Multimodal Model Homework 2

March 30, 2025

2023141460251 孙浩翔

Question 1: 降维的本质和目的是什么？

降维的本质是将原始高维数据映射到一个低维空间中，同时尽可能保留原始数据的重要信息。其目的主要包括：

- 减少计算复杂度，提升模型训练效率；
- 降低噪声影响，提高模型泛化能力；
- 便于数据可视化（如将高维数据压缩到二维或三维）；
- 避免"维度灾难"，改善样本分布密度。

Question 2: 请简述主成分分析（PCA）的算法过程

主成分分析（PCA）的基本步骤如下：

1. **数据中心化**：对每一维减去其均值。
2. **计算协方差矩阵**：设数据为 $X \in \mathbb{R}^{n\times d}$，则协方差矩阵为：
$$C = \frac{1}{n}X^\top X \tag{1}$$
3. **计算特征值与特征向量**：对协方差矩阵 C 进行特征分解，得到特征值与对应特征向量。
4. **选取前 k 个最大特征值对应的特征向量**：组成投影矩阵 $W \in \mathbb{R}^{d\times k}$。
5. **将数据映射到低维空间**：
$$Z = XW \tag{2}$$

Question 3: 请给出 PCA 的目标函数，并推导其过程

PCA 的目标是最大化数据投影后的方差，目标函数为：

$$\max_{w} \quad \mathrm{Var}(w^\top x) = w^\top C w \quad \text{s.t. } \|w\|_2 = 1 \tag{3}$$

该问题可转化为拉格朗日乘子法：

$$L(w, \lambda) = w^\top C w - \lambda(w^\top w - 1) \tag{4}$$

对 w 求导并令导数为 0：

$$\nabla_w L = 2Cw - 2\lambda w = 0 \Rightarrow Cw = \lambda w \tag{5}$$

因此，w 是 C 的特征向量，λ 是对应特征值。选择最大特征值对应的特征向量即为第一主成分。

Question 4: 请简单地解释一下特征脸是什么，如何得到的？

特征脸（Eigenfaces）是一种基于 PCA 的人脸识别方法，将人脸图像看作高维向量，通过 PCA 提取出人脸图像的主成分方向。

获取过程：

1. 收集标准化的灰度人脸图像，将其展平为向量。
2. 进行 PCA 得到特征向量，这些特征向量即为“特征脸”。
3. 任意新的人脸图像可以用这组特征脸进行线性组合进行表示与识别。

Question 5: 请解释典型相关分析（CCA）与 PCA 的联系

联系：

- 都是线性降维方法，试图找到低维空间中的代表性特征；
- 都利用特征值分解/广义特征值分解思想。

不同点：

- PCA 只考虑一个变量集（例如图像），寻找最大方差方向；
- CCA 同时考虑两个变量集（例如图像与文本），寻找两个集合之间最大相关性的线性组合方向。

Question 6: 请简述 CCA 的求解过程

设两组变量分别为 $X \in \mathbb{R}^{n\times p}, Y \in \mathbb{R}^{n\times q}$,
CCA 寻找向量 w_x, w_y 使得两组线性组合之间的相关性最大:

$$\rho = \max_{w_x, w_y} \frac{w_x^\top C_{XY} w_y}{\sqrt{w_x^\top C_{XX} w_x \cdot w_y^\top C_{YY} w_y}} \tag{6}$$

使用拉格朗日乘子法进行优化,最终可转化为解如下广义特征值问题:

$$C_{XY} C_{YY}^{-1} C_{YX} w_x = \lambda^2 C_{XX} w_x \tag{7}$$

对称地:

$$C_{YX} C_{XX}^{-1} C_{XY} w_y = \lambda^2 C_{YY} w_y \tag{8}$$

得到最优方向 w_x 与 w_y 即可构造典型变量。

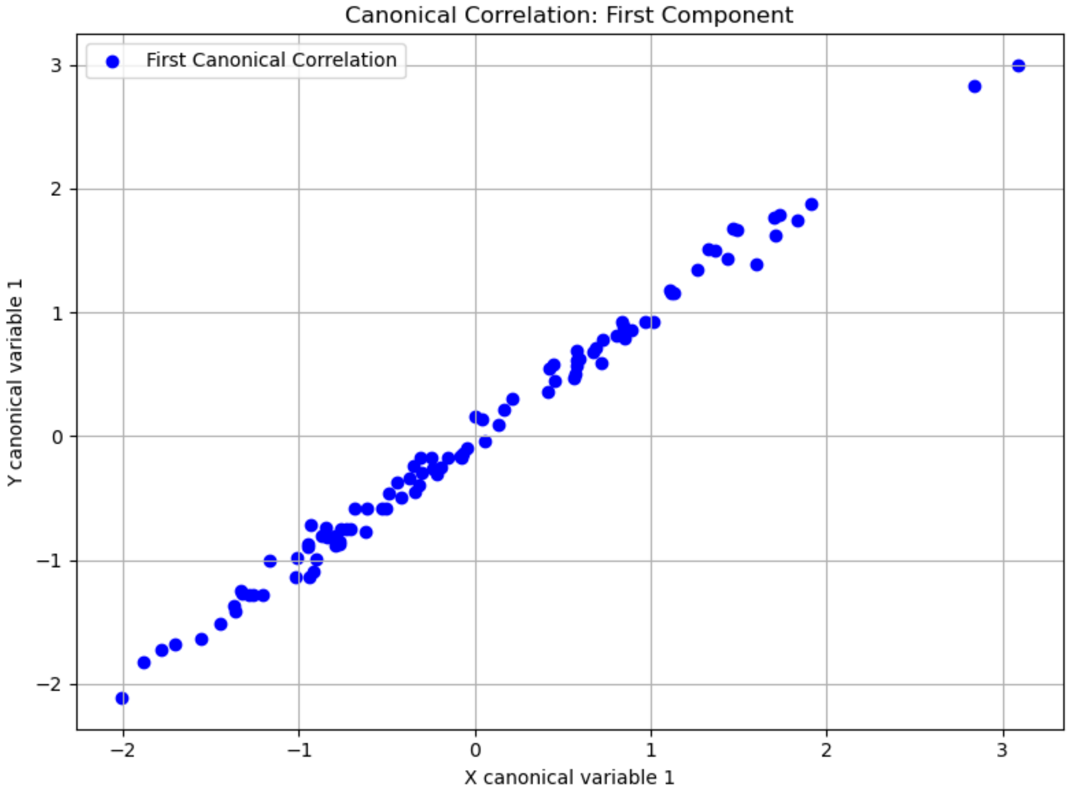
Question 7: Canonical Correlation Analysis (CCA)

```python
import numpy as np
from sklearn.cross_decomposition import CCA
import matplotlib.pyplot as plt

# 设置随机种子
np.random.seed(42)

# 生成两个相关的数据集
n_samples = 100
x = np.random.normal(size=(n_samples, 3))
y = x @ np.array([[0.5, 0.2], [-0.3, 0.8], [0.1, -0.5]]) + np.random.normal(size=(n_samples, 2)) *

# 执行 CCA
cca = CCA(n_components=2)
x_c, y_c = cca.fit_transform(x, y)

# 打印结果
print("X Canonical Variables (first 5 rows):")
print(x_c[:5])
print("Y Canonical Variables (first 5 rows):")
print(y_c[:5])

# 可视化
plt.figure(figsize=(8, 6))
plt.scatter(x_c[:, 0], y_c[:, 0], c='blue', label='First Canonical Correlation')
```

```python
plt.xlabel('X canonical variable 1')
plt.ylabel('Y canonical variable 1')
plt.title('Canonical Correlation: First Component')
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()

# 计算相关系数
corr_matrix = np.corrcoef(x_c.T, y_c.T)
corr_x1_y1 = corr_matrix[0, 2]
corr_x2_y2 = corr_matrix[1, 3]

print(f"Canonical Correlation 1: {corr_x1_y1:.4f}")
print(f"Canonical Correlation 2: {corr_x2_y2:.4f}")
```

输出结果:

```
X Canonical Variables (first 5 rows):
[[-0.23986486  0.41054012]
 [ 0.16851109  1.7431958 ]
 [ 1.13172628  1.67380057]
 [ 0.05780183  0.64085221]
 [-0.52720026  0.65774335]]

Y Canonical Variables (first 5 rows):
[[-0.25923129  0.15739204]
 [ 0.21150973  1.56527998]
 [ 1.15534691  1.37172262]
 [-0.03713477  0.69209119]
 [-0.58466592  0.55804203]]
```




```
Canonical Correlation 1: 0.9965
Canonical Correlation 2: 0.9744
```