compression.
- PCA (Principal Component Analysis) is a representative method.

Example: Housing Data

- 5 dimensions
  1. Size
  2. Number of rooms
  3. Number of bathrooms
  4. Schools around
  5. Crime rate
- 2 dimensions
  - Size feature
  - Location feature

PCA Algorithm

S1. Centering. Centering the data by subtracting mean

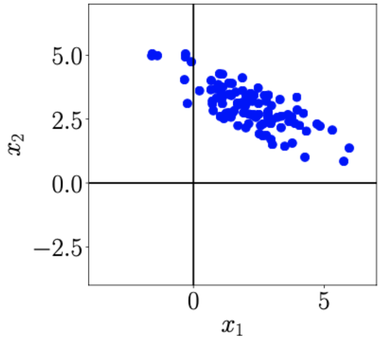
S2. Standardization. Divide the data points by the standard deviation for every dimension (original feature) $d = 1, \ldots, D$

S3. Eigenvalue/vector. Compute the M-largest eigenvalues and the eigenvectors of the data covariance matrix (M is the dimension that needs to be reduced)

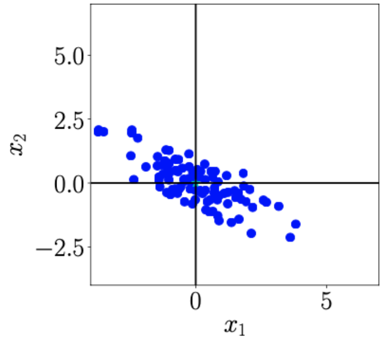
S4. Projection. Project all data points onto the space defined by the eigenvectors (i.e., principal subspace).

S5. Undo standardization and centering.

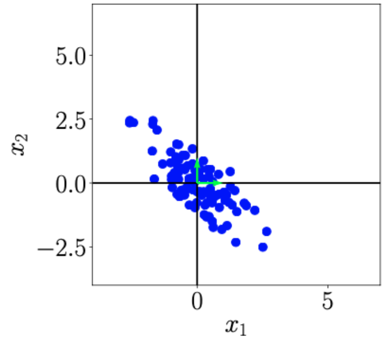
PCA Illustration



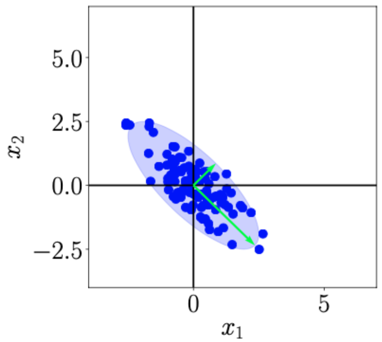
(a) Original dataset.



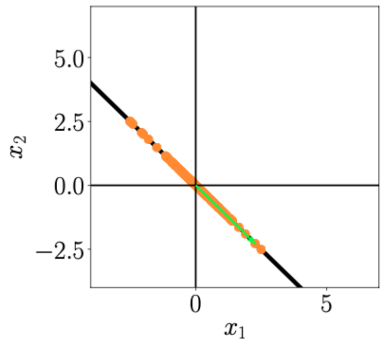
(b) Step 1: Centering by subtracting the mean from each data point.



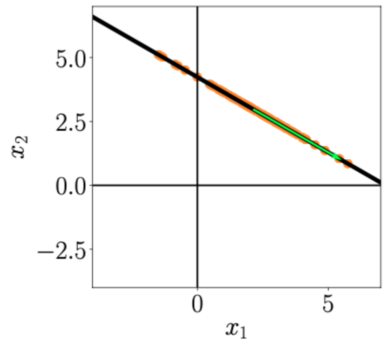
(c) Step 2: Dividing by the standard deviation to make the data unit free. Data has variance 1 along each axis.



(d) Step 3: Compute eigenvalues and eigenvectors (arrows) of the data covariance matrix (ellipse).



(e) Step 4: Project data onto the principal subspace.



(f) Undo the standardization and move projected data back into the original data space from (a).

Data Matrix and Data Covariance Matrix

- N: number of samples, D: number of measurements (or original features)
- iid dataset $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ whose mean is $\mathbf{0}$ (well-centered), where each $\boldsymbol{x}_i \in \mathbb{R}^D$, and its corresponding data matrix

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_N \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,N} \\ & \vdots & & \\ x_{D,1} & x_{D,2} & \cdots & x_{D,N} \end{pmatrix} \in \mathbb{R}^{D \times N}$$

- (data) covariance matrix

$$\boxed{\boldsymbol{S} = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\mathsf{T} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^\mathsf{T} \in \mathbb{R}^{D \times D}}$$

Covariance Matrix and Data Covariance Matrix

- Covaiance matrix for a random vector $\boldsymbol{Y} = (Y_1, \ldots, Y_D)^{\mathsf{T}}$, L6(4)

$$\boldsymbol{\Sigma}_{\boldsymbol{Y}} = \begin{pmatrix} \mathrm{cov}(Y_1, Y_1) & \mathrm{cov}(Y_1, Y_2) & \cdots \mathrm{cov}(Y_1, Y_D) \\ \vdots & \vdots & \vdots \\ \mathrm{cov}(Y_D, Y_1) & \mathrm{cov}(Y_n, Y_2) & \cdots \mathrm{cov}(Y_D, Y_D) \end{pmatrix}$$

- Data convariance matrix $\boldsymbol{S} \in \mathbb{R}^{D \times D}$
  - Each Y_i has N samples $\begin{pmatrix} x_{i,1} & \cdots & x_{i,N} \end{pmatrix}$

$$\begin{aligned} \boldsymbol{S}_{ij} = \mathrm{cov}(Y_i, Y_j) \quad &= \quad \frac{1}{N} \sum_{k=1}^{N} x_{i,k} \cdot x_{j,k} \\ &= \quad \text{average covariance (over samples) btwn feastures } i \text{ and } j \end{aligned}$$

Code: Low Dimensional Representation

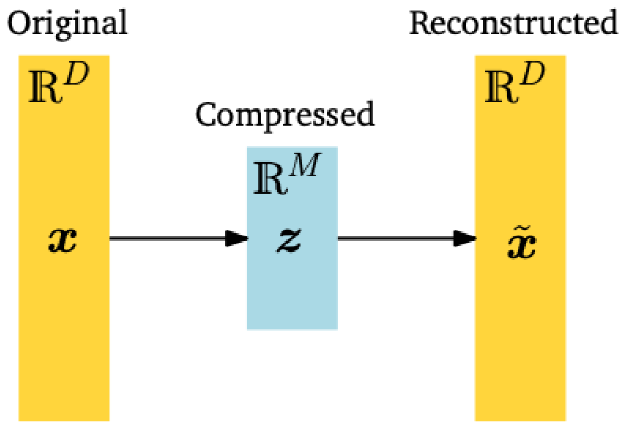
- Low-dimensional compressed representation, also called code:

$$\boldsymbol{z}_n = \boldsymbol{B}^\top \boldsymbol{x}_n \in \mathbb{R}^M,$$

  where the projection[1] matrix is $\boldsymbol{B} := (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M) \in \mathbb{R}^{D \times M}$,
- Assume that the columns of $\boldsymbol{B}$ are orthonormal, i.e., $\boldsymbol{b}_i^\top \boldsymbol{b}_j = 0$ if $i \neq j$, and $\boldsymbol{b}_i^\top \boldsymbol{b}_i = 1$ if $i = j$.
- Seek an M-dimensional subspace $U \subset \mathbb{R}^D$, $\dim(U) = M < D$ onto which we project data
- $\tilde{\boldsymbol{x}}_n \in \mathbb{R}^D$: projected data, $\boldsymbol{z}_n$: their coordinates w.r.t. the basis vectors of $\boldsymbol{B}$.

[1] In L3(8), the coordinate in the projected space becomes $\boldsymbol{\lambda} = (\boldsymbol{B}^\top \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{x}$, which is simply $\boldsymbol{B}^\top \boldsymbol{x}$ for orthonormal bases $\boldsymbol{B}$.

PCA: Encoder and Decoder Viewpoint

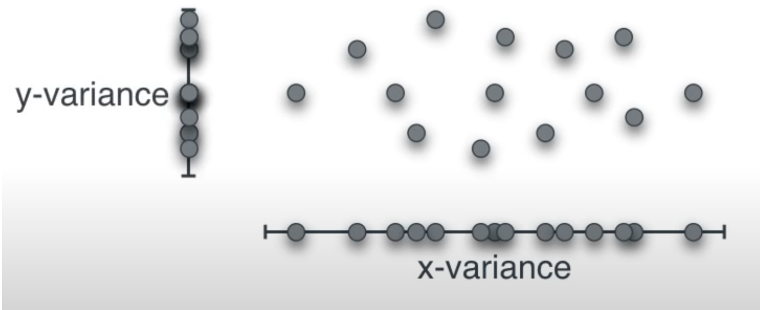


- Find a suitable matrix B such that $\boldsymbol{z} = \boldsymbol{B}^\mathsf{T}\boldsymbol{x}$ and $\tilde{\boldsymbol{x}} = \boldsymbol{B}\boldsymbol{z}$
- $\boldsymbol{B}^\mathsf{T}$: encoder, $\boldsymbol{B}$: decoder
- Example. MNIST dataset
  - handwritten digits, $N = 60,000$ data samples, $D = 28 \times 28 = 784$ pixels

Roadmap

Idea

- Information content in the data
  - space filling
  - information in the data by looking at how much data is spread out
- PCA
  - a dimensinoality reduction algorithm that maximizes the variance in the low-dimensional data representation.



source: Youtube channel by Luis Serrano

Matrix Again: B, $\boldsymbol{z}_n$, and $\boldsymbol{x}_n$

- $\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \dots & \boldsymbol{b}_M \end{pmatrix}$, where $\boldsymbol{b}_i \in \mathbb{R}^D$ and $\boldsymbol{B} \in \mathbb{R}^{D \times M}$
- $\boldsymbol{B}^\top = \begin{pmatrix} \boldsymbol{b}_1^\top \\ \vdots \\ \boldsymbol{b}_M^\top \end{pmatrix} \in \mathbb{R}^{M \times D}$, $\boldsymbol{b}_i^\top \in \mathbb{R}^{1 \times D}$, $\boldsymbol{x}_i \in \mathbb{R}^{D \times 1}$
- $\boldsymbol{z}_n = \begin{pmatrix} z_{1n} \\ \vdots \\ z_{Mn} \end{pmatrix} = \boldsymbol{B}^\top \boldsymbol{x}_n = \begin{pmatrix} \boldsymbol{b}_1^\top \\ \vdots \\ \boldsymbol{b}_M^\top \end{pmatrix} \boldsymbol{x}_n = \begin{pmatrix} \boldsymbol{b}_1^\top \boldsymbol{x}_n \\ \vdots \\ \boldsymbol{b}_M^\top \boldsymbol{x}_n \end{pmatrix}$
- z_{in}: new coordinate (for $\boldsymbol{x}_n$) in the projected space by the basis $\boldsymbol{b}_i$

What We Will Do Is ...

- Goal: Find the orthonormal bases $\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \dots & \boldsymbol{b}_M \end{pmatrix}$ that maximizes the variance.
- Result: For the M-largest eigenvalues $\lambda_1, \dots, \lambda_M$ of the data covariance matrix $\boldsymbol{S}$, their corresponding M eigenvectors become $\boldsymbol{b}_1, \dots, \boldsymbol{b}_M$
- Question. Why data covariance matrix? Why eigenvectors ordered by their eigenvalues?
- Strategy: Induction

Step 1. We seek a single vector $\boldsymbol{b}_1$ that maximizes the variance of the projected data, assuming that we project the data onto an 1D line. We show that $\boldsymbol{b}_1$ is the eigenvector of the largest eigenvalue.

Step k. Suppose that we found $\boldsymbol{b}_1, \dots, \boldsymbol{b}_{k-1}$ for the variance maximization. Then, we seek $\boldsymbol{b}_k$ that maximizes the variance of the projected data onto k-D plain with the constraint that $\boldsymbol{b}_k$ is orthogonal to $\boldsymbol{b}_1, \dots, \boldsymbol{b}_{k-1}$. We prove that $\boldsymbol{b}_k$ is the eigenvector of the k-th largest eigenvalue.

Step 1: Finding $\boldsymbol{b}_1$ (1)

- Variance (over N sample data) of the first coordinate z_1 of $\boldsymbol{z} \in \mathbb{R}^M$, so that

$$V_1 := \text{var}[z_1] = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2, \quad z_{1n} = \boldsymbol{b}_1^\top \boldsymbol{x}_n$$

where z_{1n} (z_{in}) is the first (i-th) coordinate of the low-dimensional representation $\boldsymbol{z}_n$ of $\boldsymbol{x}_n$

$$V_1 = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{b}_1^\top \boldsymbol{x}_n)^2 = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{b}_1^\top \boldsymbol{x}_n \boldsymbol{x}_n^\top \boldsymbol{b}_1 = \boldsymbol{b}_1^\top \left(\frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^\top\right) \boldsymbol{b}_1 = \boldsymbol{b}_1^\top \boldsymbol{S} \boldsymbol{b}_1$$

- Find $\boldsymbol{b}_1$ that maximizes V_1.

$$\max_{\boldsymbol{b}_1} \boldsymbol{b}_1^\top \boldsymbol{S} \boldsymbol{b}_1, \quad \text{subject to} \quad \|\boldsymbol{b}_1\|^2 = 1$$

Step 1: Finding $\boldsymbol{b}_1$ (2)

- Optimization problem

$$\max_{\boldsymbol{b}_1} \boldsymbol{b}_1^\top \boldsymbol{S} \boldsymbol{b}_1, \quad \text{subject to} \quad \|\boldsymbol{b}_1\|^2 = 1$$

- Using the Lagrange multiplier method, we get: L7(2), L7(4)

$$\boldsymbol{S}\boldsymbol{b}_1 = \lambda_1 \boldsymbol{b}_1, \quad \boldsymbol{b}_1^\top \boldsymbol{b}_1 = 1 \implies \lambda_1\text{: eigenvalue, } \boldsymbol{b}_1\text{: eigenvector of } \boldsymbol{S}$$

- Then, $V_1 = \boldsymbol{b}_1^\top \boldsymbol{S} \boldsymbol{b}_1 = \lambda_1 \boldsymbol{b}_1^\top \boldsymbol{b}_1 = \lambda_1$ (the variance V_1 is the eigenvalue of S)
- To maximize the variance, we take the largest eigenvalue, and the corresponding eigenvector is called the (first) principal component.

Step k: Finding $\boldsymbol{b}_k$ (1)

- Assume we have found the first $m - 1$ principal components as the $m - 1$ eigenvectors of $\boldsymbol{S}$ that are associated with the largest $m - 1$ eigenvalues.
- Since $\boldsymbol{s}$ is symmetric, the spectral theorem states that we can use these eigenvectors to construct an orthonormal eigenbasis of an $(m - 1)$-dimensional subspace of R^D.

Step k: Finding $\boldsymbol{b}_k$ (2)

Step k: Finding $\boldsymbol{b}_k$ (3)

Roadmap

(1) Problem Setting

(2) Maximum Variance Perspective

(3) Projection Perspective

(4) Eigenvector Computation and Low-Rank Approximations

(5) PCA in High Dimensions

(6) Key Steps of PCA in Practice

(7) Latent Variable Perspective

Storyline

- An ordered orthonormal bais (ONB) $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_D)$
- $\boldsymbol{B} = \begin{pmatrix} \boldsymbol{b}_1 & \boldsymbol{b}_2 & \ldots & \boldsymbol{b}_M \end{pmatrix}$, where $\boldsymbol{b}_i \in \mathbb{R}^D$ and $\boldsymbol{B} \in \mathbb{R}^{D \times M}$
- Encoding: $\boldsymbol{z}_n = \phi(\boldsymbol{x}_n)$ for some mapping $\phi(\cdot)$
- Decoding: $\tilde{\boldsymbol{x}}_n := \boldsymbol{B}\boldsymbol{z}_n = \sum_{m=1}^{M} z_{mn} \boldsymbol{b}_m$
- Goal: find the best linear projection of $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ onto a lower-dimensional subspace U (also, called principal subspace) of $\mathbb{R}^D$ with $\dim(U) = M$.
- Formally, minimize the following reconstruction error

$$J_M := \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2,$$

where the variables are $(\boldsymbol{z}_n : n = 1, \ldots, N)$ and $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$

Two-step Approach

Step 1. We optimize the coordinate $\boldsymbol{z}_n$ in the space U for a given ONB $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$

Step 2. Then, we find the optimal ONB, knowing the optimal $\boldsymbol{z}_n$ in **Step 1.**

Step 1: Optimal coordinate $\boldsymbol{z}_n$ for a given ONB

- Intuition: Orthogonal projection L3(8)

$$\textit{Result}: \tilde{\boldsymbol{x}}_n = \boldsymbol{B}(\boldsymbol{B}^\top \boldsymbol{B})^{-1}\boldsymbol{B}^\top \boldsymbol{x}_n = \boldsymbol{B}\boldsymbol{B}^\top \boldsymbol{x}_n = \boldsymbol{B}\boldsymbol{z}_n, \quad \boldsymbol{z}_n = \boldsymbol{B}^\top \boldsymbol{x}_n$$

- Proof. Assume an ONB $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$. Noting that J_M is a function of $\tilde{\boldsymbol{x}}_n$ and $\tilde{\boldsymbol{x}}_n$ is a function of $\boldsymbol{z}_n$,

$$\frac{\partial J_M}{\partial z_{in}} = \frac{\partial J_M}{\partial \tilde{\boldsymbol{x}}_n}\frac{\partial \tilde{\boldsymbol{x}}_n}{\partial z_{in}}, \quad \frac{\partial J_M}{\partial \tilde{\boldsymbol{x}}_n} = -\frac{2}{N}(\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n)^\top, \quad \frac{\partial \tilde{\boldsymbol{x}}_n}{\partial z_{in}} = \frac{\partial}{\partial z_{in}}\left(\sum_{m=1}^{M} z_{mn}\boldsymbol{b}_m\right) = \boldsymbol{b}_i$$

$$\frac{\partial J_M}{\partial z_{in}} = -\frac{2}{N}(\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n)^\top \boldsymbol{b}_i = -\frac{2}{N}\left(\boldsymbol{x}_n - \sum_{m=1}^{M} z_{mn}\boldsymbol{b}_m\right)^\top \boldsymbol{b}_i \overset{\text{ONB}}{=} -\frac{2}{N}(\boldsymbol{x}_n^\top \boldsymbol{b}_i - z_{in}\boldsymbol{b}_i^\top \boldsymbol{b}_i)$$

$$= -\frac{2}{N}(\boldsymbol{x}_n^\top \boldsymbol{b}_i - z_{in})$$

- $z_{in} = \boldsymbol{x}_n^\top \boldsymbol{b}_i = \boldsymbol{b}_i^\top \boldsymbol{x}_n$ for $i = 1, \ldots, M$ and $n = 1, \ldots, N$ (ortho. proj. onto 1D L3(8))

Step 2: Finding Optimal Basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$ (1)

- The difference: $\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n = \left(\sum_{j=M+1}^{D} \boldsymbol{b}_j \boldsymbol{b}_j^\mathsf{T} \right) \boldsymbol{x}_n = \sum_{j=M+1}^{D} (\boldsymbol{x}_n^\mathsf{T} \boldsymbol{b}_j) \boldsymbol{b}_j$

$$\tilde{\boldsymbol{x}}_n = \sum_{m=1}^{M} z_{mn} \boldsymbol{b}_m \overset{\text{Step 1}}{=} \sum_{m=1}^{M} (\boldsymbol{x}_n^\mathsf{T} \boldsymbol{b}_m) \boldsymbol{b}_m = \sum_{m=1}^{M} \boldsymbol{b}_m (\boldsymbol{b}_m^\mathsf{T} \boldsymbol{x}_n) = \left(\sum_{m=1}^{M} \boldsymbol{b}_m \boldsymbol{b}_m^\mathsf{T} \right) \boldsymbol{x}_n$$

$$\boldsymbol{x}_n = \sum_{d=1}^{D} z_{dn} \boldsymbol{b}_d = \left(\sum_{m=1}^{M} \boldsymbol{b}_m \boldsymbol{b}_m^\mathsf{T} \right) \boldsymbol{x}_n + \left(\sum_{j=M+1}^{D} \boldsymbol{b}_j \boldsymbol{b}_j^\mathsf{T} \right) \boldsymbol{x}_n$$

- The projection of the data point onto the orthogonal complement of the principal subspace L3(6)

Step 2: Finding Optimal Basis $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$ (2)

$$J_M = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \left\| \sum_{j=M+1}^{D} (\boldsymbol{b}_j^\mathsf{T}\boldsymbol{x}_n)\boldsymbol{b}_j \right\|^2 = \frac{1}{N}\sum_{n=1}^{N}\sum_{j=M+1}^{D} (\boldsymbol{b}_j^\mathsf{T}\boldsymbol{x}_n)^2$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{j=M+1}^{D} \boldsymbol{b}_j^\mathsf{T}\boldsymbol{x}_n\boldsymbol{x}_n^\mathsf{T}\boldsymbol{b}_j = \sum_{j=M+1}^{D} \boldsymbol{b}_j^\mathsf{T}\left(\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n\boldsymbol{x}_n^\mathsf{T}\right)\boldsymbol{b}_j = \sum_{j=M+1}^{D} \boldsymbol{b}_j^\mathsf{T}\boldsymbol{S}\boldsymbol{b}_j$$

- minimizing the squared reconstruction error $=$ minimizing the variance when projected onto the orthogonal complement of the principal subspace $=$ maximizing the variance of the projection in the principal subspace

- $J_M = \sum_{j=M+1}^{D} \lambda_j$ (because of the projection). To minimize this error, we need to choose the smallest $D-M$ eigenvalues, which means that we need to choose the M largest eigenvalues and take their corresponding eigenvectors for projection.

Roadmap

(1) Problem Setting

(2) Maximum Variance Perspective

(3) Projection Perspective

(4) Eigenvector Computation and Low-Rank Approximations

(5) PCA in High Dimensions

(6) Key Steps of PCA in Practice

(7) Latent Variable Perspective

Eigenvector Computation

- Approach 1: EVD L4(4)
  - Perform an eigendecomposition and compute the eigenvalues and eigenvectors of the symmetric matrix $\boldsymbol{S}$ directly.
- Approach 2: SVD L4(5)
  - SVD of the data matrix $\boldsymbol{X}$: $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ $([D \times N] = [D \times D] \cdot [D \times N] \cdot [N \times N])$
  - $\boldsymbol{U}$ and $\boldsymbol{V}^\top$: orthogonal matrices, $\boldsymbol{\Sigma}$: only nonzero entries are the singular values $\sigma_{ii} \geq 0$.

$$\boldsymbol{S} = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\top = \frac{1}{N}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top\boldsymbol{V}\boldsymbol{\Sigma}^\top\boldsymbol{U}^\top \overset{(\boldsymbol{V}^\top=\boldsymbol{V}^{-1})}{=} \frac{1}{N}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top\boldsymbol{U}^\top$$

  - The columns of $\boldsymbol{U}$ are the eigenvectors of $\boldsymbol{X}\boldsymbol{X}^\top$ (thus $\boldsymbol{S}$)
  - The eigenvalues λ_d of $\boldsymbol{S}$ are related to the singular values of $\boldsymbol{X}$: $\lambda_d = \dfrac{\sigma_d^2}{N}$

PCA as Low-Rank Matrix Approximations

- In SVD, $\boldsymbol{U}$ corresponds to the projection matrix $\boldsymbol{B}$, so that we maximize the variance of the projected data or minimize the average squared reconstruction error.
- Consider the best rank-M approximation

$$\tilde{\boldsymbol{X}}_M := \arg \min_{\mathsf{rk}(\boldsymbol{A})=M} \|\boldsymbol{X} - \boldsymbol{A}\|_2$$

- From Eckart-Young Theorem, by truncating the SVD at the top-M singular value, we obtain the reconstructed data matrix $\tilde{\boldsymbol{X}}_M$ as: L4(5), L4(6)

$$\tilde{\boldsymbol{X}}_M = \overbrace{\boldsymbol{U}_M}^{D\times M} \overbrace{\Sigma_M}^{M\times M} \overbrace{\boldsymbol{V}_M{}^{\mathsf{T}}}^{M\times N} \iff \tilde{\boldsymbol{X}}_M = \sum_{i=1}^{M} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i{}^{\mathsf{T}},$$

where σ_i is the i-th singular value.

Roadmap

PCA as Low-Rank Matrix Approximations

- In some practical cases, $\boldsymbol{S} = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\mathsf{T} \in \mathbb{R}^{D \times D}$, where D is pretty high.
  - Example. 100×100 pixel image: $D = 10,000$.
- What if $N << D$?
  - With no duplicate data, $\text{rk}(\boldsymbol{S}) = N$, and $D - N + 1$ eigenvalues are 0! $\Longrightarrow$ no need to maintain $D \times D$ data covariance matrix.
- In PCA, $\boldsymbol{S}\boldsymbol{b}_m = \lambda_m \boldsymbol{b}_m$, $m = 1, \ldots, M$.

$$\boldsymbol{S}\boldsymbol{b}_m = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\mathsf{T}\boldsymbol{b}_m = \lambda_m \boldsymbol{b}_m \quad \Longrightarrow \quad \frac{1}{N}\underbrace{\boldsymbol{X}^\mathsf{T}\boldsymbol{X}}_{N\times N}\underbrace{\boldsymbol{X}^\mathsf{T}\boldsymbol{b}_m}_{:=\boldsymbol{c}_m} = \lambda_m \boldsymbol{X}^\mathsf{T}\boldsymbol{b}_m \Longleftrightarrow \frac{1}{N}\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{c}_m = \lambda_m \boldsymbol{c}_m$$

- λ_m is an eigenvalue of $\frac{1}{N}\boldsymbol{X}^\mathsf{T}\boldsymbol{X}$ with its associated eigenvector $\boldsymbol{c}_m = \boldsymbol{X}^\mathsf{T}\boldsymbol{b}_m$
- $\frac{1}{N}\boldsymbol{X}^\mathsf{T}\boldsymbol{X} \in \mathbb{R}^{N \times N}$, so much easier to compute the eigenstuff
- To recover the eigenvector of $\boldsymbol{S}$, by left-multiplying X, we get $\frac{1}{N}\boldsymbol{X}\boldsymbol{X}^\mathsf{T}\boldsymbol{X}\boldsymbol{c}_m = \lambda_m \boldsymbol{X}\boldsymbol{c}_m$

PCA Algorithm

S1. Centering. Centering the data by subtracting mean

S2. Standardization. Divide the data points by the standard deviation for every dimension (original feature) $d = 1, \ldots, D$

S3. Eigenvalue/vector. Compute the M-largest eigenvalues and the eigenvectors of the data covariance matrix (M is the dimension that needs to be reduced)

S4. Projection. Project all data points onto the space defined by the eigenvectors (i.e., principal subspace).

S5. Undo standardization and centering.

Roadmap

(1) Problem Setting

(2) Maximum Variance Perspective

(3) Projection Perspective

(4) Eigenvector Computation and Low-Rank Approximations

(5) PCA in High Dimensions

(6) Key Steps of PCA in Practice

(7) Latent Variable Perspective

Generative Modeling with Latent Variables

Please go back to L8(4) for the background on generative models via latent variable models (LVMs).

Probabilistic PCA: Linear Latent Models

- $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
- A linear relationship between $\boldsymbol{z}$ and $\boldsymbol{x}$: For Guassian observation noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ and affine mapping defined by $\boldsymbol{B} \in \mathbb{R}^{D \times M}$ and $\boldsymbol{\mu} \in \mathbb{R}^D$,

$$\boldsymbol{x} = \boldsymbol{B}\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \in \mathbb{R}^D$$

- Conditional distribution for the links between latent and observed variables

$$\boxed{p(\boldsymbol{x} | \boldsymbol{z}, \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) = \mathcal{N}(\boldsymbol{x} | \boldsymbol{B}\boldsymbol{z} + \boldsymbol{\mu}, \sigma^2 \boldsymbol{I})}$$

- Data point generation: ancestral sampling
  - First, sample $\boldsymbol{z}_n$ from $p(\boldsymbol{z})$
  - Then, use $\boldsymbol{z}_n$ to generate a sample $\boldsymbol{x}_n \sim p(\boldsymbol{x} | \boldsymbol{z}_n, \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2)$

Probabilistic Model and Likelihood

- Probabilistic model: joint distribution

$$p(\boldsymbol{x}, \boldsymbol{z} | \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) = p(\boldsymbol{x} | \boldsymbol{z}, \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) p(\boldsymbol{z})$$

- Likelihood

$$\begin{aligned} p(\boldsymbol{x} | \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) &= \int p(\boldsymbol{x} | \boldsymbol{z}, \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) p(\boldsymbol{z}) \mathrm{d}\boldsymbol{z} = \int \mathcal{N}(\boldsymbol{x} | \boldsymbol{B}\boldsymbol{z} + \boldsymbol{\mu}, \sigma^2 \boldsymbol{I}) \mathcal{N}(\boldsymbol{z} | 0, \boldsymbol{I}) \mathrm{d}\boldsymbol{z} \\ &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{B}\boldsymbol{B}^\mathsf{T} + \sigma^2 \boldsymbol{I}) \end{aligned}$$

  - Using the property of marginal and conditional Gaussians L6(5)

Posterior Distribution

- The joint Gaussian distribution $p(\boldsymbol{x}, \boldsymbol{z} | \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2)$ leads us to the posterior distribution

$$p(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{m}, \boldsymbol{C}), \text{ where}$$

$$\boldsymbol{m} = \boldsymbol{B}^\mathsf{T}(\boldsymbol{B}\boldsymbol{B}^\mathsf{T} + \sigma^2 \boldsymbol{I})^{-1}(\boldsymbol{x} - \boldsymbol{\mu}), \; \boldsymbol{C} = \boldsymbol{I} - \boldsymbol{B}^\mathsf{T}(\boldsymbol{B}\boldsymbol{B}^\mathsf{T} + \sigma^2 \boldsymbol{I})^{-1}\boldsymbol{B}$$

Learning Probabilistic PCA: MLE

- For data samples $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$, we are able to compute the likelihood as:

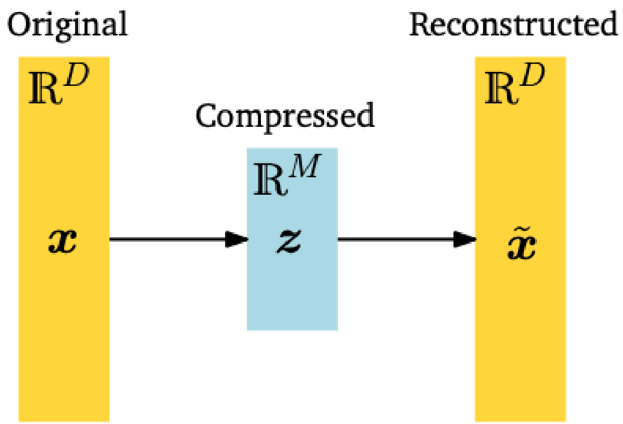
$$\log p(\boldsymbol{X} | \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2) = \sum_{n=1}^{N} \log p(\boldsymbol{x}_n | \boldsymbol{B}, \boldsymbol{\mu}, \sigma^2)$$

$$\boldsymbol{\mu}_{\text{ML}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n, \quad \boldsymbol{B}_{\text{ML}} = \boldsymbol{U}(\boldsymbol{\Lambda} - \sigma^2 \boldsymbol{I})^{1/2} \boldsymbol{R}, \quad \sigma_{\text{ML}} = \frac{1}{D-M} \sum_{j=M+1}^{D} \lambda_j, \quad \text{where}$$

  - $\boldsymbol{U}$ is a $D \times M$ matrix whose columns are eigenvectors of $\boldsymbol{S}$
  - $\boldsymbol{\Lambda}$ is a $M \times M$ diagonal matrix whose elements are eigenvalues of $\boldsymbol{S}$
  - $\boldsymbol{R}$ is an arbitrary orthogonal matrix (i.e., rotation)

- In the noise-free limit where $\sigma \to 0$, PPCA and PCA provide the identical solution.

PCA as Linear Auto-Encoder



- Non-linear auto-encoder: we replace the linear mapping of PCA with a non-linear mapping. An example is a deep auto-encoder with deep neutral networks.
- (Fully) Bayesian PCA: place a prior on the model parameters and integrate them out, rather than having a point estimate.
- Factor analysis: allow each observation dimension d to have a different variance σ_d^2

Questions?

References

[1] This lecture slide is mainly based upon `https://yung-web.github.io/home/courses/mathml.html` (made by Prof. Yung Yi, KAIST EE)

[2] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. Mathematics for machine learning. Cambridge University Press, 2020.