Extendable System for Multicriterial Outlier Detection

Vladislav D. Dineev[0000-0002-0218-0338], Victor A. Dudarev[0000-0001-7243-9096]

HSE University, 109028, Russia
vddineev@edu.hse.ru

Annotation. The article is devoted to developing a convenient extendable software system for outliers detection in data. The developed information system is based on the use of statistical analysis and machine learning methods to find suspicious values in data and the use of Web technologies and micro-service architecture to implement the user interface and system extensibility. As a development result, a software system was implemented that can analyze multidimensional numerical data and find outliers in them using a set of customizable analysis methods with the opportunity to vote algorithms. New algorithms could be easily added to the system as microservices interacting with the parent Web-service. End users can access the system through the Web application using any Web browser. The developed system can be used in data analysis and to process experimental results, which can potentially contain errors. This delivers the necessary degree of automation for an expert analyzing the data correctness.

Keywords: outlier detection, data analysis, data cleansing, Web-technologies.

Расширяемая система для многокритериального поиска выбросов в данных

Динеев В.Д.[0000-0002-0218-0338], Дударев В.А.[0000-0001-7243-9096]

Национальный исследовательский университет «Высшая школа экономики», 109028, Россия
vddineev@edu.hse.ru

Аннотация. Статья посвящена созданию удобного расширяемого инструмента для поиска выбросов в данных. Разработанная информационная система базируется на использовании методов статистического анализа и машинного обучения для поиска выбросов в данных и применении Веб-технологий и микросервисной архитектуры для реализации пользовательского интерфейса и расширяемости системы. В результате разработки получен программный инструмент, способный анализировать многомерные числовые данные и находить в них выбросы с помощью набора настраиваемых

Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-2790/paper10.pdf

методов анализа с возможностью голосования алгоритмов. Новые алгоритмы добавляются в состав системы в качестве микросервисов, взаимодействующих с родительским сервисом. Доступ конечных пользователей к системе может осуществляться через Веб-приложение с помощью Веб-браузера. Разработанный инструмент может найти применение при анализе данных и результатов экспериментов, в которых потенциально могут содержаться ошибки, добавляя необходимую степень автоматизации для эксперта, анализирующего корректность данных.

Ключевые слова: поиск выбросов в данных, анализ данных, очистка данных, Веб-технологии.

1 Введение

Поиск выбросов в данных [1] на сегодняшний день используется во многих областях: от обнаружения неисправностей в показаниях приборов до обнаружения подозрительной активности клиентов банков. Для решения задач данного класса существует множество широко используемых, проверенных временем методов, таких как статистические критерии и методы машинного обучения.

Существующие наиболее популярные инструменты поиска выбросов в основном представляют собой платные приложения (например STATISTICA [2]), которые устанавливаются на персональный компьютер. Несмотря на обширный функционал, такие приложения ограничивают пользователя в выборе технических и программных средств для использования данных решений, при этом требуя от пользователя покупки дополнительного, возможно, не нужного ему функционала. Существуют и бесплатные open-source решения, которые, на наш взгляд, являются пригодными только для потребителей, хорошо разбирающихся в ИТ [3]. Часто отмечается разработка авторских методов, нацеленных на учет специфики данных в конкретной предметной области и последующее написание соответствующих проприетарных систем [4]. В связи с этим возникает актуальность разработки открытого модульного приложения, которое будет поддерживаться большинством существующих на данный момент операционных систем, распространяться бесплатно и доступно пользователям из прикладных областей без специальной математической подготовки или знаний в ИТ.

Для поддержки кроссплатформенности, то есть обеспечения запуска на большинстве систем, оптимальным подходом является предоставление необходимой функциональности в виде Веб-сервиса (для программных средств) и Веб-приложения (для конечных пользователей) с доступом из сети интернет, для чего в работе были использованы Веб-технологии [5].

Поскольку существует большое количество различных алгоритмов поиска выбросов и процедур коллективного принятия решений, было решено применить микросервисную архитектуру [6], чтобы не привязывать реализацию конкретных алгоритмов к какому-либо языку программирования и обеспечить легкую расширяемость системы, то есть предоставить пользователю возможность добавлять при необходимости новые алгоритмы в виде Веб-сервисов в систему.

2 Методы

2.1 Статистические критерии

Статистические критерии в большинстве своем основываются на вычислении значения критерия для наибольшего или наименьшего значения выборки. После этого вычисленное значение сравнивается с табличным, на основании чего делается вывод о том, является ли число выбросом.

Критерий Граббса. Критерий Граббса основывается на предположении, что выборка, поступающая для анализа, подчиняется закону о нормальном распределении [7]. Для его применения вычисляется значение критерия для наибольшего по модулю элемента x_i в выборке по следующей формуле:

$$K_{\text{расчетное}} = \frac{|x_i - \bar{x}|}{\sigma} \quad (1)$$

где x_i −проверяемое значение,

$\bar{x}$ – выборочное среднее,

σ – среднеквадратичное отклонение.

Если $K_{\text{расчетное}} > K_{\text{табличное}}$ для заданного уровня значимости, то x_i считается выбросом.

Правило трёх сигм. Для нормально распределенных величин часто применяется классическое правило трёх сигм, которое утверждает, что вероятность отклонения любой такой величины от своего среднего значения на величину, меньшую, чем три среднеквадратичным отклонения, приблизительно равна 99.7% [8]:

$$P(-3\sigma < x_j < 3\sigma) = 0{,}997 \quad (2)$$

где x_j – проверяемое значение,

σ – среднеквадратичное отклонение.

Таким образом, если предполагать, что входные данные подчиняются закону о нормальном распределении, то данный критерий можно использовать для нахождения в них выбросов.

Тест Диксона. Тест Диксона используется для проверки минимального или максимального значения выборки на выброс для размера выборок менее 30.

Для применения критерия Диксона значения выстраиваются в порядке неубывания, то есть $x_1 \leq x_2 \leq x_2, \leq \cdots, \leq x_n$, затем, вычисляется значения критерия для наибольшего значения [9]:

$$K_{max} = \frac{x_n - x_{n-1}}{x_n - x_1} \quad (2)$$

или для наименьшего:

$$K_{min} = \frac{x_2 - x_1}{x_n - x_1} \quad (3)$$

Если $K_{\text{расчетный}} > K_{\text{табличный}}$, то, соответственно, минимальное или максимальное значение считается выбросом.

Критерий Шовене. Критерий Шовене заключается в том, что вычисляется вероятность получения числа, отклоняющегося от среднего выборочного больше, чем $x_{\text{сомн}}$. Для этого сначала по сомнительному значению находят значение интегральной функции нормального распределения F(x) с параметрами, рассчитанными по выборке [10]. Далее, если сомнительно максимальное значение вариационного ряда, указанную вероятность $P_{\text{прев}}$ находят по формуле:

$$P_{\text{прев}} = \big(1 - F(x)\big) \cdot 2 \tag{4}$$

Если же сомнительно минимальное значение, то $P_{\text{прев}}$ находят по формуле:

$$P_{\text{прев}} = F(x) \cdot 2 \tag{5}$$

Умножая $P_{\text{прев}}$ на объём испытаний n, получают ожидаемое число результатов N, отклоняющегося от среднего значения выборки больше, чем сомнительное значение:

$$N = P_{\text{прев}} \cdot n \tag{6}$$

Если $N < 0.5$, то сомнительное значение считают выбросом.

2.2 Методы Машинного Обучения

Isolation Forest. Данный алгоритм “изолирует” значения, случайно выбирая признак и затем случайно выбирая разделяющее значение в интервале между минимальным и максимальным значением данного признака [11].

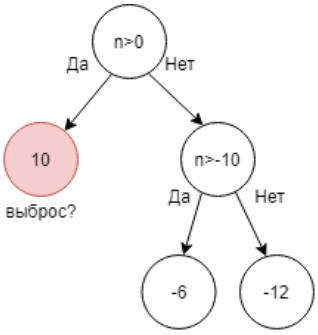

Рисунок 1. Пример дерева разделений.

Поскольку рекурсивное разделение можно представить в виде дерева (рис. 1), число разделений, которые требуются для изолирования объекта эквивалентно длине пути от корня до завершающего узла.

Эта длина, усредненная по лесу таких случайных деревьев, является величиной нормальности и функцией принятия решения.

Случайное разделение генерирует намного более короткие пути для выбросов. Поэтому, когда лес случайных деревьев коллективно генерирует короткие пути для одних и тех же значений, эти значения скорее всего являются выбросами.

Local Outlier Factor. Локальный уровень выброса рассчитывает локальную плотность для каждого объекта. Локальность определяется *k*-ближайшими соседями, до которых рассчитывается расстояние для вычисления плотности.

Сравнивая плотности объектов друг с другом (рис. 2), можно выделить объекты, локальная плотность которых намного меньше, чем других. Такие объекты считаются выбросами [12].

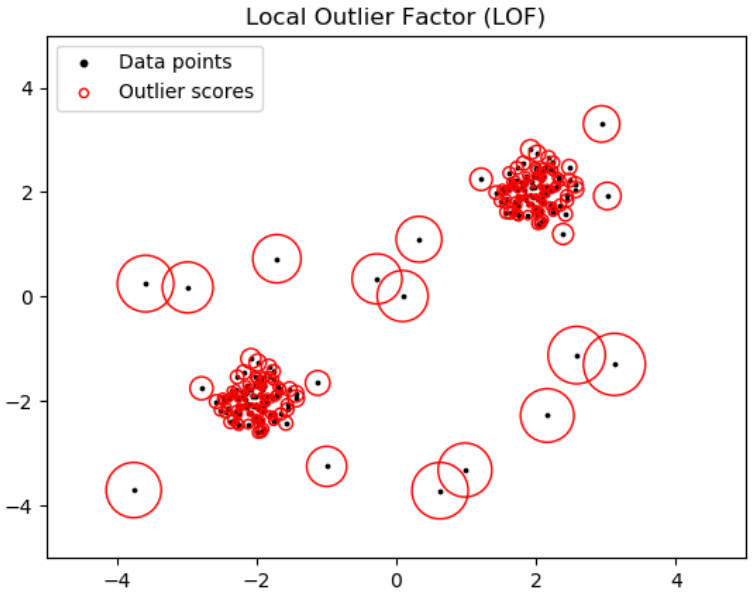

Рисунок 2. Пример локальных плотностей значений.

One Class SVM. One Class SVM располагает значения в пространстве и пытается найти такую функцию, которая была бы положительна на нормальных значениях и отрицательна на выбросах. Иными словами, данный алгоритм строит гиперплоскость, чтобы разделить значения на выбросы и не выбросы [13].

2.3 Процедуры коллективного принятия решения

Для получения более достоверного результата при использовании нескольких алгоритмов поиска выбросов используются такие комбинации результатов, как среднее значение и голосование по большинству.

Процесс голосования по большинству подсчитывает количество голосов за каждую из категорий (выброс или нормальное значение) для конкретного значения в выборке и на основании большинства голосов делается вывод о принадлежности данного значения к аномальному.

При использовании среднего значения итоговый результат для каждого из элементов выборки вычисляется как среднее арифметическое вероятностей выбросов, возвращаемых алгоритмами (где есть два предельных случая: выброс обозначается единицей, нормальное значение – нулём). Отличие от голосования по большинству заключается в возможности учесть степень “уверенности” классификатора.

3 Описание разработанной системы

3.1 Архитектура системы

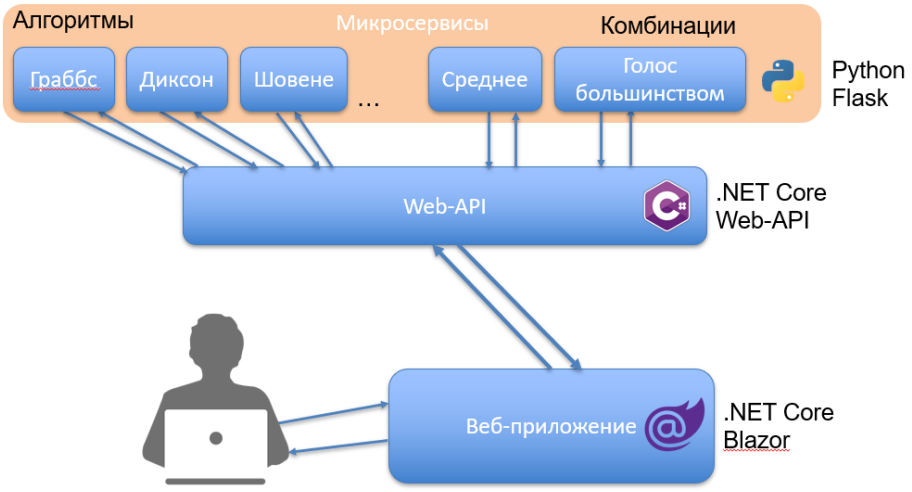

Рисунок 3. Архитектура системы.

Разработанный инструмент построен на основе архитектуры микросервисов (рис. 3). Каждый модуль реализуется в виде самостоятельного микросервиса, доступного через REST API и выполняющего вычисления над заданными оболочкой входными данными. Микросервис, реализующий отдельный вычислительный метод, конфигурируется с помощью JSON-документа, который состоит из

набора специфичных для реализуемого математического метода параметров. В качестве краткого примера приведем фрагменты файлов config.json, содержащих параметры по умолчанию для модулей поиска выбросов по методу Граббса и SVM (формат документа определяется индивидуально для каждого модуля в соответствии с настраиваемыми параметрами математического метода):

```
{
  "double": {
    "alpha": {
      "type": "double",
      "min": 0.01,
      "max": 1,
      "default": 0.01,
      "fullName": "Significance
level"
    }
  },
  "select": {
    "method": {
      "fullName": "Grubbs
method",
      "type": "select",
      "options": [
        "double_sided",
        "left_sided",
        "right_sided"
      ]
    }
  }
}
```

```
{
  "select": {
    "kernel": {
      "default":
"rbf",
      "options": [
        "rbf", "lin-
ear", "poly", "sig-
moid"
      ]
    }
  },
  "int": {
    "degree": {
      "default": 3,
      "min": 1,
      "max": 10
    }
  }
}
```

Модули разрабатываются в “минималистическом” виде и не поддерживают состояние (stateless). Каждый модуль реализует один алгоритм: выполняет вычисления над входными данными и возвращает результат.

Архитектура позволяет легко добавлять новые модули и тем самым расширять набор доступных вычислительных методов, что дает большую гибкость, учитывая, что различные модули могут разворачиваться на разных серверах. Это подразумевает возможность написания модулей на любых языках программирования и с использованием любых платформ. Единственное требование – доступность модуля с помощью API через протокол HTTP(S). В настоящее время, все модули (написанные на Python) функционируют под управлением Microsoft IIS в рамках ASP .Net Core 3.1 приложения на одной виртуальной машине, но в будущем, они могут быть легко перенесены (распределены) при необходимости на несколько виртуальных машин, например, для балансировки нагрузки.

Результаты вычислений, полученные от расчетных модулей, обрабатываются дополнительным сервисом, доступным через Web API. Для взаимодействия с ним разработано отдельное Веб-приложение, для построения отчета о решении

задачи различными методами или высокоуровневым методом принятия коллективного решения. Веб-приложение содержит список активных модулей в конфигурационном файле, который легко позволяет добавить модуль или изменить его настройки (все URI в настоящее время относительные, что иллюстрирует работу подмодулей в рамках одного приложения, но в будущем они могут быть переконфигурированы и вынесены на отдельные компьютеры с абсолютными ссылками в URI):

```
{
  "algorithms": {
    "grubbs": {
      "type": "outlier",
      "uri": "/algorithms/grubbs",
      "fullName": "Grubbs criterion"
    },
    "svm": {
      "type": "classification",
      "uri": "/algorithms/svm",
      "fullName": "SVM"
    },
...
  },
  "combinations": {
    "average": {
      "uri": "/combinations/average",
      "fullName": "Average value"
    },
    "majority": {
      "uri": "/combinations/majority",
      "fullName": "Voting by majority"
    },
...
  }
}
```

Отметим, что предусмотрена индивидуальная настройка параметров используемых алгоритмов (сервисов), в том числе при формировании запроса пользователем.

3.2 Пользовательский интерфейс

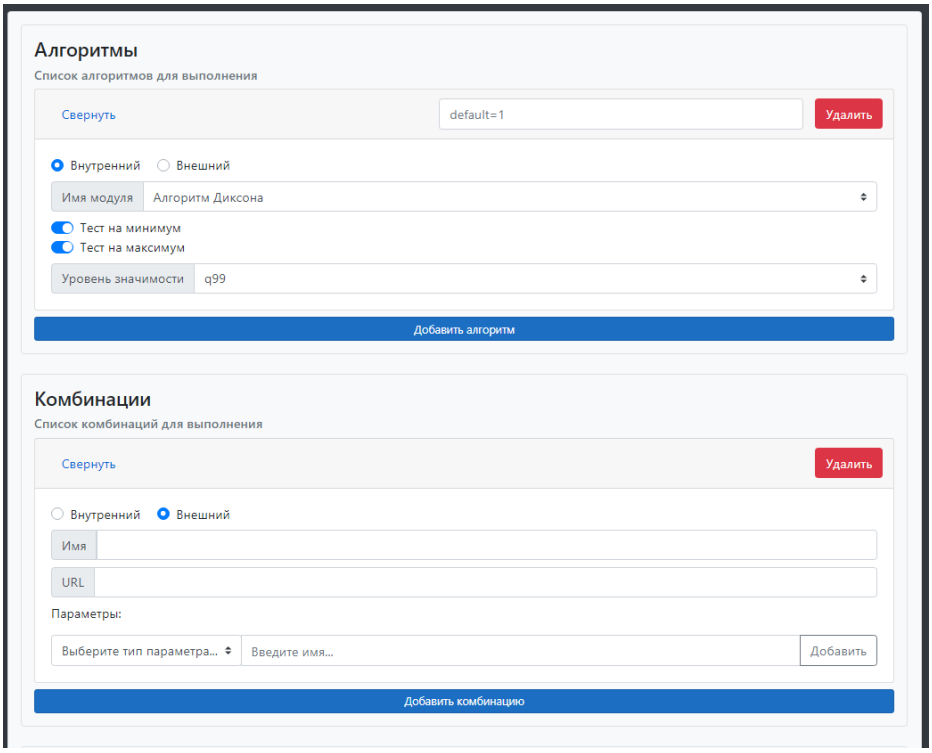

Рисунок 4. Пользовательский интерфейс Веб-приложения.

Веб-интерфейс приложения построен по принципу редактирования и отправки запроса к API с помощью интерактивных форм (рис. 4). Поскольку размер данных может оказаться достаточно большим, чтобы вводить его вручную, а затем просматривать в окне браузера, была разработана поддержка импорта входных данных и экспорта результатов работы инструмента, соответственно. Для этого используется CSV-формат файлов, поскольку его достаточно для представления простых табличных данных.

4 Применение системы на примере поиска выбросов в обучающей выборке задачи из неорганической химии

Приведем пример использования разработанной системы для решения задачи поиска выбросов (ошибок) в обучающей выборке для прогнозирования возможности образования и типа кристаллической структуры соединений состава $A^{2+}_2 B^{+3}C^{+5}O_6$. Обучающая выборка состоит из 551 прецедента, характеризующегося

вектором размера 105 (или точкой в 105-мерном пространстве признаков), отнесенного к одному из 11 классов (от отсутствия соединения до образования соединений со специфическими кристаллическими структурами).

При использовании системы возможен не только выбор алгоритмов, участвующих в анализе данных, но и применение простейших коллективных алгоритмов типа голосования по большинству или усреднения результата отдельных алгоритмов. Более того, можно выбрать режим работы программы, чтобы она показывала бинарный ответ (является или нет объект выбросом) или же оценивала вероятность этого события. При анализе обучающей выборки на предмет выбросов мы использовали усреднение бинарного ответа хорошо зарекомендовавших себя алгоритмов Isolation Forest и Local Outlier Factor.

В исследуемой задаче найдено пять “подозрительных” объектов-соединений, класс которых, по мнению всего коллектива методов, является ошибочным и подлежит тщательной проверке специалистом-предметником (на предмет возможной ошибки в классификации объектов обучающей выборки): Ba_2LaPuO_6 – скорее всего не имеет кристаллическую структуру K_3FeF_6-I, пр.гр. Fm3(-)m, Z=4; Ba_2RhTaO_6 и Ba_2RhUO_6 - не относятся к типу $BaTiO_3$(V), пр.гр. $P6_3/mmc$, Z=6; Sr_2RhTaO_6 – не относится к типу I4/m; а отсутствие соединения для CaO-Rh_2O_3-Ir_2O_5 указано ошибочно. Таким образом, использование программы позволяет существенно сузить количество объектов для проверки на выбросы (нужно проверить 5 из 551). При более жестком контроле выбросов дополнительной экспертизе могут подвергаться прецеденты, на которых хотя бы один из методов показал возможность выброса (за исключением 5 вышеуказанных, дополнительной проверке подлежат еще 58 прецедентов). Однако, в любом случае, даже проверки 63 объектов обучающей выборки является гораздо менее трудоемкой нежели ручная проверка всех 551 прецедентов.

5 Заключение

В результате было разработано расширяемое Веб-приложение, основанное на микросервисной архитектуре (исходный код доступен по адресу https://github.com/dineev-vd/MultiCriteriaOutlierSearch, развернуто в тестовом режиме по адресу http://outliers.imet-db.ru/outliers), которое способно анализировать данные на наличие в них выбросов с помощью различных алгоритмов, а затем объединять результаты их работы с помощью процедур коллективного принятия решений для повышения точности и надежности анализа.

Данный инструмент можно использовать для анализа числовых данных и выделения из них “подозрительных”, относящихся с большой вероятностью к категории аномальных. При этом окончательное решение должен принимать эксперт на основании учета всех особенностей входных данных и предметной области.

Работа выполнена при частичной финансовой поддержке РФФИ, проект 18-07-00080.

Список литературы

1. Zimek A., Schubert E.: Outlier Detection // Encyclopedia of Database Systems. — Springer New York (2017). DOI: https://doi.org/10.1007/978-1-4899-7993-3_80719-1
2. Боровиков, В.: STATISTICA. Искусство анализа данных на компьютере: Для профессионалов: 2-е изд. СПб: Питер (2003).
3. Flach, M., Gans, F., Brenning, A., Denzler, J., Reichstein, M., Rodner, E., Bathiany, S., et al.: Multivariate anomaly detection for Earth observations: a comparison of algorithms and feature extraction techniques, Earth Syst. Dynam., 8, 677–696, https://doi.org/10.5194/esd-8-677-2017, (2017).
4. Ожерельев, И.С., Сенько, О. В., Киселева, Н.Н.: Метод поиска выпадающих объектов с использованием параметров неустойчивости обучения // Системы и средства информатики. - 2019. - Т.29. - N.2. - С.122-134.
5. Чамберс, Д., Пэкетт, Д., Тиммс, С.: ASP.NET Core. Разработка приложений. – СПб.: Питер (2018).
6. Ричардсон, К.: Микросервисы. Паттерны разработки и рефакторинга. – СПб.: Питер (2019).
7. Лемешко, Б. Ю., Лемешко, С. Б.: Расширение области применения критериев типа Граббса, используемых при отбраковке аномальных измерений // Измерительная техника (2005).
8. Пискунов, Н. С.: Дифференциальное и интегральное исчисления для втузов, т. 2: Учебное пособие для втузов. — 13-е изд.— М.: Наука, Главная редакция физико-математической литературы (1985).
9. Сергеев, А. Г., Крохин, В. В.: Метрология: Учебное пособие для вузов. М.: Логос (2000).
10. Тейлор, Дж.: Введение в теорию ошибок. Пер. с англ. – М.: Мир (1985).
11. Isolation Forest, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html (Дата обращения: 30.05.2020).
12. Local Outlier Factor, https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html (Дата обращения: 30.05.2020).
13. Support Vector Machines, https://scikit-learn.org/stable/modules/svm.html (Дата обращения: 30.05.2020).