Segmentation: Clustering

Luis Francisco Gómez López

FAEDIS

2024-09-19

UNIVERSIDAD MILITAR NUEVA GRANADA

Table of contents I

- This presentation is based on (Chapman and Feit 2019, chap. 11)

- Find groups of customers that differ in different dimensions to engage in more effective promotion

- **age**: age of the consumer in years
- **gender**: if the consumer is male of female
- **income**: yearly disposable income of the consumer
- **kids**: number of children of the consumer
- **ownHome**: if the consumer owns a home
- **subscribe**: if the consumer is subscribed or not

- **Import data**

```
segmentation <- read_csv(file = "http://goo.gl/qw303p") |>
  select(-Segment) # Remove Segment column to understand how it was build
segmentation |> head(n = 5)
```

```
# A tibble: 5 x 6
    age gender income  kids ownHome subscribe
  <dbl> <chr>   <dbl> <dbl> <chr>   <chr>
1  47.3 Male   49483.     2 ownNo   subNo
2  31.4 Male   35546.     1 ownYes  subNo
3  43.2 Male   44169.     0 ownYes  subNo
4  37.3 Female 81042.     1 ownNo   subNo
5  41.0 Female 79353.     3 ownYes  subNo
```

- **Inspect data**

```r
segmentation |> glimpse()
```

```
Rows: 300
Columns: 6
$ age       <dbl> 47.31613, 31.38684, 43.20034, 37.31700, 40.95439, 43.03387, ~
$ gender    <chr> "Male", "Male", "Male", "Female", "Female", "Male", "Male", ~
$ income    <dbl> 49482.81, 35546.29, 44169.19, 81041.99, 79353.01, 58143.36, ~
$ kids      <dbl> 2, 1, 0, 1, 3, 4, 3, 0, 1, 0, 0, 0, 2, 3, 1, 3, 0, 0, 1, 2, ~
$ ownHome   <chr> "ownNo", "ownYes", "ownYes", "ownNo", "ownYes", "ownYes", "o~
$ subscribe <chr> "subNo", "subNo", "subNo", "subNo", "subNo", "subNo", "subNo~
```

- Transform data

```r
segmentation <- segmentation |>
  mutate(gender = factor(gender, ordered = FALSE),
         kids = as.integer(kids),
         ownHome = factor(ownHome, ordered = FALSE),
         subscribe = factor(subscribe, ordered = FALSE))

segmentation |> head(n = 5)
```

```
# A tibble: 5 x 6
    age gender income  kids ownHome subscribe
  <dbl> <fct>   <dbl> <int> <fct>   <fct>
1  47.3 Male   49483.     2 ownNo   subNo
2  31.4 Male   35546.     1 ownYes  subNo
3  43.2 Male   44169.     0 ownYes  subNo
4  37.3 Female 81042.     1 ownNo   subNo
5  41.0 Female 79353.     3 ownYes  subNo
```

- **Summarize data**
  - Ups the table is really big!!! Try it in your console to see the complete table

```
segmentation |> skim()
```

Segmentation

- Classification (**We will not cover this topic**)
  - Supervised learning
    - Dependent variable is known and the goal is to predict the dependent variable from the independent variables
    - Naive bayes, Random Forest
- Clustering (**This topic will be covered**)
  - Unsupervised learning
    - Dependent variable is unknown and the goal is to discover it from the independent variables
    - Model-based clustering, Latent Class Analysis (**We will not cover these methods**)
    - Hierarchical clustering, k-means (**These methods will be covered**)

- Clustering
    - Grouping a set of observations in such a way that observations in the same group (cluster) are more similar to each other than to those in other groups (clusters).
    - A notion of how **"close"** 2 observations is necessary to group objects where this is formalized using the concept of **distance** (known as metric[1] in mathematics)
        - There are many notions of distance (Deza and Deza 2016) where in this chapter the **Euclidean** and the **Gower** distance will be used

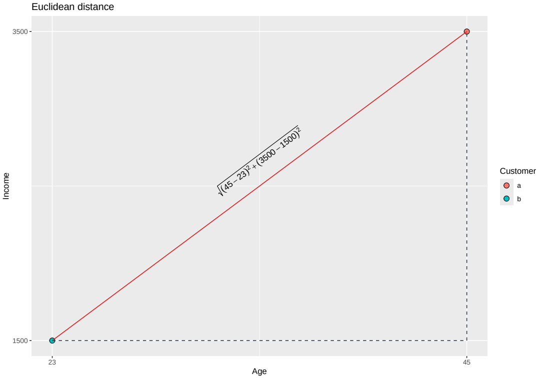
[1] https://en.wikipedia.org/wiki/Metric_space

- **Euclidean distance**: it can only be used for numerical data
  - $x = (x_1, x_2, \dots, x_n)$
  - $y = (y_1, y_2, \dots, y_n)$

$$
\begin{aligned}
d(x,y) &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2} \\
&= \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}
\end{aligned}
$$

- An example:
  - 2 customers characteristic by age and income
    - $a = (45, 3500)$
    - $b = (23, 1500)$

- Manual calculation
  - $d(a,b) = \sqrt{(45-23)^2 + (3500-1500)^2} = 2000.121$



- Using R

```
customers <- tibble(Customer = c("a", "b"),
                    Age = c(45, 23),
                    Income = c(3500, 1500))
customers
```

```
# A tibble: 2 x 3
  Customer   Age Income
  <chr>    <dbl>  <dbl>
1 a           45   3500
2 b           23   1500
```

```
library(cluster)
customers |>
  select(-Customer) |>
  daisy(metric = "euclidean")
```

```
Dissimilarities :
         1
2 2000.121

Metric :  euclidean
Number of objects : 2
```

- **Gower distance**: it can be used for categorical, numerical data and missing values
    - $x = (x_1, x_2, \dots, x_n)$
    - $y = (y_1, y_2, \dots, y_n)$

$$
\begin{aligned}
d(x,y) &= \left[\frac{w_1 \delta^k_{x_1 y_1}}{\sum_{k=1}^n w_k \delta^k_{x_i y_i}}\right] d^1_{x_1 y_1} + \left[\frac{w_2 \delta^k_{x_2 y_2}}{\sum_{k=1}^n w_k \delta^k_{x_i y_i}}\right] d^2_{x_2 y_2} + \dots + \left[\frac{w_n \delta^k_{x_n y_n}}{\sum_{k=1}^n w_k \delta^k_{x_i y_i}}\right] d^n_{x_n y_n} \\
&= \frac{\sum_{k=1}^n w_k \delta^k_{x_i y_i} d^k_{x_i y_i}}{\sum_{k=1}^n w_k \delta^k_{x_i y_i}}
\end{aligned}
$$

Where:

$$w_k \in \mathbb{R} \text{ for } k = 1, 2, \dots, n$$

$$\sum_{k=1}^n w_k \delta^k_{x_i y_i} = w_1 \delta^1_{x_1 y_1} + w_2 \delta^2_{x_2 y_2} + \dots + w_n \delta^n_{x_n y_n}$$

- **Gower distance**: it can be used for categorical, numerical data and missing values
  - $x = (x_1, x_2, \dots, x_n)$
  - $y = (y_1, y_2, \dots, y_n)$

$$d(x, y) = \frac{\sum_{k=1}^{n} w_k \delta_{x_k y_k}^{k} d_{x_k y_k}^{k}}{\sum_{k=1}^{n} w_k \delta_{x_k y_k}^{k}}$$

Where[2]:

$$\delta_{x_k y_k}^{k} = \begin{cases} 0 & \text{if } x_k \text{ or } y_k \text{ is a missing value} \\ 0 & \text{if } x_k, y_k \text{ represent an asymmetric binary variable and } x_k = y_k = 0 \\ 1 & \text{otherwise} \end{cases}$$

[2] See (Kaufman and Rousseeuw 1990, 25–27) for a definition of **asymmetric binary variable**

- **Gower distance**: it can be used for categorical, numerical data and missing values
    - $x = (x_1, x_2, \dots, x_n)$
    - $y = (y_1, y_2, \dots, y_n)$

$$d(x, y) = \frac{\sum_{k=1}^{n} w_k \delta_{x_k y_k}^{k} d_{x_k y_k}^{k}}{\sum_{k=1}^{n} w_k \delta_{x_k y_k}^{k}}$$

Where:

$$d_{x_k y_k}^{k} = \begin{cases} 0 & \text{if } x_k, y_k \text{ represent a nominal or binary variable and } x_k = y_k \\ 1 & \text{if } x_k, y_k \text{ represent a nominal or binary variable and } x_k \neq y_k \\ \frac{|x_k - y_k|}{max(x_k, y_k) - min(x_k, y_k)} & \text{otherwise} \end{cases}$$

If x_k, y_k represent an ordinal variable they are replaced by their integer codes. For example if $x_k \precsim y_k$ then 1 is assigned to x_k and 2 is assigned to y_k

- An example:
  - 2 customers characteristic by sex (nominal), income (numerical), satisfaction (ordinal with levels $Low \precsim Medium \precsim High$) and age (with a missing value (NA))
    - $a = (Female, 3500, Medium, 45)$
    - $b = (Male, 1500, High, NA)$
- Manual calculation:
  - In R $w_k = 1$ for every k as a default value where in this example $k = 1, 2, 3, 4$
  - $\sum_{k=1}^{4} w_k \delta_{x_k y_k}^{k} = 1 * 1 + 1 * 1 + 1 * 1 + 1 * 0 = 1 + 1 + 1 + 0 = 3$
  - $\sum_{k=1}^{4} w_k \delta_{x_k y_k}^{k} d_{x_k y_k}^{k} = 1 * 1 + 1 * \frac{|3500-1500|}{3500-1500} + 1 * \frac{|2-3|}{3-2} + 0 = 3$
  - $d(x, y) = \frac{\sum_{k=1}^{4} w_k \delta_{x_k y_k}^{k} d_{x_k y_k}^{k}}{\sum_{k=1}^{4} w_k \delta_{x_k y_k}^{k}} = \frac{3}{3} = 1$

- **Gower distance** range:
  - $d(x,y) \in [0,1]$
  - If $d(x,y) \longrightarrow 0$ is more similar
  - If $d(x,y) \longrightarrow 1$ is more dissimilar

- Using R

```
customers2 <- tibble(Customer = c("a", "b"),
                     Sex = c("Female", "Male"),
                     Income = c(3500, 1500),
                     Satisfaction = c("Medium", "High"),
                     Age = c(45, NA)) |>
  mutate(Sex = factor(x = Sex,
                      ordered = FALSE),
         Satisfaction = factor(x = Satisfaction,
                               levels = c("Low", "Medium", "High"),
                               ordered = TRUE))
customers2
```

```
# A tibble: 2 x 5
  Customer Sex    Income Satisfaction   Age
  <chr>    <fct>   <dbl> <ord>        <dbl>
1 a        Female   3500 Medium          45
2 b        Male     1500 High            NA
```

- Using R

```
customers2 |>
  select(-Customer) |>
  daisy(metric = "gower")
```

```
Dissimilarities :
  1
2 1

Metric :  mixed ;  Types = N, I, O, I
Number of objects : 2
```

- In this case:
  - `Metric: mixed` because it includes categorical and numerical data
  - For `Types = N, I, O, I` check out `?cluster::dissimilarity.object`[3]
    - `N`: Nominal (factor)
    - `I`: Interval scaled (numeric)
    - `O`: Ordinal (ordered factor)

[3] See (Stevens 1946) and Level of measurement

- Using R

```
customers2 |>
  select(-Customer) |>
  daisy(metric = "gower")
```

```
Dissimilarities :
  1
2 1

Metric :  mixed ;  Types = N, I, O, I
Number of objects : 2
```

- In this case:
  - `Number of objects : 2`
    - There are 2 observations that correspond to customers **a** and **b**: $a = (Female, 3500, Medium, 45)$ and $b = (Male, 1500, High, NA)$

- The original dissimilarity matrix is of dimension 300×300
  - Showing only the relation between the first 5 observations
  - The position (i, j) means the dissimilarity between the observations i and j
    - For example $(4, 3)$, which is equal to 0.425, is the dissimilarity between the observations 4 and 3

```
segmentation_dist <- segmentation |>
  daisy(metric = "gower")

segmentation_dist |>
  as.matrix() |>
  as_tibble() |>
  select(`1`:`5`) |>
  slice(1:5)
```

```
# A tibble: 5 x 5
    `1`    `2`    `3`   `4`   `5`
  <dbl>  <dbl>  <dbl> <dbl> <dbl>
1 0      0.253  0.233  0.262 0.416
2 0.253 0       0.0680 0.413 0.301
3 0.233 0.0680 0       0.425 0.293
4 0.262 0.413  0.425  0      0.227
5 0.416 0.301  0.293  0.227 0
```

```
customers3 <- tibble(Customer = c("a", "b", "c", "d", "e"),
                     Sex = c("Female", "Male", "Female", "Female", "Male"),
                     Income = c(3500, 1500, 200, 450, 5000),
                     Satisfaction = c("Medium", "High", "Low", "Low", "Medium"),
                     Age = c(45, NA, 34, 23, 55)) |>
  mutate(Sex = factor(x = Sex,
                      ordered = FALSE),
         Satisfaction = factor(x = Satisfaction,
                               levels = c("Low", "Medium", "High"),
                               ordered = TRUE))

customers3
```

```
# A tibble: 5 x 5
  Customer Sex    Income Satisfaction   Age
  <chr>    <fct>   <dbl> <ord>        <dbl>
1 a        Female   3500 Medium          45
2 b        Male     1500 High            NA
3 c        Female    200 Low             34
4 d        Female    450 Low             23
5 e        Male     5000 Medium          55
```

- Hierarchical clustering
  - **Method**: Complete Linkage Clustering

```
customers3_dist <- daisy(x = select(customers3, -Customer),
                         metric = "gower")

customers3_dist
```

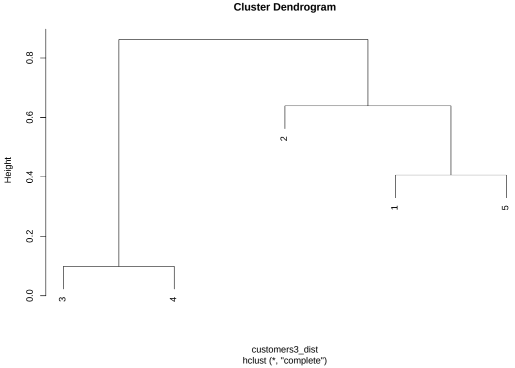
```
Dissimilarities :
           1          2          3          4
2 0.63888889
3 0.38281250 0.75694444
4 0.45572917 0.73958333 0.09895833
5 0.40625000 0.40972222 0.78906250 0.86197917

Metric :  mixed ;  Types = N, I, O, I
Number of objects : 5
```

```
customers3_hc <- hclust(d = customers3_dist,
                        method = "complete")

customers3_hc
```

```
Call:
hclust(d = customers3_dist, method = "complete")

Cluster method   : complete
Number of objects: 5
```

- Hierarchical clustering
  - **Method**: Complete Linkage Clustering

```
plot(customers3_hc)
```



- Compare each observation and find the pair that is more similar

	1	2	3	4	5
1	0.0000000	0.6388889	0.3828125	0.4557292	0.4062500
2	0.6388889	0.0000000	0.75694444	0.7395833	0.4097222
3	0.3828125	0.7569444	0	0.0989583	0.7890625
4	0.4557292	0.7395833	0.09895833	0.0000000	0.8619792
5	0.4062500	0.4097222	0.7890625	0.8619792	0.0000000

- Now we have the first cluster that includes the observations 3 and 4: $C(3,4)$
- Then we need to create clusters with observations 1, 2 and 5 and the cluster $C(3,4)$
  - How we compare a cluster with an observation
    - **Complete Linkage Clustering**: Use the maximum distance between an observation and an observation that belongs to the cluster

- Compare each observation, including the clusters build, and find the pair that is more similar
  - In our case 1, 2, 5 and $C(3,4)$
    - The distance between 1 and $C(3,4)$ is 0.45572917
    - The distance between 2 and $C(3,4)$ is 0.7569444
    - The distance between 5 and $C(3,4)$ is 0.8619792

	1	2	3	4	5
1	0	0.6388889	0.3828125	0.4557292	0.4062500
2	0.63888889	0.0000000	0.75694444	0.7395833	0.4097222
3	0.3828125	0.7569444	0	0.0989583	0.7890625
4	0.45572917	0.7395833	0.09895833	0.0000000	0.8619792
5	0.40625	0.4097222	0.7890625	0.8619792	0.0000000

- Now we have the second cluster that includes the observations 1 and 5: $C(1,5)$
- Then we need to create clusters with observation 2 and clusters $C(3,4)$ and $C(1,5)$
  - How we compare a cluster with another cluster
    - **Complete Linkage Clustering**: Use the maximum distance between an observation that belongs to the first cluster and an observation that belongs to the second cluster

- Compare each observation, including the clusters build, and find the pair that is more similar
  - In our case 2, $C(3,4)$ and $C(1,5)$
    - The distance between 2 and $C(3,4)$ is 0.7569444
    - The distance between 2 and $C(1,5)$ is 0.6388889

	1	2	3	4	5
1	0	0.6388889	0.3828125	0.4557292	0.4062500
2	0.63888889	0.0000000	0.75694444	0.7395833	0.4097222
3	0.3828125	0.7569444	0	0.0989583	0.7890625
4	0.45572917	0.7395833	0.09895833	0.0000000	0.8619792
5	0.40625	0.4097222	0.7890625	0.8619792	0.0000000

- Now we have the third cluster that includes the observation 2 and the cluster $C(1,5)$: $C(2,C(1,5))$
- Then we need to create clusters with cluster $C(2,C(1,5))$ and cluster $C(3,4)$
  - This is the cluster that includes all the observations

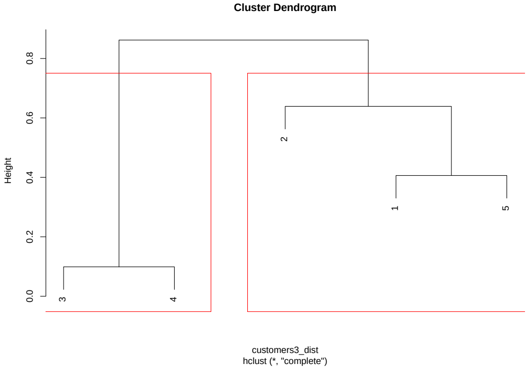
- Compare each observation, including the clusters build, and find the pair that is more similar
    - In our case $C(3,4)$ and $C(2,C(1,5))$
        - The distance between $C(3,4)$ and $C(2,C(1,5))$ is 0.86197917

	1	2	3	4	5
1	0	0.6388889	0.3828125	0.45572917	0.4062500
2	0.63888889	0.0000000	0.75694444	0.73958333	0.4097222
3	0.3828125	0.7569444	0	0.09895833	0.7890625
4	0.45572917	0.7395833	0.09895833	0	0.8619792
5	0.40625	0.4097222	0.7890625	0.86197917	0.0000000

- The heights of the **Cluster Dendrogram** are: 0.09895833, 0.40625, 0.63888889 and 0.86197917

- Select a number of clusters, for example: 2 clusters

```
plot(customers3_hc)
rect.hclust(customers3_hc, k = 2, border = "red")
```



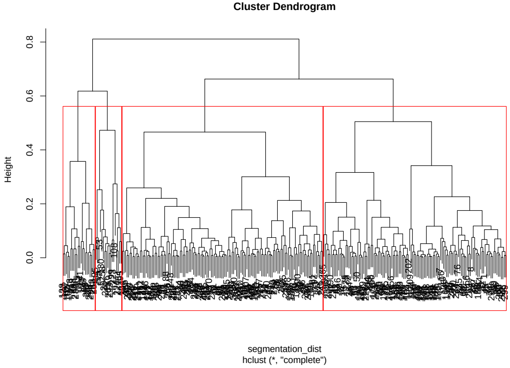
- Extract clusters and assign them to observations

```
customers3_hc_clusters <- cutree(customers3_hc, k = 2)
customers3 |>
  mutate(cluster = customers3_hc_clusters)
```

```
# A tibble: 5 x 6
  Customer Sex    Income Satisfaction   Age cluster
  <chr>    <fct>   <dbl> <ord>        <dbl>   <int>
1 a        Female   3500 Medium          45       1
2 b        Male     1500 High            NA       1
3 c        Female    200 Low             34       2
4 d        Female    450 Low             23       2
5 e        Male     5000 Medium          55       1
```

- Select a number of clusters, using `segmentation`, for example: 4 clusters

```
segmentation_hc <- hclust(d = segmentation_dist,
                          method = "complete")
plot(segmentation_hc)
rect.hclust(segmentation_hc, k = 4, border = "red")
```



- Extract clusters and assign them to observations, using `segmentation`

```
segmentation_hc_clusters <- cutree(segmentation_hc, k = 4)
segmentation |>
  mutate(cluster = segmentation_hc_clusters)
```

```
# A tibble: 300 x 7
     age gender income  kids ownHome subscribe cluster
   <dbl> <fct>   <dbl> <int> <fct>   <fct>       <int>
 1  47.3 Male   49483.     2 ownNo   subNo           1
 2  31.4 Male   35546.     1 ownYes  subNo           1
 3  43.2 Male   44169.     0 ownYes  subNo           1
 4  37.3 Female 81042.     1 ownNo   subNo           2
 5  41.0 Female 79353.     3 ownYes  subNo           2
 6  43.0 Male   58143.     4 ownYes  subNo           1
 7  37.6 Male   19282.     3 ownNo   subNo           1
 8  28.5 Male   47245.     0 ownNo   subNo           1
 9  44.2 Female 48333.     1 ownNo   subNo           2
10  35.2 Female 52568.     0 ownYes  subNo           2
# i 290 more rows
```

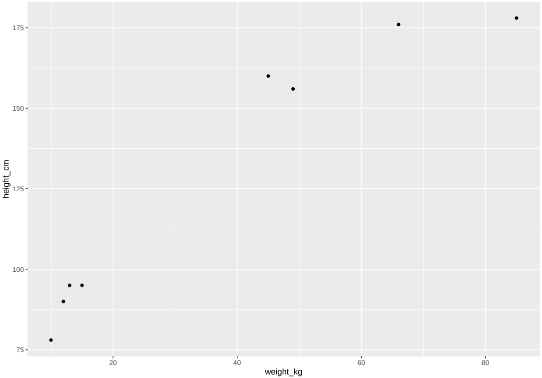
- K-means clustering example (Kaufman and Rousseeuw 1990, 5)

```
kaufman_example <- tibble(name = c("Ilan", "Jacqueline", "Kim", "Lieve", "Leon", "Peter", "Talia", "Tina"),
                          weight_kg = c(15, 49, 13, 45, 85, 66, 12, 10),
                          height_cm = c(95, 156, 95, 160, 178, 176, 90, 78))

kaufman_example
```

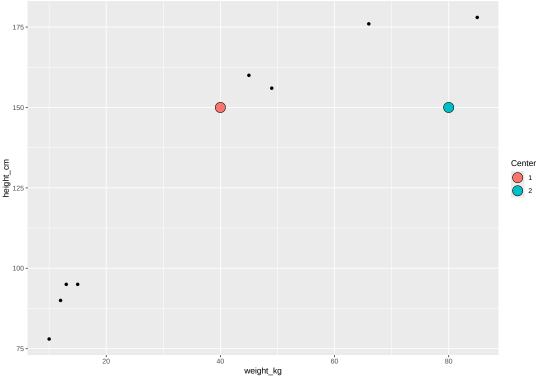
```
# A tibble: 8 x 3
  name       weight_kg height_cm
  <chr>          <dbl>     <dbl>
1 Ilan              15        95
2 Jacqueline        49       156
3 Kim               13        95
4 Lieve             45       160
5 Leon              85       178
6 Peter             66       176
7 Talia             12        90
8 Tina              10        78
```

- K-means clustering example (Kaufman and Rousseeuw 1990, 5)

```
kaufman_example |>
  ggplot() +
  geom_point(aes(x = weight_kg, y = height_cm))
```

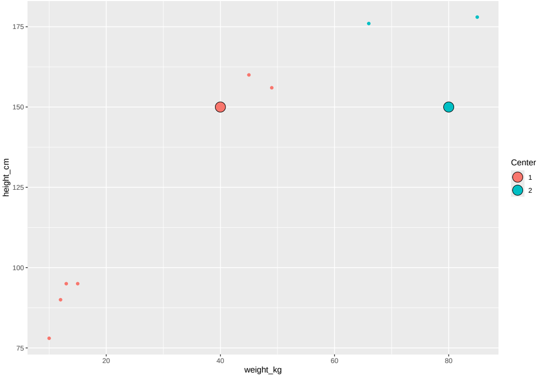


- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Lloyd's algorithm**
    - Choose k centers or the computer will choose k centers at random, in our case we choose $k = 2$



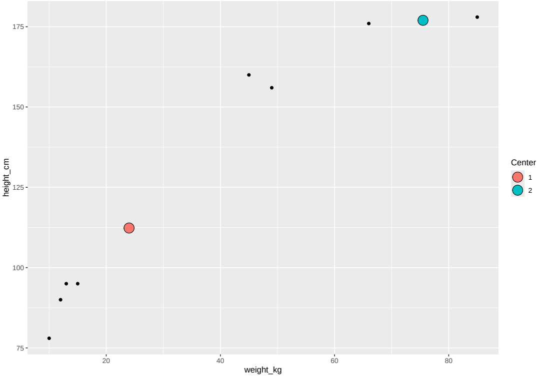
- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
    - Applying the **Lloyd's algorithm**
        - Calculate the squared euclidean distance for each point to the k centers and assign each point to the nearest center
        - For example for the point $Ilan = (15, 95)$ the distance to $Center_1 = (40, 150)$ is $(15 - 40)^2 + (95 - 150)^2 = 3650$ and the distance to $Center_2 = (80, 150)$ is $(15 - 80)^2 + (95 - 150)^2 = 7250$
        - Therefore $Ilan = (15, 95)$ is assigned to $Center_1$

- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
    - Applying the **Lloyd's algorithm**

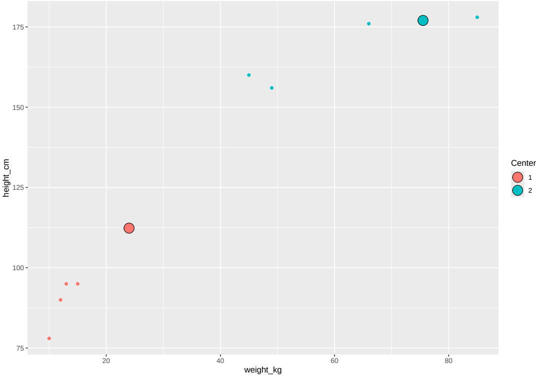


- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Lloyd's algorithm**
    - Now calculate new centers using the assigned points by using the mean
    - For example for the new $Center_1$ the new position will be $x = \frac{15+49+13+45+12+10}{6} = 24$ and $y = \frac{95+156+95+160+90+78}{6} \approx 112.33$
    - Therefore we update as $Center_1 \approx (24, 112.33)$

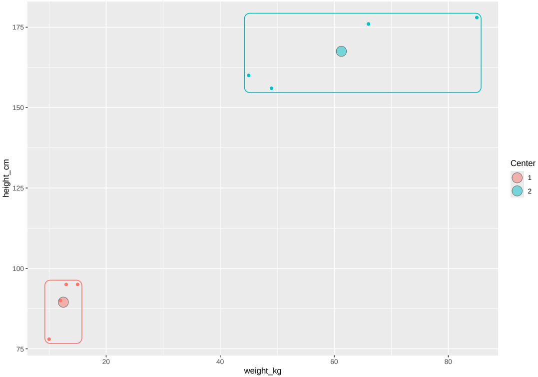
- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Lloyd's algorithm**



- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Lloyd's algorithm**
  - Repeat the process by calculating the squared euclidean distance for each point to the new k centers and assign each point to the nearest center



- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Lloyd's algorithm**
  - Repeat the process until the k centers don't change and assign each point to the nearest final center



- K-means clustering example (Kaufman and Rousseeuw 1990, 5)
  - Applying the **Hartigan-Wong algorithm**

```
kaufman_example_kmeans <- kaufman_example |>
  select(weight_kg, height_cm) |>
  kmeans(centers = 2,
         algorithm = "Hartigan-Wong") # R uses this algorithm by default

kaufman_example_kmeans
```

```
K-means clustering with 2 clusters of sizes 4, 4

Cluster means:
  weight_kg height_cm
1     61.25     167.5
2     12.50      89.5

Clustering vector:
[1] 2 1 2 1 1 1 2 2

Within cluster sum of squares by cluster:
[1] 1371.75  206.00
 (between_SS / total_SS =  91.5 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

- Extract clusters and assign them to observations

```
kaufman_example_kmeans_clusters <- kaufman_example |>
  mutate(cluster = kaufman_example_kmeans$cluster)
kaufman_example_kmeans_clusters
```

```
# A tibble: 8 x 4
  name       weight_kg height_cm cluster
  <chr>          <dbl>     <dbl>   <int>
1 Ilan              15        95       2
2 Jacqueline        49       156       1
3 Kim               13        95       2
4 Lieve             45       160       1
5 Leon              85       178       1
6 Peter             66       176       1
7 Talia             12        90       2
8 Tina              10        78       2
```

- Select a number of clusters, using `segmentation`, for example: 4 clusters
  - k-means only work with numerical data
  - A possible solution is to transform categorical data into numerical data
    - If a variable is nominal only works if you have 2 categories
    - If a variable is ordinal you assume that the notion of distance between them is constant or you need to specify integers to determine what distance is appropiate
    - Also you need to scale the variables taking into account that you are mixing categorical and numerical variables

- Convert binary nominal data to numerical data
  - Only make sense when you have 2 categories

```
segmentation_numeric <- segmentation |>
  mutate(gender = as.integer(gender),
         ownHome = as.integer(ownHome),
         subscribe = as.integer(subscribe))

segmentation_numeric
```

```
# A tibble: 300 x 6
     age gender income  kids ownHome subscribe
   <dbl>  <int>  <dbl> <int>   <int>     <int>
 1  47.3      2 49483.     2       1         1
 2  31.4      2 35546.     1       2         1
 3  43.2      2 44169.     0       2         1
 4  37.3      1 81042.     1       1         1
 5  41.0      1 79353.     3       2         1
 6  43.0      2 58143.     4       2         1
 7  37.6      2 19282.     3       1         1
 8  28.5      2 47245.     0       1         1
 9  44.2      1 48333.     1       1         1
10  35.2      1 52568.     0       2         1
# i 290 more rows
```

- Scale data to map each variable to a common scale
    - We are going to scale each variable to $[0, 1]$
        - Use across and rescale

```
segmentation_numeric_scale <- segmentation_numeric |>
  mutate(across(.cols = age:subscribe,
                # scales is a package that is
                # installed with the tidyverse
                # but it is not loaded automatically
                # You can use a particular function of a package using the notation
                ## <package>::<function>
                .fns = scales::rescale))

segmentation_numeric_scale |> head()
```

```
# A tibble: 6 x 6
    age gender income  kids ownHome subscribe
  <dbl>  <dbl>  <dbl> <dbl>   <dbl>     <dbl>
1 0.458      1  0.458 0.286       0         0
2 0.198      1  0.341 0.143       1         0
3 0.391      1  0.413 0           1         0
4 0.295      0  0.722 0.143       0         0
5 0.354      0  0.708 0.429       1         0
6 0.388      1  0.530 0.571       1         0
```

- Apply k-means with $k = 4$ and **Hartigan-Wong algorithm**
  - k-means start with $k = 4$ random centers so you need to fix this initial decision using `set.seed` if the clusters tend to change

```
set.seed(seed = 1234)

segmentation_numeric_scale_kmeans <- segmentation_numeric_scale |>
  kmeans(centers = 4,
         algorithm = "Hartigan-Wong")

segmentation_numeric_scale_kmeans |> str()
```

```
List of 9
 $ cluster     : int [1:300] 2 3 3 4 1 3 2 2 4 1 ...
 $ centers     : num [1:4, 1:6] 0.431 0.278 0.446 0.298 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4] "1" "2" "3" "4"
  .. ..$ : chr [1:6] "age" "gender" "income" "kids" ...
 $ totss       : num 218
 $ withinss    : num [1:4] 18.6 17.5 14.4 15.4
 $ tot.withinss: num 65.9
 $ betweenss   : num 152
 $ size        : int [1:4] 76 78 65 81
 $ iter        : int 3
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

- Extract clusters and assign them to observations

```
segmentation_kmeans_clusters <- segmentation |>
  mutate(cluster = segmentation_numeric_scale_kmeans$cluster)

segmentation_kmeans_clusters
```

```
# A tibble: 300 x 7
     age gender income  kids ownHome subscribe cluster
   <dbl> <fct>   <dbl> <int> <fct>   <fct>       <int>
 1  47.3 Male   49483.     2 ownNo   subNo           2
 2  31.4 Male   35546.     1 ownYes  subNo           3
 3  43.2 Male   44169.     0 ownYes  subNo           3
 4  37.3 Female 81042.     1 ownNo   subNo           4
 5  41.0 Female 79353.     3 ownYes  subNo           1
 6  43.0 Male   58143.     4 ownYes  subNo           3
 7  37.6 Male   19282.     3 ownNo   subNo           2
 8  28.5 Male   47245.     0 ownNo   subNo           2
 9  44.2 Female 48333.     1 ownNo   subNo           4
10  35.2 Female 52568.     0 ownYes  subNo           1
# i 290 more rows
```

- To my family that supports me
- To the taxpayers of Colombia and the **UMNG students** who pay my salary
- To the **Business Science** and **R4DS Online Learning** communities where I learn **R** and π**-thon**
- To the **R Core Team**, the creators of **RStudio IDE**, **Quarto** and the authors and maintainers of the packages **tidyverse**, **skimr**, **latex2exp**, **kableExtra**, **cluster** and **tinytex** for allowing me to access these tools without paying for a license
- To the **Linux kernel community** for allowing me the possibility to use some **Linux distributions** as my main **OS** without paying for a license

References I

Chapman, Chris, and Elea McDonnell Feit. 2019. *R For Marketing Research and Analytics*. 2nd ed. 2019. Use R! Cham: Springer International Publishing : Imprint: Springer. https://doi-org.ezproxy.umng.edu.co/10.1007/978-3-030-14316-9.

Deza, Michel Marie, and Elena Deza. 2016. *Encyclopedia of Distances*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-52844-0.

Kaufman, Leonard, and Peter J. Rousseeuw. 1990. *Finding Groups in Data: An Introduction to Cluster Analysis*. 1st ed. Wiley Series in Probability and Statistics. Wiley. https://doi.org/10.1002/9780470316801.

Stevens, S. S. 1946. "On the Theory of Scales of Measurement." *Science* 103 (2684): 677–80. https://doi.org/10.1126/science.103.2684.677.