Streszczenie

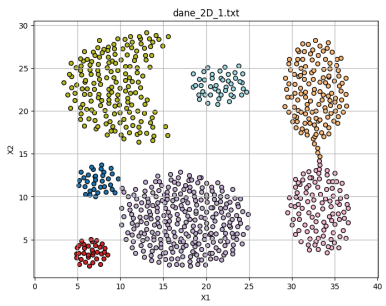
W poniższym dokumencie opisuję moją pracę w związku z 4 miniprojektem z MPUM, dotyczącym algorytmów klasteryzacji

Wprowadzenie

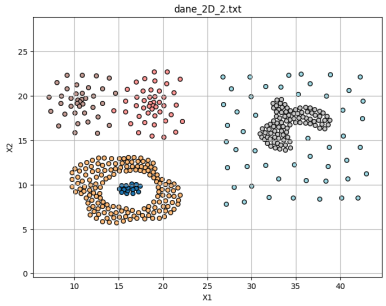
Na potrzeby tego projektu porównuję działanie algorytmów `klasteryzacji hierarchicznej` `algorytmu k-średnich` i `klasteryzacji spektralnej`. W pierwszej części raportu zajmę się wyznaczaniem liczby klastrów dla zbiorów `dane_18D.txt` i `dane_2D_n.txt`. Następnie dla każdego z algorytmów porównam jego działanie na zbiorach `rp.data`, `dane_2D_n.txt` dla k będącego podaną liczbą klas, zaś dla pliku `dane_18D.txt` wyznaczę wartość k w oparciu o później opisane metody. Na początku przyjrzyjmy się podanym zbiorom 2D.

1 Dane 2D

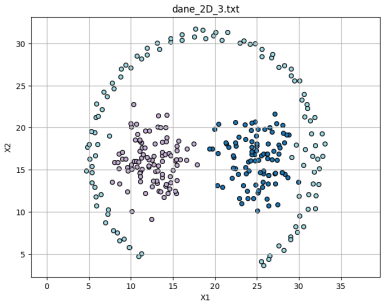
Do miniprojektu zostały dołączone dane 2D. Żeby wiedzieć z czym pracujemy poniżej zamieszczam ich wizualizację, wraz z opisem liczby klas.



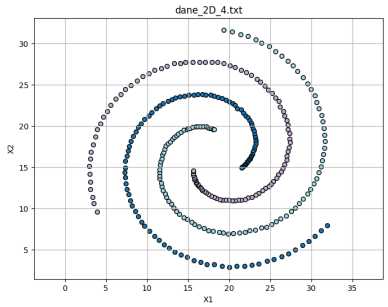
Zestaw 1: 7 Klas



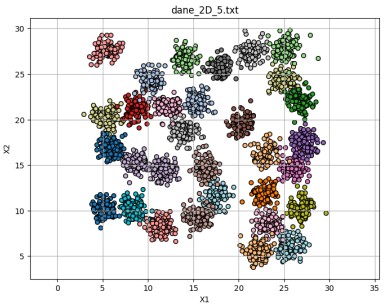
Zestaw 2: 6 Klas



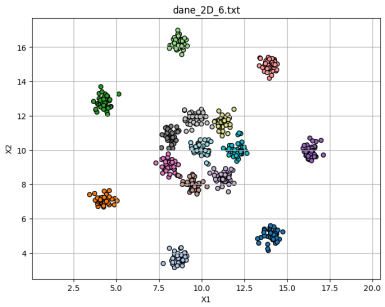
Zestaw 3: 3 Klas



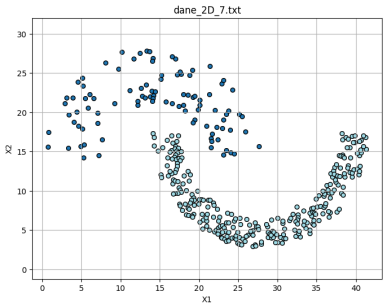
Zestaw 4: 3 Klas



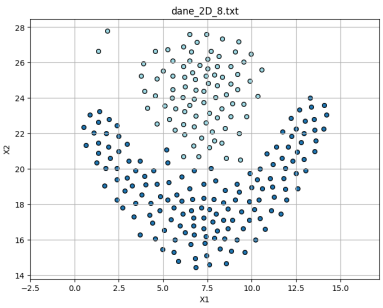
Zestaw 5: 31 Klas



Zestaw 6: 15 Klas



Zestaw 7: 2 Klasy



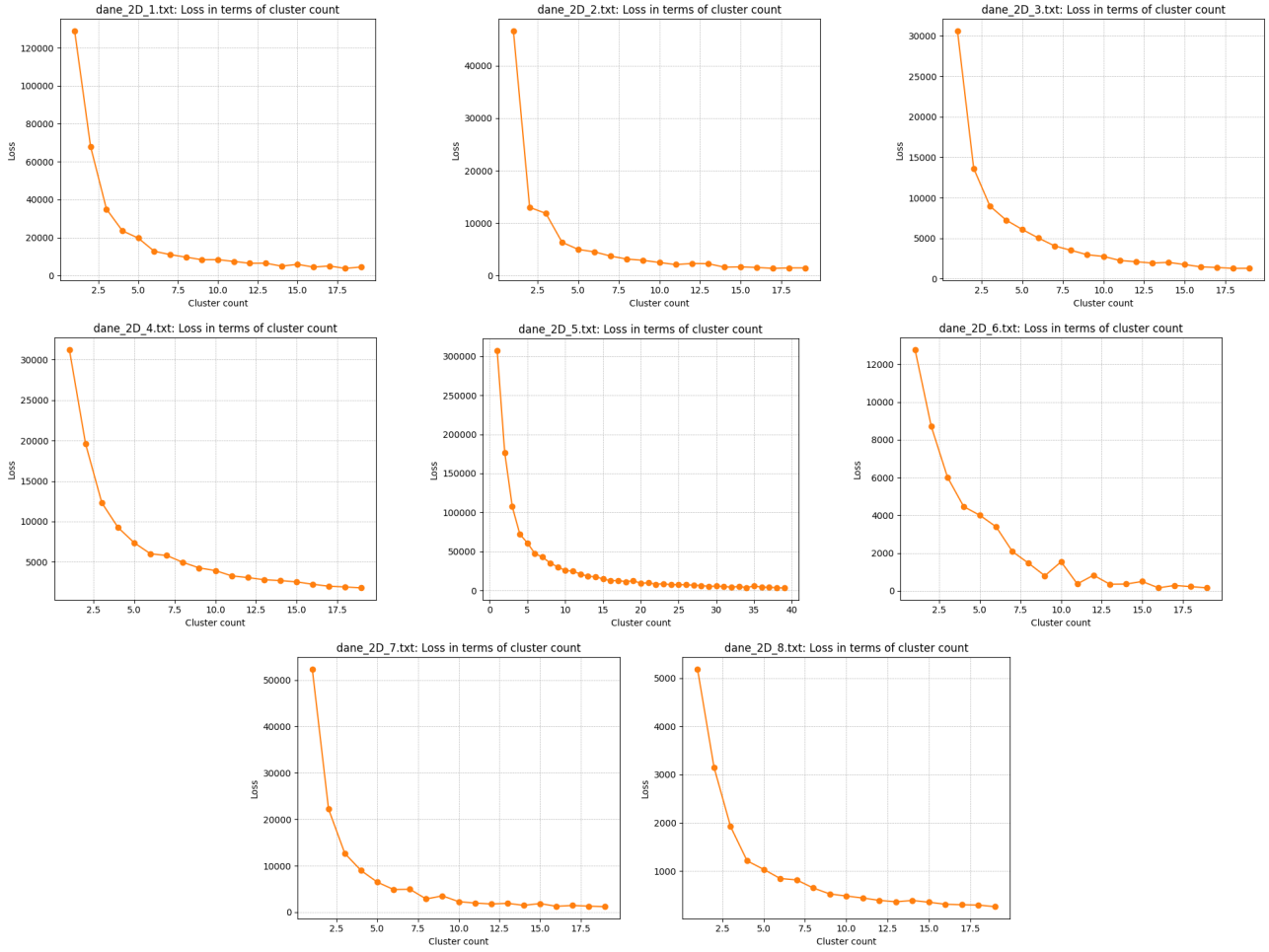
Zestaw 8: 2 Klasy

Przyglądając się danym widzimy, że wiele z tych klasyfikacji nie daje kształtów kulistych, w związku z czym zakładam że algorytmy takie jak `k-średnich`, czy niektóre metody łączenia

w `grupowaniu hierarchicznym` mogą dawać odmienne od zadanych wyniki. Zanim jednak do tego przejdziemy, spróbujmy wyznaczyć liczbę klastrów dla naszych zbiorów danych.

Wyznaczanie liczby klastrów

Spróbujmy do tego skorzystać z metody łokciowej. Poniżej przedstawiam wykresy kolejnych wykresów błędu względem liczby klas w metodzie `k-średnich`



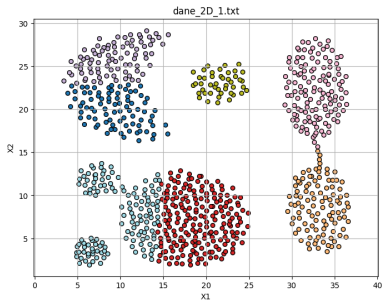
Widzimy, że pierwszy i drugi wykres się wypłaszcza po $k = 6$, zaś szósty przy $k = 15$, a przy siódmym i ósmym wykresie najwiekszy spadek jest przy $k = 2$. Jednak korzystając z tej metody ciężko jednoznacznie wyznaczyć liczbę klastrów. W późniejszej części raportu przeanalizujemy resztę zbiorów korzystając z innej metody.

Miara dokładności

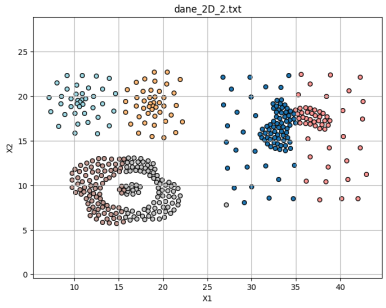
Przed dalszą częścią materiału zaznączę tylko, że dokładność mierzę w ten sposób że spośród wszystkich par punktów (X, Y) ze zbioru w którym znamy klastry patrzę czy X, Y są klasyfikowane przez model jako należące do jednej klasy wtedy i tylko wtedy gdy w danych testowych też należą do jednej klasy. Wtedy dokładnością jest średnia wartość tego po wszystkich parach (X, Y).

Algorytm k-średnich

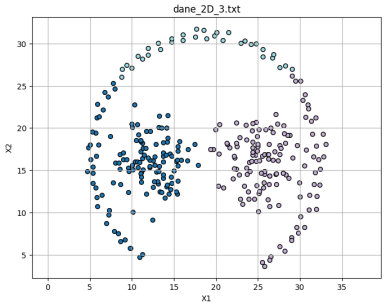
Na początek przeanalizujmy wyniki które zwrócił algorytm `k-średnich`. Dla danych 2D uruchomiłem algorytm na tylu klasach, ile zostało wyszczególnionych wśród danych wejściowych. Poniżej znajdują się wyniki podziału tych danych na klasy. Spośród wszystkich metryk które testowałem najciekawsze wyniki dała `metryka Euklidesowa`, więc jej wynikami się zajmijmy



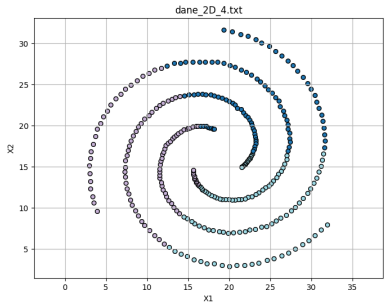
Zestaw 1: 7 Klas



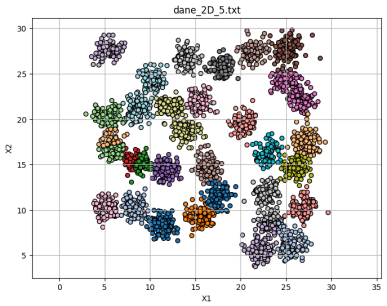
Zestaw 2: 6 Klas



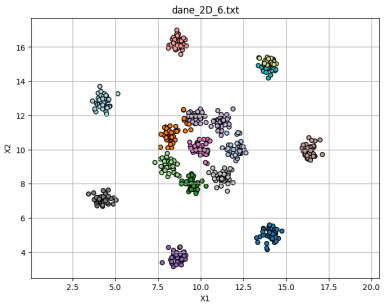
Zestaw 3: 3 Klas



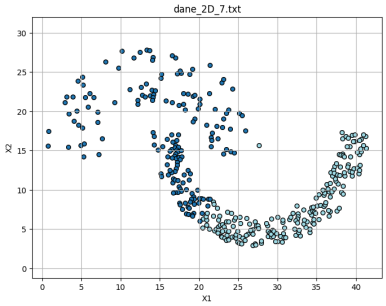
Zestaw 4: 3 Klas



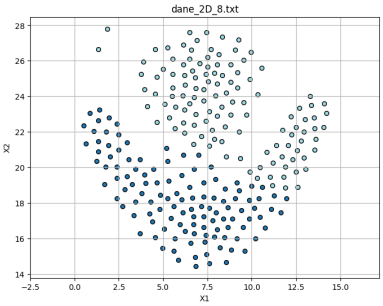
Zestaw 5: 31 Klas



Zestaw 6: 15 Klas



Zestaw 7: 2 Klasy



Zestaw 8: 2 Klasy

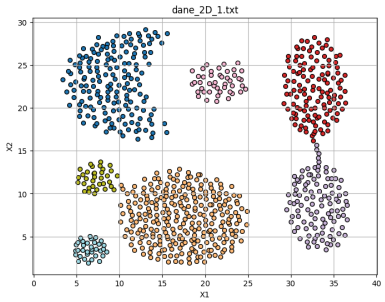
Wnioski są dosyć spodziewane, algorytm bardzo ładnie poradził sobie w zestawach 5 i 6, gdzie rozważane dane były kuliste. Również nienagannie zaklasyfikował niektóre klasy spośród zestawu 1 i 2, oraz przyzwoicie poradził sobie z zestawami 7 i 8. Największy problem miał on w zestawach 3 i 4, gdzie szukane klasy nie są skupione wokół punktów, tylko dane w jednej klasie zależą od siebie w inny sposób. Spójrzmy na wyniki naszego modelu na poszczególnych zestawach danych.

Algorytm grupowania hierarchicznego

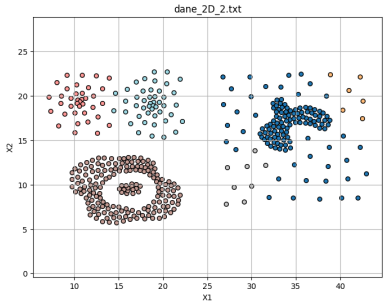
Na początek zdecydowałem się sprawdzić, jak poradzi sobie łączenie centroidalne. Poniżej przedstawiam wyniki zarówno w formie graficznej, jak i tabelki dokładności:

Zestaw	**Dokładność**
Dane nr 1	0.927
Dane nr 2	0.906
Dane nr 3	0.749
Dane nr 4	0.554
Dane nr 5	0.979
Dane nr 6	0.974
Dane nr 7	0.662
Dane nr 8	0.744

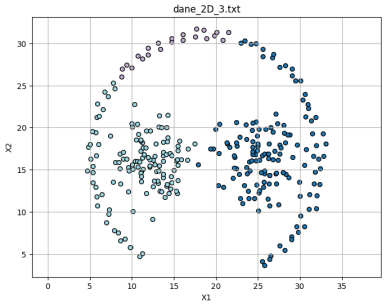
Tabela 1: Dokładności algorytmu k-średnich



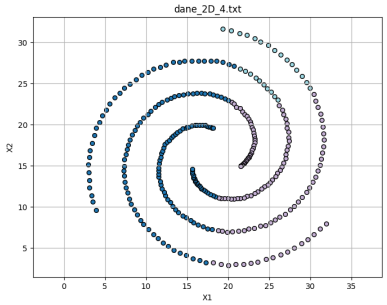
Zestaw 1: 7 Klas



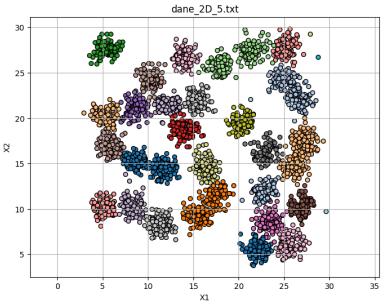
Zestaw 2: 6 Klas



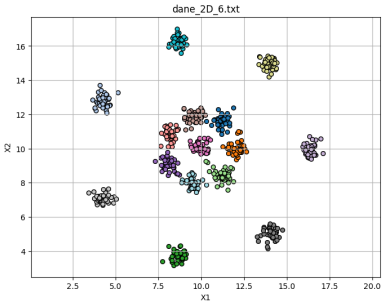
Zestaw 3: 3 Klas



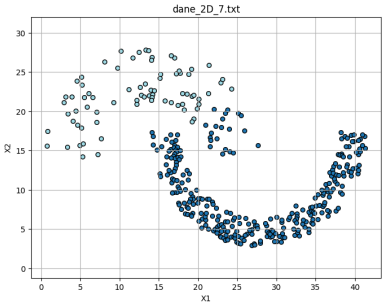
Zestaw 4: 3 Klas



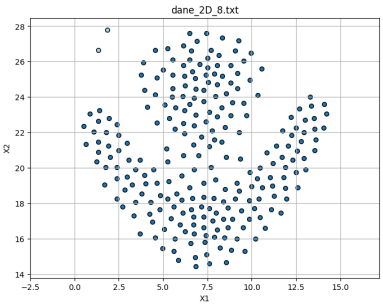
Zestaw 5: 31 Klas



Zestaw 6: 15 Klas



Zestaw 7: 2 Klasy

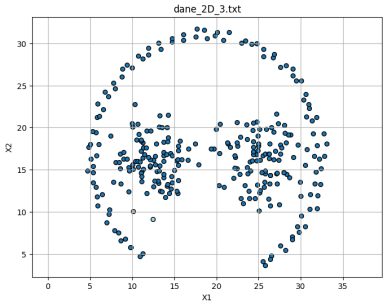


Zestaw 8: 2 Klasy

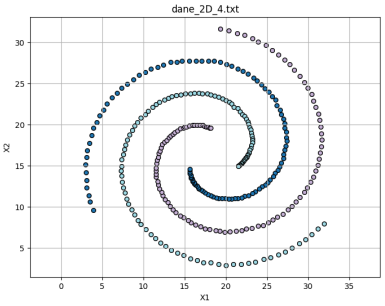
Porównując tą technikę do algorytmu `k-średnich`, widzimy wyższe dokładności na zbiorach 1, 5, 6 gdzie algorytm uzyskał ekstremalnie wysokie dokładności, jednak w zestawie 8 zaklasyfikował prawie wszystkie punkty do jednej klasy a tylko dwa z boku do drugiej. Mimo to nadal nie poradził sobie z klasyfikacją zestawów 3 i 4, zatem wytestujmy na nich łączenie pojedyncze.

Zestaw	**Dokładność**
Zestaw nr 1	0.998
Zestaw nr 2	0.918
Zestaw nr 3	0.725
Zestaw nr 4	0.526
Zestaw nr 5	0.988
Zestaw nr 6	0.998
Zestaw nr 7	0.893
Zestaw nr 8	0.541

Tabela 2: Dokładności łączenia centroidalnego



Zestaw 3: Dokładność 0.337



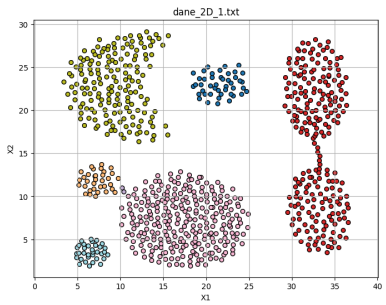
Zestaw 4: Dokładność 1.000

Rysunek 6: Wyniki łączenia pojedynczego

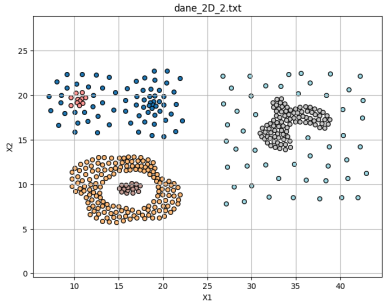
Widzimy, że dla zestawu 4 algorytm wszystkie punkty zaklasyfikował poprawnie, jednak nadal beznadziejnie mu poszło na zestawie 3, gdzie prawie wszystko wrzucił do jednej klasy. Testując inne metody łączenia nie uzyskałem lepszych rezultatów dla zestawu 4 niż grupowanie centroidalne.

Klasteryzacja Spektralna

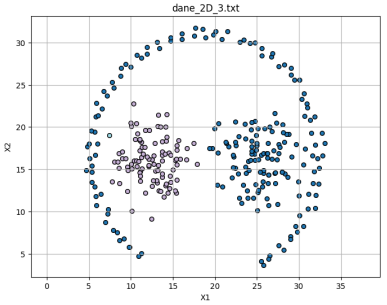
Ten algorytm zdecydowałem się napisać w oparciu o załączony do miniprojektu artykuł `On Spectral Clustering: Analysis and an algorithm`. Jedyna różnica jest taka, że po przekształceniu danych używam algorytmu `grupowania hierarchicznego` z łączeniem pojedynczym do wyznaczenia klastrów. Jeśli chodzi o parametr σ, to najlepiej sprawdzał się $\sigma = 1$. Zobaczmy jak sobie radzi z naszymi zestawami danych.



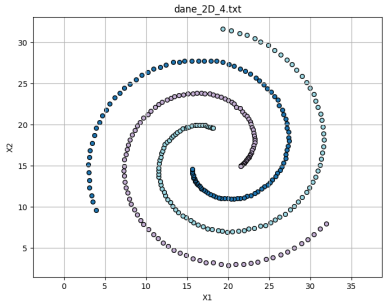
Zestaw 1: 7 Klas



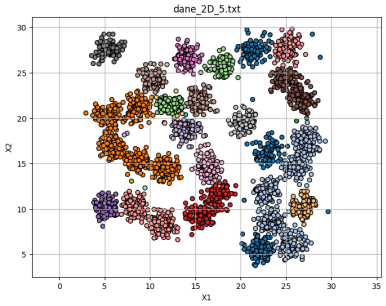
Zestaw 2: 6 Klas



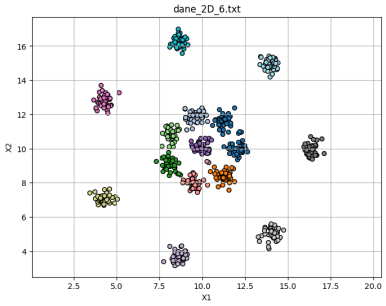
Zestaw 3: 3 Klas



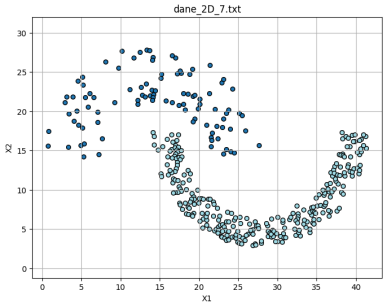
Zestaw 4: 3 Klas



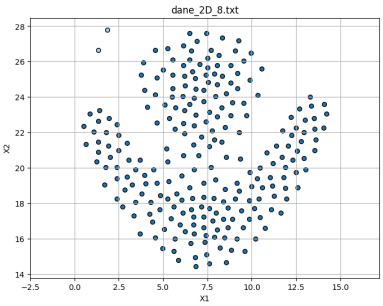
Zestaw 5: 31 Klas



Zestaw 6: 15 Klas



Zestaw 7: 2 Klasy

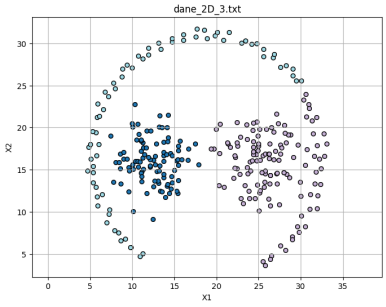


Zestaw 8: 2 Klasy

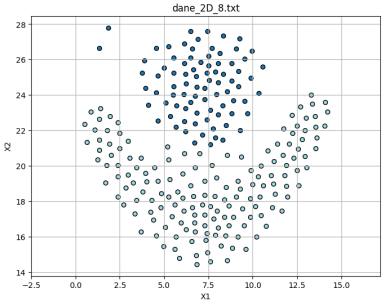
Zestaw	**Dokładność**
Zestaw nr 1	0.955
Zestaw nr 2	0.981
Zestaw nr 3	0.772
Zestaw nr 4	1.000
Zestaw nr 5	0.960
Zestaw nr 6	0.990
Zestaw nr 7	1.000
Zestaw nr 8	0.541

Tabela 3: Dokładności łączenia centroidalnego

Zauważmy, że przy tak dobranych parametrach nasz algorytm bajecznie poradził sobie z zestawami 4, 6 i 7, świetny wynik uzyskał też na zestawach nr 1, 2 i 5. Do poprawnego sklasyfikowania zostają zestawy 3 i 8. Zmieniając metodę łączenia klastrów na łączenie centroidalne udało mi się uzyskać następujące wyniki. Jest to najlepszy wynik jaki uzyskałem na tych zbiorach danych.



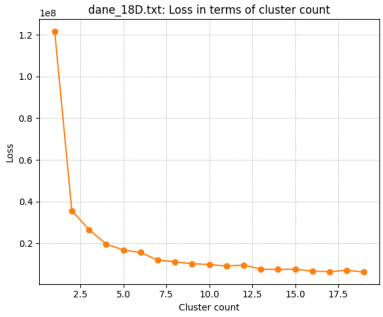
$\sigma = 1$, dokładność 0.853



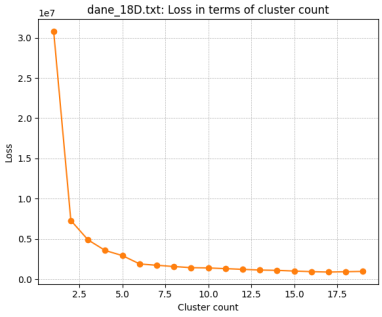
$\sigma = 0.25$, dokładność 0.967

2 Dane 18D i metoda sylwetkowa

Zacznijmy od analizy z użyciem metody łokciowej. Wykresy zależności błędu od liczby klastrów przedstawiają się następująco.

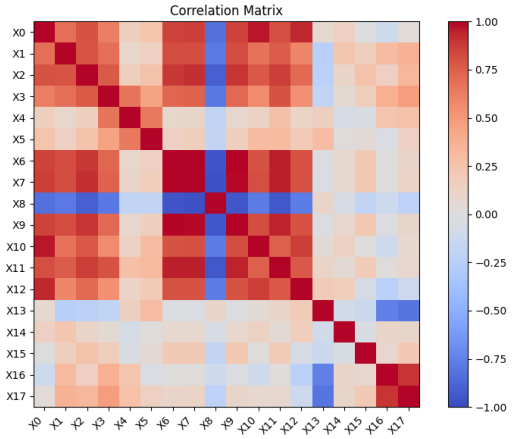


Metryka Manhattan



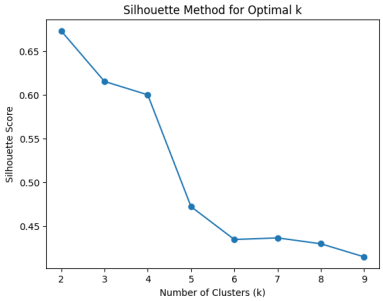
Metryka Euklidesowa

Pierwsze co wywnioskowałem, to że w obydwu przypadkach $k = 6$ jest pewnym punktem przełamania. Jednak postanowiłem posłużyć się również metodą sylwetkową w celu ustalenia optymalnego k. Jednak w związku z dużym rozmiarem danych spróbowałem uprościć trochę wektor cech. Przyjrzyjmy się macierzy korelacji naszych danych

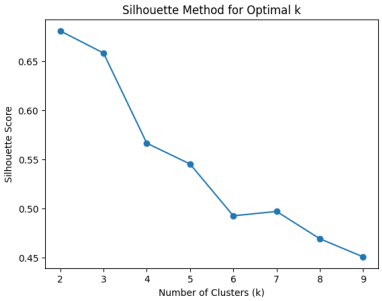


Macierz korelacji

Widzimy że wiele cech jest między sobą silnie zależnych, więc spośród nich wszystkich wybrałem cechy $[0, 4, 5, 6, 13, 14, 15, 16, 17]$. Jednak zdziwił mnie wynik metody sylwetkowej więc algorytm powtórzyłem na zbiorze danych bez zależności.



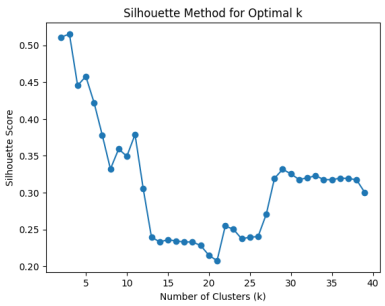
Metoda sylwestkowa dla pełnego zbioru



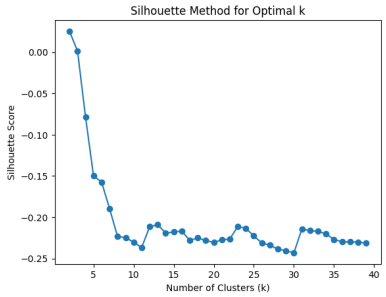
Metoda sylwetkowa dla uproszczonego zbioru

Widzimy że w obydwu przypadkach szczyt wykresu przypada dla $k = 2$. Mimo że wnioski dla obu metod się nie pokrywają, to optymalna liczba klas jest gdzieś pomiędzy 2 i 6.

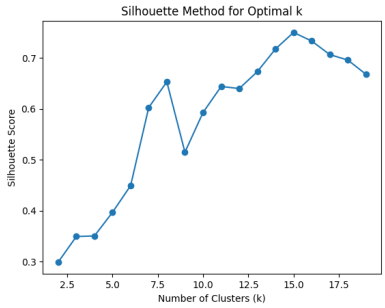
Korzystając z okazji, sprawdźmy jak metoda sylwetkowa poradzi sobie na wyznaczaniu liczby cech dla zestawów 3, 4, 5 i 6. Ze względu na kształt danych, dla zestawu 4 używam łączenia pojedynczego, a dla pozostałych łączenia centroidalnego. Spójrzmy na wykresy dla naszych danych



Metoda sylwestkowa dla zestawu 3



Metoda sylwetkowa dla zestawu 4



Metoda sylwetkowa dla zestawu 6

Widzimy, że dla zestawu 3 maximum występuje dla $k = 3$, dla zestawu 4 dla $k = 2$, a dla zestawu 6 dla $k = 16$. Niestety brakuje tu danych dla zestawu 5, gdyż tą sekcję robiłem bliżej deadline i miałem jeszcze kilka algorytmów do odpalenia, i jak zobaczyłem że po $3h$ jeszcze się nie policzyło to się poddałem.

W tym przypadku metoda sylwetkowa okazała się być bardzo skuteczna (z dokładnością +/- 1), zatem w ostatecznym rozrachunku skłaniałbym się w kierunku 2 klas dla danych 18D. Na koniec porównajmy nasze modele na znanym nam z wcześniejszego projektu pliku.

3 Plik rp.data

W tym rozdziale chciałbym przedstawić wyniki moich modeli na pliku `rp.data`, biorąc liczbę klas $k = 2$.

Model	**Accuracy**
KMeans (Euclidean Metric (E))	0.924
KMeans (Hamming Metric (H))	0.545
KMeans (Manhattan Metric (M))	0.897
Hierarchical (E, Singular)	0.545
Hierarchical (E, Full)	0.800
Hierarchical (E, Mean)	0.892
Hierarchical (E, Centroid)	0.545
Hierarchical (E, Ward)	0.929
Hierarchical (M, Full)	0.837
Hierarchical (M, Mean)	0.887
Hierarchical (M, Ward)	0.932
Spectral (E, $\sigma = 0.0125$)	0.897
Spectral (E, $\sigma = 0.05$)	0.854
Spectral (E, $\sigma = 0.5$)	0.547

Tabela 4: Benchmark modeli na zbiorze rp.data

4 Podsumowanie

Porównując algorytmy, każdy z nich przydaje się do różnych zbiorów danych. Były przypadki gdy algorytm `k-średnich` radził sobie najlepiej, w innych `grupowanie hierarchiczne` czy `klasteryzacja spektralna` wypadały lepiej. Jednak najszybszy z nich zdecydowanie był algorytm `k-średnich`. Różne kombinacje różnych ustawień sprawdzały się inaczej dla każdego ze zbiorów danych. Zaskakująco, algorytm `k-średnich` zdominował przy pliku `rp.data`, pokonywało go nieznacznie grupowanie hierarchiczne dla metryki Manhattan i Łączenia Warda. Dla klasteryzacji spektralnej było jeszcze więcej parametrów, bo nie tylko kontrolowaliśmy końcowy algorytm łączenia ale również hiperparametr σ, który zachowywał się różnie dla różnych zbiorów danych.

Porównując różne metody wyznaczania liczby klastrów, żadna nie wypadła bezkonkurencyjnie. Metoda łokciowa nie była łatwa do rozczytania, za to dosyć szybka bo korzystała z algorytmy k-średnich. Z drugiej strony metoda sylwetkowa często dawała jednoznaczny rezultat, ale z uwagi na wykorzystanie grupowania hierarchicznego była kosztowna obliczeniowo. Bywało też tak że analizując dane obydwoma metodami dochodziłem do różnych wniosków.

Ostatecznie na wszystkich zbiorach osiągnąłem zadowalający wynik, jednak nie było algorytmu który poradziłby sobie przyzwoicie w każdej sytuacji.