K-Means Clustering in R

RStudio Instructor Certification Teaching Exam

Uli Niemann

2020/08/21

Photo courtesy of Ulrich Arendt

Learner Persona

(**Course:** "Introduction to Data Mining with R")

Meet Nathalie...

1. Background: Nathalie is a 22-year-old undergraduate student pursuing a Bachelor's degree in "Operations Research and Business Analytics". She is born in Turkey, and moved to Germany to take up her study.

2. Relevant Experience: Currently, Nathalie is in her 5th semester. Until now, she has been analyzing data mainly with Excel, and has some working knowledge of SPSS, but she has never programmed in R prior to taking this course.

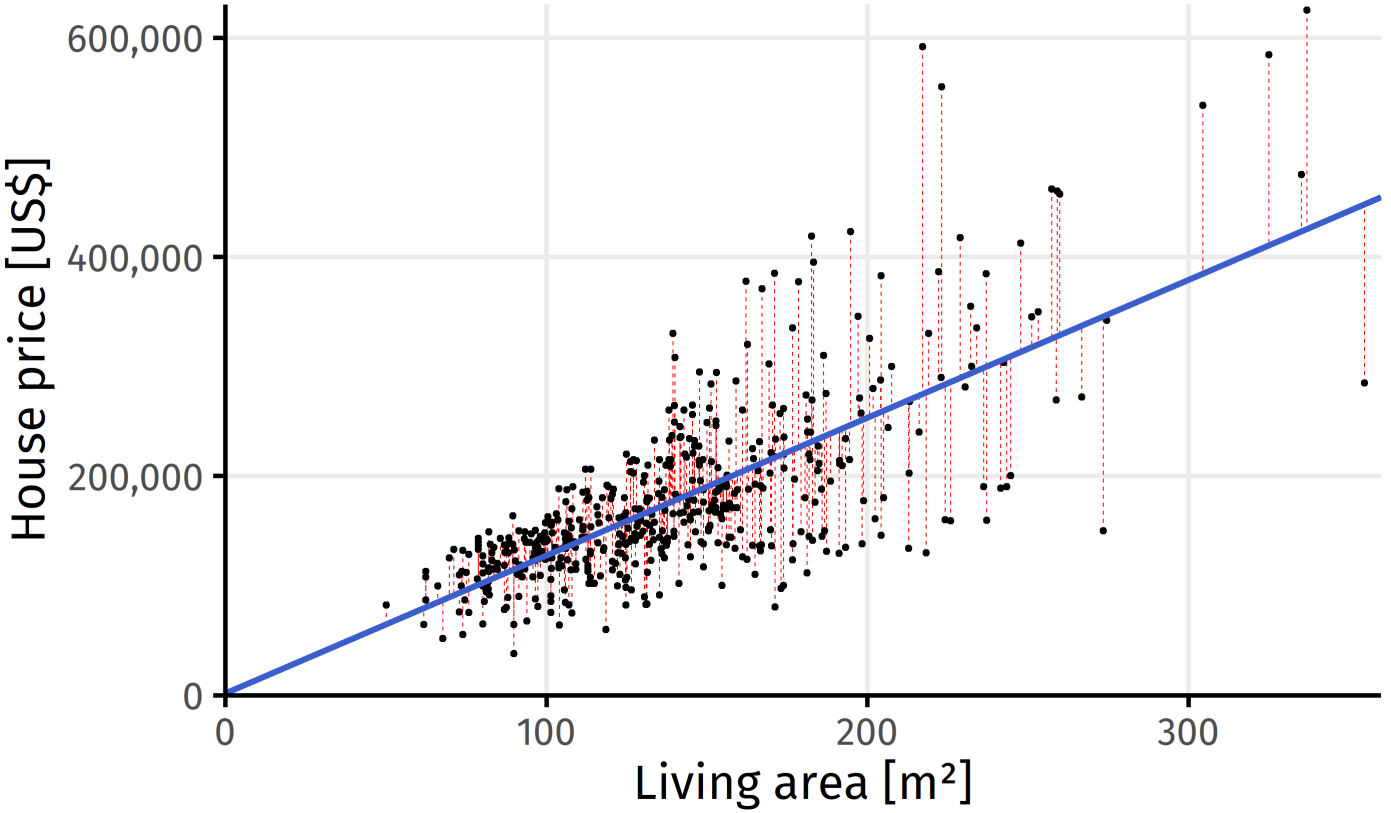
3. Needs: Nathalie is enrolled in this course which is required for her degree, so she needs to pass it. However, she is also interested in this course because she knows that showing R proficiency on her CV will increase her value on the job market. She needs a gentle introduction to R, also focusing on differences between the tools she currently uses and a programming language (*What are loops?*, *What is a function?*, ...).

4. Special considerations: Nathalie has little experience in programming and she doesn't have a very powerful laptop.

Recap: Supervised Learning

General idea: train a model which estimates the value of the response based on the values of the predictors.

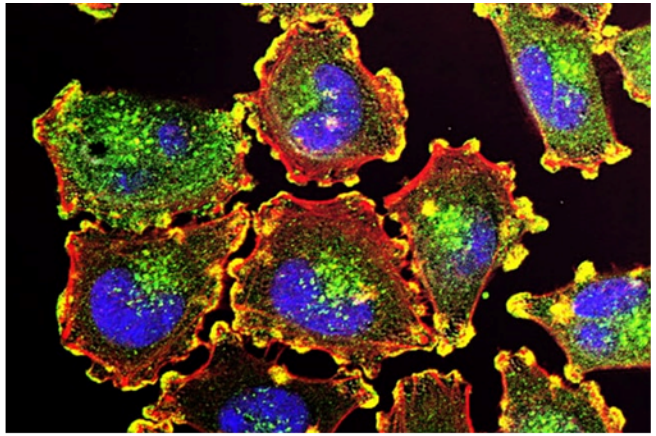
$$\hat{Y} = f(\mathbf{X})$$



Unsupervised Learning

Goal: look for interesting patterns in the data without having a response

Customer Subgroups



Disease Phenotypes

Website Usage Patterns

Clustering: methods to find homogeneous subgroups in a dataset

Properties of a good clustering results:

- objects within a cluster are as **similar** as possible
- objects from different clusters are as **dissimilar** as possible

→ require: **similarity measure**

Photos: left: Lucrezia Carnelos; center: National Cancer Institute; right: Agence Olloweb

K-Means Clustering

General idea: partition the observations into a pre-specified number of subgroups (clusters)

Centroid: mean vector of a cluster's observations

Objective: minimize total intra-cluster variation

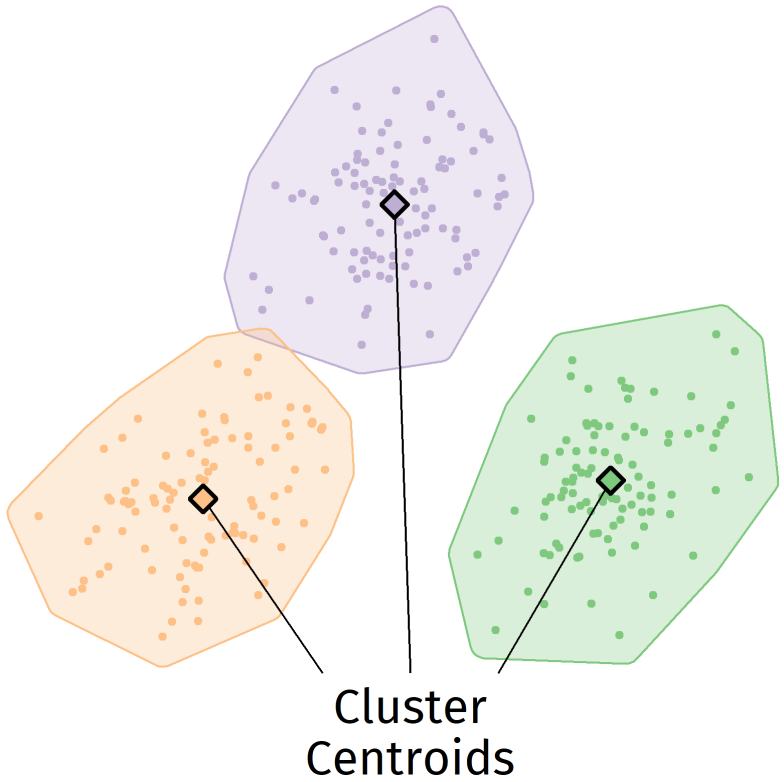
$$\min_{C_1, C_2, \ldots, C_K} \left\{ \sum_k W(C_k) \right\}$$

Squared **Euclidean distance** is often used to measure the **within-cluster variation** W of a cluster C_k:

$$W(C_k) = \sum_{i \in C_k} \sum_{j}^{p} (x_{ij} - \bar{x}_{kj})^2$$

where

- i is the index of an observation assigned to C_k
- $\bar{x}_k$ is the centroid of C_k

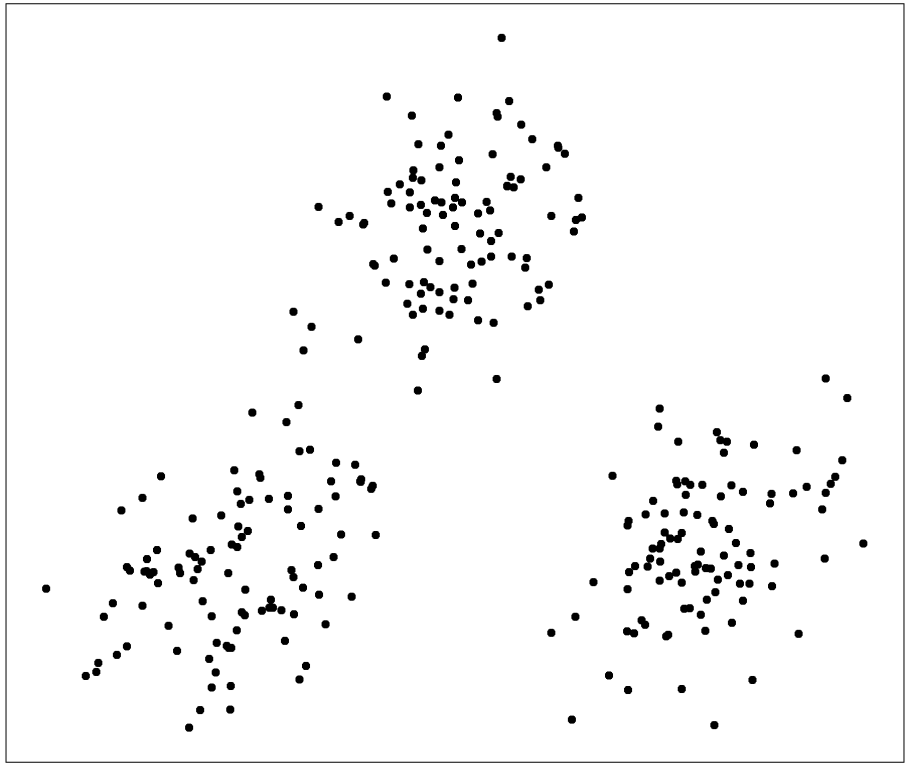


K-Means Algorithm

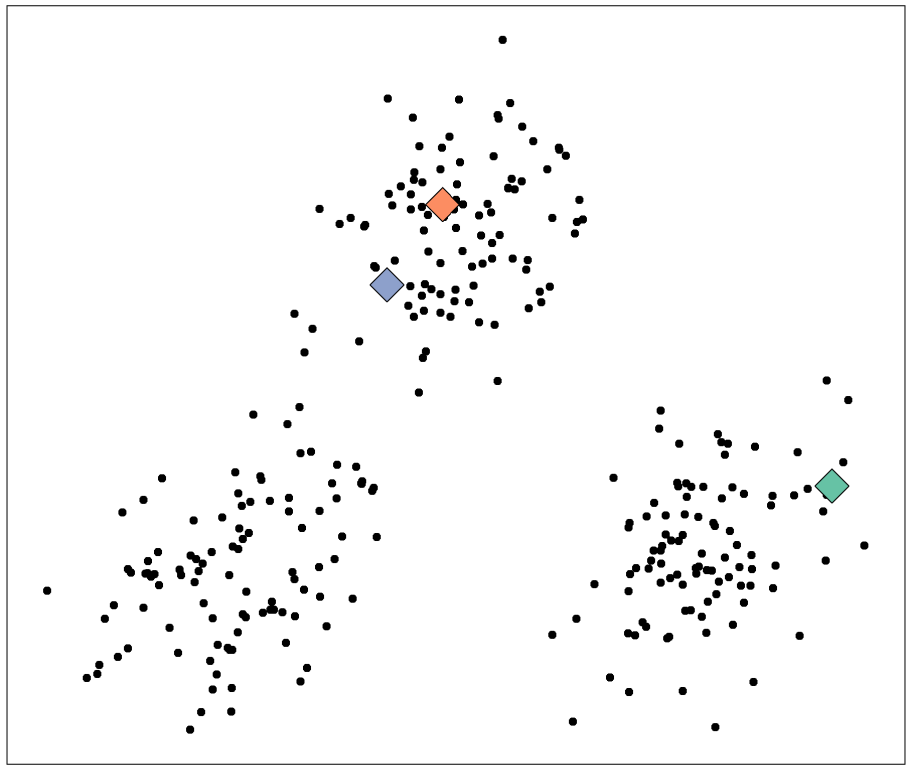
```
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.
```

Example: K=3

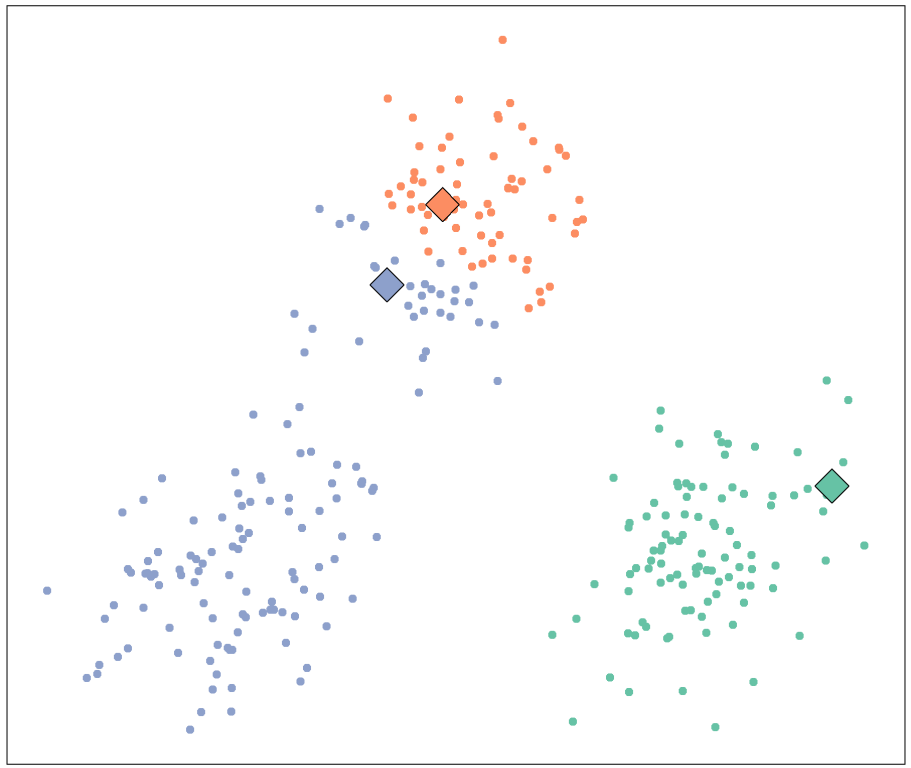
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



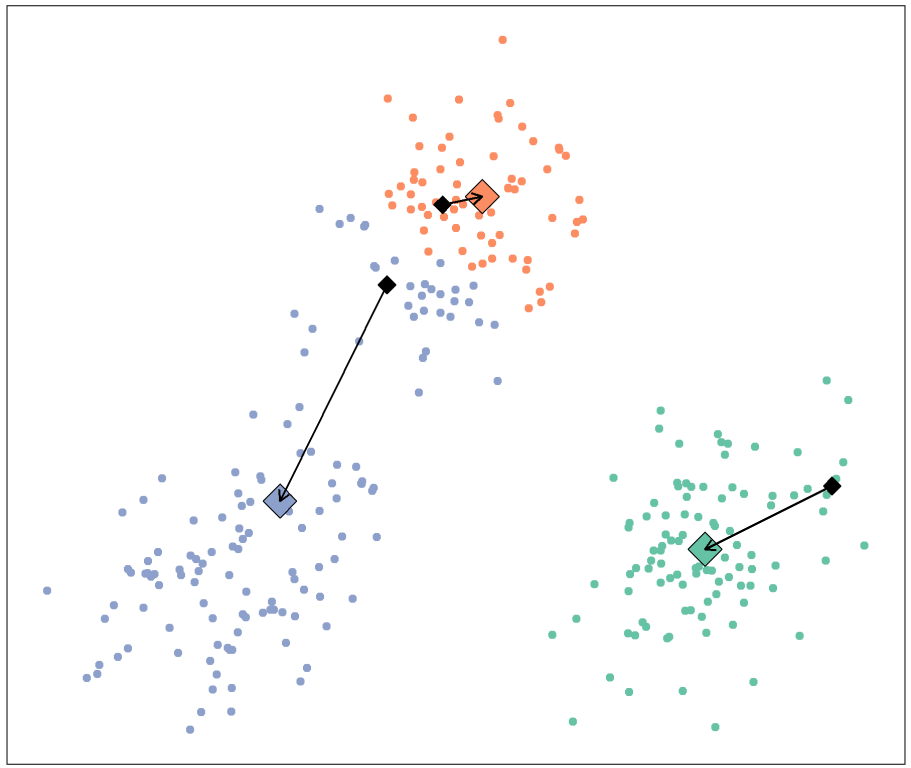
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



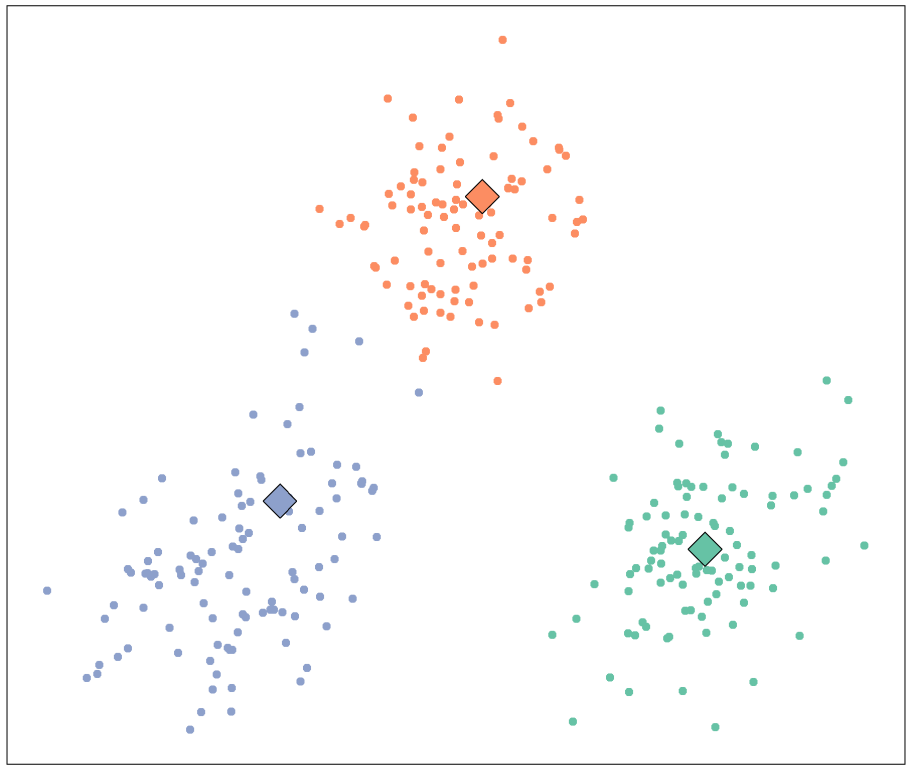
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



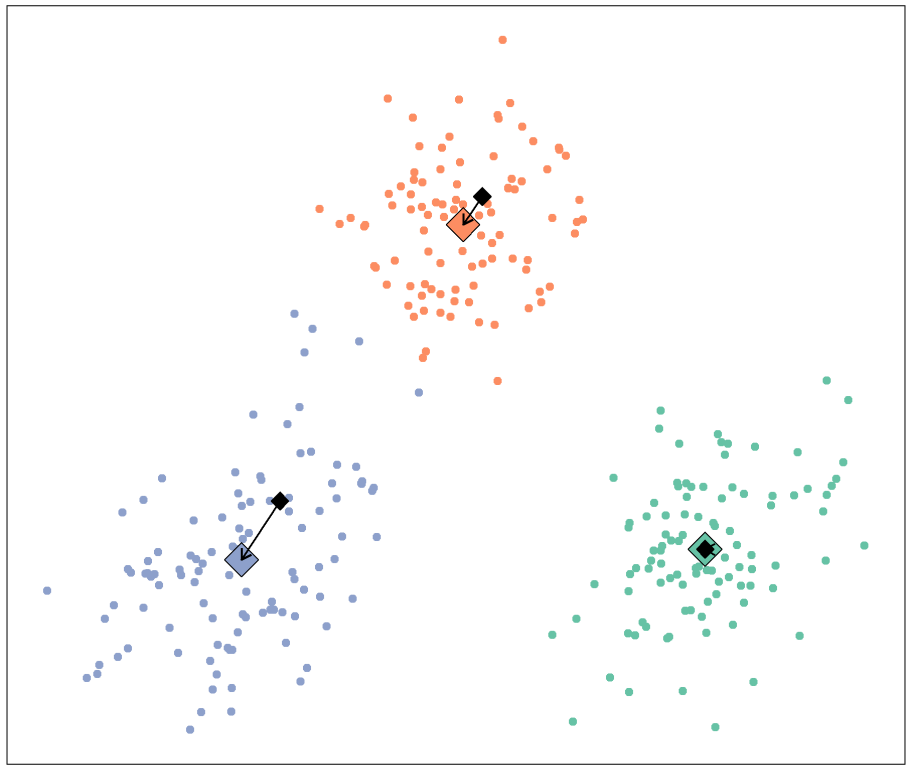
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



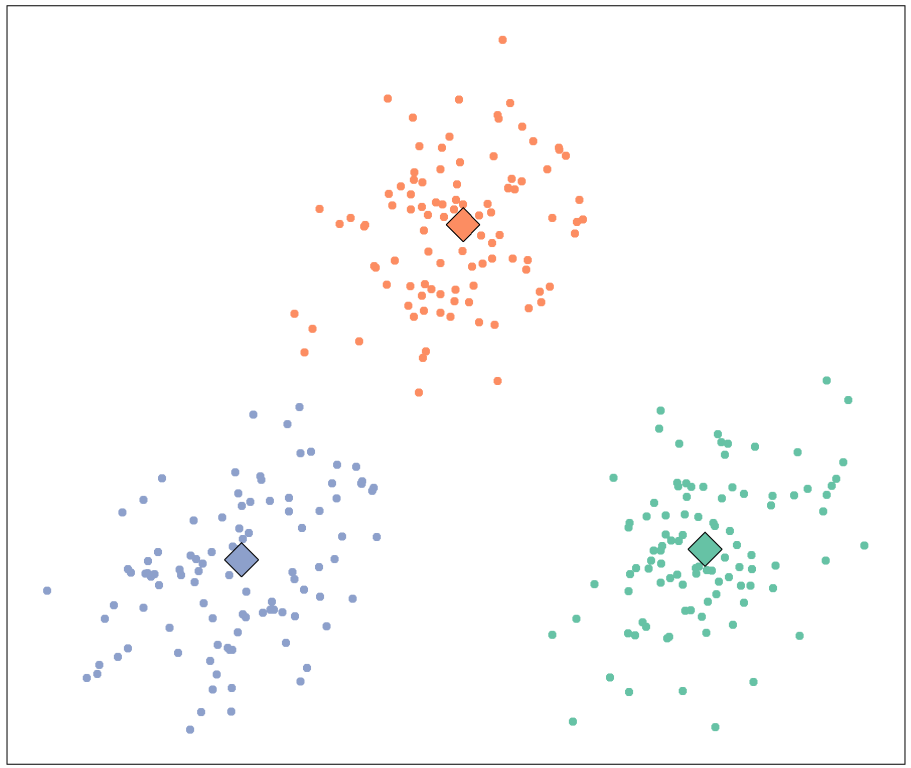
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



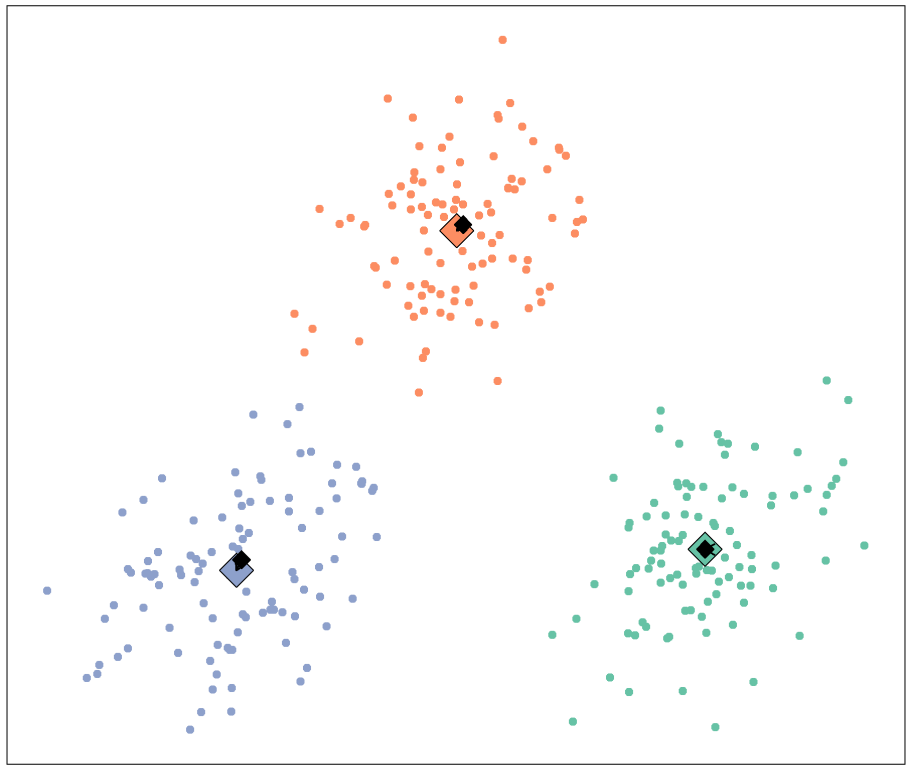
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



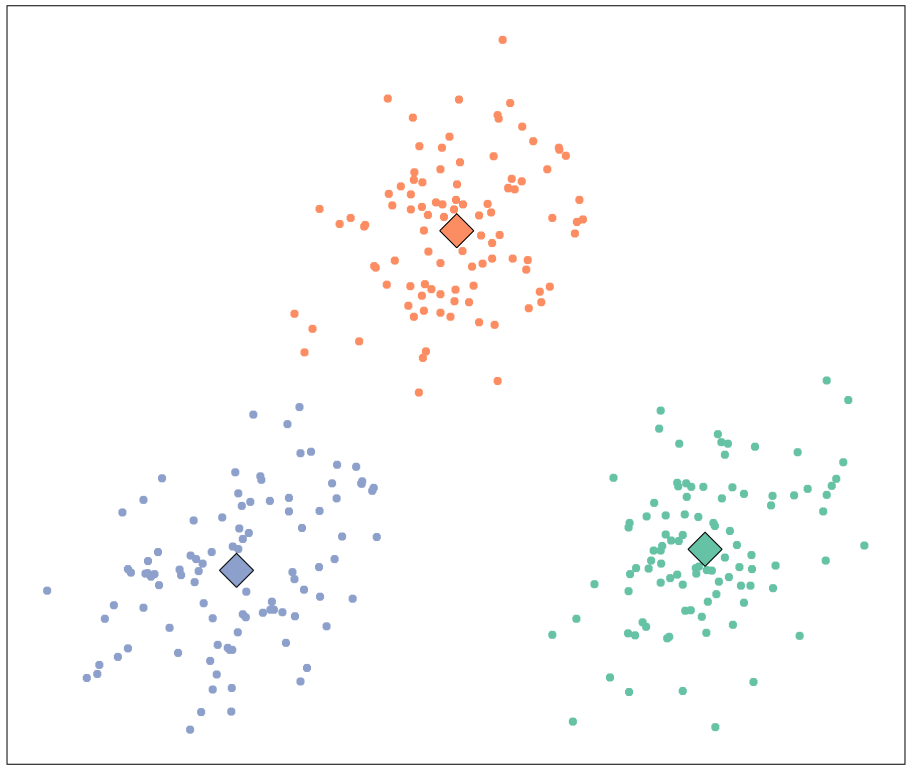
1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



1. Select K initial centroids.
2. Repeat until the cluster assignments do not change anymore:
    a. Assign each observation to the closest centroid.
    b. Recompute the positions of all centroids.



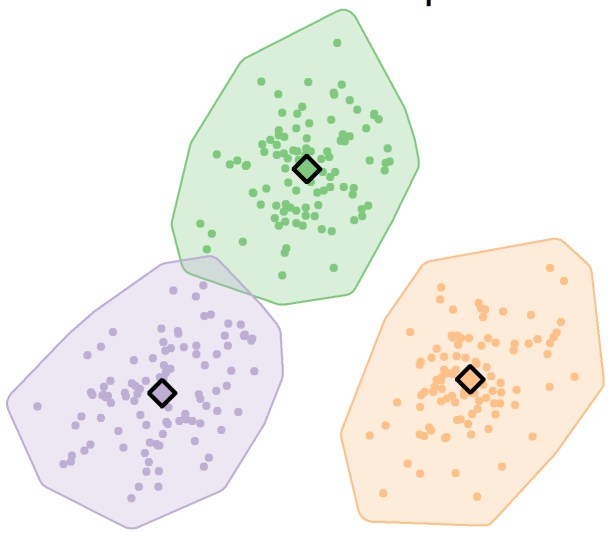
Properties of K-Means

K-means finds a **local** rather than a global optimum.

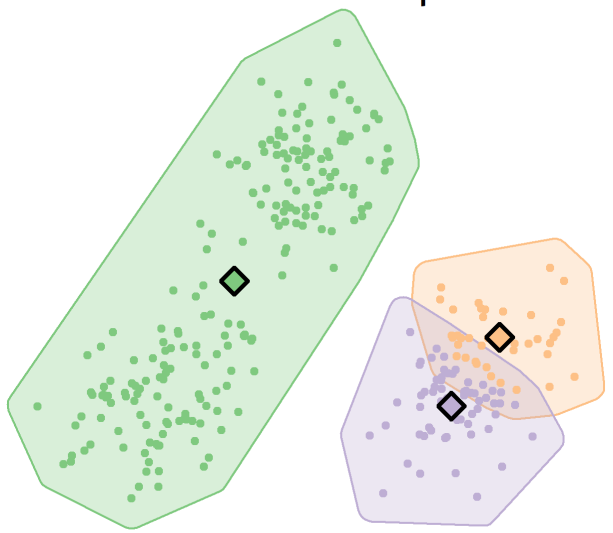
The clustering result will depend on the **initial (random) positions of the centroids**.

⟶ Run the algorithm **multiple times** from different initial configurations and select the result which yields minimum total intra-cluster variation.

Good initial centroid positions

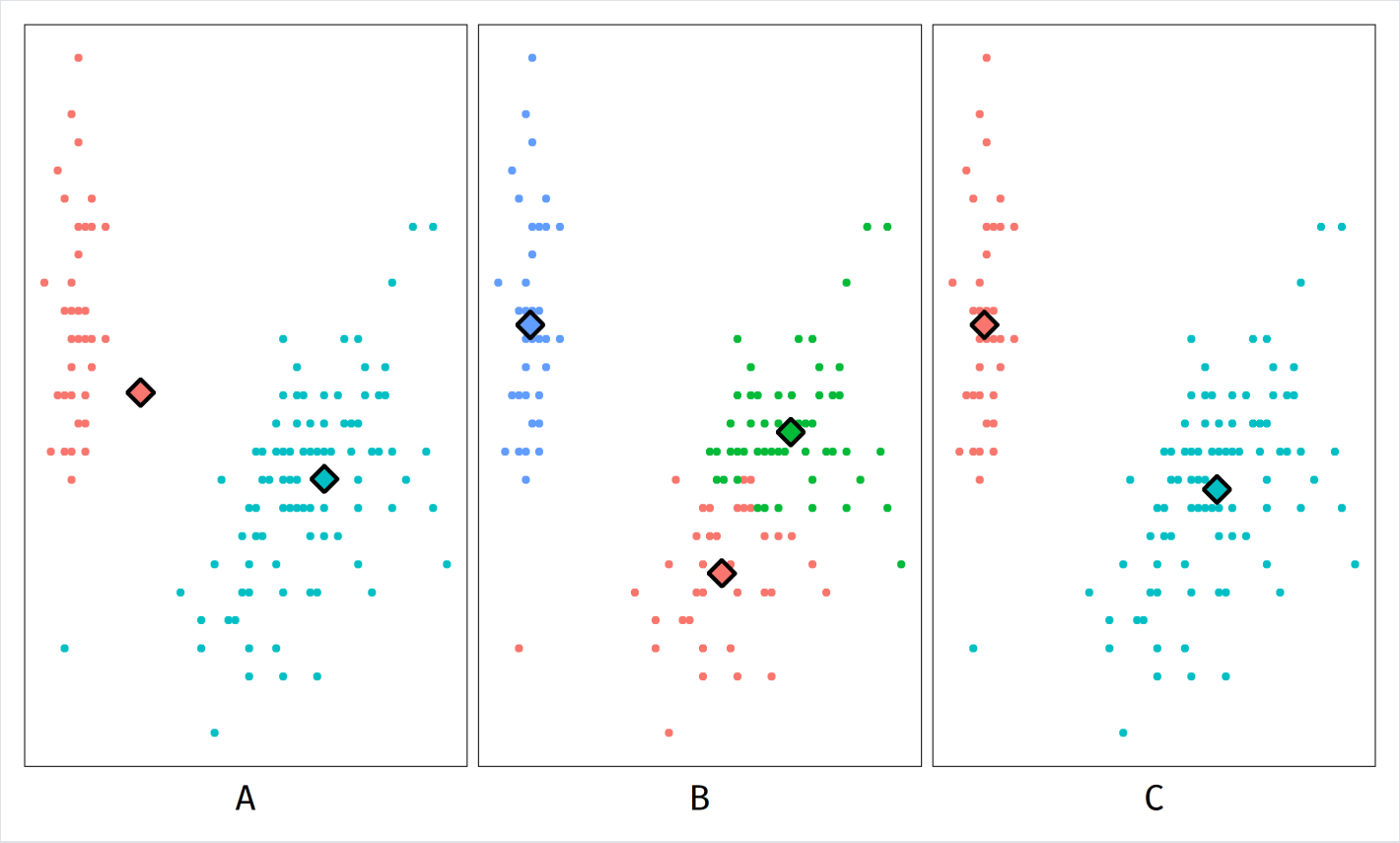


Poor initial centroid positions



Your Turn

Which of the plots below shows the correct result of K-means clustering with K=2? A, B or C?



K-Means in `R`

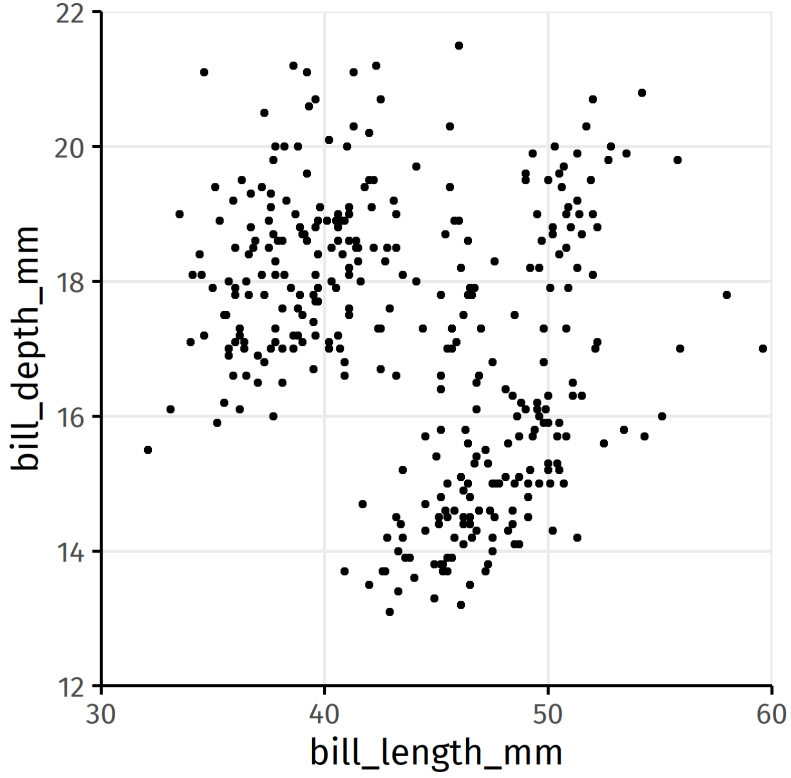
The `kmeans()` function takes two mandatory arguments:

- `x`: the data (data frame or numeric matrix)
- `centers`: number of clusters K or a set of initial centroids

Example: Clustering Palmer Penguins Bill Sizes

```
# remotes::install_github("allisonhorst/palmerpenguins")
(df <- palmerpenguins::penguins %>%
  select(bill_length_mm, bill_depth_mm) %>%
  drop_na())
```

```
## # A tibble: 342 x 2
##    bill_length_mm bill_depth_mm
##             <dbl>         <dbl>
##  1           39.1          18.7
##  2           39.5          17.4
##  3           40.3          18
##  4           36.7          19.3
##  5           39.3          20.6
##  6           38.9          17.8
##  7           39.2          19.6
##  8           34.1          18.1
##  9           42            20.2
## 10           37.8          17.1
## # ... with 332 more rows
```



K-Means in R

```
(clustering <- kmeans(x = df, centers = 3))
```

```
## K-means clustering with 3 clusters of sizes 116, 141, 85
##
## Cluster means:
##   bill_length_mm bill_depth_mm
## 1       45.51379      15.64397
## 2       38.40355      18.27943
## 3       50.90353      17.33647
##
## Clustering vector:
##   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [43] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 1 2 2 2
##  [85] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2
## [127] 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 3 1 3 1 1 1 1 1 1 1 3 1 1 1 3 1
## [169] 3 1 3 3 1 1 1 1 1 1 1 1 3 1 1 1 3 3 3 1 1 1 3 1 3 1 3 3 1 1 3 1 1 1 1 1 3 1 1 1 1 1
## [211] 3 1 1 1 3 1 3 1 3 1 3 1 1 1 1 1 3 1 3 1 1 3 3 1 3 1 3 1 3 1 3 1 3 1 3 3 1 1 3 1
## [253] 3 1 3 1 1 1 3 1 1 3 3 1 3 1 3 1 3 1 1 3 1 3 1 3 3 1 3 1 1 3 1 3 1 3 1 3 1 3 3 3 1 3
## [295] 1 3 1 3 1 3 3 3 1 3 2 3 1 3 3 1 1 3 1 3 3 1 3 1 3 3 3 3 3 3 1 3 1 3 1 3 1 3 3 1 3 1
## [337] 1 3 1 3 3 3
##
## Within cluster sum of squares by cluster:
## [1] 754.7437 944.4986 617.9859
##  (between_SS / total_SS =  79.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Components of a `kmeans` object

```
str(clustering)
```

```
## List of 9
##  $ cluster     : int [1:342] 2 2 2 2 2 2 2 2 2 2 ...
##  $ centers     : num [1:3, 1:2] 45.5 38.4 50.9 15.6 18.3 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:2] "bill_length_mm" "bill_depth_mm"
##  $ totss       : num 11494
##  $ withinss    : num [1:3] 755 944 618
##  $ tot.withinss: num 2317
##  $ betweenss   : num 9177
##  $ size        : int [1:3] 116 141 85
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
str(clustering)
```

```
## List of 9
##  $ cluster     : int [1:342] 2 2 2 2 2 2 2 2 2 2 ...
##  $ centers     : num [1:3, 1:2] 45.5 38.4 50.9 15.6 18.3 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:2] "bill_length_mm" "bill_depth_mm"
##  $ totss       : num 11494
##  $ withinss    : num [1:3] 755 944 618
##  $ tot.withinss: num 2317
##  $ betweenss   : num 9177
##  $ size        : int [1:3] 116 141 85
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

- `cluster`: cluster ID for each observation
- `centers`: centroid positions

```
df %>%
  mutate(cluster = as.factor(clustering$cluster)) %>%
  ggplot(aes(bill_length_mm, bill_depth_mm)) +
  geom_point(aes(color = cluster)) +
  geom_point(data = as_tibble(clustering$centers),
             pch = 18, size = 5) +
  guides(color = FALSE)
```

```
str(clustering)
```

```
## List of 9
##  $ cluster     : int [1:342] 2 2 2 2 2 2 2 2 2 2 ...
##  $ centers     : num [1:3, 1:2] 45.5 38.4 50.9 15.6 18.3 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:2] "bill_length_mm" "bill_depth_mm"
##  $ totss       : num 11494
##  $ withinss    : num [1:3] 755 944 618
##  $ tot.withinss: num 2317
##  $ betweenss   : num 9177
##  $ size        : int [1:3] 116 141 85
##  $ iter        : int 2
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

- `totss`: total sum of squares (sum of squared distances to global mean over all observations)
- `withinss`: within sum of squares (sum of squared distances to cluster centroid)
- `tot.withinss`: sum of `withinss`
- `betweenss`: `totss` - `tot.withinss` (reduction in sum of squares)
- `size`: number of cluster members
- `iter`: number of iterations until convergence

Your Turn (exercise.R)

Run K-means clustering on the Palmer penguins data with K=3.

Instead of initializing the centroids randomly, use to provided custom coordinates.

Return the number of observations for each cluster.

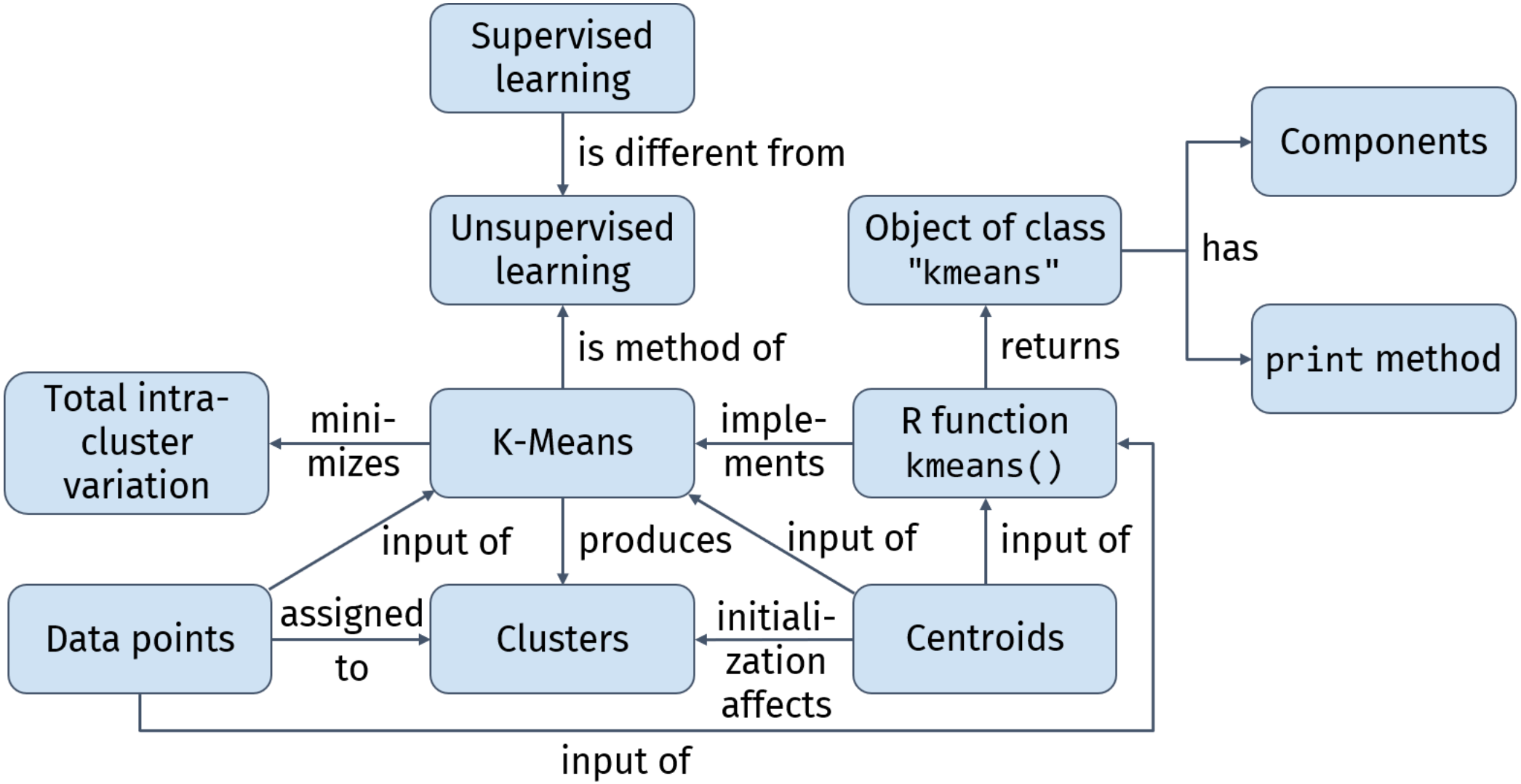
```
# remotes::install_github("allisonhorst/palmerpenguins")
library(dplyr)
library(tidyr)
library(palmerpenguins)

# Prepare 2-dimensional data for clustering
df <- penguins %>%
  select(bill_length_mm, bill_depth_mm) %>%
  drop_na()

# Run K-means clustering using custom initial centroids
clustering <- kmeans(_____,
                     _____ = tibble(
                       bill_length_mm = c(40, 50, 60),
                       bill_depth_mm = c(15, 18, 21)
                     )
)
clustering

# Return cluster sizes
clustering$_____
```

Concept Map


Supervised learning
is different from
Unsupervised learning
is method of
Total intra-cluster variation
mini-mizes
K-Means
imple-ments
R function kmeans()
returns
Object of class "kmeans"
has
Components
print method
input of
produces
input of
input of
Data points
assigned to
Clusters
initiali-zation affects
Centroids
input of