Обзор методов очистки данных для машинного обучения

А.В. Макаров, Д.Е. Намиот

***Аннотация* — В последние несколько лет модели машинного обучения и нейросети начали активно внедряться в повседневную жизнь. Основными параметрами при их обучении являются точность и эффективность. Один из главных этапов, который позволяет улучшить эти показатели, заключается в подготовке набора данных. Перед применением любого метода необходимо произвести предварительную очистку данных, так как иначе полученные результаты могут быть неточными или некорректными. Даже несмотря на то, что начинающие исследователи подготавливают наборы данных, зачастую очистка производится некорректно или неэффективно с множеством ошибок. В данной статье представлен обзор основных методов, рассмотрены их достоинства и недостатки, а таже даны общие рекомендации, позволяющие улучшить процесс очистки данных. Помимо этого, особое внимание уделено важности умения пользоваться различными инструментами для очистки данных. Рассмотрены основные библиотеки, такие как Pandas, scikit-learn и NumPy, специализированные программы типа OpenRefine, различные возможности языка R, а также методы нормализации, стандартизации и обработки текстовых данных. Правильное использование инструментов для очистки данных существенно влияет на качество анализа и моделирования, способствуя более точным и надежным результатам.**

***Ключевые слова* — машинное обучение, нейросети, очистка данных, инструменты для очистки данных.**

I. Введение

Технологические достижения последних нескольких лет положили начало эпохе моделей машинного обучения и нейросетей. Они играют важную роль в науке и бизнесе. Модели машинного обучения позволяют извлекать ценную информацию из больших объемов данных, обрабатывать ее, находить скрытые закономерности и использовать эти знания для принятия решений и разработки новых продуктов и услуг [1]. Например, нейросети используют для распознавания образов и голосов, перевода между разными языками, предсказания цен на акции, анализа клиентского поведения, медицинской диагностики и автоматического управления транспортом [2].

Большинству компаний становится все проще хранить и обрабатывать большие объемы данных. Они помогают бизнесу производить более точную аналитику и эффективнее принимать решения.

Статья получена 9 сентября 2023.
Макаров А.В. – МГУ имени М.В. Ломоносова (email: archie1602@hotmail.com). Намиот Д.Е – МГУ имени М.В. Ломоносова (email: dnamiot@gmail.com).

В большинстве образовательных задач используются чистые наборы данных, например, такие как MNIST [3] или CIFAR-10 [4]. Однако в реальности наборы данных зачастую являются «грязными», так как содержат различные ошибки. Их примерами могут быть пропущенные значения, опечатки, дубликаты, ошибки, связанные с форматированием, например, некорректно записанные даты и денежные единицы.

«Чистота» и точность данных являются ключевыми факторами при обучении модели и нейросети, так как использование ошибочных данных может привести к неправильным прогнозам и повлиять на исследование в целом.

Процесс очистки данных (data cleaning) включает в себя удаление дубликатов, обработку пропущенных значений, удаление выбросов, корректировку ошибок в данных. Целью очистки является создание набора данных, который является точным, согласованным и полным.

В настоящей статье осуществляется обзор различных методов и рекомендаций для очистки данных. Работа выполнена на основе изучения и анализа опубликованных в последние годы статей по данной тематике.

В разделе II даны общие рекомендации по очистке данных. В разделе III описаны основные методы, позволяющие осуществить процесс очистки данных. В разделе IV рассмотрены основные инструменты для подготовки данных.

II. Обзор общих рекомендаций по очистке данных

Прежде чем приступить к очистке, сначала необходимо произвести исследовательский анализ данных (exploratory data analysis). На этом шаге предлагается ознакомиться с типом данных, а также осуществить их визуализацию с использованием различных библиотек, таких как matplotlib [5], plotly [6], ggplot2 [7].

Следующим шагом является очистка форматирования. Если данные взяты из различных источников, то вполне вероятно, что они имеют разное форматирование. Это может привести к тому, что модели машинного обучения не смогут их обработать. Именно поэтому нужно удалить любое форматирование, которое было применено к данным. Это можно сделать с помощью таких инструментов, как Excel [8] или Google таблицы [9].

Немаловажным этапом является приведение всех записей в наборе данных к одному языку. Это связано с

тем, что модели машинного обучения в основном работают только с одним языком.

Преобразование типов данных является одним из самых важных этапов при очистке. В большинстве случаев, набор данных состоит из чисел, которые часто представляются, как текст. Из-за этого алгоритмы машинного обучения не могут применять над ними математические действия. Именно поэтому очень важно произвести преобразование типов и осуществить конвертацию чисел в соответствующие числовые типы данных.

Разведочный анализ и очистка данных являются обязательными компонентами доверенных плафтформ машинного обучения [99].

III. Обзор различных методов очистки данных

A. *Обработка пропущенных значений*

При работе с реальными данными, например, полученными из биомедицинских источников [10], пропущенные значения являются одной из наиболее распространенных проблем, возникающих во время очистки. В наборе данных отсутствие значения обычно указывается как NA (Not Available). Обработка пропущенных значений может быть выполнена различными способами в зависимости от типа данных и целей анализа. Одним из распространенных методов обработки пустых значений является заполнение недостающих данных новыми значениями. Рассмотрим несколько подходов:

1) Удаление строк с пустыми значениями

Если пустые значения не являются критически важными для исследования, тогда можно удалить строки их содержащие. Несмотря на предельную простоту, данный метод может оказаться неэффективным в случае, когда процент отсутствующих данных слишком высок. К тому же, у него есть недостаток, заключающийся в игнорировании возможных связей между переменными, что может привести к потере информации и внести систематическую ошибку [11]. Более сложные подходы позволяют заменить отсутствующие значения правдоподобными, полученными на основе имеющихся данных.

2) Заполнение пустых значений

Пропущенные значения можно попытаться заменить средним или медианным значением. Этот метод может быть применен для заполнения пустых значений числовых переменных, например, для заполнения пустых значений возраста пациентов или цены на товары. Стоит отметить, что данный метод может быть неэффективен в случае, если пустые значения не распределены равномерно в наборе данных, что может привести к низкой точности и искажению распределения переменной [12]. К тому же, этот подход может быть неприменим для категориальных переменных, так как они не могут быть представлены в виде числовых значений.

3) Использование алгоритмов машинного обучения

Для заполнения пропущенных значений также применяются методы машинного обучения. Например, можно использовать линейную регрессию [13, 14], чтобы заполнить пустые значения на основе других переменных в наборе данных или метод k-ближайших соседей [15, 16]. Этот метод может быть эффективен, если пустые значения имеют сложную зависимость с другими переменными в наборе данных. К недостаткам данного метода можно отнести его трудозатраты в вычислительном и временном отношении. Также стоит отметить, что если модель недостаточно точна и не учитывает все возможные варианты заполнения значений, то результаты исследования могут получиться искаженными.

Каждый из описанных методов имеет свои достоинства и недостатки. Выбор метода должен зависеть от конкретных характеристик и требований к набору данных.

B. *Удаление выбросов*

Выброс (outliers) – это значение, которое не согласуется с остальной частью набора данных. Аномалии могут появляться по разным причинам. Например, это может быть связано с человеческим фактором при внесении данных или неточностью измерительных приборов. Цель обнаружения выбросов – удалить записи, которые действительно выбиваются из общей массы, чтобы построить более точно работающую модель. Рассмотрим несколько методов, которые помогут определить аномалию в данных.

1) Метод на основе стандартного отклонения

Если записи в наборе данных следуют нормальному распределению [17], то для обнаружения выбросов можно использовать стандартное отклонение. Эта величина измеряет разброс данных вокруг среднего и, по сути, показывает, насколько далеко от него находятся точки. Чем отклонение больше, тем больше разброс значений. Согласно правилу трёх сигм [18] для данных с нормальным распределением в пределах одного среднеквадратического отклонения лежит 68,26% значений. Около 95,44% и 99,72% данных находятся в пределах двух и трёх среднеквадратических отклонений соответственно.

Обозначим стандартное отклонение распределения через σ, а среднее значение через μ. Данный метод заключается в том, чтобы установить нижний предел на три стандартных отклонения ниже среднего ($\mu - 3\sigma$), а верхний на три выше ($\mu + 3\sigma$). Выбросом при таком подходе является любая точка данных, выходящая за пределы этого диапазона.

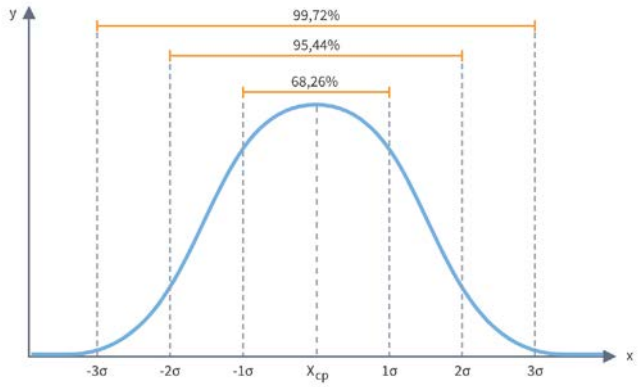

Рис 1. Иллюстрация правила трех сигм для данных с нормальным распределением

И поскольку 99,72% записей обычно находятся в пределах трёх стандартных отклонений, то количество выбивающихся из общей массы точек будет близко к 0,28% от размера набора данных. Минусом данного метода является то, что его можно применять только в том случае, если данные в наборе соответствуют нормальному распределению.

2) Метод Z-оценки

Z-оценка для нормального распределения с средним значением μ и стандартным отклонением σ определяется по формуле $z = (x - \mu)/\sigma$. Стоит отметить, что данный метод эквивалентен оценкам, основанным на стандартном отклонении, которые были описаны ранее. При таком преобразовании все точки, лежащие ниже предела $\mu-3\sigma$ теперь отображаются в точки, которые меньше -3 по шкале z-оценки. По аналогии, все точки, лежащие выше верхнего предела $\mu + 3\sigma$ соответствуют значению, которое больше 3. К основным недостаткам данного метода относится то, что он работает только с одномерными данными, которые должны быть нормально распределены.

3) Метод межквартильного расстояния

Межквартильное расстояние (interquartile range) – это разница между первым и третьем квартилями или между 25 (1 квартиль) и 75 (3 квартиль) процентилями. Графически это можно построить с помощью коробчатого графика, напоминающего «ящик с усами» (рис. 2).

Горизонтальная линия внутри ящика соответствует медиане, а верхний и нижний концы 25% и 75% квантилям соответственно. График содержит верхнюю ограду или «ус», который продолжается вверх вплоть до максимального значения, но не выше полуторного межквартильного расстояния от верхней части ящика. По аналогии и противоположная ограда продолжается вниз практически до минимального значения, но не дальше полуторного межквартильного расстояния от нижней кромки ящика. Края «усов» обозначаются горизонтальными линиями, расположенными по обе стороны от «коробки». Выбросами можно считать значения, которые находятся за пределами «усов».

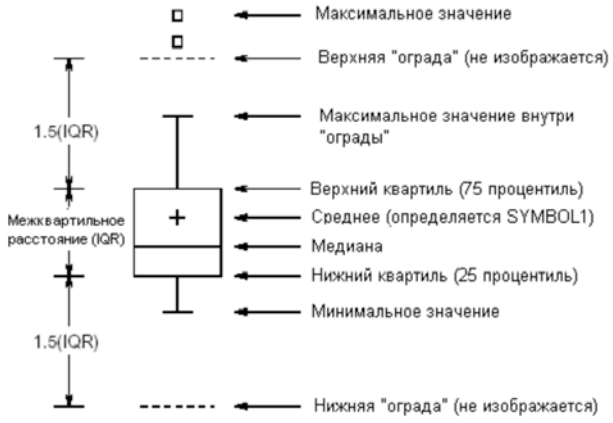

Рис 2. Коробчатый график

Преимуществом этого метода перед предыдущими является то, что его можно использовать в случае, если записи в наборе не соответствуют нормальному распределению. Однако данный подход имеет один большой недостаток, который заключается в том, что что его можно использовать только с двумерными данными. Это означает, что он подходит для случаев, когда имеется только один столбец, представляющий вход, и другой, представляющий выход. Сложность заключается в том, что если будет 10 столбцов, которые представляют входные данные, и один – выходные, то при использовании данного метода нужно будет построить 10 графиков. Таким образом, несмотря на эффективность графических методов, их можно использовать не во всех ситуациях.

4) Метод изолирующего леса

Изолирующий лес (isolation forest) – это алгоритм, который очень быстро позволяет выявлять выбросы в наборе данных [19]. Особенностью данного метода является то, что он способен напрямую обнаруживать аномалии, используя изоляцию (насколько далеко точка данных находится от остальных данных). Из этого следует, что алгоритм изолирующего леса может работать с линейной временной сложностью, как и другие модели, которые связаны с расстоянием. Аналогично методу «случайный лес» данный алгоритм построен на древовидной структуре данных и не является моделью супервизорного управления, так как в нём нет заранее заданных меток. Подход с использованием изолирующего леса вводит такое понятие, как ансамбль бинарных деревьев решений, в котором каждое дерево называется «изолирующим деревом». Это означает, что данный алгоритм работает со случайной выборкой данных, обрабатывающихся в древовидной структуре на основе случайно выбранных признаков. Образцы, которые глубже проникают в дерево, с меньшей вероятностью являются выбросами, так как для их изолированности требуется больше срезов. Из-за того, что алгоритм «изолирующий лес» не является моделью супервизорного управления, то он не оставляет исследователю никакой возможности контролировать данные. Достоинством данного метода является то, что он позволяет обрабатывать и удалять выбросы в многомерных таблицах (dataframe), а также предоставляет возможность работать со всем набором данных, не уменьшая его.

C. *Удаление дубликатов*

Следующим подходом очистки данных является устранение повторяющихся значений. Дубликат – это две или более повторяющиеся записи, которые имеют одинаковые значения для всех признаков или атрибутов [20]. Такие записи могут возникать в результате ошибки, связанной с человеческим фактором, или по причине сбора данных из разных источников, которые содержат одинаковые значения.

Дубликаты записи могут повлиять на результаты модели машинного обучения, поэтому их удаление является важным шагом в предварительной обработке данных. Однако делать это нужно с осторожностью, так как в некоторых случаях это может привести к серьезным ошибкам. Например, удаление дубликатов, которые содержат информацию, имеющую отношение к анализу, может привести к её потере и неточным результатам. Именно поэтому, прежде чем производить эту операцию крайне важно определить природу данных и цель анализа.

Рассмотрим несколько техник, которые можно использовать для удаления повторяющихся записей.

1) *Удаление полных дубликатов*

Данный метод удаляет записи, которые полностью совпадают со строками в других местах. Этот подход является достаточно простым в реализации, но он не учитывает частичные дубликаты и неэффективен по производительности, если набор данных содержит множество неповторяющихся строк.

2) *Удаление частичных дубликатов*

Этот метод удаляет строки, которые имеют одинаковые значения в некоторых, но не во всех столбцах. Он может быть полезен, если важны только определенные атрибуты. Однако данный метод может привести к потере важной информации, если в строках есть отличия в других столбцах.

3) *Использования уникальных идентификаторов*

Идея данного метода заключается в том, чтобы каждой строке в наборе данных присвоить уникальный идентификатор, который затем используется для удаления дубликатов. Этот подход является более точным чем предыдущие, так как он не удаляет строки, которые могут содержать различия в других столбцах. Однако данный метод неэффективен, если в наборе данных присутствует большое количество строк.

4) *Использования хэш-функций*

Данный метод применяет хэш-функцию [21] к каждой строке в наборе данных, а затем использует полученные хэши для удаления дубликатов. Его достоинство заключается в том, что такой подход работает более эффективно, чем предыдущие, так как использует хэш-функции. Однако данный метод также может привести к потере информации, если две строки имеют одинаковый хэш, но отличаются в некоторых столбцах. Именно поэтому такой подход сводит задачу к выбору оптимальной хэш-функции, что зачастую является нетривиальным.

Устранение дубликатов является важным и необходимым этапом в очистке данных, поскольку помогает повысить эффективность, уменьшить шум и улучшить точность, что может повлиять на производительность модели.

IV. Инструменты для очистки данных

Инструменты для очистки данных, будь то фреймворки, библиотеки или специализированные программные решения, предоставляют средства для автоматизации процесса выявления и решения проблем, связанных с некачественными данными. Они позволяют обнаруживать и устранять пропущенные значения, дубликаты, выбросы, ошибки формата, а также проводить трансформации и структурные изменения данных. Благодаря этим инструментам, специалисты по анализу данных и машинному обучению могут оперативно и эффективно подготавливать данные для дальнейшего анализа, что в свою очередь способствует повышению точности и достоверности результатов исследований.

Существует разные инструменты для очистки данных, охватывающие различные языки программирования и предоставляющие разнообразные возможности.

Рассмотрим основные инструменты, которые можно использовать для очистки данных.

1) *Программный комплекс OpenRefine*

OpenRefine [22], ранее известный как Google Refine, это мощный инструмент с открытым исходным кодом, предназначенный для очистки и преобразования данных. Он предоставляет средства для обработки и предварительной обработки больших объемов данных различных форматов, что делает его незаменимым инструментом для подготовки данных перед анализом или интеграцией в другие системы. OpenRefine распространяется в виде настольного приложения, которое открывается в браузере через локальный веб-сервер. Интерфейс данного инструмента, изображенный на рисунке 3, похож на приложения для работы с электронными таблицами, но ведёт себя больше, как база данных.

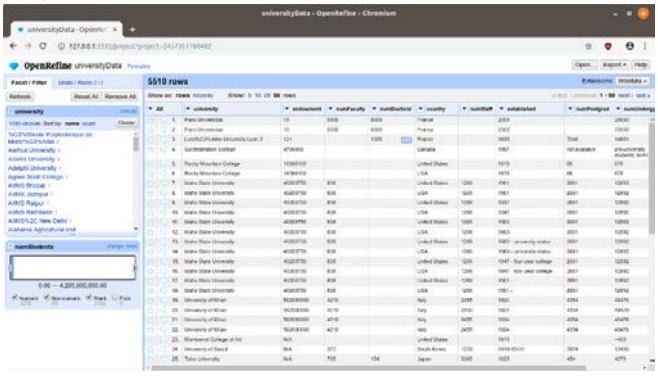

Рис 3. Интерфейс приложения OpenRefine

Он работает со строками данных, ячейки которых находятся под столбцами, аналогично тому, как работают таблицы реляционной базы данных. Проекты OpenRefine состоят из одной таблицы, строки которой можно фильтровать с помощью фасетов, определяющих критерии. Выражения для формул могут быть написаны на таких языках, как General Refine Expression Language (GREL) [23], Jython [24] и Clojure [25].

Основной функционал для очистки данных, которым обладает OpenRefine, заключается в следующем.

a. Устранение дубликатов

OpenRefine может обнаруживать и объединять дубликаты строк данных, что позволяет создать чистый набор данных без повторений.

b. Обработка пропущенных значений

Инструмент предоставляет специальные методы для работы с пропущенными значениями, такие как заполнение пропусков, удаление строк с пропущенными значениями или замена на стандартные значения.

c. Трансформация данных

OpenRefine позволяет создавать и применять различные трансформации к данным, такие как преобразования форматов дат, разделение и объединение строк, а также вычисление новых значений на основе существующих.

d. Обработка текста

OpenRefine поддерживает операции по очистке текстовых данных, такие как удаление лишних пробелов, приведение к нижнему или верхнему регистру, а также удаление невидимых символов.

e. Поддержка различных форматов данных

Данный инструмент умеет работать с разнообразными форматами данных, включая CSV, TSV, Excel и JSON, что делает его удобным для обработки данных из разных источников.

Таким образом, OpenRefine имеет простой и интуитивно понятный пользовательский интерфейс, который позволяет работать с данными как визуально, так и с использованием специальных выражений и функций. Этот инструмент особенно полезен для специалистов по обработке и анализу данных, а также для всех тех, кто сталкивается с необходимостью подготовки разнообразных данных к анализу или интеграции.

2) Язык R

Язык R [26] – это мощный и гибкий язык программирования и среда разработки, специально созданные для статистического анализа, визуализации данных и выполнения задач машинного обучения. R является языком с открытым исходным кодом, который доступен для различных операционных систем. Язык R предоставляет богатый набор инструментов и библиотек для эффективной очистки данных, обеспечивая аналитикам и исследователям точность и надежность исходных данных. К основным методам для очистки данных в R можно отнести следующие.

a. Обработка пропущенных значений

Для обработки пропущенных значений в языке R предусмотрены функции is.na() и complete.cases(). Данный язык также содержит функции na.omit(), которая удаляет строки с пропущенными значениями и na.fill(), которая позволяет заполнить эти строки заданным значением. Стоит отметить, что для заполнения пропущенных значений правдоподобными для языка R существует пакет MICE (Multivariate Imputation via Chained Equations) [27]. Эти правдоподобные значения берутся из распределения, специально разработанного для каждой отсутствующей точки данных.

b. Обработка дубликатов

Для обработки дубликатов в языке R существует две основные функции, duplicated() и unique(). Первая функция позволяет обнаруживать дубликаты строк данных, а вторая удаляет дубликаты из вектора или матрицы.

c. Обработка выбросов

Для обнаружения выбросов в языке R существуют различные методы построения гистограмм, а также функция boxplot(), которая позволяет построить коробчатый график. Помимо этого, в языке R реализована функция IQR для вычисления межквартильного расстояния.

Язык R также предоставляет множество сторонних пакетов, таких как Janitor, tidyr, stringr, и plyr, которые значительно упрощают и ускоряют процесс очистки данных [28].

Таким образом, весь функционал, который доступен в языке R, позволяет аналитикам и исследователям эффективно подготовить данные для дальнейшего анализа, обеспечивая точность и достоверность результатов.

3) Библиотеки для Python

Для языка программирования Python существует множество библиотек, которые предоставляют инструменты для очистки данных. Рассмотрим основные библиотеки для очистки данных.

a. Pandas

Pandas [29] – это одна из наиболее популярных библиотек для работы с данными в Python. Она предоставляет множество функций для очистки данных:

- drop_duplicates() – удаление дубликатов строк данных
- dropna() – удаление строк или столбцов с пропущенными значениями
- fillna() – заполнение пропущенных значений заданными
- replace() – замена значений на заданные
- функции str.strip(), str.lower() позволяют производить манипуляции с текстовыми данными

b. NumPy

Библиотека NumPy [30] широко используется для работы с числовыми данными и массивами. Для определения пропущенных значений в массиве, NumPy предоставляет функцию np.isnan(). Эти значения затем можно удалить с помощью функции np.delete(). Для удаления повторяющихся значений в NumPy предусмотрена функция np.unique(). Данный пакет также содержит функции для нормализации данных, которые позволяют привести значения в массиве, например, к диапазону от 0 до 1.

c. Scikit-learn

Библиотека scikit-learn [31] специализируется на машинном обучении, но также предоставляет следующие инструменты для предварительной обработки данных:

- Для вычисления пропущенных значений можно использовать класс SimpleImputer. Он позволяет заменить пропущенные значения наиболее часто

встречающимся, средним или медианным значением

- Для обнаружения выбросов, можно использовать класс EllipticEnvelope. Этот класс использует эллиптическую огибающую для идентификации точек данных, которые, вероятно, будут выбросами
- Для выбора наилучших признаков из набора данных в пакете scikit-learn реализован класс SelectKBest. Основная идея SelectKBest заключается в том, чтобы оценить статистическую важность каждого признака и выбрать заданное количество наилучших признаков для использования в анализе или моделировании
- Класс StandardScaler позволяет произвести нормализацию данных. Он масштабирует данные до общего диапазона путем вычитания среднего значения и деления на стандартное отклонение

Таким образом, каждая из рассмотренных библиотек обладает своими преимуществами и специализированными функциями. Выбор подходящей библиотеки и методов очистки данных зависит от конкретных требований задачи, типов данных и специфики анализа. Комбинируя эти инструменты, можно обеспечить надежность, точность и готовность данных для дальнейших этапов анализа и моделирования.

V. Заключение

В данной статье представлен краткий обзор основных методов по очистке данных с рассмотрением их достоинств и недостатков, а также даны общие рекомендации, позволяющие выполнить процесс очистки более точно. Помимо этого, рассмотрены основные инструменты для очистки данных, а также описан их функционал. Важность умения грамотно использовать разнообразные инструменты в зависимости от конкретной задачи не может быть недооценена. Эффективное комбинирование инструментов, а также глубокое понимание их функциональности, позволяет создавать чистые, надежные и анализируемые наборы данных. В результате, это содействует более точным и обоснованным выводам, повышая эффективность и качество анализа, моделирования и принятия решений на основе данных.

Настоящая статья будет полезна как начинающим, так и опытным аналитикам при работе с наборами данных из разных источников.

Дальнейшие исследования могут быть посвящены рассмотрению различных методов, основанных на применении машинного обучения для повышения эффективности и точности при очистке данных. Помимо этого, также можно уделить отдельное внимание рассмотрению и сравнению других инструментов для очистки данных.

Исследование выполнено при поддержке Междисциплинарной научно-образовательной школы Московского университета «Мозг, когнитивные системы, искусственный интеллект».

Библиография

[1] Ni, Du, Zhi Xiao, and Ming K. Lim. "Machine learning in recycling business: an investigation of its practicality, benefits and future trends." Soft Computing 25 (2021): 7907-7927.

[2] Kumar, Yogesh, Komalpreet Kaur, and Gurpreet Singh. "Machine learning aspects and its applications towards different research areas." In 2020 International conference on computation, automation and knowledge management (ICCAKM), pp. 150-156. IEEE, 2020.

[3] Baldominos, Alejandro, Yago Saez, and Pedro Isasi. "A survey of handwritten character recognition with mnist and emnist." Applied Sciences 9, no. 15 (2019): 3169.

[4] Obaid, Kavi B., Subhi Zeebaree, and Omar M. Ahmed. "Deep learning models based on image classification: a review." International Journal of Science and Business 4, no. 11 (2020): 75-81.

[5] Yim, Aldrin, Claire Chung, and Allen Yu. Matplotlib for Python Developers: Effective techniques for data visualization with Python. Packt Publishing Ltd, 2018.

[6] Dabbas, Elias. Interactive Dashboards and Data Apps with Plotly and Dash: Harness the power of a fully fledged frontend web framework in Python–no JavaScript required. Packt Publishing Ltd, 2021.

[7] Villanueva, Randle Aaron M., and Zhuo Job Chen. "ggplot2: elegant graphics for data analysis." (2019): 160-167.

[8] Elliott, Alan C., Linda S. Hynan, Joan S. Reisch, and Janet P. Smith. "Preparing data for analysis using Microsoft Excel." Journal of investigative medicine 54, no. 6 (2006): 334-341.

[9] Aini, Qurotul, Untung Rahardja, Indri Handayani, Marviola Hardini, and Ahad Ali. "Utilization of google spreadsheets as activity information media at the official site alphabet incubator." In Proc. Int. Conf. Ind. Eng. Oper. Manag, no. 7, pp. 1330-1341. 2019.

[10] Chicco, Davide, Luca Oneto, and Erica Tavazzi. "Eleven quick tips for data cleaning and feature engineering." PLOS Computational Biology 18, no. 12 (2022): e1010718.

[11] Peng, Chao-Ying Joanne, Michael Harwell, Show-Mann Liou, and Lee H. Ehman. "Advances in missing data methods and implications for educational research." Real data analysis 3178 (2006): 102.

[12] Donders, A. Rogier T., Geert JMG Van Der Heijden, Theo Stijnen, and Karel GM Moons. "A gentle introduction to imputation of missing values." Journal of clinical epidemiology 59, no. 10 (2006): 1087-1091.

[13] Yoon, Jinsung, William R. Zame, and Mihaela van der Schaar. "Estimating missing data in temporal data streams using multi-directional recurrent neural networks." IEEE Transactions on Biomedical Engineering 66, no. 5 (2018): 1477-1490.

[14] Kim, Joo-Chang, and Kyungyong Chung. "Multi-modal stacked denoising autoencoder for handling missing data in healthcare big data." IEEE Access 8 (2020): 104933-104943.

[15] Beretta, Lorenzo, and Alessandro Santaniello. "Nearest neighbor imputation algorithms: a critical evaluation." BMC medical informatics and decision making 16, no. 3 (2016): 197-208.

[16] Tavazzi, Erica, Sebastian Daberdaku, Rosario Vasta, Andrea Calvo, Adriano Chiò, and Barbara Di Camillo. "Exploiting mutual information for the imputation of static and dynamic mixed-type clinical data with an adaptive k-nearest neighbours approach." BMC Medical Informatics and Decision Making 20, no. 5 (2020): 1-23.

[17] Patel, Jagdish K., and Campbell B. Read. Handbook of the normal distribution. Vol. 150. CRC Press, 1996.

[18] Pukelsheim, Friedrich. "The three sigma rule." The American Statistician 48, no. 2 (1994): 88-91.

[19] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." In 2008 eighth ieee international conference on data mining, pp. 413-422. IEEE, 2008.

[20] Rubin, Donald B. Multiple imputation for nonresponse in surveys. Vol. 81. John Wiley & Sons, 2004.

[21] Maetouq, Ali, Salwani Mohd Daud, Noor Azurati Ahmad, Nurazean Maarop, Nilam Nur Amir Sjarif, and Hafiza Abas. "Comparison of hash function algorithms against attacks: A review." International Journal of Advanced Computer Science and Applications 9, no. 8 (2018).

[22] Miller, Meg, and Natalie Vielfaure. "OpenRefine: An Approachable Open Tool to Clean Research Data." Bulletin-Association of Canadian Map Libraries and Archives (ACMLA) 170 (2022).
[23] Ma, Hong. "Google Refine–http://code. google. com/p/google-refine." Technical Services Quarterly 29, no. 3 (2012): 242-243.
[24] Juneau, Josh, Jim Baker, Frank Wierzbicki, Leo Soto Muoz, Victor Ng, Alex Ng, and Donna L. Baker. The definitive guide to Jython: Python for the Java platform. Apress, 2010.
[25] Hickey, Rich. "The Clojure programming language." In Proceedings of the 2008 symposium on Dynamic languages, pp. 1-1. 2008.
[26] R Core Team, R. "R: A language and environment for statistical computing." (2013): 275-286.
[27] Hallam, Antony, Debajoy Mukherjee, and Romain Chassagne. "Multivariate imputation via chained equations for elastic well log imputation and prediction." Applied Computing and Geosciences 14 (2022): 100083.
[28] Boehmke, Bradley C. Data wrangling with R. New York: Springer, 2016.
[29] Bernard, J. and Bernard, J., 2016. Python data analysis with pandas. Python Recipes Handbook: A Problem-Solution Approach, pp.37-48.
[30] Nelli, Fabio. "Python data analytics with Pandas, NumPy, and Matplotlib." (2018).
[31] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al. "Scikit-learn: Machine learning in Python." the Journal of machine Learning research 12 (2011): 2825-2830.
[32] Namiot, Dmitry, Eugene Ilyushin, and Oleg Pilipenko. "On Trusted AI Platforms." International Journal of Open Information Technologies 10.7 (2022): 119-127. (in Russian).

Overview of data cleaning methods for machine learning

Artem Makarov, Dmitry Namiot

***Abstract* — In the last few years, machine learning models and neural networks have been actively introduced into everyday life. The main parameters in their training are accuracy and efficiency. One of the main steps that allows you to improve these indicators is to prepare a data set. Before applying any method, it is necessary to perform a preliminary cleaning of the data, since otherwise the results obtained may be inaccurate or incorrect. Even though novice researchers prepare data sets, cleaning is often performed incorrectly or inefficiently with lots of errors. This article provides an overview of the main methods, considers their advantages and disadvantages, and gives general recommendations to improve the data cleaning process. In addition, special attention is paid to the importance of the ability to use various tools for data cleaning. The main libraries such as Pandas, scikit-learn, and NumPy, specialized programs such as OpenRefine, various features of the R language, as well as methods of normalization, standardization, and processing of text data are considered. The correct use of data cleaning tools significantly affects the quality of analysis and modeling, contributing to more accurate and reliable results.**

***Keywords* — machine learning, neural networks, data cleaning, data cleaning tools**

REFERENCES

[1] Ni, Du, Zhi Xiao, and Ming K. Lim. "Machine learning in recycling business: an investigation of its practicality, benefits and future trends." Soft Computing 25 (2021): 7907-7927.

[2] Kumar, Yogesh, Komalpreet Kaur, and Gurpreet Singh. "Machine learning aspects and its applications towards different research areas." In 2020 International conference on computation, automation and knowledge management (ICCAKM), pp. 150-156. IEEE, 2020.

[3] Baldominos, Alejandro, Yago Saez, and Pedro Isasi. "A survey of handwritten character recognition with mnist and emnist." Applied Sciences 9, no. 15 (2019): 3169.

[4] Obaid, Kavi B., Subhi Zeebaree, and Omar M. Ahmed. "Deep learning models based on image classification: a review." International Journal of Science and Business 4, no. 11 (2020): 75-81.

[5] Yim, Aldrin, Claire Chung, and Allen Yu. Matplotlib for Python Developers: Effective techniques for data visualization with Python. Packt Publishing Ltd, 2018.

[6] Dabbas, Elias. Interactive Dashboards and Data Apps with Plotly and Dash: Harness the power of a fully fledged frontend web framework in Python–no JavaScript required. Packt Publishing Ltd, 2021.

[7] Villanueva, Randle Aaron M., and Zhuo Job Chen. "ggplot2: elegant graphics for data analysis." (2019): 160-167.

[8] Elliott, Alan C., Linda S. Hynan, Joan S. Reisch, and Janet P. Smith. "Preparing data for analysis using Microsoft Excel." Journal of investigative medicine 54, no. 6 (2006): 334-341.

[9] Aini, Qurotul, Untung Rahardja, Indri Handayani, Marviola Hardini, and Ahad Ali. "Utilization of google spreadsheets as activity information media at the official site alphabet incubator." In Proc. Int. Conf. Ind. Eng. Oper. Manag, no. 7, pp. 1330-1341. 2019.

[10] Chicco, Davide, Luca Oneto, and Erica Tavazzi. "Eleven quick tips for data cleaning and feature engineering." PLOS Computational Biology 18, no. 12 (2022): e1010718.

[11] Peng, Chao-Ying Joanne, Michael Harwell, Show-Mann Liou, and Lee H. Ehman. "Advances in missing data methods and implications for educational research." Real data analysis 3178 (2006): 102.

[12] Donders, A. Rogier T., Geert JMG Van Der Heijden, Theo Stijnen, and Karel GM Moons. "A gentle introduction to imputation of missing values." Journal of clinical epidemiology 59, no. 10 (2006): 1087-1091.

[13] Yoon, Jinsung, William R. Zame, and Mihaela van der Schaar. "Estimating missing data in temporal data streams using multi-directional recurrent neural networks." IEEE Transactions on Biomedical Engineering 66, no. 5 (2018): 1477-1490.

[14] Kim, Joo-Chang, and Kyungyong Chung. "Multi-modal stacked denoising autoencoder for handling missing data in healthcare big data." IEEE Access 8 (2020): 104933-104943.

[15] Beretta, Lorenzo, and Alessandro Santaniello. "Nearest neighbor imputation algorithms: a critical evaluation." BMC medical informatics and decision making 16, no. 3 (2016): 197-208.

[16] Tavazzi, Erica, Sebastian Daberdaku, Rosario Vasta, Andrea Calvo, Adriano Chiò, and Barbara Di Camillo. "Exploiting mutual information for the imputation of static and dynamic mixed-type clinical data with an adaptive k-nearest neighbours approach." BMC Medical Informatics and Decision Making 20, no. 5 (2020): 1-23.

[17] Patel, Jagdish K., and Campbell B. Read. Handbook of the normal distribution. Vol. 150. CRC Press, 1996.

[18] Pukelsheim, Friedrich. "The three sigma rule." The American Statistician 48, no. 2 (1994): 88-91.

[19] Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." In 2008 eighth ieee international conference on data mining, pp. 413-422. IEEE, 2008.

[20] Rubin, Donald B. Multiple imputation for nonresponse in surveys. Vol. 81. John Wiley & Sons, 2004.

[21] Maetouq, Ali, Salwani Mohd Daud, Noor Azurati Ahmad, Nurazean Maarop, Nilam Nur Amir Sjarif, and Hafiza Abas. "Comparison of hash function algorithms against attacks: A review." International Journal of Advanced Computer Science and Applications 9, no. 8 (2018).

[22] Miller, Meg, and Natalie Vielfaure. "OpenRefine: An Approachable Open Tool to Clean Research Data." Bulletin-Association of Canadian Map Libraries and Archives (ACMLA) 170 (2022).

[23] Ma, Hong. "Google Refine–http://code. google. com/p/google-refine." Technical Services Quarterly 29, no. 3 (2012): 242-243.

[24] Juneau, Josh, Jim Baker, Frank Wierzbicki, Leo Soto Muoz, Victor Ng, Alex Ng, and Donna L. Baker. The definitive guide to Jython: Python for the Java platform. Apress, 2010.

[25] Hickey, Rich. "The Clojure programming language." In Proceedings of the 2008 symposium on Dynamic languages, pp. 1-1. 2008.

[26] R Core Team, R. "R: A language and environment for statistical computing." (2013): 275-286.

[27] Hallam, Antony, Debajoy Mukherjee, and Romain Chassagne. "Multivariate imputation via chained equations for elastic well log imputation and prediction." Applied Computing and Geosciences 14 (2022): 100083.

[28] Boehmke, Bradley C. Data wrangling with R. New York: Springer, 2016.

[29] Bernard, J. and Bernard, J., 2016. Python data analysis with pandas. Python Recipes Handbook: A Problem-Solution Approach, pp.37-48.

[30] Nelli, Fabio. "Python data analytics with Pandas, NumPy, and Matplotlib." (2018).

[31] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al. "Scikit-learn: Machine learning in Python." the Journal of machine Learning research 12 (2011): 2825-2830.

[32] Namiot, Dmitry, Eugene Ilyushin, and Oleg Pilipenko. "On Trusted AI Platforms." International Journal of Open Information Technologies 10.7 (2022): 119-127.