­­­ a

System rekomendacji

Projekt 1

Robert Grochowski (s17124)  
Maciej Lis (s17321)

Polsko-Japońska Akademia Technik Komputerowych

PRO1D

Robert Grochowski

Opis zadania

## Streszczenie

Celem projektu jest zaimplementowanie algorytmu, który wyszukuje reguły asocjacyjne w bazie transakcyjnej pewnego portalu handlowego. Dzięki nim, możliwe jest tworzenie rekomendacji i sugestii dla klientów, które mogą ich zainteresować do kupna innego przedmiotu. Dzięki otrzymanym regułom można:

* Planować promocje często kupowanych ze sobą produktów,
* Lepiej ustawiać towary na pułkach tak aby klient kupił więcej produktów,
* Przy zakupie jednego produktu rekomendować inne produkty, które mają wysokie prawdopodobieństwo kupna właśnie z tym produktem np. na stronie internetowej sklepu.

## Wybrany algorytm

Wybrany algorytm to algorytm **Apriori**.

## Wejście programu

Plik tekstowy o podanej specyfikacji:

* Każdy wiersz w pliku odpowiada jednej transakcji
* Każdy numer oddzielony spacją odpowiada unikalnemu identyfikatorowi produktu

## Wyjście programu

* Zbiory częste (zbiory często kupowanych ze sobą produktów)
* Reguły asocjacyjne

Opis programu

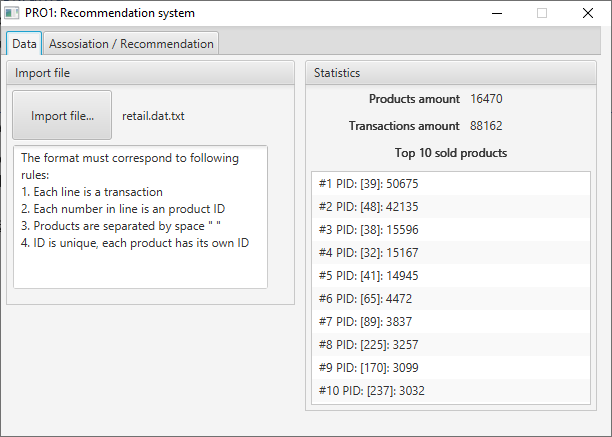
Po uruchomieniu programu należy wczytać plik (zgodny ze specyfikacją) klikając na przycisk *Import File…* w zakładce *Data*. Następnie program dokona wstępnej analizy pliku i wypisze wstępną analizę zbioru danych w polu *Statisctics*.

Po załadowaniu pliku, możemy korzystać z zakładki *Assosiation / Recommendatio*n. W polu *Settings* należy wpisać wartości od (0 do 1) min\_support oraz min\_confidence, które odpowiadają wartościom procentowym. Następnie aby zbudować zbiory częste oraz reguły asocjacyjne trzeba nacisnąć przycisk *Build*.  
  
Przy dużym zbiorze danych i małych wartościach wsparcia lub wiarygodności, obliczenia mogą zająć chwilę. Będzie o tym informować pojawiające się okienko informujące o działaniu algorytmu.

Po ukończeniu obliczeń przez algorytm, pola w tabeli *Frequent Items Set* (zbiory częste) oraz *Assosiation Rules* (reguły asocjacyjne) wypełnią się w zależności od wyników. Aby sprawdzić który produkt występuje często wraz z innymi, wystarczy wpisać w polu *product id* id produktu który chcemy sprawdzić a następnie nacisnąć przycisk *Filter*. To samo dotyczy się pola w regułach asocjacyjnych. Aby wyczyścić filtr, należy wyczyścić pole tekstowe i nacisnąć przycisk *Filter*.  
  
Na dole okna programu, wypisywana jest informacja o ilości znalezionych zbiorów częstych oraz reguł asocjacyjnych.

Analiza zbioru danych

Statystyki  
Analizowany zbiór danych to *PRO1\_Datasets/Retails/retail.dat.txt*

**Ilość produktów**: 16470  
**Ilość transakcji**: 88162  
  
Dane te można zaobserwować po zaimportowaniu pliku do programu.   
  
Ponadto, program wyświetla TOP10 najczęściej sprzedawanych produktów. PID (Product ID) oznacza identyfikator produktu, natomiast po dwukropku występuje ilość wystąpień.

Wstępne przetwarzanie danych

Podane dane były w formacie przyjaznym do wczytania przez program. Ponadto, zbiór danych zawierał jedynie wartości liczbowe, zatem są już dyskretyzowane.

Zbieranie statystyk następuje w momencie wywołania konstruktora z klasy MyApriori.

Ilość niepustych linii reprezentuje ilość transakcji. Pozyskanie najczęściej kupowanych produktów jest otrzymywane dzięki sprowadzeniu zbioru danych do mapy o strukturze <Integer, Integer> co odpowiada następująco <PID, Wystąpienia>. Ilość produktów odpowiada wielkości mapy, natomiast Top10 produktów jest otrzymywanych poprzez posortowanie mapy malejąco względem wartości i zredukowania jej ilości do 10 elementów.

Narzędzia, środowisko i biblioteki

Język  
Użyty język to Java 8.0.

## Środowisko programistyczne

Użyte środowisko programistyczne to IntelliJ IDEA.  
  
Biblioteki użyte w projekcie

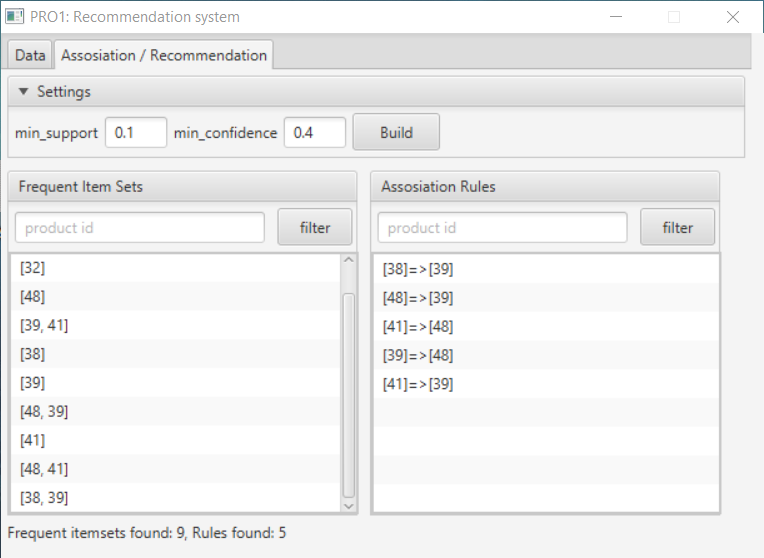
1. JavaFX (Interfejs graficzny)
2. Algorytm Apriori[[1]](#footnote-1) do pozyskiwania zbiorów częstych (Klasa Apriori)

Do pozyskania reguł asocjacyjnych napisaliśmy własną implementację algorytmu Apriori.

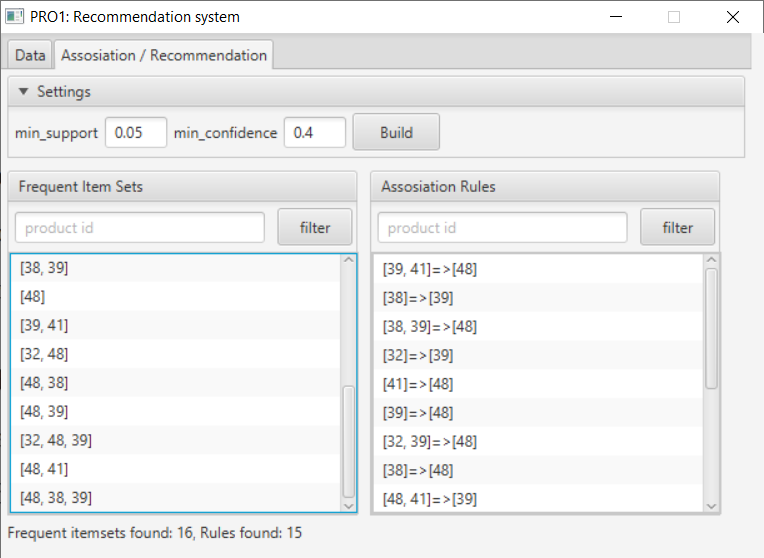
## Narzędzia

Do sprawdzenia poprawności wyników użyliśmy programu Weka. Niestety, musieliśmy testować nasz program na mniejszych zbiorach, ponieważ Weka nie jest ona w stanie przyjąć pliku w podanym formacie, a jego konwersja do obsługiwanego formatu generuje plik o wielkich rozmiarach, którego nie jest w stanie obsłużyć.

Wyniki eksperymentalne

Wyniki oczywiście zależą od dwóch parametrów: min\_support (wsparcie) oraz min\_confidence (wiarygodność). Im mniejsze wartości, tym więcej wyników.  
  
Dla przykładu dla wartości min\_support=0.1 oraz min\_confidence=0.4 program zwraca 9 zbiorów częstych oraz 5 reguł.

Po zmniejszeniu parametru min\_support do 0.05 pojawiają się już 3-elementowe zbiory częste, w sumie jest ich 16.



Komentarze, wnioski

Algorytm Apriori jest bardzo przydatnym i miarodajnym algorytmem. Jego dodatkową zaletą jest nieskomplikowana zasada działania, jednak jego głównym obciążeniem jest wielokrotne przeglądanie całej bazy w celu wyznaczenia wsparcia i wiarygodności dla kandydatów. Przestaje być efektywnym gdy zbiór danych jest zbyt duży.

1. <https://gist.github.com/monperrus/7157717> [↑](#footnote-ref-1)