Dimensionality Reduction and Visualization
Exercise set 3 Solutions.

Md. Abdullah-Al Mamun

Part B: Linear Dimensionality Reduction

Problem B1: Mathematics of Principal Component Analysis

Given,

$Z = W^T X$

Here, W = projections
X = data
variance $W_{1T} W_1 = 1$

First Principal component variance is
$\text{Var}(W^T X) = W^T (\text{Var}(X))W_1 = W_{1T}EW_1 = W_{1T}EW_1$
$\text{Var}(W^T X) = W_{1T}EW_1 - q(1-1) = W_{1T}EW_1 - q(W_{1T}W_1 - 1) = W_{1T}EW_1 - qW_{1T}W_1$

By doing calculus:

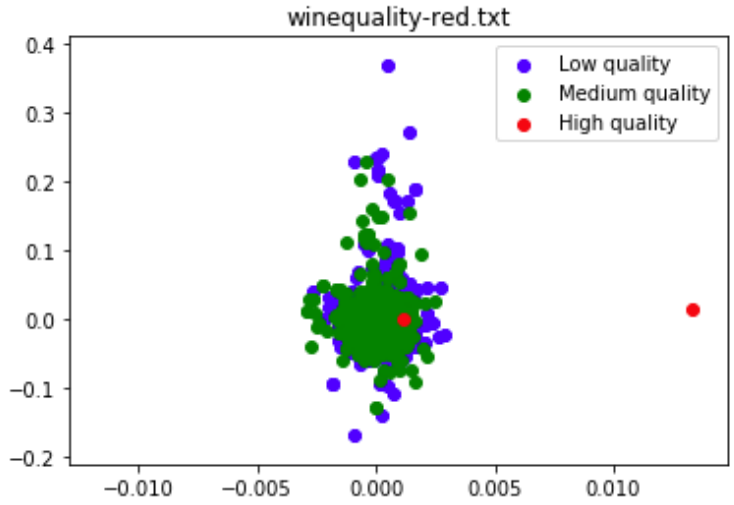
$EW_1 - q W_1 = 0$
Where $EW_1 = qW_1$
eigen value of E= Var(X)
W_1 is the eigen vector with eigen value q.
So:
$\text{Var}(W^T X) = W^T (\text{Var}(X))W_1 = W_{1T}EW_1 = q$

Variance is maximized with eigen value meaning largest eigen value maximize the variance.

Problem B2: Principal Component Analysis of Wines

Principal Component Analysis of Red Wine:

```
Variance [94.60795135  4.83483474]%
Component 1 [ 6.13296554e-03 -3.84670318e-04 -1.70762384e-04 -8.64864277e-03
 -6.37476516e-05 -2.18852809e-01 -9.75669835e-01 -3.72590009e-06
  2.67974074e-04 -2.23244233e-04  6.35985376e-03  4.31953676e-03]
Component 2 [-2.38646792e-02 -2.02021707e-03 -3.02675912e-03  1.11453593e-02
 -2.37525597e-04  9.75212313e-01 -2.18850408e-01 -2.50439091e-05
  3.26939011e-03  6.25945868e-04  1.46377527e-02  1.15350784e-02]
```




```
(6, 1081.42563558916)
(5, 109.34645676374501)
```

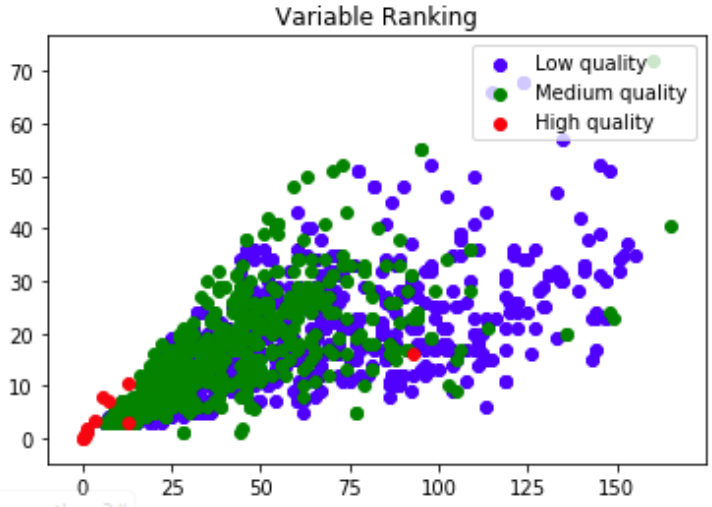
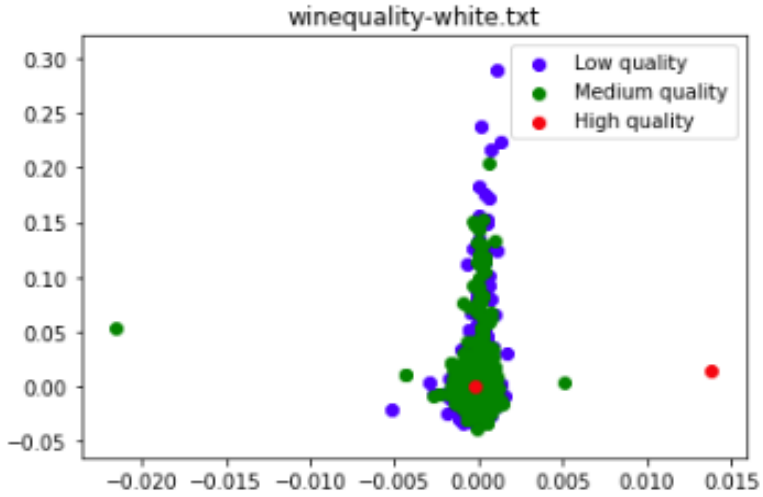


Figure 1: Above result shows that variance of 1st two components (94.607951354.83483474%). It also depicted top two variables (6, 1805.7167514665443) and (5, 289.18366676943).

High quality wines are very few, looks like outlier. Medium quality wines are looks like concentrated in the middle of the plot.

Principal Component Analysis of White wine:

```
Variance [90.9331225   7.93141114]%
Component 1 [-1.54452453e-03 -1.69030937e-04 -3.38646756e-04 -4.73275083e-02
 -9.75793989e-05 -2.61872279e-01 -9.63853329e-01 -3.59706391e-05
 -3.36199734e-06 -3.40888191e-04  1.25043553e-02  3.28041170e-03]
Component 2 [-9.16673296e-03 -1.54624759e-03  1.40367326e-04  1.49314295e-02
 -7.20390584e-05  9.64637649e-01 -2.62682018e-01 -1.83976939e-05
 -4.08057870e-05 -3.60533010e-04  6.47965595e-03  1.09933430e-02]
```




```
(6, 1805.7167514665443)
(5, 289.18366676943)
```

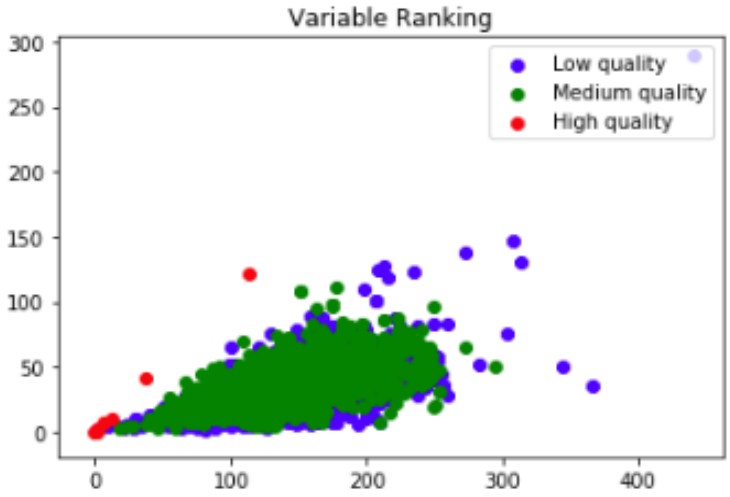


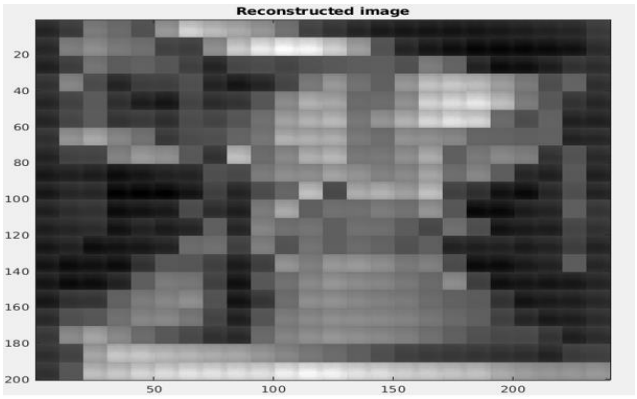
Figure 2: Above result shows that variance of 1st two components (90.9331225 7.93141114) %. It also depicted top two variables (6, 1805.7167514665443) and (5, 289.18366676943). We can clearly see the outlier for medium and high-quality wines.

Problem B3: Principal Component Analysis of an Image

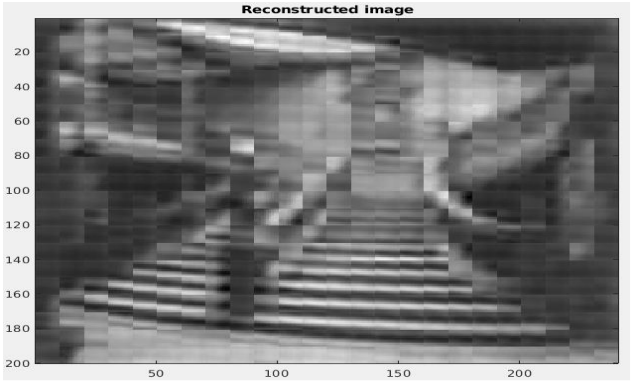
Firstly, principal Component Analysis (PCA) is a mathematical technique to reduce the dimensionality of data. It works on the principal of factoring matrices to extract the principal pattern of a linear system.

As the number of principal components increase, the more representative of the original image the reconstruction becomes.

The four images below represent the principal component for 5, 10, 20 ,30 PCAs of a grayscale picture of staircase. The images look very much alike but there are some differences in their respective qualities when enlarged. The 4th image will have the best image quality.



(1)



(2)

(3)

(4)

[Problem B2 and Problem B3 codes are in an Index below]

[Problem B4 codes is in musiccompress_template.m file inside the zip folder]

Problem B4: Principal Component Analysis of an Audio File

To load the sound files and to write them the code has been readjusted. Finally, different signals have been plotted. Matlab were used to solve problem 4.

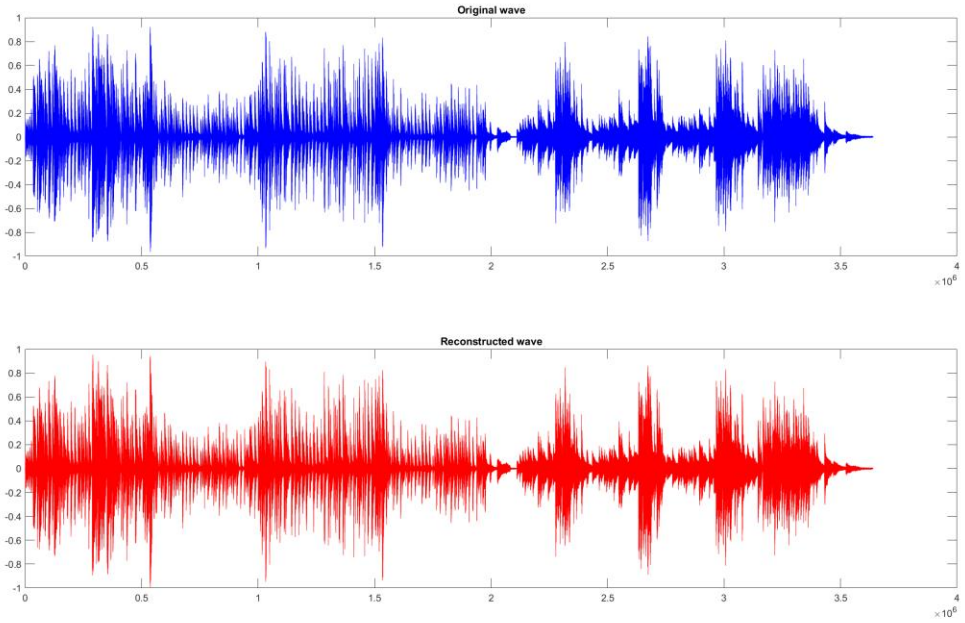


Figure 1: Reconstructed sound wave

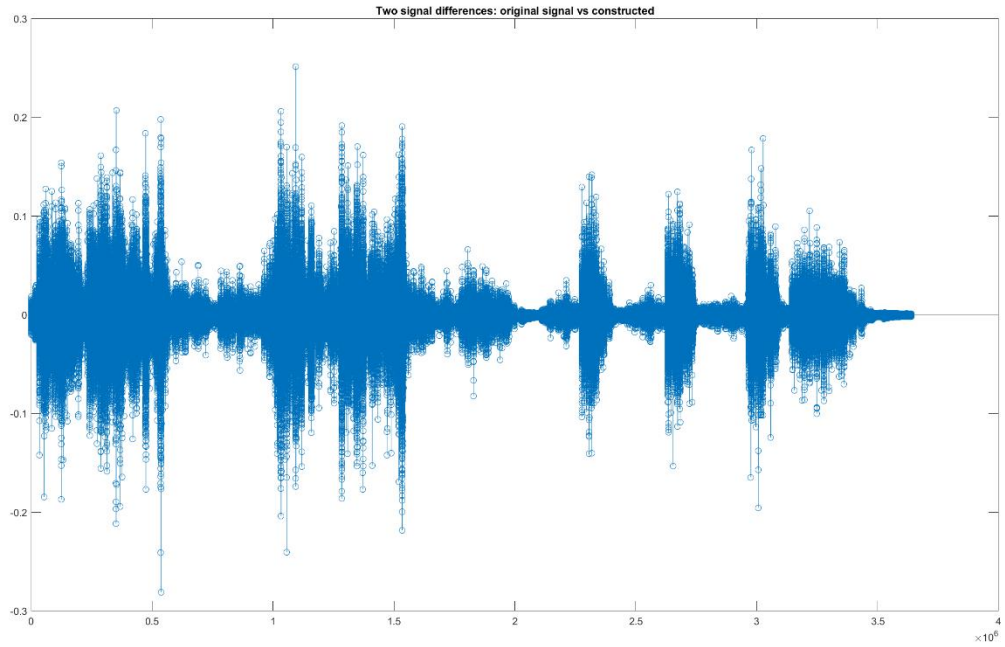


Figure 2: Signal differences between original and constructed signal

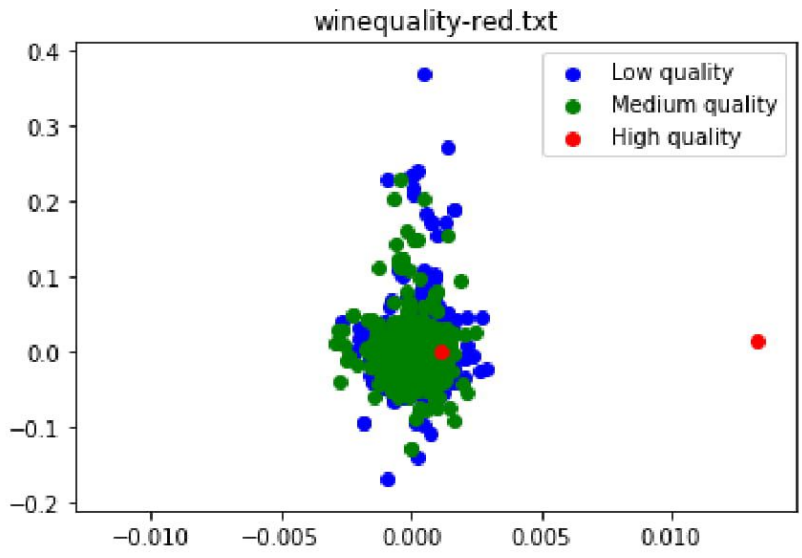
Index

In [10]:

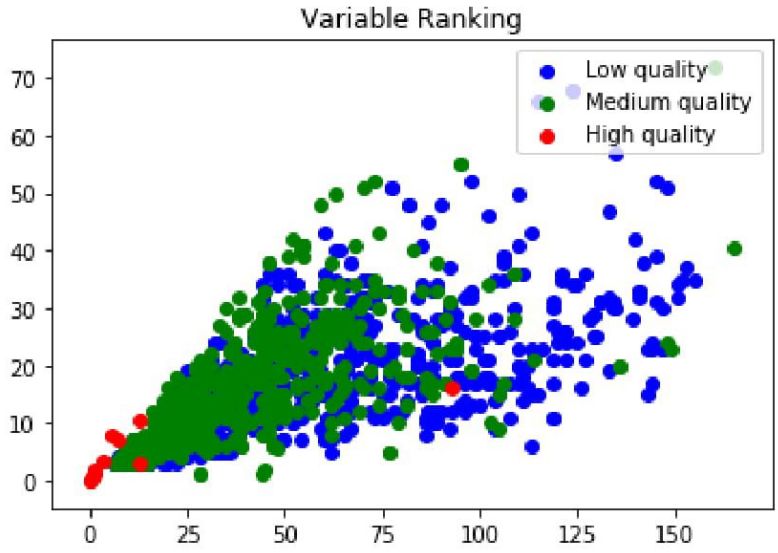
```
#Problem B2: Principal Component Analysis of Wines

import matplotlib.pyplot as plt
from numpy.linalg import eig
from numpy import cov

def PCA(data,n_components):
    data_file = np.loadtxt(data)
    D, V = eig(cov(data_file, rowvar=False))
    V = V[:, np.argsort(D)]

    X = data_file - np.tile(data_file.mean(axis=0), [data_file.shape[0], 1])
    D = sorted(D ,reverse=True)

    var=[]
    for k in range(n_components):
        var.append(D[k])
    print ("Variance",str(var/sum(D)*100)+'%')
    print ("Component 1",V[:,-1])
    print ("Component 2",V[:,-2])
    X_rot = np.matmul(X, V)
    transformed = X_rot[:,:n_components]
    low=[]
    medium=[]
    high=[]
    labels=["Low quality","Medium quality","High quality"]
    for i in range(data_file.shape[0]):
        if data_file[i][11]<=5:
            low.append(i)
        elif data_file[i][11]==6:
            medium.append(i)
        elif data_file[i][11] >=7:
            high.append(i)
    plt.scatter(transformed[low,0],transformed[low,1],label=labels[0],color='blue')
    plt.scatter(transformed[medium,0], transformed[medium,1], label=labels[1],color=
    plt.scatter(transformed[high][0], transformed[high][0], label=labels[2],color='r
    plt.legend(loc='upper right',ncol=1,fontsize=10)
    plt.title(data)
    plt.show()


def VariableRanking(data):
    var=[]
    low = []
    medium = []
    high = []
    data_file=np.loadtxt(data)
    for i in range(data_file.shape[1]):
        a=(i,np.var(data_file[:,i]))
        var.append(a)
    var = sorted(var ,key=lambda variance:variance[1],reverse=True)
    for i in range(2):
        print (var[i])
    labels = ["Low quality", "Medium quality", "High quality"]
```

```
    for i in range(data_file.shape[0]):
        if data_file[i][11]<=5:

            low.append(i)
        elif data_file[i][11]==6:

            medium.append(i)
        elif data_file[i][11] >=7:

            high.append(i)
    plt.scatter(data_file[low, 6], data_file[low, 5], label=labels[0],color='blue')
    plt.scatter(data_file[medium, 6], data_file[medium, 5], label=labels[1],color='g
    plt.scatter(data_file[high][6], data_file[high][5], label=labels[2],color='red')
    plt.legend(loc='upper right',ncol=1,fontsize=10)
    plt.title("Variable Ranking")
    plt.show()

def Main():
    file1="winequality-red.txt"
    file2="winequality-white.txt"
    n_components=2
    PCA(file1,n_components)
    VariableRanking(file1)

Main()
```

```
Variance [94.60795135  4.83483474]%
Component 1 [ 6.13296554e-03 -3.84670318e-04 -1.70762384e-04 -8.648642
77e-03
 -6.37476516e-05 -2.18852809e-01 -9.75669835e-01 -3.72590009e-06
  2.67974074e-04 -2.23244233e-04  6.35985376e-03  4.31953676e-03]
Component 2 [-2.38646792e-02 -2.02021707e-03 -3.02675912e-03  1.114535
93e-02
 -2.37525597e-04  9.75212313e-01 -2.18850408e-01 -2.50439091e-05
  3.26939011e-03  6.25945868e-04  1.46377527e-02  1.15350784e-02]
```




```
(6, 1081.42563558916)
(5, 109.34645676374501)
```

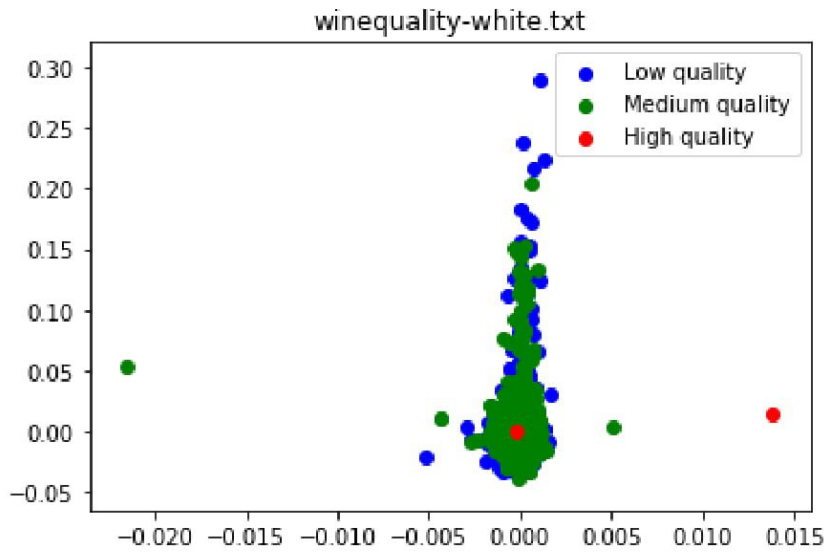

Variable Ranking
Low quality
Medium quality
High quality
70
60
50
40
30
20
10
0
0
25
50
75
100
125
150

In [11]:

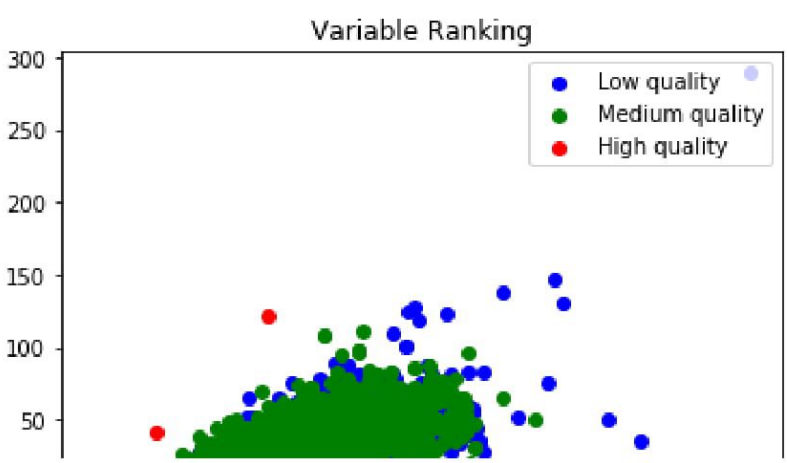
```
# Problem B2: Principal Component Analysis of Wines

import matplotlib.pyplot as plt
from numpy.linalg import eig
from numpy import cov

def PCA(data,n_components):
    data_file = np.loadtxt(data)
    D, V = eig(cov(data_file, rowvar=False))
    V = V[:, np.argsort(D)]

    X = data_file - np.tile(data_file.mean(axis=0), [data_file.shape[0], 1])
    D = sorted(D ,reverse=True)

    var=[]
    for k in range(n_components):
        var.append(D[k])
    print ("Variance",str(var/sum(D)*100)+'%')
    print ("Component 1",V[:,-1])
    print ("Component 2",V[:,-2])
    X_rot = np.matmul(X, V)
    transformed = X_rot[:,:n_components]
    low=[]
    medium=[]
    high=[]
    labels=["Low quality","Medium quality","High quality"]
    for i in range(data_file.shape[0]):
        if data_file[i][11]<=5:
            low.append(i)
        elif data_file[i][11]==6:
            medium.append(i)
        elif data_file[i][11] >=7:
            high.append(i)
    plt.scatter(transformed[low,0],transformed[low,1],label=labels[0],color='blue')
    plt.scatter(transformed[medium,0], transformed[medium,1], label=labels[1],color=
    plt.scatter(transformed[high][0], transformed[high][0], label=labels[2],color='r
    plt.legend(loc='upper right',ncol=1,fontsize=10)
    plt.title(data)
    plt.show()


def VariableRanking(data):
    var=[]
    low = []
    medium = []
    high = []
    data_file=np.loadtxt(data)
    for i in range(data_file.shape[1]):
        a=(i,np.var(data_file[:,i]))
        var.append(a)
    var = sorted(var ,key=lambda variance:variance[1],reverse=True)
    for i in range(2):
        print (var[i])
    labels = ["Low quality", "Medium quality", "High quality"]
```

```
    for i in range(data_file.shape[0]):
        if data_file[i][11]<=5:
            low.append(i)
        elif data_file[i][11]==6:
            medium.append(i)
        elif data_file[i][11] >=7:
            high.append(i)
    plt.scatter(data_file[low, 6], data_file[low, 5], label=labels[0],color='blue')
    plt.scatter(data_file[medium, 6], data_file[medium, 5], label=labels[1],color='g
    plt.scatter(data_file[high][6], data_file[high][5], label=labels[2],color='red')
    plt.legend(loc='upper right',ncol=1,fontsize=10)
    plt.title("Variable Ranking")
    plt.show()

def Main():
    file1="winequality-red.txt"
    file2="winequality-white.txt"
    n_components=2
    PCA(file2,n_components)
    VariableRanking(file2)

Main()
```

```
Variance [90.9331225   7.93141114]%
Component 1 [-1.54452453e-03 -1.69030937e-04 -3.38646756e-04 -4.732750
83e-02
 -9.75793989e-05 -2.61872279e-01 -9.63853329e-01 -3.59706391e-05
 -3.36199734e-06 -3.40888191e-04  1.25043553e-02  3.28041170e-03]
Component 2 [-9.16673296e-03 -1.54624759e-03  1.40367326e-04  1.493142
95e-02
 -7.20390584e-05  9.64637649e-01 -2.62682018e-01 -1.83976939e-05
 -4.08057870e-05 -3.60533010e-04  6.47965595e-03  1.09933430e-02]
```




```
(6, 1805.7167514665443)
(5, 289.18366676943)
```


Variable Ranking
300
250
200
150
100
50
Low quality
Medium quality
High quality

In [ ]:

```
#Problem B3: Principal Component Analysis of an Image

import numpy as np
import random
import operator

import matplotlib.pyplot as plt
from PIL import Image
from numpy.linalg import eig
from numpy import cov
from pylab import imread,subplot,imshow,title,gray,figure,show,NullLocator

def LoadImage():
    figure = Image.open("staircase.png")
    fig = np.array(figure)

    im_width,im_height =image.size
    pixel_block = 10

    Features=[]
    for i in range(0,im_height,pixel_block):
        for j in range(0,im_width,pixel_block):
            end_row = i+pixel_block
            end_column = j+pixel_block
            Features.append(fig[end_row-pixel_block:end_row,end_column-pixel_block:e
    Features=np.array(Features)
    Features=Features.reshape(Features.shape[0],Features.shape[1]*Features.shape[2])
    return(Features,img)

def PCA(Features,image,n_components):
    D, V = eig(cov(Features, rowvar=False))
    V = V[:, np.argsort(D)]
    print(max(D))

    X = Features - np.tile(Features.mean(axis=0), [Features.shape[0], 1])
    D = sorted(D, reverse=True)
    var = []

    for k in range(3):
        var.append(D[k])
    print (np.sum(var)/np.sum(D)*100)
    Z=np.matmul(X, V)
    num=1

    for component in n_components:
        T = Z[:,V.shape[0]-component:V.shape[0]]
        E = V[:, V.shape[0]-component:V.shape[0]]
        R = np.dot(T,np.transpose(E)) + np.mean(Features,axis=0)

        pixel_block = 10
        im_height= 200
        im_width=240
        Reconstructed_image = np.zeros(shape=(200,240))
        x=0
```

```
        for i in range(0, im_height, pixel_block):
            for j in range(0, im_width, pixel_block):
                end_row = i + pixel_block
                end_column = j + pixel_block
                Reconstructed_image[end_row-pixel_block:end_row,end_column-pixel_blc
                x+=1

        ax = subplot(3, 2, num, frame_on=False)
        ax.xaxis.set_major_locator(NullLocator())
        ax.yaxis.set_major_locator(NullLocator())
        num += 1
        plt.imshow(np.flipud(Reconstructed_image),cmap='gray',origin='lower')
        plt.title('PCs # ' + str(component))
        plt.show()

def Main():
    n_components=[5,10,20,30]
    Features,image = LoadImage()
    PCA(Features,image,n_components)

Main()
```