Week 8 Training Assignment

K0021N, Luleå University of Technology

Rickard Garvare

2025-05-07

Contents

Introduction

This week's theme is about what is sometimes referred to as *unsupervised learning*. We will focus the training on two methods within this area:

- Principal Component Analysis (PCA)
- K-Means Clustering.

Please read Chapter 7 of the course book before you start with this assignment!

Accolades to Jakob Andersson Sjöberg, Olivia Norberg, Olof Norberg, Kevin Mejia, and other previous students of this course for creating parts of the code and explanations that are used in this assignment.

In an attempt to make the document easier to distinguish text from code, we use the following formatting with light grey background and a blue side border for code chunks:

```
R.version # Example of executable code
```

Recommended work process

We strongly recommend that you find a student colleague and work in pairs with the training assignments. However, make sure that all perform the tasks on their own computer so that you get the hang of it! We also recommend that you save your work using a script or quarto document so that you easily can revisit your code later.

These assignments are designed to enhance your understanding of the course material. Throughout the course, you have the opportunity to delve into the assignments, ask questions, and participate in discussions during problem-solving sessions. These are integral to reinforcing your learning. A goal of these assignments is to immerse you in the weekly theme and familiarize you with tools that are crucial to solving the examination assignments.

Remember the intended work process for the week

1. Participate in the introductory lecture in the beginning of the week.
2. Read the literature and watch recommended videos.
3. Engage with the training assignments.
4. Work with the examination assignments.
5. At week's end, join the problem-solving session to discuss any challenges or questions you may have.

Training Assignment Week 8

Background

AID International is a fictive humanitarian NGO committed to fighting poverty and providing people of developing countries with basic amenities during times of disasters and natural calamities. AID International has been able to raise USD 300 million. Now the CEO of the NGO has to decide how to use this money strategically and effectively. Hence, your job as a data scientist is to categorize the countries and then suggest the countries which AID International should focus on the most.

Dataset

In this training assignment you will work with the file ***country_data.csv***, which contains 167 rows and 10 columns of information on different countries in the world. The objective is to analyze these countries using socio-economic and health factors that may reflect their overall development, see also https://www.kaggle.com/datasets/manusmitajha/countrydatacsv/data

Task 1: General preparations

Install packages if necessary. Call libraries. Set various default formats.

Disclaimer: There are most probably other packages and commands that can do the tasks that we will try below. The packages suggested in this course were chosen by us teachers. However, this does not mean that our way is the only way, or the best way, of doing things. Please feel free to experiment and improve!

1.1 Install packages and call libraries

```
pkgs <- c("readr", "tidyverse", "dlookr", "flextable", "tinytex",
  "complexlm", "corrplot", "robust", "hexbin", "GGally", "dplyr",
  "plotly", "factoextra", "ggbiplot", "devtools")
for (i in pkgs){
  if(! i %in% installed.packages()){
  install.packages(i,repos = "http://cran.us.r-project.org",
                   dependencies = TRUE)}
require(i)}

# Call some libraries
library(readr)
library(tidyverse)
library(dlookr)
library(flextable)
library(tinytex)

# We may also specify a general format for our flextables
set_flextable_defaults(
  font.size = 8, theme_fun = theme_vanilla,
  padding = 6, background.color = "#FFFFFF")
```

1.1 Import and inspect dataset

You could use the "Import Dataset" option in RStudio to get parts of the code below.

```
# Import dataset from our current Working Directory
country_data <- read_csv("country_data.csv")

# Check that it is really a data frame
country_df <- as.data.frame(country_data)

# Examine dataset for dimensions, datatypes etc.
glimpse(country_df)
summary(country_df)

# Are there any missing values?
diagnose(country_df) %>%
  flextable()
```

Our data looks good and we have no missing values. However, the first variable contains country names, i.e. a non numeric class! For the analyses that we intend to do we need to work only on numeric data. To still keep track of which country each row belongs to we can use the country name as row name. Let's create a new data frame, where the character variable 'country' has been deleted and its content has been put into the row names of our new data frame:

```
# Create a new data frame "country_num" with only numerical variables
country_num <- country_df[, -1]

# Replace default row names in "country_num" with the country names
# that are present in the "country" variable of our original dataframe
rownames(country_num) <- country_df$country

# Inspect the results
diagnose(country_num)
# row.names(country_num) # An optional check
```

Ok! Now we may head on with the Principal Component Analysis (PCA).

Task 2: Principal Component Analysis

2.1 Scale and center variables

Since both PCA and K-Means Clustering are sensitive to scale it is often a good idea to normalize examined variables, i.e. subtract each variable with its mean value and then divide it with its standard deviation. By this transformation we ensure that variables with larger ranges will not unwantedly dominate the analysis.

```
# Normalize data
countrynum_scaled <- scale(country_num, center = TRUE, scale = TRUE)
countrynum_scaled <- as.data.frame(countrynum_scaled)

# View the normalized result
summary(countrynum_scaled)
```

As a result the variable means of 'countrynum_scaled' are zero and our analysis will not be sensitive to differences in scale between variables.

2.2 Calculate principal components

The R functions 'princomp()' and 'prcomp()' work in slightly different ways but can both be used to calculate principal components:

```
# Do PCA on our scaled all numeric data frame using the
# generic R functions princomp and prcomp. Put the results
# in two new lists named country_pca1 and country_pca2:
country_pca1 <- princomp(countrynum_scaled)
country_pca2 <- prcomp(countrynum_scaled)
```

2.3 Inspect the results of our PCA

The total number of Principal Components (PC) will be equal to the total number of variables included in the analysis. We have nine (9) variables in our 'countrynum_scaled' data frame and therefore get the same number of PCs from the principle component analysis. Let's check out these PCs:

```
summary(country_pca1)
summary(country_pca2)
```

As might be seen in the summaries the cumulative proportion of variation explained increases with increasing number of principal components.

In this training assignment we will from now on mostly use 'country_pca1', i.e. the results of applying the 'princomp()' function to our data.

2.4 Choose a model

How many principal components should we keep in our model? Of course we want the model to be as simple as possible, but also to represent our data accurately enough. One way to determine how many components to retain is to use a rule of thumb: keep all principal components that show a variance or

eigenvalue of at least 1. To go this way just check the results obtained in Section 2.3 above. Another way of determining an appropriate number of PCs to retain is to make a so called Scree plot:

```
# Scree plot:
graph <- screeplot(country_pca1,type ="lines", main='') +
  abline(h=1, col = "red")

# We may also, for instance, use the 'factoextra' library
# to produce another variant of Scree plot:
library(factoextra)
fviz_eig(country_pca1, addlabels = TRUE, main ='') +
  theme(axis.title=element_text(face="bold"))
```

Scree plots visualize the relative importance of principal components, either via percentage of total variance explained, or via eigenvalues. There are no absolute rules of how much variance that needs to be explained by a PCA model. By inspecting the scree plot one might determine if there is a notable knee/change/break, and chose accordingly. Another thumb rule is to choose PCs with eigenvalues greater than 1. The scree plot thus helps determining which components to continue working with.

Based on our scree plots it could be argued that we should keep five (5) principal components in our model. According to the rule of thumb discussed above we should only keep three (3) principal components. Let's compromise! Principal component number 4 (PC4) is very close to the cutoff limit. To go forward in this training assignment we choose to include four (4) PCs in our model. Now we can take a closer look on the variable loadings:

```
# Show variable loadings (eigenvectors) of the first four
# principal components:
country_pca1$loadings[, 1:4]
```

PCA loading plots show how the variables/features are weighted in selected principal components. Loading plots may also indicate correlations, possible dependencies, and possible cause-and-effect relationships between variables that might be confirmed via experiments. Often, loading plots are only presented for a smaller selection of top-ranked principal components.

A positive loading indicates a positive correlation between the variable and the principal component. Negative loadings indicate negative correlations. We can also visualize by plotting the loadings for each PC:

```
# View variable loadings of our four first principal components
loadings <- country_pca1$loadings[,1:4]
loadings <- as.data.frame(loadings)
loadings$Symbol <- row.names(loadings)
loadings <- gather(loadings, 'Component', 'Weight', -Symbol)
loadings$Color = loadings$Weight > 0
graph <- ggplot(loadings, aes(x=Symbol, y=Weight, fill=Color)) +
  geom_bar(stat='identity', position='identity', width=.75) +
  facet_grid(Component ~ ., scales='free_y') +
  guides(fill='none') +
```

```
  ylab('Variable Loadings on the first four PCs') +
  theme_bw() +
  theme(axis.title.x=element_blank(),
  axis.text.x=element_text(angle=90, vjust=0.5))
graph
```

The figure produced by the R code above might help us understand what fundamental features that the principal components represent, and which variables that are the most influential for each principal component.

One interpretation could be that PC1 mainly differentiates those countries that have high child mortality and high fertility rates from those countries that have high life expectancy, high income, and high gdpp. It seems as PC2 mainly differentiates between countries that have high trade (high exports and imports) from countries that have lesser trade. PC3 differentiates the countries with high health and low inflation from the others.

As we all know by now Jens A. is a big fan of score plots. Let's make two of those! In the first plot PC1 and PC2 provide diagram axes, and PC3 gives us blue dots of varying levels of saturation. In the second plot PC1 and PC3 provide diagram axes, and PC4 gives us differently saturated blue dots:

```
# Score plot
ggplot(data=country_pca1$scores, aes(x=Comp.1, y=Comp.2)) +
  geom_point(aes(colour = Comp.3))

ggplot(data=country_pca1$scores, aes(x=Comp.1, y=Comp.3)) +
  geom_point(aes(colour = Comp.4))
```

PCA score plots show how the observations/records/samples/cases are distributed according to selected principal components. Score plots might reveal patterns between observations, and therefore also opportunities for data reduction via clustering of observations. Score plots are often only presented for a smaller selection of top-ranked principal components.

Clearly, based on our score plots there seems to be some differences between the countries included in our analysis. Perhaps we could find clusters of countries with similar characteristics?

Task 3: Cluster Analysis using K-Means

3.1 Find an appropriate number of clusters

Let's make a loop that plots the cumulative variance explained by a range of possible numbers of clusters. From this we might perhaps identify an "elbow" somewhere in the plot:

```
pct_var <- data.frame(pct_var = 0,
  num_clusters=2:15)

totalss <- kmeans(countrynum_scaled, centers=13, nstart=50, iter.max=5000)$totss

for (i in 2:15) {
  kmCluster <- kmeans(countrynum_scaled, centers=i, nstart=50, iter.max = 5000)
  pct_var[i-1, 'pct_var'] <- kmCluster$betweenss / totalss
  }

graph <- ggplot(pct_var, aes(x=num_clusters, y=pct_var)) +
  geom_line() +
  geom_point() +
  labs(y='% Variance Explained', x='Number of Clusters') +
  scale_x_continuous(breaks=seq(2, 14, by=2)) +
  theme()
graph
```

There seems to be no apparent "knee" where an increase in the number of clusters suddenly stops increasing the variance explained. It could be argued that there is no single standard method to find the "best" number of clusters. Some things that we should consider are:

- Is there knowledge beforehand? If so, this could be used to select the number of clusters (K).
- Plot cumulative percent of variance explained for the default data as we have done above; is there a "knee" in this graph?
- Plot reduction of variance divided by number of clusters; is there a "knee" in this graph?
- Clusters should make sense conceptually!
- Clusters should be as homogenous as possible!
- Having clusters of similar size is often preferred.
- Active experimentation may be done to find clusters of different characteristics.
- Variables may need to be standardized to neutralize effects of having different measurement units. Standardization helps to ensure that some variables do not overly influence a model simply due to the scale of their original measurements.
- We might use PCA to visualize clusters plotted on PCs instead of being plotted on original values. More on this later in this training assignment!
- The K-Means algorithm might be rerun multiple times to increase its reliability.

- Selecting too many clusters (a high K value) might result in over fitting the model, which should be avoided.
- Selecting too few clusters (a low K value) might result in oversimplification, which should also be avoided.

Ok. Based on all this, let us choose to go forward with three (3) clusters.

3.2 Use the K-Means algorithm to find three clusters

Let's run the K-Means algorithm with K = 3. Before we do this we also set a new seed value, to improve randomization. You could choose your own seed number, or use 321 as I do below (or, like all else do, as a pure reflex action just choose 666…).

```
set.seed(321)
country_km <- kmeans(countrynum_scaled, centers=3, nstart=100)
country_km$size # Show the number of countries in each cluster
country_km$centers # Get a matrix of cluster centres

#country_km # Get an extensive report
```

Add the cluster numbers that have been created into a new column. Let's put this new column in a new and extended data frame:

```
# Add cluster numbers as an extra column in a new extended scaled data frame:
countrynumscaled_ext <- as.data.frame(countrynum_scaled)
countrynumscaled_ext$cluster <- factor(country_km$cluster)
glimpse(countrynumscaled_ext)
```

3.3 Examine the results of K-Means Clustering

```
km_means <- data.frame(cluster = factor(1:3), country_km$centers)

graph <- ggplot(data=countrynumscaled_ext, aes(
  x=life_expec, y=gdpp,
  color=cluster, shape=cluster)) +
  geom_point() +
  scale_shape_manual (values = c(1, 2, 3),
    guide = guide_legend(override.aes=aes(size=1))) +
  geom_point(data=km_means, aes(x=life_expec, y=gdpp), size=2, stroke=2, color='black') +
  theme_bw() +
  scale_x_continuous(expand=c(0.1,0)) +
  scale_y_continuous(expand=c(0.1,0))
graph
```

From the figure created by the code above we can see that one of the clusters has both high gdpp and high life expectancy. Another cluster has low gdpp and also low life expectancy. The last cluster is

somewhere in between. We might create bar graphs and tables to display the means of the variables in each cluster:

```
# Making bargraph distances between scaled variable means and cluster centers:
centers <- as.data.frame(t(country_km$centers))
names(centers) <- paste('Cluster', 1:3)
centers$Symbol <- row.names(centers)
centers <- gather(centers, 'Cluster', 'Mean', -Symbol)
centers$Color = centers$Mean > 0

graph <- ggplot(centers, aes(x=Symbol, y=Mean, fill=Color)) +
  geom_bar(stat='identity', position='identity', width=.75) +
  facet_grid(Cluster ~ ., scales='free_y') +
  guides(fill='none') +
  ylab('Distance between each scaled variable mean and cluster center') +
  theme_bw() +
  theme(axis.title.x=element_blank(),
  axis.text.x=element_text(angle=90, vjust=0.5))
graph
```

We can also make a table with non scaled variable means for each cluster:

```
# Make a new non scaled all numeric data frame that includes cluster numbers:
countrynum_ext <- country_num
countrynum_ext$cluster <- factor(country_km$cluster)

# Show a table with non scaled variable means for each cluster:
countrynum_ext %>%
  group_by(cluster) %>%
    summarise_all(mean)
```

Our figures and tables indicate that countries in one of the clusters have negative means for financial factors and life expectancy but positive for child mortality, inflation and fertility. Since I used 'set.seed(321)' this cluster got named "Cluster 2" in my analysis. A possible interpretation is that Cluster 2 represents developing countries.

Let us list the countries of each cluster:

```
# Listing the countries of each cluster
for (i in 1:3) {
  cat("Countries in Cluster number", i, ":\n")
  print(country_df$country[countrynumscaled_ext$cluster == i])
  cat("\n")
}
```

Task 4: Combine PCA and K-Means Clustering

4.1 Make a PCA plot with the clusters included

Let's put together a plot where we combine our results from PCA with the results from clustering:

```
library(ggbiplot)
biplot = ggbiplot(pcobj = country_pca1,
  choices = c(1,2), # Selection of Principal Components
  obs.scale = 1, var.scale = 1, # Scaling of axes
  labels = row.names(countrynumscaled_ext), # Use rownames as scatter labels
  labels.size = 2, # Select size of characters
  varname.size = 4,
  varname.abbrev = FALSE, # Dont abbreviate variable names
  var.axes = TRUE, # Keep variable vectors
  circle = TRUE, # Add unit variance circle
  ellipse = TRUE,
  groups = countrynumscaled_ext$cluster) +
    scale_color_manual(values = c("red", "blue", "green")) +
    theme_minimal() +
  ggtitle("PCA scoreplot with loadings and K-Means clusters") +
  theme(legend.position = "right")
print(biplot)
```

Cluster 2 (negative means for financial factors and life expectancy but positive for child mortality, inflation and fertility) might perhaps be of special interest to us, given the overall purpose of providing a basis for decision making at AID International.

As we know Erik V. is very skeptical towards 3D-plots, at least when the data to be displayed is in only one or two dimensions. Let's make a plot that is not only in some kind of four dimensional format (although using colors for one dimension and projecting it all on your 2D computer screen) but also interactive in the way that you can zoom in and out of the object created, and also rotate and pan! Here is the code for a "3D-scatterplot", where the axes represent our first three principal components, and the colors of the dots represent our three K-Means clusters:

```
# R code mainly generated by Chat GPT:
library(plotly)

# Note that we here use 'country_pca2' generated by the function prcomp:
pca_data <- data.frame(country_pca2$x[, 1:3])  # PC1, PC2, PC3
pca_data$cluster <- factor(country_km$cluster) # K-Means clusters

fig <- plot_ly(pca_data,
  x = ~PC1, y = ~PC2, z = ~PC3,
  color = ~cluster, symbol = ~cluster,
  type = 'scatter3d',
  mode = 'markers')
fig # Test to zoom and rotate the object that is created from this!
```

4.2 Focus on the most interesting cluster

Select the cluster that you find most interesting. Scatter plot the countries within this cluster:

```
countrypca_with_clusters <- as.data.frame(country_pca1$scores[, 1:4])
countrypca_with_clusters$cluster <- factor(country_km$cluster)

cluster2_scores <- countrypca_with_clusters %>%
  filter(cluster == 2) # Select Cluster 2

graph <- ggplot(data=cluster2_scores, aes(
  x=Comp.1, y=Comp.3,
  color=Comp.2)) +
  geom_point(color = "darkblue", size = 1) +
  geom_text(aes(label = rownames(cluster2_scores)),
            vjust = -1, hjust = 0.5, size = 4) +
  scale_x_continuous(expand=c(0.1,0)) +
  scale_y_continuous(expand=c(0.1,0)) +
  labs(title = "Scatterplot of countries within Cluster 2",
       x = "PC1",
       y = "PC3") +
  theme_minimal()
graph
```

Expand on the results provided by the code above!

4.3 Conclusions and Discussion

Prepare a concluding section summarizing your key findings. Based on the results of your analysis, provide a well-supported recommendation to the CEO of AID International regarding which country or countries should be prioritized for future initiatives.

In addition, critically evaluate the methodological choices made during your analysis. Discuss both advantages and potential limitations of using Principal Component Analysis (PCA) and K-Means Clustering in this context. Specifically, address the implications of the linearity assumption inherent in PCA, as well as the sensitivity of the K-Means algorithm to the initial selection of centroids and the underlying shape of the data clusters. Reflect on how these factors may influence the robustness and interpretability of your findings.

Disclaimer

Please note that for all examples provided in these training assignments there are further improvement potential, both in the plots, in the coding, and in the interpretations of results obtained. There are lots of fun tweaks and improvements that can be made using various packages for R, and we encourage you to further explore these on your own. Extra tweaks and improvements could be especially worthwhile when finishing plots for presentations or to improve clarity and communication of information. Please share with us teachers the suggestions that you have for improving all parts of this course!

You made it!

You have now made it to the end of this training assignment! Well done! Remember that you are encouraged to ask questions at the problem-solving session if you ran into trouble with some parts of the assignments. Now you should start working with the examination assignment.

Good luck!