РЕШЕНИЕ ЗАДАЧИ ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ КАК ЧАСТЬ АНАЛИТИЧЕСКОГО СЕРВИСА, ПРЕДОСТАВЛЯЮЩЕЙ РЕКОМЕНДАЦИИ

А. В. Усиков
Факультет математики и информатики, Вычислительные машины и системы, Гродненский государственный университет им. Я. Купалы
Гродно, Республика Беларусь
E-mail: andrew.usikov@gmail.com

В статье изложены основные принципы построения аналитического сервиса, связанного с интеллектуальной добычей данных и предоставлением прогнозов. Выделены требования к сервису и указаны преимущества многоуровневой архитектуры сервиса. Кратко описаны составляющие компоненты сервиса, их взаимодействие и организация.

Введение

Целью поиска ассоциативных правил (association rule) является нахождение закономерностей между связанными событиями в базах данных.

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).

Рыночная корзина - это набор товаров, приобретенных покупателем в рамках одной отдельно взятой транзакции.

Транзакции являются достаточно характерными операциями, ими, например, могут описываться результаты посещений различных магазинов. Транзакция - это множество событий, которые произошли одновременно. Регистрируя все бизнес-операции в течение всего времени своей деятельности, торговые компании накапливают огромные собрания транзакций. Каждая такая транзакция представляет собой набор товаров, купленных покупателем за один визит. Полученные в результате анализа шаблоны включают перечень товаров и число транзакций, которые содержат данные наборы.

Транзакционная или операционная база данных (Transaction database) представляет собой двумерную таблицу, которая состоит из номера транзакции (TID) и перечня покупок, приобретенных во время этой транзакции. TID - уникальный идентификатор, определяющий каждую сделку или транзакцию.

I. Масштабируемый алгоритм поиска ассоциативных правил

Ассоциативные правила позволяют находить закономерности между связанными событиями. Примером такого правила, служит утверждение, что покупатель, приобретающий «Хлеб», приобретет и «Молоко» с вероятностью 75%.

Задача нахождения ассоциативных правил разбивается на две подзадачи: 1. Нахождение всех наборов элементов, которые удовлетворяют порогу поддержки. Такие наборы элементов называются часто встречающимися. 2. Генерация правил из наборов элементов, найденных согласно п.1. с достоверностью, удовлетворяющей пороговому значению достоверности.

Одним из алгоритмов решения задач такого рода является алгоритм Apriori. Для того, чтобы было возможно применить алгоритм, необходимо провести предобработку данных: во-первых, привести все данные к бинарному виду; во-вторых, изменить структуру данных.

На первом шаге алгоритма подсчитываются 1-элементные часто встречающиеся наборы. Для этого необходимо пройтись по всему набору данных и подсчитать для них поддержку, т.е. сколько раз встречается в базе. Следующие шаги будут состоять из двух частей: генерации потенциально часто встречающихся наборов элементов (их называют кандидатами) и подсчета поддержки для кандидатов.

Подсчет поддержки для каждого кандидата происходит с помощью эффективного подхода, основанного на хранении кандидатов в хэш-дереве. Внутренние узлы дерева содержат хэш-таблицы с указателями на потомков, а листья – на кандидатов. Хэш-дерево строится каждый раз, когда формируются кандидаты. Первоначально дерево состоит только из корня, который является листом, и не содержит никаких кандидатов-наборов. Каждый раз когда формируется новый кандидат, он заносится в корень дерева и так до тех пор, пока количество кандидатов в корне-листе не превысит некоего порога. Как только количество кандидатов становится больше порогового значения, корень преобразуется в хэш-таблицу, т.е. становится внутренним узлом, и для него создаются потомки-листья.

После чего происходит подсчет поддержки для каждого кандидата. Для этого нужно «пропустить» каждую транзакцию через дерево и увеличить счетчики для тех кандидатов, чьи эле-

менты также содержатся и в транзакции. Кандидаты, для которых значения поддержки удовлетворяют минимальному пороговому значению, переносятся в разряд часто встречающихся.

Для подсчета достоверности правила достаточно знать поддержку самого набора и множества, лежащего в условии правила. Чтобы извлечь правило из часто встречающегося набора, следует найти все его непустые подмножества. И для каждого подмножества мы сможем сформулировать правило, если достоверность правила не меньше порога достоверности.

II. Разновидности алгоритма Apriori

В зависимости от размера самого длинного часто встречающегося набора алгоритм Apriori сканирует базу данных определенное количество раз. Разновидности алгоритма Apriori, являющиеся его оптимизацией, предложены для сокращения количества сканирований базы данных, количества наборов-кандидатов или того и другого. Были предложены следующие разновидности алгоритма Apriori: AprioriTID и AprioriHybrid.

AprioriTid. Интересная особенность этого алгоритма - то, что база данных D не используется для подсчета поддержки кандидатов набора товаров после первого прохода. С этой целью используется кодирование кандидатов, выполненное на предыдущих проходах. В последующих проходах размер закодированных наборов может быть намного меньше, чем база данных, и таким образом экономятся значительные ресурсы.

AprioriHybrid. Анализ времени работы алгоритмов Apriori и AprioriTid показывает, что в более ранних проходах Apriori добивается большего успеха, чем AprioriTid; однако AprioriTid работает лучше Apriori в более поздних проходах. Кроме того, они используют одну и ту же процедуру формирования наборов-кандидатов. Основанный на этом наблюдении, алгоритм AprioriHybrid предложен, чтобы объединить лучшие свойства алгоритмов Apriori и AprioriTid. AprioriHybrid использует алгоритм Apriori в начальных проходах и переходит к алгоритму AprioriTid, когда ожидается, что закодированный набор первоначального множества в конце прохода будет соответствовать возможностям памяти. Однако, переключение от Apriori до AprioriTid требует вовлечения дополнительных ресурсов.

Алгоритм DHP, также называемый алгоритмом хеширования. В основе его работы - вероятностный подсчет наборов-кандидатов, осуществляемый для сокращения числа подсчитываемых кандидатов на каждом этапе выполнения алгоритма Apriori. Сокращение обеспечивается за счет того, что каждый из k-элементных наборов-кандидатов помимо шага сокращения проходит шаг хеширования. В алгоритме на k-1 этапе во время выбора кандидата создается так называемая хеш-таблица. Каждая запись хеш-таблицы является счетчиком всех поддержек k-элементных наборов, которые соответствуют этой записи в хеш-таблице. Алгоритм использует эту информацию на этапе k для сокращения множества k-элементных наборов-кандидатов. После сокращения подмножества алгоритм может удалить набор-кандидат, если его значение в хеш-таблице меньше порогового значения, установленного для обеспечения.

Выводы

Спроектированный сервис будет полезен всем организациями, которые заинтересованы в получения аналитических сводок и другой ранее неизвестной информации по своим накопленным данным. Например, организации могут проводить анализ воздействия рекламы, сегментацию клиентов, поиск признаков прибыльных клиентов, анализ предпочтений товаров, прогнозирование объемов продаж и многое другое.

1. Кудрявцев, Ю. А. OLAP технологии: обзор решаемых задач и исследований / Ю. А. Кудрявцев // Бизнес-информатика. –2008. – № 1. – С. 66–70.
2. Усиков, А. В. Общие подходы создания и организации универсальной и гибкой архитектуры кли-ентской части веб-систем, специализирующихся на сборе и анализе различной ин-формации / А. В. Усиков , В. А. Ломакин // Наука–2012: сб. науч. ст. В 2 ч. Ч. 2. Гродно : ГрГУ. –2012. – С. 116–118.