Rmarkdown - PCA example

R - course

25 november, 2025

```
# Every single chunk can have different setup defined in curly brackets
# For example: this chuck setup is to evaluate the code (eval = T) without including the same in final
# In order to change general setup of the printed chunks we'll use 'knitr' package
# NOTE - this rmarkdown will be converted to pdf document and saved to your working directory by using
library(knitr)

opts_chunk$set(tidy.opts = list(width.cutoff = 65), tidy = TRUE)
```

Dependencies Pipeline depends on the following packages: ‘ggbiplot’, ‘factoextra’, ‘kableExtra’, ‘broom’ and ‘ggrepel’. NOTE - in order to run Rmarkdown each package needs to be already installed in you R for pipeline to work. If you need to install required packages you can always use install.pkg() function from ‘ANOVA Lecture R course.Rmd’ script or by using base function install.packages(). Each time you knit new clean session is created thus, all packages for the pipeline are supposed to be specified in the Rmarkdown otherwise knitting process will break.

```
# load packages

library(pacman)
p_load("ggbiplot", "factoextra", "kableExtra", "broom", "ggrepel")
```

1. Data exploration

This document includes pipeline for data exploration with principal component analysis or simply PCA. The basic principle of PCA is to reduce dimensionality of the input data with many variables while preserving max variation. By doing so original variables are transformed to new set of variables called principal components. It is worth to remember that number of resulting PCs is always less or equal to the number of original variables. The first PC retains the max variation from input data. In this example we’ll use data MTCAR which is pre-load in the base R.

- **MTCARS DATA SET** *provides information extracted from the Motor Trend US magazine (1974), and comprises of fuel consumption and 10 different aspects of automobile design and performance for 32 automobiles (1973–74 models).*
- **NOTE** that you can access all pre-loaded data set in R by simply calling **data()** in the command line.

```
# Assign data to an object and show the table
df <- mtcars
kable(df[1:5, ], caption = 'MTCARS data set', format = 'latex', booktabs = T) %>%
            kable_styling(latex_options = c('striped',"HOLD_position"))
```

Table 1: MTCARS data set

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2

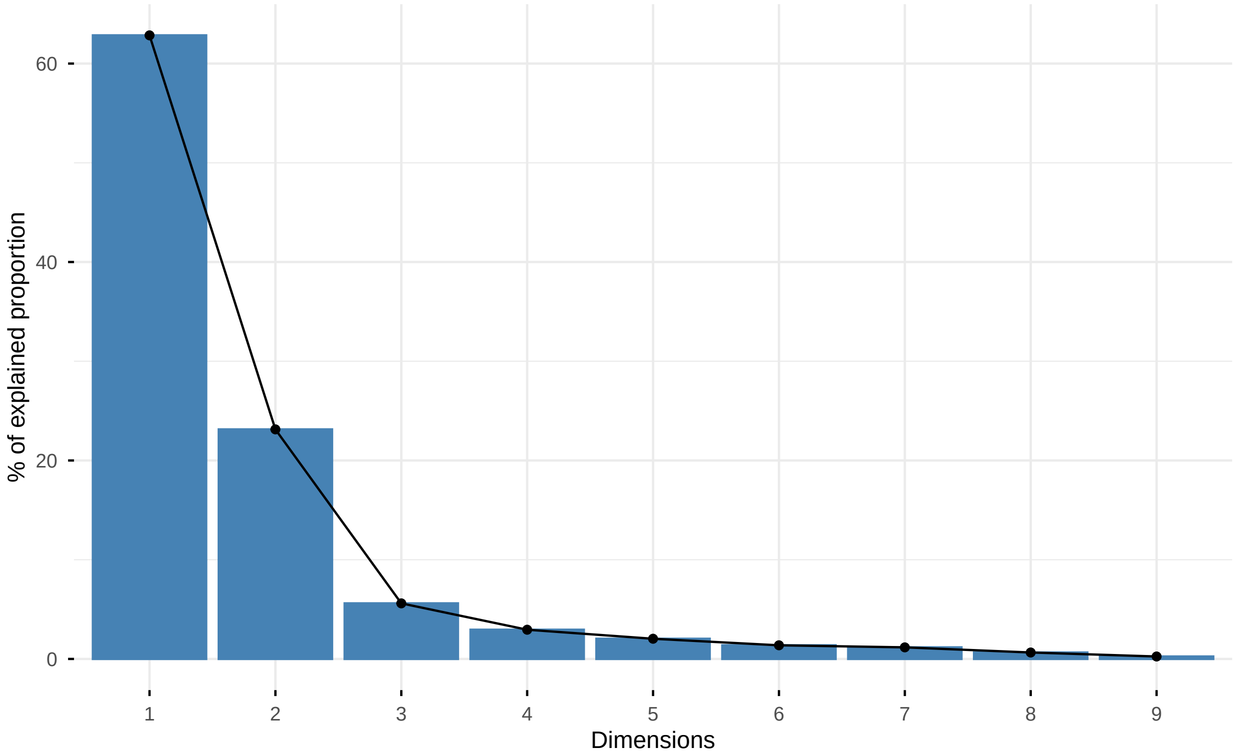
Table 1. Description: mpg -miles per gallon, **cyl** -number of cylinders, **disp** -displacement, **hp** -horse power, **draft** - rear axle ratio **wt** - weight, **qsec** - quarter mile time, **vs** - tipe of the engine block, **am** - transmission, **gear** - numer of gears and **carb** - carburetors.

2. PCA analysis

```
## PCA work the best with numerical data
## We'll exclude categorical variables
df.pca <- prcomp(df[,c(1:7,10,11)], center = T, scale. = T)

## Plot PCA summary
fviz_eig(df.pca, main = '', ylab = '% of explained proportion')
```

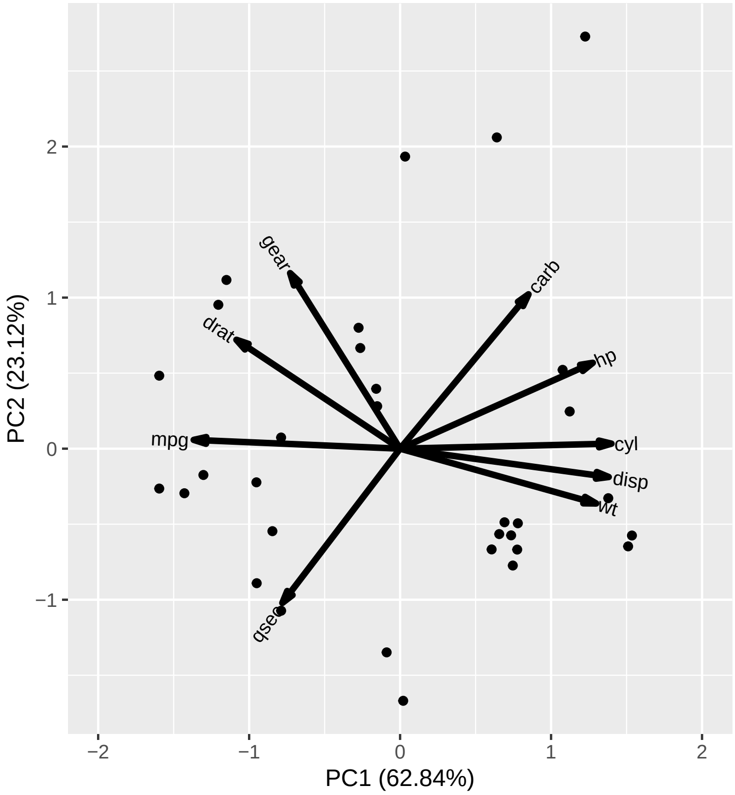
```
## Warning in geom_bar(stat = "identity", fill = barfill, color = barcolor, :
## Ignoring empty aesthetic: ‘width‘.
```



PCA plot just shows the proportion of variance explained by each principal component. Created PCA object (df.pca) contains several information that you can access using $ sign (exp: df.pca$center): center - mean, scale - sd, rotation - correlation between initial variable and PC, x - values of each sample in terms of PC.

3a. PCA plot

```
# ggbiplot - how variables relate to each other?
ggbiplot(df.pca) + xlim(c(-2,2)) + ylab ('PC2 (23.12%)') + xlab('PC1 (62.84%)')
```

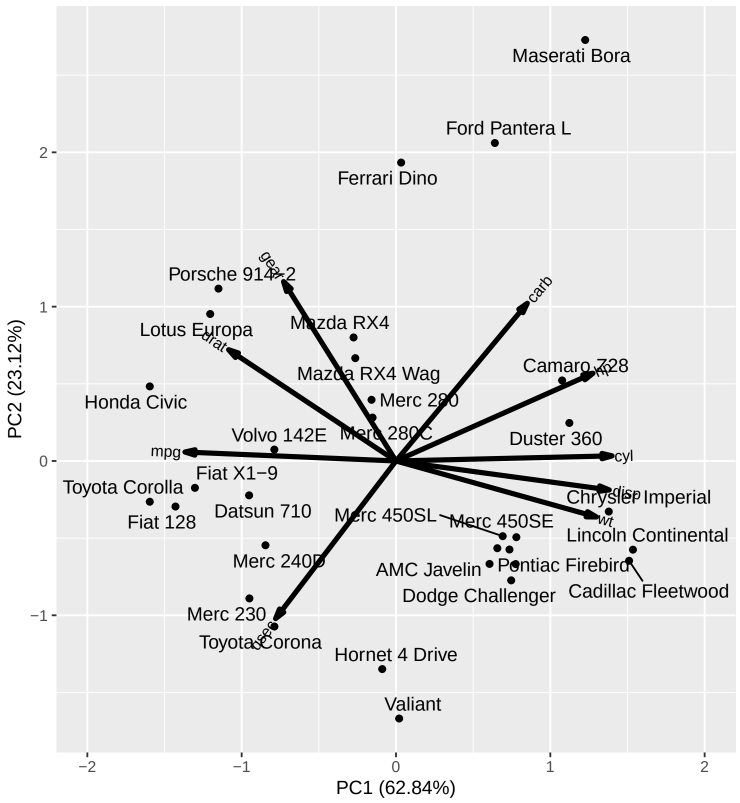


More closely variables appear to each other more correlated they are. Furthermore, variables carb, hp, cyl, disp and wt are positively correlated with PC1 while the rest of the variables present negative correlation towards PC1. To make this plot more informative we should relate each point to corresponding car.

3b. Plot pca

```r
# ggbiplot - how cars relate to each other?
ggbiplot(df.pca) + xlim(c(-2,2)) +  ylab ('PC2 (23.12%)') +
                   xlab('PC1 (62.84%)') + geom_text_repel(label = rownames(df))
```

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

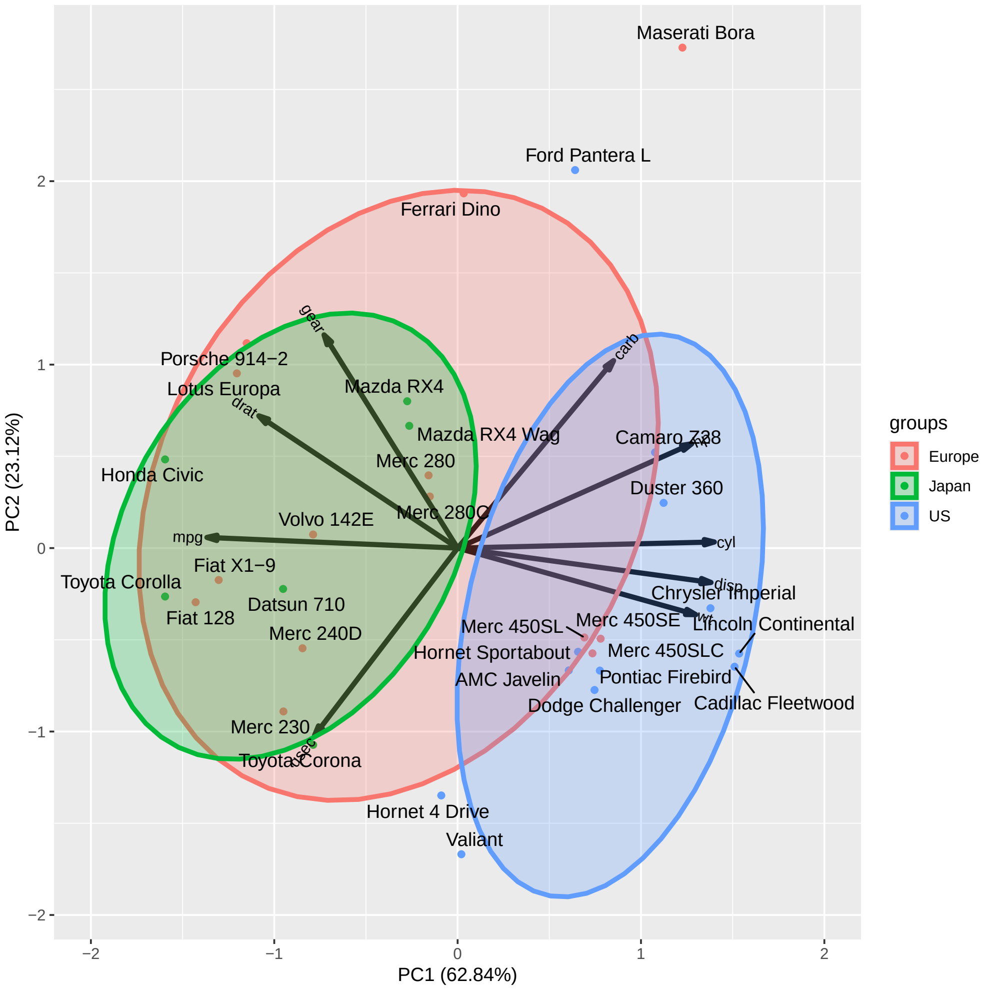


Here we can see which car is more similar to each other. For example in top right corner we see cluster of 3 cars: Maserati Bora, Ford Pantera and Ferrari Dino which makes sense as they are all sports cars.

3c. Plot pca

```
# WE'll manualy assign country of origin
df$country <- c(rep("Japan", 3), rep("US",4), rep("Europe", 7),rep("US",3), "Europe", rep("Japan", 3), r

# ggbiplot - group cars based on the country of origin
ggbiplot(df.pca, ellipse = TRUE,  ellipse.prob = 0.68,
                  groups = df$country) + xlim(c(-2,2)) + ylab ('PC2 (23.12%)') +
                  xlab('PC1 (62.84%)') +
                  geom_text_repel(label = rownames(df), max.overlaps = Inf)
```



From here we can see that US and Japanese cars form 2 distinct clusters, while European cars are less tightly cluster however, more similar to Japanese cars. Japanese and European cars are obviously more fuel efficient, while US cars have more horse power, higher displacement and more cylinders.