



Prof. Dr. Steffen Wagner
Angewandte Statistik
Fachbereich II
Berliner Hochschule für Technik

Exercise 01: K-Means Clustering

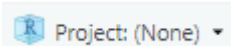
Machine Learning I – SoSe 24

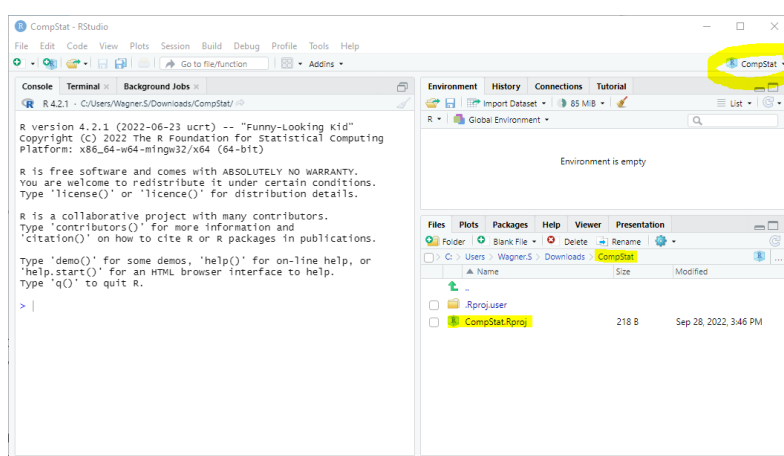
1 Preparations	2
1.1 Recap: Setting up a RStudio Project	2
1.2 Required Packages	3
2 K-Means in R	4
2.1 Simulated Data	4
2.2 Clustering City Locations	6
2.3 Tidying up	7
3 Written Exercises	8
3.1 Clustering by hand	8
3.2 Efficient computing	11
4 Congratulations	12





1 Preparations

1.1 Recap: Setting up a RStudio Project

1. Make sure you work inside a RStudio project.
 - a. Via the menu *File/New Project ...* or click at the right of the RStudio window on  and choose *New Project ...*
 - b. Select *New Directory* and *New Project* as Project Type.
 - c. Choose a meaningful name for the new directory, e.g. ML1 and browse to the directory within you want to create the R project directory. (This could be your document folder.)
 - d. Then press *Create Project*. RStudio should look now more or less like¹

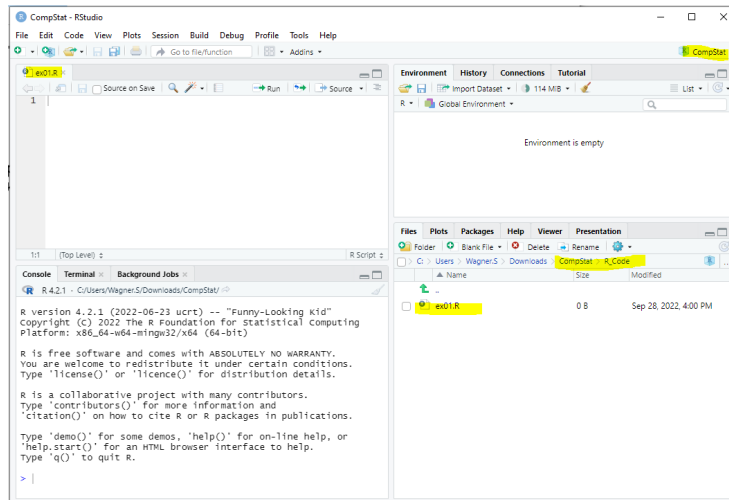


2. Create an R Script file to perform the exercise:

- a. Click on the  and create a new folder named `R_Code`.
- b. Create a new R file either by choosing *File/New File/R Script* from the menu, using the shortcut `Ctrl+Shift+N` or by pressing .
- c. Save the file with the name `ex01.R` in the folder `R_Code`.

3. Check that your RStudio now look (more or less) as follows, especially the yellow marked spots are important:

¹ Screenshot shows different project name than suggested, but you get the point - I assume!



- Remember all the things about R and RStudio you learned in the course Statistical Computing. Please ask the lecturer for help, whenever anything is unclear or if you haven't yet used R up to now.

1.2 Required Packages

For this exercise you require the following additional R packages. Please make sure that you have installed them on your computer before coming to the workshop session since Eduroam is still not working.

```
# check if packages can be loaded, i.e. they are already installed  
library(dplyr)  
library(mdsr)
```

If you get an error at this stage, you need to install the packages using

```
install.packages("dplyr")  
install.packages("mdsr")
```



2 K-Means in R

2.1 Simulated Data

Simulate a matrix with two columns representing the x and y coordinates of 50 points. The first 25 points will be shifted in a South-East direction to create two clusters. Note you can copy and paste this code from the PDF file.

```
# data generation
set.seed(1234567890)
x <- matrix(rnorm(50*2), ncol=2)
x[1:25,1] <- x[1:25,1] + 2
x[1:25,2] <- x[1:25,2] - 2
plot(x, pch = 16, asp = 1) # why `asp=1`? Check help for `plot()` to find out!
```

The option `asp=1` means the *aspect ratio* is 1, i.e. the x - and y -axes are on the same scale. This helps to visually judge the euclidean distances.

Execute the following code **one line at a time**:

```
# run with two clusters:
km.out <- kmeans(x, centers = 2, nstart = 1)
km.out$cluster      # a vector specifying which cluster
                    # each row (=observation) belongs to
names(km.out)       # all the different elements of the kmeans output
km.out$totss        # the sum of squares without clustering
km.out$tot.withinss # the sum of squares with this clustering
km.out$withinss     # the sum of squares within each cluster
km.out$centers      # matrix with the center coordinates

plot(x, col = km.out$cluster, pch = 16, asp = 1) # plot points coloured by cluster
points(km.out$centers, col=1:2, pch=3, cex = 2) # add the cluster centers
```

The `kmeans` command has an argument called `nstart`. This indicates how many times the algorithm is to be repeated using different random starting configurations. `nstart=1` is too small, you might be unlucky and have found a poor local minimum.

Therefore, repeat this using 20 repeats

```
# clustering
km.out <- kmeans(x, centers = 2, nstart = 20)

# visualisation
plot(x, col = km.out$cluster, pch = 16, asp = 1)
points(km.out$centers, col=1:2, pch=3, cex = 2)

# stats
km.out$tot.withinss
```

Ok it seems that the first attempt was a good (or maybe the best) solution.

We can now see what happens when we try to force more than two clusters to the data.



```
set.seed(4)

# 3 clusters
km.out <- kmeans(x, centers = 3, nstart = 20)
plot(x, col = km.out$cluster, pch = 16, asp = 1)
km.out$tot.withinss

# 4 clusters
km.out <- kmeans(x, centers = 4, nstart = 20)
plot(x, col = km.out$cluster, pch = 16, asp = 1)
km.out$tot.withinss
```

The within cluster sum of squares W is now much lower even though there are only really two clusters in our data matrix. W almost always decreases with increasing number of clusters

⇒ The best number of clusters to fit cannot be found by simply by minimising W . We will look at how to choose K next week.

We will now work through a similar example where the data has 3 clusters.

We will now work through a similar example where the data has 3 clusters.

```
# data generation
x <- matrix(rnorm(50*3), ncol = 2)
x[1:25, 1] <- x[1:25, 1] + 2
x[1:25, 2] <- x[1:25, 2] - 2
x[50+1:25, 1] <- x[50 + 1:25, 1] + 2
x[50+1:25, 2] <- x[50 + 1:25, 2] + 2

# clustering
km.out <- kmeans(x, 3, nstart = 20)

# visualisation
plot(x, col = km.out$cluster, pch = 16, asp = 1)
points(km.out$centers, col = 1:3, pch = 3, cex = 2)
```



2.2 Clustering City Locations

We will use the packages `dplyr` and `msdr`:

```
# load packages
library(dplyr) # we will use this for easier data handling
library(msdr)  # contains the data set we want to work with
```

The data set we want to work with is named `world_cities` and can be loaded into the global environment of the R session with the `data` function:

```
data("world_cities")
```

First and always have a look at the data before starting any complex analysis. Find answers to the following questions using the functions `str` and `summary`:

- What variables are contained in the data set?
- How many observations are listed in the data set?
- How many cities have at least 200.000 inhabitants?

```
# we learn about the data:
# - dimension (= rows x columns)
# - names and class of columns
# - example for the first observations
str(world_cities) # we see the dimension

# summary statistics
summary(world_cities)
```

We want to restrict our analysis to cities with at least 200.000 inhabitants and then keep only the two columns `longitude` and `latitude` stored in a new data.frame named `big_cities`. Use the functions of the `dplyr` package to do so.

```
big_cities <- world_cities %>%
  filter(population >= 200000) %>%
  select(longitude, latitude)
```

We will now use the K-Means algorithm using 7 clusters and plot the clusters

```
set.seed(1234567890)

# clustering
cities_km <- kmeans(big_cities, centers = 7)

# visualisation
plot(big_cities, col = cities_km$cluster, pch = 16, asp = 1, cex = 0.3)
points(cities_km$centers, col = 1:7, pch = 3, cex = 2)
```

Please write down what you notice in the scatter plot.

It is important to realise that there is nothing in the data defining the continents, but the clustering method can easily and surprisingly effectively identify them. **Clustering** is a **structure-detecting** method!



Try using different numbers of clusters from 2 upwards. Also notice that the borders of each cluster can differ depending on the seed.

2.3 Tidying up

1. Save your script file.
2. Leave RStudio and **do not** save the workspace!



3 Written Exercises

Solutions to written exercises will not be uploaded in Moodle until a few weeks before the exam. An important skill to learn is to check and have confidence in your answers, without being able to compare it with the correct answer. You can always check with the lecturer if you are unsure about anything.

3.1 Clustering by hand

In this problem, you will perform K-means clustering manually, with $K=2$, on a small example where $n=6$ observations and $p=2$ variables based on the following data set

Observation	X_1	X_2	Initial group
1	8	1	1
2	2	3	2
3	3	3	1
4	6	-1	2
5	3	2	1
6	7	0	2

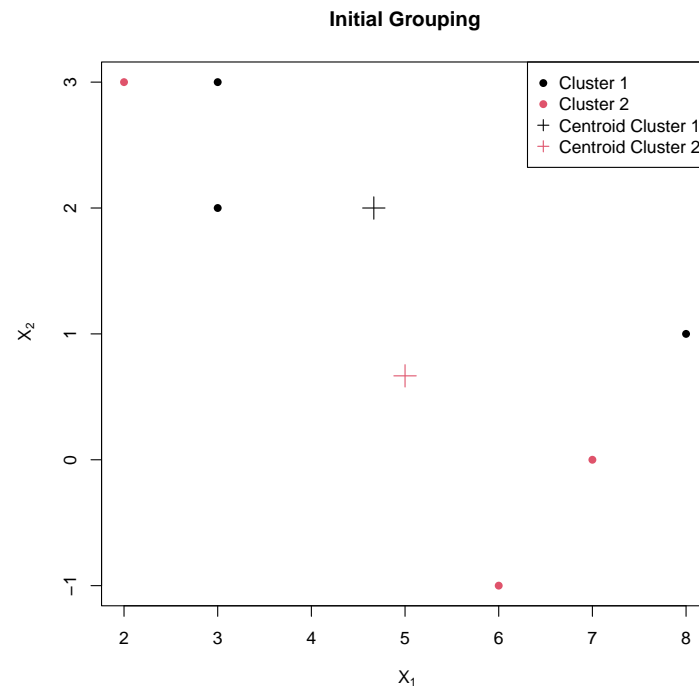
Remarks:

- The initial clustering is given in the 4th column. Sketch the the observations in a drawn scatter plot, indicating the initial clusters with different colours or shapes.
- Compute the centroid for each cluster and depict them in your plot.
- Assign each observation to the centroid to which it is closest, in terms of squared Euclidean distance. Calculate the squared Euclidean distances based on the new cluster assignments..
- Repeat (b) and (c) until the answers obtained stop changing.
- In your plot from (a), colour the observations according to the cluster labels obtained in (d) and also depict the final centroids.



Solution:

a) Graph with initial clustering and initial centroids:



b) The coordinates for the initial centroids are as follows:

Centroid	X_1	X_2
μ_1	4.67	2.00
μ_2	5.00	0.67

c) Distances for each observation regarding the initial centroids are as follows and new grouping:

Observation	distance μ_1	distance μ_2	new cluster
1	3.48	3.02	2
2	2.85	3.80	1
3	1.94	3.07	1
4	3.28	1.94	2
5	1.67	2.40	1
6	3.07	2.11	2

d) Next iteration: update centroids, derive new distances and grouping

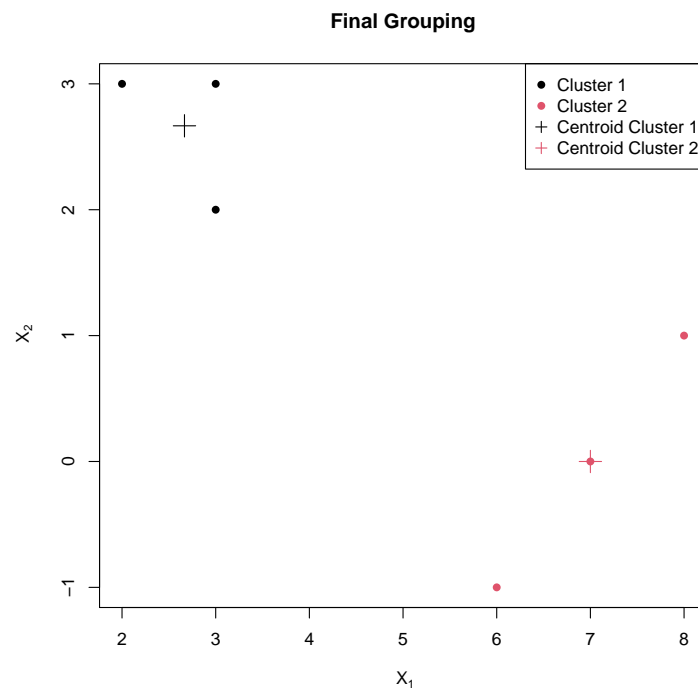
Centroid	X_1	X_2
μ_1	2.67	2.67
μ_2	7.00	0.00



Observation	distance μ_1	distance μ_2	new cluster
1	5.59	1.41	2
2	0.75	5.83	1
3	0.47	5.00	1
4	4.96	1.41	2
5	0.75	4.47	1
6	5.09	0.00	2

As one can see, the cluster results did not change with respect to the former calculations in (c).

e) Visualisation:





3.2 Efficient computing

As discussed in the lecture using the equality

$$\frac{1}{|C_k|} \sum_{i,j \in C_k} \|x_i - x_j\|^2 = 2 \sum_{i \in C_k} \|x_i - \mu_k\|^2$$

means that the squared distances between all pairs of observations have only to be calculated once instead of recalculating the distances to the updated centroids at each iteration step.

It is possible to prove this algebraically, but we will only consider the simplest case: one dimension ($p = 1$) and one cluster. In practice clustering is not done on just one variable, but it gives an insight to the proof for p -dimensional clustering:

$$\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n (x_i - x_j)^2 = 2 \sum_{i=1}^n (x_i - \bar{x})^2.$$

- a) Show that this is the case for the data X_1 given in exercise 3.1, i.e.
 $x_1 = 8, x_2 = 2, x_3 = 3, x_4 = 6, x_5 = 3, x_6 = 7$
- b) Show that equation for the 1-dim case is true for x_1, x_2, \dots, x_n in general.
 Hint: Add and subtract \bar{x} to the left hand side of the equation, giving:

$$\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n ((x_i - \bar{x}) - (x_j - \bar{x}))^2$$

Solution:

- a) Given the data from exercise 3.1 one obtains for the LHS of the 1-dim equation

$$\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n (x_i - x_j)^2 = \frac{370}{6} = 61.67$$

and for the RHS with $\bar{x} = 4.83$ one obtains

$$2 \sum_{i=1}^n (x_i - \bar{x})^2 = 2 \cdot 30.83 = 61.66$$

which is equal ignoring rounding issues.

- b) General case:

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n ((x_i - \bar{x}) - (x_j - \bar{x}))^2 &= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n [(x_i - \bar{x})^2 - 2(x_i - \bar{x})(x_j - \bar{x}) + (x_j - \bar{x})^2] \\ &= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n (x_i - \bar{x})^2 + \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n (x_j - \bar{x})^2 \\ &= \sum_{i=1}^n (x_i - \bar{x})^2 + \sum_{j=1}^n (x_j - \bar{x})^2 \\ &= 2 \sum_{i=1}^n (x_i - \bar{x})^2 \end{aligned}$$

□



4 Congratulations

You have learnt a lot today!