IE 529 Fall 2016 Assignment 2
Jifu Zhao Date: 09/29/2016

1 PCA

Following the procedures of PCA, the mean value is:

$$\vec{m} = [1.87275,\ 1.48783,\ 1.87275]$$

And the variance is:

$$variance = [2.38297,\ 0.234668,\ 2.54 \times 10^{-16}]$$

And the corresponding eigenvector is:

$$eig1 = [-0.6694\ \ -0.3220\ \ -0.6694]^T$$

$$eig2 = [0.2277\ \ -0.9467\ 0.2277]^T$$

$$eig3 = [-0.7071\ \ -1.4140 \times 10^{-15}\ 0.7071]^T$$

The de-biased dataset and corresponding eigenvectors are plotted in Figure 1.

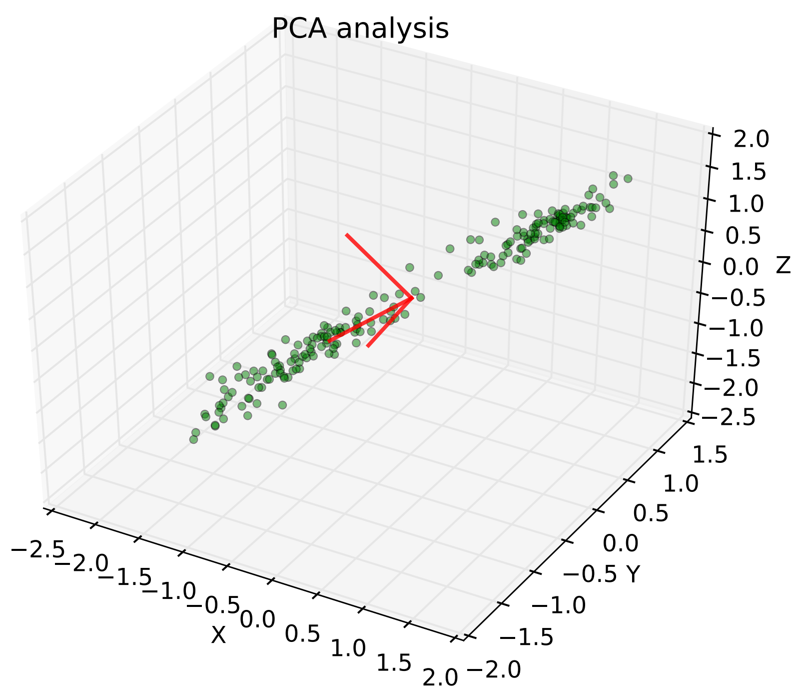

Figure 1: PCA eigenvector and scatter plot

The original dataset can be projected onto the first two principal components, the result is shown in Figure 2.

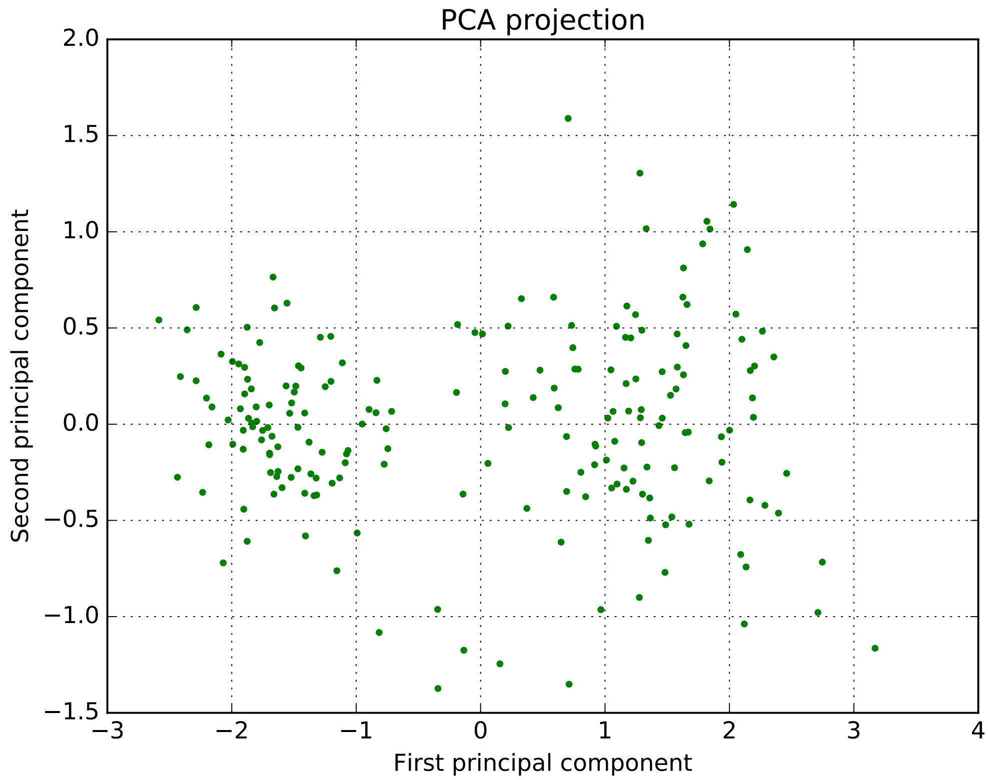

Figure 2: PCA Projection

2 Discussion

Currently the PCA is conducted through

$$[U, S, V] = SVD(A * A^T)$$

In fact this process can be simplified to:

$$[U', S', V'] = SVD(A)$$

Then, we can get that:

$$U = U'$$

$$S = S'^2$$

In this way, this process can be directly performed on the original data, rather than the new data.

The Python code is attached.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
__author__      = "Jifu Zhao"
__email__       = "jzhao59@illinois.edu"
__date__        = "09/29/2016"
"""

import warnings
warnings.simplefilter('ignore')
import copy
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def pca(data):
    """ function to perform PCA """
    data = copy.deepcopy(data)
    n, m = data.shape
    mean = np.mean(data, axis=0)
    data -= mean
    covariance = np.dot(data.T, data) / (n - 1)
    U, S, V = np.linalg.svd(covariance)
    return S, mean, U


def main():
    # PCA analysis
    data = pd.read_csv('./PCAdata.csv', header=None).values.T
    variance, mean, component = pca(data)
    project = np.dot(data - mean, component)
    data = data - mean

    print('Mean:\t', mean)
    print('Variance:\t', variance)
    print('Eigenvector 1\t', component[:, 0])
    print('Eigenvector 2\t', component[:, 1])
    print('Eigenvector 3\t', component[:, 2])

    # 3D plot
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    ax.plot(data[:, 0], data[:, 1], data[:, 2], 'o', markersize=4, color='green',
    alpha=0.5)
    for i in range(3):
        ax.plot([0, component[0, i]], [0, component[1, i]], [0, component[2, i]],
                color='red', alpha=0.8, lw=2)
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
    ax.set_title('PCA analysis')
    ax.view_init(40)
    fig.savefig('./result/3d.pdf')
    plt.show()

    # PCA projection
    fig, ax = plt.subplots()
    ax.plot(project[:, 0], project[:, 1], 'g.')
```

```
    ax.set_title('PCA projection')
    ax.set_xlabel('First principal component')
    ax.set_ylabel('Second principal component')
    ax.grid('on')
    fig.savefig('./result/projection.pdf')
    plt.show()


if __name__ == '__main__':
    main()
```