Uniwersytet Przyrodniczy we Wrocławiu

Data Mining
Wykład 9

Iteracyjno-optymalizacyjne metody grupowania
Algorytm k-srednich

Metody iteracyjno-optymalizacyjne (1)

- Dane k – ustalona liczba klastrów, iteracyjno-optymalizacyjne metody grupowania tworzą jeden podział zbioru obiektów (partycje) w miejsce hierarchicznej struktury podziałów
- Tworzony jest podział początkowy (zbiór klastrów k), a następnie, stosując technikę iteracyjnej realokacji obiektów pomiędzy klastrami, podział ten jest modyfikowany w taki sposób, aby uzyskać poprawę podziału zbioru obiektów pomiędzy klastry.

Osiągniecie „optimum" globalnego podziału obiektów wymaga przeanalizowania wszystkich możliwych podziałów zbioru n obiektów pomiędzy k klastrów

Metody iteracyjno-optymalizacyjne (2)

- Metody iteracyjno-optymalizacyjne realokują obiekty pomiędzy klastrami optymalizując funkcję kryterialną zdefiniowaną lokalnie (na podzbiorze obiektów) lub globalnie (na całym zbiorze obiektów)
- Przeszukanie całej przestrzeni wszystkich możliwych podziałów zbioru obiektów pomiędzy k klastrów jest, praktycznie, nie realizowalne
- W praktyce, algorytm grupowanie jest uruchamiany kilkakrotnie, dla różnych podziałów początkowych, a następnie, najlepszy z uzyskanych podziałów jest przyjmowany jako wynik procesu grupowania

Metody iteracyjno-optymalizacyjne (3)

- Dowolny problem eksploracji danych mona zdefiniować w kontekście 5 elementów:

Zadanie

Model

Funkcja kryterialna

Metoda przeszukiwania przestrzeni rozwiązań

Algorytmy i struktury danych wspierające proces eksploracji

Metody iteracyjno-optymalizacyjne (4)

- Zadanie

Podział zbioru obiektów D na k rozłącznych zbiorów (klastrów, skupień)

- Najczęściej problem maksymalizacji (lub minimalizacji) funkcji kryterialnej jest problemem nierozstrzygalnym obliczeniowo

Funkcje kryterialne (1)

- **Notacja**: d(x, y) odległość pomiędzy obiektami x, y Î D
- Dwa aspekty grupowania:
  - Klastry powinny być zwarte
  - Klastry powinny być maksymalnie rozłączne
- Odchylenie wewnątrzklasowe - **wc(C)**
- Odchylenie miedzy klastrowe **– bc(C)**
- Średnia klastra (mean) – r_k

$$r_k = \frac{1}{n_k} \sum x \qquad x \in C_k$$

gdzie n_k liczba obiektów należących do k-tego klastra

Funkcje kryterialne (2)

- **Prosta miara wc(C):**

$$wc(C) = \sum_{j=1}^{k} wc(C_j) = \sum_{j=1}^{k} \sum_{x(i) \in C_j} d(x(i), r_j)^2$$

- **Prosta miara bc(C):**

$$bc(C) = \sum_{1 \le i < j \le k} d(r_j, r_j)^2$$

Funkcje kryterialne (3)

- Miarę jakości grupowania C można zdefiniować jako kombinację wc(C) i bc(C), np. jako stosunek bc(C)/wc(C)
- Przyjęcie miary wc(C), jako miary zwartości klastrów, prowadzi do generowania klastrów sferycznych (algorytm k-średnich)
- Dane jest grupowanie C: jak złożony jest proces obliczania wartości wc(C) i bc(C)?
- Obliczenie wc(C) wymaga O(∑i |Ci |) = O(n) operacji
- Obliczenie bc(C) wymaga O(k2) operacji
- Obliczenie wartości funkcji kryterialnej dla pojedynczego grupowania wymaga przejrzenia całego zbioru obiektów D

Funkcje kryterialne (4)

- Inna definicja odchylenia wewnątrz klastrowego (wc(C)):

> Dla każdego obiektu należącego do klastra obliczamy odległość tego punktu do najbliższego obiektu w tym klastrze, i bierzemy max z tych odległości

- Przyjęcie powyższej miary odchylenia wewnątrzklasowego prowadzi do generowania klastrów podłużnych:

$$wc(C) = \max_{i} \min_{y(j) \in C_k} \{ d(x(i), y(j)) \mid x(i) \in C_k, x \neq y \}$$

Grupowanie iteracyjno-optymalizacyjne(1)

- Problem wyboru algorytmu, który optymalizowałby funkcję kryterialną
- W celu znalezienia optimum globalnego należy przejrzeć wszystkie możliwe podziały C obiektów na k klastrów i wybrać ten podział, który optymalizuje funkcję kryterialną
- Liczba możliwych podziałów na klastry wynosi ≈ k^n

Do penetracji przestrzeni rozwiązań można zastosować jedną z wielu technik optymalizacji kombinatorycznej: iterative improvement, tabu search, simulating annealing, genetic algorithms, itp.

Grupowanie iteracyjno-optymalizacyjne(2)

Ogólna idea:

Wybieramy losowo początkowy podział zbioru obiektów na k klastrów, a następnie, stosując technikę iteracyjnej realokacji obiektów pomiędzy klastrami, początkowy podział jest modyfikowany w taki sposób, aby uzyskać poprawę funkcji kryterialnej aż do osiągnięcia warunku stopu – **tzw. algorytm zachłanny**

Przykładem takiego podejścia jest **algorytm k-średnich**

Algorytm k-średnich (1)

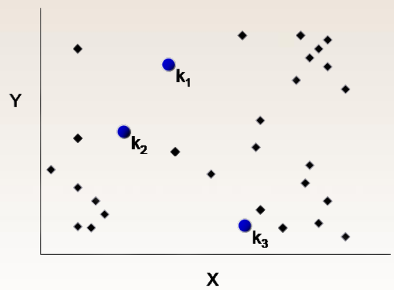
- Dane wejściowe: liczba klastrów k, baza danych n : obiektów
- Dane wyjściowe: zbiór k klastrów minimalizujący kryterium błędu średniokwadratowego

1. Wybierz losowo k obiektów jako początkowe środki k klastrów;
2. **while** występują zmiany przydziału obiektów do klastrów **do**
   - Przydziel każdy obiekt do tego klastra, dla którego odległość obiektu od środka klastra jest najmniejsza;
   - Uaktualnij środki klastrów – środkiem klastra jest wartość średniej danego klastra;

Przykład 1, krok 1

Założenie:
k=3

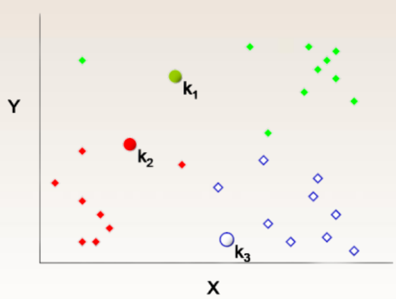
wybierz 3 początkowe środki klastrów (losowo)



Uniwersytet Przyrodniczy we Wrocławiu

Przykład 1, krok 2

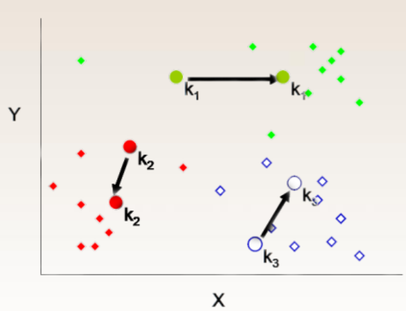
Przydziel każdy obiekt do klastra w oparciu o odległość obiektu od środka klastra



Uniwersytet Przyrodniczy we Wrocławiu

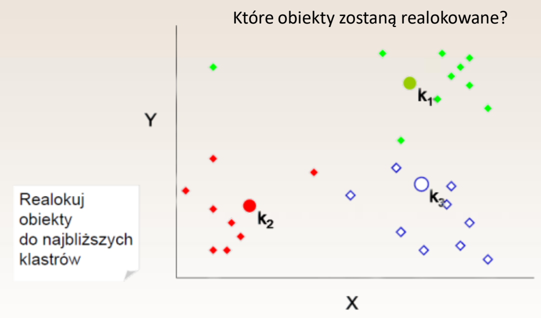
Przykład 1, krok 3

Uaktualnij środki (średnie) wszystkich klastrów

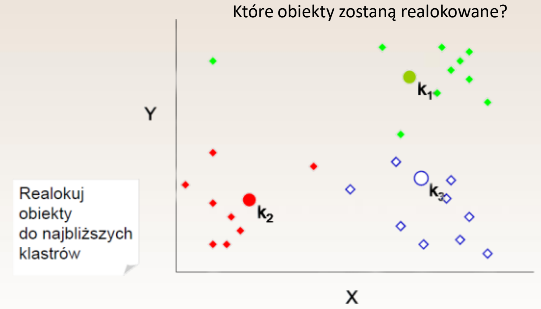


Uniwersytet Przyrodniczy we Wrocławiu

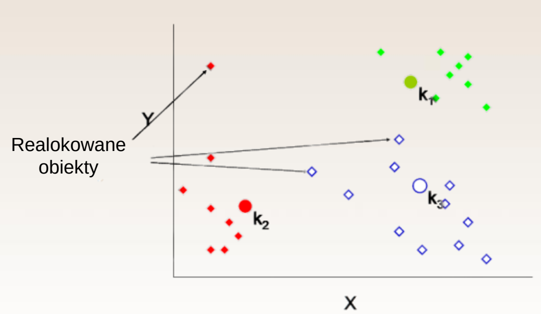
Przykład 1, krok 4



Przykład 1, krok 4

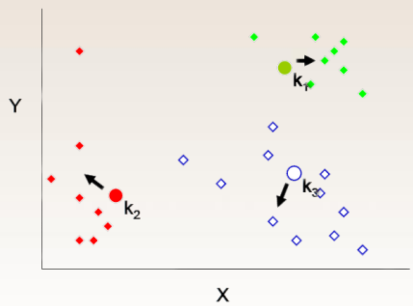


Przykład 1, krok 4a



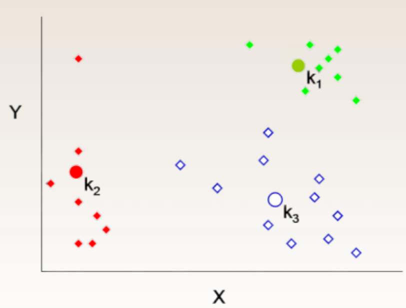
Przykład 1, krok 4b

Oblicz nowe średnie klastrów



Przykład 1, krok 5

Przesuń środki klastrów do nowo obliczonych średnich



Algorytm k-średnich (2)

- Złożoność algorytmu k-średnich wynosi O(knl), gdzie l oznacza liczbę iteracji
- Dla danego zbioru środków klastrów rk , w ramach jednokrotnego przeglądu bazy danych można obliczyć wszystkie k*n odległości d(rk , x) i dla każdego obiektu x wybrać minimalną odległość; obliczenie nowych środków klastrów można wykonać w czasie O(n)

Algorytm bardzo czuły na dane zaszumione lub dane zniekształcenie zawierające punkty osobliwe, gdyż punkty takie w istotny sposób wpływają na średnie klastrów powodując ich

Algorytm k-średnich (3)

- Wynik działania algorytmu (tj. ostateczny podział obiektów pomiędzy klastrami) silnie zależy od początkowego podziału obiektów
- Algorytm może „wpaść” w optimum lokalne
- W celu zwiększenia szansy znalezienia optimum globalnego należy kilkakrotnie uruchomić algorytm dla różnych podziałów początkowych