УДК 735.29

ПРИМЕНЕНИЕ МЕТОДОВ АССОЦИАТИВНЫХ ПРАВИЛ ДЛЯ ИЗУЧЕНИЯ ПОТРЕБИТЕЛЬСКОГО СПРОСА

Гордеева К.В.,
научный руководитель канд. физ.-мат. наук, доцент Баранова И.В.
Сибирский Федеральный Университет
Институт математики и фундаментальной информатики

1. Интеллектуальный анализ данных

Современные предприятия торговли и оказания услуг собирают подробную информацию о каждой отдельной покупке (заказе), используя кредитные карточки и компьютеризованные системы контроля. Используемые технологии хранения и записи данных позволили накопить большой объем данных о совершенных потребителями покупках, заказах и услугах. В последние годы наблюдается огромный интерес специалистов в области менеджмента и маркетинга к задачам нахождения закономерностей в поведении клиентов, поскольку знание потребностей клиентов, особенностей их поведения позволяет управлять товарной и маркетинговой политикой предприятий и способствовать повышению их прибыльности и конкурентоспособности. Поэтому одним из актуальных направлений современных информационных технологий является анализ накопленных статистических данных с помощью методов интеллектуального анализа данных.

Интеллектуальный анализ данных (Data Mining) − мультидисциплинарная область, возникшая и развивающаяся на базе прикладной статистики, искусственного интеллекта, теории баз данных и другие. Термин Data Mining получил свое название из двух понятий: поиска ценной информации в большой базе данных (Data) и добычи горной руды (Mining). Термин переводится как «добыча» или «раскопка» данных. Понятие интеллектуального анализа данных было предложено Григорем Пиатецким-Шапиро в 1989 году.

Интеллектуальный анализ данных − это процесс обнаружения в данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности.

К методам интеллектуального анализа данных (ИАД) относятся дескриптивный анализ, корреляционный и регрессионный анализ, факторный анализ, дисперсионный анализ, компонентный анализ, дискриминантный анализ, анализ временных рядов. Также методами ИАД являются искусственные нейронные сети, деревья решений, нечеткая логика, генетические алгоритмы, эволюционное программирование, различные методы классификации, моделирования и прогнозирования.

Возникновению Data Mining способствовало совершенствование аппаратного и программного обеспечения, технологий хранения и записи данных, а также совершенствование алгоритмов обработки информации.

В интеллектуальном анализе данных существуют следующие стадии:

1. Выявление закономерностей (свободный поиск),
2. Прогностическое моделирование,
3. Анализ исключений.

На стадии выявления закономерностей осуществляется исследование набора данных с целью поиска скрытых закономерностей. Свободный поиск представлен такими действиями: выявление закономерностей условной логики, выявление

закономерностей ассоциативной логики, выявление трендов и колебаний. Прогностическое моделирование использует результаты работы первой стадии. Действиями данной стадии являются предсказание неизвестных значений и прогнозирование развития процессов. Третьей стадией ИАД является стадия анализа исключений или аномалий, выявленных в найденных закономерностях. Назначение данной стадии − выявление отклонений. Для выявления отклонений в данных необходимо определить норму, которая рассчитывается на стадии свободного поиска. В дополнение к этим стадиям иногда вводят валидацию, следующую за стадией свободного поиска. Цель валидации − проверка найденных закономерностей.

2. Основы ассоциативных правил в интеллектуальном анализе данных

В ходе решения задачи поиска ассоциативных правил отыскиваются закономерности между связанными событиями в наборе данных. Отличие ассоциации от задач классификации и кластеризации заключается в том, что поиск закономерностей осуществляется не на основе свойств анализируемого объекта, а между несколькими событиями, которые происходят одновременно. Рассмотрим два разных алгоритма поиска ассоциативных правил в интеллектуальном анализе данных: метод Apriori и метод Frequent Pattern-Growth Strategy **(**FPG**).**

Приведем основные понятия, связанные с данными методами.
Транзакция − это множество событий, которые произошли одновременно.
Транзакционная (операционная) база данных представляет собой двумерную таблицу, состоящую из номера транзакции (TID) и перечня событий, происходящих во время этой транзакции.
Поддержка − количество или процент транзакций, содержащих определенный набор данных**.**

3. Алгоритм Apriori

Приведем обозначения, используемые в алгоритме:
L_k − множество k-элементных наборов, чья поддержка не меньше заданной пользователем.
C_k − множество потенциально частых k-элементных наборов.

Алгоритм поиска ассоциативных правил Apriori имеет следующий вид:
Шаг 1. Присвоить k = 1 и выполнить отбор всех 1-элементных наборов, у которых поддержка больше минимально заданной пользователем Suppmin.
Шаг 2. k = k + 1.
Шаг 3. Если не удается создавать k-элементные наборы, то завершить алгоритм, иначе выполнить следующий шаг.
Шаг 4. Создать множество k-элементных наборов кандидатов из частых наборов. Для этого необходимо объединить в k-элементные кандидаты (k-1)-элементные частые наборы. Каждый кандидат будет формироваться путем добавления к (k-1) - элементному частому набору - p элемента из другого (k-1) − элементного частого набора - q. Причем добавляется последний элемент набора q, который по порядку выше, чем последний элемент набора p. При этом все k-2 элемента обоих наборов одинаковы.
Шаг 5. Для каждой транзакции T из множества D выбрать кандидатов Ct из множества C_k, присутствующих в транзакции T. Для каждого набора из построенного множества

C_k удалить набор, если хотя бы одно из его (k-1) подмножеств не является часто встречающимся т.е. отсутствует во множестве L_{k-1}.
Шаг 6. Для каждого кандидата из C_k увеличить значение поддержки на единицу.
Шаг 7. Выбрать только кандидатов L_k из множества C_k, у которых значение поддержки больше заданной пользователем Suppmin. Вернуться к шагу 2.
Результатом работы алгоритма является объединение всех множеств L_k для всех k.

4. Алгоритм Frequent Pattern-Growth Strategy (FPG)

В основе метода лежит предобработка базы транзакций, в процессе которой эта база данных (БД) преобразуется в компактную древовидную структуру, называемую Frequent-Pattern Tree – *дерево популярных предметных наборов* (откуда и произошло название алгоритма). В дальнейшем для краткости будем называть эту структуру FP-дерево. К основным преимуществам данного метода относятся:

1. Сжатие БД транзакций в компактную структуру, что обеспечивает очень эффективное и полное извлечение частых предметных наборов;
2. При построении FP-дерева используется технология разделения и захвата (англ.: *divide and conquer*), которая позволяет выполнить декомпозицию одной сложной задачи на множество более простых задач.

Цель: для БД требуется обнаружить все популярные предметные наборы с минимальной поддержкой, заданной экспертом, используя алгоритм FPG, который выполняется в 2 этапа.

Рассмотрим работу алгоритма FPG на конкретном примере. Пусть имеется БД транзакций (табл. 1) с минимальной поддержкой, равной 3.

N	Предметный набор
1	a b c d e
2	a b c
3	a c d e
4	b c d e
5	b c
6	b d e
7	c d e

Табл. 1

Алгоритм FPG имеет следующий вид:

1 этап. Строится FP-дерево, которое в компактном виде представляет информацию о частых предметных наборах.

1.1. Производится первое сканирование транзакций БД, и отбирается множество часто встречающихся предметов, т.е. предметов, у которых поддержка больше или равна минимальной. Обнаруженные частые предметы упорядочиваются в порядке возрастания их поддержки.

1.2. Строится FP-дерево. Сначала упорядочиваются предметы в транзакциях по убыванию значений их поддержек.

Для этого создается начальный (корневой) узел FP-дерева (ROOT). И начинается построение дерева с транзакции №1 для упорядоченных предметных наборов.

Правило построения: если для очередного предмета в дереве встречается узел, имя которого совпадает с именем предмета, то предмет не создает нового узла, а индекс соответствующего узла в дереве увеличивается на 1. В противном случае для этого предмета создается новый узел и ему присваивается индекс 1.

Таким образом, после первого прохода БД и выполнения соответствующих манипуляций с предметными наборами мы построим FP-дерево, которое в компактном виде представляет информацию о частых предметных наборах и позволяет производить их эффективное извлечение, что и выполняется при втором сканировании БД.

2 этап. Производится процесс извлечения из FP-дерева частых предметных наборов.

2.1. Выбирается предмет, и в дереве находятся все пути, которые ведут к узлам этого предмета. Затем для каждого пути производится подсчёт, сколько раз данный предмет встречается в нем. Например, для предмета **a** это запишется в виде (cbdea, 1), (cba, 1) и (cdea, 1).

2.2. Удаляется сам предмет (суффикс набора) из ведущих к нему путей, т.е. {cbde~~a~~, cb~~a~~, cde~~a~~}. После это остаются только префиксы: {cbde, cb, cde}.

2.3. Выполняется подсчёт, сколько раз каждый предмет появляется в префиксах путей, полученных на предыдущем шаге, и производится упорядочение в порядке убывания этих значений. После чего получается новый набор транзакций.

2.4. На его основе строится новое FP-дерево, которое будем называть ***условным FP-деревом*** (conditional FP-tree), поскольку оно связано только с одним объектом (в нашем случае, a).

2.5. В этом FP-дереве находятся все предметы (узлы), для которых поддержка (количество появлений в дереве) равна или больше заданной минимальной поддержки. Если предмет встречается два или более раза, то его индексы, т.е. частоты появлений в условном базисе, суммируются.

2.6. Начиная с верхушки дерева, записываются пути, которые ведут к каждому узлу, для которого поддержка/индекс больше или равны заданной, возвращаем назад предмет (суффикс шаблона), удаленный на шаге 2, и подсчитывается индекс (поддержку), полученную в результате.

5. Сравнение алгоритмов

Выгода представления базы данных транзакций в виде FP-дерева очевидна. Если в исходной базе данных каждый предмет повторяется многократно, то в FP-дереве каждый предмет представляется в виде узла, а его индекс указывает на то, сколько раз данный предмет появляется. Иными словами, если предмет в исходной базе данных транзакций появляется 100 раз, то в дереве для него достаточно создать узел и установить индекс 100.

Сравнительные исследования классического алгоритма Apriori и FPG показали, что с увеличением числа транзакций в БД временные затраты на поиск частых предметных наборов растут для FPG намного медленнее, чем для Apriori (рис. 1).

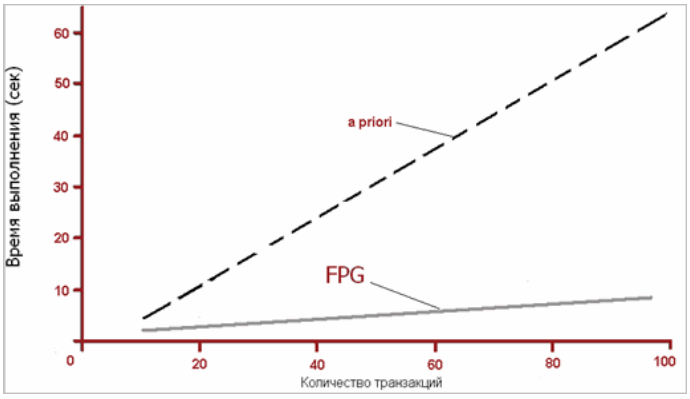

рис. 1