多模态学习导论 第四周 作业2

> 黄家曦 2022141460084

1. 降维的本质与目的是什么？

降维的本质是在尽可能保留数据关键信息（如主要结构或变化趋势）的前提下，通过数学变换将高维数据映射到低维空间，从而实现数据的简化与低维表示（目的）。

2.请简述PCA的算法过程

考虑一个高维空间，我们希望用一个超平面来对所有样本进行恰当表达。它需要满足如下要求

- 样本点到这个超平面都足够近
- 样本点在这个超平面的投影尽可能的分割开(baozheng)
  对于前一种性质。假设样本$X_1 - X_n$都进行了中心化，即$\sum_i x_i = 0$。通过投影过后得到的新坐标为$w_1, w_2, \cdots, w_d$，其中w_i为正交基向量，$w_i^T w_j = 0$,$\|W_i\|_2 = 1$,将x_i降维到低维空间，$\hat{x_i} = \sum_{j=1}^{d} z_{ij} w_j$，其中$z_{ij}$是$x_i$在低维空间下第j维的坐标。
  对整个样本空间，原样本点与投影后的样本点之间的距离为:

  $$\begin{aligned}\sum_{i=1}^{m} \| \sum j = 1^d z_{ij} w_j - x_i \|_2^2 &= \sum_{i=1}^{m} z_i^T z_i - 2 \sum_{i=1}^{m} z_i^T W^T x_i + c \\ &\simeq -tr(W^T(\sum_{i=1}^{m} x_i x_i^T)W)\end{aligned}$$

  则需要最小化距离，即:

  $$\begin{aligned}\min_W -tr(W^T X X^T W) \\ s.t. W^T W = I\end{aligned}$$

  去掉负号，即：

  $$\begin{aligned}\max_W tr(W^T X X^T W) \\ s.t. W^T W = I\end{aligned}$$

  从后一种性质考虑，原样本x_i在新的超平面上的投影为$W^T x_i$，目标是让投影尽可能分开，等价于让投影后样本点方差最大化。
  投影后方差为$Var(\hat{X}) = \frac{1}{n} \sum_{i=1}^{n} (w^T(\hat{x_i} - \bar{x}))^2$，即$Var(\hat{X}) = \frac{1}{n} \sum_{i=1}^{n} w^T(\hat{x_i} - \bar{x})(\hat{x_i} - \bar{x})^T w$，这里因为$X$已经中心化，所以$\hat{x} = 0$，所以投影后样本方差为$\sum_i W^T x_i x_i^T W$，于是优化目标为

  $$\begin{aligned}\max_W tr(W^T X X^T W) \\ s.t. W^T W = 1\end{aligned}$$

  对目标函数，$X X^T$是原样本的协方差，我们另$X X^T = \Lambda$，构造拉格朗日函数：

  $$L(w) = w^T \Lambda w + \lambda(1 - w^T w)$$

  对w求导：

  $$\begin{aligned}\frac{\partial L}{\partial w} = 2\Lambda w - 2\lambda w = 0 \\ \Lambda w = \lambda w\end{aligned}$$

  此时方差为：

  $$Var(\hat{x}) = w^T \Lambda w = w^T \lambda w = \lambda w^T w = \lambda$$

即：

$$tr(W^TXX^TW) = \lambda$$

于是我们发现，x 投影后的方差就是协方差矩阵的特征值。我们要找到最大方差也就是协方差矩阵最大的特征值，最佳投影方向就是最大特征值所对应的特征向量，次佳就是第二大特征值对应的特征向量，以此类推。
对协方差矩阵XX^T做特征值分解，将求得的特征值排序：$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$，取前$d$个特征值构成特征向量$W = (w_1, w_2, \cdots, w_d)$，这就是PCA的解。

3.请给出PCA的目标函数以及推导过程

问题2已经描述，这里复制粘贴一下
优化目标为

$$\max_W tr(W^TXX^TW)$$
$$s.t. W^TW = 1$$

原样本x_i在新的超平面上的投影为W^Tx_i，目标是让投影尽可能分开，等价于让投影后样本点方差最大化。
投影后方差为$Var(\hat{X}) = \frac{1}{n}\sum_{i=1}^{n}(w^T(\hat{x_i} - \bar{x}))^2$，即$Var(\hat{X}) = \frac{1}{n}\sum_{i=1}^{n} w^T(\hat{x_i} - \bar{x})(\hat{x_i} - \bar{x})^T w$，这里因为$X$已经中心化，所以$\hat{x} = 0$，所以投影后样本方差为$\sum_i W^Tx_ix_i^TW$，于是目标函数为:

$$\max_W tr(W^TXX^TW)$$
$$s.t. W^TW = 1$$

4.请简单的解释一下特征脸是什么？如何得到的？

对于m*m个像素，n个样本构成的数据集，将数据处理为一个列向量，所有样本构造一个矩阵，对样本求均值，并对样本做中心化处理。对中心化后的样本，求协方差矩阵XX^T的特征值和对应的特征向量，里面每一个特征向量都对应一个**特征脸**，理论上原样本中的脸可以表示为特征脸的线性组合。

5.请简单的解释一下CCA和PCA的联系

- 目标
  CCA（典型相关分析）是研究两个多变量（向量）之间之间的线性相关关系，能够揭示出两组变量之间的内在联系。即最大化两个数据集投影后变量的相关性。
  PCA是研究单个数据集的内部结构，并试图通过线性变换提取最重要的特征组成，从而确定最重要的几个特征并进行降维。即最大化投影后的数据方差。
- 数学推导
  CCA 的核心是通过对两个数据集的协方差矩阵进行广义特征值分解，找到最大化两者相关性的投影方向。给定两个数据集 X 和 Y，CCA 通过对联合协方差矩阵 $\Sigma XX, \Sigma YY, \Sigma XY$ 进行广义特征值分解来找到典型相关向量。
  PCA 的核心是求解协方差矩阵的特征值分解问题，找到使数据方差最大的方向。具体来说，给定数据矩阵 X，PCA 通过对 X^TX（协方差矩阵）进行特征值分解来找到主成分。
  如果将 CCA 应用于同一个数据集（即 X=Y），那么 CCA 实际上退化为 PCA。换句话说，PCA 是 CCA 的一个特例。

6. 请简述CCA的求解过程

给定一个数据集Z，我们想研究X, Y，两个Z中互不相关的子集之间的关系，其中:

$$X \in R^{n \times p}$$
$$Y \in R^{n \times q}$$

n为样本数，p为对应特征数
CCA的优化目标是找到两组投影$W_x \in R^p$和$W_y \in R^q$，使得X和Y分别在W_x和W_y上投影后的相关性最大。即损失函数为：

$$\max_{W_x, W_y} Corr(W_xX, W_yY)$$

首先对样本做中心化处理，此时协方差矩阵为：

- $\sum_{XX} = \frac{1}{n}X^TX$
- $\sum_{YY} = \frac{1}{n}Y^TY$
- $\sum_{XY} = \frac{1}{n}X^TY$
  即CCA的损失函数为：

$$\max_{w_x,w_y} \frac{w_x^T \sum_{XY} w_y}{\sqrt{w_x^T \sum_{XX} w_x}\sqrt{w_y^T \sum_{YY} w_y}}$$

  通过归一化等约束，如$w_x^T \sum_{XX} w_x = 1$、$w_y^T \sum_{YY} w_y = 1$，可以简化求解广义特征值
  对于 w_x，求解广义特征值问题：

$$\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{YX}w_x = \rho^2 w_x, \tag{1}$$

  对于 w_y，求解广义特征值问题：

$$\Sigma_{YY}^{-1}\Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}w_y = \rho^2 w_y. \tag{2}$$

  将特征值从大到小排序提取对应的特征向量w_x、w_y，对应特征值的平方根为典型相关系数

7.请用matlab或python实现CCA,并给出运行实验结果和代码

```
通过python实现CCA
```

```python
def manual_cca(X, Y, n_components=None):
    """
    手动实现 Canonical Correlation Analysis (CCA), 支持指定 n_components

    参数:
        X: 第一组变量 (n_samples, n_features_x)
        Y: 第二组变量 (n_samples, n_features_y)
        n_components: 提取的典型变量对数量（默认为 min(X.shape[1], Y.shape[1]))

    返回:
        canonical_correlations: 前 n_components 个典型相关系数
        x_weights: 第一组变量的典型向量（每列对应一个典型变量）
        y_weights: 第二组变量的典型向量（每列对应一个典型变量）
    """
    # 数据中心化
    X = X - np.mean(X, axis=0)
    Y = Y - np.mean(Y, axis=0)

    # 计算协方差矩阵
    Sigma_xx = np.cov(X, rowvar=False)  # X 的协方差矩阵
    Sigma_yy = np.cov(Y, rowvar=False)  # Y 的协方差矩阵
    Sigma_xy = np.cov(np.hstack([X, Y]), rowvar=False)[:X.shape[1], X.shape[1]:]  # X 和 Y 的交叉协方差矩阵
    Sigma_yx = Sigma_xy.T  # Y 和 X 的交叉协方差矩阵

    # 求解广义特征值问题
    A = np.linalg.inv(Sigma_xx) @ Sigma_xy @ np.linalg.inv(Sigma_yy) @ Sigma_yx
    eigenvalues, eigenvectors_x = np.linalg.eig(A)

    # 排序特征值和特征向量
    sorted_indices = np.argsort(eigenvalues)[::-1]
    eigenvalues = eigenvalues[sorted_indices]
    eigenvectors_x = eigenvectors_x[:, sorted_indices]

    # 计算 Y 的典型向量
    eigenvectors_y = np.linalg.inv(Sigma_yy) @ Sigma_yx @ eigenvectors_x

    # 如果未指定 n_components，则默认为 min(X.shape[1], Y.shape[1])
    if n_components is None:
        n_components = min(X.shape[1], Y.shape[1])

    # 截取前 n_components 个典型相关系数和典型向量
    canonical_correlations = np.sqrt(eigenvalues[:n_components])
    x_weights = eigenvectors_x[:, :n_components]
    y_weights = eigenvectors_y[:, :n_components]

    return canonical_correlations, x_weights, y_weights
```

对于kaggle上一个数据集：
`Correlation between Posture & Personality Trait`
我们想看看两组变量的相关性

```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

dataset = pd.read_csv("/kaggle/input/correlation-between-posture-personality-trait/Myers Briggs Table_S1.csv")
```

可以看到原始数据集为：

S No	AGE	HEIGHT	WEIGHT	SEX	ACTIVITY LEVEL	PAIN 1	PAIN 2	PAIN 3	PAIN 4	MBTI	E	I
0	1	53	62	Female	Low	0.0	0.0	0.0	0.0	ESFJ	18	3

S No	AGE	HEIGHT	WEIGHT	SEX	ACTIVITY LEVEL	PAIN 1	PAIN 2	PAIN 3	PAIN 4	MBTI	E	I
1	2	52	69	Male	High	7.0	8.0	5.0	3.0	ISTJ	6	15
2	3	30	69	Male	High	0.0	0.0	0.0	0.0	ESTJ	15	6
3	4	51	66	Male	Moderate	9.5	9.5	9.5	1.5	ISTJ	6	15
4	5	45	63	Female	Moderate	4.0	5.0	2.0	2.0	ENFJ	14	7
...	...	...	...	...	...	...	...	...	...	...	..	..
92	93	16	58	100	Male	Moderate	0.0	0.0	0.0	3.0	ESTP	19
93	94	45	62	134	Female	Moderate	0.0	4.0	0.0	0.0	ESFJ	11
94	95	43	69	188	Male	Moderate	2.0	0.0	0.0	0.0	ENFP	12
95	96	28	67	180	Female	Low	0.0	0.0	0.0	0.0	ESFJ	11
96	97	43	69	188	Male	Moderate	4.0	0.0	0.0	0.0	ENFP	12
提取数据特征并进行预处理：												

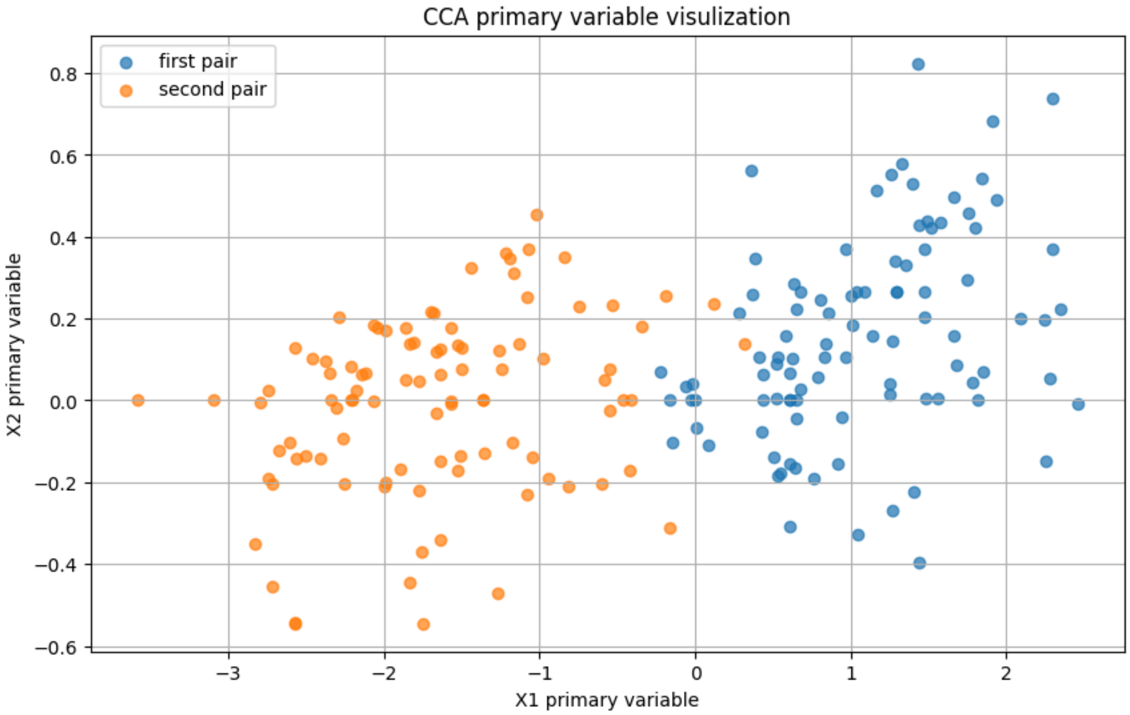
```
x_1 = dataset[["AGE","SEX","POSTURE","ACTIVITY LEVEL"]]
x_2 = dataset[["PAIN 1","PAIN 2","PAIN 3","PAIN 4"]]

x_1['SEX'] = x_1['SEX'].map({'Female': 0, 'Male': 1})
x_1['POSTURE'] = x_1['POSTURE'].map({'A': 0, 'B': 1, 'C': 2, 'D': 3})
x_1['ACTIVITY LEVEL'] = x_1['ACTIVITY LEVEL'].map({'Low': -1, 'Moderate': 0, 'High': 1})
x_1 = np.array(x_1)
x_2 = np.array(x_2)
```

调用manual_cca函数并计算降维后的典型变量：

```
canonical_correlations, x_weights, y_weights = manual_cca(x_1,x_2,n_components=2)

print("典型相关系数:", canonical_correlations)

x1_c = x_1 @ x_weights  # 第一组典型变量
x2_c = x_2 @ y_weights  # 第二组典型变量
```

可视化降维后结果：

```
import matplotlib.pyplot as plt

# 可视化CCA结果
plt.figure(figsize=(10, 6))
plt.scatter(x1_c[:, 0], x2_c[:, 0], label="first pair", alpha=0.7)
plt.scatter(x1_c[:, 1], x2_c[:, 1], label="second pair", alpha=0.7)
plt.xlabel("X1 primary variable")
plt.ylabel("X2 primary variable")
plt.title("CCA primary variable visulization")
plt.legend()
plt.grid()
plt.show()
```


CCA primary variable visulization
first pair
second pair
X2 primary variable
X1 primary variable
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−3
−2
−1
0
1
2