Algorytmy eksploracji danych: Wykład 12

Wojciech Kempa

Politechnika Śląska
Wydział Matematyki Stosowanej

Algorytm Max-Miner (1)

- **Algorytm Max-Miner** jest algorytmem zaproponowanym w 1998 roku przez R. J. Bayardo do znajdowania maksymalnych zbiorów częstych.

Algorytm Max-Miner (1)

- **Algorytm Max-Miner** jest algorytmem zaproponowanym w 1998 roku przez R. J. Bayardo do znajdowania maksymalnych zbiorów częstych.
- Algorytm ten próbuje „wybiegać w przyszłość”, by szybko zidentyfikować zbiory częste o dużej liczności.

Algorytm Max-Miner (1)

- **Algorytm Max-Miner** jest algorytmem zaproponowanym w 1998 roku przez R. J. Bayardo do znajdowania maksymalnych zbiorów częstych.
- Algorytm ten próbuje „wybiegać w przyszłość”, by szybko zidentyfikować zbiory częste o dużej liczności.
- Dzięki wczesnej identyfikacji tego typu zbiorów można usunąć wszystkie ich podzbiory z rozważanej bazy danych (będą one zbiorami częstymi), by zredukować jej rozmiar.

Algorytm Max-Miner (1)

- **Algorytm Max-Miner** jest algorytmem zaproponowanym w 1998 roku przez R. J. Bayardo do znajdowania maksymalnych zbiorów częstych.
- Algorytm ten próbuje „wybiegać w przyszłość", by szybko zidentyfikować zbiory częste o dużej liczności.
- Dzięki wczesnej identyfikacji tego typu zbiorów można usunąć wszystkie ich podzbiory z rozważanej bazy danych (będą one zbiorami częstymi), by zredukować jej rozmiar.
- Algorytm Max-Miner wykorzystuje zarówno przycinanie podzbiorów jak i nadzbiorów. Dodatkowo używa się w nim również drzewa enumeracji zbiorów (SE-drzewa).

Algorytm Max-Miner (2)

- Każdy węzeł drzewa enumeracji zbiorów nazywamy **grupą kandydującą** (oznaczamy ją literą g).

Algorytm Max-Miner (2)

- Każdy węzeł drzewa enumeracji zbiorów nazywamy **grupą kandydującą** (oznaczamy ją literą g).
- Z grupą kandydującą g związane są dwa zbiory. Pierwszy z nich nazywamy **głową** i oznaczamy przez $h(g)$. Głowa reprezentuje zbiór odpowiadający danemu wierzchołkowi.

Algorytm Max-Miner (2)

- Każdy węzeł drzewa enumeracji zbiorów nazywamy **grupą kandydującą** (oznaczamy ją literą g).
- Z grupą kandydującą g związane są dwa zbiory. Pierwszy z nich nazywamy **głową** i oznaczamy przez $h(g)$. Głowa reprezentuje zbiór odpowiadający danemu wierzchołkowi.
- Drugi ze zbiorów nazywamy **ogonem** i oznaczamy przez $t(g)$. Zbiór $t(g)$ zawiera elementy, które nie znajdują się w zbiorze $h(g)$, ale które mogą pojawić się w każdym następnym podwęźle węzła g.

Algorytm Max-Miner (2)

- Każdy węzeł drzewa enumeracji zbiorów nazywamy **grupą kandydującą** (oznaczamy ją literą g).
- Z grupą kandydującą g związane są dwa zbiory. Pierwszy z nich nazywamy **głową** i oznaczamy przez $h(g)$. Głowa reprezentuje zbiór odpowiadający danemu wierzchołkowi.
- Drugi ze zbiorów nazywamy **ogonem** i oznaczamy przez $t(g)$. Zbiór $t(g)$ zawiera elementy, które nie znajdują się w zbiorze $h(g)$, ale które mogą pojawić się w każdym następnym podwęźle węzła g.
- Aby wyznaczyć wartość miary wsparcia grupy kandydującej g, obliczamy wsparcie zbiorów $h(g)$, $h(g) \cup t(g)$ oraz $h(g) \cup \{i\}$ dla każdego $i \in t(g)$.

Algorytm Max-Miner (2)

- Każdy węzeł drzewa enumeracji zbiorów nazywamy **grupą kandydującą** (oznaczamy ją literą g).
- Z grupą kandydującą g związane są dwa zbiory. Pierwszy z nich nazywamy **głową** i oznaczamy przez $h(g)$. Głowa reprezentuje zbiór odpowiadający danemu wierzchołkowi.
- Drugi ze zbiorów nazywamy **ogonem** i oznaczamy przez $t(g)$. Zbiór $t(g)$ zawiera elementy, które nie znajdują się w zbiorze $h(g)$, ale które mogą pojawić się w każdym następnym podwęźle węzła g.
- Aby wyznaczyć wartość miary wsparcia grupy kandydującej g, obliczamy wsparcie zbiorów $h(g)$, $h(g) \cup t(g)$ oraz $h(g) \cup \{i\}$ dla każdego $i \in t(g)$.
- Wartości miar wsparcia zbiorów $h(g) \cup t(g)$ oraz $h(g) \cup \{i\}$ wykorzystujemy w procesie przycinania drzewa.

Algorytm Max-Miner (3)

- Zbiór $h(g) \cup t(g)$ zawiera każdy element, który znajduje się w dowolnym podwęźle węzła g.

Algorytm Max-Miner (3)

- Zbiór $h(g) \cup t(g)$ zawiera każdy element, który znajduje się w dowolnym podwęźle węzła g.
- W konsekwencji, jeśli g jest zbiorem częstym, to każdy zbiór częsty znaleziony w podwęźle węzła g nie będzie maksymalnym zbiorem częstym.

Algorytm Max-Miner (3)

- Zbiór $h(g) \cup t(g)$ zawiera każdy element, który znajduje się w dowolnym podwęźle węzła g.
- W konsekwencji, jeśli g jest zbiorem częstym, to każdy zbiór częsty znaleziony w podwęźle węzła g nie będzie maksymalnym zbiorem częstym.
- Przycinanie wykorzystujące nadzbiory można zaimplementować w algorytmie poprzez zatrzymanie rozwijania podwęzła w dowolnej grupie kandydującej g, dla której zbiór $h(g) \cup t(g)$ jest zbiorem częstym.

Algorytm Max-Miner (3)

- Zbiór $h(g) \cup t(g)$ zawiera każdy element, który znajduje się w dowolnym podwęźle węzła g.
- W konsekwencji, jeśli g jest zbiorem częstym, to każdy zbiór częsty znaleziony w podwęźle węzła g nie będzie maksymalnym zbiorem częstym.
- Przycinanie wykorzystujące nadzbiory można zaimplementować w algorytmie poprzez zatrzymanie rozwijania podwęzła w dowolnej grupie kandydującej g, dla której zbiór $h(g) \cup t(g)$ jest zbiorem częstym.
- Jeśli zbiór $h(g) \cup \{i\}$ nie jest zbiorem częstym, to również każda głowa podwęzła, który zawiera element i nie będzie zbiorem częstym.

Algorytm Max-Miner (3)

- Zbiór $h(g) \cup t(g)$ zawiera każdy element, który znajduje się w dowolnym podwęźle węzła g.
- W konsekwencji, jeśli g jest zbiorem częstym, to każdy zbiór częsty znaleziony w podwęźle węzła g nie będzie maksymalnym zbiorem częstym.
- Przycinanie wykorzystujące nadzbiory można zaimplementować w algorytmie poprzez zatrzymanie rozwijania podwęzła w dowolnej grupie kandydującej g, dla której zbiór $h(g) \cup t(g)$ jest zbiorem częstym.
- Jeśli zbiór $h(g) \cup \{i\}$ nie jest zbiorem częstym, to również każda głowa podwęzła, który zawiera element i nie będzie zbiorem częstym.
- Przycinanie podzbiorów polega zatem na usuwaniu dowolnego takiego elementu końcowego i z grupy kandydującej g przed rozwinięciem jej podwęzłów.

Algorytm Max-Miner (4)

Algorytm Max-Miner ma następujący przebieg.

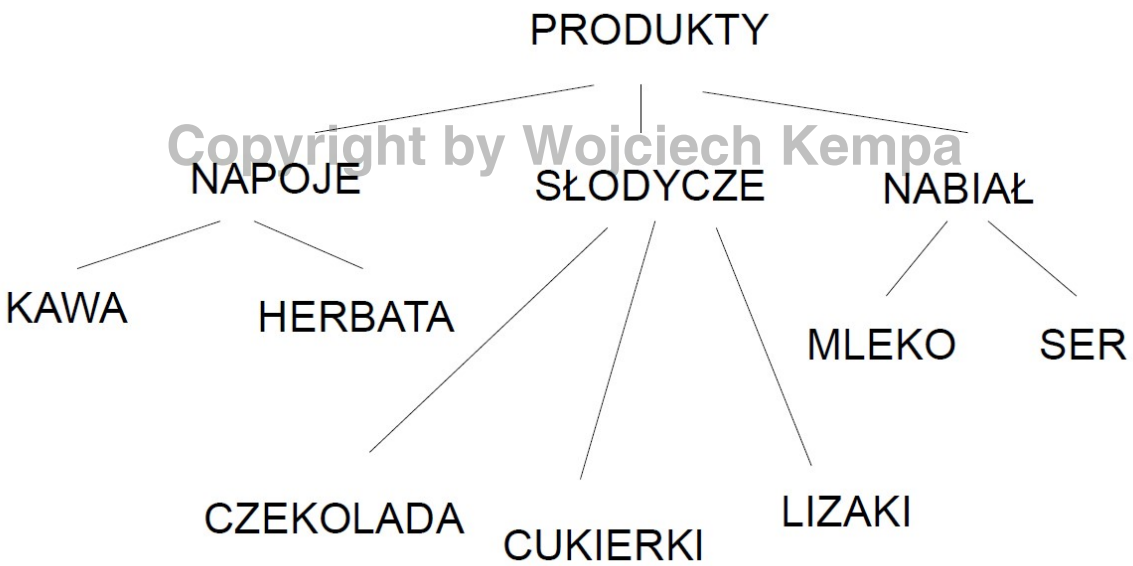
1. Znajdujemy 1-elementowe zbiory częste w bazie danych D. Zbiory te generują pierwsze grupy kandydujące.
2. Dla ustalonego węzła (grupy kandydującej) g obliczamy wsparcie zbioru $h(g) \cup t(g)$.
3. Jeżeli $supp(h(g) \cup t(g)) \geq min\, supp$, to nie rozwijamy drzewa z węzła g. Zbiór $h(g) \cup t(g)$ jest maksymalnym zbiorem częstym.
4. Jeżeli $supp(h(g) \cup t(g)) < min\, supp$, to dla każdego $i \in t(g)$ sprawdzamy, czy zbiór $h(g) \cup \{i\}$ jest zbiorem częstym.

Algorytm Max-Miner (5)

5. Jeżeli zbiór $h(g) \cup \{i\}$ nie jest zbiorem częstym, wówczas usuwamy element i z $t(g)$ i nie bierzemy go pod uwagę w tworzeniu kolejnych podwęzłów.
6. W przypadku gdy zbiór $h(g) \cup \{i\}$ jest zbiorem częstym, generujemy odpowiedni podwęzeł itd.
7. Powyższą procedurę przeprowadzamy dla każdego węzła (grupy kandydującej) utworzonego z 1-elementowego zbioru częstego bazy danych D.

Wielopoziomowe reguły asocjacyjne (1)

Proces odkrywania **wielopoziomowych reguł asocjacyjnych** wymaga zdefiniowania pojęcia taksonomii. **Taksonomia** $\mathcal{T}$ jest rodzajem logicznego uporządkowania zbioru produktów z wyszczególnieniem poziomu ich abstrakcji. Przykładową taksonomię prezentuje Rysunek 3.



Rysunek: 3: Przykładowa taksonomia $\mathcal{T}$

Wielopoziomowe reguły asocjacyjne (2)

- Mówimy, że transakcja T **wspiera element** x, jeżeli $x \in \mathcal{T}$ oraz x jest poprzednikiem dowolnego elementu $y \in T$ w taksonomii $\mathcal{T}$ (w taksonomii zilustrowanej na Rysunku 3 np. poprzednikiem czekolady są słodycze, czyli element taksonomii o wyższym poziomie abstrakcji, ogólniejszy).

Wielopoziomowe reguły asocjacyjne (2)

- Mówimy, że transakcja T **wspiera element** x, jeżeli $x \in \mathcal{T}$ oraz x jest poprzednikiem dowolnego elementu $y \in T$ w taksonomii $\mathcal{T}$ (w taksonomii zilustrowanej na Rysunku 3 np. poprzednikiem czekolady są słodycze, czyli element taksonomii o wyższym poziomie abstrakcji, ogólniejszy).
- Jeżeli L jest zbiorem rozpatrywanych produktów, to wielopoziomową regułę asocjacyjną definiujemy teraz jako relację $X \longrightarrow Y$ gdzie $X, Y \in L$, $X \cap Y = 0$ oraz, dodatkowo, żaden element $y \in Y$ nie jest poprzednikiem żadnego elementu $x \in X$.

Wielopoziomowe reguły asocjacyjne (2)

- Mówimy, że transakcja T **wspiera element** x, jeżeli $x \in \mathcal{T}$ oraz x jest poprzednikiem dowolnego elementu $y \in T$ w taksonomii $\mathcal{T}$ (w taksonomii zilustrowanej na Rysunku 3 np. poprzednikiem czekolady są słodycze, czyli element taksonomii o wyższym poziomie abstrakcji, ogólniejszy).
- Jeżeli L jest zbiorem rozpatrywanych produktów, to wielopoziomową regułę asocjacyjną definiujemy teraz jako relację $X \longrightarrow Y$ gdzie $X, Y \in L$, $X \cap Y = 0$ oraz, dodatkowo, żaden element $y \in Y$ nie jest poprzednikiem żadnego elementu $x \in X$.
- Podobnie jak w przypadku reguł jednopoziomowych, przyjmuje się próg minimalnego wsparcia $min\,supp$ oraz próg minimalnej ufności $min\,conf$, które są jednakowe dla wszystkich reguł, niezależnie od tego, czy opisują one asocjacje na najniższym poziomie abstrakcji czy też na najwyższym poziomie.

Miary atrakcyjności reguł asocjacyjnych (1)

- W analizie asocjacji wprowadza się dwie podstawowe **miary atrakcyjności reguł asocjacyjnych**.

Miary atrakcyjności reguł asocjacyjnych (1)

- W analizie asocjacji wprowadza się dwie podstawowe **miary atrakcyjności reguł asocjacyjnych**.
- Weźmy pod uwagę regułę $A \longrightarrow B$. Miarę *interest*, określającą poziom współzależności obiektów (produktów) A i B, definiujemy w następujący sposób:

$$interest(A \longrightarrow B) = interest(B \longrightarrow A) = \frac{\mathbf{P}(AB)}{\mathbf{P}(A)\mathbf{P}(B)},$$

gdzie odpowiednie prawdopodobieństwa szacuje się na podstawie częstości względnych.

Miary atrakcyjności reguł asocjacyjnych (1)

- W analizie asocjacji wprowadza się dwie podstawowe **miary atrakcyjności reguł asocjacyjnych**.
- Weźmy pod uwagę regułę $A \longrightarrow B$. Miarę *interest*, określającą poziom współzależności obiektów (produktów) A i B, definiujemy w następujący sposób:

$$interest(A \longrightarrow B) = interest(B \longrightarrow A) = \frac{\mathbf{P}(AB)}{\mathbf{P}(A)\mathbf{P}(B)},$$

gdzie odpowiednie prawdopodobieństwa szacuje się na podstawie częstości względnych.

- Jeżeli dla danej reguły miara ta przyjmuje wartość równą 1, to mówimy, że A i B są niezależne. W przypadku wartości mniejszych od 1 mówimy o **korelacji negatywnej**, a jeżeli $interest > 1$ – o **korelacji pozytywnej**.

Miary atrakcyjności reguł asocjacyjnych (2)

- Alternatywą dla miary *interest* jest miara *lift*. Dla binarnych reguł asocjacyjnych jest ona równoważna mierze *interest*.

Miary atrakcyjności reguł asocjacyjnych (2)

- Alternatywą dla miary *interest* jest miara *lift*. Dla binarnych reguł asocjacyjnych jest ona równoważna mierze *interest*.
- Definiujemy ją w następujący sposób:

$$lift(A \longrightarrow B) = \frac{conf(A \longrightarrow B)}{supp(B)}.$$

Miary atrakcyjności reguł asocjacyjnych (2)

- Alternatywą dla miary *interest* jest miara *lift*. Dla binarnych reguł asocjacyjnych jest ona równoważna mierze *interest*.
- Definiujemy ją w następujący sposób:

$$lift(A \longrightarrow B) = \frac{conf(A \longrightarrow B)}{supp(B)}.$$

- W praktyce w ocenie reguł asocjacyjnych stosuje się także alternatywną postać współczynnika korelacji liniowej Pearsona, mianowicie

$$\varrho = \frac{\mathbf{P}(AB) - \mathbf{P}(A)\mathbf{P}(B)}{\sqrt{\mathbf{P}(A)\mathbf{P}(B)\mathbf{P}(\sim A)\mathbf{P}(\sim B)}},$$

gdzie odpowiednie prawdopodobieństwa szacuje się na podstawie częstości względnych.

Odkrywanie wzorców sekwencji (1)

- Istotą **analizy sekwencji** jest wykrywanie pewnych, powtarzających się często, specyficznych zachowań klientów. Jeśli wyjściowe dane miałyby charakter ilościowy, analiza sekwencji „wchodziłaby" w obszar zainteresowań analizy szeregów czasowych.

Odkrywanie wzorców sekwencji (1)

- Istotą **analizy sekwencji** jest wykrywanie pewnych, powtarzających się często, specyficznych zachowań klientów. Jeśli wyjściowe dane miałyby charakter ilościowy, analiza sekwencji „wchodziłaby" w obszar zainteresowań analizy szeregów czasowych.
- Wyjściową bazę danych tworzą jednak **ciągi transakcji**, a zatem zdarzenia związane z konkretnymi decyzjami klientów analizowane w pewnym okresie czasu.

Odkrywanie wzorców sekwencji (1)

- Istotą **analizy sekwencji** jest wykrywanie pewnych, powtarzających się często, specyficznych zachowań klientów. Jeśli wyjściowe dane miałyby charakter ilościowy, analiza sekwencji „wchodziłaby" w obszar zainteresowań analizy szeregów czasowych.
- Wyjściową bazę danych tworzą jednak **ciągi transakcji**, a zatem zdarzenia związane z konkretnymi decyzjami klientów analizowane w pewnym okresie czasu.
- Efektywna analiza sekwencji pozwala na skuteczną predykcję zachowań klientów w przyszłości, a tym samym jest pomocna w uzyskaniu odpowiedzi na wiele pytań kluczowych z punktu widzenia marketingowego.

Odkrywanie wzorców sekwencji (1)

- Istotą **analizy sekwencji** jest wykrywanie pewnych, powtarzających się często, specyficznych zachowań klientów. Jeśli wyjściowe dane miałyby charakter ilościowy, analiza sekwencji „wchodziłaby" w obszar zainteresowań analizy szeregów czasowych.
- Wyjściową bazę danych tworzą jednak **ciągi transakcji**, a zatem zdarzenia związane z konkretnymi decyzjami klientów analizowane w pewnym okresie czasu.

- Efektywna analiza sekwencji pozwala na skuteczną predykcję zachowań klientów w przyszłości, a tym samym jest pomocna w uzyskaniu odpowiedzi na wiele pytań kluczowych z punktu widzenia marketingowego.
- Na przykład: z jakim prawdopodobieństwem klient, który kupił artykuł A, a po jakimś czasie artykuł B, będzie skłonny nabyć w najbliższej przyszłości artykuł C?

Odkrywanie wzorców sekwencji (2)

- Analiza sekwencji znajduje zastosowanie np. w przewidywaniu występowania kataklizmów i klęsk żywiołowych (jaka sekwencja zdarzeń do nich prowadzi), w medycynie (w analizie poszczególnych jednostek chorobowych poprzez obserwację ciągu następujących po sobie objawów, w badaniu efektywności terapii farmakologicznej, analizie skutków ubocznych stosowania danego leku), na rynku usług telekomunikacyjnych (sekwencja zdarzeń prowadzących do zmiany operatora) itp.

Odkrywanie wzorców sekwencji (2)

- Analiza sekwencji znajduje zastosowanie np. w przewidywaniu występowania kataklizmów i klęsk żywiołowych (jaka sekwencja zdarzeń do nich prowadzi), w medycynie (w analizie poszczególnych jednostek chorobowych poprzez obserwację ciągu następujących po sobie objawów, w badaniu efektywności terapii farmakologicznej, analizie skutków ubocznych stosowania danego leku), na rynku usług telekomunikacyjnych (sekwencja zdarzeń prowadzących do zmiany operatora) itp.
- Niech $L = \{l_1, ..., l_m\}$ będzie pewnym zbiorem elementów.

Odkrywanie wzorców sekwencji (2)

- Analiza sekwencji znajduje zastosowanie np. w przewidywaniu występowania kataklizmów i klęsk żywiołowych (jaka sekwencja zdarzeń do nich prowadzi), w medycynie (w analizie poszczególnych jednostek chorobowych poprzez obserwację ciągu następujących po sobie objawów, w badaniu efektywności terapii farmakologicznej, analizie skutków ubocznych stosowania danego leku), na rynku usług telekomunikacyjnych (sekwencja zdarzeń prowadzących do zmiany operatora) itp.

- Niech $L = \{l_1, ..., l_m\}$ będzie pewnym zbiorem elementów.
- **Sekwencją** S nazywamy uporządkowaną listę (ciąg transakcji) $S = (T_1, ..., T_n)$, gdzie $T_i \subseteq L$, $T_i \neq \emptyset$.

Odkrywanie wzorców sekwencji (2)

- Analiza sekwencji znajduje zastosowanie np. w przewidywaniu występowania kataklizmów i klęsk żywiołowych (jaka sekwencja zdarzeń do nich prowadzi), w medycynie (w analizie poszczególnych jednostek chorobowych poprzez obserwację ciągu następujących po sobie objawów, w badaniu efektywności terapii farmakologicznej, analizie skutków ubocznych stosowania danego leku), na rynku usług telekomunikacyjnych (sekwencja zdarzeń prowadzących do zmiany operatora) itp.

- Niech $L = \{l_1, ..., l_m\}$ będzie pewnym zbiorem elementów.
- **Sekwencją** S nazywamy uporządkowaną listę (ciąg transakcji) $S = (T_1, ..., T_n)$, gdzie $T_i \subseteq L$, $T_i \neq \emptyset$.
- Zbiór $T_i = \{x_1, ..., x_k\}$, gdzie $x_j \in L$, $1 \leq j \leq k$, nazywamy i**-tym wyrazem sekwencji**.

Odkrywanie wzorców sekwencji (2)

- Analiza sekwencji znajduje zastosowanie np. w przewidywaniu występowania kataklizmów i klęsk żywiołowych (jaka sekwencja zdarzeń do nich prowadzi), w medycynie (w analizie poszczególnych jednostek chorobowych poprzez obserwację ciągu następujących po sobie objawów, w badaniu efektywności terapii farmakologicznej, analizie skutków ubocznych stosowania danego leku), na rynku usług telekomunikacyjnych (sekwencja zdarzeń prowadzących do zmiany operatora) itp.

- Niech $L = \{l_1, ..., l_m\}$ będzie pewnym zbiorem elementów.
- **Sekwencją** S nazywamy uporządkowaną listę (ciąg transakcji) $S = (T_1, ..., T_n)$, gdzie $T_i \subseteq L$, $T_i \neq \emptyset$.
- Zbiór $T_i = \{x_1, ..., x_k\}$, gdzie $x_j \in L$, $1 \leq j \leq k$, nazywamy i-**tym wyrazem sekwencji**.
- Dowolny element x_j występuje tylko raz w pojedynczym wyrazie, ale może wystąpić w wielu wyrazach.

Odkrywanie wzorców sekwencji (3)

- Mówimy, że wyraz T **wspiera** $x \in L$, jeżeli $x \in T$.

Odkrywanie wzorców sekwencji (3)

- Mówimy, że wyraz T **wspiera** $x \in L$, jeżeli $x \in T$.
- Mówimy, że wyraz T **wspiera zbiór** $X \subseteq L$, jeżeli wspiera każdy element tego zbioru.

Odkrywanie wzorców sekwencji (3)

- Mówimy, że wyraz T **wspiera** $x \in L$, jeżeli $x \in T$.
- Mówimy, że wyraz T **wspiera zbiór** $X \subseteq L$, jeżeli wspiera każdy element tego zbioru.
- Liczbę wyrazów sekwencji S nazywamy **długością sekwencji**. Sekwencję o długości k nazywamy k-**sekwencją**.

Odkrywanie wzorców sekwencji (3)

- Mówimy, że wyraz T **wspiera** $x \in L$, jeżeli $x \in T$.
- Mówimy, że wyraz T **wspiera zbiór** $X \subseteq L$, jeżeli wspiera każdy element tego zbioru.
- Liczbę wyrazów sekwencji S nazywamy **długością sekwencji**. Sekwencję o długości k nazywamy k-**sekwencją**.
- **Rozmiarem sekwencji** S nazywamy liczbę wystąpień pojedynczych elementów x_j w tej sekwencji.

Odkrywanie wzorców sekwencji (3)

- Mówimy, że wyraz T **wspiera** $x \in L$, jeżeli $x \in T$.
- Mówimy, że wyraz T **wspiera zbiór** $X \subseteq L$, jeżeli wspiera każdy element tego zbioru.
- Liczbę wyrazów sekwencji S nazywamy **długością sekwencji**. Sekwencję o długości k nazywamy k-**sekwencją**.
- **Rozmiarem sekwencji** S nazywamy liczbę wystąpień pojedynczych elementów x_j w tej sekwencji.
- Na przykład, rozmiar sekwencji

$$\langle (2)(4,5)(7,8,9)(3,6,7) \rangle$$

wynosi 9, a jej długość jest równa 4.

Odkrywanie wzorców sekwencji (4)

- Mówimy, że sekwencja $\langle X_1, ..., X_n \rangle$ zawiera się w sekwencji $\langle Y_1, ..., Y_m \rangle$, jeżeli

$$\exists_{i_1 < i_2 < ... < i_n} : X_i \subseteq Y_{i_1}, X_2 \subseteq Y_{i_2}, ..., X_n \subseteq Y_{i_n}.$$

Odkrywanie wzorców sekwencji (4)

- Mówimy, że sekwencja $\langle X_1, ..., X_n \rangle$ zawiera się w sekwencji $\langle Y_1, ..., Y_m \rangle$, jeżeli

$$\exists_{i_1 < i_2 < ... < i_n} : X_i \subseteq Y_{i_1}, X_2 \subseteq Y_{i_2}, ..., X_n \subseteq Y_{i_n}.$$

- Na przykład, sekwencja $S = \langle (4)(6,7)(9) \rangle$ zawiera się w sekwencji $Q = \langle (3,4)(4,7,8)(6,7,8,9)(2,9)(5,6) \rangle$.

Odkrywanie wzorców sekwencji (5)

- Mówimy, że S jest sekwencją **maksymalną**, jeżeli nie zawiera się w żadnej innej sekwencji.

Odkrywanie wzorców sekwencji (5)

- Mówimy, że S jest sekwencją **maksymalną**, jeżeli nie zawiera się w żadnej innej sekwencji.
- Zbiór sekwencji $D_S = (S_1, ..., S_n)$ nazywamy **bazą danych sekwencji**.

Odkrywanie wzorców sekwencji (5)

- Mówimy, że S jest sekwencją **maksymalną**, jeżeli nie zawiera się w żadnej innej sekwencji.
- Zbiór sekwencji $D_S = (S_1, ..., S_n)$ nazywamy **bazą danych sekwencji**.
- Mówimy, że $S \in D_S$ **wspiera** sekwencję Q, jeżeli Q zawiera się w S.

Odkrywanie wzorców sekwencji (5)

- Mówimy, że S jest sekwencją **maksymalną**, jeżeli nie zawiera się w żadnej innej sekwencji.
- Zbiór sekwencji $D_S = (S_1, ..., S_n)$ nazywamy **bazą danych sekwencji**.
- Mówimy, że $S \in D_S$ **wspiera** sekwencję Q, jeżeli Q zawiera się w S.

- Definiujemy **wsparcie sekwencji** S w bazie danych D_S w następujący sposób:

$$supp(S) = \frac{|\{S_i \in D_S : S_i \text{ wspiera } S\}|}{|D_S|}.$$

Odkrywanie wzorców sekwencji (6)

- Zasadniczym celem procesu analizy sekwencji jest znalezienie wszystkich sekwencji, których wsparcie w bazie D_S jest co najmniej równe pewnej minimalnej wartości wsparcia $min\,supp$.

Odkrywanie wzorców sekwencji (6)

- Zasadniczym celem procesu analizy sekwencji jest znalezienie wszystkich sekwencji, których wsparcie w bazie D_S jest co najmniej równe pewnej minimalnej wartości wsparcia $min\,supp$.
- Takie sekwencje nazywamy **sekwencjami częstymi** lub **wzorcami sekwencji**.

Odkrywanie wzorców sekwencji (6)

- Zasadniczym celem procesu analizy sekwencji jest znalezienie wszystkich sekwencji, których wsparcie w bazie D_S jest co najmniej równe pewnej minimalnej wartości wsparcia $min\,supp$.
- Takie sekwencje nazywamy **sekwencjami częstymi** lub **wzorcami sekwencji**.
- Sekwencje częste, które równocześnie są sekwencjami maksymalnymi nazywamy **maksymalnymi wzorcami sekwencji**.

Podstawowy algorytm odkrywania wzorców sekwencji

- Podstawowy algorytm odkrywania wzorców sekwencji, **algorytm AprioriAll**, sformułowany w 1995 roku, jest rozszerzeniem algorytmu Apriori, wykorzystywanego w analizie asocjacji i również wykorzystuje własność antymonotoniczności miary wsparcia.

Podstawowy algorytm odkrywania wzorców sekwencji

- Podstawowy algorytm odkrywania wzorców sekwencji, **algorytm AprioriAll**, sformułowany w 1995 roku, jest rozszerzeniem algorytmu Apriori, wykorzystywanego w analizie asocjacji i również wykorzystuje własność antymonotoniczności miary wsparcia.
- Rzeczywiście, jeżeli sekwencja X nie jest wzorcem sekwencji (nie jest sekwencją częstą), to żadna sekwencja Y zawierająca X nie może być wzorcem sekwencji. Mamy bowiem

$$X \subseteq Y \implies supp(Y) \leq supp(X).$$

Ogólny schemat odkrywania wzorców sekwencji

Ogólny schemat odkrywania wzorców sekwencji wygląda w następujący sposób:

1. **sortowanie** – sortujemy bazę danych D do postaci bazy danych sekwencji D_S;
2. **znajdowanie zbiorów częstych** – znajdujemy wszystkie zbiory częste w bazie D_S;
3. **transformacja** – przyporządkowujemy każdemu wyrazowi T należącemu do danej sekwencji listę zbiorów częstych zawierających się w tym wyrazie;
4. **sekwencjonowanie** – stosując algorytm AprioriAll, znajdujemy w D_S sekwencje częste (wzorce sekwencji);
5. **maksymalizacja** – w zbiorze znalezionych wzorców sekwencji znajdujemy maksymalne wzorce sekwencji (jest to krok opcjonalny).

Algorytm AprioriAll

W algorytmie AprioriAll (podobnie, jak to było w algorytmie Apriori) generujemy k-sekwencje kandydujące na podstawie $(k-1)$-sekwencji częstych. Ma to miejsce w dwóch etapach.

- W **etapie połączenia** łączymy dwie sekwencje ze zbioru LS_{k-1} (czyli zbioru $(k-1)$-sekwencji częstych) w celu wygenerowania k-sekwencji kandydującej (elementu zbioru CS_k). Warunkiem połączenia dwóch sekwencji z LS_{k-1} jest zgodność $k-2$ wyrazów obu sekwencji.
- W **etapie eliminacji** usuwamy wszystkie sekwencje kandydujące $S \in CS_k$ takie, dla których istnieją $(k-1)$-podsekwencje sekwencji S nienależące do LS_{k-1}.