Clustering: k-means

2024-09-02

- **M1 MIDS/MFA**
- **Université Paris Cité**
- Année 2024-2025
- Course Homepage

- Moodle

> **!** **Objectives**

Setup

```r
stopifnot(
  require(DT),
  require(skimr),
  require(GGally),
  require(patchwork),
  require(ggforce),
  require(glue),
  require(ggfortify),
  require(ggvoronoi),
  require(magrittr),
  require(broom),
  require(tidyverse)
)

tidymodels::tidymodels_prefer(quiet = TRUE)

old_theme <-theme_set(
  theme_minimal(base_size=9,
                base_family = "Helvetica")
)
```

```r
knitr::opts_chunk$set(
  message = FALSE,
```

```
  warning = FALSE,
  comment=NA,
  prompt=FALSE,
  cache=FALSE,
  echo=TRUE,
  results='asis'
)
```

```
gc <- options(ggplot2.discrete.colour="viridis")
gc <- options(ggplot2.discrete.fill="viridis")
gc <- options(ggplot2.continuous.fill="viridis")
gc <- options(ggplot2.continuous.colour="viridis")
```

Foreword

This lab is dedicated to the *k-means* clustering method. In words, *k-means* takes as input a collection of points in $\mathbb{R}^d$ (a numerical dataset) and a positive integer k. It returns a collection of k points (the *centers*) from $\mathbb{R}^d$. The centers define a Voronoï *tesselation/partition/diagran* of $\mathbb{R}^d$. The Voronoï *cells* define a clustering of the original dataset.

Voronoi tesselation/partition/diagram

Wikipedia on Voronoï diagrams

In the next chunk, we generate a Voronoï diagram on $\mathbb{R}^2$ with 100 cells defined from 100 random points drawn from the uniform distribution on a square. Function `stat_voronoi()` comes from ggvoronoi

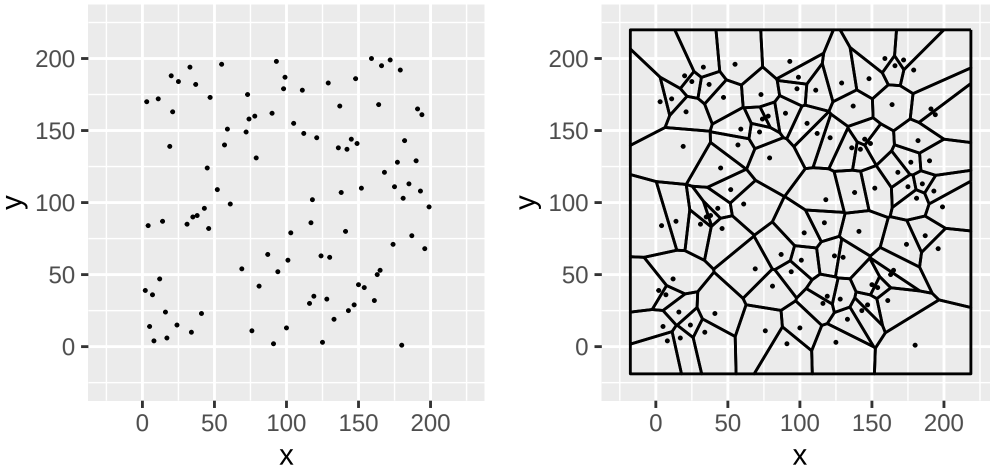
```
set.seed(45056)
x <- sample(1:200,100)
y <- sample(1:200,100)
points <- tibble(x,
                 y,
                 distance = sqrt((x-100)^2 + (y-100)^2))

p <- ggplot(points) +
    aes(x=x, y=y) +
    geom_point(size=.2) +
    coord_fixed() +
    xlim(c(-25, 225)) +
    ylim(c(-25, 225))

p + (p + stat_voronoi(geom="path")) +
  patchwork::plot_annotation(
    title="Voronoi tesselation",
    subtitle = "Left: 100 random points\nRight: Voronoï diagram")
```

Voronoi tesselation

Left: 100 random points
Right: Voronoï diagram



- Two adjacent Voronoï cells are separated by a (possibly semi-infinite) line segment
- Let the so-called *centers* be denoted by $c_1, \ldots, c_n$. They form the *codebook* $\mathcal{C}$.

- The Voronoï cell with *center* c_i is defined by

$$\left\{x : x \in \mathbb{R}^d, \qquad \|x - c_i\|_2 = \min_{j \leq n} \|x - c_j\|_2\right\}$$

- The center of a Voronoï cell is usually not its barycenter

k-means objective function

The k-means algorithm aims at building a *codebook* $\mathcal{C}$ that minimizes

$$\mathcal{C} \mapsto \sum_{i=1}^{n} \min_{c \in \mathcal{C}} \|X_i - c\|_2^2$$

over all codebooks with given cardinality
If $c \in \mathcal{C}$ is the closest centroid to $X \in \mathbb{R}^p$,

$$\|c - X\|^2$$

is the *quantization/reconstruction error* suffered when using codebook $\mathcal{C}$ to approximate X
⚠ If there are no restrictions on the dimension of the input space, on the number of centroids, or on sample size, computing an optimal codebook is a NP -hard problem

`kmeans()` is a wrapper for a collection of Algorithms that look like the Lloyd algorithm
Initialize by sampling from the data `k-Mean++` try to take them as separated as possible.
Iterate the two phases until ?
No guarantee to converge to a global optimum!
Proceeed by trial and error.
Repeat and keep the best result.

Iris data

Run `kmeans()` on the projection of the Iris dataset on the Sepal plane. We look for a partition into three cells.

```
data(iris)

kms <- iris %>%
  select(starts_with("Petal")) %>%
  kmeans(3)
```

The result is an object of class `kmeans`. The class is equiped with `broom` methods.

Summarizing a clustering

Question

Check the structure of objects of class `kmeans` and use `broom::tidy()` to get a summary. Compare with `summary()`

Visualizing a clustering

Question

Use `broom::augment()` and `broom::tidy()` to prepare two dataframes that will allow you to overlay a scatterplot of the dataset and a Voronoï diagram defined by the centers outpu by `kmeans()`.
Compare the result with `plot()`

Question

Redo the same operations but choose the `Sepal.xxx` dimension.
Design a function to avoid repetitive coding.

Playing with k

The number of cells/clusters may not be given a priori.

> **Question**
>
> Perform kmeans clustering with $k = 2$. Use `glance`, `tidy`, `augment` to discuss the result.

> **Question**
>
> Perform k-means for $k = 2, \ldots 10$, plot within sum of squares as function of k. Comment.

Lloyd's iterations

> **Initialize** Choose k centroids
>
> Iterations: Two phases
>
> **Movement** Assign each sample point to the closest centroid Assign each sample point to a class in the Voronoi partition defined by the centroids
>
> **Update** For each class in the current Voronoi partition, update teh centroid so as to minimize the Within Cluster Sum of Squared distances.

References

Vignette `k_means` from `tidyclust`