Dimensionality Reduction

Makoto Yamada

myamada@i.kyoto-u.ac.jp

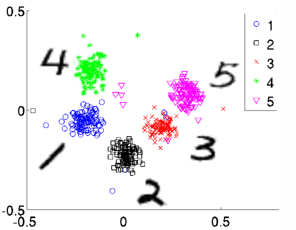
Kyoto University

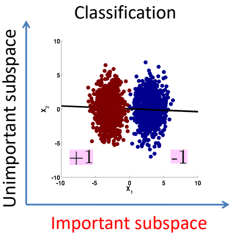
June/29/2020

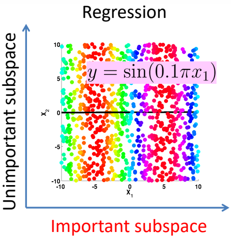
Dimensionality Reduction

Dimensionality reduction is a method to reduce the dimensionality of data.

- Feature selection is a dimensionality reduction method. Select a set of m features among d features ($m < d$).
- We use feature selection for interpretation.
- We use dimensionality reduction to compress data, to visualize data, etc.



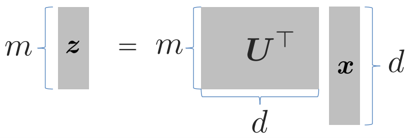




Problem Formulation

Dimension reduction (DR) is to find a low-dimensional mapping $f : \mathbb{R}^d \to \mathbb{R}^m$ $(d > m)$ $(\boldsymbol{x} \in \mathbb{R}^d)$

- It is useful for data visualization.
- Keep the original information as much as possible
- The DR outputs the combination of features.
- Linear dimension reduction $\boldsymbol{z} = \boldsymbol{U}^\top \boldsymbol{x}$ $(\boldsymbol{U} \in \mathbb{R}^{d \times m})$.



- Nonlinear dimension reduction $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{x})$. For example, deep learning model: $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{\sigma}(\boldsymbol{W}_1(\boldsymbol{\sigma}(\boldsymbol{W}_2)))$

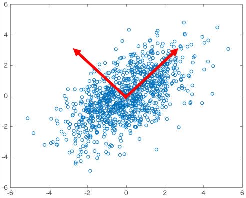
Principal Component Analysis

The principal component analysis (PCA) is given as:

$$\widehat{\boldsymbol{U}} = \underset{\boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{I}.}{\text{argmax}} \quad \text{tr}(\boldsymbol{U}^\top \boldsymbol{R} \boldsymbol{U}),$$

where $\boldsymbol{R} = \frac{1}{n} \sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{x}_i^\top \in \mathbb{R}^{d \times d}$ (we assume $\mathbb{E}[\boldsymbol{x}] = \boldsymbol{0}$) is the covariance matrix.

Find a direction that maximizes the variance. For 1d case i.e., $\boldsymbol{u} \in \mathbb{R}^d$, $\text{tr}(\boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u}) = \frac{1}{n} \sum_{i=1}^n (\boldsymbol{u}^\top \boldsymbol{x}_i)^2$ and $\mathbb{E}[\boldsymbol{u}^\top \boldsymbol{x}] = 0$.



Obtain the first principal component

To obtain the first principal component:

$$\underset{\boldsymbol{u}^\top \boldsymbol{u}=1.}{\operatorname{argmax}} \quad \boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u} = \underset{\boldsymbol{u}}{\operatorname{argmax}} \frac{\boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u}}{\|\boldsymbol{u}\|_2^2},$$

where $\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|_2}$ is a unit vector and $\frac{\boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u}}{\|\boldsymbol{u}\|_2^2}$ is called as the Rayleigh quotient.

Using the Lagrange multiplier λ to find a critical point:

$$L(\boldsymbol{u}) = \boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u} - \lambda(\boldsymbol{u}^\top \boldsymbol{u} - 1)$$

To take the derivative with respect to $\boldsymbol{u}$, we have

$$\frac{\partial L(\boldsymbol{u})}{\partial \boldsymbol{u}} = 2\boldsymbol{R}\boldsymbol{u} - 2\lambda \boldsymbol{u} = \boldsymbol{0} \rightarrow \boldsymbol{R}\boldsymbol{u} = \lambda \boldsymbol{u}.$$

This is an eigenvalue decomposition problem where λ is the eigenvalue and $\boldsymbol{u}$ is the eigenvector. Variance is $\boldsymbol{u}^\top \boldsymbol{R} \boldsymbol{u} = \lambda$.

PCA with eigenvalue decomposition

PCA can be solved by using eigenvalue decomposition of the covariance matrix $\boldsymbol{R}$!

The eigenvalue decomposition of covariance matrix $\boldsymbol{R} \in \mathbb{R}^{d \times d}$:

$$\boldsymbol{R} = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^\top \text{ or } \boldsymbol{U}^\top \boldsymbol{R} \boldsymbol{U} = \boldsymbol{\Lambda}$$

where

- $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d) \in \mathbb{R}^{d \times d}$, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$. If $\boldsymbol{R}$ is a positive definite matrix $\lambda_d \geq 0$.
- $\boldsymbol{U} \in \mathbb{R}^{d \times d}$ is an orthogonal matrix $\boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{U} \boldsymbol{U}^\top = \boldsymbol{I}_d$
- $\text{tr}(\boldsymbol{U}^\top \boldsymbol{R} \boldsymbol{U}) = \text{tr}(\boldsymbol{U}^\top \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^\top \boldsymbol{U}) = \sum_{i=1}^d \lambda_i$.

Relationship to Linear Auto-encoder (1/2)

Assume that $\mathbb{E}[\boldsymbol{x}] = \mathbf{0}$. Then, consider the following linear Auto-encoder problem:

$$\widehat{\boldsymbol{U}} = \underset{\boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{I}}{\text{argmin}} \quad \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x}_i\|_2^2,$$

The loss function term can be written as

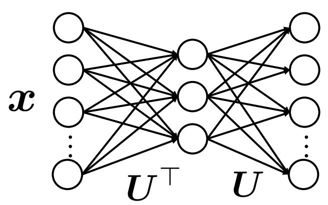
$$\begin{aligned}
\frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x}_i\|_2^2 &= \frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{x}_i - 2\boldsymbol{x}_i^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x}_i + \boldsymbol{x}_i^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x}_i\right) \\
&\propto -\frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{x}_i^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x}_i\right) \quad (\boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{I}) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \left(\mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{U})\right) \quad (\mathrm{tr}(\boldsymbol{A}\boldsymbol{B}) = \mathrm{tr}(\boldsymbol{B}\boldsymbol{A})) \\
&= -\mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{R} \boldsymbol{U}), \quad (\boldsymbol{R} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top)
\end{aligned}$$

Relationship to Linear Auto-encoder (2/2)

The minimization problem can be written as the maximization problem:

$$\underset{\boldsymbol{U}^\top\boldsymbol{U}=\boldsymbol{I}}{\text{argmin}} \quad \frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{x}_i - \boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{x}_i\|_2^2, \leftrightarrow \quad \underset{\boldsymbol{U}^\top\boldsymbol{U}=\boldsymbol{I}}{\text{argmax}} \quad \text{tr}(\boldsymbol{U}^\top\boldsymbol{R}\boldsymbol{U})$$

Thus, PCA is related to the linear Auto-encoder.

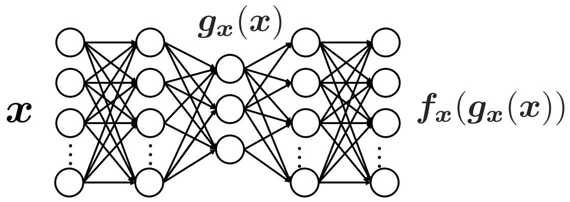


Nonlinear Auto-encoder

We consider the following Auto-encoder problem:

$$\widehat{\boldsymbol{\Theta}} = \underset{\boldsymbol{\Theta}}{\text{argmin}} \quad \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}_i))\|_2^2,$$

The nonlinear auto-encoder can be illustrated as



Stochastic Neighbor Embedding (SNE)

The asymmetric probability p_{ij} that i-th sample would pick j-th sample as its neighbor:

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ki}^2)}, \quad d_{ij}^2 = \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma_i^2},$$

where σ_i is a tuning parameter.

The model:

$$q_{ij} = \frac{\exp(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2)}{\sum_{k \neq i} \exp(-\|\boldsymbol{y}_k - \boldsymbol{y}_i\|_2^2)}$$

Optimization:

$$\widehat{\boldsymbol{y}}_1, \ldots, \widehat{\boldsymbol{y}}_n = \underset{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n}{\operatorname{argmin}} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Symmetric SNE

The symmetric probability p_{ij} that i-th sample would pick j-th sample as its neighbor:

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq l} \exp(-d_{kl}^2)}, \quad d_{ij}^2 = \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma^2},$$

where σ is a tuning parameter.

The model:

$$q_{ij} = \frac{\exp(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2)}{\sum_{k \neq l} \exp(-\|\boldsymbol{y}_k - \boldsymbol{y}_l\|_2^2)}$$

Optimization:

$$\widehat{\boldsymbol{y}}_1, \ldots, \widehat{\boldsymbol{y}}_n = \operatorname*{argmin}_{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

t-Stochastic Neighbor Embedding (t-SNE)

The asymmetric probability p_{ij} that i-th sample would pick j-th sample as its neighbor:

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq l} \exp(-d_{ik}^2)}, \quad d_{ij}^2 = \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{2\sigma^2},$$

where σ is a tuning parameter.

The model (Cauchy distribution):

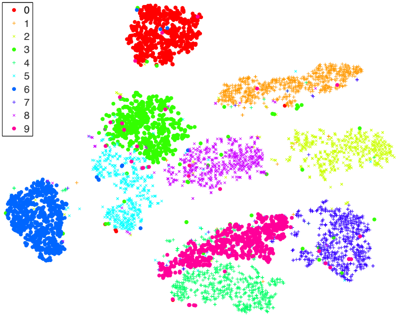
$$q_{ij} = \frac{(1 + \|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2)^{-1}}{\sum_{k \neq l}(1 + \|\boldsymbol{y}_k - \boldsymbol{y}_l\|_2^2)^{-1}}$$

Optimization:

$$\widehat{\boldsymbol{y}}_1, \ldots, \widehat{\boldsymbol{y}}_n = \underset{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n}{\operatorname{argmin}} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

t-SNE illustration

Image taken from [1]



(a) Visualization by t-SNE.

t-SNE is heavily used in biology data such as the expression data.

Multi-modal Dimensionality Reduction

PCA and auto-encoders are for uni-modal input (i.e., only image or only text).

How to do dimensionality reduction for multi-modal data (i.e., image and text)?

We have $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x} \in \mathbb{R}^{d_x}$ and $\boldsymbol{y} \in \mathbb{R}^{d_y}$.

- Linear dimension reduction $\boldsymbol{z}_x = \boldsymbol{U}^\top \boldsymbol{x}$ and $\boldsymbol{z}_y = \boldsymbol{V}^\top \boldsymbol{y}$. $\boldsymbol{U} \in \mathbb{R}^{d_x \times m}$ and $\boldsymbol{U} \in \mathbb{R}^{d_y \times m}$.
- Nonlinear dimension reduction $\boldsymbol{z}_x = \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x})$ and $\boldsymbol{z}_y = \boldsymbol{g}_{\boldsymbol{y}}(\boldsymbol{y})$.

Canonical Correlation Analysis (1/3)

Canonical Correlation Analysis (CCA) is to find dimensionality reduction that maximize the similarity between $\boldsymbol{z}_x = \boldsymbol{U}^\top \boldsymbol{x}$ and $\boldsymbol{z}_y = \boldsymbol{V}^\top \boldsymbol{y}$.

Assume that $\mathbb{E}[\boldsymbol{x}] = \boldsymbol{0}$ and $\mathbb{E}[\boldsymbol{y}] = \boldsymbol{0}$.

$$\mathrm{Corr}(X, Y) = \frac{\frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_{x,i}^\top \boldsymbol{z}_{y,i}}{\sqrt{\frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_{x,i}^\top \boldsymbol{z}_{x,i}}\sqrt{\frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_{y,i}^\top \boldsymbol{z}_{y,i}}}$$

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_{x,i}^\top \boldsymbol{z}_{y,i} &= \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^\top \boldsymbol{U}\boldsymbol{V}^\top \boldsymbol{y}_i \\ &= \mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{R}_{xy} \boldsymbol{V})\end{aligned}$$

where $\boldsymbol{R}_{xy} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i \boldsymbol{y}_i^\top \in \mathbb{R}^{d_x \times d_y}$.

Canonical Correlation Analysis (2/3)

The optimization problem of CCA is given as

$$\widehat{\boldsymbol{U}}, \widehat{\boldsymbol{V}} = \underset{\boldsymbol{U},\boldsymbol{V}}{\operatorname{argmax}} \quad \operatorname{tr}(\boldsymbol{U}^\top \boldsymbol{R}_{xy} \boldsymbol{V}),$$
$$\text{s.t.} \quad \boldsymbol{U}^\top \boldsymbol{R}_{xx} \boldsymbol{U} = \boldsymbol{I}, \boldsymbol{V}^\top \boldsymbol{R}_{yy} \boldsymbol{V} = \boldsymbol{I},$$

where $\boldsymbol{R}_{xx} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top$ and $\boldsymbol{R}_{yy} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{y}_i \boldsymbol{y}_i^\top$.

Then, CCA can be written as

$$\max_{\boldsymbol{U},\boldsymbol{V}} \quad \operatorname{tr}\left(\begin{bmatrix} \boldsymbol{U}^\top & \boldsymbol{V}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{O} & \boldsymbol{R}_{xy} \\ \boldsymbol{R}_{xy}^\top & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{V} \end{bmatrix} \right),$$
$$\text{s.t.} \quad \begin{bmatrix} \boldsymbol{U}^\top & \boldsymbol{V}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{xx} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_{yy} \end{bmatrix} \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{V} \end{bmatrix} = \boldsymbol{I},$$

This is a generalized eigenvalue decomposition (GEV) problem.

Canonical Correlation Analysis (3/3)

Let us transform the variables as

$$\begin{bmatrix} \bar{\boldsymbol{U}} \\ \bar{\boldsymbol{V}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{xx}^{1/2} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_{yy}^{1/2} \end{bmatrix} \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{V} \end{bmatrix}$$

we can rewrite the CCA optimization problem as

$$\begin{aligned} \max_{\bar{\boldsymbol{U}},\bar{\boldsymbol{V}}} \quad & \frac{1}{2}\mathrm{tr}\left(\begin{bmatrix} \bar{\boldsymbol{U}}^\top & \bar{\boldsymbol{V}}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{O} & \boldsymbol{R}_{xx}^{-1/2}\boldsymbol{R}_{xy}\boldsymbol{R}_{yy}^{-1/2} \\ (\boldsymbol{R}_{xx}^{-1/2}\boldsymbol{R}_{xy}\boldsymbol{R}_{yy}^{-1/2})^\top & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{U}} \\ \bar{\boldsymbol{V}} \end{bmatrix} \right), \\ \text{s.t.} \quad & \begin{bmatrix} \bar{\boldsymbol{U}}^\top & \bar{\boldsymbol{V}}^\top \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{U}} \\ \bar{\boldsymbol{V}} \end{bmatrix} = \boldsymbol{I}, \end{aligned}$$

Thus, we can solve the CCA problem by using eigenvalue decomposition!

Other dimensionality reduction methods

- Fisher Discriminant Analysis (FDA)
- Independent Component Analysis (ICA)
- Sufficient Dimensionality Reduction (SDR)
- Locally Linear Embedding (LLE)
- etc.

Summary of today's lecture

- Dimensionality reduction (Reduce the dimensionality of features).
- Feature selection is to interpret features, while dimensionality reduction is to reduce the dimensionality (for compression and visualization).
- Principal Component Analysis (PCA)
- Canonical Correlation Analysis (Multi-modal data)

[1] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.