STA 135 Project

Name : Bohao Zou

ID Number : 917796070

1. Introduction

The Principal Component analysis (PCA) processing can decompose the sample covariance matrix and get the eigenvector and eigenvalues. By using those eigenvalues and eigenvector, we can reduce the dimension of original data and retain most of information in original data. Therefore, it is a advantage tool for visualization data and extract the main features from original data without using any supervised labels. Thus, PCA procedure is an unsupervised learning.

In this project, i want to explore what will happen if we input an image (Like cat or dog) into PCA processing and reduce the columns of the image. The rows of this image will be the same as original image. This is because we treat the rows as samples number and the columns as x_i variables. $i = 1, .., n$

2. Summary

2.1 Analysis plan

How computer storage an image? There are 3 channels in one image. Its are Red, Green and Blue. There is an $n * p$ matrix in one of channels. So, there are 3 matrix in one image. In this project, i treat the row of matrix as samples and columns as x_i variables. I input those three matrix (R,G,B) into PCA procedure and get three matrix which has processed by PCA. The features which will be remained will be manipulated by users.Because the pixel value in one image is from 0 to 255. For preventing overflow 255, i will use Min-Max normalization to each columns and then multiply 255. Finally, i combine those matrix into one image and visualize the processed PCA image. In this project, i input an dog image into the PCA procedure. The dog picture is this :

Figure 1: The dog image which will be used in this project

2.2 plots

At this part, i will visualize some distributions of x_i variables. I do not know how to interpret those x_i variables because its just are the pixel value in one of image.

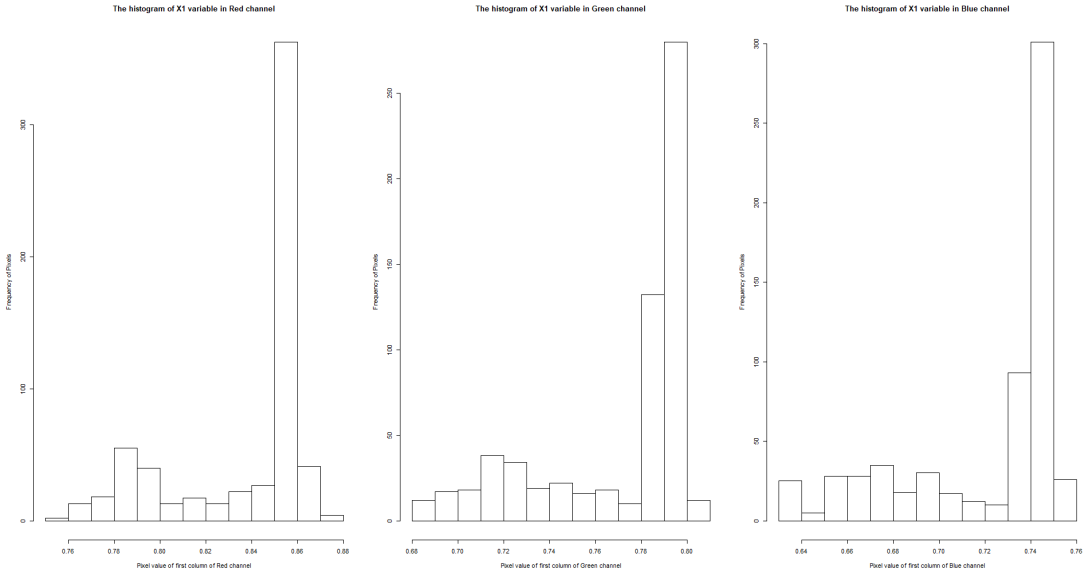

Figure 2: The distrubutions of some x variables in the dog image.(A). The first plot is the distribution of the first column in the Red channel matrix. (B). The second plot is the distribution of the first column in the Green channel matrix. (C). The third plot is the distribution of the first column in the Blue channel matrix.

From the plot we can know that the distribution of each x_1 varibale in each channel is roughly the same. Most of values are concentrated on the interval between 0.7 and 1.0.

3. Analysis

At this part, i will introduce how dose the PCA procedure work.

At the beginning, let's suppose S be the sample covariance matrix. We will make a **spectral decomposition** on this sample covariance matrix and we will get p number of eigenvalues ($\lambda_i \ i = 1, ..., p$) and corresponding eigenvectors ($V_i, \ i = 1, ..., p$). We sort those eigenvalues from big to small. The largest eigenvalue is the variance of first principle component of this sample covariance matrix. The second largest eigenvalue is the variance of second principle component and so on.

The corresponding k-th principle component is : $Y_k = v_{k1} * x_1 + ... + v_{kp} * x_p, \ k = 1, ..., p$

Finally, if we have selected m, ($m < p$) principle components, we will use $Y = X * [V_1, V_2, ..., V_m]$ to calculate the output Y. (X is the data matrix which row represents samples and column represents x_i variables.)

For building the function of PCA in R. I used two strategies. The first is you can input how many information you would to remain in the output matrix Y. This method will select principle components automatically. The second way is that you can assign the number of principle components which you would like to remain in output Y. In this project, i used the second method. Because i need to fix the number of principle components.

4. Conclusion

The dimensions of Figure 1 is 628 by 1100. This means that there are 628 samples and 1100 x variables. I selected half of variables. It is 550. So, in the final output, the matrix dimensions is 628 by 550.

This is the final output. The matrices of Red, Green and Blue channels are processed by PCA separately and conbined into one image at last. This image contains 99.99985% information of figure 1.

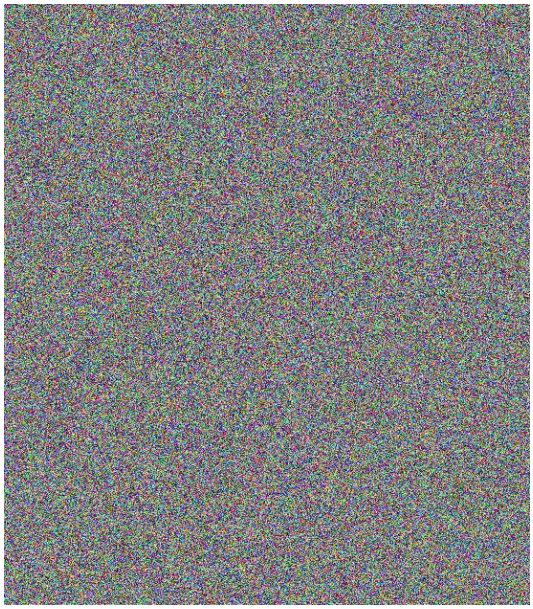

Figure 3: The dog image which has been processed by PCA

5. Appendix

R CODE

Parameters introduce

data : is n x p data frame

informationSaveRatio : is the number between [0,1]. How many information would you want to remain in final output.

manualSelectFeature : if selecting principle componets by user.

selectedNumber : if the parameter **manualSelectFeature** is TRUE, then how many principle components would you select.

```
PCA = function(data,informationSaveRatio = 0.9,manualSelectFeature = FALSE, selectedNumber = 2){
sampleCovariance = cov(data)
eigInfor = eigen(sampleCovariance)
eigVectors = eigInfor$vectors
eigValues = eigInfor$values
if (manualSelectFeature == FALSE){
```

```
totalVariance = sum(eigValues)

add = 0

k = 0

for (i in c(1:length(eigValues))){

  add = add + eigValues[i]

  if (add / totalVariance >= informationSaveRatio){

    k = i

    break()

  }

}

savedEigVectors = eigVectors[,1:k]

return(data %*% savedEigVectors)

}
else{

savedEigVectors = eigVectors[,1:selectedNumber]

return(data %*% savedEigVectors)

}
}
normalization = function(data){
sampleNumber = dim(data)[1]
features = dim(data)[2]
matrixR = c(1:sampleNumber)
for (i in c(1:features)){

minz = min(data[,i])

maxz = max(data[,i])

matrixR = cbind(matrixR,(data[,i] - minz) / (maxz - minz))

}
return(matrixR[,-1])
}
```

```
imagePath = “F:\Dog.jpg”

PCA_Save_Path = “F:\Dog_PCA.jpg”

scaleR = 2

library(“jpeg”)

img = readJPEG(imagePath)

RMatrix = img[„1]

GMatrix = img[„2]

BMatrix = img[„3]

sampleNumber = dim(RMatrix)[1]

featureNumber = dim(RMatrix)[2]

RPCA = normalization(PCA(RMatrix,manualSelectFeature = T,selectedNumber = round(featureNumber / scaleR)))

GPCA = normalization(PCA(GMatrix,manualSelectFeature = T,selectedNumber = round(featureNumber / scaleR)))

BPCA = normalization(PCA(BMatrix,manualSelectFeature = T,selectedNumber = round(featureNumber / scaleR)))

imgPCA = array(c(RPCA,GPCA,BPCA),dim = c(sampleNumber,round(featureNumber / scaleR),3)) * 255.0

writeJPEG(imgPCA,target = PCA_Save_Path)

Rvector = RMatrix[,1]

Gvector = GMatrix[,1]

Bvector = BMatrix[,1]

par(mfrow=c(1,3))

hist(Rvector,xlab = “Pixel value of first column of Red channel”,ylab = “Frequency of Pixels”, main = “The histogram of X1 variable in Red channel”)

hist(Gvector,xlab = “Pixel value of first column of Green channel”,ylab = “Frequency of Pixels”, main = “The histogram of X1 variable in Green channel”)

hist(Bvector,xlab = “Pixel value of first column of Blue channel”,ylab = “Frequency of Pixels”, main = “The histogram of X1 variable in Blue channel”)
```