SpatialDataAnalysis_Cracow

Bartłomiej Piwowar

2025-04-28

```
options(repos = c(CRAN = "https://cloud.r-project.org/"))
install.packages(c("sf", "dbscan"))
```

```
## Instalowanie pakietów w 'C:/Users/barte/AppData/Local/R/win-library/4.3'
## (ponieważ 'lib' nie jest określony)

## pakiet 'sf' został pomyślnie rozpakowany oraz sumy MD5 zostały sprawdzone
## pakiet 'dbscan' został pomyślnie rozpakowany oraz sumy MD5 zostały sprawdzone
##
## Pobrane pakiety binarne są w
##  C:\Users\barte\AppData\Local\Temp\RtmpeCfQfK\downloaded_packages
```

```
library(sf) # sf: obsługa danych przestrzennych
```

```
## Warning: pakiet 'sf' został zbudowany w wersji R 4.3.3

## Linking to GEOS 3.11.2, GDAL 3.8.2, PROJ 9.3.1; sf_use_s2() is TRUE
```

```
library(dbscan) # dbscan: algorytmy klasteryzacji
```

```
## Warning: pakiet 'dbscan' został zbudowany w wersji R 4.3.3

##
## Dołączanie pakietu: 'dbscan'

## Następujący obiekt został zakryty z 'package:stats':
##
##     as.dendrogram
```

```
# 1. Wczytanie danych przestrzennych
# Wczytanie granic osiedli w Krakowie (dane poligonowe) oraz punktów wykroczeń
osiedla_shp <- st_read("C:/Users/barte/Desktop/STUDIA/5 SEMESTR/ANALIZA DANYCH PRZESTRZENNYCH/Ćwiczenia/
```

```
## Reading layer 'osiedla' from data source
##   'C:\Users\barte\Desktop\STUDIA\5 SEMESTR\ANALIZA DANYCH PRZESTRZENNYCH\Ćwiczenia\5_Projekt\files\os
##   using driver 'ESRI Shapefile'
## Simple feature collection with 141 features and 30 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: 7413437 ymin: 5537344 xmax: 7443955 ymax: 5555031
## Projected CRS: ETRS89 / Poland CS2000 zone 7
```

```r
zestaw_shp <- st_read("C:/Users/barte/Desktop/STUDIA/5 SEMESTR/ANALIZA DANYCH PRZESTRZENNYCH/Ćwiczenia/5
```

```
## Reading layer 'zestaw5_XYTableToPoi_Project' from data source
##   'C:\Users\barte\Desktop\STUDIA\5 SEMESTR\ANALIZA DANYCH PRZESTRZENNYCH\Ćwiczenia\5_Projekt\files\z
##   using driver 'ESRI Shapefile'
## Simple feature collection with 2000 features and 4 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 7415455 ymin: 5538340 xmax: 7441815 ymax: 5553322
## Projected CRS: ETRS_1989_Poland_CS2000_Zone_7
```

```r
# 2. Dopasowanie układu współrzędnych
# Transformacja CRS (układu współrzędnych) na wspólny dla obu zbiorów danych
# Umożliwia dokładne nakładanie granic osiedli i punktów
osiedla_shp <- st_transform(osiedla_shp, crs = st_crs(zestaw_shp))
```

```r
# 3. Przygotowanie danych do klasteryzacji
# Ekstrakcja współrzędnych punktów w formie macierzy (niezbędne dla algorytmu DBSCAN)
punkty_coords <- st_coordinates(zestaw_shp)
```

```r
# 4. Wykonanie klasteryzacji DBSCAN
# DBSCAN - Density-Based Spatial Clustering of Applications with Noise
# HDBSCAN - Hierarchical Density-Based Spatial Clustering of Applications with Noise
# Parametry:
# eps - promień sąsiedztwa (tutaj 200 jednostek CRS, np. metrów)
# minPts - minimalna liczba punktów w promieniu eps, aby utworzyć klaster
db1 <- dbscan(punkty_coords, eps = 500, minPts = 10) # (promień sąsiedztwa jest duży) Wynikiem jest wię
db2 <- dbscan(punkty_coords, eps = 200, minPts = 5)
db3 <- dbscan(punkty_coords, eps = 10, minPts = 1) # niskie argumenty powodują, że niemal każdy punkt l

hdb1 <- hdbscan(punkty_coords, minPts = 100) # Przy dużym minPts powstają duże klastry
hdb2 <- hdbscan(punkty_coords, minPts = 30)
hdb3 <- hdbscan(punkty_coords, minPts = 5)# Przy małym minPts małe obszary są brane jako osobne klastry
```

```r
# 5. Przypisanie wyników klasteryzacji do danych przestrzennych
# Wartości w klastrze są przypisywane jako kolumna "cluster" w zestawie punktów
zestaw_shp$cluster1 <- as.factor(db1$cluster)
zestaw_shp$cluster2 <- as.factor(db2$cluster)
zestaw_shp$cluster3 <- as.factor(db3$cluster)

zestaw_shp$cluster4 <- as.factor(hdb1$cluster)
zestaw_shp$cluster5 <- as.factor(hdb2$cluster)
zestaw_shp$cluster6 <- as.factor(hdb3$cluster)
```

```r
# 6. Wizualizacja wyników
# Przygotowanie kolorów
colors1 <- db1$cluster
colors2 <- db2$cluster
colors3 <- db3$cluster

colors4 <- hdb1$cluster
colors5 <- hdb2$cluster
colors6 <- hdb3$cluster
```

```
################### WYNIKI KLASTERYZACJI ######################
```

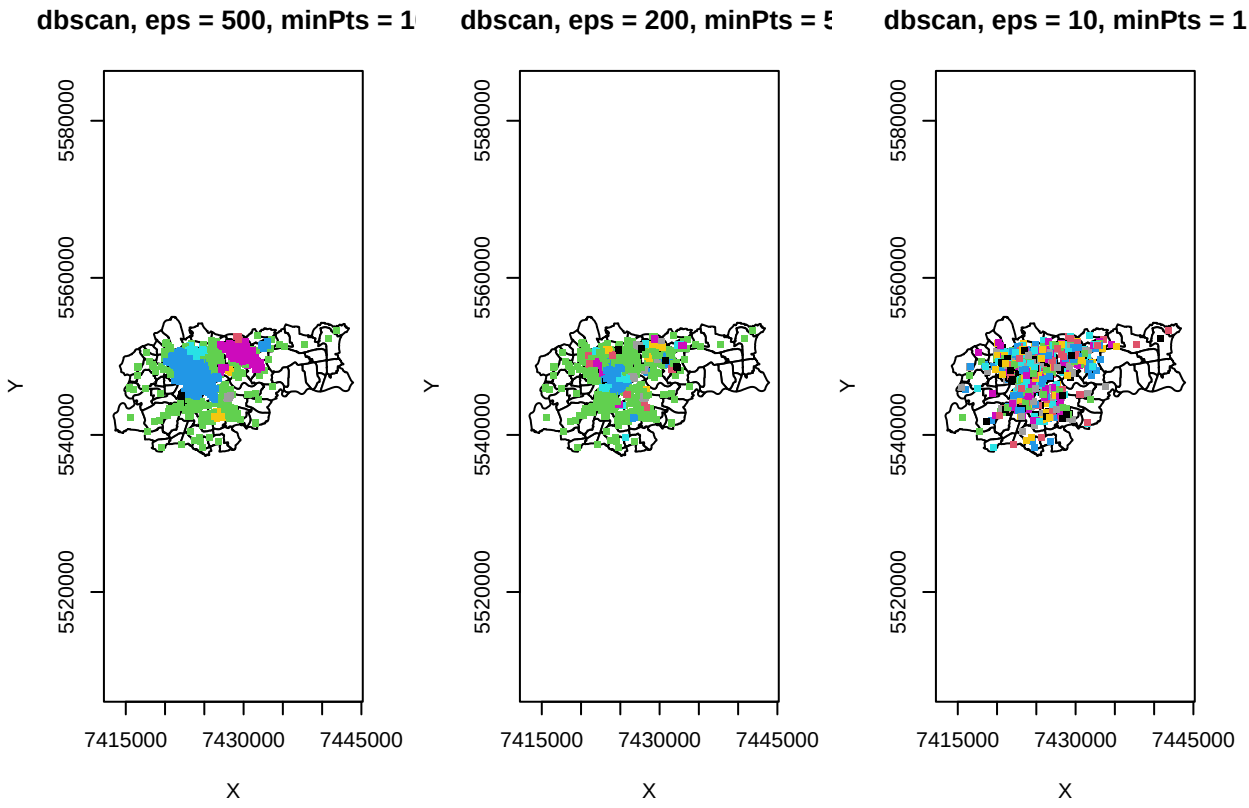
```
par(mfrow = c(1, 3))
# Rysowanie mapy granic osiedli
plot(osiedla_shp$geometry,
     main = "dbscan, eps = 500, minPts = 10",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors1 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów



plot(osiedla_shp$geometry,
     main = "dbscan, eps = 200, minPts = 5",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors2 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów



plot(osiedla_shp$geometry,
     main = "dbscan, eps = 10, minPts = 1",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors3 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów
```

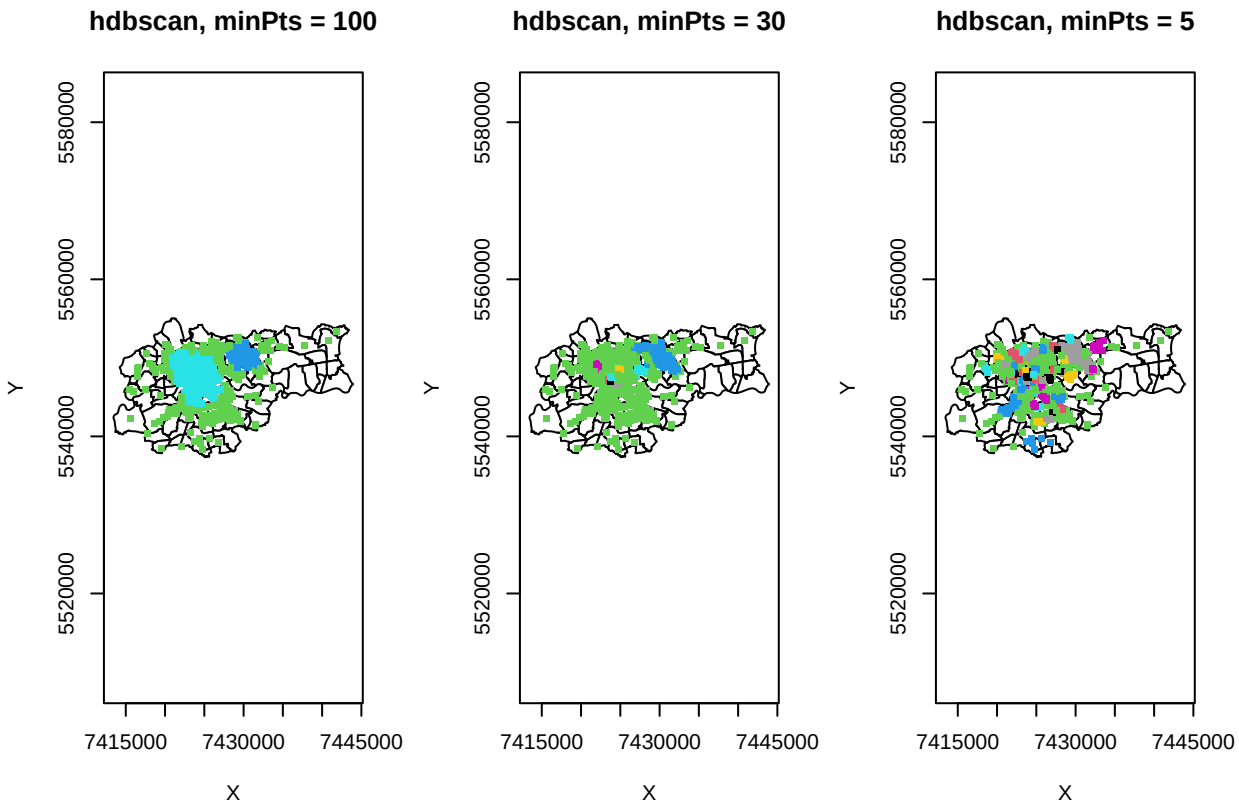



```
par(mfrow = c(1, 3))
plot(osiedla_shp$geometry,
     main = "hdbscan, minPts = 100",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors4 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów



plot(osiedla_shp$geometry,
     main = "hdbscan, minPts = 30",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors5 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów
```

```r
plot(osiedla_shp$geometry,
     main = "hdbscan, minPts = 5",
     axes = TRUE,
     xlab = "X",
     ylab = "Y",
     col = "white",
     border = "black")
points(punkty_coords,
       col = colors6 + 3,  # Kolory klastrów
       pch = 15,
       cex = 0.7)  # Styl punktów
```




```r
################ WNIOSKI ###################

# DBSCAN to algorytm klasteryzacji oparty na gęstości, który wymaga ustawienia
# stałego promienia sąsiedztwa (eps) i minimalnej liczby punktów (minPts),
# co czyni go skutecznym dla danych o jednolitej gęstości, ale trudnym w
# przypadku zróżnicowanych danych. HDBSCAN jest bardziej elastyczny, ponieważ
# automatycznie dostosowuje gęstość w różnych częściach danych, używając jedynie
# parametru minPts, co pozwala lepiej identyfikować klastry w danych
# nieregularnych. Dodatkowo HDBSCAN tworzy hierarchiczne klastry i lepiej
# identyfikuje szum, co sprawia, że jest bardziej wszechstronny od DBSCAN.
```