УДК 004.942

ПРИМЕНЕНИЕ APACHE SPARK И APACHE HADOOP ДЛЯ АНАЛИЗА ИНФОРМАЦИИ О ЗАКАЗАХ ЕДЫ

А.В. Паньков, И.С. Хололович

В статье рассматривается задача анализа данных о заказах крупной системы управления ресторанным бизнесом с целью извлечения информации о поведении пользователей. Описывается необходимое программное обеспечение (Apache Spark, Apache Hadoop), а так же рассматривается применение алгоритмов интеллектуального анализа данных: алгоритм Apriori и алгоритм Frequent Pattern-Growth .

Введение

Большие данные, машинное обучение, интеллектуальный анализ данных - термины, которые можно всё чаще и чаще услышать в IT сообществе. Объемы данных современных компаний и организаций стремительно растут, и появляется огромное количество задач, связанных с анализом этих данных и извлечением из них полезной информации. Вместе с задачами, появляются и различные средства, предназначенные для их решения. К примеру, самыми обсуждаемыми и используемыми проектами на сегодняшний день являются Apache Hadoop, Apache Spark и Apache Storm.

Постановка задачи

Рассмотрим реально существующую систему, задачей которой является управление ресторанным бизнесом. Система работает по следующему принципу - пользователь заходит на сайт, осуществляет заказ еды и в назначенное время некоторые организации доставляют его. В системе зарегистрировано около 500 000 пользователей, которые в течение последних 4 лет сделали около 1 500 000 заказов, которые занимают уже около 5Гб памяти. К системе постоянно присоединяются новые рестораны, регистрируются новые пользователи и, соответственно, объемы данных имеют тенденцию к увеличению.

Необходимо провести анализ заказов и выделить пользователей, которые делают заказы с какой-либо частотой или периодичностью, с целью поддержания интереса к системе. Таким пользователям можно предоставлять скидки или бонусы, напоминать сделать заказ в удобное для них время, и т.д. Кроме того, можно найти пользователей, которые либо давно не пользовались сервисом, либо пользовались им только один-два раза, узнать причину их недовольства и попытаться ее устранить.

Таким образом, целью проведения анализа является поддержание интереса к системе и стремление не потерять потенциальных пользователей.

Выбор инструментов для проведения анализа

Python - универсальный язык, простой и интуитивно-понятный. Кроме того, Python является одним из лучших инструментов для работы с данными.

Для обработки данных используется такой инструмент как Apache Spark и его библиотека mllib. Он разрабатывался для параллельной обработки данных, и считается более эффективным по сравнению с Hadoop MapReduce. Хотя в ряде случаев Spark работает в 100 раз быстрее Hadoop, недостатком платформы является отсутствие у нее собственной системы распределённого хранения данных. Всвязи с этим в большинстве проектов, связанных с большим данным, Hadoop и Apache Spark устанавливаются вместе [1].

Для запуска Spark в режиме кластера, существует три подсистемы управления кластером:

1. Standalone (не предполагает установку дополнительного программного обеспечения и проведения дополнительной настройки ОС);
2. Apache Mesos;
3. Hadoop YARN.

На текущий момент только подсистема YARN поддерживает режим кластера с возможностью запуска Python приложений [2].

Данные системы о заказах хранятся в MongoDB (документо-ориентированная СУБД), поэтому логичным решением было бы запустить Spark Standalone кластер. Однако, учитывая сказанное выше, необходимо использовать Hadoop YARN.

Для того чтобы считывать данные напрямую из MongoDB, используется MongoDB коннектор для Hadoop. Это библиотека, которая позволяет использовать MongoDB как источник входных, так и выходных данных для задач, выполняющихся на Hadoop кластере.

Таким образом, для решения поставленной задачи выбрано следующее программное обеспечение (ПО):

1. Hadoop YARN;
2. Apache Spark;
3. MongoDB;
4. MongoDB connector for Hadoop.

Кроме того, на каждом узле кластера необходимо установить дополнительные инструменты, такие как Java, Scala, Python, SSH и pymongo_spark.

Алгоритмы поиска ассоциативных правил

Существует множество различных алгоритмов для анализа данных, однако, учитывая структуру исследуемых данных и результат, который необходимо получить, имеет смысл использовать алгоритмы поиска ассоциативных правил. Ассоциативные правила позволяют находить закономерности между связанными событиями. Примером такого правила может служить утверждение, что покупатель, приобретающий 'Хлеб', приобретет и 'Молоко' с вероятностью 0,72. Примерами ассоциативных правил для поставленной задачи также могут быть:

1. если сейчас вечер понедельника, то пользователь User сделает заказ с вероятностью 0,75;

2. если сегодня рабочий день, то пользователь User сделает заказ с вероятностью 0,8.

Ассоциативное правило $X => Y$ будет иметь поддержку s, если s% всех заказов удовлетворяют этому правилу.

Ассоциативное правило $X => Y$ будет иметь достоверность c, если c% заказов соответствующих X также соответствуют и Y.

Алгоритмы поиска ассоциативных правил предназначены для нахождения всех правил $X => Y$, причем поддержка и достоверность этих правил должны быть выше некоторых заранее определенных порогов, называемых соответственно минимальной поддержкой (minsupport) и минимальной достоверностью (minconfidence).

Одним из базовых и самых распространенных алгоритмов поиска ассоциативных правил является алгоритм Apriori [3]. Преимуществом данного алгоритма является использование свойства анти-монотонности: поддержка любого набора элементов не может превышать минимальной поддержки любого из его подмножеств. Смысл алгоритма, заключается в генерации всех возможных наборов (сначала одноэлементных, затем двухэлементных и т.д.), а затем исключении из них тех, которые имеют поддержку меньше заданной. Таким образом, используя инструменты и методы, предоставляемые Apache Spark, Apriori применим даже в случае, когда размер базы данных превышает объем оперативной памяти компьютера. Однако он подразумевает многократное прохождение по базе данных, что значительно увеличивает время работы.

Для уменьшения времени работы алгоритма, а так же минимизации количества обращений к базе данных, можно воспользоваться другим алгоритмом - Frequent Pattern-Growth (FPG) [5], который подразумевает только два обращения к БД. В основе метода лежит предварительная обработка базы транзакций, в процессе которой эта база данных преобразуется в компактную древовидную структуру, называемую Frequent-Pattern Tree - дерево популярных предметных наборов. 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. Некоторая последовательность узлов дерева является шаблоном. В дальнейшем FP-дерево используется для эффективного поиска частых предметных наборов.

Результаты

Исходные данные представляют собой информацию о заказе, а именно кто сделал заказ, когда, какой статус заказа, был ли это предварительный заказ, использовал ли пользователь систему доставки или сам забрал еду, информация о браузере пользователя, что конкретно он заказал, была ли предоставлена ему скидка, сумма заказа и др.

При предварительной обработке заказа извлекается вся информация о том, в какой день (день недели, рабочий или выходной), в какое время суток пользователь оформил заказ и т.д. Таким образом, ненужные данные

отбрасываются и в памяти хранится только информация с которой можно дальше работать. Эти полезные данные пропускаются через алгоритм и генерируются часто встречающиеся последовательности, например:

```
user@example.com, evening, not_weekend - support = 0,7 ,
```

откуда получаем ассоциативное правило:

если сегодня вечер рабочего дня, то пользователь user@example.com оформит заказ с вероятностью 0,7 .

После генерации часто встречающихся последовательностей, генерируются ассоциативные правила, удовлетворяющие минимальной поддержке.

Заключение

В ходе проделанных работ был запущен Hadoop YARN кластер, состоящих из трех ПК и установлено всё необходимое дополнительное ПО. На кластере можно запускать любые Spark приложения, а хранение данных можно осуществлять не только на HDFS, но и в MongoDB, т.к. настроена интеграция с данной БД. Результаты, полученные в ходе анализа работы алгоритмов, позволяют сделать вывод об оптимальности использования алгоритма FPG для получения необходимой информации, т.к. он использует меньшее количество ресурсов и экономит время.

Информация, полученная в ходе анализа заказов, позволяет выделить нужных нам пользователей и оповещать их о возможных скидках, акциях и предоставляемых бонусах, для того чтобы постоянно поддерживать интерес к пользованию системой. Кроме того, можно найти пользователей, которые делают заказы реже, и так же попытаться их заинтересовать, напомнить о системе.

В дальнейшем планируется использование этих знаний для предсказания того времени, когда обычно пользователь делает заказ. Если, к примеру, пользователь заказывает ужин каждый вторник в 19:00, то можно в 10:00 отправлять ему напоминание, а также периодически предоставлять бонусы. Таким образом, можно избежать отправления ненужных сообщений и отсылать пользователю только актуальную для него информацию.

Использование алгоритмов интеллектуального анализа данных позволяет оптимизировать работу маркетингового отдела предприятия и, как следствие, получать больший доход. Кроме того, полученные знания часто представляют из себя неочевидные закономерности, полезные при исследовании изучаемого бизнес-процесса.

Список литературы

1. Mebius.io - интернет-издание, посвящённое сфере BigData и Data Science [Электронный ресурс] / mebius.io - Режим доступа: https://mebius.io. - Дата доступа: 01.03.2016

2. Apache Spark - Lightning-fast cluster computing [Электронный ресурс] / The Apache Software Foundation - Режим доступа: http://spark.apache.org. - Дата доступа: 01.02.2016.
3. Fast Algorithms for Mining Association Rules in Large Databases: papers from the 20th International Conference on VLDB, Santiago, Chile, Sept. 1994 / Morgan Kaufmann Publishers; ed.: J. B. Bocca [et al.]. - Santiago: Morgan Kaufmann Publishers, 1994. - 487p.
4. Apriori - масштабируемый алгоритм поиска ассоциативных правил [Электронный ресурс] / BaseGroup Labs ООО «Аналитические технологии» - Режим доступа: https://basegroup.ru. - Дата доступа: 01.02.2016.
5. FPG - альтернативный алгоритм поиска ассоциативных правил [Электронный ресурс] / BaseGroup Labs ООО «Аналитические технологии» - Режим доступа: https://basegroup.ru. - Дата доступа: 01.02.2016.

Паньков Андрей Витальевич, доцент кафедры стохастического анализа и эконометрического моделирования Гродненского государственного университета, кандидат физико-математических наук, доцент, a.pankov@gmail.com

Хололович Ирина Сергеевна, студентка 5 курса Гродненского государственного университета, специальности экономическая кибернетика, iryna.khal@gmail.com