Untitled

Jakub Slazyk

25 11 2021

Zaladowanie wykorzystanych bibliotek

```
library("sp")
library("rgdal")
library("ggplot2")
library("dbscan")
```

```
## Warning: pakiet 'dbscan' został zbudowany w wersji R 4.1.2
```

```
library("repr")
```

```
## Warning: pakiet 'repr' został zbudowany w wersji R 4.1.2
```

```
library("broom")
library("opticskxi")
```

```
## Warning: pakiet 'opticskxi' został zbudowany w wersji R 4.1.2
```

Wczytanie danych

```
getwd()
```

```
## [1] "C:/Users/jakub/OneDrive/Dokumenty"
```

```
data_conv<-read.csv2("zestaw9.csv",sep=";")
head(data_conv)
```

```
##       Long      Lat
## 1 19.95673 50.02497
## 2 19.92715 50.06300
## 3 19.80796 50.01776
## 4 19.93616 50.06218
## 5 19.95938 50.04922
## 6 19.93493 50.05555
```

Zmiana ukladu wspolrzednych danych

```
cord.dec = SpatialPoints(cbind(data_conv$Long, data_conv$Lat), proj4string=CRS("+proj=longlat"))
cord.UTM <- spTransform(cord.dec, CRS("+init=epsg:2178"))
```

```
## Warning in showSRID(uprojargs, format = "PROJ", multiline = "NO", prefer_proj
## = prefer_proj): Discarded datum European_Terrestrial_Reference_System_1989 in
## Proj4 definition
```

Podglad danych

```
head(cord.UTM)
```

```
## SpatialPoints:
##      coords.x1 coords.x2
## [1,]   7425247   5543719
## [2,]   7423188   5547979
## [3,]   7414575   5543077
## [4,]   7423832   5547878
## [5,]   7425474   5546414
## [6,]   7423734   5547143
## Coordinate Reference System (CRS) arguments: +proj=tmerc +lat_0=0
## +lon_0=21 +k=0.999923 +x_0=7500000 +y_0=0 +ellps=GRS80 +units=m
## +no_defs
```

Wczytanie tla(mapy z podzialem na osedla/dzielnice Krakowa)

```
x<-getwd()
my_spdf <- readOGR(dsn=path.expand('C:/Users/jakub/OneDrive/Dokumenty/osiedla'), layer='osiedla' )
```

```
## Warning in OGRSpatialRef(dsn, layer, morphFromESRI = morphFromESRI, dumpSRS = dumpSRS, : Discarded da
##  but +towgs84= values preserved
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\jakub\OneDrive\Dokumenty\osiedla", layer: "osiedla"
## with 141 features
## It has 30 fields
## Integer64 fields read as strings:  ID IL_K_MOBIL IL_K_NIEMO IL_K_POPRO IL_K_PROD IL_K_PRZED IL_L_MOBI
```

Podglad mapy z podzialem na osedla/dzielnice Krakowa

```
par(mar=c(0,0,0,0))
plot(my_spdf, col="#f2f2f2", bg="skyblue", lwd=0.25, border=0 )
```

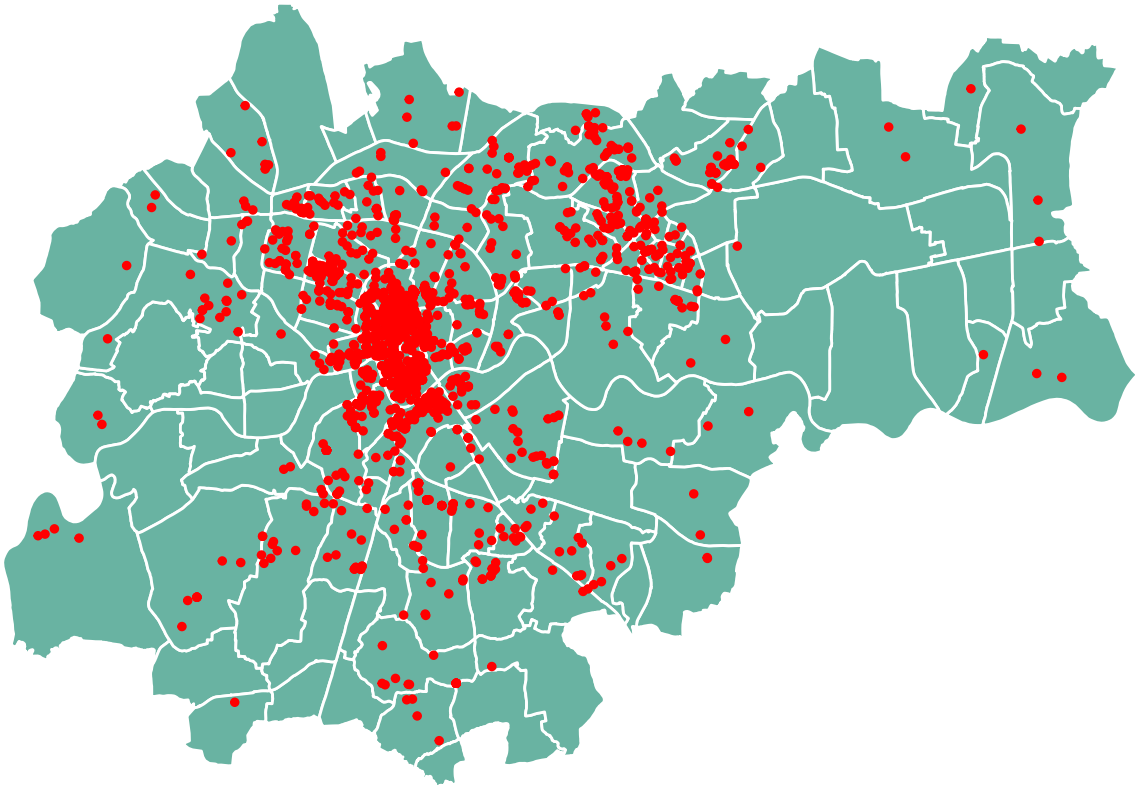
Przygotowanie obu warstw do wygodnego korzystania w trakcie tworzenia plotow

```
spdf_fortified <- tidy(my_spdf)
cords <- as.data.frame(cord.UTM)
colnames(cords)<-c("X","Y")
head(cords)
```

```
##         X       Y
## 1 7425247 5543719
## 2 7423188 5547979
## 3 7414575 5543077
## 4 7423832 5547878
## 5 7425474 5546414
## 6 7423734 5547143
```

Podglad na zgloszone wykroczenia na tle mapy osiedli/dzielnic

```
ggplot() +
  geom_polygon(data =my_spdf, aes( x = long, y = lat, group = group), fill="#69b3a2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y),color="red",
  size = 1)+
  theme_void()
```



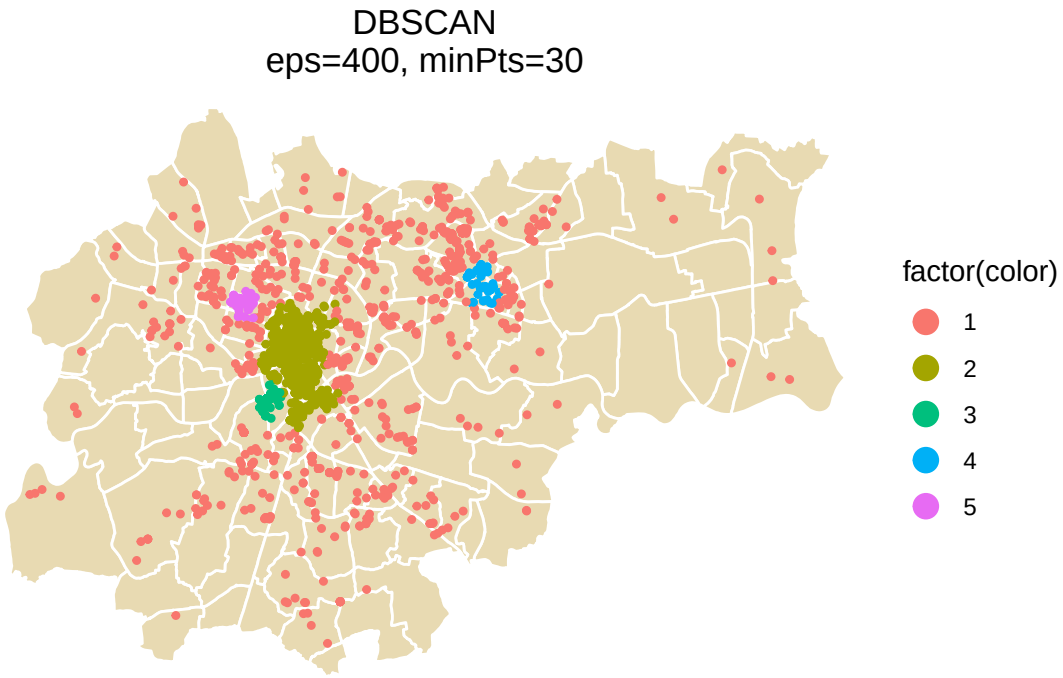
Na wstepie warto zauwazyc, ze we wszystkich wizualizacjach punkty nalezace do pierwszego klastra to szum i w rzeczywistosci punkty te nie naleza do zadnego z klastrow.

Algorytm DBSCAN: Parametry wejsciowe: eps (epsilon) - promien sasiedztwa, maksymalna dopuszczalna odleglosc pomiedzy dwoma punktami, aby rozwazac czy naleza do tego sameka klastra minPts - minimalna wymagana liczba punktow znajdujaca sie w odpowiedniej odleglosci, aby uznac je za klaster Schemat dzialania: Dla wszystkich obserwacji znajdujemy sasiadow, znajdujacych sie w odleglosci nie wiekszej, niz epsilon. Jesli ich liczba wynosi przynajmiej wartosc minPts, punkt zaliczany jest do punktow centralnych (Core Points). Wszystkie punkty, spelniajace to zalozenie i lezace w swoim sasiedztwie tworza klaster, nastepnie wszystkie punkty, lezace w odleglosci nie wiekszej niz epsilon od punktow, tworzacych czlon klastru, sa dolaczane jako punkty graniczne klastra (Border Points) Zalety: -jego dzialanie jest proste w zrozumieniu -jest odporny na wplyw obserwacji odstajacych, nienalezacych do zadnego klastra -daje dobre rezultaty, przy relatywnie szybkim czasie dzialania -mozliwosc modelowania wyniku za pomoca parametrow wejsciowych - mamy spory wplyw na finalny rezultat Wady: -nie daje nam mozliwosci zdefiniowania oczekiwanej liczby klastrow - ich liczba wynika z wartosci parametrow wejsciowych -mozliwosc manewru argumentami wejsciowymi moze niesc za soba problem, ich dobor moze byc trudny w sytuacji, gdy nie wiemy jakiego rezultatu sie spodziewamy/oczekujemy

Klasteryzacja 1 z wykorzystaniem algorytmu DBSCAN

```
set.seed(123456789)
dbscan_res <- dbscan(cords, eps = 400, minPts = 30)
color<-dbscan_res$cluster+1
plot1<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
```

```
  ggtitle("DBSCAN\neps=400, minPts=30") +
  theme(plot.title = element_text(hjust = 0.5))+
  guides(colour = guide_legend(override.aes = list(size=4)))
plot1
```



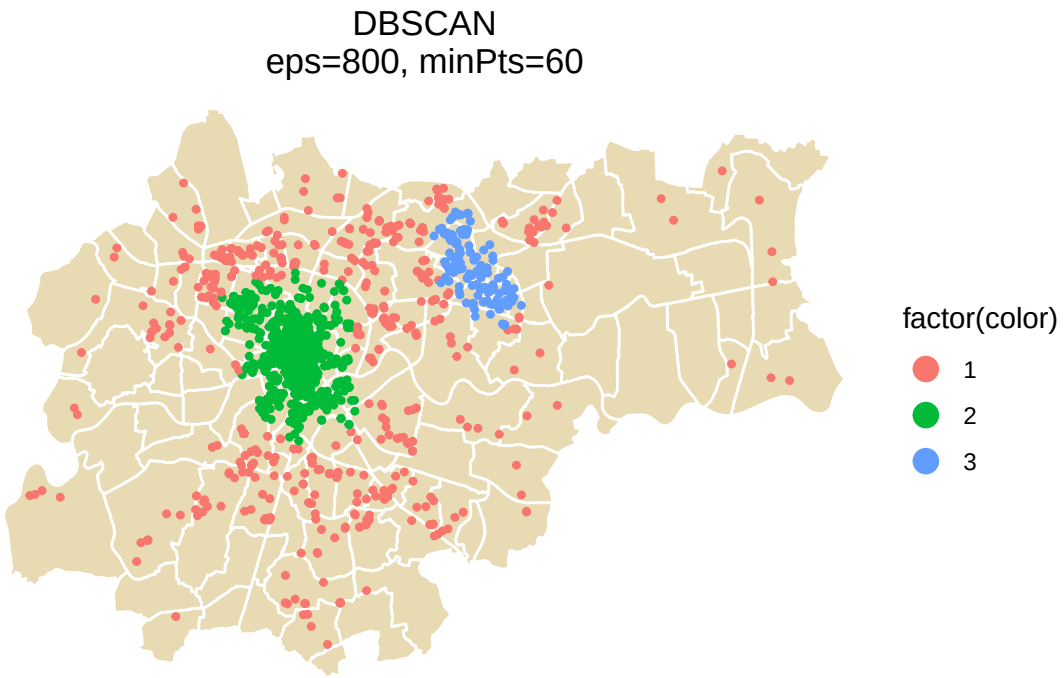
Patrzac na mape mozna nabrac watpliwosci czy w kazdym klastrze liczba minimalna faktycznie jest spelniona (glownie chodzi o klaster numer 3), dlatego sprawdzam liczbe punktow w kazdym klastrze aby sprawdzic czy warunek jest spelnony. Okazuje sie faktycznie byc spelniony, co oznacza, ze punkty bardzo mocno musza na siebie nachodzic. Zdecydowanie najwiekszy klaster (numer 2) znaljduje sie na calym obszarze Starego Miasta, nalezy do niego nieco ponad polowa obserwacji. Pozostale 3 klastry (numer 3-5) sa zdecydowanie mniejsze i zlokalizowane sa w nastepujacych miejscach: Krowodrza, Polnocno-Wschodnia czesc Debnikow bezposrednio sasiadujaca ze Starym Miastem, a takze okolice Bienczyc.

```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1  824
## 2     2 1014
## 3     3   51
## 4     4   64
## 5     5   47
```

Klasteryzacja 2, ponownie z wykorzystaniem algorytmu DBSCAN, natomiast zmienionymi parametrami:

```
dbscan_res <- dbscan(cords, eps = 800, minPts = 60)
color<-dbscan_res$cluster+1
plot2<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("DBSCAN\neps=800, minPts=60") +
  theme(plot.title = element_text(hjust = 0.5))+
  guides(colour = guide_legend(override.aes = list(size=4)))
plot2
```



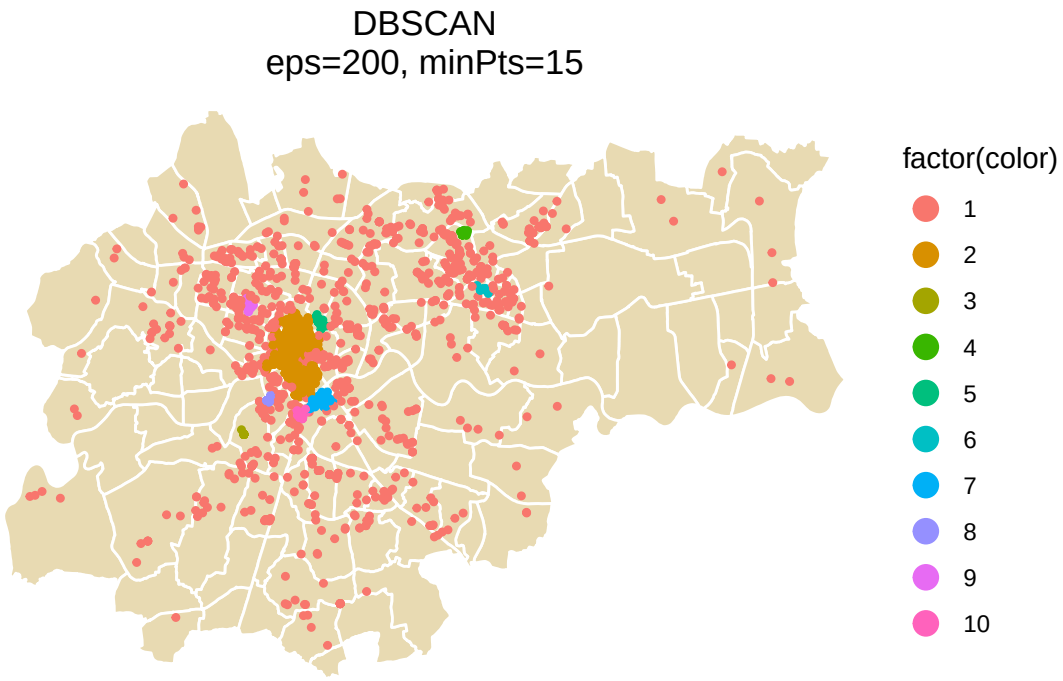
Podwajajac wartosci parametrow spodziewalem sie, ze liczba klastrow drastycznie zmaleje. Mozna zauwazyc, ze trzy klastry z poprzedniej wizualizacji zostaly scalone (numer 2,3 oraz 5). Do zielonego klastra nalezy juz ponad 60% obserwacji, natomiast 10% zarejestrowanych wykroczen odnotowane zostaly na obszarze lub w okolicach Bienczyc oraz zachodniej czesci Nowej Huty i Wzgorz Krzeslawickich (mozemy zaobserwowac, ze jest to rozszerzony klaster numer 4 z poprzedniej wizualizacji, jego licznosc zwiekszyla sie trzykrotnie)

```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1  563
## 2     2 1241
## 3     3  196
```

Kolejna klasteryzacja (3.) z wykorzystaniem algorytmu DBSCAN oraz parametramii o znacznie mniejszej wartosci.

```
dbscan_res <- dbscan(cords, eps = 200, minPts = 15)
color<-dbscan_res$cluster+1
plot3<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("DBSCAN\neps=200, minPts=15") +
  theme(plot.title = element_text(hjust = 0.5) )+
  guides(colour = guide_legend(override.aes = list(size=4)))
plot3
```



Liczba klastrow wzorsla odwrotnie proporcjonalnie do ich licznosci. W dalszej czesci klastrem zdecydowanie dominujacym jest klaster numer 2, zlokalizowany na obszarze calego Starego Miasta. Dwa kolejne klastry pod wzgledem wielkosci zlokalizowane sa w okolicach Olszy oraz Podgorza (oba bezposrednio sasiaduja z klastrem numer 2). Kolejne klastry(wszystkie o licznosci z zakresu 18-23 obserwacji) znajduja sie na nastepujacych obszarach: Zakrzowek, Debniki, Krowodrza, oraz dwa w okolicach Bienczyc i Mistrzejowic. Zwazywszy na maly epsilon mniejsze klastry obejmuja obszar danej ulicy lub skrzyzowania. Gdyby zaobserwowane wykroczenia dotyczyly np zlego parkowania, moznaby wnioskowac, ze dane miejsca sa slabo oznaczone, przez co kierowcy parkuja w niedozwolony sposob. Uwazam rozwniez, ze dobrane parametry z praktycznego punktu widzenia sa zbyt male i nie dostarczaja zetelnych informacji i moga wprowadzac w blad, gdyz wiekszosc klastrow bezposrednio sasiaduje ze soba.

```
as.data.frame(table(color))
```

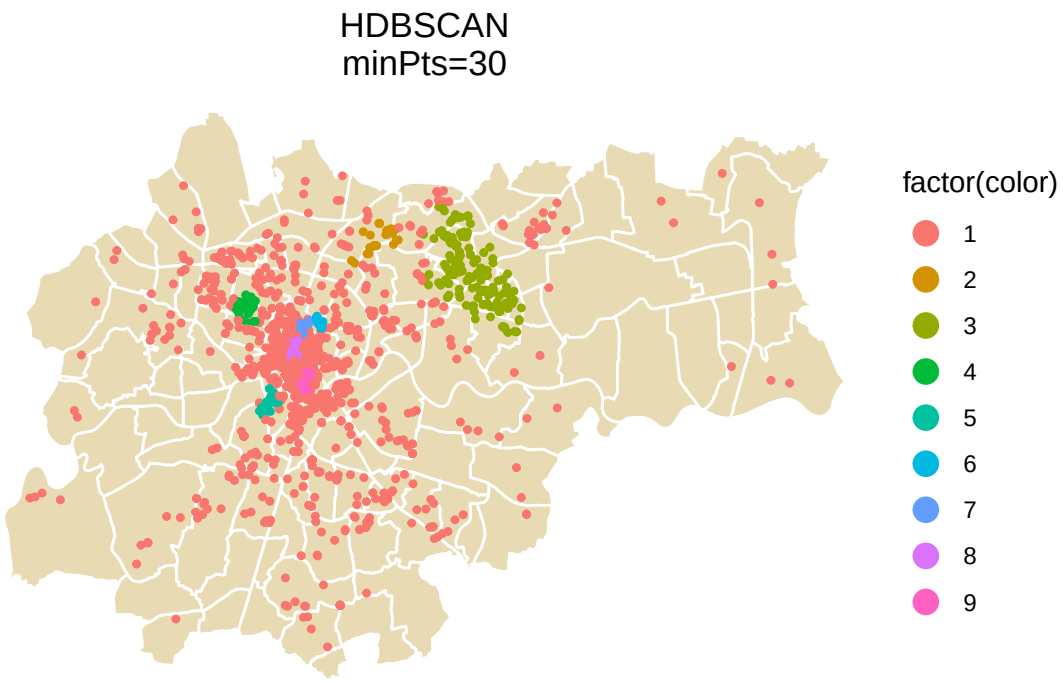
```
##    color Freq
```

```
## 1      1  991
## 2      2  783
## 3      3   19
## 4      4   19
## 5      5   56
## 6      6   23
## 7      7   49
## 8      8   21
## 9      9   21
## 10    10   18
```

Algorytm HDBSCAN: Parametry wejsciowe: minPts - minimalna wymagana liczba punktow znajdujaca sie w pewnejej odleglosci, aby uznac je za klaster Schemat dzialania: Jest to modyfkiacja algorytmu DBSCAN. Pierwszym krokiem jest znalezienie odleglosci rdzeniowej, czyli odleglosci w jakiej lezy ilosc punktow podanej w parametrze minPts. Nastepnie algorytm wyszukuje najwieksze skupiska, biorac pod uwage gestosc rozmieszczenia punktow wokol Zalety: -jest zbudowany dla rzeczywistego scenariusza, w którym dane są o różnej gęstości -jest jeszcze szybszy od DBSCAN -jedynym argumentem jest minimalny rozmiar klastra, jest to dla nas bardziej intuicyjny parametr, gdyz na ogol wiemy jakiego rozmiaru klastrow sie spodziewamy/oczekujemy Wady: -tak jak w DBSCAN - nie daje bezposredniej mozliwosci zdefiniowania oczekiwanej liczby klastrow - ich liczba wynika z wartosci parametrow wejsciowych

Klasteryzacja 1 z wykorzystaniem algorytmu HDBSCAN:

```
hdbscan_res <- hdbscan(cords, minPts = 30)
color<-hdbscan_res$cluster+1
plot4<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("HDBSCAN\nminPts=30") +
  theme(plot.title = element_text(hjust = 0.5) ) +
  guides(colour = guide_legend(override.aes = list(size=4)))
plot4
```



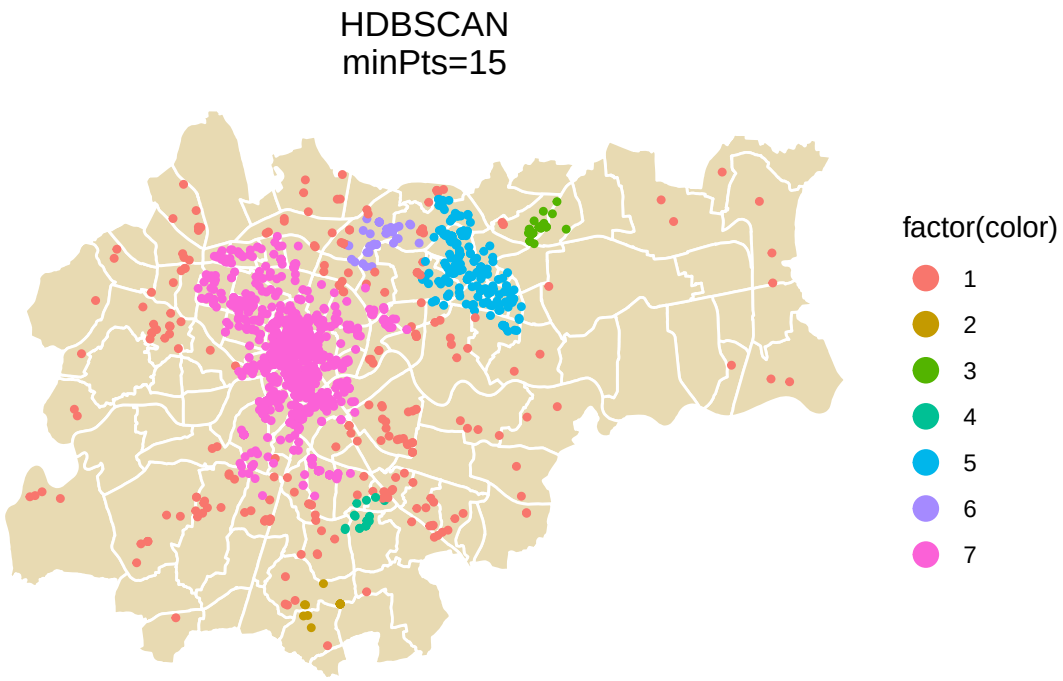
Porownujac rezultat z pierwsza wizualizacja algorytmem DBSCAN od razu zauwazamy, ze przy tej samej liczbie minimalnej liczby punktow tworzacej klastrow ich ilosc podwoila sie. Moze to sugerowac, ze przyjety tam parametrs eps=400 byl zbyt maly i moglby sie nie sprawdzic w pewnych zastosowaniach. Potwierdza sie rowniez, ze algorytm HDBSCAN przy tworzeniu klastrow bierze zmienna gestosc wystepwania zjawisk w zaleznosci od otoczenia - bardzo dobrze widac to porownujac klastry 3 oraz 8. Punkty wchodzace w sklad trzecie klastra (zlokalizowany na pograniczu Mistrzejowic, Bienczyc oraz Krzeslawic) sa mocno odseparowane i dlatego, mimo ze gestosc punktowa tego klastra w porownaniu do innych klastrow jest mala, zostaly uznany jako klaster. Dla porownania osmy klaster, znajdujacy sie w samym centrum starego miasta (prawdopodobnie sa to okolice Rynku), na mapie sprawia wrazenie raczej fragmentu klastra. Wynika to z faktu, ze gestosc punktowa wokol tego obszaru, mimo ze jest spora, jest nieco mniejsza i na tej podstawie algorytm stworzyl klaster w miejscu o najwiekszej gestosci zarejestrowanych wykroczen w obrebie Starego Miasta. Pozostale klastry znajduja sie tak jak poprzednio na nastepujacych obszarach: Krowodrza, Pradnik Czerowny, Debniki/Zakrzowek, ponadto w sumie na Starym MIescie wydzielono cztery klastry. Rozmiary klastrow:

```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1 1212
## 2     2   30
## 3     3  214
## 4     4   40
## 5     5   36
## 6     6   53
## 7     7   80
## 8     8  233
## 9     9  102
```

Klasteryzacja 2 z wykorzystaniem algorytmu HDBSCAN oraz mniejszym parametrem wejsciowym minPts:

```
hdbscan_res <- hdbscan(cords, minPts = 15)
color<-hdbscan_res$cluster+1
plot5<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("HDBSCAN\nminPts=15") +
  theme(plot.title = element_text(hjust = 0.5) ) +
  guides(colour = guide_legend(override.aes = list(size=4)))
plot5
```



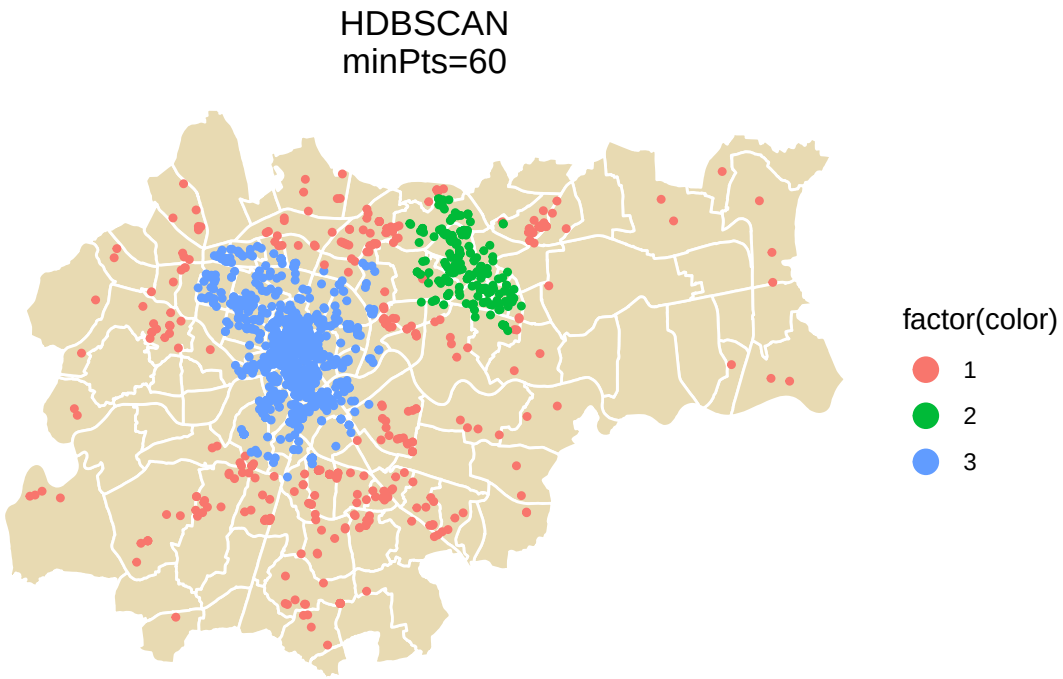
Znaczne zmienjszenie minimalnej liczby punktow wymaganej do utworzenia klastra prawdopodobnie nie dalo satysfakcjonujacego rezultatu. Ponad 87% punktow nalezy do ktoregos z szesciu klastrow, przy czym najwiekszy stanowi ponad 70% obserwacji, natomiast najmniejszy niespelna 0.8%. Uwazam, ze algorytm ten z tak malym argumentem minPts nie sprawdzil sie w tym przypadku, poniewaz gestosc punktowa jest zbyt zroznicowana, przez co obszar, gdzie zarejestrowana liczba wykroczen jest najwyzsza (Stare Miasto, Krowodrza, czesc Pradnika Bialego, Bronowic, Zwierzynca i Debnikow) uznany zostal za obszar wystepowania jednego, ogromnego klastra. Na tej samej mapie na obszarze Swoszowic zauwazyc mozemy klaster numer 2, na ktory sklada sie 16 punktow. Intuicyjnie na obszarze tym ciezko stwierdzic, zeby faktycznie wystepowal jakis klaster, natomiast ze wzgledu na znikoma ilosc wykroczen wokol tego obszaru algorytm zakwalifikowal to zgrupowanie jako samodzielny klaster. Wnioski jakie nalezaloby wyciagnac w tym miejscu to, ze parametr minPts nalezy dobierac odpowiednio proporcjonalnie do liczby punktow.

```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1  247
## 2     2   16
## 3     3   27
## 4     4   17
## 5     5  223
## 6     6   39
## 7     7 1431
```

Klasteryzacja 3 z wykorzystaniem algorytmu HDBSCAN:

```
hdbscan_res <- hdbscan(cords, minPts = 60)
color<-hdbscan_res$cluster+1
plot6<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("HDBSCAN\nminPts=60") +
  theme(plot.title = element_text(hjust = 0.5) ) +
  guides(colour = guide_legend(override.aes = list(size=4)))
plot6
```



Prezentowana wizualizacja jest mocno zblizona do prezentowanego wyzej rezultatu klastrowania za

pomoca algorytmu DBSCAN z parametrami minPts=60 i eps=800. W tym przypadku oba klastry sa nieznacznie wieksze, natomiast intuicyjnie obszar klastrow jest wlasciwy. Zastosowanie takiej, stosunkowo duzej, wartosci parametru minPts moze byc przydatne w momencie, gdy nie potrafimy okreslic punktow granicznych dla glownych klastrow z zbiorze.

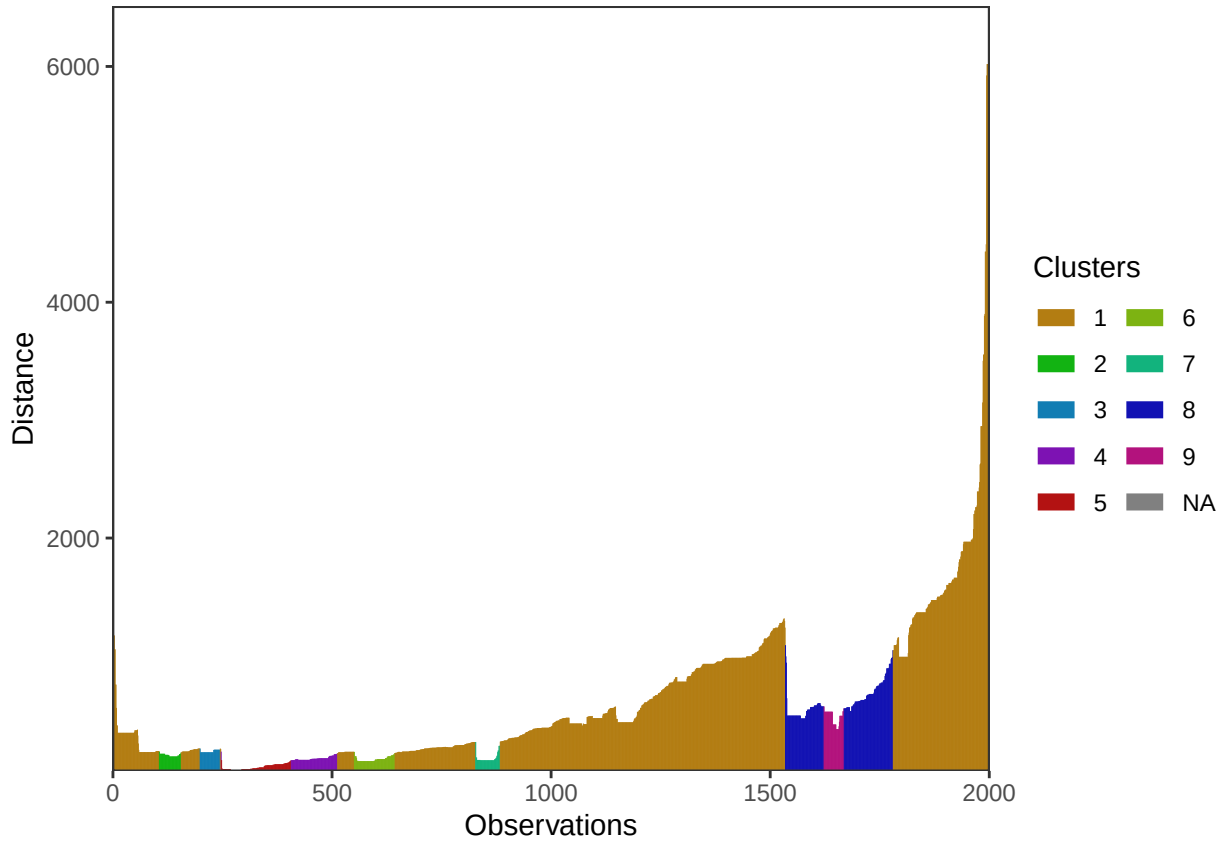
```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1  370
## 2     2  236
## 3     3 1394
```

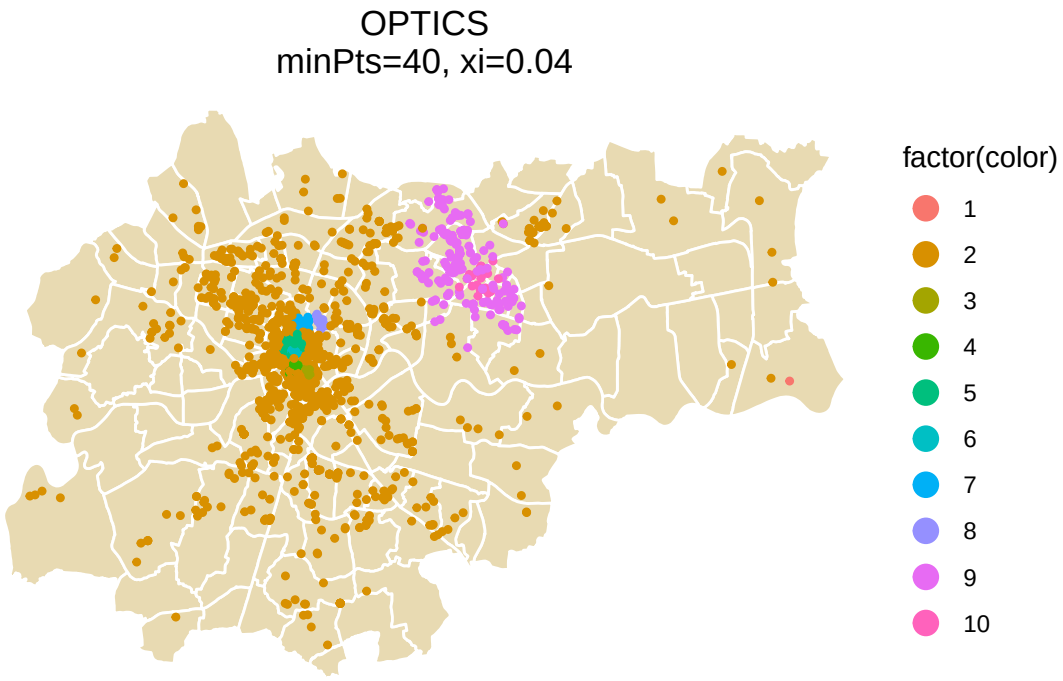
Algorytm DBSCAN: Parametry wejsciowe: minPts - jak wyzej, zdefiniowana minimalna liczba obiektow rozwazanych jako grupa/klaster Xi - wspolczynnik stromosci Schemat dzialania: Algorytm wybiera losowo jeden obiekt, od ktorego rozpoczyna proces przetwarzania obiektow, nastepnie rozpoczyna analize kolejnego, najblizszego obiektu, dodaje go do listy przetworzonych obiektow i uznaje za prekursora w poszukiwaniu kolejnego obiektu. Poszukiwanie dla kolejnych punktow najblizszych sasiadow do momentu, gdy przeanalizowane zostana wszystkie punkty. Zasadnicza roznica pomiedzy DBSCAN i OPTICS jest taka, ze w tej drugiej wazna jest kolejnosc przetwarzanych obiektow. Algorytm OPTICS zwraca uporzadkowana baze danych h, przechowującą dla każdego elementu informację o odlgosci-jadra, odleglosci-osiagalnosci. Graficzna prezentacja wyzej uporzadkowanych punktow pozwala na latwe zrozumienie danych. Doliny na takim wykresie oznaczaja wystepowanie klastrow. Im dolina jest szersza, tym kalster jest wiekszy. Rozna glebokosc dolin wskazuje natomiast na rozna gestosc klastrow. Zalety: -eliminuje wade swojego pierwowzoru(DBSCAN) i potrafi znalezc zgrupowania hierarchicznych (mniejsze skupienie wewnatrz wiekszego) i roznym stopniu zageszczenia Wady: -brak przetestowanej heurystyki do wyznaczenia wartosci parametru MinPts -wolniejszy okolo 1,6 raza od DBSCAN

Klasteryzacja 1 z wykorzystaniem algorytmu OPTICS

```
optic_res <- optics(cords, minPts = 40)
optic_res2 <- extractXi(optic_res, xi = 0.04)
ggplot_optics(optic_res, groups = optic_res2$cluster)
```




```
color<-optic_res2$cluster+1
plot7<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("OPTICS\nminPts=40, xi=0.04") +
  theme(plot.title = element_text(hjust = 0.5) ) +
  guides(colour = guide_legend(override.aes = list(size=4)))
plot7
```



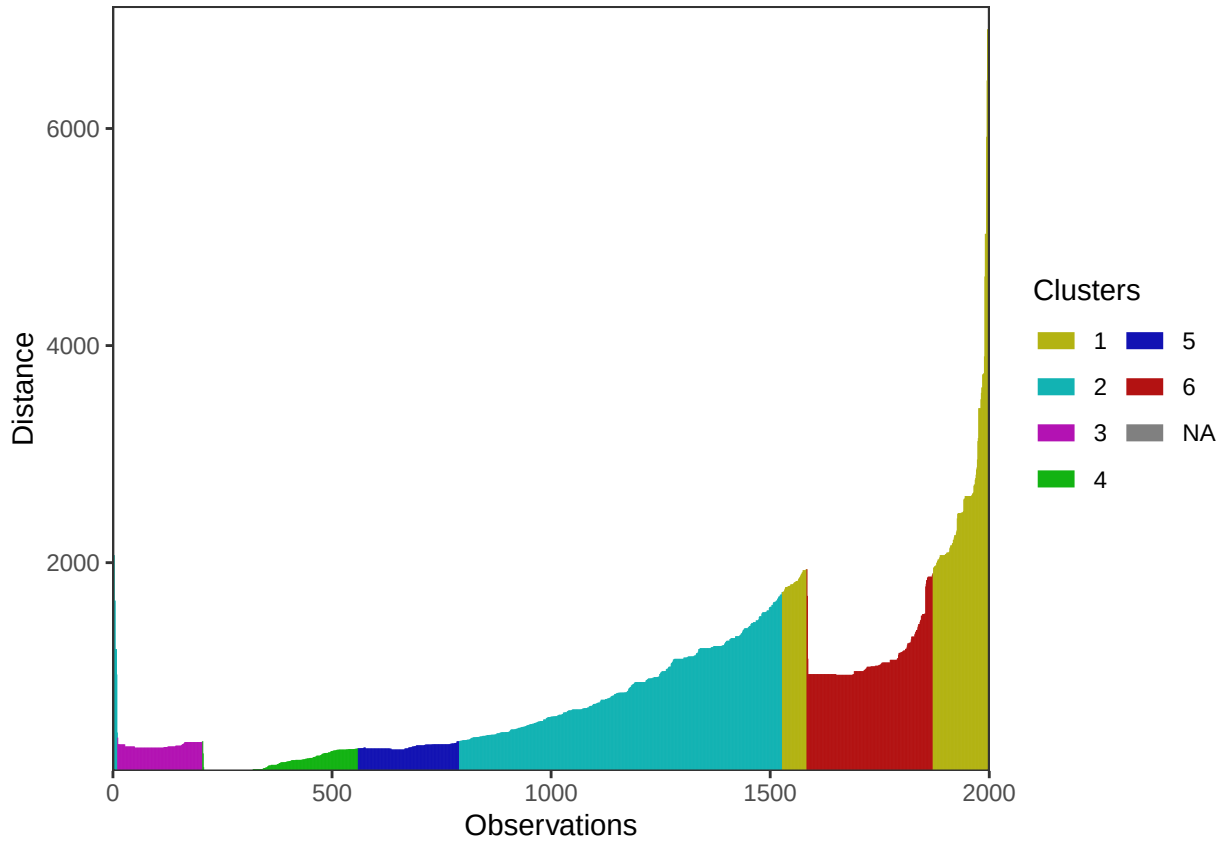
Jak widac dla wybranych parametrow, wszystkie punkty (z wyjatkiem jednego) zostaly zakwalifikowane do ktoregos z klastrow. Przy danych parametrach wejsciowych otrzymalem 8 mniejszych klastrow o bardzo duzej gestosci punktowej oraz jeden zawierajacy ponad 60% obserwacji. Piec mniejszych klastrow zlokalizowanych jest na terenie Starego Miasta, kolejne na Czyzynach oraz Bienczycach. Nalezaloby sie zastanowic czy parametry zostaly dobrane odpowiednio i czy klaster numer 2 faktycznie miesci sie w definicji slowa klaster.

```
as.data.frame(table(color))
```

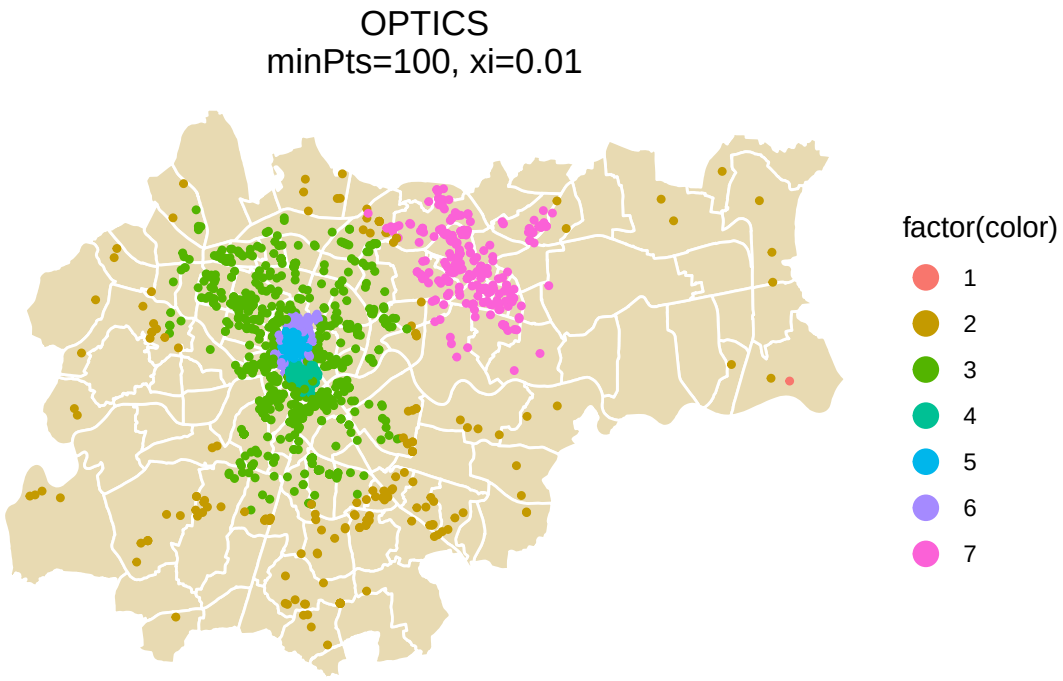
```
##    color Freq
## 1      1    1
## 2      2 1242
## 3      3   50
## 4      4   46
## 5      5  107
## 6      6  159
## 7      7   93
## 8      8   56
## 9      9  200
## 10    10   46
```

Klasteryzacja 2 z wykorzystaniem algorytmu OPTICS

```
optic_res <- optics(cords, minPts = 100)
optic_res2 <- extractXi(optic_res, xi = 0.01)
ggplot_optics(optic_res, groups = optic_res2$cluster)
```




```
color<-optic_res2$cluster+1
plot8<-ggplot() +
  geom_polygon(data = my_spdf, aes( x = long, y = lat, group = group), fill="#E8DAB2", color="white") +
  geom_point(data = cords, aes(x = X, y = Y,color=factor(color)), size = 0.9) +
  theme_void() +
  theme(plot.margin = unit(c(1,1,1,1), "cm")) +
  ggtitle("OPTICS\nminPts=100, xi=0.01") +
  theme(plot.title = element_text(hjust = 0.5) ) +
  guides(colour = guide_legend(override.aes = list(size=4)))
plot8
```



Diametralna zmiana parametrow w celu znalezienia glownych klastrow. Tym razem na terenie starego miasta znajduja sie 3 klastry, wszystkie zdarzenia wokol Starego Miasta zakwalifikowane zostaly do nastepnego klastra (37% wszystkich zarejetrowanych wykroczen) i jeszcze jeden na terenie Czyzyn, Bienczyc i Mistrzejowic. Ponownie wszystkie pozostale punkty zakwalifikowane zostaly do zewnetrznego klastra, co moim zdaniem jest bledem.

```
as.data.frame(table(color))
```

```
##   color Freq
## 1     1    1
## 2     2  186
## 3     3  745
## 4     4  194
## 5     5  355
## 6     6  231
## 7     7  288
```

PODSUMOWANIE I WNIOSKI Kazda z metod klasteryzacji charakteryzuje sie pewnymi cechami, ktore w danwej sytuacji moga okazac sie kluczowe do wykorzystania tego konkretnego algorytmu. Algorytm DBSCAN jest najbardziej intuicyjny i daje nam najwieksze pole manewru. Oosobiscie uwazam jednak, ze lepszym algorytmem jest HDBSCAN, ktory bierze pod uwage zmienna gestosc punktowa i w lepszy sposob wyznacza granice klastrow. Jesli chodzi o algorytm OPTICS to jest on rowniez zmodyfikowana wersja algorytmu DBSCAN i uwazam, ze jego bardzo duzym atutem jest umiejetnosc tworzenia klastrow w klastrach, natomiast nie udalo mi sie dobrac argumentow w taki sposob, aby punkty niezgrupowane uznane zostaly za szum.