Algorytm grupowania danych typu kwantyzacji wektorów

Wstęp

Definicja problemu: Typowe, rozważane dotychczas problemy koncentrowały się na nauczeniu na podstawie zbioru treningowego i zbioru etykiet klasyfikacji wzorców, które należały do jednej z kilku kategorii (np. automatyczne przypisywanie odpowiedniego gatunku kwiatów). Innym typem często spotykanych problemów jest znalezienie w danych treningowych charakterystycznych grup cechujących się pewnymi podobnymi właściwościami, przy czym nie mamy wstępnej informacji o przynależności danego wektora do danej kategorii. Informacje o kategoriach w tym przypadku są niedostępne.

Przykładem zastosowania metod klasteryzacji danych jest problem znajdywania grup podobnych dokumentów zwracanych przez wyszukiwarkę. (np. http://clusty.com/)

Algorytm VQ

Algorytm VQ (Victor quantization) działa podobnie do algorytmu LVQ, w którym pominięto część odpowiadającą za karanie pozostawiając jedynie część nagradzającą (w zależności od etykiety klasy). Takie rozwiązanie powoduje że wektory kodujące podążają za jednorodnymi grupami danych.

Optymalizacja wektorów kodujących odbywa się wg. zależności:

$$p_i = p_i + \alpha\left(x_j - p_i\right) \quad (1)$$

Gdzie

α - współczynnik uczenia

p_i – i-ty wektor kodujący leżący najbliżej wektora treningowego x_j

Wartości α powinny być aktualizowane wg. zależności $\alpha = \frac{\alpha}{1+\alpha}$ po każdej iteracji algorytmu.

Algorytm VQ do działania wymaga zdefiniowania odpowiednich wartości α_0 oraz liczby wektorów kodujących *l_w*.

Opis algorytmu

Uczenie

1. Wylosuj położenie *l_w* neuronów - w tym celu najlepiej jest wykorzystać istniejące już wektory danych i losowo wybrać ze zbioru danych wektory których położenie będzie inicjowało położenie neuronów

2. Iteracyjnie l-razy

    1. Dla każdego wektora treningowego

a. Znajdź najbliższy wektor kodujący (dla danej metryki)
b. Dokonaj aktualizacji położenia (wag) neuronu zgodnie z zależnością (1)

3. Dokonaj aktualizacji wsp. ? wg. zależności (2)

Testowanie (określanie przynależności danego wektora testowego do danej grupy)

1. Ponumeruj wektory kodujące
2. Dla każdego wektora testowego
   a. Policz odległości pomiędzy wektorem testowym a wszystkimi neuronami (wektorami kodującymi)
   b. Znajdź neuron leżący najbliżej wektora testowego (min(odległości(x,y)))
   c. Przypisz numer najbliższego wektora kodującego do danego wektora testowego

Algorytm k-śrtednich (k-means)

Jest wiele odmian algorytmu k-średnich, jego najczęściej spotykana wersja działa na zasadzie wsadowej (batch) tzn. że najpierw tworzona jest tablica/macierz przynależności *U* o wymiarach: *l_w* x *l_n* gdzie *l_w* to liczba klasterów, *l_n* – to liczba wektorów danych. Macierz ta ma postać binarną tzn. przyjmuje wartości 1 jeśli dany wektor danych należy do grupy związanej z danym wektorem kodującym oraz 0 jeśli dany wektor danych nie należy do danej grupy (danego wektora kodującego).

$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccc} p_1 & p_2 & p_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \end{array}$$

Każdy wiersz powyższej macierzy powinien spełniać zależność iż suma elementów wiersza musi być mniejsza od liczby wektorów danych. $\forall_{j\in[1,l\_w]} \sum_{i=1}^{l\_n} U_{j,i} < l\_n$

Takie ograniczenie pozwala zabezpieczyć się aby pojedynczy klaster nie zagarnął wszystkich wektorów danych. Drugie ograniczenie służy zapewnieniu by dany wektor przynależał dokładnie do jednej grupy, a sprowadza się ono do zależności $\forall_{i\in[1,l\_n]} \sum_{j=1}^{l\_w} U_{j,i} = 1$

Algorytm k-średnich działa na zasadzie realizacji dwóch kroków 1) wyznaczenia położenia środków centrów klasterów *P* na podstawie macierzy U 2) aktualizacji macierzy U na podstawie nowych środków centrów *P*

Opis algorytmu

Uczenie

1. Zainicjuj położenie centrów P

2. Dokonaj aktualizacji macierzy U, (przypisz dane do odpowiednich środków klasterów P)
3. Wyznacz położenia środków klasterów - oblicz wartość średnią z wszystkich przypadków należących do danego klastera
4. Porównaj położenia środków centrów klastra z poprzedniej iteracji i obecnej. Jeśli nie uległy zmianie to koniec
5. Idź do 2

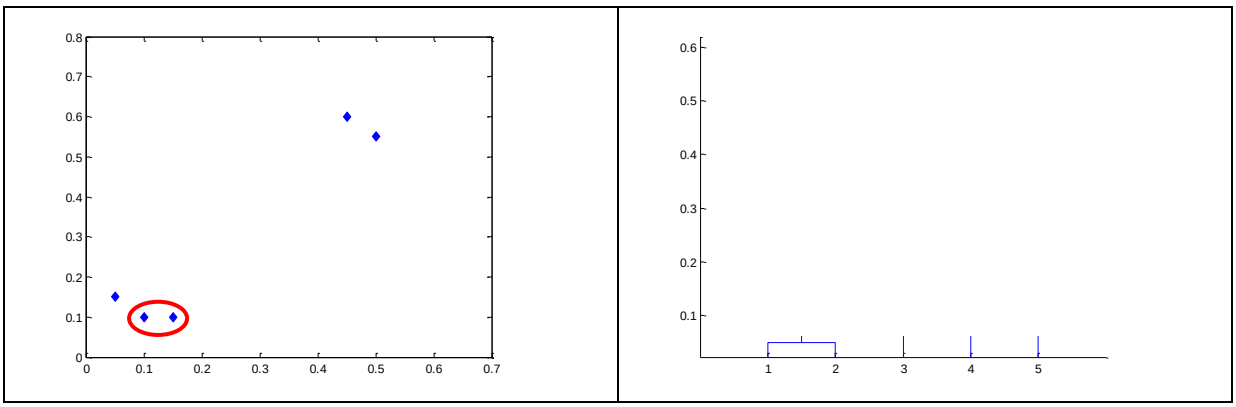
Grupowanie hierarchiczne

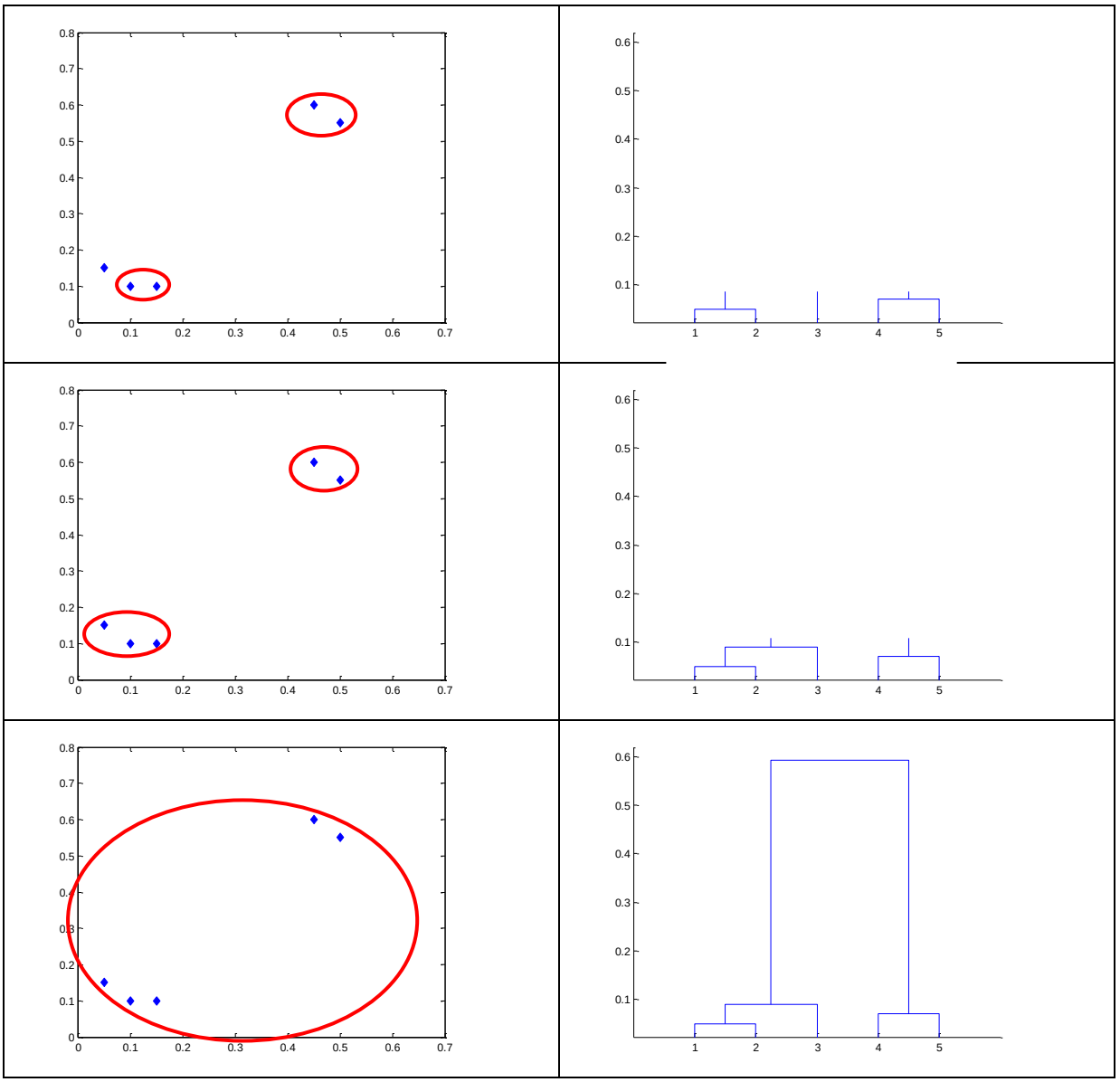
Podstawy teoretyczne

Grupowanie hierarchiczne jest odmienną grupą metod grupowania danych w stosunku do metod bazujących na minimalizacji skalarnego współczynnika jakości jak np. algorytm k-means. Metoda ta bazuje na budowie grafu w postaci drzewa. Algorytm ten jest typem algorytmów z dołu do góry „buttom – top" gdzie zakłada się, iż każdy wektor stanowi oddzielny klaster, a następnie łączy się małe klastry w coraz to większe. Proces łączenia realizowany jest na zasadzie poszukiwania klasterów leżących najbliżej siebie i zastępowania ich nowym większym klastrem, stanowiącym połączenie dwóch poprzednich. Proces ten stopniowo postępuje aż do chwili, w której zostanie osiągnięta właściwa liczba klastrów (określona przez użytkownika) lub do momentu gdy wszystkie wektory znajdują się w jednym klastrze.

Charakterystyczną cechą tego algorytmu jest momzliwość reprezentacji struktury klasteryzacji w postaci drzewa dendrogramu. Dendrogram na osi x posiada etykiety punktów, natomiast na osi y podobieństwo między grupami. Taka reprezentacja wyników klasteryzacji daje szerego możliwości jak np. ocenę liczby klastrów (jeśli wcześniej liczba ta jest nie znana), możliwość analizy pojawienia się wektorów odstających itp.

Przykład procesu grupowania hierarchicznego:





Proces klasteryzacji w tym algorytmie można podzielić na trzy etapy:

1. Liczenie odległości
2. Budowa drzewa na podstawie odległości
3. Przycięcie drzewa na określonym poziomie

Gdzie w drugim etapie jednym z parametrów jest określenie sposobu liczenia podobieństwa pomiędzy poszczególnymi grupami.

Typowymi parametrami są tutaj:

- Uśredniona wartość odległości pomiędzy wektorami w grupach

$$d(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} dist(x_{ri}, x_{sj})$$

- Wyszukiwanie dwóch najodleglejszych obiektów w klastrach

$$d(r,s) = max(dist(x_{ri}, x_{sj})), i \in (1, \ldots, n_r), j \in (1, \ldots, n_s)$$

- Wyszukiwanie dwóch najbardziej podobnych obiektów w klastrach

$$d(r,s) = min(dist(x_{ri}, x_{sj})), i \in (i, \ldots, n_r), j \in (1, \ldots, n_s)$$

- Wyznaczenie centroidy dla każdej grupy I liczenie odległości pomiędzy centroidami

$d(r,s) = \|\bar{x}_r - \bar{x}_s\|_2$ gdzie $\bar{x}_r = \frac{1}{n_r} \sum_{i=1}^{n} x_{ri}$

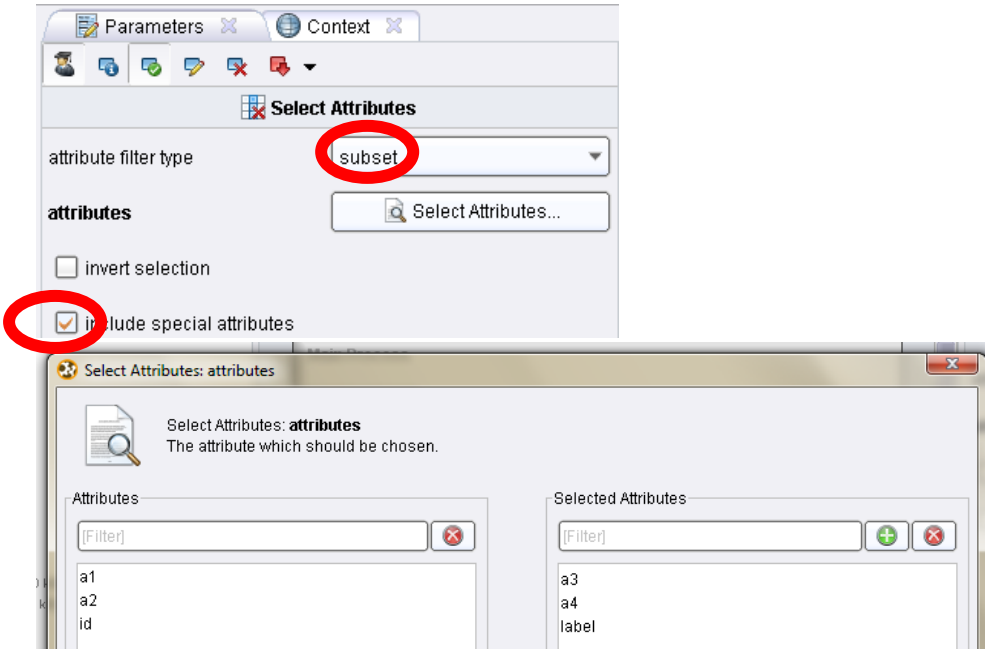
- Wyznaczenie centroidy dla każdej grupy I liczenie odległości pomiędzy centroidami, jednak centoida wyznaczana jest jako średnia centroida z już istniejących centroid (tylko odległośc Euklidesa, duża szybkość)

$d(r,s) = \|\tilde{x}_r - \tilde{x}_s\|_2$ gdzie $\tilde{x}_r$ i $\tilde{x}_s$ są centroidami klasterów *r* i *s* powstałymi z dwóch centroid *p* i *q* na podstawie których powstał dany klaster $\tilde{x}_r = \frac{1}{2}(\tilde{x}_p + \tilde{x}_q)$

Cechą algorytmu grupowania hierarchicznego jest możliwość wpływu na kształt powstałych grup poprzez wybór odpowiedniego typu łączenia grup. Cechy tej nie posiadają inne algorytmy grupowania, w szczególności bazujące na centroidach jak algorytm k-średnich czy VQ .

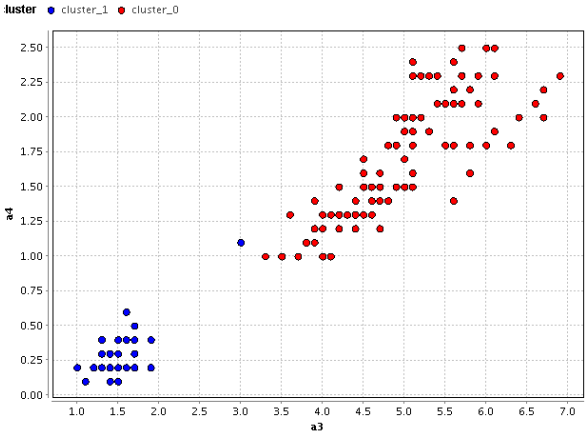
W RapidMinerze

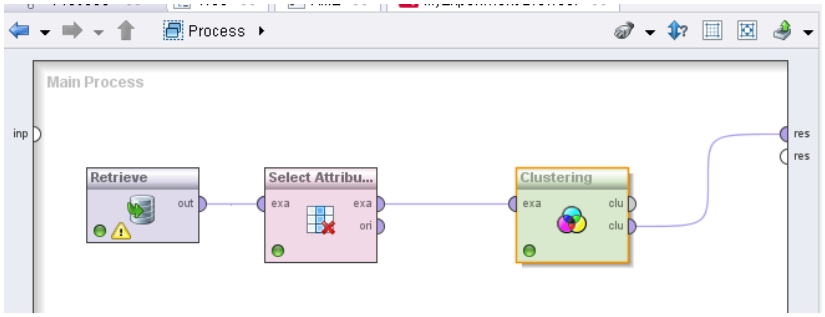
1. Dokonaj wizualnej oceny różnych metod grupowania, w tym celu wczytaj zbiór *Iris*, a następnie korzystając z operatora: Data Transformations -> Attribute Set Reduction and Transformations -> Selection -> Select Attributes dokonaj selekcji jedynie atrybutów z numerem 3 i 4 oraz label. Konfiguracja jak na rys.



Następnie podłącz operator grupowania:
Modeling -> Clustering and Segmentation -> k-Means,

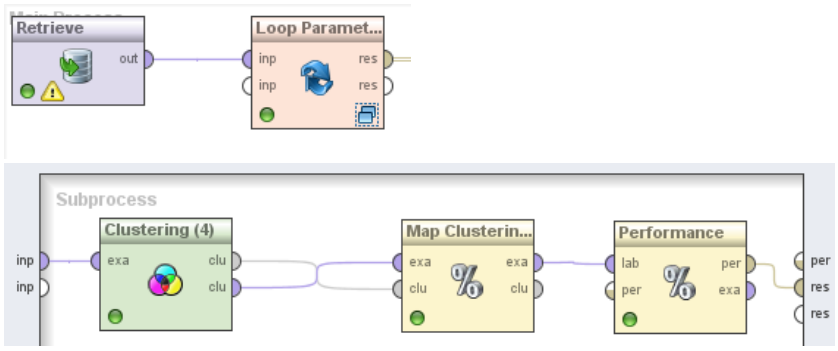
I dokonaj analizy uzyskanych wyników. Zbadaj przy tym wpływ parametru *k* na kształ i zachowanie się procesu klasteryzacji. Schemat oraz przykładowy wykres przedstawia poniższy rysunek:





Zwróć uwagę na nietypowy wynik klasteryzacji dla dwóch klasterów, spróbuj go zinterpretować.

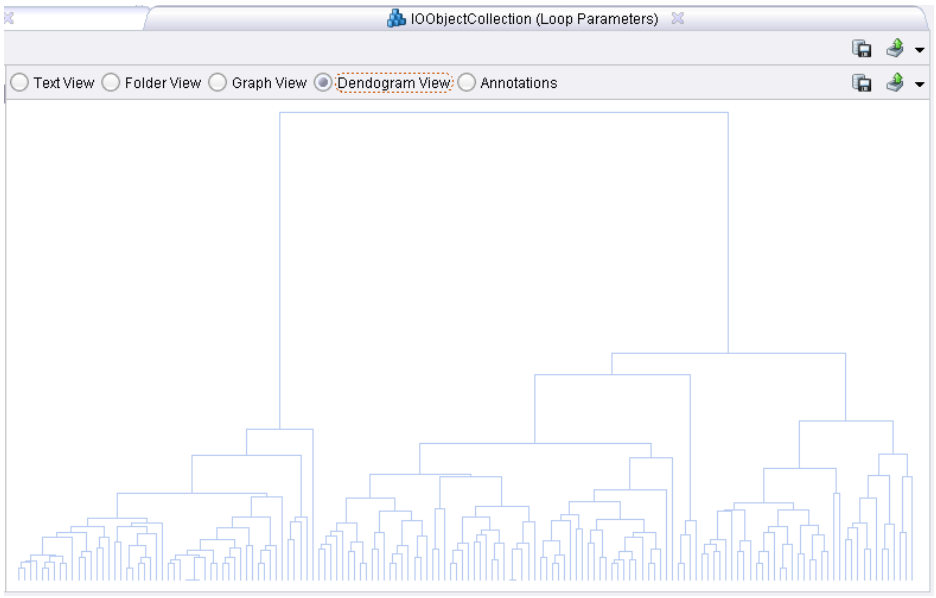
2. Ustal parametr „*max runs*” na 1 a następnie dokonaj analizy stabilności algorytmu. Dla danej wartości parametru k np. k=5 powtórz obliczenia rejestrując wyniki graficzne. Czy uzyskane wyniki są powtarzalne (zawsze takie same)? Uwaga, pamiętaj aby przy kolejnych uruchomieniach zmieniać parametr Radnom_Seed głównego procesu.
3. Zbuduj układ jak na rys:



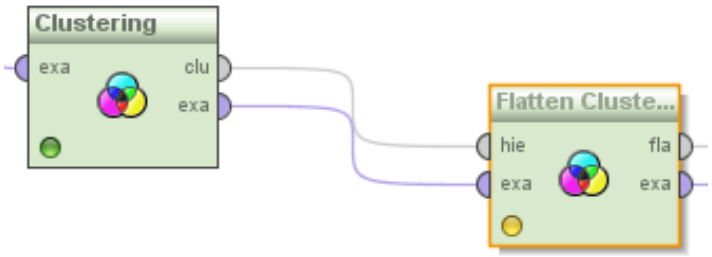
Ustaw parametr „*max runs*” = 1 i dokonaj iteracji po różnych wartościach parametru *seed* algorytmu klasteryzacji. Sprawdź stabilność uzyskanych wyników. (pamiętaj o włączeniu parametru *Use local random seed*)

4. Powtórz obliczenia z 1 i 2 również dla innych operatorów: Prules -> Clustering -> VQ; Prules -> Clustering -> FCM

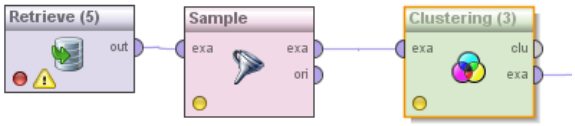
5. Powtórz operacje od 1 do 2 dla algorytmu grupowania hierarchicznego Modeling -> Clustering -> Agglomerative Clustering. Za każdym razem rejestruj jednak zarówno wynik jak i zbudowany dendrogram:



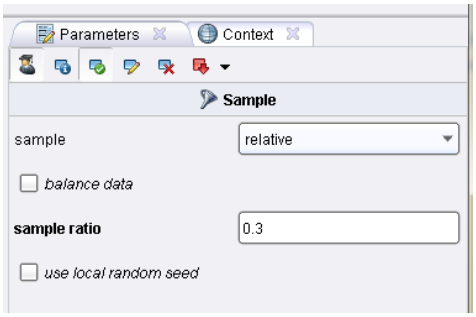
Pamiętaj jednak że operator Agglomerative Clustering dokonuje jedynie budowy dendrogramu, natomiast jego wykorzystanie wymaga odpowiedniego przycięcia dokonywanego za pomocą operatora Modeling -> Clustering -> Flatten Clustering



6. Dla metody grupowania: Modeling -> Clustering -> Agglomerative Clustering dokonaj analizy (1 do 3) dla różnych wartości parametru *mode.* Parametr ten modyfikuje sposób obliczenia podobieństw między grupami punktów.
   CompleteLink - oblicza podobieństwa na zasadzie najbardziej niepodobnych obiektów w klasterach,
   SingleLink – oblicza podobieństwa na zasadzie najbardziej podobnych obiektów w klasterach
   AverageLink – oblicza podobieństwo na zasadzie uśrednionej odległości pomiędzy odległościami w obiektów w klastrach
7. Wczytaj zbiór *spiral* i powtórz obliczenia dla powyższych metod grupowania.
   Zwróć uwagę na zachowanie algorytmu hierarchicznego w zależności od wartości parametru *Mode*
8. Dokonaj klasteryzacji zbioru Diabetes w oparciu o algorytm *k-means* oraz algorytm hierarchiczny. Ustaw liczbę grup na 4. Po wczytaniu zbioru dokonaj przepróbowania jak na rysunku (Operator Sample)



Konfigurując go na wybór wektorów w sposób relatywny:



Dodaj operator logowania i narysuj zależność czasu obliczeń w funkcji rozmiaru zbioru danych poddanych klasteryzacji