Principal Component Analysis

Implementing PCA on Building Dataset and Testing the Results

Principal Component Analysis

In PCA, we first calculate variation in each one of the attributes have. Chances are that same amount of variation can also be explained by some other column. Hence, we can keep only those columns that are explanation big amount of variation and remove those columns that are not explaining that much variation in data since.

After calculating best fit line for each one of the column, we calculate the distance, or the sum of least square error. That value of distance is calculated from the origin. So if the value of least square distance from origin to points is highest for some column, then that column is explanation the most variation in the data.

So, once we create a 3d plot, we have the choice of selecting any two of the axes. Those two axes must be the one that are explaining the most variation of 3d plain 2d plain. Only Minimal amount of information will be lost only.

I will keep y-z axis in this case.

Introduction

Principal component analysis is a technique to reduce the number of dimension in our dataset at the expense of loss of very minimal information. This technique is used particularly for data with very high dimensionality.

When we draw data on a graph, we can visualize the variation along the axis. That variation is called Eigen value of that axis. The higher the Eigen value higher the variation that axis is enplaning in data.

Hence, we calculate variation in data along all the different axes we have. Chances are, that almost 905 or 95% of information can be explained by only a few Eigen values (PCs). That is why we only keep those column that are explaining big amount of variation and remove the columns that are not explaining much value in data.

Data Description

Data name: Rooftop

The data is collected during building assessment based on 8 different features. The last two columns are there in dataset for prediction purposes, but we are going to discuss PCA and we are not predicting values using PCA. Moreover, to simply the discussing I have removed many columns and has just kept 5 columns.

These 5 columns are termed as

X1: Relative compactness
X2: Surface area of building
X3: Wall area
X4: Roof Area
X5: Overall height

All 5 values are continuous and none of them is categorical which is an essential element of PCA is.

Step1.

```
#step 1
df <- file.choose() #file name is rooftop
df <- read.csv(df) # convert the file into csv
str(df) # check the stuctur of data
df <- df[,-c(9,10,8,7,6,11,12)]
str(df)
summary(df)
df <- na.omit(df) # removing na
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Loading the file and copying it into variable df as csv file. After which we check the structure of the data frame. It looks like this:

```
 str(df) # check the stuctur of data
data.frame': 1296 obs. of  12 variables:
$ X1 : num  0.98 0.98 0.98 0.98 0.9 0.9 0.9 0.9 0.86 0.86 ...
$ X2 : num  514 514 514 514 564 ...
$ X3 : num  294 294 294 294 318 ...
$ X4 : num  110 110 110 110 122 ...
$ X5 : num  7 7 7 7 7 7 7 7 7 7 ...
$ X6 : int  2 3 4 5 2 3 4 5 2 3 ...
$ X7 : num  0 0 0 0 0 0 0 0 0 0 ...
$ X8 : int  0 0 0 0 0 0 0 0 0 0 ...
$ Y1 : num  15.6 15.6 15.6 15.6 20.8 ...
$ Y2 : num  21.3 21.3 21.3 21.3 28.3 ...
$ X  : logi  NA NA NA NA NA NA ...
$ X.1: logi  NA NA NA NA NA NA ...
```

We see that last two additional columns has been automatically added during reading phase of file and has been induced with NAs. I have deleted all the columns above X5 for simplicity of concept.

Then we use the summary command to get better understanding of data.

```
> summary(df)
       X1               X2              X3              X4              X5              X6              X7
 Min.   :0.6200   Min.   :514.5   Min.   :245.0   Min.   :110.2   Min.   :3.50   Min.   :2.00   Min.   :0.0000
 1st Qu.:0.6825   1st Qu.:606.4   1st Qu.:294.0   1st Qu.:140.9   1st Qu.:3.50   1st Qu.:2.75   1st Qu.:0.1000
 Median :0.7500   Median :673.8   Median :318.5   Median :183.8   Median :5.25   Median :3.50   Median :0.2500
 Mean   :0.7642   Mean   :671.7   Mean   :318.5   Mean   :176.6   Mean   :5.25   Mean   :3.50   Mean   :0.2344
 3rd Qu.:0.8300   3rd Qu.:741.1   3rd Qu.:343.0   3rd Qu.:220.5   3rd Qu.:7.00   3rd Qu.:4.25   3rd Qu.:0.4000
 Max.   :0.9800   Max.   :808.5   Max.   :416.5   Max.   :220.5   Max.   :7.00   Max.   :5.00   Max.   :0.4000
 NA's   :528      NA's   :528     NA's   :528     NA's   :528     NA's   :528    NA's   :528    NA's   :528
       X8              Y1              Y2              X               X.1
 Min.   :0.000   Min.   : 6.01   Min.   :10.90   Mode:logical   Mode:logical
 1st Qu.:1.750   1st Qu.:12.99   1st Qu.:15.62   NA's:1296      NA's:1296
 Median :3.000   Median :18.95   Median :22.08
 Mean   :2.812   Mean   :22.31   Mean   :24.59
 3rd Qu.:4.000   3rd Qu.:31.67   3rd Qu.:33.13
 Max.   :5.000   Max.   :43.10   Max.   :48.03
 NA's   :528     NA's   :528     NA's   :528
```

We see that other than last two columns, all other columns have 528 missing values, so our next step is to remove all those rows that have these NA values. I used na.omit command to remove all rows with NA values.

Step2.

```
#step 2
r <- sample(1:768,.50*768) # generating random sample

df["color"] <- "0"    # creating a extra column in data frame

df$color[r] <- "red"    # assiging 50% random values to be red
 df$color[-r] <- "green"  #assigning other 50% values to be green

 table(df$color) # to check the result are 50-50
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

First I calculate a random sample in order to use it to impute “red” and “green” value in an additional column. After imputing the random values in color column I used table command to see if they are equally distributed.

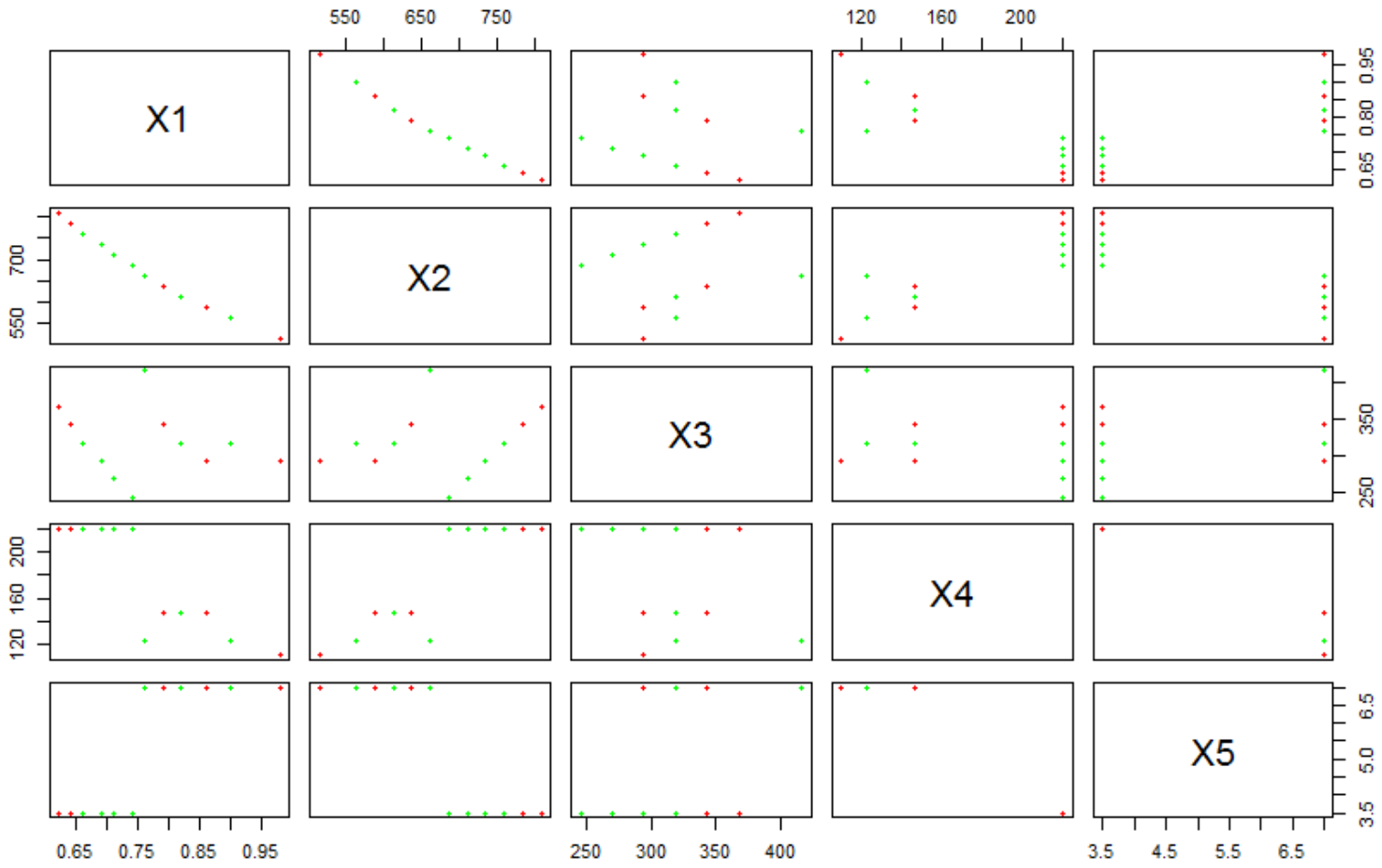
```
> table(df$color) # to check the result are 50-50

green   red
  384   384
>
```

Step3.

```
#step 3
measure <- df[,1:5] # extracting continous values
color <- df[,6] # extracting categorical values
plot(measure,col=color,pch=19,)  # plotting the cross sectional graph
```

Then I separated the continuous values and stored them into variable called measure, and stored the categorical value of colors into variable called color.

Using plot command I drew a cross sectional graph to see how each one of those attributes corresponds with each other. The graph looks like this.

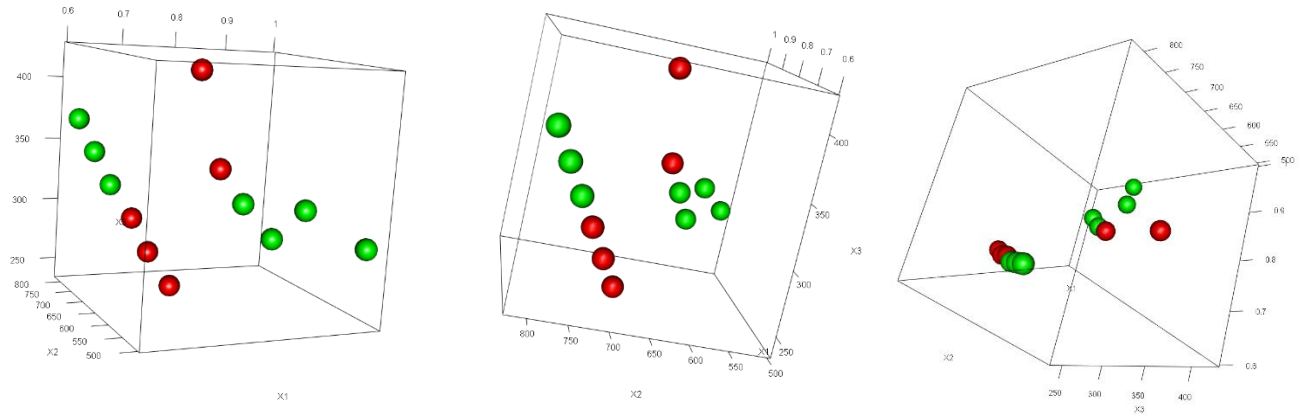


I see that x1, x2 have negative correlation. Meaning, compactness of building will decreases if the surface area of building increases. On other hand, X2 and X3 also have certain amount of liner relation. Meaning, if the surface area of building is high, chances are the wall area will also be high. Whereas x4 and x5 does not seems to have much correlation with X1 and X2. Meaning, roof area and height of building is not correlated with compactness and surface area.

Step4.

```
#step4
install.packages("rgl")
library(rgl)
plot3d(measure,type="s",col=color)
```

I installed a package called “rgl” which is used for 3d graphics on R studio. After loading the package I used plot3d command to create a 3D graph of the variable “measure”. Note that it is only taking 3 column to create a 3d graph.



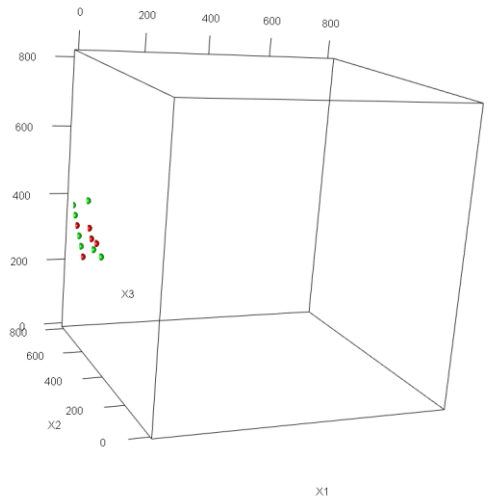
Looks like points are going down in x3 vs x1 plain. I have shown this visualization with three different directions to get intuitive understanding of data point sin space.

Step5.

```
#step5
lims <- c(min(measure),max(measure))
plot3d(measure,type="s",col=color,xlim = lims,ylim = lims,zlim = lims)
sapply(measure,mean)
```

We can also see in the 3d graph above that all the axis have different scale. For example, x2(surface area of building) is measure on scale of 500-800 while X1 (compactness) have a scale of 0.6 to 1.

So, to make all the dimension spread on same scale I am using limits that defines the minimum and maximum value each of the axis should lie into. Doing that will show us this result:



We can see that the dots appears on the left side of the 3d plot. That is because the X1 axis initially had axis between .6 and 1 and scaling it up to 800 would make those dots look smaller. This graph also depicts that the dots seems to be going down across X3 axis. Where x1 is compactness, x2 is surface area of building and x3 is wall area.

To get the picture even clearer, we can find the mean of each one of the columns and observe the difference between their scaling.

```
> sapply(measure,mean)
        X1          X2          X3          X4          X5
 0.7641667 671.7083333 318.5000000 176.6041667   5.2500000
>
```

It is quite clear that x1 (compactness) and x2(surface area) have a very big difference between their values. Meaning, this is creating a scaling problem and for PCA they all has to lie on one particular scale.

Step6.

```
#step6
centered_measure <- scale(measure,center = T,scale = F) # cernteralizing the data not nomalizing,

summary(centered_measure)

lims <- c(min(centered_measure),max(centered_measure))
plot3d(centered_measure,type="s",col=color,xlim = lims,ylim = lims,zlim = lims)

sapply(as.data.frame(centered_measure),sd)
```

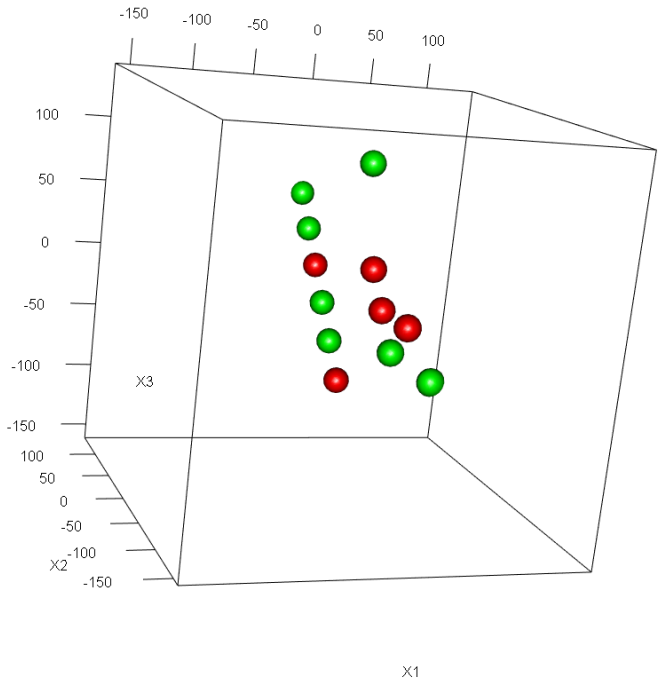
We need to centralize the data. What it means is that we have to bring centers of all of the column to the origin. By bringing the center of all the columns or of the data in each axis to the origin will explain a better picture of how much the data is spreading out with respect to each dimension.

Note: Normalization is a different technique where we convert data to lie between 0 and 1. In scaling we are bring their mean value to zero.

```
> summary(centered_measure)
       X1                  X2                 X3                X4                X5
 Min.   :-0.14417   Min.   :-157.208   Min.   :-73.5   Min.   :-66.354   Min.   :-1.75
 1st Qu.:-0.08167   1st Qu.: -65.333   1st Qu.:-24.5   1st Qu.:-35.729   1st Qu.:-1.75
 Median :-0.01417   Median :   2.042   Median :  0.0   Median :  7.146   Median : 0.00
 Mean   : 0.00000   Mean   :   0.000   Mean   :  0.0   Mean   :  0.000   Mean   : 0.00
 3rd Qu.: 0.06583   3rd Qu.:  69.417   3rd Qu.: 24.5   3rd Qu.: 43.896   3rd Qu.: 1.75
 Max.   : 0.21583   Max.   : 136.792   Max.   : 98.0   Max.   : 43.896   Max.   : 1.75
```

It is clear from the summary that all the columns have their value centered on zero.

Then we again calculate the minimum and maximum limit of the centered data so as to use it for making another 3d plot. The centering of data is done by subtracting mean from each data point. The plot now will look something like this:



Here it is quite clear that all data is present in the middle of the box. Which means that for all three axis the data has become centered, so the data points that were appearing on the left earlier are now in the center of the box.

However, there is still one problem, that even though data is centered, the standard deviation of all the columns still have a lot of difference between. Meaning, one column is spread out less from the center, but another column is spread out more from the origin.

```
> sapply(as.data.frame(centered_measure),sd)
        X1         X2         X3         X4         X5
 0.1057775 88.0861161 43.6264814 45.1659502  1.7511404
> |
```

Here, the compactness (x1) is less spread out from origin than surface area (x2) is spread out from origin.

Step7.

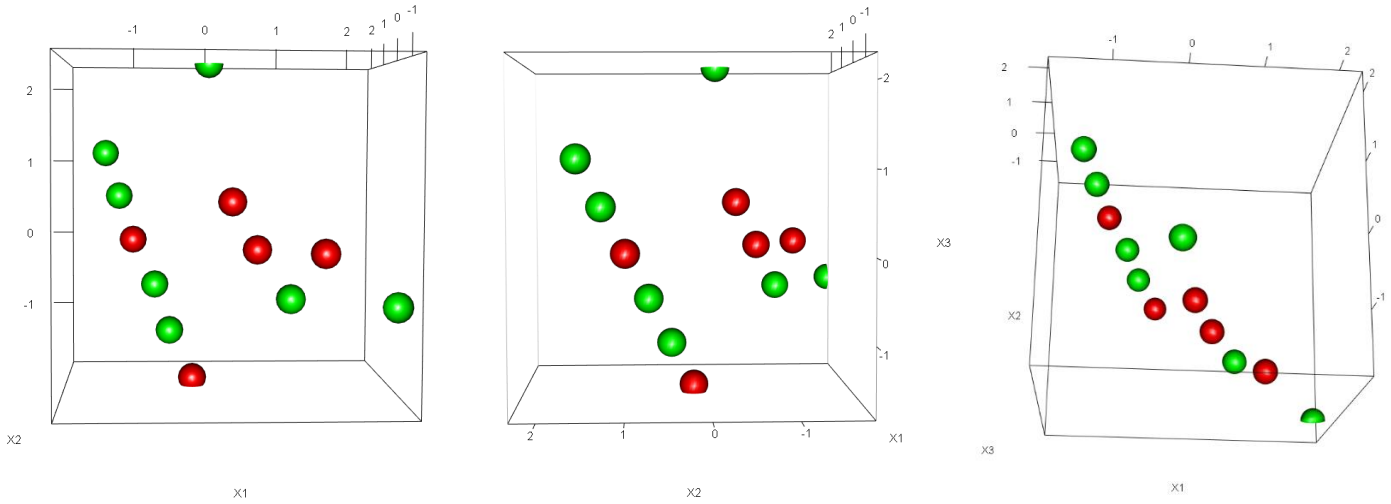
```
#step7
reduced_measure <- scale(measure) # this function scales as well as centeres
summary(reduced_measure)
lims <- c(min(reduced_measure),max(reduced_measure))
plot3d(reduced_measure,type="s",col=color,xlim = lims,ylim = lims,zlim = lims)
# two things has to be done, one is to center the data points so they they spread from zero
# then to reduce the value so that they all belongs between 0 and 1
```

To scale the data as well as center it simultaneously, we can use the scale command with just one argument, which is the original data frame. The other two argument “scale” and “center” are true by default.

```
> summary(reduced_measure)
       X1                 X2                 X3                 X4                 X5
 Min.   :-1.3629   Min.   :-1.78471   Min.   :-1.6848   Min.   :-1.4691   Min.   :-0.
 1st Qu.:-0.7721   1st Qu.:-0.74170   1st Qu.:-0.5616   1st Qu.:-0.7911   1st Qu.:-0.
 Median :-0.1339   Median : 0.02318   Median : 0.0000   Median : 0.1582   Median : 0.
 Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.
 3rd Qu.: 0.6224   3rd Qu.: 0.78805   3rd Qu.: 0.5616   3rd Qu.: 0.9719   3rd Qu.: 0.
 Max.   : 2.0404   Max.   : 1.55293   Max.   : 2.2463   Max.   : 0.9719   Max.   : 0.
> |
```

By looking at the summary, we can observe that all of the dimensions, or columns are centered on zero, whereas their minimum value and maximum value is also close to each other, meaning their standard deviation has become same now.

After taking the limit of minimum value and maximum value, after centering all the data points on origin and after making their standard deviation same, this is how it looks on 3d plain now:



Conclusion

From these pictures above, I conclude that most of the variation is explained by X1 and x3, which is compactness and wall area respectively. So, even if we use only these two columns for any kind of prediction, we can still attain a very good accuracy.

References

A. B, Dufour. “Principal Component Analysis”. *Amazonaws.com.* https://s3.us-east-1.amazonaws.com/blackboard.learn.xythos.prod/5a3148150d016/14706416?response-content-disposition=inline%3B%20filename%2A%3DUTF-8%27%27Week3_PCA_Assignment.pdf&response-content-type=application%2Fpdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20190313T191843Z&X-Amz-SignedHeaders=host&X-Amz-Expires=21600&X-Amz-Credential=AKIAIL7WQYDOOHAZJGWQ%2F20190313%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Signature=f1ae25292a8ccfe7a9a75f1751ca463a4caf90c83ca7f01ca160d88cd036ad5a.

L, Francisco. “Principal Component Analysis in R”. *R-BOLGGER.* https://www.r-bloggers.com/principal-component-analysis-in-r/.