计算机科学与技术学院2018机器学习—实验报告

- 课程名称： 机器学习
- 课程类型： 选修
- 实验题目：PCA模型实验
- 学号：1160100626
- 姓名：单心茹

实验目的

实现一个PCA模型，能够对给定数据进行降维（即找到其中的主成分），可以利用已有的矩阵特征向量提取方法。

实验要求求及实验环境

实验要求

1. 首先人工生成一些数据（如三维数据），让它们主要分布在低维空间中，如首先让某个维度的方差远小于其它维度，然后对这些数据旋转。生成这些数据后，用你的PCA方法进行主成分提取。
2. 利用手写体数字数据mnist，用你实现PCA方法对该数据降维，找出一些主成分，然后用这些主成分对每一副图像进行重建，比较一些它们与原图像有多大差别（可以用信噪比衡量）。

实验环境

Win10 + 8GB RAM + python 3.6

设计思想与实现

PCA

PCA是一种无监督的学习方式，是一种很常用的降维方法。PCA可以把可能具有线性相关性的高维变量合成为线性无关的低维变量，称为**主成分（principal components）**，新的低维数据集会尽可能的保留原始数据的变量，可以将高维数据集映射到低维空间的同时，尽可能的保留更多变量。

PCA数学理论

基

在二维空间中，任何两个线性无关的二维向量都可以成为一组基。推广到多维空间，**要准确描述向量，首先要确定一组基，然后给出在基所在的各个直线上的投影值。**

内积

向量A和B的内积公式为：

$$A \cdot B = |A||B|cos(\alpha)$$

A与B的内积等于A到B的投影长度乘以B的模。若假设$|B| = 1$那么就变成了

$$A \cdot B = |A|cos(\alpha)$$

基变换的矩阵表示

一般的，有M个N维向量，想将其变换为由R个N维向量（R个基）表示的新空间中，那么首先将R个基按行组成矩阵P，然后将待变换向量按列组成矩阵X，那么两矩阵的乘积就是变换结果。R可以小于N，而R决定了变换后数据的维数。也就是说，我们可以将一N维数据变换到更低维度的空间中去，变换后的维度取决于基的数量。 因此这种矩阵相乘可以表示降维变换：

$$P_{R\times N} \times X_{N\times M} = Y_{R\times M}$$

$$\begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_R \end{pmatrix} \begin{pmatrix} a_1 & a_2 & \cdots & a_M \end{pmatrix} = \begin{pmatrix} p_1a_1 & p_1a_2 & \cdots & p_1a_M \\ p_2a_1 & p_2a_2 & \cdots & p_2a_M \\ \vdots & \vdots & \ddots & \vdots \\ p_Ra_1 & p_Ra_2 & \cdots & p_Ra_M \end{pmatrix}$$

优化目标

如何选择R个基才能最大程度的保留原来的信息？我们希望降维后的数据尽可能的分散，**数据的分散程度用方差描述。**当目标是降到k>1维时，我们又如何选取第二个特征？**用协方差表示两个维度的相关性。**为了让不同维度尽可能表示更多的原始信息，我们不希望它们之间存在（线性）相关性，那么他们之间重复表示的信息就越少。

- 降维问题的优化目标：

将N维降维R维，选择R个单位正交基，使得原始数据变换到这组基上后，各维度两两间的**协方差**为0，而每个维度的**方差**则尽可能大（在正交的约束下，取最大的R个方差）。

协方差矩阵

设我们有M个N维数据记录，将其按列排成N乘M的矩阵X，设$C = \frac{1}{m}XX^T$，则C是一个对称矩阵，其对角线分别个各个字段的方差，而第i行j列和j行i列元素相同，表示i和j两个字段的协方差。

设原始矩阵为X(N×M)，表示M个N维向量，其协方差矩阵为C(N×N)；P(R×N)为变换矩阵；Y(R×M)为目标矩阵，其协方差矩阵为D。我们要求降维后的矩阵Y的每一维包含的数据足够分散，也就是每一行（维）方差足够大，而且要求行之间的元素线性无关，也就是要求行之间的协方差全部为0，故**将协方差矩阵对角化，除对角线外的其他元素全为0，并且在对角线上将元素按大小从上到下排列。**

$$D = \frac{1}{M}YY^T = \frac{1}{M}PXX^TP^T = PCP^T$$

- 优化目标变成了**寻找一个矩阵P，满足PCP^T是一个对角矩阵，并且对角元素按从大到小依次排列，那么P的前K行就是要寻找的基，用P的前K行组成的矩阵乘以X就使得X从N维降到了K维并满足上述优化条件。**

C是X的协方差矩阵，是实对称矩阵，**整个PCA降维过程其实就是一个实对称矩阵对角化的过程。**

Algorithm：

设有M条N维数据。

1. 将原始数据按列组成N行M列矩阵X
2. 将X的每一行（一个维度）进行零均值化，即减去这一行的均值。
3. 求出协方差矩阵$C = \frac{1}{M}XX^T$
4. 求出协方差矩阵的特征值及对应的特征向量
5. 将特征向量按对应特征值大小从上到下按行排列成矩阵，取前k行组成矩阵P
6. $Y = PX$即为降维到k维后的数据

PCA核心代码实现

```python
# 行表示样本，列表示特征维度
def my_pca(X, k):
    mean_vals = np.mean(X, axis=0)  # 对每一维求均值
    mean_scaling = X - mean_vals    # 每一维零均值化
    cov = np.cov(mean_scaling, rowvar=0)    # 计算方差
    eig_vals, eig_vects = np.linalg.eig(np.mat(cov))    # 计算特征值和特征向量

    eig_val_sort = np.argsort(eig_vals)          # 对特征值进行排序
    eig_val_sort = eig_val_sort[:-(k+1):-1]      # 从升序排好的特征值，从后往前取k个
    feature = eig_vects[:, eig_val_sort]         # 返回主成分
    low_dimension = mean_scaling * feature       # 将原始数据投影到主成分上得到新的低维数据
    recon_data = (low_dimension * feature.T) + mean_vals     # 重构数据
    return low_dimension, recon_data
```

生成数据利用PCA提取

利用高斯分布生成三维数据，其中一维的方差远小于其余两维，所有均值，方差均人为设定。

输出利用上面实现的my_pca与sklearn pca的结果进行比较

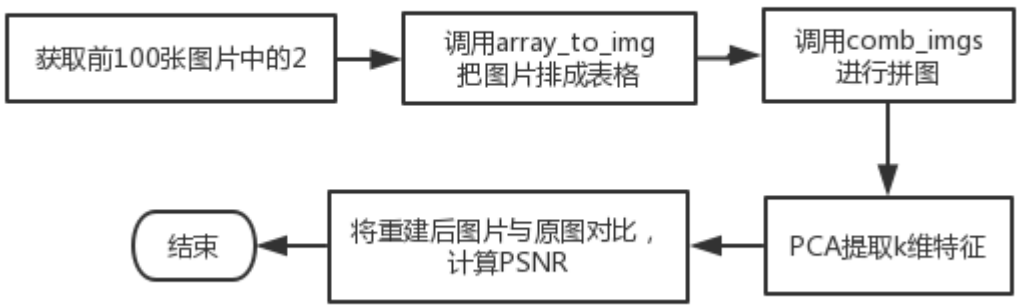
具体实现如下：

```
def create_data():
    np.random.seed()
    s1 = np.random.normal(3, 2, 50).reshape(50, 1)
    s2 = np.random.normal(3.5, 2.4, 50).reshape(50, 1)
    s3 = np.random.normal(8, 0.05, 50).reshape(50, 1)
    s = np.hstack((s1, s2))
    x = np.hstack((s, s3))

    low_data, recon_data = my_pca(np.mat(x), 1)
    pca = PCA(n_components=1)
    print('--------------第一行为my_pca结果，第二行为sklearn结果-----------------')
    result = np.hstack((-low_data.T, pca.fit_transform(np.mat(x)).T))
    print(result)
    print('-------------------------------------------------------------------')

```

MNIST数据验证

获取mnist数据集里前100张图片中的手写数字2，对它们进行主成分分析，**通过计算峰值信噪比（PSNR）将两张图片进行对比。**

对比流程图如下：



PSNR用来评价一幅图像处理后和原图像相比质量的好坏，**PSNR越高，失真越小。**主要定义两个值，一个是均方差MSE，另一个是峰值信噪比$PSNR$，公式如下：

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}||I(i,j) - K(i,j)||^2$$

$$PSNR = 20 \cdot log_{10}(\frac{MAX_I}{MSE^{\frac{1}{2}}})$$

其中PSNR计算实现如下：

```python
def psnr(img1, img2):
    mse = np.mean((img1-img2)**2)
    if mse == 0:
        return 100
    PIXEL_MAX = 255.0
    return 20 * math.log10(PIXEL_MAX / math.sqrt(mse))
```

实验结果与分析

生成数据并PCA提取

分别为my_pca和sklearn训练的结果，对比相同：

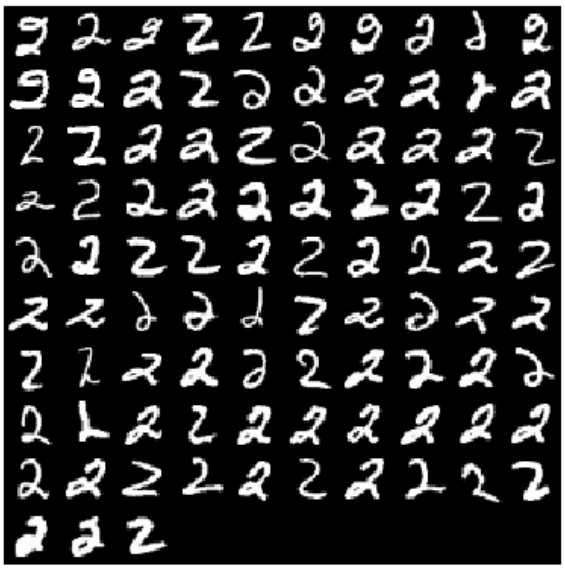
```
--------------第一行为my_pca结果，第二行为sklearn结果-----------------
[[ 1.38826216 -2.99465621  0.6657436   0.32263214 -0.21988861  2.6312614
   1.67154662 -2.30445933 -2.04055624  1.14006782  2.04974655  1.89193344
   1.38161814 -3.99992001 -1.46489535  1.08371041  0.02260352 -1.79221213
   3.18321158  3.44276313  7.71845106 -0.6552852   0.70208006  2.13568239
   0.87865212  0.99027984  3.8450544  -1.158675    0.4287544  -1.90284081
  -1.50428293  4.10443489 -1.02561814 -0.54770034  0.36233683 -3.69792284
  -0.62618416 -2.66680608  0.09139524  2.45753072 -1.11262645 -4.19748449
  -0.94583172 -0.20286289 -6.3191831   0.69226839  0.78482346 -1.91454716
  -0.24730894 -2.52509619]
 [ 1.38826216 -2.99465621  0.6657436   0.32263214 -0.21988861  2.6312614
   1.67154662 -2.30445933 -2.04055624  1.14006782  2.04974655  1.89193344
   1.38161814 -3.99992001 -1.46489535  1.08371041  0.02260352 -1.79221213
   3.18321158  3.44276313  7.71845106 -0.6552852   0.70208006  2.13568239
   0.87865212  0.99027984  3.8450544  -1.158675    0.4287544  -1.90284081
  -1.50428293  4.10443489 -1.02561814 -0.54770034  0.36233683 -3.69792284
  -0.62618416 -2.66680608  0.09139524  2.45753072 -1.11262645 -4.19748449
  -0.94583172 -0.20286289 -6.3191831   0.69226839  0.78482346 -1.91454716
  -0.24730894 -2.52509619]]
-----------------------------------------------------------------------
```

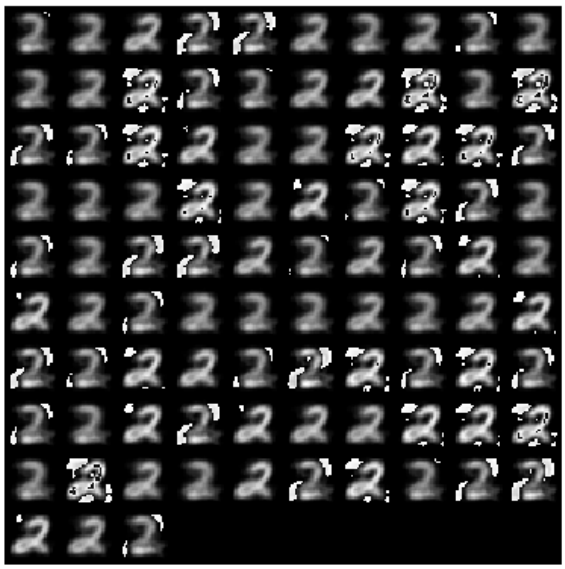
MNIST手写数据PCA提取

PCA提取k维：

- k = 1，如图PSNR = 31.33

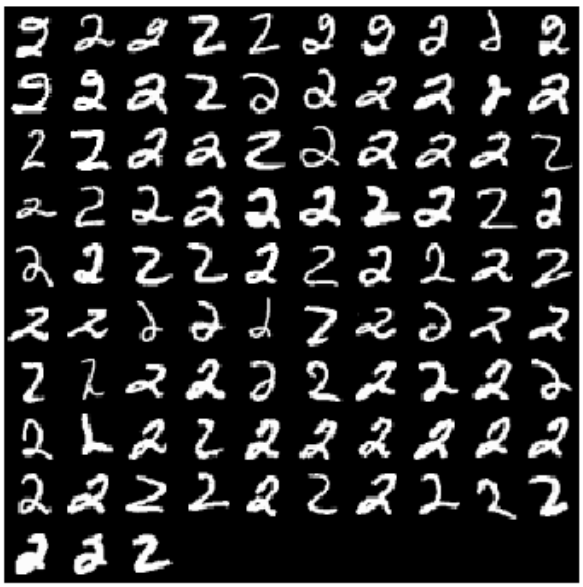
PSNR: 31.33

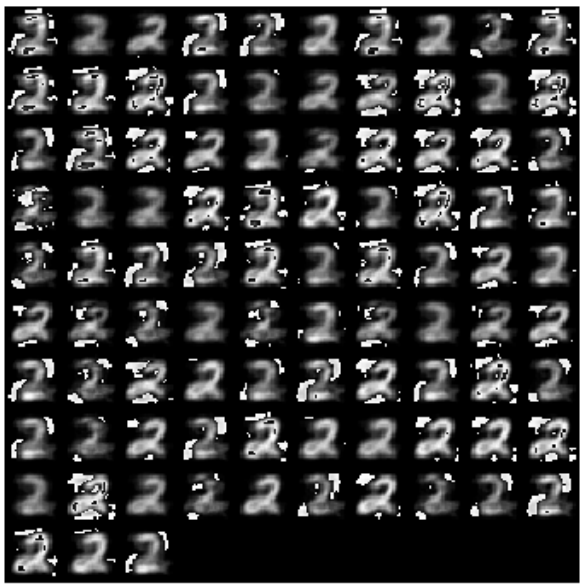




- k = 2，PSNR = 31.30

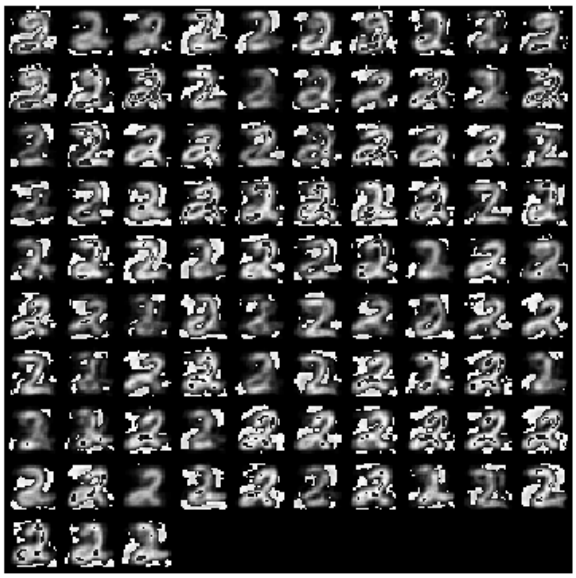
PSNR: 31.30





- k = 6, PSNR = 31.27

PSNR: 31.27



结论

- 在对mnist数据集的100个2的主成分提取中，随着提取维数增加，重构后的图片更趋近于原手写字体，也就是说特征更丰富。降为1维时，重构后的图像明显比原始图像的数字“规整”得多，证明提取出了主要成分
- 随着提取维数的增加，PSNR递减，虽然PSNR越大，代表与原图像失真率越低。由于人类视觉特性的差异性，通常出现PSNR的评价结果与人的主要感觉不一致。但此处可能是较多特征提取后，重构的图像数字边缘明显噪声增大，故PSNR递减。

参考文献

李航——《统计学习方法》

[PCA的数学原理]|CodingLabs

PCA原理及Python实现|Yu Blog

How To Calculate PSNR Value Of Two Images Using Python?

[马同学高等数学]如何理解主元分析（PCA）？

主成分分析降维（MNIST数据集）

附录：源代码

详见压缩包内PCA_mnist.py和PCA_test.py