PRINCIPAL COMPONENTS ANALYSIS MINI-PROJECT
December 2018

OVERVIEW

1. Project Background and Description

The goal is to apply what we have learned about Principal Components Analysis (PCA) in a real life dataset, and to interpret the results.

Team group: HAMZA SABER – MEHDI BACHIR

2. Objective

With a large number of variables, the dispersion matrix may be too large to study and interpret properly. There would be too many pairwise correlations between the variables to consider. Graphical displays may also not be particularly helpful when the data set is very large.

To interpret the data in a more meaningful form, it is necessary to reduce the number of variables to a few, interpretable linear combinations of the data. Each linear combination will correspond to a principal component.

3. Computation

- **R packages**:

For this project, we will use R language with the R Studio software. Several functions from different packages are available for computing PCA, the packages that we will use are:

- *FactoMineR* package
- *factoextra* package

- **Used Dataset:** Mobile App Statistics (Google Play App Store):

The ever-changing mobile landscape is a challenging space to navigate. . The percentage of mobile over desktop is only increasing. Android holds about 53.2% of the smartphone market, while iOS is 43%. To get more people to download your app, you need to make sure they can easily find your app. Mobile app analytics is a great way to understand the existing strategy to drive growth and retention of future user.

4. Start

We start by subsetting active individuals and active variables for the principal component analysis:

```
#Read the dataset

data <- read.csv(file.choose())

#Select the active data

myvars <- c("size_bytes", "price",
"rating_count_tot", "user_rating", "sup_devices.num",
"lang.num")

data.active <- data[myvars]
```

- **Data standardization:**

In principal component analysis, variables are often scaled (i.e. standardized). This is particularly recommended when variables are measured in different scales (In our dataset we have: Currency, Bytes …); otherwise, the PCA outputs obtained will be severely affected.

The goal is to make the variables comparable. Generally, variables are scaled to have

1) Standard deviation

2) Mean zero.

By default, the function **PCA** () that we will use after [in *FactoMineR*], standardizes the data automatically during the PCA; so we do not need to do this transformation before the PCA.

- **R implementation:**

Now we perform the function PCA() on our active data.

```
> library("FactoMineR")

> result.pca<-PCA (data.active ,graph=FALSE)
```

The output of the function *PCA*() is a list, including the following components :

```
> print(result.pca)
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 23 individuals, described by 10
variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

- **Visualization / Interpretation:**

We'll use the *factoextra* R package to help in the interpretation of PCA.

Step 1: Examine the eigenvalues

```
> get_eigenvalue(result.pca)
```

Table1: Eigenvalues and the proportion of variation explained by the principal components.

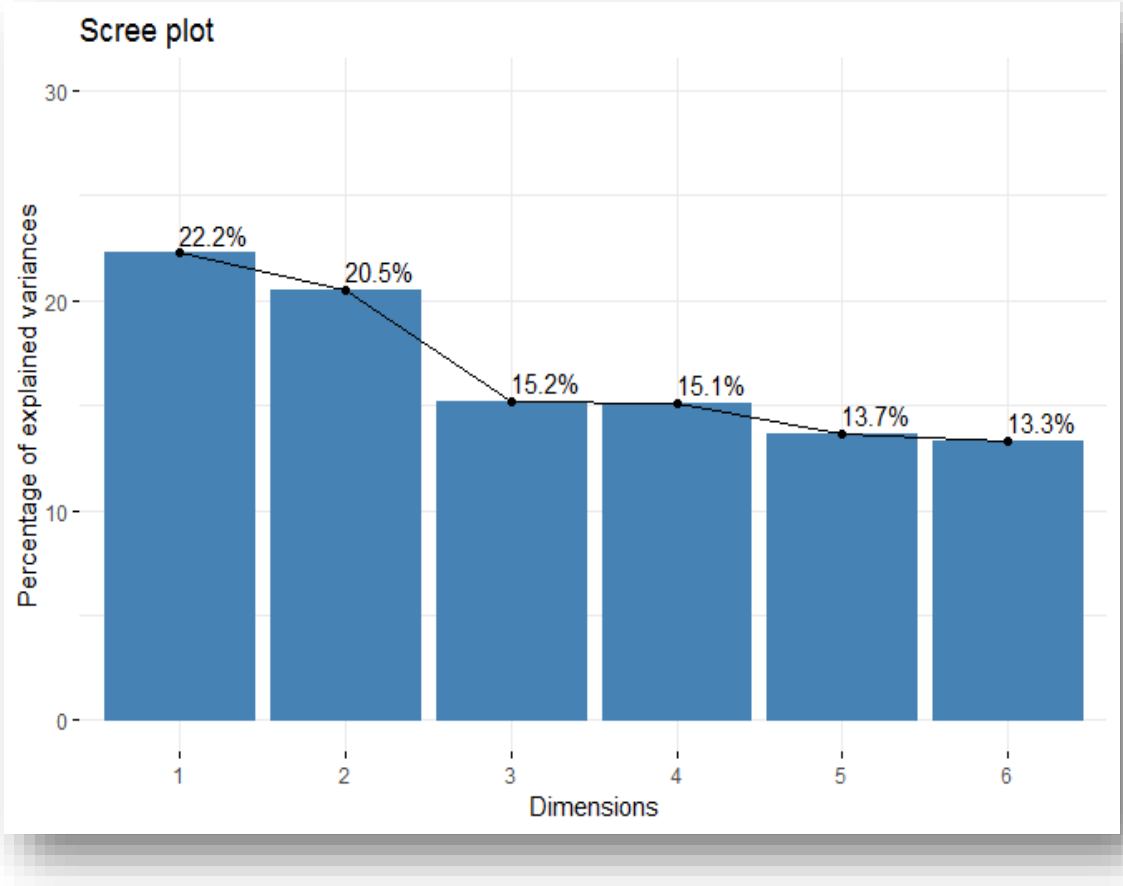
Component	Eigenvalue	Proportion	Cumulative
Dimension 1	1.3345395	22.24232	22.24232
Dimension 2	1.2307102	20.51184	42.75416
Dimension 3	0.9096699	15.16116	57.91533
Dimension 4	0.9063703	15.10617	73.02150
Dimension 5	0.8202269	13.67045	86.69195
Dimension 6	0.7984832	13.30805	100.00000

The proportion of variation explained by each eigenvalue is given in the second column. The cumulative percentage explained is obtained by adding the successive proportions of variation explained to obtain the running total. Therefore, about **57.91533%** of the variation is explained by the first three eigenvalues together.

An alternative method to determine the number of principal components is to look at a Scree Plot.

The scree plot can be produced using the function **fviz_eig():**

```
> fviz_eig(result.pca, addlabels=TRUE, ylim= c(0,50))
```



Step 2: To interpret each component, we must compute the correlations between the original data and each principal component.

- ***Results***

A simple method to extract the results, for variables, from a PCA output is to use the function **`get_pca_var()`** [*factoextra*package].

```
> var <- get_pca_var(result.pca)
```

```
> head(var$cor)
```

```
## Principal Component Analysis Results for variables
##  ===================================================
##    Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

In order ro get the Correlations between variables and dimensions, we write the code below:
And the output is:

	Dimension 1	Dimension 2	Dimension 3	Dimension 4	Dimension 5
Size_Bytes	0.5661722	-0.3722970	0.37824692	-0.12629792	-0.4208004165
Price	0.5140703	-0.4641761	0.27498378	0.07702983	0.6063839222
Rating_count_tot	0.2390704	0.5662144	0.30142511	-0.67887875	-0.0007907371
User_rating	0.5220842	0.3838735	0.05031712	0.55714188	-0.3375214728
lang.num	0.4237931	0.5797621	-0.21028113	0.13062215	0.3831750625

Interpretation of the principal components is based on finding which variables are most strongly correlated with each component.

Here a correlation above **0.5** is considered important.

First Principal Component Analysis - PCA1:

This component can be viewed as a measure of the quality of the app.

We can explain this as a "**PREMIUM APP**". It can be much expensive than the "Standard" version because of additional features, which also explains the increased size of app.

In terms of mobile specifications, some "**PREMIUM APPs**" demand high specifications of hardware (larger RAM, fast Processor…), or the newest software in the market (Android OS Version…) to properly operate.

Second Principal Component Analysis – PCA2:

The second principal component increases with two of the values, increasing Total count of rating and increasing number of languages.

What we can conclude from this is the more languages are in an app, the more of ratings we can get.

Third Principal Component Analysis – PCA3:

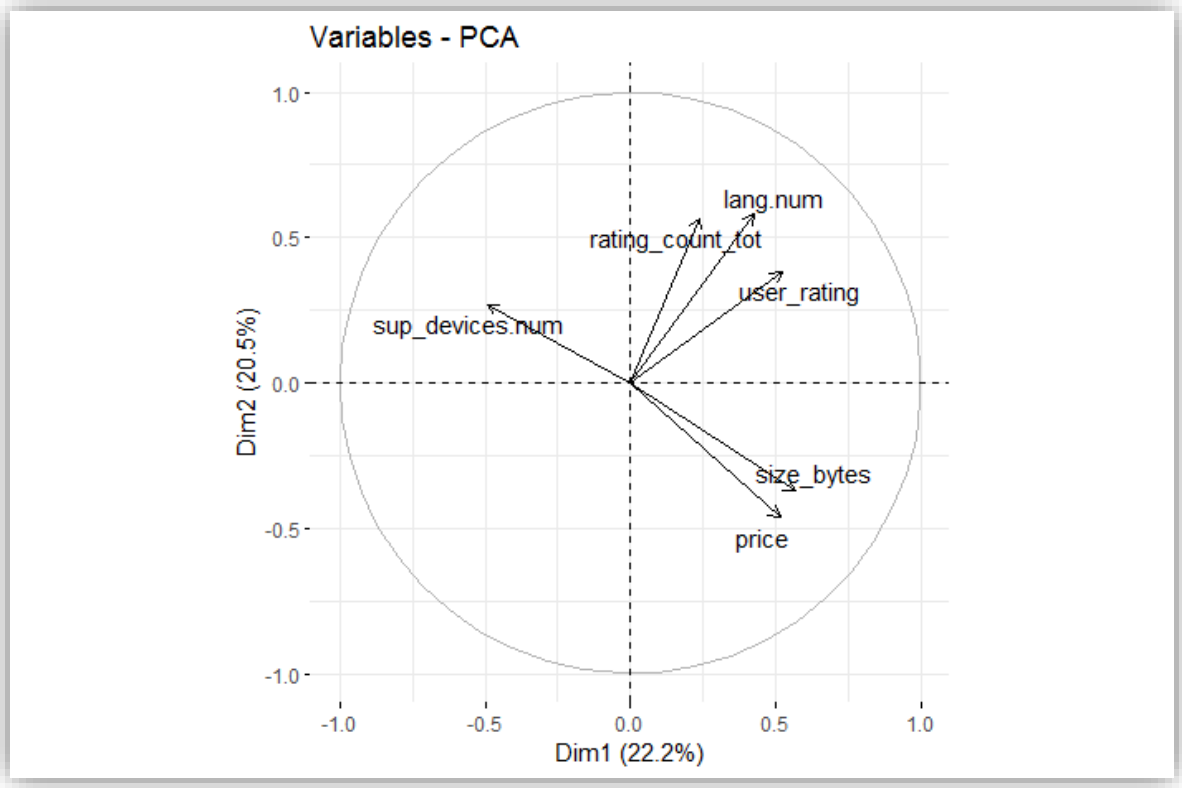
The third principal component increases with only one of the values.

This component can be viewed as a measure of how many devices are supported.

Step 3: Correlation Circle

To plot variables, type this:

```
> fviz_pca_var (res.pca, col.var="black", repel=TRUE)
```



It is possible to color variables by their cos2 values using the argument col.var = "cos2".

- variables with low cos2 values will be colored in “Red”
- variables with mid cos2 values will be colored in “blue”
- variables with high cos2 values will be colored in “Green”

```
>fviz_pca_var(result.pca, col.var="cos2",gradient.cols=c("red",
"blue","green"),repel=TRUE)
```