2-D Canonical Correlation Analysis (2-D CCA)

Bhanupratap Baghel (0801CS183D04)
Lokesh Dohare (0801CS171038)
Sachi Parashar (0801CS183D14)

December 13, 2020

Contents

Chapter 1

Introduction

1.1 CCA

Canonical correlation analysis (CCA) is an important method in feature extraction and multivariate data analysis, which aims to find the correlation between two sets of variables. In order to extract the lower-dimensional and effective features, CCA seeks a pair of linear transformations such that the projected variables have the maximal correlation. Up to now, CCA has been applied in many fields, for instance, statistical analysis, information retrieval, facial expression recognition, genomic data analysis, image dehazing, closed-loop data identification, fingerprint recognition, palmprint recognition [9], and machine learning.

1.2 Need of 2-D CCA

For the defects of the 1D method, researchers have proposed some two-dimensional (2D) data analysis methods, 2D methods use images directly without reshaping them into vectors, thus they can achieve better performance and need less computing time. But their disadvantage is that the matrices used in two variable groups must have the same number of rows, because there exist trace operator in the objective functions.

In the conventional CCA, each image data is reshaped into a long vector, Here we present 2D-CCA where we directly use image data to determine relations between them.

1.3 (2D) square CCA

(2D) square CCA is a two directional two dimensional variant of CCA, it can reduce the computational load drastically. (2D)2mCCA extends (2D)2CCA to the multi-set case, and it can deal with the multi-view learning problem. In order to find the nonlinear correlation between two groups of images, L(2D)2CCA uses local spatial information to discover the essential manifold structure. However, the shortcoming of L(2D)2CCA is that it is sensitive to the nearest neighbor parameter *k* and different parameters may lead to different results.

Like CCA, when the two groups of samples are not linearly correlated, (2D)2CCA may fail to discover the intrinsic correlation of the data due to its substantial linearity.

Chapter 2

Mathematical Formulation

2.1 Formulation

Now we consider two sets of image data, $\{\boldsymbol{X}_t \in \mathbb{R}^{m_x \times n_x},\ t = 1, \ldots, N\}$ and $\{\boldsymbol{Y}_t \in \mathbb{R}^{m_y \times n_y},\ t = 1, \ldots, N\}$ that are realizations of random variable matrix $\boldsymbol{X}$ and $\boldsymbol{Y}$,

$$\boldsymbol{M}_x = \frac{1}{N}\sum_{t=1}^{N}\boldsymbol{X}_t, \qquad \boldsymbol{M}_y = \frac{1}{N}\sum_{t=1}^{N}\boldsymbol{Y}_t.$$

$$\widetilde{\boldsymbol{X}}_t = \boldsymbol{X}_t - \boldsymbol{M}_x, \qquad \widetilde{\boldsymbol{Y}}_t = \boldsymbol{Y}_t - \boldsymbol{M}_y.$$

l_x = left transform matrix of X, l_y = left transform matrix of Y

r_x = right transform matrix of X, r_y = right transform matrix of Y

$$\text{argmaxcov}(l_x^{T}Xr_x, l_y^{T}Yr_y)$$

such that

$$\operatorname{var}\left(\boldsymbol{l}_x^{\top}\boldsymbol{X}\boldsymbol{r}_x\right) = 1, \operatorname{var}\left(\boldsymbol{l}_y^{\top}\boldsymbol{Y}\boldsymbol{r}_y\right) = 1.$$

eq(1)

We define

$$\boldsymbol{\Sigma}_{xy}^{r} = \left\langle \widetilde{\boldsymbol{X}}\boldsymbol{r}_x\boldsymbol{r}_y^{\top}\widetilde{\boldsymbol{Y}}^{\top} \right\rangle = \frac{1}{N}\sum_{t=1}^{N}\widetilde{\boldsymbol{X}}_t\boldsymbol{r}_x\boldsymbol{r}_y^{\top}\widetilde{\boldsymbol{Y}}_t^{\top}$$

$$\boldsymbol{\Sigma}_{xx}^{r} = \left\langle \widetilde{\boldsymbol{X}}\boldsymbol{r}_x\boldsymbol{r}_x^{\top}\widetilde{\boldsymbol{X}}^{\top} \right\rangle = \frac{1}{N}\sum_{t=1}^{N}\widetilde{\boldsymbol{X}}_t\boldsymbol{r}_x\boldsymbol{r}_x^{\top}\widetilde{\boldsymbol{X}}_t^{\top}$$

$$\boldsymbol{\Sigma}_{yy}^{r} = \left\langle \widetilde{\boldsymbol{Y}}\boldsymbol{r}_y\boldsymbol{r}_y^{\top}\widetilde{\boldsymbol{Y}}^{\top} \right\rangle = \frac{1}{N}\sum_{t=1}^{N}\widetilde{\boldsymbol{Y}}_t\boldsymbol{r}_r\boldsymbol{r}_y^{\top}\widetilde{\boldsymbol{Y}}_t^{\top}.$$

$$\operatorname{cov}\left(\boldsymbol{l}_x^{\top}\boldsymbol{X}\boldsymbol{r}_x, \boldsymbol{l}_y^{\top}\boldsymbol{Y}\boldsymbol{r}_y\right) = \left\langle \boldsymbol{l}_x^{\top}\widetilde{\boldsymbol{X}}\boldsymbol{r}_x\boldsymbol{r}_y^{\top}\widetilde{\boldsymbol{Y}}^{\top}\boldsymbol{l}_y \right\rangle = \boldsymbol{l}_x^{\top}\boldsymbol{\Sigma}_{xy}^{r}\boldsymbol{l}_y.$$

$$\arg\max \boldsymbol{l}_x^\top \boldsymbol{\Sigma}_{xy}^r \boldsymbol{l}_y, \ \text{ s.t. } \boldsymbol{l}_x^\top \boldsymbol{\Sigma}_{xx}^r \boldsymbol{l}_x = 1, \ \boldsymbol{l}_y^\top \boldsymbol{\Sigma}_{yy}^r \boldsymbol{l}_y = 1. \qquad \text{eq(2)}$$

$$\text{cov}\left(\boldsymbol{l}_x^\top \boldsymbol{X} \boldsymbol{r}_x, \boldsymbol{l}_y^\top \boldsymbol{Y} \boldsymbol{r}_y\right) = \text{cov}\left(\boldsymbol{r}_x^\top \boldsymbol{X}^\top \boldsymbol{l}_x, \boldsymbol{r}_y^\top \boldsymbol{Y}^\top \boldsymbol{l}_y\right).$$

We can rewrite eq(1) as –

$$\arg\max \boldsymbol{r}_x^\top \boldsymbol{\Sigma}_{xy}^l \boldsymbol{r}_y, \ \text{ s.t. } \boldsymbol{r}_x^\top \boldsymbol{\Sigma}_{xx}^l \boldsymbol{r}_x = 1 \ \ \boldsymbol{r}_y^\top \boldsymbol{\Sigma}_{yy}^l \boldsymbol{r}_y = 1 \qquad \text{eq(3)}$$

where

$$\begin{aligned} \boldsymbol{\Sigma}_{xy}^l &= \left\langle \widetilde{\boldsymbol{X}}^\top \boldsymbol{l}_x \boldsymbol{l}_y^\top \widetilde{\boldsymbol{Y}} \right\rangle \\ \boldsymbol{\Sigma}_{xx}^l &= \left\langle \widetilde{\boldsymbol{X}}^\top \boldsymbol{l}_x \boldsymbol{l}_x^\top \widetilde{\boldsymbol{X}} \right\rangle \\ \boldsymbol{\Sigma}_{yy}^l &= \left\langle \widetilde{\boldsymbol{Y}}^\top \boldsymbol{l}_y \boldsymbol{l}_y^\top \widetilde{\boldsymbol{Y}} \right\rangle. \end{aligned}$$

The 2D-CCA determines l_x and l_y by solving eq(2) with r_x and r_y. The right transforms r_x and r_y are found by solving eq(3) with l_x and l_y.

$$\text{eq(4)} \qquad \begin{bmatrix} 0 & \boldsymbol{\Sigma}_{xy}^r \\ \boldsymbol{\Sigma}_{yx}^r & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{l}_x \\ \boldsymbol{l}_y \end{bmatrix} = \lambda \begin{bmatrix} \boldsymbol{\Sigma}_{xx}^r & 0 \\ 0 & \boldsymbol{\Sigma}_{yy}^r \end{bmatrix} \begin{bmatrix} \boldsymbol{l}_x \\ \boldsymbol{l}_y \end{bmatrix}.$$

$$\text{eq(5)} \qquad \begin{bmatrix} 0 & \boldsymbol{\Sigma}_{xy}^l \\ \boldsymbol{\Sigma}_{yx}^l & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_x \\ \boldsymbol{r}_y \end{bmatrix} = \lambda \begin{bmatrix} \boldsymbol{\Sigma}_{xx}^l & 0 \\ 0 & \boldsymbol{\Sigma}_{yy}^l \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_x \\ \boldsymbol{r}_y \end{bmatrix}.$$

Left transforms and right transforms are determined by iteratively solving above two equations until convergence.
In the case of image data, 2-D CCA, involves the generalized eigenvalue problem for much smaller size matrices, compared to the standard CCA, which reduces the complexity dramatically.

2.2 Universe subspace

CCA is viewed as learning a common subspace such that correlation between two data views is maximized. The common subspaces between two views can be formulated as

$$C = \frac{Xw_x + Y w_y}{2} \qquad (2.5)$$

Eq. (2.5) can be written in general form as:

$$U = \frac{1}{m} \sum_{v=1}^{m} (X_v w_v) \qquad (2.6)$$

Where, m is a total numbers of views, and U is universe subspace of all views.

Chapter 3

Algorithm

3.1 2-D CCA algorithm

Input –

- Image data X_t where t € (1,N) and Y_t where t € (1,N)
- The size of the canonical variable matrix : d_1 x d_2

Output –

- Left Transforms L_x and L_y
- Right Transforms R_x and R_y

Do centering image data to get

$$\widetilde{\boldsymbol{X}}_t = \boldsymbol{X}_t - \boldsymbol{M}_x, \qquad \widetilde{\boldsymbol{Y}}_t = \boldsymbol{Y}_t - \boldsymbol{M}_y.$$

Now, we will calculate the following matrices –

$$\begin{aligned}
\boldsymbol{\Sigma}_{xy}^{r} &= \left\langle \widetilde{\boldsymbol{X}} \boldsymbol{r}_x \boldsymbol{r}_y^\top \widetilde{\boldsymbol{Y}}^\top \right\rangle = \frac{1}{N} \sum_{t=1}^{N} \widetilde{\boldsymbol{X}}_t \boldsymbol{r}_x \boldsymbol{r}_y^\top \widetilde{\boldsymbol{Y}}_t^\top \\
\boldsymbol{\Sigma}_{xx}^{r} &= \left\langle \widetilde{\boldsymbol{X}} \boldsymbol{r}_x \boldsymbol{r}_x^\top \widetilde{\boldsymbol{X}}^\top \right\rangle = \frac{1}{N} \sum_{t=1}^{N} \widetilde{\boldsymbol{X}}_t \boldsymbol{r}_x \boldsymbol{r}_x^\top \widetilde{\boldsymbol{X}}_t^\top \\
\boldsymbol{\Sigma}_{yy}^{r} &= \left\langle \widetilde{\boldsymbol{Y}} \boldsymbol{r}_y \boldsymbol{r}_y^\top \widetilde{\boldsymbol{Y}}^\top \right\rangle = \frac{1}{N} \sum_{t=1}^{N} \widetilde{\boldsymbol{Y}}_t \boldsymbol{r}_r \boldsymbol{r}_y^\top \widetilde{\boldsymbol{Y}}_t^\top .
\end{aligned}$$

Compute d_1 largest generalized eigenvectors in eq(4) : L_x and L_y

Compute the following matrices

$$\begin{aligned}
\boldsymbol{\Sigma}^{l}_{xy} &= \left\langle \widetilde{\boldsymbol{X}}^{\top} \boldsymbol{l}_x \boldsymbol{l}_y^{\top} \widetilde{\boldsymbol{Y}} \right\rangle \\
\boldsymbol{\Sigma}^{l}_{xx} &= \left\langle \widetilde{\boldsymbol{X}}^{\top} \boldsymbol{l}_x \boldsymbol{l}_x^{\top} \widetilde{\boldsymbol{X}} \right\rangle \\
\boldsymbol{\Sigma}^{l}_{yy} &= \left\langle \widetilde{\boldsymbol{Y}}^{\top} \boldsymbol{l}_y \boldsymbol{l}_y^{\top} \widetilde{\boldsymbol{Y}} \right\rangle .
\end{aligned}$$

Compute d_2 largest generalized eigenvectors in eq(5): R_x and R_y until converged.

Chapter 4

Documentation of API

4.1 Package organization

sklearn.cross decomposition.CCA

xindim

Get the dimension of `X`, the first set of variables.

yindim

Get the dimension of `Y`, the second set of variables.

outdim

Get the output dimension, *i.e* that of the common space.

xmean

Get the mean vector of `X` (of length `dx`).

ymean

Get the mean vector of `Y` (of length `dy`).

xprojection

Get the projection matrix for `X` (of size `(dx, p)`).

yprojection

Get the projection matrix for `Y` (of size `(dy, p)`).

correlations

The correlations of the projected componnents (a vector of length `p`).

4.2 Methods

xtransform(*M*, *x*)

Transform observations in the X-space to the common space.

Here, `x` can be either a vector of length `dx` or a matrix where each column is an observation.

ytransform(*M*, *y*)

Transform observations in the Y-space to the common space.

Here, `y` can be either a vector of length `dy` or a matrix where each column is an observation.

fit(*CCA*, *X*, *Y; ...*)

Perform CCA over the data given in matrices `X` and `Y`. Each column of `X` and `Y` is an observation.

`X` and `Y` should have the same number of columns (denoted by `n` below).

This method returns an instance of `CCA`.

ccacov(*Cxx*, *Cyy*, *Cxy*, *xmean*, *ymean*, *p*)

Compute CCA based on analysis of the given covariance matrices, using generalized eigenvalue decomposition.

Chapter 5

Example

5.1 Example 1

```
from sklearn.cross_decomposition import CCA
X = [[1, 2], [4,5]]
Y= [[8, 9], [5,2]]
```

Covariance between left and right transform matrix w.r.t X and Y.

```
[[1624.5, 4702.5], [4702.5,13612.5]]
[[353640.5,158528.5],[158528.5,71064.5]]
```

Cov(x,y) for right transform –
```
[[4.5, -94.5],[-94.5,1984.5]]
```

Cov(x,x) for right transform –
```
[[364.5, -850.5],[-850.5,1984.5]]
```

Cov(y,y) for right transform –
```
[[364.5, -850.5],[-850.5,1984.5]]
```

Cov(x,y) for left transform –
```
[[262812.5, -304862.5],[304862.5,353640.5]]
```

Cov(x,y) for left transform –
```
[[74884.5, -81850.5],[-81850.5,89464.5]]
```

Cov(x,y) for left transform –
```
[[8841012.5, -8841012.5],[ 8841012.5, 8841012.5]]
```

Chapter 6

Learning Outcomes

6.1 In this project, we have presented a two dimensional extension of CCA that directly takes image data as inputs without reshaping them into vectors, in order to correlate relationships between them.

6.2 The main advantage of 2D-CCA is two fold:

- Low computational complexity.
- Preserving spatial structure of image data in the calculation of canonical variable matrices.

Bibliography

[1] Sun Ho Lee and Seungjin Choi, Member, IEEE: Two – dimensional Canonical Correlation Analysis, vol. 14, 10 October 2007.

[2] Cai-rong Zou, Ning Sun, Zhen-hai Ji, Li Zhao 2, Foshan University, Foshan 528000, China 2 Department of Radio Engineering, Southeast University, Nanjing 210096, China, IEEE : 2DCCA: A Novel Method for Small Sample Size Face Recognition, 0-7695-2794-9/07