Projekt - klasteryzacja przestrzenna danych punktowych

Katarzyna Tokarczuk

2024-11-19

Wczytanie plików oraz załadowanie potrzebnych bibliotek

```
library(sf)
library(ggplot2)
library(dbscan)
library(dplyr)
library(RColorBrewer)
punkty <- st_read("C:/Users/kasia/Studies/SEMESTR 5/Analiza danych przestrzennych/projekt/zestaw8_XYTabl
osiedla <- st_read("C:/Users/kasia/Studies/SEMESTR 5/Analiza danych przestrzennych/projekt/osiedla.shp")
```

1.Metoda DBSCAN

Algorytm DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Zasada działania:
Jest to algorytm grupowania danych (klasteryzacji) oparty na analizie wzajemnego położenia i wzajemnych odległości punktów, czyli gęstości przestrzennej punktów.

Kroki tego algorytmu:
1. Obliczenie sąsiedztwa - dla każdego punktu w danych określa się jego sąsiedztwo jako punkty znajdujące się w promieniu eps od danego punktu.
2. Identyfikacja punktów rdzeniowych - punkt jest uznawany za punkt rdzeniowy, jeśli liczba punktów w jego sąsiedztwie jest większa lub równa wartości minPts.
3. Łączenie punktów rdzeniowych w klastry - punkty rdzeniowe, które są dostępne gęstościowo, zostają połączone w klastry.
4. Przypisanie punktów brzegowych - punkty brzegowe, które nie są punktami rdzeniowymi, ale znajdują się w sąsiedztwie punktów rdzeniowych, zostają przypisane do klastrów utworzonych przez te punkty rdzeniowe.
5. Przypisanie punktów szumu - punkty, które nie są ani rdzeniowe, ani brzegowe, są oznaczane jako szum (klaster 0).

Wady:
- nie może dobrze klasteryzować zbiorów danych o dużych różnicach gęstości (nie da się wówczas odpowiednio dla wszystkich klastrów dobrać kombinacji minPts -)
- jakość tego algorytmu zależy od miary odległości
- nie jest całkowicie deterministyczny

Zalety:
- nie wymaga określenia z góry liczby klastrów w danych
- wydziela szum i jest odporny na wartości odstające
- wymaga tylko dwóch parametrów i jest w większości przypadków niewrażliwy na kolejność punktów w bazie danych
- może znaleźć dowolnie ukształtowane klastry

Używam funkcji dbscan oraz ustawiam różne wartości parametrów dla tej metody:
eps - promień otoczenia, w którym sprawdzamy ilość sąsiadów
minPts - minimalna ilość punktów, które muszą tworzyć klaster

```
#Wyodrębnienie współrzędnych z pliku z punktami
coords <- st_coordinates(punkty)
head(coords)
```

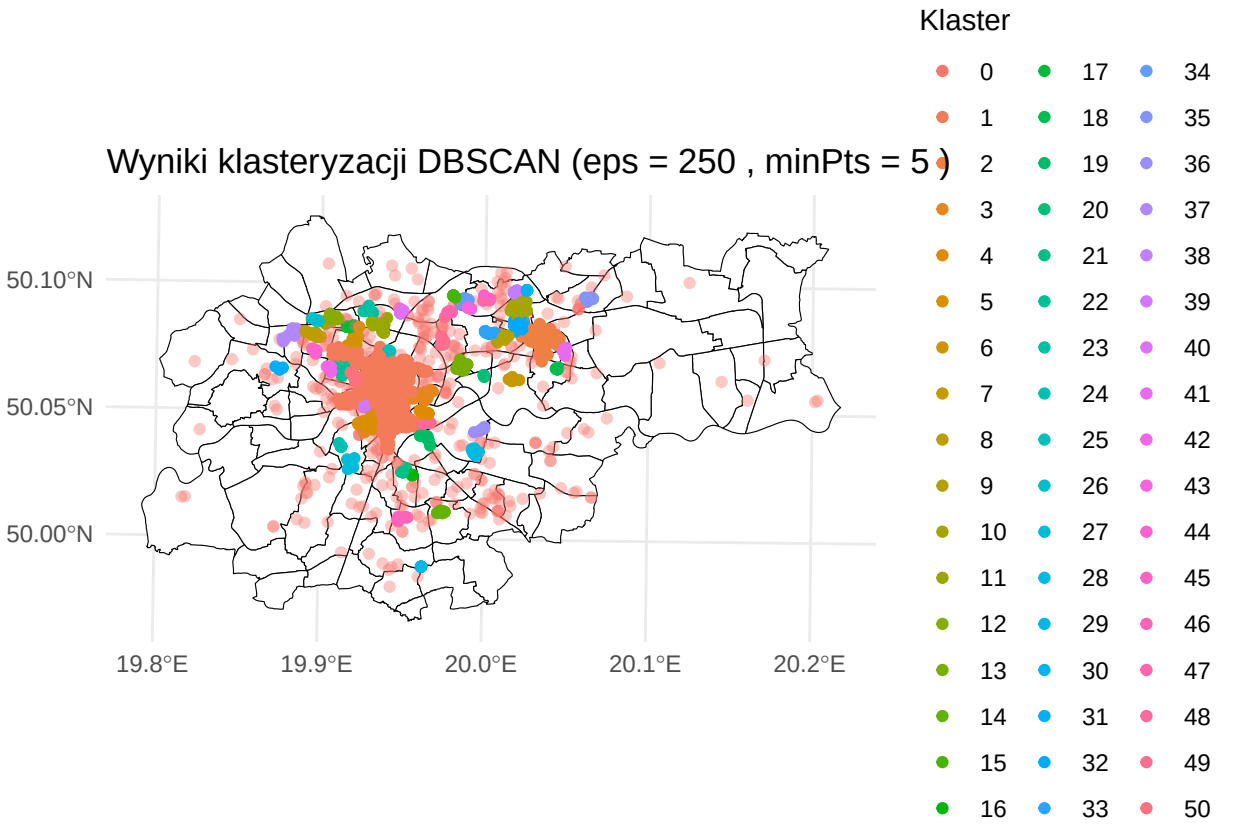
```
            X       Y
  [1,] 7423881 5547947
  [2,] 7430823 5549041
  [3,] 7425816 5547329
  [4,] 7423685 5548121
  [5,] 7423867 5547753
  [6,] 7429003 5551706
```

```
#Ilość punktów
coords_count <- nrow(coords)
print(coords_count)
```

```
  [1] 2000
```

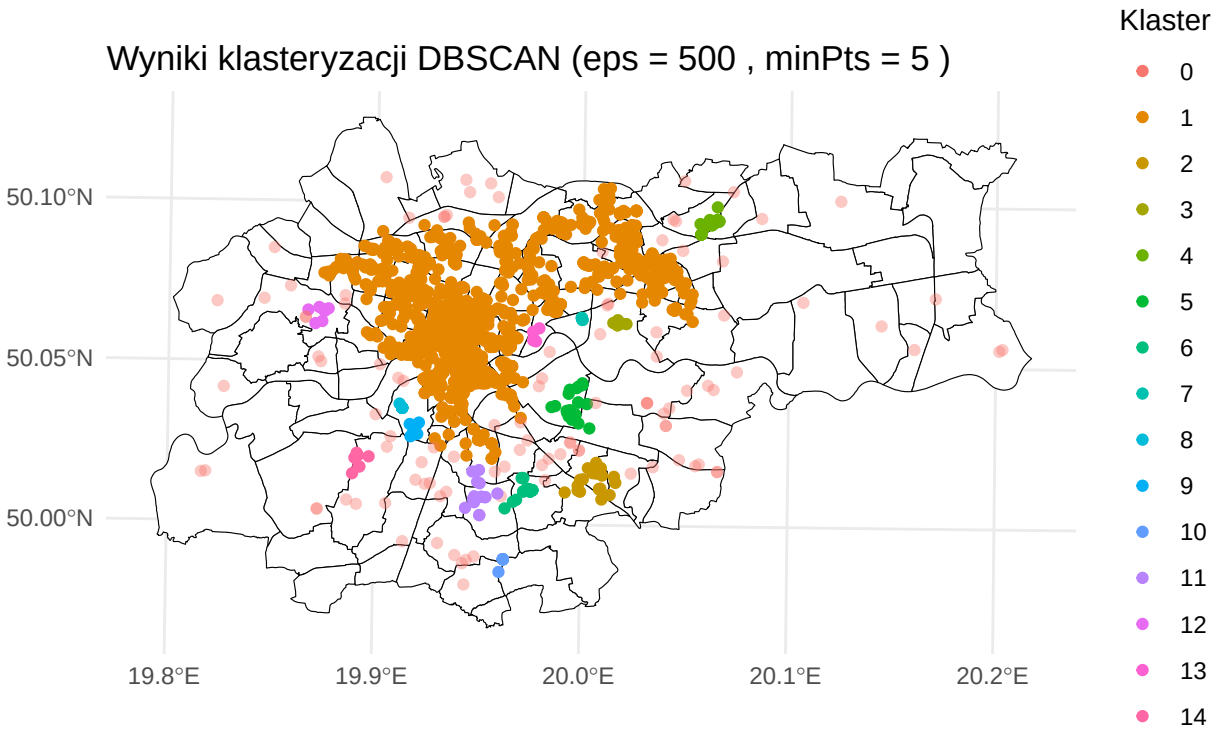
```
#Funkcja pomocnicza, która przeprowadza klasteryzację metodą DBSCAN (przypisuje wyniki do danych i je w
#Punkty odstające (szum) na mapie są wyświetlone z ustawioną półprzezroczystością

klasteryzacja_dbscan <- function(dane_punkty, dane_osiedla, eps, minPts) {

  coords <- st_coordinates(dane_punkty)  #Wyodrębnienie współrzędnych

  db <- dbscan(coords, eps = eps, minPts = minPts)    #Klasteryzacja DBSCAN
  dane_punkty$cluster <- db$cluster  #Przypisanie wyników do danych

  #Wizualizacja
  ggplot() +
    geom_sf(data = dane_osiedla, fill = "white", color = "black") +
    geom_sf(data = dane_punkty, aes(color = as.factor(cluster), alpha = ifelse(cluster == 0, 0.4, 1))) +
    theme_minimal() +
    labs(color = "Klaster", alpha = "Przezroczystość") +
    ggtitle(paste("Wyniki klasteryzacji DBSCAN (eps =", eps, ", minPts =", minPts, ")")) +
    scale_alpha_identity()
}
```

```
# Klasteryzacja DBSCAN (eps=250 oraz minPts=5)
klasteryzacja_dbscan(punkty, osiedla, eps = 250, minPts = 5)
```



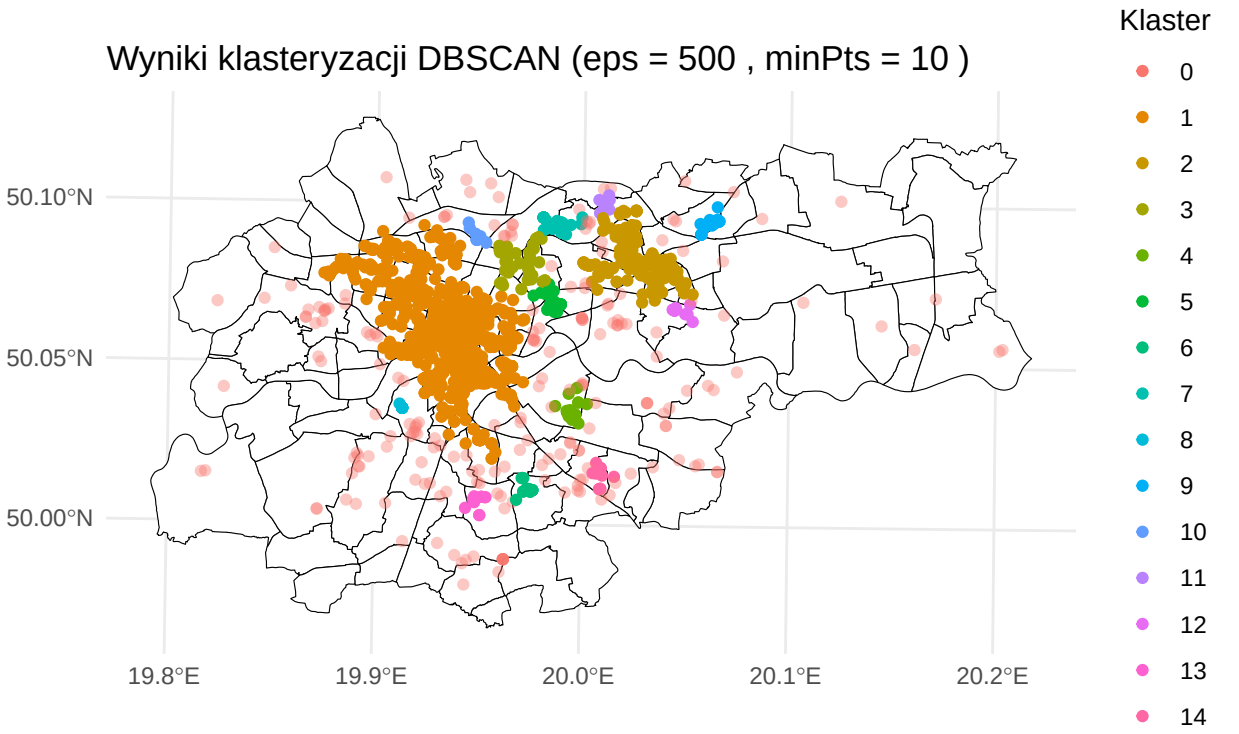
Wartość minPts=5 przy promieniu otoczenia równym 250 sprawiła, iż duża liczba punktów została przypisana do wielu małych klastrów.

```
# Klasteryzacja DBSCAN (eps=500 oraz minPts=5)
klasteryzacja_dbscan(punkty, osiedla, eps = 500, minPts = 5)
```



W tym przypadku parametr eps został zwiększony do 500, przy czym minPts pozostało równe 5. Powstało 14 klastrów.

```
# Klasteryzacja DBSCAN (eps=500 oraz minPts=10)
klasteryzacja_dbscan(punkty, osiedla, eps = 500, minPts = 10)
```



W kolejnym przypadku eps pozostało równe 500, natomiast zwiększona została minimalna ilość punktów tworząca klaster. Można z tych danych odczytać więcej informacji o zwiększonej intensywności wykroczeń.

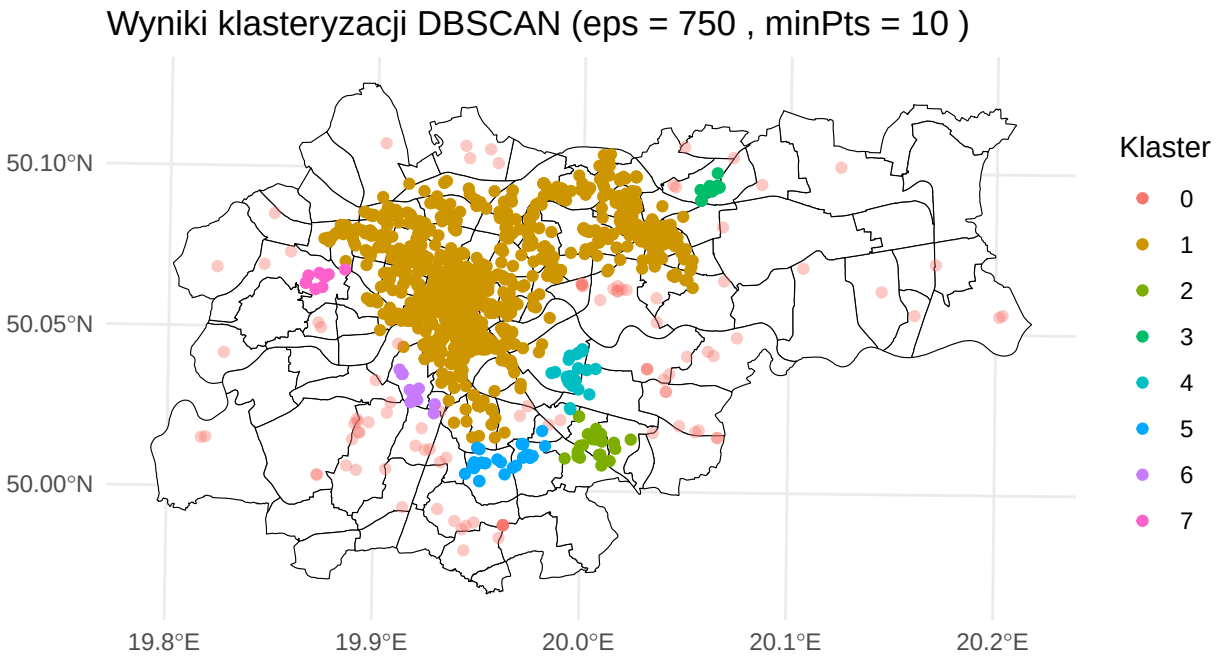
```
# Funkcja pomocnicza do wypisywania miejsc, w których wystąpiło najwięcej wykroczeń
klastry_result <- function(dane_punkty, dane_osiedla, eps, minPts) {

  coords <- st_coordinates(dane_punkty)  #Wyodrębnienie współrzędnych

  db <- dbscan(coords, eps = eps, minPts = minPts)  #Klasteryzacja DBSCAN
  punkty$cluster <- db$cluster  #Przypisanie wyników do danych

  punkty_bez_szumu <- punkty %>% filter(cluster != 0)  #Pominięcie szumu

  #Połączenie punktów z osiedlami
  punkty_z_osiedlami <- st_join(punkty_bez_szumu, osiedla, join = st_within)

  #Podsumowanie klastrów na osiedlach
  result <- punkty_z_osiedlami %>%
    group_by(cluster, NAZWA_JEDN) %>%
    summarise(liczba_wykroczen = n(), .groups = "drop") %>%
    st_drop_geometry() %>%
    arrange(desc(liczba_wykroczen))

  head(result)

}
```

```
klastry_result(punkty, osiedla, eps = 500, minPts = 10)
```

```
# A tibble: 6 x 3
  cluster NAZWA_JEDN          liczba_wykroczen
    <int> <chr>                          <int>
1       1 "Stare Miasto"                   404
2       1 "Kazimierz"                      169
3       1 "Warszawskie"                     83
4       1 "Nowy \x8cwiat"                   64
5       1 "Stare Podg\xf3rze"               60
6       1 "Kleparz"                         59
```

Powyżej znajduje sie wynik funkcji pomocniczej. Występowanie klastrów świadczy o tym, że obszary o zwiększonej intensywności zarejestrowanych wykroczeń to: Stare Miasto, Kazimierz, Warszawskie, Nowy świat, Stare Podgórze, Kleparz.

```
# Klasteryzacja DBSCAN (eps=1000 oraz minPts=10)
klasteryzacja_dbscan(punkty, osiedla, eps = 750, minPts = 10)
```



Gdy minPts pozostało równe 10, a promień eps został zwiększony do 750, zmniejszyła się liczba klastrów.

2.Metoda HDBSCAN

Algorytm HDBSCAN (Hierarchical density-based spatial clustering of applications with noise)

Zasada działania
Algorytm hierarchicznej klasteryzacji przestrzennej w obecności szumu opartej na gęstości zjawisk. Algorytm ten identyfikuje tzw.rdzenie klastrów jako te punkty, w sąsiedztwie których zlokalizowana jest wystarczająca ilość punktów sąsiadujących (większa bądź równa przyjętej wartości granicznej). Do tak zdefiniowanych rdzeni klastrów dołączane są punkty pozostające w bliskiej odległości od któregokolwiek z punktów rdzenia (bliżej niż zadana wartość), które w sumie tworzą klaster.

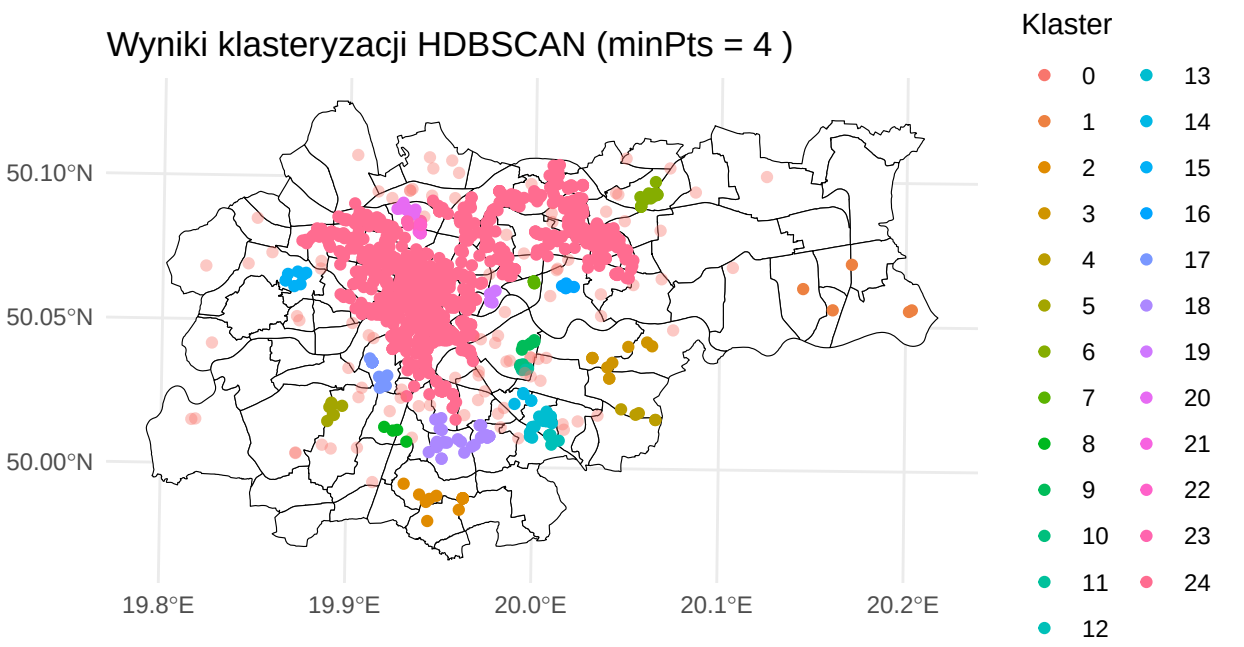
Kroki tego algorytmu:
1. Obliczenie odległości wzajemnej osiągalności.
2. Tworzenie drzewa minimalnego rozpinania.
3. Przycinanie drzewa na podstawie stabilności.
4. Ekstrakcja klastrów.

Wady:
- wrażliwy na rzadkie dane
- jest wolniejszy niż DBSCAN, szczególnie dla dużych zbiorów danych

Zalety:
- HDBSCAN radzi sobie z klastrami o zróżnicowanej gęstości
- automatycznie dobiera parametry (nie trzeba ustawiać eps)
- wybiera stabilne klastry, co prowadzi do bardziej sensownych wyników w przypadku danych złożonych

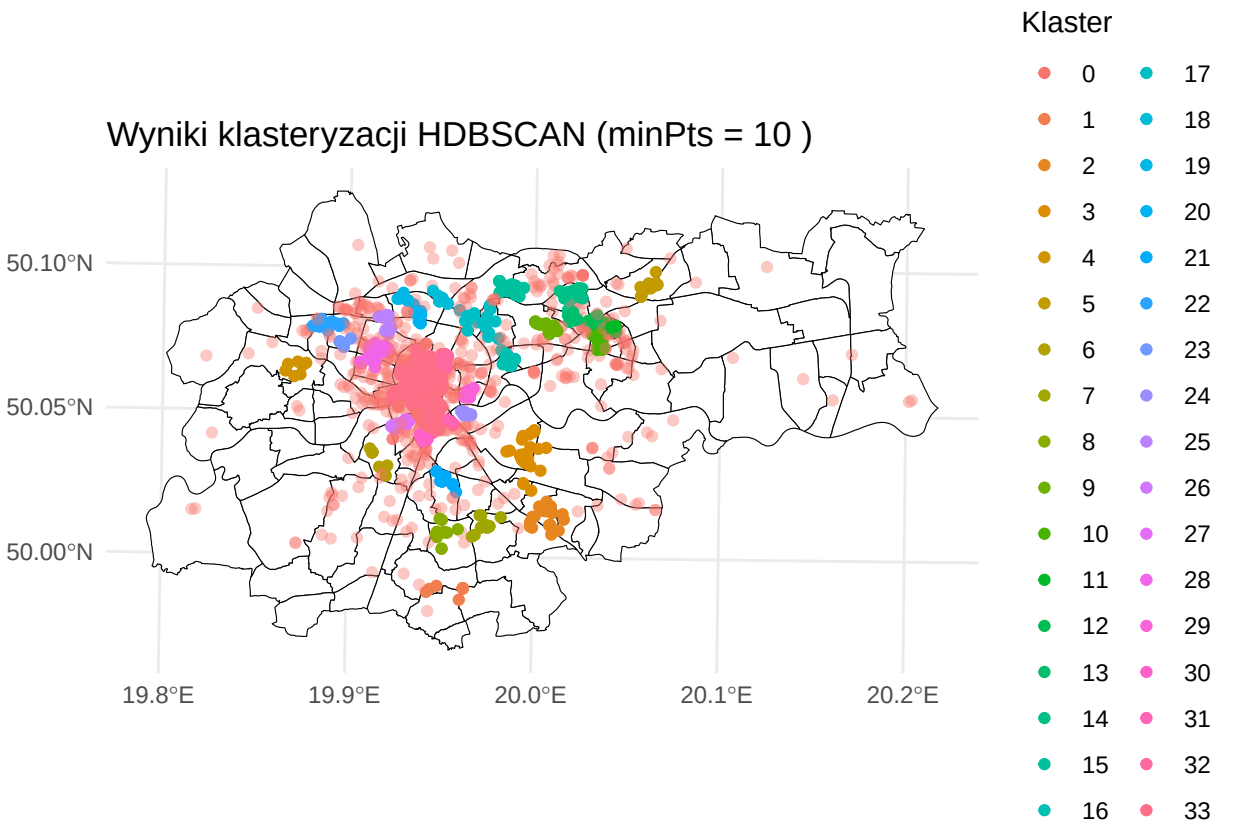
```
#Funkcja pomocnicza, która przeprowadza klasteryzację metodą HDBSCAN (przypisuje wyniki do danych i je
#Punkty odstające (szum) na mapie są wyświetlone z ustawioną półprzezroczystością

klasteryzacja_hdbscan <- function(dane_punkty, dane_osiedla, eps, minPts) {

  coords <- st_coordinates(dane_punkty)  #Wyodrębnienie współrzędnych

  db <- hdbscan(coords, minPts = minPts)    #Klasteryzacja HDBSCAN
  dane_punkty$cluster <- db$cluster  #Przypisanie wyników do danych

  # Wizualizacja
  ggplot() +
    geom_sf(data = dane_osiedla, fill = "white", color = "black") +
    geom_sf(data = dane_punkty, aes(color = as.factor(cluster), alpha = ifelse(cluster == 0, 0.4, 1))) +
    theme_minimal() +
    labs(color = "Klaster", alpha = "Przezroczystość") +
    ggtitle(paste("Wyniki klasteryzacji HDBSCAN (minPts =", minPts, ")")) +
    scale_alpha_identity()
}
```

```
# Klasteryzacja HDBSCAN (minPts=4)
klasteryzacja_hdbscan(punkty, osiedla, minPts = 4)
```



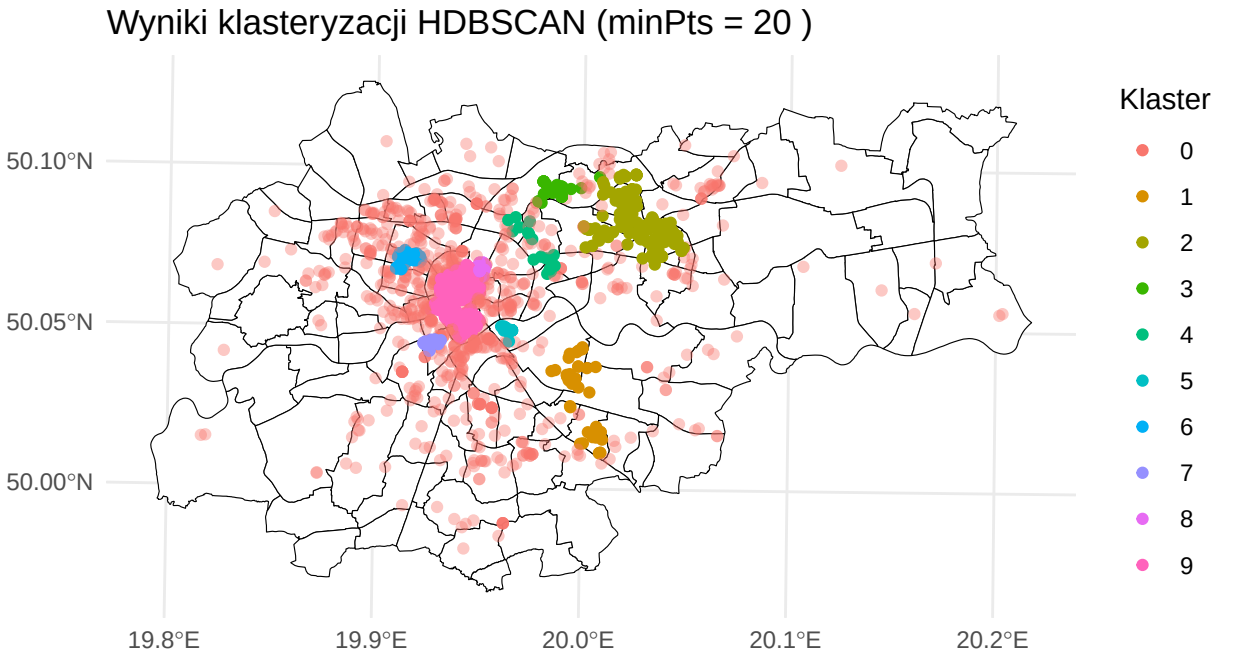
W tym przykładzie powstały 24 klastry. Widać, iż duża część punktów została zakwalifikowana do jednego klastra.

```
# Klasteryzacja HDBSCAN (minPts=10)
klasteryzacja_hdbscan(punkty, osiedla, minPts = 10)
```



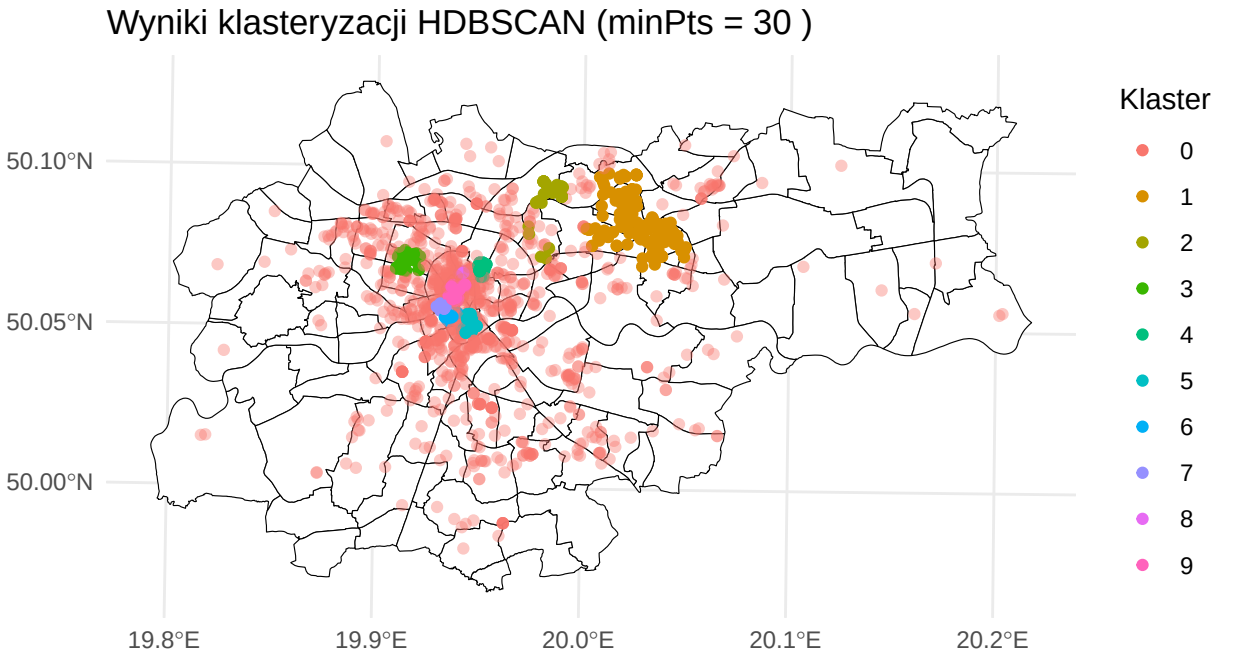
W kolejnym przypadku powstało więcej klastrów, a rozkład danych na mapie wskazuje, że klastry są bardziej "zorganizowane" (łatwiej odczytać, gdzie są największe skupiska).

```
# Klasteryzacja HDBSCAN (minPts=20)
klasteryzacja_hdbscan(punkty, osiedla, minPts = 20)
```



W tym przypadku utworzyło się 9 klastrów. Klasteryzacja jest bardziej restrykcyjna. Widać tylko te najbardziej gęste skupiska punktów, podczas gdy mniejsze nie zostały uzwględnione.

```
# Klasteryzacja HDBSCAN (minPts=30)
klasteryzacja_hdbscan(punkty, osiedla, minPts = 30)
```



W ostatnim przykładzie zdecydowana większość punktów została potraktowana jako szum, ze względu na dużą wartość minPts.

Porównanie wyników i wnioski

Dzielnice Krakowa, w których wyznaczono klastry: Największa intensywność zarejestrowanych wykroczeń pojawiła się na Starym Mieście, Kleparzu, Nowym Świecie, Stradomiu, Piasku Północy i Południu. Ponadto zwiększoną intensywność zarejestrowano również w okolicach Bieńczyc Nowych, Osiedla Albertyńskiego, Prokocimia, Nowej Wsi Południe, Rakowic, Grzegórzki.

W projekcie zostały użyte dwie metody klasteryzacji danych punktowych DBSCAN oraz HDBSCAN. DBSCAN pozwolił na identyfikację obszarów o dużej ilości punktów klasteryzacji (takich jak Stare Miasto i Kazimierz). Przy większym eps (np. 500), pojawiło się mniej klastrów o większej liczebności. Natomiast HDBSCAN przy niskich wartościach minPts (np. minPts = 4), identyfikował wiele małych klastrów, w tym mniej gęste obszary. Wyższe wartości minPts (np. minPts = 20 lub 30) powodowały bardziej restrykcyjną selekcję klastrów, eliminując mniej intensywne skupiska danych. Ta metoda była bardziej skuteczna w znajdowaniu stabilnych klastrów w obszarach o zróżnicowanej gęstości.

Zarówno DBSCAN, jak i HDBSCAN zidentyfikowały kluczowe obszary, gdzie pojawiała sie największa liczba wykroczeń. Natomiast HDBSCAN był bardziej elastyczną metodą (nie wymagał również ustalenia wartości eps, która w metodzie DBSCAN czasami była trudna do ustalenia z odpowiednią wartością minPts) oraz dobrze poradził sobie z identyfikacją klastrów w tym układzie przestrzennym.