УДК 336.74:004.738.5

МАСШТАБИРУЕМЫЙ АЛГОРИТМ ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ APRIORI

С.А. Николайчик
Магистрант кафедры математического и информационного обеспечения экономических систем
УО ГрГУ.им. Я.Купалы

Н.В. Марковская
Доцент кафедры математического и информационного обеспечения экономических систем
УО ГрГУ.им. Я.Купалы

УО Гродненский государственный университет имени Янки Купалы
E-mail: sergeygrodno94@gmail.com, n.markovskaya@grsu.by

С.А. Николайчик
Окончил Гродненский государственный университет имени Янки Купалы. Магистрант кафедры математического и информационного обеспечения экономических систем УО ГрГУ им. Я. Купалы.

Н.В. Марковская
Доцент кафедры математического и информационного обеспечения экономических систем УО ГрГУ им.Я.Купалы, кандидат физико-математических наук, доцент.

Аннотация. В данной работе рассмотрен механизм нахождения логических закономерностей между связанными элементами – ассоциативные правила. Описан алгоритм поиска ассоциативных правил Apriori, реализованный в пакете arules, для языка программирования R. В работе используется набор данных “Income” из пакета arules.

Ключевые слова: алгоритм Apriori, ассоциативные правила, поддержка, достоверность, R, правила, arules.

Ассоциативные правила представляют собой механизм нахождения логических закономерностей между связанными элементами (событиями или объектами). Пусть имеется A = {a1, a2, a3, …, an} – конечное множество уникальных элементов (list of items). Из этих компонентов может быть составлено множество наборов T (sets of items), т.е. $T \subseteq A$.

Ассоциативные правила A→T имеют следующий вид: если <условие> то <результат>, где в отличие от деревьев классификации, <условие> - не логическое выражение, а набор объектов из множества A, с которыми связаны (ассоциированы) объекты того же множества, включенные в <результат> данного правила. Например, ассоциативное правило если (смородина, тля) то (муравьи) означает, что если на кусте смородины встретилась тля, то ищи поблизости и муравьев.

Понятие «вид элемента a_k» легко может быть обобщено на ту или иную его категорию или вещественное значение, т.е. концепция ассоциативного анализа может быть применена для комбинаций любых переменных.

Выделяют три вида правил:

- полезные правила, содержащие действительную информацию, которая ранее была неизвестна, но имеет логическое объяснение;
- тривиальные правила, содержащие действительную и легко объяснимую информацию, отражающую известные законы в исследуемой области, и поэтому не приносящие какой-либо пользы;
- непонятные правила, содержащие информацию, которая не может быть объяснена (такие правила или получают на основе аномальных исходных данных, или они содержат глубоко скрытые закономерности, и поэтому для интерпретации непонятных правил нужен дополнительный анализ).

Поиск ассоциативных правил обычно выполняют в два этапа:

- в пуле имеющихся признаков A находят наиболее часто встречающиеся комбинации элементов T;
- из этих найденных наиболее часто встречающихся наборов формируют ассоциативные правила.

Для оценки полезности и продуктивности перебираемых правил используются различные частотные критерии, анализирующие встречаемость кандидата в массиве экспериментальных данных. Важнейшими из них являются поддержка (support) и достоверность (confidence). Правило A→T имеет поддержку s, если оно справедливо для s% взятых в анализ случаев:

$$\text{support}(A \rightarrow T) = P(A \cup T)$$

Достоверность правила показывает, какова вероятность того, что из наличия в рассматриваемом случае условной части правила следует наличие заключительной его части (т.е. из A следует T):

$$\text{confidence}(A \rightarrow T) = P(A \cup T)/P(A) = \text{support}(A \rightarrow T) / \text{support}(A).$$

Алгоритмы поиска ассоциативных правил отбирают тех кандидатов, у которых поддержка и достоверность выше некоторых наперед заданных порогов: minsupport и minconfidence. Если поддержка имеет большое значение, то алгоритмы будут находить правила, хорошо известные аналитику или настолько очевидные, что нет никакого смысла проводить такой анализ. Большинство интересных правил находят именно при низком значении порога поддержки. С другой стороны, низкое значение minsupport ведет к генерации огромного количества вариантов, что требует существенных вычислительных ресурсов или ведет к генерации статистически необоснованных правил.

В пакете arules для R используются и другие показатели - подъемная сила, или лифт (lift), которая показывает, насколько повышается вероятность нахождения T в анализируемом случае, если в нем уже имеется A:

$$\text{lift}(A \rightarrow T) = \text{confidence}(A \rightarrow T) / \text{support}(T)$$

и усиление (leverage), которое отражает, насколько интересной может быть более высокая частота A и T в сочетании с более низким подъемом:

$$\text{leverage}(A \rightarrow T) = \text{support}(A \rightarrow T) - \text{support}(A) \times \text{support}(T)$$

Первый алгоритм поиска ассоциативных правил был разработан в 1993 г. сотрудниками исследовательского центра IBM, что сразу возбудило интерес к этому направлению. Каждый год появлялось несколько новых алгоритмов (DHP, Partition, DIC и др.), из которых наиболее известным остался алгоритм “Apriori” (Agrawal, Srikant, 1994).

Пакет arules позволяет находить часто встречающиеся сочетания элементов в данных (frequent itemsets) и отбирать ассоциативные правила, обеспечивая интерфейс к модулям на языке C, которые реализуют алгоритмы “Apriori” и “Eclat”. Так как обычно обрабатываются большие множества наборов и правил, то для уменьшения объёмов требуемой памяти пакет содержит развитый инструментарий преобразования разреженных входных матриц в компактные наборы транзакций (Hahsler et al., 2016; Огнева, 2012).

Для реализации работы с алгоритмами выделения ассоциаций в arules реализованы специальные типы данных, относящиеся к объектам трех классов: входной массив транзакций (transactions) и на выходе - часто встречающиеся фрагменты данных (itemsets) и правила (rules).

Объекты класса transactions представляют собой специально организованные бинарные матрицы со строками-наборами и столбцами-признаками, содержащие значения элемента 1, если соответствующий признак есть в транзакции, и 0, если он отсутствует. В зависимости от типа данных и способа их загрузки, эти объекты могут иметь разные способы организации и состав дополнительных слотов. В частности, подкласс itemMatrix является одновременно средством представления разреженных матриц с использованием функционала пакета Matrix. Другим способом формирования экземпляров класса transactions является загрузка данных из файла функцией read.transactions().

Поиск ассоциативных правил является не вполне тривиальной задачей, т.к. с ростом числа элементов в A экспоненциально растет число их потенциальных комбинаций. Алгоритм “Apriori” является итерационным, при этом сначала выполняются действия для одноэлементных наборов, затем для 2-х, 3-х элементных и т.д.

На первом шаге первой итерации алгоритма подсчитываются одноэлементные часто встречающиеся наборы. Для этого необходимо пройтись по всему массиву данных и подсчитать для них поддержку, т.е. сколько раз набор встречается в имеющемся наборе данных. При последующем поиске k-элементных наборов генерация претендентов состоит из двух фаз - формирование кандидатов нового уровня на основе (k−1) -элементных наборов, которые были определены на предыдущей итерации алгоритма, и удаление избыточных кандидатов. После того как найдены все часто встречающиеся наборы элементов, выполняют процедуру непосредственного извлечения правил из построенного хеш-дерева.

Результатом анализа транзакций с помощью пакета arules являются объекты класса associations, включающие описания множества отношений между признаками (в виде часто встречающихся фрагментов, или правил), которые отбираются в соответствии с различными перечисленными выше мерами качества. Подкласс rules состоит из двух объектов itemMatrix, представляющих левую lhs (left-hand-side) и правую rhs (right-hand-side) сторону правила A→T, т.е. A - lhs, T - rhs.

Формирование правил осуществляется функцией apriori() с указанием пороговых значений поддержки и достоверности. Функция summary() обеспечивает частотный анализ правил по их длине и достигнутым мерам качества.

Функция plot() из пакета arulesViz позволяет получать различные формы визуализации синтезированных правил.

Метод "graph" функции plot() показывает правила и составляющие их признаки в виде графа, размер узлов которого пропорционален уровню поддержки каждого представленного правила.

Рассмотрим задачу предобработки и анализа анкетных данных. Для этого возьмём базу Income, содержащуюся в пакете “arules”. Данные были извлечены Барри Беккером из базы данных переписи 1994 года.

Загрузим набор данных «Income» (Рисунок 1).

```
> data("Income")
> Income
transactions in sparse format with
 6876 transactions (rows) and
 50 items (columns)
```

Рисунок 1. Загрузка набора данных «Income». Вывод информации о наборе данных.

Это маркетинговые данные содержащие 6876 транзакций по 50 характеристикам.
Далее воспользуемся следующей командой:

itemFrequencyPlot(Income,support=0.3,cex.name=0.8)

В результате получим график, представленный на рисунке 2.

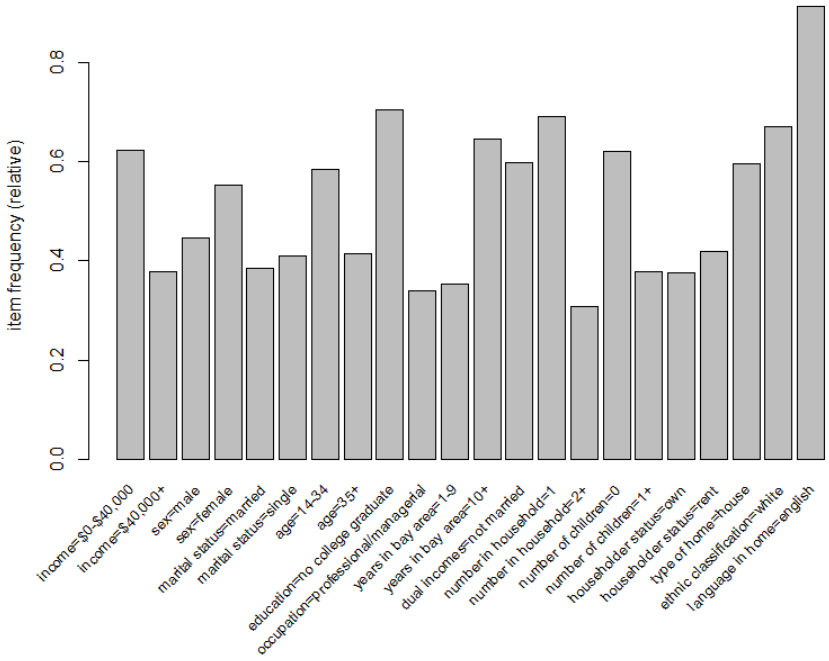

Рисунок 2. Визуализация бинаризованных данных

На данном графике представлены основные характеристики:
- Доход: $0-$40000(Income=$0-$40000), $40000 и более (Income=$40000+);
- Пол: мужской (sex=male), женский(sex=female);
- Семейный статус: женат/замужем (marital status=married), холост/не замужем (marital status=single);
- Возраст: 14-34 (age=14-34), 35 и старше (age=35+);
- Количество детей: нет детей (number of children=0), один и более (number of children=1+);
- Родной язык: английский (language in home=english), и другие.

Для анализа данных используем алгоритм Apriori с минимальной поддержкой 0.15 и значимостью 0.6 (Рисунок 3).

```
> rules<-apriori(Income,parameter = list(support=0.15,confidence=0.6))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.6    0.1    1 none FALSE            TRUE       5    0.15      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1031

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[50 item(s), 6876 transaction(s)] done [0.02s].
sorting and recoding items ... [27 item(s)] done [0.00s].
creating transaction tree ... done [0.05s].
checking subsets of size 1 2 3 4 5 6 done [0.04s].
writing ... [4881 rule(s)] done [0.01s].
creating S4 object  ... done [0.00s].
```

Рисунок 3. Применение алгоритма Apriori к набору данных “Income”

В результате работы алгоритм вывел 4881 правило.

Проведем частотный анализ правил по их длине и достигнутым мерам качества (Рисунок 4).

```
> summary(rules)
set of 4881 rules

rule length distribution (lhs + rhs):sizes
   1    2    3    4    5    6
   7  196 1189 2068 1245  176

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   3.000   4.000   3.999   5.000   6.000

summary of quality measures:
    support         confidence          lift            count
 Min.   :0.1501   Min.   :0.6000   Min.   :0.8508   Min.   :1032
 1st Qu.:0.1638   1st Qu.:0.6927   1st Qu.:1.0370   1st Qu.:1126
 Median :0.1827   Median :0.7839   Median :1.1723   Median :1256
 Mean   :0.2041   Mean   :0.7917   Mean   :1.2663   Mean   :1403
 3rd Qu.:0.2192   3rd Qu.:0.8900   3rd Qu.:1.3540   3rd Qu.:1507
 Max.   :0.9129   Max.   :1.0000   Max.   :3.5500   Max.   :6277

mining info:
   data ntransactions support confidence
 Income          6876    0.15        0.6
```

Рисунок 4. Частотный анализ правил

Воспользуемся функцией plot() из пакеты aruleViz (Рисунок 5).

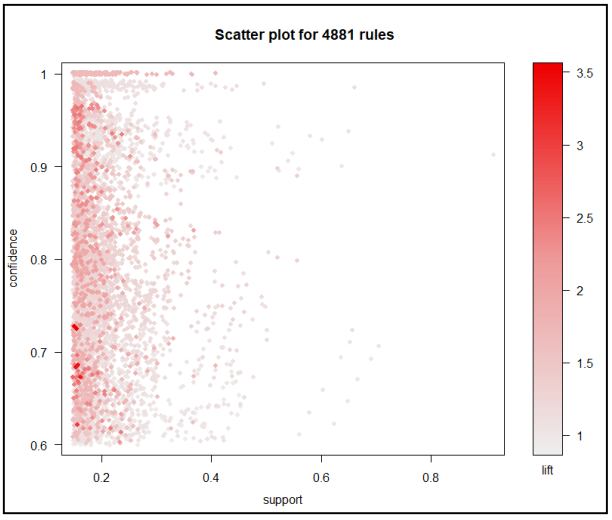

Рисунок 5. Поддержка, лифт и достоверность сгенерированных правил

Для решения задачи выявления характерных особенностей групп населения нас интересуют, в первую очередь, высококачественные правила, имеющие соответствующий признак группы в правой части. Определим, например, признаки группы людей у которых семейное положение женат/замужем (Рисунок 6).

```
> rulesMarried<-subset(rules,subset=rhs %in% "marital status=married"&lift>2.45)
> rulesMarried
set of 5 rules
> inspect(head(rulesMarried,n=5,by = "support"))
    lhs                                rhs                          support confidence     lift count
[1] {dual incomes=yes,
     type of home=house}       => {marital status=married} 0.1671030  0.9647355 2.501328  1149
[2] {income=$40,000+,
     dual incomes=yes}         => {marital status=married} 0.1580861  0.9560246 2.478743  1087
[3] {years in bay area=10+,
     dual incomes=yes}         => {marital status=married} 0.1575044  0.9491674 2.460963  1083
[4] {dual incomes=yes,
     type of home=house,
     language in home=english} => {marital status=married} 0.1548866  0.9646739 2.501168  1065
[5] {income=$40,000+,
     dual incomes=yes,
     language in home=english} => {marital status=married} 0.1509599  0.9549218 2.475883  1038
```

Рисунок 6. Определение признаков для женатых/замужних лиц. Вывод признаков.

Согласно полученным результатам можно сделать выводы, что для женатых/замужних людей чаще всего встречаются следующие группы:

- оба зарабатывают, живут в собственности дом;
- заработок больше 40000, оба зарабатывают;
-проживают вместе больше 10 лет, разговаривают на английском и др.

С помощью команды plot() построим график 5 лучших правил для женатых/замужних лиц в параллельных координатах (Рисунок 7).

plot(head(sort(rulesMarried,by = "support"),5),method = "paracoord")

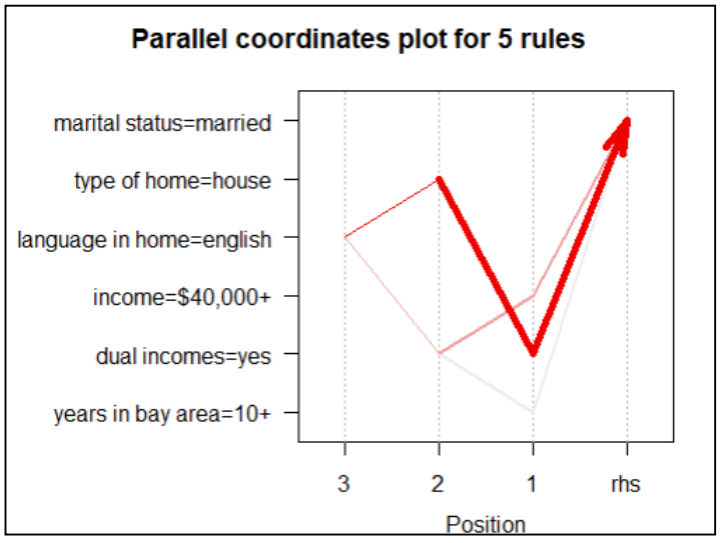

Рисунок 7. График 5 лучших правил для женатых/замужних лиц в параллельных координатах

График в параллельных координатах (method="paracoord") на рисунке 3 показывает, как формируются комбинации признаков правой части при увеличении ее размера, а толщина линий соответствует уровню поддержки.

Построим граф 5 лучших правил для женатых/замужних лиц (Рисунок 8).

```
plot(head(sort(rulesMarried,by = "support"),6),method = "graph",control = lis
    t(nodeCol=grey.colors(10),edgeCol=grey(.7),alpha=1))
```

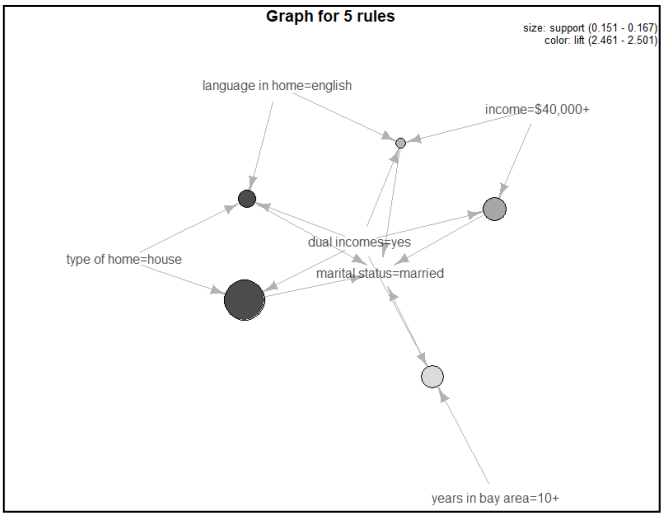

Рисунок 8. Визуализация в форме графа 5 лучших правил для женатых/замужних лиц

Метод «graph» функции plot() показывает правила и составляющие их признаки в виде графа, размер узлов которого пропорционален уровню поддержки каждого представленного правила.

Литература:

[1]. A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases", In Proc. 21st Int'l Conf. Very Large Data Bases, Morgan Kaufmann, San Francisco, 1995.

[2]. R. Agrawal, R. Srikant. "Fast Discovery of Association Rules", In Proc. of the 20th International Conference on VLDB, Santiago, Chile, September 1994.

[3]. Шитиков В. К., Мастицкий С. Э. (2017) Классификация, регрессия, алгоритмы Data Mining с использованием R. - Электронная книга, адрес доступа: https://github.com/ranalytics/data-mining

A SCALABLE SEARCH ALGORITHM FOR APRIORI ASSOCIATIVE RULES

S.A. NIKALAICHYK
Master student of the Department of Mathematical and Information Support of Economic Systems,YKSUG

N.V. MARKOVSKAYA
Associate professor of the Department of Mathematical and Information Support of Economic Systems,YKSUG

Yanka Kupala State University of Grodno, Republic of Belarus
E-mail: sergeygrodno94@gmail.com, n.markovskaya@grsu.by

Abstract. In this paper, we consider the mechanism for finding logical laws between related elements - association rules. The algorithm for searching the associative rules Apriori, implemented in the arules package, is described for the programming language R. We use the "Income" data set from the arules package.

Keywords: Apriori algorithm, association rules, support, validity, R, rules, arules.