Санкт-Петербургский государственный университет

Кафедра информационно-аналитических систем

Группа 20Б.08-мм

Щека Дмитрий Вадимович

Реализация поиска аномалий в данных с помощью алгоритма поиска алгебраических ограничений BHUNT

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
Ассистент кафедры ИАС Чернышев Г.А.

Санкт-Петербург
2023

Оглавление

Введение

Аномалии [1] — это редкие случаи, что-то значительно отличающееся от общей массы. Их появление может оказаться следствием каких-то нарушений, поэтому обнаружение аномалий даёт шанс исправить или хотя бы найти подобные нарушения. Выявление аномалий также нужно для предварительной обработки данных, где эти аномалии будут удалены, чтобы, например, улучшить результаты предсказаний статистических моделей.

Алгоритм BHUNT [3] позволяет найти приближенные зависимости между столбцами в таблице данных. Записи, что не удовлетворяют такую зависимость, можно считать исключениями или аномалиями.

1. Постановка задачи

Целью работы является реализация поиска исключений в данных с помощью алгоритма BHUNT. Для достижения цели были поставлены следующие задачи:

- добавить в алгоритм BHUNT функциональность для изменения представления зависимостей;
- реализовать поиск исключений;
- провести тестирование.

2. Описание зависимости и исключения

Алгоритм поиска алгебраических зависимостей BHUNT уже был реализован для системы высокопроизводительного профилирования данных Desbordante [2]. С его результатами и будет производиться работа.

Зависимость в алгоритме BHUNT вычисляется для двух колонок. Для этого используется некоторая арифметическая операция $\oplus$, которая применяется к множеству пар значений из двух колонок. Алгоритм находит нечеткие зависимости, потому что выбирается только часть пар. В итоге имеется набор результатов операции $\oplus$. Затем результаты объединяются в непересекающиеся промежутки, что даёт нам зависимость вида $(column\_1 \oplus column\_2) \in [a_1,\, b_1] \cup [a_2,\, b_2] \cup ... \cup [a_n,\, b_n]$. Предположим у нас таблица доставки заказов из двух колонок. В первой колонке находится день отправки заказа, во второй — день его прибытия до места доставки. Пусть у нас тривиальное разбиение на пары (тривиальное означает, что в пару берутся значения из одной строки). Результаты $\oplus$ для всех строк отобразим на гистограмме Рис. 1.

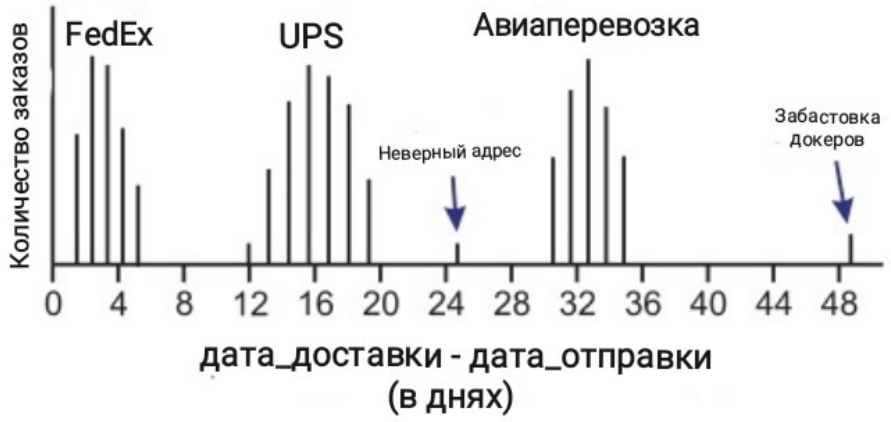

Рис. 1: гистограмма разностей, рисунок адаптирован из работы [3].

С правильно подобранными параметрами алгоритма результатом будут промежутки, которые своими концами совпадают с границами заметных «кочек». Получается, что записи, которые не попали в обозначенные промежутки являются исключениями. В данном конкретном примере даже приведены причины появляения таких аномалий: отправка на неверный адрес и забастовка в доках.

3. Дополнение к алгоритму BHUNT

Результатом алгоритма BHUNT являются наборы промежутков для каждой пары колонок, которые хранят в себе значения одинаковых числовых типов. На поиск исключений, естественно, влияют все параметры алгоритма. Поэтому, чтобы как-то изменить полученный набор исключений, приходится запускать поиск заново с новыми параметрами, что довольно ресурсозатратно. Одним из параметров, что сильно влияет на полученный набор исключений, является **weight**. Чем ближе значение к нулю, тем больше маленьких промежутков, чем ближе к единице, тем промежутки больше и их количество меньше. Для наглядности можно взглянуть на Рис. 2

Разбиение на промежутки можно проводить повторно уже после завершения работы алгоритма, если хранить все результаты операции $\oplus$ над значениями пар. Тогда можно запускать отдельно разбиение на промежутки для получения нового набора промежутков и, соответственно, исключений. Основные вычислительные затраты происходят на этапе проведения арифметических операций для значений пар. Поэтому, пропуская этот этап и изменяя только значение **weight**, мы можем получать новые результаты от алгоритма.

4. Поиск исключений

4.1. Новое определение исключений

До этого момента исключения представляли собой результаты операции $\oplus$, которые не попали в промежутки. Поскольку разбиение на пары тривиально (в пару попадают значения из одной строки), то можем рассматривать исключение для пары колонок как строку, из которой была взята пара значений, для которой результат операции $\oplus$ не попал в промежутки. Тогда для всей таблицы исключением могла бы является строка, в которой содержится хотя бы одна пара, являющаяся исключением для пары колонок. Тогда мы получаем набор индексов строк таблицы и пары колонок, для которых пара из этой строки стала исключением.

Значит исключение представляется в виде:

$row\_index, \{(column\_i, column\_k)\} \mid i \neq k$

Чем значительнее исключение, тем больше пар колонок в нём указано. Поэтому исключения можно возвращать либо в порядке их появления в таблице, либо отсортировав по количеству пар колонок.

4.2. Алгоритм поиска

Алгоритм поиска достаточно прост. Имея для каждой пары колонок промежутки, нужно для соответствующей пары колонок проверить, для каких пар значений результат операции $\oplus$ между элементами пар не попадает в промежутки. Поскольку исключения строятся для всей таблицы, приходится применять арифметическую операцию ко всем парам, что не попали в выборку. Поэтому для длинных таблиц поиск исключений может быть длительным.

Получив пары-исключения, мы добавляем в существующую строку-исключение новую пару колонок или создаём новую строку-исключение с одной парой колонок.

Затем строки-исключения можно отсортировать по количеству пар колонок или индексу.

4.3. Затруднения

К сожалению, так находятся не все строки, которые мы могли бы посчитать исключениями.

Причиной этому в первую очередь является принцип работы алгоритма BHUNT, основанный на случайной выборке. Если в эту выборку попадут интересующие нас исключения, то они будут участвовать в создании промежутков. Тогда, при большинстве значений параметров алгоритма, интересующие нас записи не будут считаться исключениями. Потенциально этого можно избежать, если подсчитывать количество значений, которые попадают в промежутки. Если таких значений мало, то промежуток можно удалить или сдвинуть его границы.

4.4. Тестирование

Тестирование проводилось на маленькой таблице, для которой возможно было провести ручные вычисления верных результатов. Также, поскольку алгоритм использует случайную выборку, был установлен фиксированный **seed**.

Также было протестировано пересоздание промежутков с изменённым **weight**.

Заключение

В ходе работы были выполнены следующие задачи:

- была адаптирована существующая функциональность алгоритма BHUNT для изменения представления зависимостей;
- был реализован поиск исключений;
- было произведено тестирование.

Можно выделить следующие направления продолжения работы:

- Обнаружение малозначительных промежутков и их удаление;
- Сортировка исключений по новым параметрам значимости.

Код доступен на Github[1].

[1] https://github.com/Mstrutov/Desbordante/pull/190

Список литературы

[1] Anomaly detection. — URL: https://en.wikipedia.org/wiki/Anomaly_detection.

[2] Desbordante github. — URL: https://github.com/Mstrutov/Desbordante.

[3] IBM. B-HUNT: Automatic Discovery of Fuzzy Algebraic Constraints in Relational Data. — 2003. — URL: https://people.cs.umass.edu/~phaas/files/vldb2003.pdf.

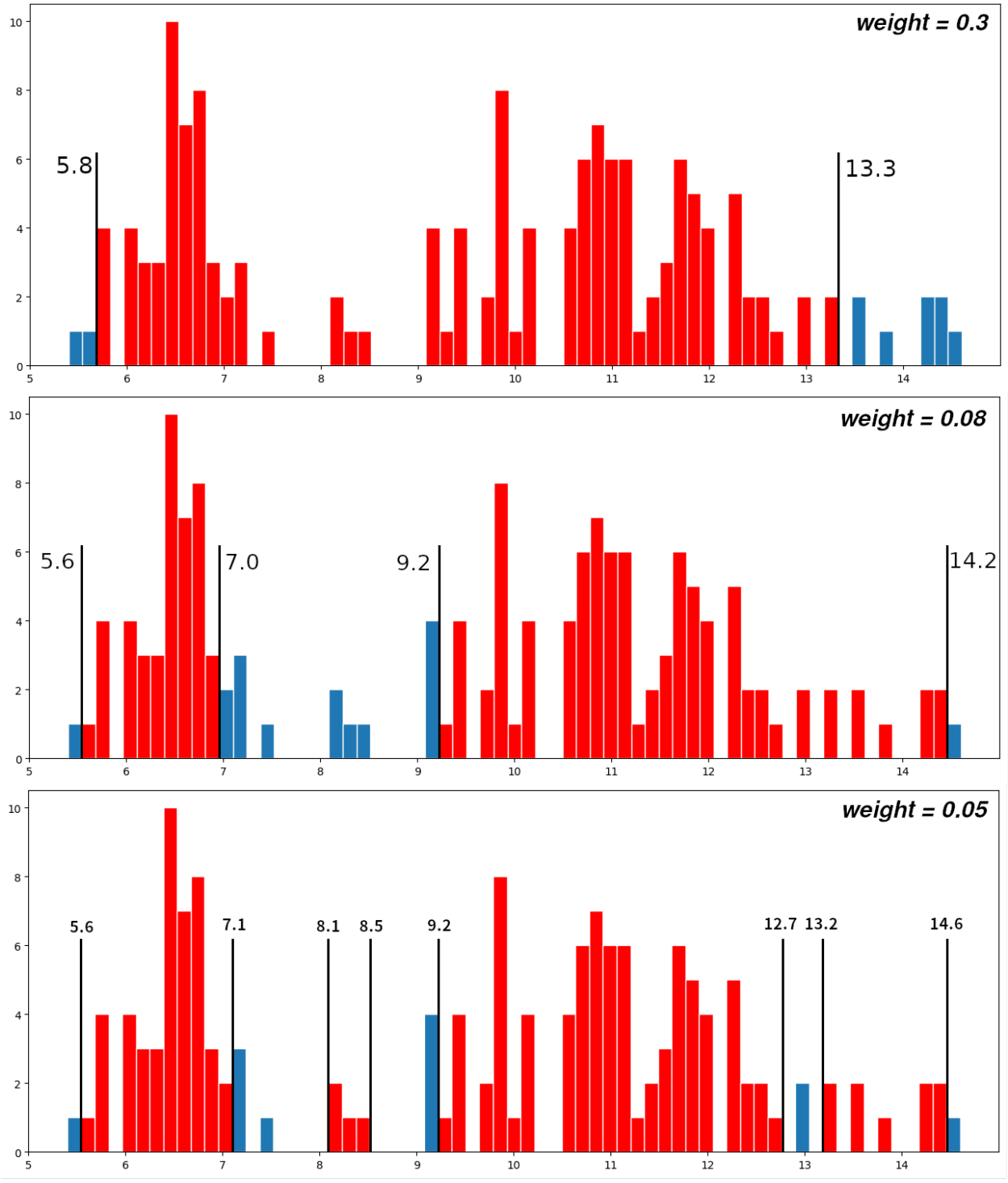

Рис. 2: разбиение результатов $\oplus$ на промежутки в зависимости от **weight**