DBSCAN: Density-based spatial clustering of applications with noise. Algorytm łączy w klastry obserwacje leżące blisko siebie (wg zasady sąsiad mojego sąsiada należy do tej samej grupy, co ja). Algorytm ignoruje pojedyncze punkty lub niewielkie skupiska.

Dwa paramtery:

•epsilon -jeśli dla dwóch punktów odległość $d(x1,x2)<epsilon$, to należą do tej samej grupy

•min_points–ignorowane są grupy $|Ci|$<=min_$points$(zawierające mniej niż min_$points$ obserwacji)

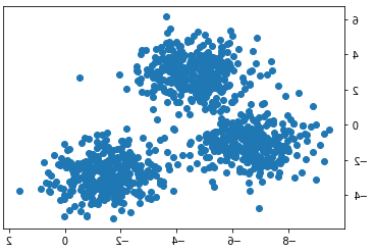
Algorytm hierarchiczny (aglomeracyjny),

Ważne elementy, decydujące o jego działaniu:

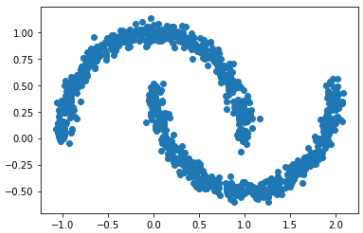
- Dobór metryki
- Metoda łączenia grup

Należy porównać działanie K-means, DBSCAN i alogorytmu hierarchicznego. Porównanie przeprowadzić na 3 zbiorach danych, które należy wygenerować. Te zbiory danych uwypuklają wady i zalety różnych algorytmów znacznie lepiej niż "rzeczywiste" zbiory.

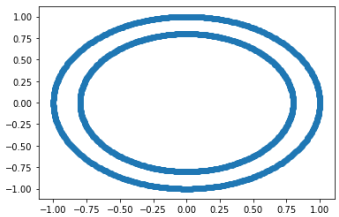
1 zbiór dane na płaszczyźnie tworzące trzy zwarte koła.



2 zbiór dwa rogale



3 pierścień w pierścieniu



W pythonie generujemy dane:

```
x, y_true = make_blobs(n_samples=1000, centers=3, cluster_std=0.99,
random_state=3042019)
df = pd.DataFrame(x, columns = ['f1', 'f2'])


x, y = make_moons(1000, noise=.05, random_state=0)
df_moons = pd.DataFrame(x, columns = ['f1','f2'])

X, _ = make_circles(n_samples=1000, random_state=0)
df_circles = pd.DataFrame(X, columns = ['f1', 'f2'])
```

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.datasets.samples_generator import make_blobs, make_moons, make_circles
#x, y = make_moons(1000, noise=.05, random_state=0)
# df_moons = pd.DataFrame(x, columns = ['f1','f2'])
# df_moons = df_moons.round(4)
# df_moons.to_csv(r"C:\Users\User\PythonPlikiCsv\ksiezyceCSV.csv", index = False)


#X, y_true = make_blobs(n_samples=1000, centers=3, cluster_std=0.99, random_state=3042019)
#df = pd.DataFrame(X, columns = ['f1', 'f2'])
#df=df.round(4)
#df.to_csv(r"C:\Users\User\PythonPlikiCsv\kuleCSV.csv", index = False)

X, _ = make_circles(n_samples=1000, random_state=0)
df_circles = pd.DataFrame(X, columns = ['f1', 'f2'])
df_circles=df_circles.round(4)
df_circles.to_csv(r"C:\Users\User\PythonPlikiCsv\pierscienieCSV.csv", index = False)
```

Przeprowadzić klasteryzację na całym zbiorze czyli z opcją: *use training set*

Jako wynik wkleić wizualizację dla każdego z 3 algorytmów i podać parametry, dla których wynik był najlepszy. Wnioski: Czy kształt klastrów ma wpływ na wynik działania danego algorytmu?