УДК 004.9

ЗАДАЧА ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ В СТАТИСТИКЕ ПОКУПОК ПРОДУКТОВ

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

1 Введение

Интеллектуальный анализ данных (Data Mining) —мультидисциплинарная область, возникшая и развивающаяся на базе прикладной статистики, искусственного интеллекта, теории баз данных и др. Оригинальное англоязычное название Data Mining было предложено Григорием Пиатецким-Шапиро в 1989 году. Название происходит от двух понятий: поиска ценной информации в большой базе данных (Data) и добычи горной руды (Mining). Термин переводится как «добыча» или «раскопка» данных.

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. Data Mining представляет собой технологию, предназначенную для поиска в больших объемах данных неочевидных и полученных на практике закономерностей.

К методам интеллектуального анализа данных относятся всевозможные методы классификации, моделирования и прогнозирования, основанные на применении деревьев решений, искусственных нейронных сетей, генетических алгоритмов, эволюционного программирования, ассоциативной памяти, нечеткой логики. Также методами Data Mining считаются статистические методы: дескриптивный анализ, корреляционный и регрессионный анализ, факторный анализ, дисперсионный анализ, компонентный анализ, дискриминантный анализ, анализ временных рядов.

Основными задачами интеллектуального анализа данных являются:

1. классификация (classification),
2. кластеризация (clustering),
3. ассоциация (associations),
4. прогнозирование (forecasting),
5. оценивание (estimation),
6. визуализация (visualization, graph mining),
7. подведение итогов (summarization).

Сфера применения Data Mining ничем не ограничена, он применим в любых сферах человеческой деятельности, в которых существуют статистические данные: в розничной торговле, банковском деле, телекоммуникации, страховании, медицина, молекулярной генетике, прикладной химии и т.д.

Одним из самых востребованных методов интеллектуального анализа данных является метод поиска ассоциативных правил. Данный метод предназначен для выявления взаимосвязей между наборами данных из статистики. Поиск закономерностей осуществляется не на основе свойств анализируемого объекта, а между несколькими событиями, которые происходят одновременно. Наиболее известным алгоритмом решения задачи поиска ассоциативных правил является алгоритм ***Apriori***.

Впервые задача поиска ассоциативных правил (association rule mining) была предложена для нахождения типичных шаблонов покупок, совершаемых в супермаркетах, поэтому иногда ее еще называют анализом рыночной корзины (market basket analysis). Часто покупатели приобретают не один товар, а некоторый набор товаров. В большинстве случаев между этими товарами существует взаимосвязь. С помощью поиска ассоциативных правил мы найдем закономерности между связанными событиями в базах данных.

В работе приводится подробное описание двух наиболее известных алгоритмов поиска ассоциативных правил. А также решается практический пример нахождения ассоциативных правил на основе реальной статистики покупок товаров.

2 Основы ассоциативных правил в интеллектуальном анализе данных

Рассмотрим алгоритмы работы метода ассоциативных правил в интеллектуальном анализе данных. Приведем основные понятия, связанные с данным методом.

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.

2.1 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:

1. Присвоить k = 1 и выполнить отбор всех 1-элементных наборов, у которых поддержка больше минимально заданной пользователем Suppmin.
2. k = k + 1.
3. Если не удается создавать k-элементные наборы, то завершить алгоритм, иначе выполнить следующий 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.
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}.
6. Для каждого кандидата из C_k увеличить значение поддержки на единицу.
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.

Поиск закономерностей в классическом методе ассоциативных правил осуществляется между несколькими событиями, которые происходят одновременно.

2.2 Алгоритм AprioriSome

Алгоритм Apriori формирует частые последовательности-кандидаты всех возможных длин. Однако если из числа последовательностей определенной длины формируется мало частых последовательностей, то эту длину можно пропустить. Алгоритм ***AprioriSome*** использует как параметр длину последовательностей, анализируемых на предыдущем проходе, и возвращает длину последовательностей, которые будут анализироваться на следующем. Иными словами, длина последовательностей, искомых на следующем проходе, определяется длиной последовательностей, найденных на предыдущем.

Если обозначить номер прохода переменной t, то можно записать, что k(t+1)=k(t)+p. Это означает, что на следующем проходе будут анализироваться последовательности с длиной на p больше, чем на предыдущем. В случае если p=1, то алгоритм идентичен Apriori, т.е. будут анализироваться все последовательности. Задача заключается в том, чтобы определить, какие длины последовательностей могут быть пропущены.

Обозначим отношение числа частых k-последовательностей к числу всех k-последовательностей-кандидатов, т.е. $h_k = F_k/C_k$. Для выбора длины последовательности воспользуемся алгоритмом, приведенным на рис. 1.

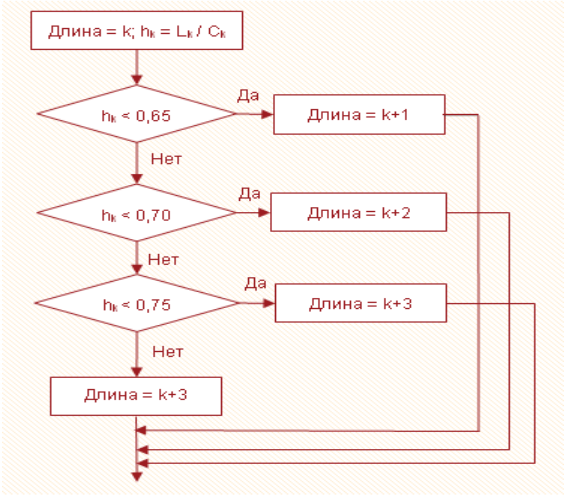

Рис 1. Алгоритм выбора длины последовательности в AprioriSome

3 Практический пример нахождения ассоциативных правил

Решим практическую задачу нахождения ассоциативных правил в статистике покупок продуктов. Одним из продуктовых розничных магазинов города Красноярска была предоставлена статистика покупок продуктов за октябрь 2012 года и февраль 2013 года. Всего статистика содержала около 1000 транзакций (чеков покупок). В статистике встречалось 50 различных наименований товаров: молоко, хлеб, сметана, творог, сыр, яйца, колбаса, сахар, различные виды мяса, полуфабрикаты и т.д. На основе статистики было необходимо найти закономерности между покупками в наборе данных.

Приведенные выше товары были обозначены переменными, в Таблице 1 приведены 13 самых часто встречающихся товаров.

Таблица 1. Список товаров из статистики покупок

Наименование	обозначение	поддержка
молоко	a	100
сметана	b	31
сыр твердый	c	44
мясо куриное	d	26
яйца куриные	e	26
полуфабрикат	f	31
хлею	g	125
конфеты	h	33
газировка	i	27
овощи	j	34
фрукты	k	52
консервы	l	28
соусы	m	31
крупа	n	32

Был определен уровень поддержки для данной задачи Suppmin=25. На первом этапе происходило формирование одноэлементных кандидатов. Согласно алгоритму были подсчитаны поддержки одноэлементных наборов. Наборы с уровнем поддержки меньше установленного отсекались. В нашем примере это были все товары, которые имели поддержку менее 25. Оставшиеся наборы товаров считаются часто встречающимися одноэлементными наборами товаров: это наборы a, b, c, d, e, …, n.

На следующем этапе выполнялось формирование двухэлементных кандидатов, подсчет их поддержки и отсечение наборов с уровнем поддержки, меньшим 25. В результате данного процесса остались шесть двухэлементных наборов, принимающих участие в дальнейшей работе алгоритма. Это пары ac, ag, ak, bg, cg, gk.

Далее формировались трехэлементные наборы товаров. Были получены два набора с поддержкой более 25: acg и agk (с поддержкой 25 и 26, соответственно).

Формирование четырехэлементных наборов не удалось, поэтому работа алгоритма была прекращена. Итак, в результате работы алгоритма Apriori было получено два ассоциативных правила: «При покупке молока и сыра, покупатель, скорее всего, купит хлеб» и «При покупке молока и хлеба, покупатель, скорее всего, купит фрукты».

Теперь рассмотрим работу алгоритма AprioriSome на той же самой статистике покупок товаров.

При первом проходе алгоритма по базе данных было найдено множество частых последовательностей с единичной длиной. На втором шаге было определено множество кандидатов C_2 для того, чтобы получить множество частых последовательностей F_2. На следующем шаге из F_2 было получено C_3. На четвертом шаге сформированное множество C_4 оказалось пустым. После чего первая фаза была закончена, и была запущена обратная фаза алгоритма. Из F_3 ничего не было исключено, поскольку более длинные последовательности отсутствуют в БД. В результате работы алгоритма были получены те же ассоциативные правила, что и при работе алгоритма Apriori.

Основное преимущество алгоритма AprioriSome над Apriori заключается в том, что он позволяет избежать вычисления большого числа немаксимальных последовательностей. Однако данное преимущество сокращается. Причина этого заключается в том, что кандидаты C_k в Apriori формируются с использованием L_{k-1}, а в AprioriSome – с использованием C_{k-1}. Поскольку C_{k-1} является подмножеством L_{k-1}, число кандидатов, формируемых алгоритмом AprioriSome, может быть больше.