Cognitive Algorithms - Exercise 5

Unsupervised Learning

Daniel Wujecki

December 20, 2020

Organizational Remarks

- Grading of exercise 2 is not finished
  - Many submissions
  - Hopefully done within the next days
- Teaching evaluation in next week, also for elective courses
- please send mails **ONLY** from your **TU-Mail**

Today's Tutorial

- Due to your feedback no webcam - focus on the content ;)
- Repetition of theoretical concepts
  - Proofs are shown in the lecture videos

Today's Tutorial

- Due to your feedback no webcam - focus on the content ;)
- Repetition of theoretical concepts
  - Proofs are shown in the lecture videos
- If to many repetitions for you, feel free to skip!
  - Since you can skip, I am quite detailed for those who have more difficulties
- Many practical examples to visualize the concepts
  - Code will note be provided (contains Quiz solutions)
  - You do not need to understand the code I show

1. Task 1 + 4 - Eigenvalues and Eigenvectors
2. Task 2 + 3 + 4 - Principal Component Analysis
3. Task 5 - Non-negative matrix factorization
4. K-Means Clustering

Repetition: Eigenvalues and Eigenvectors

Given a Matrix $A \in \mathbb{R}^{d \times d}$, then a non-zero vector $\mathbf{v} \in \mathbb{C}^d \setminus \{\mathbf{0}\}$ is called an eigenvector, if there is an eigenvalue $\lambda \in \mathbb{C}$, such that

$$A\mathbf{v} = \lambda\mathbf{v}$$

Repetition: Eigenvalues and Eigenvectors

Given a Matrix $A \in \mathbb{R}^{d \times d}$, then a non-zero vector $\mathbf{v} \in \mathbb{C}^d \setminus \{\mathbf{0}\}$ is called an eigenvector, if there is an eigenvalue $\lambda \in \mathbb{C}$, such that

$$A\mathbf{v} = \lambda \mathbf{v}$$

- eigenvectors are special directions, for which a matrix A is only scales, but not rotate
- A matrix is singular if one or more eigenvalues are zero

$$\lambda = 0 \Leftrightarrow \mathrm{Kern}(A) \neq \emptyset$$

Repetition: Scaled Eigenvectors

If $\mathbf{v} \in \mathbb{R}^d$ is a eigenvector of $A \in \mathbb{R}^{d \times d}$, then also $\alpha\mathbf{v}$, $\alpha \in \mathbb{R}$, $\alpha \neq 0$ is a eigenvector (Task 4.1).

Proof

$$A(\alpha\mathbf{v}) = \alpha(A\mathbf{v}) = \alpha(\lambda\mathbf{v}) = \lambda(\alpha\mathbf{v})$$

where we used the homogeneity of linear mappings.

Repetition: Eigenvalues for special matrices

upper and lower triangular matrices

$$\begin{pmatrix} \lambda_1 & a_{1,2} & \cdots & a_{1,n} \\ & \lambda_2 & \cdots & a_{2,n} \\ & & \ddots & \vdots \\ 0 & & & \lambda_n \end{pmatrix}$$

diagonal matrices

$$\begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}$$

$\Rightarrow$ eigenvalues are on the diagonal (Task 1.2, 1.3)

Rotation Matrices I

What about rotation matrices?

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Rotation Matrices I

What about rotation matrices?

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

- They rotate vectors by an angle θ, but do not scale any vector
  $\Rightarrow$ no real eigenvalues, but complex eigenvalues (Task 1.4)

Rotation Matrices I

What about rotation matrices?

$$R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

- They rotate vectors by an angle θ, but do not scale any vector
  $\Rightarrow$ no real eigenvalues, but complex eigenvalues (Task 1.4)

- If $\theta = k\pi$, $k \in \mathbb{Z}$ they scale by $\lambda = \pm 1$
  - special case with real eigenvalues (Task 1.5)

Rotation Matrices II

- Rotation matrices R_θ are always orthogonal with $\det|R_\theta| = +1$
- Are all orthogonal matrices also rotation matrices? (Task 4.2)

Rotation Matrices II

- Rotation matrices R_θ are always orthogonal with $\det|R_\theta| = +1$
- Are all orthogonal matrices also rotation matrices? (Task 4.2)
- For a orthogonal matrix V the determinant is $\det|V| = \pm 1$

$$V = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

- V is orthogonal, since $V^\top V = I$, but $\det|V| = -1$ and thus it is not a rotation matrix

Repetition: Eigendecomposition

- If a matrix $A \in \mathbb{R}^{d \times d}$ is symmetric
  - all eigenvalues $\lambda_1, \ldots \lambda_d$ are real
  - all eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_d$ are orthogonal to each other (Task 1.1)

Repetition: Eigendecomposition

- If a matrix $A \in \mathbb{R}^{d \times d}$ is symmetric
  - all eigenvalues $\lambda_1, \dots \lambda_d$ are real
  - all eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_d$ are orthogonal to each other (Task 1.1)
- For symmetric matrices there is a decomposition $A = U \Lambda U^T$ into
  - a diagonal matrix Λ with the eigenvalues on the diagonal
  - a orthogonal matrix U with the eigenvectors in the columns

Dimensionality of Data

- Given some high dimensional data $X = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$

- Naturally, not all dimensions contain the same amount of information
- Extrem Example: if $d \gg n$, samples can not span d-dimensional vector space, even if all samples are linearly independent

Dimensionality of Data

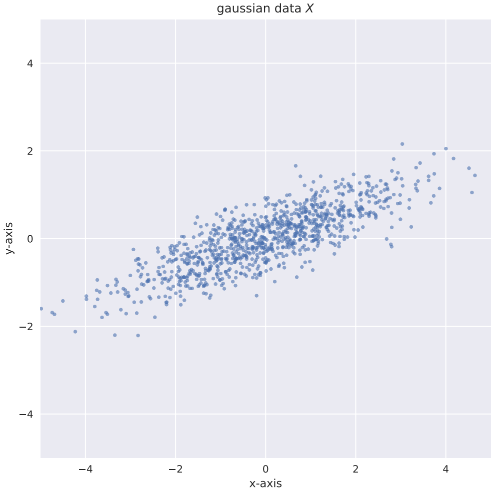
- Given some high dimensional data $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$
- Naturally, not all dimensions contain the same amount of information
- Extrem Example: if $d \gg n$, samples can not span d-dimensional vector space, even if all samples are linearly independent

Dimensionality of Data

- Given some high dimensional data $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$
- Naturally, not all dimensions contain the same amount of information
- Extrem Example: if $d \gg n$, samples can not span d-dimensional vector space, even if all samples are linearly independent

- How do we measure the ‘amount of information’?

Principle Component Analysis



- Which dimensions contain the most information?

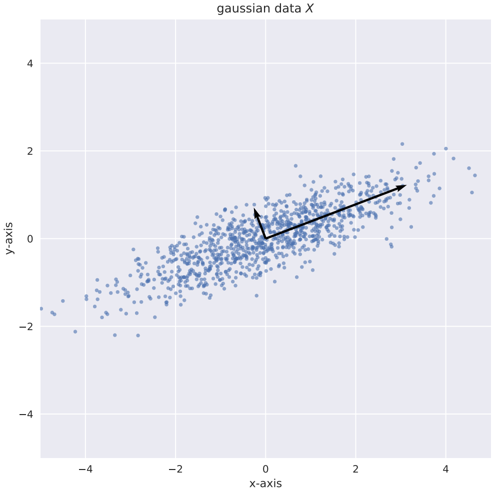
$\Rightarrow$ directions $\mathbf{w}$ with the highest variance

$$\underset{\mathbf{w}}{\operatorname{argmax}}\, \operatorname{var}(\mathbf{w}^\top X) = \underset{\mathbf{w}}{\operatorname{argmax}}\, \mathbf{w}^\top \Sigma_X \mathbf{w}$$

$\Rightarrow$ directions $\mathbf{w}$ that minimizes the noise

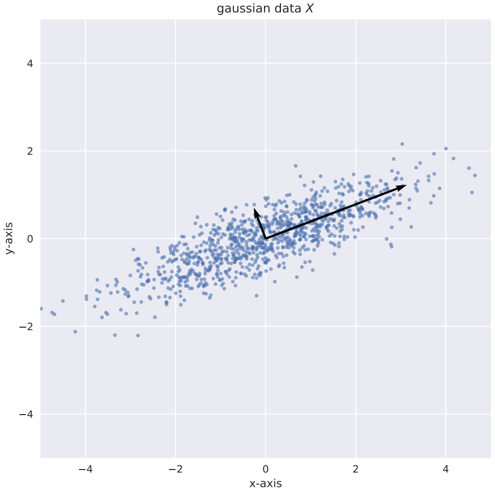
$$\underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} ||(\mathbf{w}^\top \mathbf{x}_i)\mathbf{w} - \mathbf{x}_i||^2 = \underset{\mathbf{w}}{\operatorname{argmax}}\, \mathbf{w}^\top \Sigma_X \mathbf{w}$$

Principle Component Analysis



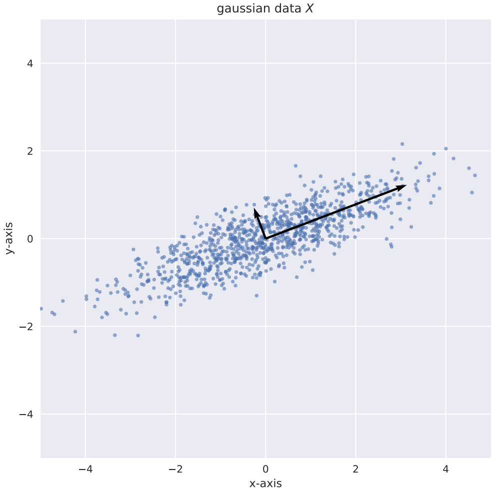
- $\mathbf{w}^T \Sigma_X \mathbf{w}$ is maximized by the eigenvector of Σ_X with the largest eigenvalue
- The eigenvectors of Σ_X are the principle components

Principle Component Analysis



- $\mathbf{w}^T \Sigma_X \mathbf{w}$ is maximized by the eigenvector of Σ_X with the largest eigenvalue
- The eigenvectors of Σ_X are the principle components
- PCs can be constrained to unit length $||\mathbf{w}|| = 1$

Principle Component Analysis



- $\mathbf{w}^T\Sigma_X\mathbf{w}$ is maximized by the eigenvector of Σ_X with the largest eigenvalue
- The eigenvectors of Σ_X are the principle components
- PCs can be constrained to unit length $||\mathbf{w}|| = 1$
- Variance along PC is given by the corresponding eigenvalue, since

$$\mathrm{var}(\mathbf{w}^\top X) = \mathbf{w}^T\Sigma_X\mathbf{w} = \mathbf{w}^T\lambda\mathbf{w} = \lambda\mathbf{w}^T\mathbf{w} = \lambda$$

Dimensionality Reduction with PCA - 3D example

- Given a 3D dataset $X \in \mathbb{R}^{3 \times n}$
- Calculate the EVD of the covariance $\Sigma_X = W \Lambda W^T$
- What happens when we map the data onto the PC?

Dimensionality Reduction with PCA - 3D example

- Given a 3D dataset $X \in \mathbb{R}^{3\times n}$
- Calculate the EVD of the covariance $\Sigma_X = W \Lambda W^T$

- What happens when we map the data onto the PC?

$$W^\top \mathbf{x} = \begin{bmatrix} \mathbf{w}_1^\top \\ \mathbf{w}_2^\top \\ \mathbf{w}_3^\top \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x} \\ \mathbf{w}_2^\top \mathbf{x} \\ \mathbf{w}_k^\top \mathbf{x} \end{bmatrix} = \mathbf{x}' = \begin{bmatrix} x_x' \\ x_y' \\ x_z' \end{bmatrix}$$

- Data is decorrelated: PCs become aligned with the coordinate axes
  - PCs are orthogonal to each other
  - orthogonal matrix W corresponds to a rotation

Dimensionality Reduction with PCA - 3D example

- We could drop now the dimension with lowest variance
- For the decorrelated data that is just dropping one coordinate axis

$$\begin{bmatrix} x'_x \\ x'_y \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x} \\ \mathbf{w}_2^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^2$$

Dimensionality Reduction with PCA - 3D example

- We could drop now the dimension with lowest variance
- For the decorrelated data that is just dropping one coordinate axis

$$\begin{bmatrix} x'_x \\ x'_y \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \mathbf{x} \\ \mathbf{w}_2^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^2$$

- To restore the same information we have to map the data back

$$[\mathbf{w}_1, \mathbf{w}_2] \begin{pmatrix} x'_x \\ x'_y \end{pmatrix} = x'_x \cdot \mathbf{w}_1 + x'_y \cdot \mathbf{w}_2 \approx \begin{bmatrix} x_x \\ x_y \\ x_z \end{bmatrix}$$

PCA Algorithm

Algorithm 1: Principal Component Analysis

Input: Dataset $X \in \mathbb{R}^{d \times n}$; Number of PCs k
Output: First k PCs $W \in \mathbb{R}^{d \times k}$, Hidden causes $H \in \mathbb{R}^{k \times n}$ of the dataset

Compute covariance $\Sigma_X = \frac{1}{n}(X - \overline{X})(X - \overline{X})^\top$
Compute EVD $\Sigma_X = V \Lambda V^\top$
Take first k eigenvectors corresponding to largest eigenvalues $W = [\mathbf{v}_1, \dots, \mathbf{v}_k]$
Project data $H = W^\top X$
return W *and* H

Task 2.1 - PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace

Task 2.1 - PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace
  1. $\overline{X} = \mathbf{0}$

Task 2.1 - PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace
  1. $\overline{X} = \mathbf{0}$
  2. $\Sigma_X = \frac{1}{2} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Task 2.1 - PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace
  1. $\overline{X} = \mathbf{0}$
  2. $\Sigma_X = \frac{1}{2} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
  3. $\mathbf{w}^T X = [1, 0, 0] \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = [-1, 1]$

Task 2.1 - PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace
  1. $\overline{X} = \mathbf{0}$
  2. $\Sigma_X = \frac{1}{2} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
  3. $\mathbf{w}^T X = [1, 0, 0] \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = [-1, 1]$
- Linearly dependent samples $\Rightarrow$ Representation makes sense

Kernel PCA

- If dimensionality becomes larger than number of samples $d \gg n$
  - Only $\leq n$ non-zero eigenvalues
  - Covariance Σ_X becomes singular, $\text{Rank}(\Sigma_X) \leq n$
  - Covariance is positive *semi* definite
  - Covariance $\Sigma_X \in \mathbb{R}^{d \times d}$ will be very large
  - High time and space complexity for calculation

- Kernel PCA should be used to estimate n first Principal Components

Kernel PCA Algorithm

Algorithm 2: Kernel PCA

Input: Dataset $X \in \mathbb{R}^{d \times n}$ with $d \gg n$; Number of PCs k
Output: First k PCs $W \in \mathbb{R}^{d \times k}$, Hidden causes $H \in \mathbb{R}^{k \times n}$ of the dataset

Compute Kernel $K = (X - \overline{X})^\top (X - \overline{X}) \in \mathbb{R}^{n \times n}$
Compute EVD $K = V \Lambda V^\top$
Take first k eigenvectors corresponding to largest eigenvalues $\alpha = [\mathbf{v}_1, \ldots, \mathbf{v}_k]$
Calculate $W = X\alpha \in \mathbb{R}^{d \times k}$
Project data $H = W^\top X$
return W *and* H

Task 2.2 - Kernel PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace using Kernel PCA

Task 2.2 - Kernel PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace using Kernel PCA
    1. $\overline{X} = \mathbf{0}$

Task 2.2 - Kernel PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace using Kernel PCA
  1. $\overline{X} = \mathbf{0}$
  2. $K = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}$

Task 2.2 - Kernel PCA Algorithm

- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace using Kernel PCA
  1. $\overline{X} = \mathbf{0}$
  2. $K = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}$
  3. $\mathbf{w} = X\alpha = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \frac{\mathbf{w}}{||\mathbf{w}||} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Task 2.2 - Kernel PCA Algorithm

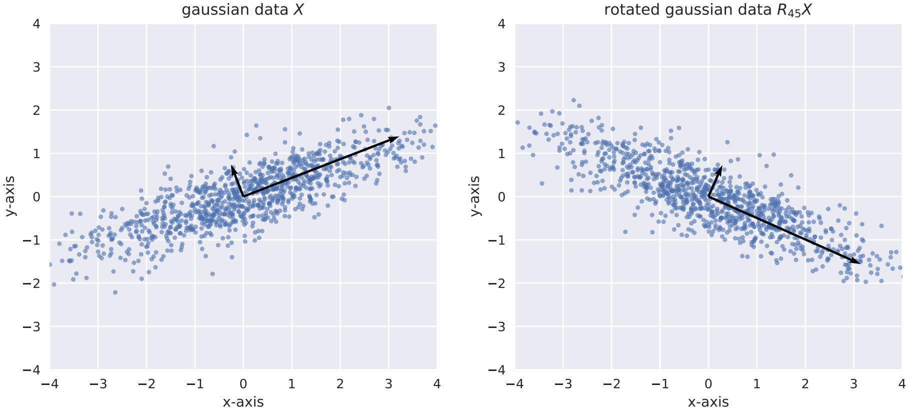
- Consider a dataset with two samples $X = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
- Project the data onto a 1-dimensional subspace using Kernel PCA
  1. $\overline{X} = \mathbf{0}$
  2. $K = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}$
  3. $\mathbf{w} = X\alpha = \begin{bmatrix} -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \frac{\mathbf{w}}{||\mathbf{w}||} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$
- Same result as before (but with smaller time and space complexity)

Task 4.3

- Consider a dataset $X \in \mathbb{R}^{d \times n}$ and a orthogonal rotation matrix $U \in \mathbb{R}^{d \times d}$
- Proof that the covariance Σ_X and the covariance Σ_{UX} of the rotated data have the same eigenvalues and the eigenvectors are also rotated by U

Task 4.3

- Consider a dataset $X \in \mathbb{R}^{d \times n}$ and a orthogonal rotation matrix $U \in \mathbb{R}^{d \times d}$
- Proof that the covariance Σ_X and the covariance Σ_{UX} of the rotated data have the same eigenvalues and the eigenvectors are also rotated by U



Task 4.3

- Consider a dataset $X \in \mathbb{R}^{d \times n}$ and a orthogonal rotation matrix $U \in \mathbb{R}^{d \times d}$
- Proof that the covariance Σ_X and the covariance Σ_{UX} of the rotated data have the same eigenvalues and the eigenvectors are also rotated by U
- We have

$$\Sigma_{UX} = \frac{1}{N} UX(UX)^\top = U \frac{1}{N} XX^\top U^\top = U \Sigma_X U^\top$$

Task 4.3

- Consider a dataset $X \in \mathbb{R}^{d \times n}$ and a orthogonal rotation matrix $U \in \mathbb{R}^{d \times d}$
- Proof that the covariance Σ_X and the covariance Σ_{UX} of the rotated data have the same eigenvalues and the eigenvectors are also rotated by U
- We have

$$\Sigma_{UX} = \frac{1}{N} UX(UX)^\top = U \frac{1}{N} XX^\top U^\top = U \Sigma_X U^\top$$

Thus

$$\Sigma_X \mathbf{v} = \lambda \mathbf{v} \Leftrightarrow U \Sigma_X \mathbf{v} = \lambda U \mathbf{v} \Leftrightarrow \underbrace{U \Sigma_X U^\top}_{\Sigma_{UX}} \underbrace{U \mathbf{v}}_{:=\mathbf{z}} = \lambda \underbrace{U \mathbf{v}}_{=\mathbf{z}}$$

Task 4.3

- Consider a dataset $X \in \mathbb{R}^{d \times n}$ and a orthogonal rotation matrix $U \in \mathbb{R}^{d \times d}$
- Proof that the covariance Σ_X and the covariance Σ_{UX} of the rotated data have the same eigenvalues and the eigenvectors are also rotated by U
- We have

$$\Sigma_{UX} = \frac{1}{N} UX(UX)^\top = U \frac{1}{N} XX^\top U^\top = U \Sigma_X U^\top$$

Thus

$$\Sigma_X \mathbf{v} = \lambda \mathbf{v} \Leftrightarrow U \Sigma_X \mathbf{v} = \lambda U \mathbf{v} \Leftrightarrow \underbrace{U \Sigma_X U^\top}_{\Sigma_{UX}} \underbrace{U \mathbf{v}}_{:= \mathbf{z}} = \lambda \underbrace{U \mathbf{v}}_{= \mathbf{z}}$$

Thus, if $\mathbf{v}$ is a eigenvector of Σ_X with corresponding eigenvalue λ, $U\mathbf{v}$ is eigenvector of Σ_{UX} with eigenvalue λ.

Task 3

Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

- Reminder: Variance along PC is given by the corresponding eigenvalue, since

$$\mathrm{var}(\mathbf{w}^\top X) = \mathbf{w}^T \Sigma_X \mathbf{w} = \mathbf{w}^T \lambda \mathbf{w} = \lambda \mathbf{w}^T \mathbf{w} = \lambda$$

Task 3.1 - Variances along PCs

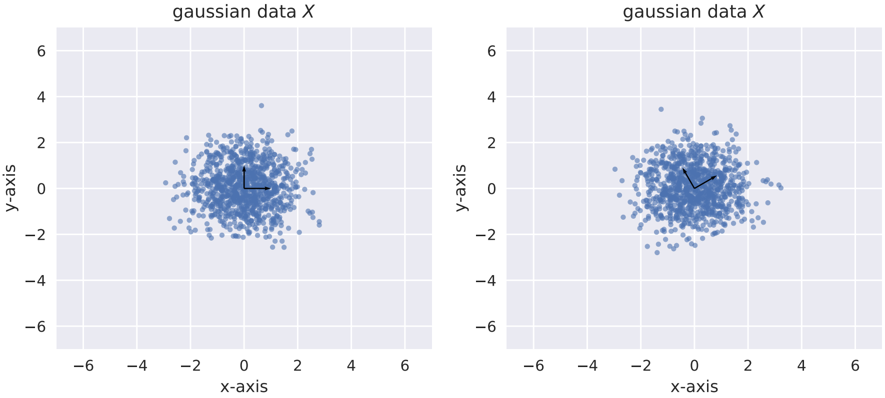
Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

$$\lambda_1 = 1$$
$$\lambda_2 = 1$$

Task 3.1 - Variances along PCs

Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

$$\lambda_1 = 1$$
$$\lambda_2 = 1$$



Task 3.2 - Variances along PCs

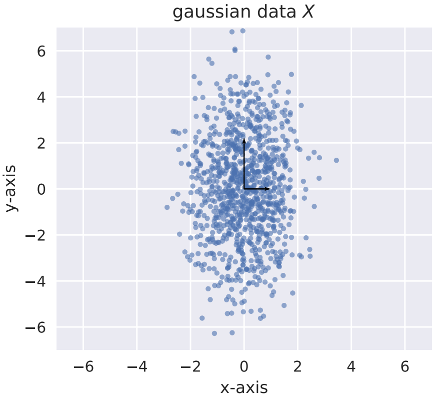
Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

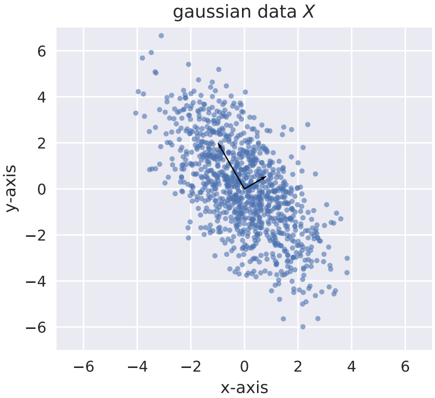
$$\lambda_1 = 1$$
$$\lambda_2 = 5$$

Task 3.2 - Variances along PCs

Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

$$\lambda_1 = 1$$
$$\lambda_2 = 5$$





Task 3.3 - Variances along PCs

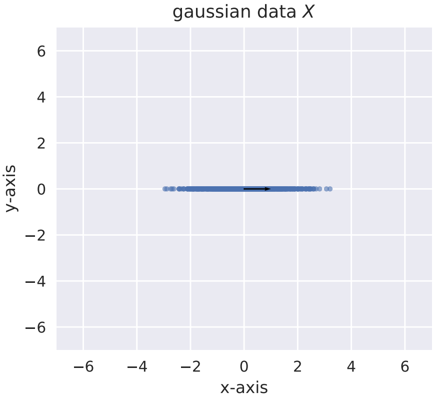
Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

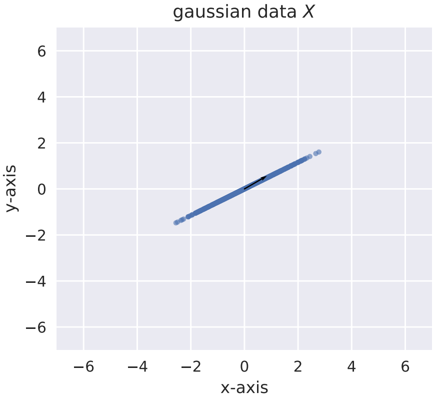
$$\lambda_1 = 1$$
$$\lambda_2 = 0$$

Task 3.3 - Variances along PCs

Given two eigenvalues of a covariance matrix, sketch two corresponding 2-dimensional datasets with uncorrelated and correlated features, respectively.

$$\lambda_1 = 1$$
$$\lambda_2 = 0$$





Repetition: Non-negative matrix factorization

- For some data PCA is not intuitive
- Example: Non-negative data
  - Principal directions will have negative entries
  - This can be hard to interpret
- Many datasets are strictly positive
  - Text data
  - Image data
  - Probabilistic data
- Very easy to implement (as the PCA algorithm)

From PCA to NMF

- Given data $X \in \mathbb{R}^{d \times n}$ and PCs in the columns of a matrix $W \in \mathbb{R}^{d \times k}$
- Data can be projected onto PCs $H = W^\top X \Leftrightarrow X \approx WH$

From PCA to NMF

- Given data $X \in \mathbb{R}^{d \times n}$ and PCs in the columns of a matrix $W \in \mathbb{R}^{d \times k}$
- Data can be projected onto PCs $H = W^\top X \Leftrightarrow X \approx WH$

- If $X > 0$, it should be possible to choose W, H such that $W > 0, H > 0$

From PCA to NMF

- Given data $X \in \mathbb{R}^{d \times n}$ and PCs in the columns of a matrix $W \in \mathbb{R}^{d \times k}$
- Data can be projected onto PCs $H = W^\top X \Leftrightarrow X \approx WH$

- If $X > 0$, it should be possible to choose W, H such that $W > 0, H > 0$

- Goal of NMF is to find $W \in \mathbb{R}^{d \times k}$ and $H \in \mathbb{R}^{k \times n}$, sucht that $||X - WH||^2_{\text{Fro}}$ is minimized

From PCA to NMF

- Given data $X \in \mathbb{R}^{d \times n}$ and PCs in the columns of a matrix $W \in \mathbb{R}^{d \times k}$
- Data can be projected onto PCs $H = W^\top X \Leftrightarrow X \approx WH$
- If $X > 0$, it should be possible to choose W, H such that $W > 0, H > 0$
- Goal of NMF is to find $W \in \mathbb{R}^{d \times k}$ and $H \in \mathbb{R}^{k \times n}$, sucht that $||X - WH||^2_{\text{Fro}}$ is minimized
- Cannot be directly calculated $\Rightarrow$ iterative optimization with gradient descent

NMF Algorithm

Algorithm 3: Non-negative Matrix Factorization

Input: Dataset $X \in \mathbb{R}_+^{d \times n}$; Dimensionality k

Output: $W \in \mathbb{R}_+^{d \times k}$, Hidden causes $H \in \mathbb{R}_+^{k \times n}$ of the dataset

1 Initialize $W \in \mathbb{R}_+^{d \times k}$ and $H \in \mathbb{R}_+^{k \times n}$ randomly

2 Add a small constant $\sigma = 10^{-19}$ to avoid zero-divisions

3 **for** *it* $\leq$ *iterations* **do**

4 $\quad H \leftarrow H + \eta \left(W^\top WH - X^\top W \right)$

5 $\quad W \leftarrow W + \eta \left(WHH^\top - XH^\top \right)$

6 **return** W *and* H

Task 5 - NMF (actually simple linear algebra)

- NMF applied to a dataset $X \in \mathbb{R}^{4\times 3}$
- After training the reconstruction $\tilde{X} = WH$ looks like this

$$\tilde{X} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \\ 0 & 0 & 2 \\ 0 & 1 & 1 \end{pmatrix}, \qquad H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- What is W?

Task 5 - NMF (actually simple linear algebra)

- NMF applied to a dataset $X \in \mathbb{R}^{4\times 3}$
- After training the reconstruction $\tilde{X} = WH$ looks like this

$$\tilde{X} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 2 \\ 0 & 0 & 2 \\ 0 & 1 & 1 \end{pmatrix}, \qquad H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- What is W?

- $\tilde{X} = WH = W$

K-Means Clustering

- Theory is completely covered by the lecture, but there are also Tutorials on youtube
- Explanation of the concept with the help of the solution of the quiz

$\Rightarrow$ Jupyter Notebook