Metody eksploracji danych - projekt

Dokumentacja

Przemysław Barcikowski, Dariusz Dudziński

18 stycznia 2015

1 Zadanie

1.1 Temat

Wyszukiwanie uogólnionych wzorców sekwencyjnych (Generalized Sequential Patterns)

1.2 Cel projektu

Celem niniejszego projektu jest stworzenie aplikacji spełniającej poniższe wymagania:

- Aplikacja służy do badania sekwencyjnych reguł asocjacyjnych na podstawie notowań giełdowych,
- Aplikacja przyjmuje pliki z danymi w formacie .csv, gdzie pierwsza kolumna to nazwa serii, druga to data. Kolejne atrybuty będą wykorzystywane do samego wyliczania sekwencji. Są to wartości liczbowe,
- Aplikacja automatycznie buduje taksonomię na atrybutach liczbowych w postaci
  - zaokrąglenie do pełnej wartości,
  - wartość dodatnia/ujemna.
- Aplikacja operuje z zadanym przez użytkownika parametrami,
- Aplikacja generuje sekwencyjne reguły asocjacyjne,
- Aplikacja jest napisana w języku Java.

Dodatkowo:

- Aplikacja powinna zostać przetestowana zarówno pod kątem poprawności, jak i wydajności,
- należy przeprowadzić eksperymenty mające na celu wyszukanie ciekawych wzorców sekwencyjnych.

2 Rozwiązanie

Do rozwiązania wyżej postawionego problemu został wykorzystany algorytm Generalized Sequential Patterns (GSP) [2]. Jako dane do testowania oraz eksperymentów wybrano notowania giełdowe indeksu Dow Jones, pobrane ze strony [1].

3 Implementacja

3.1 Funkcjonalności aplikacji

Aplikacja wykonuje algorytm GSP (opisany w [2]), czyli wyszukuje wszystkie uogólnione wzorce sekwencyjne w zadanym pliku z danymi, przy zadanych parametrach algorytmu (patrz 3.2). Jako wynik działania, aplikacja kieruje do standardowego strumienia wyjścia następujące informacje:

- Raport z wyszukiwania wzorców sekwencyjnych (przykład: Listing 1.),

```
SEQUENCE SEARCH REPORT:

Step: 1
generated candidates :76
candidates rejected by hash tree: 64
confirmed sequences  :12
ver true 12
```

Listing 1: Raport z wyszukiwania

- Podsumowanie wykonania algorytmu (przykład: Listing 2.), zawierające:
  - Zadane parametry,
  - Informacje dotyczące samego wykonania algorytmu,
  - Wskaźniki wydajności.

```
SUMMARY

Parameters:
file:         testdata/test4.csv
minSupp:      2
minGap:       28
maxGap:       49
timeConstr:   365
widnowSize:   7
useHashTree:  true
hierarchy   : false

Execution info:
execTime: 478ms
Pattern Sequence found: 325
Pattern Sequence reduced by hash tree : 91
Longest: 7

Performance indicators:
Ratio [Confirmed Sequences/Generated Candidates]: 0.3722794959908362
Exec time per confirmed sequence: 1.4707692307692308ms
```

Listing 2: Podsumowanie

- Wyszukane sekwencje (przykład: Listing 3.).

```
RESULT SERIES:

support: 20 : close.sign:1 ,
support: 20 : close_change.sign:1 ,
support: 20 : volume.sign:1 ,
support: 20 : volume_change:0 ,
```

Listing 3: Wyszukane sekwencje

3.2 Konfiguracja parametrów programu

Parametry algorytmu są podawane za pomocą pliku konfiguracyjnego. Użytkownik może regulować następujące parametry algorytmu:

- wykorzystywanie drzewa hashującego (parametr $useHashTree$, zmienna binarna),
- wykorzystywanie taksonomii (parametr $useTaxonomies$, zmienna binarna),
- wielkość okna czasowego (parametr $slidingWindowSize$, zmienna całkowita, podawana w dniach),
- minimalne wsparcie (parametr $minSupport$, zmienna całkowita, podawana w dniach),
- minimalny odstęp (parametr $minGap$, zmienna całkowita, podawana w dniach),
- maksymalny odstęp (parametr $maxGap$, zmienna całkowita, podawana w dniach),
- ograniczenie czasowe (parametr $timeConstraint$, zmienna całkowita, podawana w dniach),
- ścieżka do pliku z danymi (parametr $dataFilePath$).

Przykładowy plik konfiguracyjny został pokazany na Listingu 4.

```
useHashTree=true
useTaxonomies=true
slidingWindowSize=7
minSupport=2
minGap=28
maxGap=49
timeConstraint=365
dataFilePath=testdata/test4.csv
```

Listing 4: Przykładowy plik konfiguracyjny

3.3 Zalecany sposób uruchamiania

Aplikację można uruchomić na kilka sposobów, zalecany to stworzenie wykonywalnego pliku *.jar* i wywoływanie go w linii poleceń. Możliwe opcje wywoływania:

- Z domyślnym plikiem konfiguracyjnym, jego nazwa to $config.properties$, musi się on znajdować w tej samej lokacji, co plik wykonywalny (przykład: Listing 5.)

```
java -jar programGSP.jar
```

Listing 5: Wywołanie dla domyślnego pliku konfiguracyjnego

- Z podaniem ścieżki do pliku konfiguracyjnego (przykład: Listing 6.)

```
java -jar programGSP.jar "config.properties_custom"
```

Listing 6: Wywołanie ze specyfikacją pliku konfiguracyjnego

W zależności od wielkości pliku z danymi, wynik działania programu może składać się z wielu linii tekstu. Z tego względu zaleca się przekierowanie wyniku programu do pliku tekstowego (przykład: Listing 7.)

```
java -jar programGSP.jar "config.properties_custom" > output.txt
```

Listing 7: Wywołanie dla domyślnego pliku konfiguracyjnego

3.4 Wybrana technologia i wydajność

Aplikacja została napisana w języku Java ze względu na łatwość implementacji. Zostało to okupione wydajnością aplikacji, gdyż Java nie należy do najszybszych języków programowania. Świadomie zrezygnowano z poprawy wydajności na rzecz wygody programowania ze względu na to, że aplikacja ma charakter jedynie demonstracyjny, nie została stworzona z myślą o zastosowaniu w biznesie ani badaniach naukowych.

4 Testy

4.1 Jakościowe

Poprawność działania aplikacji zostało przetestowane przy pomocy testów jednostkowych, zawartych w klasach $SequencePatternsTest$ oraz $CSVReaderTest$ (zawarte w plikach $.java$ o takich samych nazwach)

4.2 Wydajnościowe

Sprawdzono również wydajność programu, w zależności od wielkości okna czasowego (sliding window) oraz tego, czy zostały wykorzystane drzewo hashujące oraz taksonomia. Testy przeprowadzone dla dwóch plików z danymi: małego oraz dużego (pod względem ilości rekordów).

Wyniki algorytmu uruchomionego bez użycia taksonomii nie są interesujące, niezależnie od zestawu danych oraz innych parametrów. algorytm znajdował bardzo mało sekwencji, wyniki nie nadawały się do żadnej sensownej interpretacji. Następujące parametry były stałe dla wszystkich testów:

- $useTaxonomies$ = true,
- $minSupport = 2$,
- $minGap = 28$,
- $maxGap = 49$,
- $timeConstraint = 365$.

Przyjęto następujące oznaczenia dla wyników przedstawionych w Tabelach 1. 2. 3. oraz 4.:

sW - rozmiar okna czasowego (sliding window) w dniach,

hT - zmienna binarna oznaczająca, czy zostało wykorzystane drzewo hashujące.

4.2.1 Testy dla pliku 'test4.csv'

Omawiany plik zawiera 10 rekordów. Tabele 1. oraz 2. przedstawiają wyniki testów.

Tabela 1: Śr. czas potrzebny na wygenerowanie jednej sekwencji [ms]

$sW \backslash hT$	true	false
7	14.58	14.89
14	19.27	18.51

Tabela 2: Współczynnik wygenerowanych kandydatów do potwierdzonych sekwencji

$sW \backslash hT$	true	false
7	0.400	0.400
14	0.614	0.614

Rzut oka na Tabelę 2. wystarczy aby wywnioskować, że dla okna czasowego o długości dwóch tygodni algorytm pracuje bardziej efektywnie, więcej kandydatów zostaje zaliczonych jako sekwencje. Lecz jak pokazuje Tabela 1. obliczenia przy większym oknie czasowym wymagają większego nakładu czasowego, średni czas wyszukania jednej sekwencji jest dłuższy, niż dla okna czasowego o długości jednego tygodnia. Okazuje się, że fakt wykorzystania drzewa hashującego nie ma znaczącego wpływu na omawiane wyżej wskaźniki.

4.2.2 Testy dla pliku 'testBIG.csv'

Omawiany plik zawiera 751 rekordów. Tabele 3. oraz 4. przedstawiają wyniki testów.

Tabela 3: Śr. czas potrzebny na wygenerowanie jednej sekwencji [ms]

$sW \backslash hT$	true	false
7	103.2	120.5
14	161.5	166.7

Tabela 4: Współczynnik wygenerowanych kandydatów do potwierdzonych sekwencji

$sW \backslash hT$	true	false
7	0.107	0.107
14	0.174	0.174

Wnioski z testów przeprowadzonych na większej porcji danych są analogiczne do tych przedstawionych na końcu sekcji 4.2.1.

5 Eksperymenty

5.1 Cel eksperymentów

Celem eksperymentów opisanych w niniejszej sekcji jest wyszukanie wzorców sekwencyjnych, które pomogą przewidywać zmiany cen akcji na giełdzie. Aby sekwencje nadawały się do wnioskowania, muszą mieć możliwie wysokie wsparcie. wnioskowanie nie powinno dotyczyć przedziału większego, niż dwa tygodnie (aby zapewnić wzajemny wpływ poszczególnych wydarzeń). Oczywiście, sekwencje dotyczące wolumenu obrotu i jego zmian powinny być pominięte.

5.2 Dane

Oprogramowanie zostało napisane do badania notowań giełdowych, więc eksperymenty przeprowadzono na takich właśnie danych. Badano notowania indeksu Dow Jones, plik wsadowy zawierał 751 rekordów, dane pozyskano z [1].

5.3 Eksperymenty

5.3.1 Parametry początkowe

Jako punkt wyjścia przyjęto parametry pokazane na Listingu 8.

```
useHashTree=true
useTaxonomies=true
slidingWindowSize=7
minSupport=10
minGap=28
maxGap=49
timeConstraint=365
dataFilePath=testdata/testBIG.csv
```

Listing 8: Parametry początkowe

Uruchomienie algorytmu dla parametrów przedstawionych na Listingu 8. dało zdecydowanie za dużo sekwencji, więc te ustawienia zostały potraktowane jako punkt wyjścia do dalszych poszukiwań.

Wykorzystanie zarówno drzewa hashującego, jak i taksonomii jest sensowne, więc w dalszych eksperymentach te parametry nie będą modyfikowane. Bez zmian pozostaną również ograniczenie czasowe (*timeConstraint*) oraz plik z danymi.

5.3.2 Modyfikacja min. wsparcia

Kolejne eksperymenty zostały wykonane dla kolejnych wartości min. wsparcia, aby znaleźć jak najczęściej występujące sekwencje. Ustawiono kolejno następujące wartości: 15, 25, 30. Najwyższe występujące wsparcie wynosiło właśnie 30, w ostatnim eksperymencie znaleziono 123 sekwencje, z czego 15 nie zawierało krótszych sekwencji dotyczących wolumenu. To niewielka liczba, lecz wciąż zbyt duża.

5.3.3 Modyfikacja wielkości okna czasowego

Następnie ustawiono wielkość okna czasowego na wartości: 5, 3, 2, co oznacza branie pod uwagę przedziału (kolejno) 10 dni, 6 dni oraz 4 dni. zmiana wielkości okna czasowego na 5 poskutkowało zmniejszeniem liczby znalezionych sekwencji do 71, lecz dalsza modyfikacja tego parametru nie przyniosła żadnych efektów, więc pozostano przy wartości 5. Z wyszukanych sekwencji 9 nie zawierało elementów dotyczących wolumenu, a 5 z nich składało się z więcej niż jednego elementu. To liczba odpowiednio mała, aby przystąpić do szczegółowej analizy wyników.

5.4 Wyniki

W wyniku wyżej opisanych eksperymentów otrzymano 5 sekwencji składających się z co najmniej dwóch elementów dotyczących jedynie cen akcji. Listing 9. pokazuje parametry, dla jakich uzyskano ten wynik, a wspomniane sekwencje znajdują się na Listingu 10.

```
useHashTree=true
useTaxonomies=true
slidingWindowSize=5
minSupport=30
minGap=28
maxGap=49
timeConstraint=365
dataFilePath=testdata/testBIG.csv
```

Listing 9: Parametry ostateczne

```
support: 30 : {close.sign:1 , close_change.round:-1 , } ,
support: 30 : {close.sign:1 , close_change.sign:-1 , } ,
support: 30 : {close.sign:1 , close_change.sign:1 , } ,
support: 30 : {close_change.sign:-1 , close_change.round:-1 , } ,
support: 30 : {close.sign:1 , close_change.sign:-1 , close_change.round:-1 }
```

Listing 10: Sekwencje wynikowe

Poniżej przedstawiono najbardziej interesujące zależności, jakie można wywnioskować z sekwencji wymienionych na Listingu 10. :

- Sekwencje oznaczone numerem 2 oraz 3 występują równie często, co oznacza, że równie często cena akcji spada jak i rośnie ($close_change.sign$ oznacza znak zmiany procentowej ceny akcji). Taki wniosek wzięty bez dodatkowych założeń daje do myślenia, ponieważ jest tożsamy ze stwierdzeniem, że na giełdzie równie często się zyskuje, co traci.

- Sekwencja oznaczona numerem 4 pokazuje, że najczęściej występującą zaokrągloną wielkością straty ($close_change.sign$:-1) jest -1 ($close_change.round$:-1), co oznacza, że jeżeli cena akcji spada, to najczęściej o około jeden punkt procentowy. To pocieszający wniosek, gdyż podpowiada, że nawet jeżeli inwestor traci, to najczęściej niedużo na przestrzeni 10 dni ($slidingwindowSize$=5).

Wyżej opisane zależności spełniają cel założony na początku eksperymentów, gdyż wynikają z nich sensowne wnioski, możliwe do zastosowania w rzeczywistości. Oczywiście, dla wyższej wiarygodności eksperymenty powinny zostać przeprowadzone na większych i bardziej różnorodnych zbiorach danych.

6 Wnioski

Stworzona aplikacja spełnia wszystkie cele założone na początku projektu, działa poprawnie oraz jest wystarczająco wydajna, aby można było ją używać do prostych eksperymentów. Aplikacja w pełni nadaje się do demonstracji działania algorytmu GSP oraz celów dydaktycznych. Jej wydajność można oczywiście znacząco poprawić np. przepisując cały program na język C, lecz wymagało by to odrobinę więcej czasu, niż implementacja tych samych funkcjonalności w języku Java.

Stworzone oprogramowanie nie nadaje się do zastosowań badawczych ani biznesowych przez zbyt małą wydajność oraz niewielką liczbę funkcjonalności, ale stanowi dobrą podstawę do dalszego rozwoju.

Literatura

[1] Dow jones index data set. https://archive.ics.uci.edu/ml/datasets/Dow+Jones+Index. Accessed: 2015-01-17.

[2] Ramakrishnan Srikant ans Rakesh Agrawal. Mining sequential patterns: Generalizations and performance improvements.