哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称： 机器学习

课程类型： 选修

实验题目： PCA

学号：1190202107
姓名： 姚舜宇

一、 实验目的

实现一个 PCA 模型，能够对给定数据进行降维。

二、 实验要求及实验环境

1. 人工生成一些数据，让它们主要分布在低维空间中，然后使用 PCA 模型对数据进行主成分提取。
2. 找一个人脸数据，使用 PCA 方法进行降维，找出一些主成分，然后用这些主成分对每一副人脸图像进行重建，并比较它们与原图的差别。

环境：Windows10，python3.9，pycharm

三、 设计思想（本程序中的用到的主要算法及数据结构）

定义数据集$X=[\boldsymbol{x}_1\ \ \boldsymbol{x}_2\ \ \cdots\ \ \boldsymbol{x}_N]^T{}_{N\times p}=\begin{bmatrix}x_{11}&x_{12}&\cdots&x_{1p}\\x_{21}&x_{22}&\cdots&x_{2p}\\\vdots&\vdots&\ddots&\vdots\\x_{N1}&x_{N2}&\cdots&x_{Np}\end{bmatrix}_{N\times p}$，$\boldsymbol{x}_i\in\mathcal{R}^p, i=1,2,\cdots,N$。

可以计算样本均值$\bar{\boldsymbol{x}}_{p\times 1}=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i=\frac{1}{N}[\boldsymbol{x}_1\ \ \boldsymbol{x}_2\ \ \cdots\ \ \boldsymbol{x}_N]\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}=\frac{1}{N}X^T\mathbf{1}_N$

样本协方差矩阵$S=\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x}_i-\bar{\boldsymbol{x}})(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^T=\frac{1}{N}X^THX$，其中$H=I_N-\frac{1}{N}\mathbf{1}_N\mathbf{1}_N{}^T$

定义 p 维向量$\boldsymbol{x}$到 p 维向量$\boldsymbol{y}$的线性变换

$$\boldsymbol{y}=A\boldsymbol{x}$$

其中$A=[\boldsymbol{a}_1\ \ \boldsymbol{a}_2\ \ \cdots\ \ \boldsymbol{a}_p]^T=\begin{bmatrix}a_{11}&a_{12}&\cdots&a_{1p}\\a_{21}&a_{22}&\cdots&a_{2p}\\\vdots&\vdots&\ddots&\vdots\\a_{p1}&a_{p2}&\cdots&a_{pp}\end{bmatrix}$，$\boldsymbol{a}_i=\begin{bmatrix}a_{i1}\\a_{i2}\\\vdots\\a_{ip}\end{bmatrix}, \boldsymbol{a}_i^T\boldsymbol{a}_i=1, i=1,2,\cdots,p$。

现计算最大投影方差J。假设某一个单位方向向量$\boldsymbol{a}_i$，投影方差为：

$$\begin{aligned}J&=\frac{1}{N}\sum_{i=1}^{N}[(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^T\boldsymbol{a}]^2\\&=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{a}^T(\boldsymbol{x}_i-\bar{\boldsymbol{x}})(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^T\boldsymbol{a}\\&=\boldsymbol{a}^TS\boldsymbol{a}\end{aligned}$$

现在问题就转化为优化问题：

$$\begin{cases}\hat{\boldsymbol{a}}=arg\ \max\limits_{\boldsymbol{a}}\boldsymbol{a}^TS\boldsymbol{a}\\s.t.\ \boldsymbol{a}^T\boldsymbol{a}=1\end{cases}$$

使用拉格朗日乘子法，定义$L(\boldsymbol{a},\lambda)=\boldsymbol{a}^T S\boldsymbol{a}+\lambda(1-\boldsymbol{a}^T\boldsymbol{a})$。对$\boldsymbol{a}$求导并令导数为零，可以得到$S\boldsymbol{a}=\lambda\boldsymbol{a}$。可以看出$\boldsymbol{a}$即协方差矩阵的特征向量。又因为逐一选取方差最大方向等价于直接取最大的一些特征值，所以问题就转化为求协方差矩阵的特征向量。

下面简要推导特征值分解。令$\Lambda=\begin{bmatrix}\lambda_1 & 0 & 0 & 0\\ 0 & \lambda_2 & 0 & 0\\ 0 & 0 & \ddots & 0\\ 0 & 0 & 0 & \lambda_p\end{bmatrix}$为特征值矩阵，其中$\lambda_1\geq\cdots\geq\lambda_p\geq 0$，$P=[\boldsymbol{a}_1\ \ \boldsymbol{a}_2\ \ \cdots\ \ \boldsymbol{a}_p]$为特征向量矩阵，顺序与特征值矩阵对应。由$S\boldsymbol{a}=\lambda\boldsymbol{a}$，扩展到矩阵形式，即为$SP=P\Lambda$，$S=P\Lambda P^{-1}$。

计算出特征向量$\boldsymbol{a}_i$后，即可得到变换矩阵$A=[\boldsymbol{a}_1\ \ \boldsymbol{a}_2\ \ \cdots\ \ \boldsymbol{a}_p]^T$。对于当前数据集中的某一个线性相关的样本$\boldsymbol{x}_i$，经过变换得到线性无关的样本$\boldsymbol{y}_i=A(\boldsymbol{x}_i-\bar{\boldsymbol{x}})$。转化为矩阵形式，当前数据集为$X=[\boldsymbol{x}_1\ \ \boldsymbol{x}_2\ \ \cdots\ \ \boldsymbol{x}_N]^T$，设变换之后的数据集为$Y=[\boldsymbol{y}_1\ \ \boldsymbol{y}_2\ \ \cdots\ \ \boldsymbol{y}_N]^T$。由线性变换公式可以得到$Y^T=A(X^T-\bar{\boldsymbol{x}}\mathbf{1}_N^T)$，即$Y=(X-\mathbf{1}_N\bar{\boldsymbol{x}}^T)A^T$。

需要降维时，一般选择 q 个主成分（线性无关变量）来代替 p 个原有的线性相关的变量，使问题得以简化，并能保留原有变量的大部分信息。对于选择 q 个主成分的情况，变换公式为$\boldsymbol{y}_i=B(\boldsymbol{x}_i-\bar{\boldsymbol{x}})$，代入数据集，即为$Y=(X-\mathbf{1}_N\bar{\boldsymbol{x}}^T)B^T$，其中$B=[\boldsymbol{a}_1\ \ \boldsymbol{a}_2\ \ \cdots\ \ \boldsymbol{a}_q]^T$，为$q\times p$矩阵。

为了选择合适的主成分个数 q，需要定义第 k 主成分的方差贡献率$\eta_k=\dfrac{\lambda_k}{\sum_{i=1}^{p}\lambda_i}$，p 个主成分的累计方差贡献率$\sum_{i=1}^{q}\eta_i=\dfrac{\sum_{i=1}^{q}\lambda_i}{\sum_{i=1}^{p}\lambda_i}$。累计方差贡献率反映了主成分保留信息的比例。通常取 q 使得累计方差贡献率达到规定的百分比以上，如 70%~80%以上。

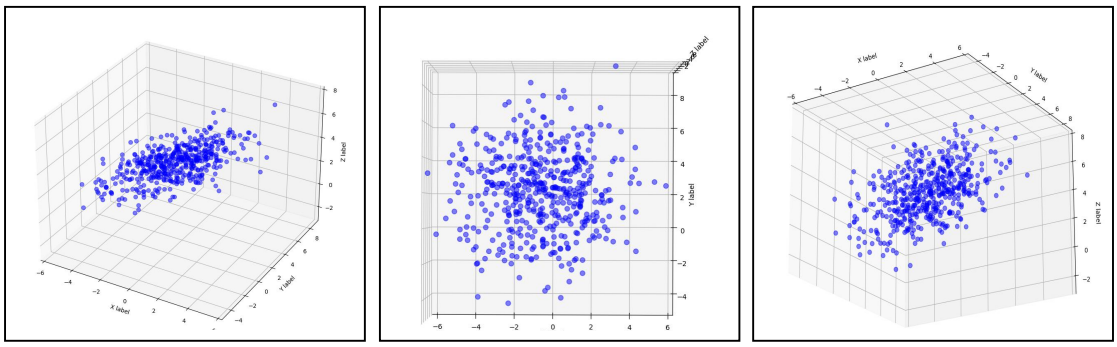
若要重建图像，由于$BB^T=I_q$，所以根据式$Y=(X-\mathbf{1}_N\bar{\boldsymbol{x}}^T)B^T$，$X=YB+\mathbf{1}_N\bar{\boldsymbol{x}}^T$，每一行即为一副图像经过降维之后的特征。将其重新修改尺寸为原来的大小，即可和原图计算信噪比。均方误差计算公式为：$MSE=\dfrac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\|I(i,j)-J(i,j)\|^2$。信噪比计算公式为：$PSNR=10\log_{10}\left(\dfrac{MAX_I^2}{MSE}\right)=20\log_{10}\left(\dfrac{MAX_I}{\sqrt{MSE}}\right)$，其中$MAX_I$表示图像点颜色的最大数值，在此问题中为 255。一般情况下，当 PSNR 大于 40 时，说明图像质量好，即与原图非常接近；大于 30 小于 40 时，说明图像质量较好，有失真但可以接受；大于 20 小于 30 时，说明图像质量差；小于 20 时，图像质量不可接受。

四、 实验结果与分析

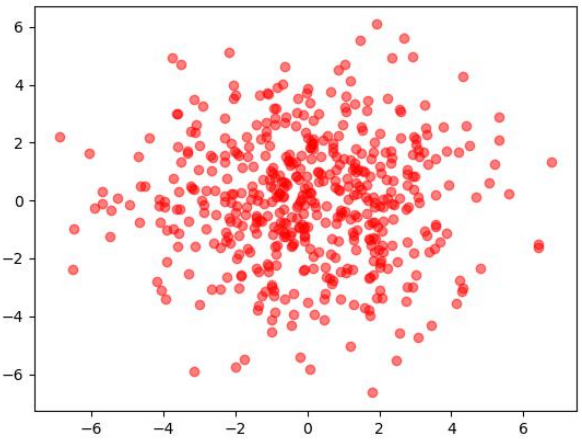
1. 使用自己生成的数据

1) 使用三维高斯分布的数据

假设样本量为 500，三个维度的均值分别为[1,2,3]，协方差矩阵为 $\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$，关于 y 轴旋转 30°，生成样本效果见下图。



可以看到某一方向上的方差较小，即旋转之前的 z 轴方向，因为协方差矩阵中 z 轴方向的样本方差较小。使用 PCA 进行降到 2 维，生成的结果如下：



从结果图可以看出，当降到 2 维时，两个维度分布较为均衡，原数据中方差最小的方向上的数据在降维过程中得到了一定程度上的忽略。根据输出结果，上述情况下，由三维降到二维，数据信息保留了 95.95%。

使用不同的协方差矩阵进行测试。

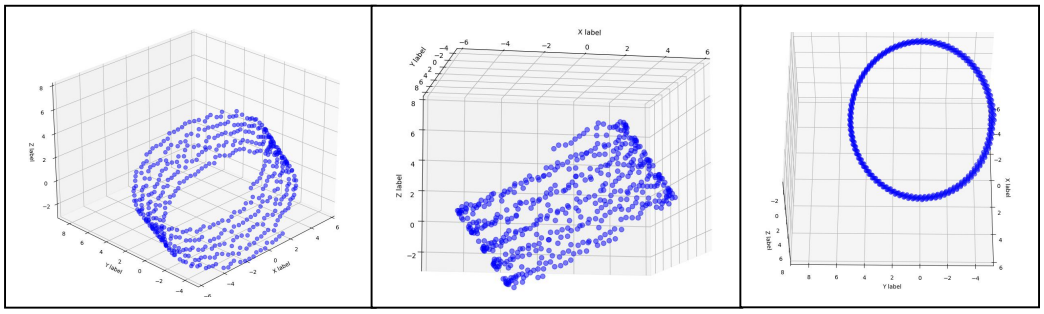
当协方差矩阵为 $\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix}$，即三个维度方差相等时，定性来说三个维度信息量相近，经过降维效果应当表现较差。经测试，由三维降到二维，数据信息保留了 70.34%。

当协方差矩阵为 $\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}$，同理分析并经测试，由三维降到二维，数据信息保留了 81.20%。

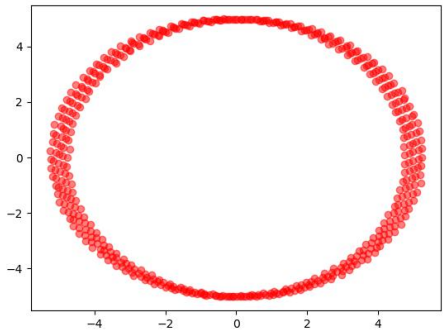
2) 使用三维螺旋线数据

假设样本量为 500，z 的范围为[-3,3]。在竖坐标上加方差为 0.2 的高斯噪声，关于 y

轴旋转 30°，生成样本效果见下图。

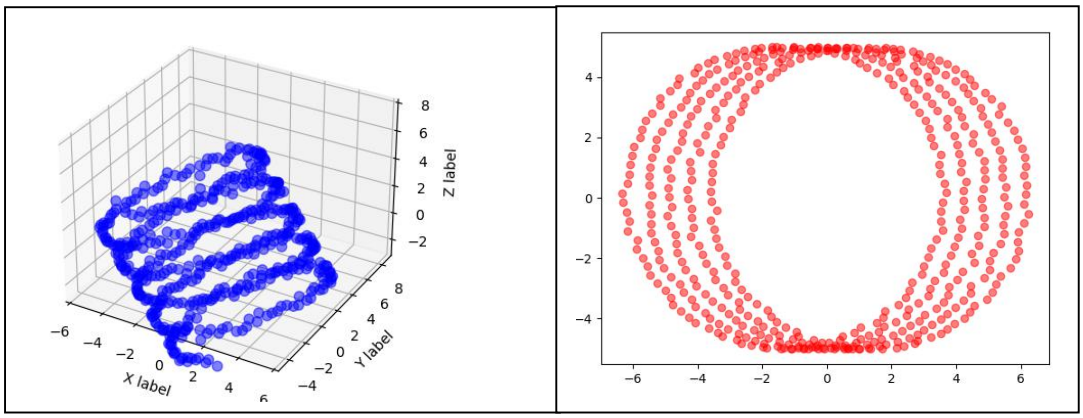


使用 PCA 降到 2 维，效果如下：



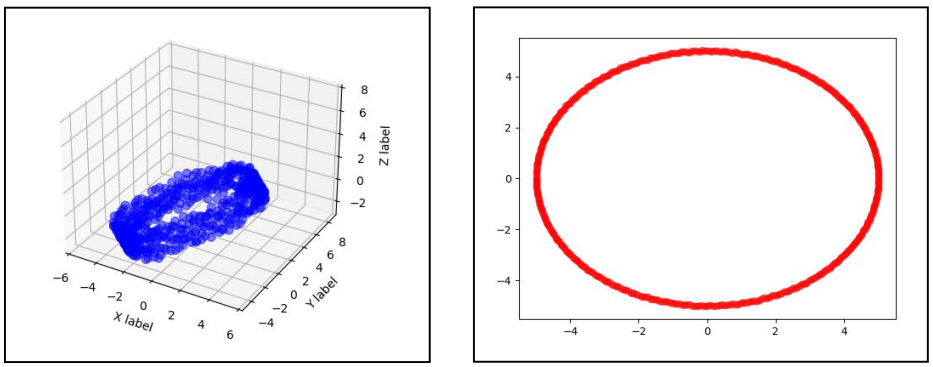
输出信息保留率为 89.29%。如果在生成数据时将旋转之前 z 的范围调整到[-5,5]或[-1,1]，根据理论，保留率应该分别降低和升高。实验结果如下。

$z \in [-5, 5]$：



信息保留率为 76.50%;

$z \in [-1, 1]$：



信息保留率为 98.55%。

2. 使用人脸数据

在网站上下载了灰度人脸数据集，选择了光线和视角较好的 38 张图片。由于计算机内存限制，难以对太大的矩阵进行操作，所以使用 opencv 将图片压缩到 42*48，效果如下。

分别使用至少 95%、90%、85%和 60%的信息保留率对图像进行降维压缩，并和原始图像计算信噪比。

考虑到图像较多，所以选择少量人脸进行展示。从左到右依次为源图像、95%、90%、85%和 60%。

明显看到从左到右依次变得模糊。

下面简要分析计算出的信噪比情况。

信噪比计算公式为：$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$，$MAX_I$ 表示图像点颜色的最大值，MSE 表示均方误差，公式为：$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\|I(i,j) - J(i,j)\|^2$。

在此问题中为 255。一般情况下，当 PSNR 大于 40 时，说明图像质量好，即与原图非常接近；大于 30 小于 40 时，说明图像质量较好，有失真但可以接受；大于 20 小于 30 时，说明图像质量差；小于 20 时，图像质量不可接受。

源图像的维度为 42*48=2016。对于保留率达到 95%的图像，降维之后的维度为 25，PSNR

都在 30 以上 40 以下，均值为 33.3。对于保留率达到 90%的图像，降维之后的维度为 18，PSNR 都在 29 以上 37 以下，均值为 31.1。对于保留率达到 85%的图像，降维之后的维度为 14，PSNR 都在 30 左右，均值为 30.3。对于保留率达到 65%的图像，降维之后的维度为 4，PSNR 都在 28 到 30，均值为 29.0。

在实验过程中，有时特征值会出现虚部，处理方式为保留实部，丢弃虚部，不影响结果。

五、 结论

主成分分析是一种常用的无监督学习方法，首先利用正交变换把由线性相关变量表示的观测数据转换为少数几个由线性无关变量表示的数据。计算中样本协方差矩阵进行了特征值分解，按照要求取前几个较大的特征值，其余舍去。一般来说，前几个特征值就占据了所有特征值之和的大部分，所以 PCA 能够在仅损失少量信息的情况下对数据进行较大程度的降维。

六、 参考文献

《统计学习方法》 李航

《机器学习》 周志华

七、 附录：源代码（带注释）

pca.py

```
1.  import numpy as np
2.
3.
4.  def cal_cov_matrix(X):
5.      """
6.      计算 X 的协方差矩阵
7.      :param X: 样本矩阵 N*M，N 个样本，每个样本 M 维
8.      :return: X 的协方差矩阵
9.      """
10.     n = X.shape[0]
11.     H = np.mat(np.identity(n)) - np.multiply(1 / n, np.mat(np.ones(n)).T @ np.ma
    t(np.ones(n)))
12.     return np.multiply(1 / n, X.T @ H @ X)
13.
14.
15. def PCA_solve(X, eta=float(1), k=0):
16.     """
17.     PCA 求解
18.     :param X: 样本矩阵
19.     :param eta: 保留信息比例
20.     :param k: 保留特征值数
21.     :return: 降维后中心化的样本 （N*降过之后的维度），变换矩阵，原样本均值
22.     """
23.     S = cal_cov_matrix(X)
```

```
    mean = np.multiply(1 / X.shape[0], X.T @ np.mat(np.ones(X.shape[0])).T)
    eigval, eigvec = np.linalg.eig(S)
    sorted_indices = np.argsort(eigval)
    eigvec = eigvec[:, sorted_indices[:-X.shape[1] - 1:-1]]
    eigval = sorted(eigval, reverse=True)
    cnt = 0
    sum_eigval = np.sum(np.array(eigval))
    pre_sum = 0
    if 1 > eta > 0 and k == 0:
        for i in range(len(eigval)):
            pre_sum += eigval[i]
            if pre_sum / sum_eigval > eta:
                cnt = i + 1
                break
    pre_sum = 0
    if k > 0:
        cnt = k
    for i in range(cnt):
        pre_sum += eigval[i]
    print('维度：'+str(cnt))
    print('信息保留百分比：' + str(np.real(pre_sum / sum_eigval)))
    B = eigvec.T[0]
    for i in range(1, cnt):
        B = np.vstack((B, eigvec.T[i]))
    B = np.real(B)
    return (X - mean.T) @ B.T, B, mean
```

mydata.py

```
import numpy as np
from pca import PCA_solve
import matplotlib.pyplot as plt


def Generate_Gaussian_data():
    """
    生成一组高斯分布的数据
    :return: 高斯数据矩阵
    """
    num = 500
    mean = [1, 2, 3]
    cov = [[5, 0, 0], [0, 5, 0], [0, 0, 2.5]]
    data = np.random.multivariate_normal(mean, cov, num)
    return np.mat(data)
```

```
16.
17.
18. def Generate_Spiral_data():
19.     """
20.     生成一组螺旋线数据
21.     :return: 螺旋线数据矩阵
22.     """
23.     num = 500
24.     theta = np.linspace(-5*np.pi, 5*np.pi, num)
25.     z = np.linspace(-1, 1, num)
26.     sigma = np.random.normal(loc=0, scale=0.2, size=num)
27.     z += sigma
28.     r = 5
29.     x = r * np.sin(theta)
30.     y = r * np.cos(theta)
31.     X = np.mat(x)
32.     X = np.vstack((X, np.mat(y)))
33.     X = np.vstack((X, np.mat(z)))
34.     return X.T
35.
36.
37. def rotate(X, axis, theta):
38.     """
39.     对数据 X 沿着某一个轴进行旋转一定的角度
40.     :param X: 数据矩阵
41.     :param axis: 轴
42.     :param theta: 旋转角度
43.     :return: 旋转之后的数据矩阵
44.     """
45.     if axis == 'x':
46.         rotate_matrix = np.mat([[1, 0, 0], [0, np.cos(theta), np.sin(theta)]
    , [0, -np.sin(theta), np.cos(theta)]])
47.         return X @ rotate_matrix.T
48.     elif axis == 'y':
49.         rotate_matrix = np.mat([[np.cos(theta), 0, -np.sin(theta)], [0, 1, 0
    ], [np.sin(theta), 0, np.cos(theta)]])
50.         return X @ rotate_matrix.T
51.     elif axis == 'z':
52.         rotate_matrix = np.mat([[np.cos(theta), np.sin(theta), 0], [-np.sin(
    theta), np.cos(theta), 0], [0, 0, 1]])
53.         return X @ rotate_matrix.T
54.     else:
55.         print('wrong')
56.         exit(0)
```

```
57.
58.
59. def show3D(X):
60.     """
61.     将数据在三维空间中绘制出来
62.     :param X: 数据
63.     :return: 无
64.     """
65.     plt.figure(1)
66.     ax1 = plt.axes(projection='3d')
67.     ax1.scatter3D(np.array(X)[:, 0], np.array(X)[:, 1], np.array(X)[:, 2], c
    ='b', marker='o', s=49, alpha=0.5)
68.     ax1.set_xlim(-6, 6)
69.     ax1.set_ylim(-5, 9)
70.     ax1.set_zlim(-3, 8)
71.     ax1.set_xlabel('X label')
72.     ax1.set_ylabel('Y label')
73.     ax1.set_zlabel('Z label')
74.
75.
76. def show2D(Y):
77.     """
78.     将数据在二维空间中绘制出来
79.     :param Y: 数据
80.     :return: 无
81.     """
82.     plt.figure(2)
83.     plt.scatter(np.array(Y)[:, 0], np.array(Y)[:, 1], c='r', marker='o', s=3
    6, alpha=0.5)
84.
85.
86. if __name__ == '__main__':
87.     # X = Generate_Gaussian_data()
88.     X = Generate_Spiral_data()  # 生成三维的数据
89.     X = rotate(X, 'y', np.pi / 6)
90.     Y, B, m = PCA_solve(X, k=2)  # 把三维的数据降到二维
91.     plt.ion()
92.     show3D(X)
93.     show2D(Y)
94.
95.     plt.ioff()
96.     plt.show()
```

pca_dataset.py

```python
import numpy as np
from pca import PCA_solve
import cv2 as cv


def compression(eta):
    """
    对人脸数据集进行压缩
    :param eta: 保留信息比例
    :return: 无
    """
    img0 = cv.imread('dataset/1.jpg', 0)
    X = np.mat(np.array(img0).flatten())
    for i in range(1, 38):
        file_path = 'dataset/' + str(i + 1) + '.jpg'
        img = cv.imread(file_path, 0)
        X = np.vstack((X, np.mat(np.array(img).flatten())))
    Y, B, m = PCA_solve(X, eta)
    x = Y @ B + m.T
    for i in range(38):
        img = x[i].reshape((48, 42))
        cv.imwrite('dataset_' + str(eta) + '/' + str(i + 1) + '.jpg', img)


def psnr(img1, img2):
    """
    计算峰值信噪比
    :param img1: 原图
    :param img2: 目的图像
    :return: 峰值信噪比
    """
    delta = img1 - img2
    mse = np.mean(np.multiply(delta, delta))
    ret = 20 * np.log10(255 / np.sqrt(mse))
    return ret


def cal_psnr():
    str2 = ''
    str3 = ''
    str4 = ''
    str5 = ''
    for i in range(0, 38):
        img1 = cv.imread('dataset/' + str(i + 1) + '.jpg', 0)
```

```
        img2 = cv.imread('dataset_0.95/' + str(i + 1) + '.jpg', 0)
        img3 = cv.imread('dataset_0.9/' + str(i + 1) + '.jpg', 0)
        img4 = cv.imread('dataset_0.85/' + str(i + 1) + '.jpg', 0)
        img5 = cv.imread('dataset_0.6/' + str(i + 1) + '.jpg', 0)
        ret2 = round(psnr(img1, img2), 4)
        ret3 = round(psnr(img1, img3), 4)
        ret4 = round(psnr(img1, img4), 4)
        ret5 = round(psnr(img1, img5), 4)
        str2 += str(i + 1) + '.jpg:  ' + str(ret2) + '\n'
        str3 += str(i + 1) + '.jpg:  ' + str(ret3) + '\n'
        str4 += str(i + 1) + '.jpg:  ' + str(ret4) + '\n'
        str5 += str(i + 1) + '.jpg:  ' + str(ret5) + '\n'

    f2 = open('dataset_0.95/psnr.txt', 'w')
    f2.write(str2)
    f3 = open('dataset_0.9/psnr.txt', 'w')
    f3.write(str3)
    f4 = open('dataset_0.85/psnr.txt', 'w')
    f4.write(str4)
    f5 = open('dataset_0.6/psnr.txt', 'w')
    f5.write(str5)


if __name__ == '__main__':
    compression(0.6)
    cal_psnr()
```