Piotr Dubiel

Projekt MED-P3 - algorytm CHARM

Metody eksploracji danych w odkrywaniu wiedzy.

Opis zadania

Celem projektu jest zaimplementowanie algorytmu wyznaczania reguł decyzyjnych o minimalnych poprzednikach, które są częstymi zbiorami zamkniętymi. Do wyznaczania zbiorów zamkniętych używany jest zmodyfikowany algorytm CHARM.

Założenia

Funkcjonalne

- Dane wejściowe są odczytywane z pliku
- Dane wyjściowe są zapisywane w dwóch formatach: surowe dane w pliku csv oraz w sformatowanym pliku txt
- Aplikacja może używać 3 strategii sortowania identyfikatorów: rosnącą, malejącą i nieuporządkowaną.
- Aplikacji można przekazać minimalne wsparcie reguł (sup(r) >= min_sup)

Niefunkcjonalne

- Język programowania: C++.
- Obsługiwane systemy operacyjne: Linux, Windows, OSX.
- Aplikacja konsolowa.

Dane wejściowe

- Plik z danymi w formacie używanym w zbiorze *car.* Każda transakcja jest oddzielona znakiem nowej linii, a atrybuty oddzielone są przecinkami
    - Przykładowy plik:

,,H,,P
,M,H,,P
,,,,P
,,,T,P
S,,,,P
,M,,,P
S,M,,T,P

- (Opcjonalne) Plik z nagłówkiem zawierający nazwy atrybutów w formacie używanym w zbiorze *car.*
    - Przykładowy plik:

      unacc, acc, good, vgood
      buying: vhigh, high, med, low.
      maint: vhigh, high, med, low.
      doors: 2, 3, 4, 5more.
      persons: 2, 4, more.
      lug_boot: small, med, big.
      safety: low, med, high.

      Nazwa atrybutu to ciąg znaków do znaku dwukropka. Jeśli nie ma takiego znaku w linii, to do zbioru wartości przypisywana jest nazwa “decision”.

Parametry wywołania aplikacji

charm [opcje] PLIK_DANYCH NAZWA_PLIKU_WYNIKOWEGO

Opcje

- -a, --asc Sortuj identyfikatory według rosnącego wsparcia.
- -d, --desc Sortuj identyfikatory według malejącego wsparcia.
- -h, --header PLIK Plik z nagłówkiem,
- -s, --support SUP Minimalne wsparcie reguł. Domyślnie minimalne wsparcie to 0
- -c, --class COL Indeks atrybutu decyzyjnego z przedziału [0..liczba atrybutów - 1]. Domyślnie - ostatni atrybut ze zbioru danych.

Wynik

Wynikiem działania aplikacji są dwa pliki o nazwach powstałych przez doklejenie do argumentu NAZWA_PLIKU_WYNIKOWEGO rozszerzeń .txt i .csv.
Pliki te zawierają wygenerowane reguły odpowiednio w formacie przyjaznym dla człowieka w pliku txt i przygotowane do zaimportowania do arkusza kalkulacyjnego w formacie csv.

Plik .txt

Plik jest podzielony na sekcje według następnika reguły. W każdej sekcji występuje lista par atrybut-wartość należących do poprzednika reguły.

Przykład

==========
Attribute 4: P

Attribute 0(S), Attribute 1(M), Attribute 3(T)

==========
Attribute 4: N

Attribute 0(S), Attribute 2(H),

Jeśli został podany plik nagłówkowy, to nazwy atrybutów są zastąpione tymi występującymi w pliku.

Plik .csv

Pierwszy wiersz zawiera nazwy wszystkich atrybutów. W kolejnych wierszach występują reguły z atrybutem decyzyjnym na końcu.

Przykład

Attribute 0;Attribute 1;Attribute 2;Attribute 3;Attribute 4
S;M;;T;P
S;;H;;N

Implementacja

Wczytywanie danych

Jeśli podano plik z nagłówkiem, generowane są identyfikatory wartości atrybutów w kolejności podania ich w pliku. W przeciwnym przypadku wczytywany jest plik z danymi, a identyfikatory są przypisywane w kolejności pojawienia się wartości w danych.
Wybranie strategii sortowania wymaga policzenia wsparcia każdego z identyfikatorów, posortowania ich według wsparcia. Następnie ponownie przypisywane są identyfikatory do wartości atrybutów.

Modyfikacja algorytmu CHARM

Modyfikacja algorytmu CHARM [1] użyta w aplikacji polega na dodaniu do każdego węzła dwóch, dodatkowych atrybutów. Pierwszy to klasa decyzyjna pierwszej transakcji występującej w węźle. Drugi to atrybut logiczny mówiący, czy węzeł należy do jednej klasy decyzyjnej.

Budowanie reguł ze zbiorów zamkniętych

Dla każdego zbioru zamkniętego, który spełnia warunek jednolitej klasy decyzyjnej zostaje dodany do zbioru reguł, o ile nie istnieje już dodana reguła będąca jego podzbiorem oraz dająca tę samą decyzję.
Na tym etapie następuje przejście z identyfikatorów liczbowych na nazwy wartości podane w zbiorze danych. Zapisywane są też nazwy atrybutów, jeśli podano nagłówek.

Przykładowy przebieg działania algorytmu

Wykrywane będą reguły o minmalnym wsparciu >= 1, dla zbioru danych:

1			H		P
2		M	H		P
3					P
4				T	P
5	S				P
6		M			P
7	S	M		T	P
8		M	H	T	P
9					P
10	S		H		N
11	S		H	T	N
12				T	N
13	S	M	H		N
14		M	H	T	N

Wywołane polecenie

$ ./charm -s 1 test.data test_results

Wynik

Zawartość pliku test_results.txt to:
==========
Attribute 4: P

Attribute 0(S), Attribute 1(M), Attribute 3(T)

==========
Attribute 4: N

Attribute 0(S), Attribute 2(H),

Wygenerowane zostały dwie reguły:

- {S, M, T} -> {P}
- {S, H} -> {N}

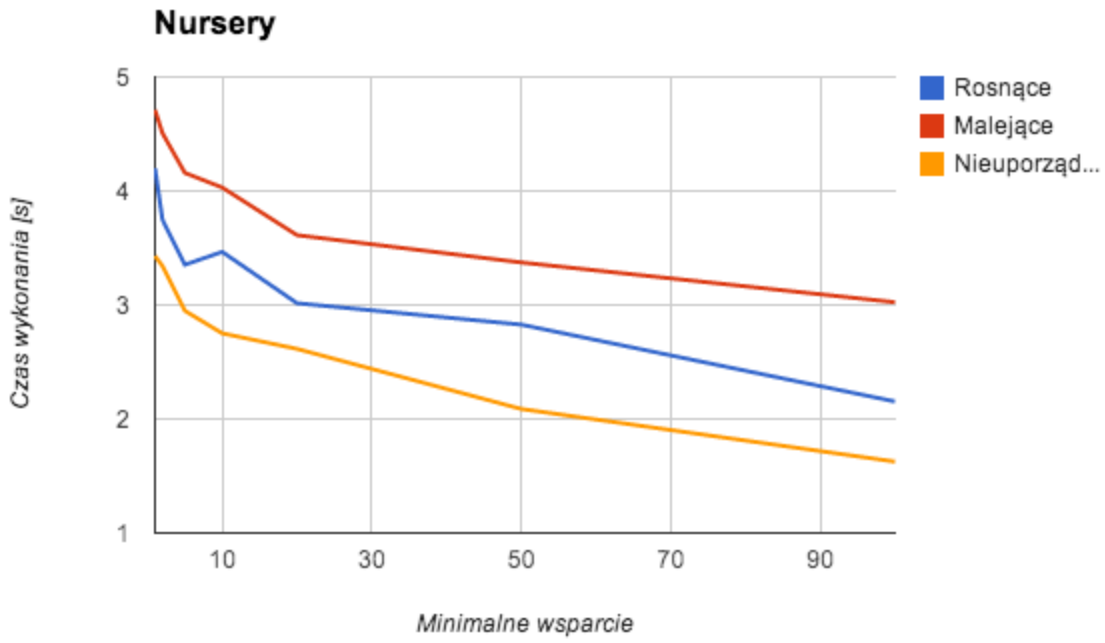
Testy wydajnościowe

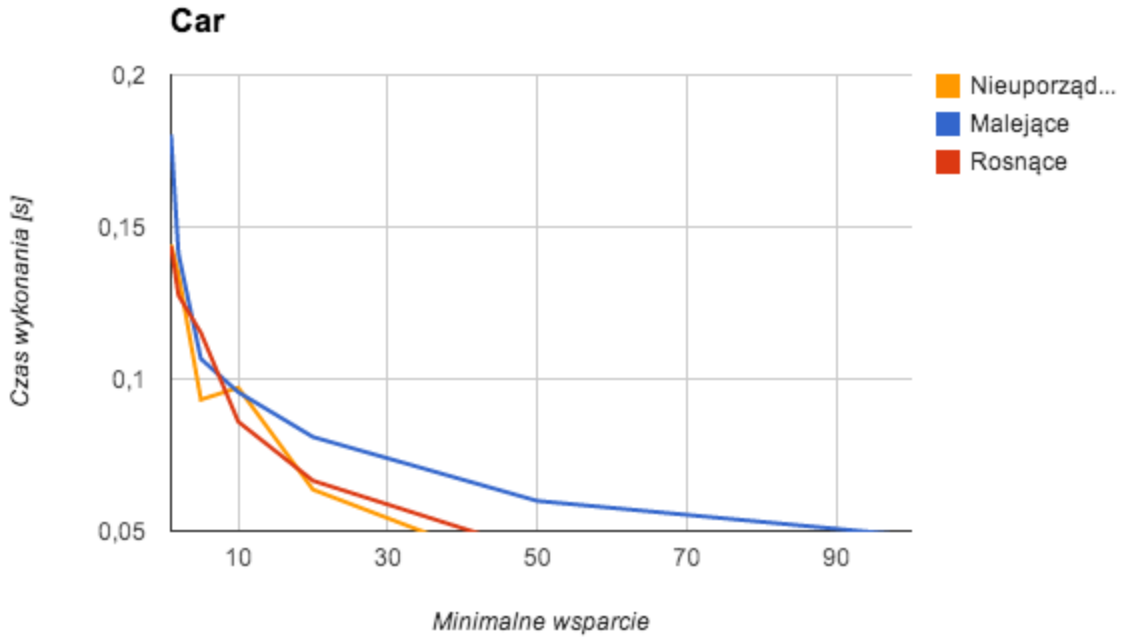
Badanie wydajności algorytmu przeprowadzono na następujących zbiorach danych:

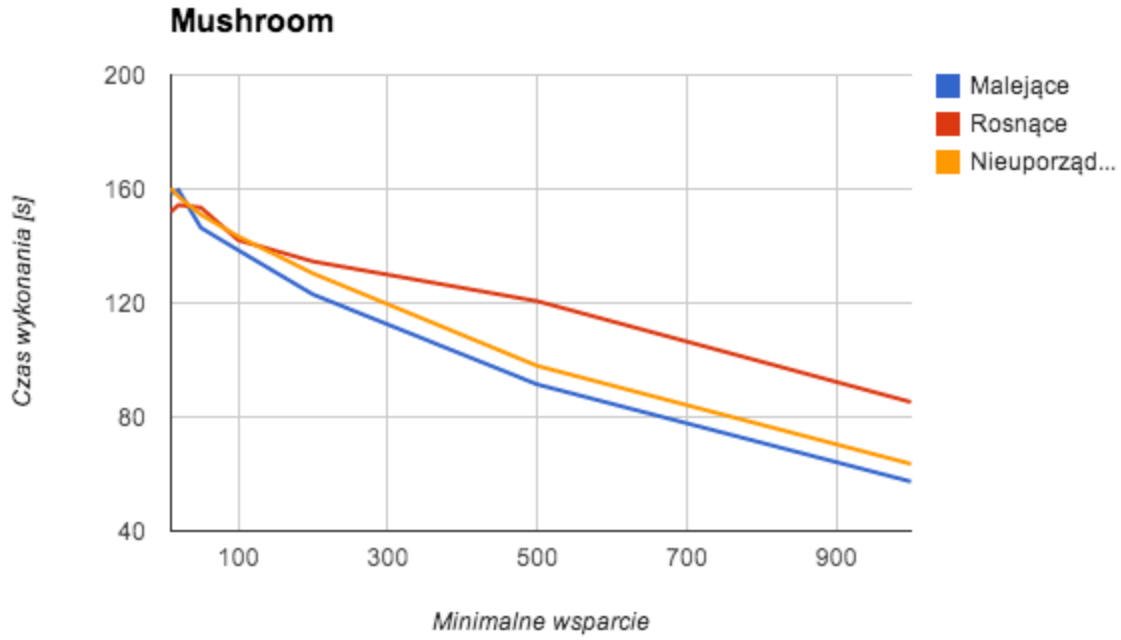
- car,
- mushroom,
- nursery.

Pliki z czasami wykonania dla poszczególnych strategii sortowania znajdują się w katalogach docs/results/{car,mushroom,nursery}/{ascending,descending,unordered}.

Wykresy przedstawiają czas wykonania alorytmu CHARM dla różnych strategii sortowania.




Car
0,2
0,15
0,1
0,05
Czas wykonania [s]
10
30
50
70
90
Minimalne wsparcie
Nieuporząd...
Malejące
Rosnące


Mushroom
200
160
120
80
40
Czas wykonania [s]
100
300
500
700
900
Minimalne wsparcie
Malejące
Rosnące
Nieuporząd...

Wnioski

Dla małych zbiorów danych sortowanie według rosnącego wsparcia jest korzystniejsze. Dla zbioru mushroom ta zależność nie jest zachowana. Może to być spowodowane błędem w algorytmie objawiającym się przy większych zbiorach.

Bibliografia

[1] CHARM: An Efficient Algorithm for Closed Itemset Mining, Zaki M., Hsiao C.