



Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатики и систем управления»

КАФЕДРА «Программная инженерия»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
НА ТЕМУ:
Метод обнаружения выбросов временных рядов

Студент ИУ7-82
(Группа)

(Подпись, дата)

Капустин А.И.
(И.О.Фамилия)

Руководитель ВКР

(Подпись, дата)

Оленев А.А.
(И.О.Фамилия)

Консультант

(Подпись, дата)

(И.О.Фамилия)

Консультант

(Подпись, дата)

(И.О.Фамилия)

Нормоконтролер

(Подпись, дата)

(И.О.Фамилия)

2018 г.

T3

T3

ПЛАН

Реферат

РПЗ 45 страниц, 10 рисунков, 6 таблиц, 24 источника.

Цель квалификационной работы бакалавра – разработка программного комплекса поиска аномалий. В квалификационной работе бакалавра проведён обзор алгоритмов поиска аномалий, а так же обзор алгоритмов для их улучшения, проанализирована предметная область. Предложен новый метод поиска аномалий. Создан программный продукт, реализующий предложенный метод, а так же вспомогательный программный продукт, которые осуществляет сбор данных для анализа.

Содержание

Введение	9
1 Аналитический раздел	10
1.1 Цель и задачи работы	10
1.2 Что такое аномалия	10
1.3 Обнаружение аномалий	11
1.3.1 Классификация методов обнаружений аномалий	11
1.4 Результат метода обнаружения аномалий	12
1.5 Виды аномалий	12
1.5.1 Нормализация данных	14
1.5.1.1 Основные методы нормализация данных	14
1.6 Неконтролируемые алгоритмы обнаружения аномалий	15
1.6.1 Вероятностный-генеративные методы	15
1.6.2 Линейные методы	16
1.6.3 Метрические методы	16
1.6.3.1 Базовые понятия	17
1.6.3.2 Оптимизация Рамасвани	17
1.6.3.3 К ближайших соседей	18
1.6.3.4 Методы Кнора-Реймонда и Байерса-Рейтери	19
1.6.3.5 Метод Танга	19
1.6.3.6 Метод ОДИН(ODIN)	20
1.6.3.7 Локальный коэффициент выбросов(LOF)	20
1.6.3.8 Компонентный коэффициент выбросов(COF)	21
1.6.3.9 Метод вероятностей локальных выбросов(LoOP)	22
1.6.4 Параметрические методы	23
1.7 Методы улучшения алгоритмов поиска аномалий	24
1.7.1 Семплирование	24
1.7.2 Ансамблирование голосованием	24
1.7.3 Итеративный отбор	25
1.8 Выводы	26
2 Конструкторский раздел	27
2.1 Метод обнаружения аномалий	27
2.2 Подбор параметров алгоритмов	27
2.2.1 Анализ результатов работы методов	27
2.2.2 Подбор параметров	28
2.3 К ближайших соседей	29
2.4 Локальный коэффициент выбросов	30
2.5 Компонентный коэффициент выбросов	31

2.6	Собирающее данные для анализа ПО	31
2.6.1	Собираемые данные	32
2.6.2	Клиентская часть	33
2.6.3	Серверная часть	33
2.6.4	Динамическое изменение данных	33
3	Технологический раздел	35
3.1	Выбор средств разработки	35
3.1.1	Выбор целевой платформы	35
3.1.2	Выбор языка программирования	35
3.1.3	Выбор среды разработки и отладки	36
3.2	Выбор базы данных	36
3.3	Система контроля версий	36
3.4	Библиотека Galgo	37
3.4.1	Формат файлов	37
3.5	Библиотека KUserFeedback	37
3.6	Установка программного обеспечения	38
3.7	Пример работы программы	38
3.8	Использование программы для поиска аномалий	39
4	Исследовательский раздел	40
4.1	Полнота и точность. Площадь по РОК-кривой	40
4.2	Сравнение алгоритмов поиска аномалий	41
	Заключение	43
	Список использованных источников	44

Глоссарий

Выборка/выборка данных — конечный набор прецедентов (объектов, случаев, событий, испытуемых, образцов, и т.п.), некоторым способом выбранных из множества всех возможных прецедентов, называемого генеральной совокупностью[1].

Метка(ярлык) — - порция данных, идентифицирующая набор данных, описывающая его определенные свойства и обычно хранимая в том же пространстве памяти, что и набор данных[2].

Задача классификации — формализованная задача, в которой имеется множество объектов (ситуаций), разделённых некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется выборкой. Классовая принадлежность остальных объектов неизвестна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества.[3]

Классификатор — алгоритм, решающий задачу классификации.

Теория распознавания образа — раздел информатики и смежных дисциплин, развивающий основы и методы классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и т.п. объектов, которые характеризуются конечным набором некоторых свойств и признаков.

Датасет — набор данных.[3]

Схождением — называется такое состояние популяции, когда все строки популяции почти одинаковы и находятся в области некоторого экстремума. [4]

Введение

Задача поиска аномалий является одной из классических задач машинного обучения. В настоящее время задачу поиска аномалий активно решают во многих областях жизнедеятельности:

- а) Защита информации и безопасность
- б) Социальная сфера и медицина
- в) Банковская и финансовая отрасль
- г) Распознавание и обработка текста, изображений, речи
- д) Другие сферы деятельности(например, мониторинг неисправностей механизмов)

Задачей поиска выбросов, как частный случай задачи поиска аномалий так же занимаются во всех вышеперечисленных отраслях.

Количество данных в мире удваивается примерно каждые два года[5]. Поэтому актуальной задачей является разработка новых методов и усовершенствования старых методов поиска выбросов.

В данной работе предлагается новый метод, позволяющий найти аномалии в выборках данных.

1 Аналитический раздел

1.1 Цель и задачи работы

Целью данной работы является создание программного комплекта для обнаружения выбросов временных рядов в собираемых данных. Для достижения данной цели необходимо решить следующие задачи:

- проанализировать предметную область и существующие методы обнаружения выбросов
- разработать метод обнаружения выбросов
- создать ПО, собирающее данные для анализа
- создать ПО, реализующего разработанный метод обнаружения выбросов
- провести исследование с использованием разработанного метода

1.2 Что такое аномалия

В анализе данных есть два основных направления, которые занимаются поиском аномалий - это детектирование новизны и обнаружение выбросов. "Объект новизны" - это так же объект, который отличается по своим свойствам от объектов выборки. Однако, в отличие от выброса, его ещё нет в самой выборке и задача анализа сводится к его обнаружению при появлении. Например, если анализировать замеры уровня шума и отбрасывать слишком высокие или слишком низкие значения, то это называется борьбой с выбросами. А если создаётся алгоритм, который для каждого нового замера оценивает, насколько он похож на прошлые, и выбрасывает аномальные, то это называется "борьбой с новизной" . [6]. Выбросы являются следствием:

- а) ошибок в данных
- б) неверно классифицированных объектов
- в) присутствием объектов других выборок
- г) намеренным искажением данных

На рисунке 1.1 находится три вида точек: зеленые, желтые, красные. Множество зеленых точек представляют собой "нормальные" данные. Множество желтых точек означает выбросы в "слабом смысле". Они незначительно отклоняются от основных нормальных данных. Красные же точки являются аномальными - выбросами "в сильном смысле" . Они значительно отклоняются от нормальных данных. В данной работе будет изучаться вопрос нахождения "сильных выбросов" и критериев отличия сильного выброса от основных данных. В дальнейшем под словом "выброс" будет подразумеваться "сильный выброс" , а под аномалией - выброс(выброс - частный случай аномалии). Понятие аномалии интерпретируют по-разному в зависимости

от характера данных. Обычно аномалией называют некоторое отклонение от нормы. В дальнейшем будет дано несколько более формальных определений аномалий, специфичных для метода их определений.

1.3 Обнаружение аномалий

В машинном обучении обнаружение "ненормальных" экземпляров в наборах данных всегда представляло большой интерес. Вероятно, первое определение было дано Граббсом[7] в 1969 году: "Относительное наблюдение или выброс - это элемент выборки, который, заметно отличается от других членов выборки, в которых он встречается ". Это определение является актуальным и сегодня, но мотивация для обнаружения аномалий изменилась. Тогда основная причина поиска аномалий заключалась в том, чтобы удалить выбросы из набора данных для обучения, поскольку используемые алгоритмы, были весьма чувствительны к выбросам в данных. Эта процедура также называется очищением данных. После разработки классификаторов устойчивых к наличию аномалий в обучающем наборе данных, интерес к их поиску угас. Однако, в начале 21 века в связи с развитием интернета и значительным увеличением объема собираемых данных для анализа, исследователи стали больше интересоваться аномалиями, поскольку они оказывались часто связаны с особенно интересными событиями. В этом контексте определение Граббса также было расширено, так что сегодня аномалии имеют две важные характеристики:

- а) Аномалия отличается от нормы по своим особенностям
- б) Аномалия редко встречается в наборе данных по сравнению с "нормальными" данными

1.3.1 Классификация методов обнаружений аномалий

Классическая система классификации предполагает предварительное обучение на обучающем наборе данных и последующую классификацию на основе этого набора. Данные делятся на "обучающую выборку" - данные, при помощи которых алгоритм обучает классификатор и, "тестовую выборку" - данные, при анализе которых, классификатор остается неизменным. Тестовая выборка нужна для того чтобы проверить корректность обучения классификатора.

Однако, в случае с поиском аномалий, возможны варианты, отличающиеся от классического. Подходящий метод классификации выбирается на основе наличия разметки данных. Выделяются три основных класса методов:

- а) Обучение с учителем. Для обучения необходимо наличие полностью размеченных данных для обучения и для тестов. Классификатор обучается один раз и

применяться впоследствии. В связи с тем, что для многих наборов данных заранее неизвестно что является аномалией, а что нет, применение этого метода ограничено.

б) Обучение с частичным привлечением учителя. Для обучения необходимо наличие тестового и учебного набора данных. Однако, в отличие от обучения с привлечением учителя, разметка данных не требуется. Все данные, представленные в выборках, считаются нормальными. На основе этих данных строится некая модель. Все данные, отклоняющиеся от этой модели, считаются аномальными. Эта идея также известна как "одноклассовая" классификация [8].

в) Обучение без учителя. Самый гибкий способ, который не требует разметки набора данных. Идея заключается в том, что алгоритм обнаружения аномалий оценивает данные исключительно на основе внутренних свойств набора данных что является нормальным, а что является выбросом. В этой работе основное внимание будет этому именно этому способу. Так же этот способ называют "неконтролируемый способ обнаружения аномалий".

1.4 Результат метода обнаружения аномалий

В результате работы алгоритма обнаружения аномалий с элементом данных связывается метка или оценка достоверности (показатель аномальности). Метка-показатель, который принимает нулевое значения, в случае если она связана с нормальными данными и единицу в противном случае. Оценка показывает вероятность того, что элемент является аномалией. Для разных алгоритмов используется разные шкалы оценок, поэтому приведение конкретных примеров оценок будет некорректным. В алгоритмах метода обучения с учителем зачастую используются метки как выходные данные, в алгоритмах с частичным привлечением учителя и без учителя обнаружения аномалий чаще встречаются оценки.

1.5 Виды аномалий

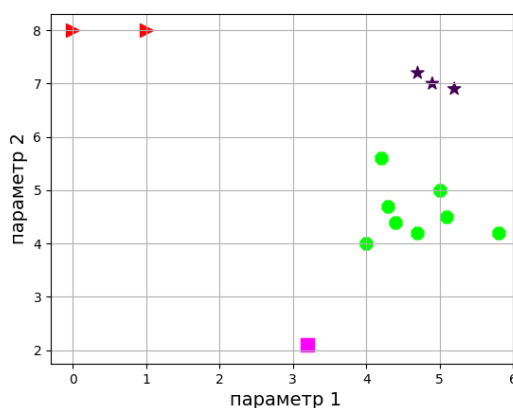


Рисунок 1.1 — Аномалии в двумерном пространстве

Основная идея алгоритмов обнаружения аномалий заключается в обнаружении экземпляров данных в наборе данных, которые отклоняются от нормы. Однако на практике существует множество случаев, когда это основное предположение является неоднозначным. На рис 1.1 показаны некоторые из этих случаев с использованием простого двумерного набора данных. Две аномалии могут быть легко идентифицированы визуально: красные точки (обозначены треугольниками) сильно отличаются значениям параметров от областей плотной группировки точек. Если смотреть на весь набор данных в целом, то фиолетовую точку (обозначена квадратом) можно отнести к тому же классу, что и зеленые точки (обозначены кругами). Однако, если сфокусироваться только на кластере зеленых точек и сравнивать его с фиолетовой точкой, пренебрегая всеми другими точками, то её можно рассматривать как аномалию. Поэтому фиолетовая точка называется локальной аномалией, так как она аномальна по сравнению с ее близкой окрестностью. В зависимости от цели анализа, локальные аномалии могут представлять интерес или нет. Другой вопрос заключается в том, что следует ли рассматривать точки черного кластера (обозначены звездочками) как три аномалии или как (небольшой) кластер. Такие небольшие кластеры явления называются микрокластерами. Показатели аномальности у точек этого кластера выше, чем у точек зеленого кластера, но меньше, чем у красных точек. Этот простой пример показывает, что задача нахождения аномалий аномалии не всегда тривиальна, а вычисление показателя аномальности иногда полезнее, чем проставление двоичной метки.

Задача обнаружения одиночных аномальных экземпляров крупном наборе данных называется обнаружением точечных аномалий [9]. Сегодня почти все неконтролируемые алгоритмы обнаружения относятся к этому типу. Если же аномалии составляют заметный процент, от набора данных, то задачу поиска аномалий называют задачей обнаружения коллективных аномалий. Пусть аномалии представляют собой некоторое множество, тогда необязательно каждый элемент этого множества должен быть аномальным. Возможен вариант когда только определенная их комбинация определяет аномалию. Третий вид - контекстуальные аномалии. Элемент выборке в отрыве от своего контекста может казаться нормальным. Однако, если рассмотреть контекст этого элемента, то очевидным станет его аномальная природа. Распространенным контекстом является время. В качестве примера предположим, что измеряется температура в диапазоне от -30°C до $+40^{\circ}\text{C}$ в течение года. Таким образом, температура 25°C кажется довольно нормальной, но когда учитывается контекстное время (например, месяц), такая высокая температура 25°C в течение зимы будет рассматриваться как аномалия.

Алгоритмы обнаружения точечных аномалий так же можно использовать для обнаружения контекстуальных и коллективных аномалий. Для этого нужно включить контекст в алгоритм как параметр алгоритма. В вышеприведенным при-

мере включение месяца как дополнительного параметра поможет обнаружить аномалию. Однако в более сложных сценариях может потребоваться один или несколько новых параметров, чтобы преобразовать задачу определения контекстной аномалии в задачу обнаружения точечной аномалии. Для того, чтобы преобразовать задачу поиска коллективной аномалии в задачу поиска одиночной, нужно произвести изменения изначального набора данных. Для этого можно использовать корреляцию, агрегация и группировка. Преобразование может быть нетривиальным.[10] . Преобразование требует глубоких знаний о наборе исходных данных и часто приводит к существенным искажениям при переводе данных в новый формат. Такое семантическое преобразование называется генерированием представления данных(*англ. data view generation*).

Таким образом можно сделать вывод, что многие задачи обнаружения аномалий требуют предварительной обработки данных перед передачей их на вход алгоритму. В противном случае можно получить формально верные, но фактически бесполезные результаты.

1.5.1 Нормализация данных

После получения предварительно обработанного датасета для поиска точечной аномалии, то последним шагом перед передачей в алгоритм, является нормализация данных. Нормализация данных предназначена для устранения зависимости от выбора единицы измерения и заключается в преобразовании диапазонов значений всех атрибутов к стандартным интервалам([0,1] или [-1,1])[11]. Нормализация данных направлена на придание всем атрибутам одинакового "веса".

1.5.1.1 Основные методы нормализация данных

а) Min-max нормализация заключается в применении к диапазону значений атрибута x линейного преобразования, которое отображает $[\min(x), \max(x)]$ в $[A, B]$.

$$x'_i = \tau(x_i) = \frac{x_i - \min(x)}{\max(x) - \min(x)} * (B - A) + A \quad (1.1)$$

$$x \in [\min(x), \max(x)] \Rightarrow \tau() \Rightarrow [A, B] \quad (1.2)$$

Min-max нормализация сохраняет все зависимости и порядок оригинальных значений атрибута. Недостатком этого метода является то, что выбросы могут сжать основную массу значений к очень маленькому интервалу

б) Z-нормализация основывается на приведении распределения исходного атрибута x к центрированному распределению со стандартным отклонением, равным 1 [11]

$$x'_i = \tau(x_i) = \frac{x_i - \bar{x}}{\sigma_x} \quad (1.3)$$

$$M[x'] = 1 \quad (1.4)$$

$$D[\bar{x}] = 0 \quad (1.5)$$

Метод полезен когда в данных содержатся выбросы.

в) Масштабирование заключается в изменении длины вектора значений атрибута путем умножения на константу [11] .

$$x'_i = \tau(x_i) = \lambda * x_i \quad (1.6)$$

Длина вектора x уменьшается при $|\lambda| < 1$ и увеличивается при $|\lambda| > 1$

1.6 Неконтролируемые алгоритмы обнаружения аномалий

Существуют различные классы методов обнаружения неконтролируемых аномалий. Выбор соответствующего класса метода зависит от характера данных и наличия априорной информации, требований к скорости работы алгоритма, затрачиваемой памяти, размеру данных и многих других параметров. Рассмотрим основные классы методов, их преимущества и недостатки. Так же можно заметить, что в рамках методы одного класса могут значительно отличаться по своим характеристикам, что дополнительно усложняет выбор.

1.6.1 Вероятностный-генеративные методы

Основная идея генеративных методов заключается в использование вероятностного смесового моделирования данных. Предлагается подобрать такую вероятностную модель, из которой было получены нормированные данные. Такие модели обычно называются генеративными моделями, где для каждой точки(элемента данных) можем посчитать генеративную вероятность(или вероятность правдоподобия). Т.е. задача сводится к нахождению плотности распределения $p(x)$. Аномалиями при этом считаются точки(элементы набора данных), имеющую низкое правдоподобие. В качестве показателя аномальности выступает функция p . Для построения генеративной модели нужно решить следующую задачу:

$$\prod_{x \in X_{norm}} p(x, \theta) \rightarrow \max_{\theta} \quad (1.7)$$

где X_{norm} - нормальные данные представленного набора данных $p(x, \theta) | \theta \in \omega$ - семейство плотностей вероятностей, параметризованные θ ;

Этот метод редко используется на практике, так как тяжело проверить полученную генеративную модель на адекватность, сложно убедиться в правильном выборе семейства смесевых распределений. Это связано с тем, что низкое значение функции правдоподобия может означать как и аномальное значение, так и неудачно подобранную модель. Этот метод применяется с опорой на априорную информацию, в случае когда можно проверить полученную модель на адекватность.

1.6.2 Линейные методы

Основной идеей линейных методов является построение некой модели, характеризующей нормальные данные. Точки, которые значительно отклоняются от этой модели, считаются аномалиями.

Предполагается, что нормальные данные находятся в подпространстве пространства атрибутов данных (размер подпространства атрибутов данных равен размерности данных). В свою очередь, задача линейного метода - найти низкоразмерное подпространство, такие что, выборка данных этого подпространства значительно отличается от остальных точек пространства данных.

Одним из возможных вариантов решения является использование линейной регрессии. Выбирается одна из наблюдаемых переменных набора данных и относительно неё решается задача линейной регрессии оставшихся атрибутов. Итоговым ответом будет являться усредненное значения показателя аномалии по всем атрибутам.

Алгоритмы, основанные на линейном подходе, требуют наличия линейной зависимости атрибутов данных.

1.6.3 Метрические методы

Метрические методы пытаются найти в данных точки, в некотором смысле изолированные от остальных [6]. Если в пространстве задана некоторая метрика $p(x1, x2)$, то необходимо задать следующие понятия:

- Аномалии – точки, не попадающие ни в один кластер. К данным применяется один из алгоритмов кластеризации; размер кластера, в котором оказалась точка, объявляется её показателем аномальности.
- Локальная плотность в аномальных точках низкая. Для данной точки показателем аномальности объявляется локальная плотность, которая оценивается некоторым непараметрическим способом.
- Расстояние от данной точки до ближайших соседей велико.

В качестве показателя аномальности может выступать:

- расстояние до k -го ближайшего соседа;

- среднее расстояние до k ближайших соседей;
- медиана расстояний до k ближайших соседей;
- гармоническое среднее до k ближайших соседей;
- доля из k ближайших соседей, для которых данная точка является не более чем k -ым соседом и много другое.

1.6.3.1 Базовые понятия

Метрические методы используют в случае когда данные не размечены. Сложность вычисления прямо пропорциональна как размерности данных m , так и их количеству n . При росте набора данных наблюдается экспоненциальный рост сложности вычислений. Однако, эти методы хорошо проявляют себя на ограниченных наборах данных [12]. Следовательно такие методы как k -ближайших соседей (так же известный как обучение на основе примеров, и описанный позднее) с нотацией асимптотического роста $O(n^2)$ недопустимы для наборов данных с большой размерностью, если их размерность не может быть уменьшена.

Существуют много различных вариации алгоритма k -ближайших соседей для обнаружения аномалий, но все они основаны на вычислении некой метрики "расстояния до соседей" такой как Евклидово расстояние или расстояние Махаланобиса. Евклидово расстояние задается следующей формулой:

$$\sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (1.8)$$

и является просто расстоянием между двумя точками, когда как расстояние Махаланобиса, задаваемое следующей формулой

$$\sqrt{(x - \mu)^T C^{-1} (x - \mu)} \quad (1.9)$$

вычисляет расстояние от точки до центра тяжести (μ), определяемого формулой коррелированных атрибутов, заданных матрицей ковариации (C). Расстояние Махаланобиса рассчитывается значительно дольше по сравнению с евклидовым по сравнению с евклидовым расстоянием для больших объемов данных, поскольку оно требует пройти через весь набор данных, чтобы идентифицировать корреляции атрибутов.

1.6.3.2 Оптимизация Рамасвани

Точка p является выбросом, если не более $n - 1$ других точек в наборе данных имеют более высокий D_m (расстояние до m соседей), где m задается. Например на рисунке 1.2 черная точка (обозначена звездочкой) является наиболее удаленной от соседей, следовательно она является выбросом. Красные точки (обозначены треугольниками) расположены рядом друг с другом, однако расстояние до других точек

велико, следовательно они тоже являются аномалиями. Такой подход восприимчив к вычислительному росту, потому что должна быть вычислена матрица расстояний точек друг от друга, поэтому Рамасвани в 2000 году предложил оптимизацию метода k -ближайших соседей (с англ. k -Nearest Neighbour - k -NN) в виде составления ранжированного списка потенциальных выбросов.

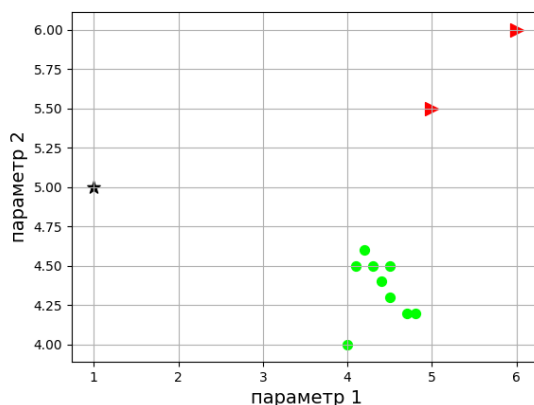


Рисунок 1.2 — Пример поиска аномалий для метода k -ближайших соседей в двумерном пространстве

Оптимизация Рамасвани заключается в разбиении данных на ячейки. Если какая-либо ячейка и ее ближайшие соседи содержат больше, чем k точек, то точки в ячейке считаются лежащими в плотной области поэтому содержащиеся там точки вряд ли будут выбросами. Если же почти все ячейки содержат больше, чем k точек, а какие-то ячейки содержат меньше, чем k точек, то тогда все точки, лежащие в ячейках, которые содержат менее k элементов, помечаются аномальным. Следовательно, необходимо обработать только небольшое количество ячеек, которые ранее не были помечены и только относительно небольшое количество расстояний необходимо вычислить для обнаружений аномалий.

1.6.3.3 K ближайших соседей

Алгоритм работы метода:

- Выбирается число K -число соседей, относительно
- Устанавливается граница показателя аномальности, на основе которой будет определяться метка элемента (задается в процентах относительно среднего показателя расстояния)
- На основе метрики, рассчитывающей расстояния между элементами, высчитывается расстояние между всеми элементами и всеми его соседями.
- Полученный результат сортируется на

— На основе полученных расстояний и границы показателя аномальности элементам присваиваются метки.

В качестве метки, рассчитывающей расстояния между элементами можно использовать следующую формула:

$$L = \sum_0^k x_j 0 - x_j i \quad (1.10)$$

где $x_j i$ - значение j-того атрибута элемента до которого ищется расстояние, а x_0 -искомый элемент.

1.6.3.4 Методы Кнора-Реймонда и Байерса-Рейтери

Кнор и Реймонд предложили свой эффективный метод КНН подхода обучения без учителя[13]. Если m из k ближайших соседей (где $m < k$) лежат в пределах определенного порогового значения d , тогда считается, что данные точки лежат в достаточно плотной области распределения данных, подлежащей классификации и подлежат классификации как нормальные, в противном случае они помечаются как аномальные.

Очень похожий метод был придуман для идентификации наземных мин на спутниковых снимках поверхности Земли Байерсом с соавторстве с Рейтери[14](этот метод можно использовать и для других целей) Он заключается в том, что берется m точек, для них ищется расстояние D_m . Если расстояние меньшего некого порогового значения d , тогда считается, что данные точки лежат в достаточно плотной области распределения данных, подлежащей классификации и подлежат классификации как нормальные, в противном случае они помечаются как аномальные. Этот метод уменьшается количество варьируемых параметров, по сравнению с методом Кнора-Реймонда: остаются только параметры d и m , параметр k убирается. подход оригинальный подход k -NN, поскольку только k ближайших соседей должны быть вычислены для каждой точки, а не всей матрицы расстояния для всех точек

1.6.3.5 Метод Танга

Метод Танга заключается в вычислении средней цепочки расстояний между точкой p и k её соседями. Ранним расстояниям присваиваются более высокие веса, поэтому, если точка находится в разреженной области как черная точка на рисунке 1.2, то путь до ее ближайших соседей будет относительно далеким, а среднее расстояние цепочки будет высоким. Этот метод выгодно отличается от вышеописанных тем, что учитывает как плотность, так и изоляцию. Рассмотрим рисунок 1.3. Очевидно, что черные точки(обозначены звездочками) являются аномалиями, а скопление зеленых точек - множеством "нормальных" точек. Однако, алгоритмы k -NN классификации могут столкнуться с проблемой того, что расстояние от черных точек до зеленого

кластера примерно равно, значит эти точки можно отнести к одной группе и при определенных значениях параметров алгоритма эти точки не будут считаться аномалиями. Метод Танга поможет избежать таких ошибок при обнаружении выбросов. Однако метод является вычислительно сложным с временем выполнения как у ори-

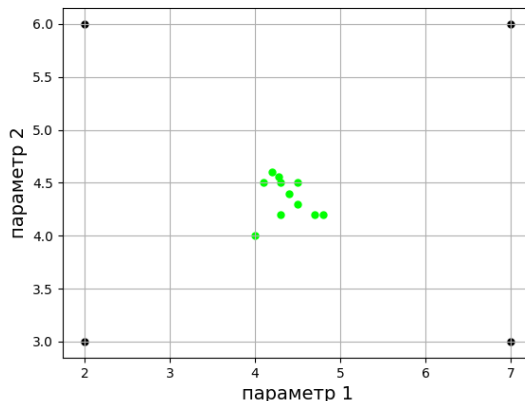


Рисунок 1.3 — Пример поиска аномалий методом Танга в двумерном пространстве

нального k-NN, поскольку он полагается на вычисление путей между всеми точками и их k соседей.

1.6.3.6 Метод ОДИН(ODIN)

Существуют модификации метода KNN, основанные на использовании графов. Рассмотрим один из них. Предположим, что мы нашли у каждого элемента K ближайших соседей(используя евклидово расстояние). Тогда можно сделать взвешенный ориентированный граф, где элемент представляет собой вершину, соединенную с K своими соседями ребрами[15]. Весом ребра будет евклидово расстояние между двумя точками. При построении такого графа можно столкнуться с проблемой известной как "проблема всех k-соседей". Проблема состоит в том, что построение такого графа может сопровождаться большой вычислительной сложностью. С решением этой проблемы можно ознакомиться в [16].

После построения графа, начинается поиск аномалий. Аномалиями считаются все такие вершины, которые имеют 0 входящих ребер, а так же "замкнутые" группы элементов(такая группа элементов, которая имеет ребра только внутри себя, но не имеет связей с окружающими элементами).

1.6.3.7 Локальный коэффициент выбросов(LOF)

Этот метод является одним из самых известных алгоритмов обнаружения локальных аномалий. Недостатком метрических методов является тот факт, что все лежащие в их основе предположения верны лишь в дополнении друг с другом: локальная плотность точки, лежащей в центре небольшого кластера аномалий, может

оказаться выше, чем для любой точки из большого кластера нормальных данных. Возможно и обратное: изолированная точка-аномалия может располагаться, например, в центре масс кластера нормальных точек, и тогда среднее расстояние от неё до соседей будет меньше, чем для нормальных точек. Это "свойство" метрических алгоритмов пытается учесть алгоритм локальных коэффициентов выбросов (англ. Local Outlier Factor).

Чтобы вычислить LOF необходимо произвести следующие действия:

а) Для каждой записи найти всех соседей, расстояния до которых не превышает k . Их количество может быть больше, чем k .

б) Используя эти записи для каждой точки N_k , вычислить локальную плотность точки, основанную на локальной плотности достижимости (англ. local reachability density (LRD)):

$$LRD_k(x) = \frac{1}{\left(\frac{\sum_{o \in N_k(x)} d_k(x,o)}{|N_k(x)|} \right)} \quad (1.11)$$

где $d(x,o)$ расстояние достигаемости. За редким исключением в качестве расстояния достигаемости используется евклидово расстояние [17]

в) Вычисляем LOF путем сравнения LRD записи с LRD соседей.

$$LOF(x) = \frac{\sum_{o \in N_k(x)} \frac{LRD_k(o)}{LRD_k(x)}}{|N_k(x)|} \quad (1.12)$$

Таким образом LOF является отношением локальных плотностей. Нормальные записи, плотности которых равны плотности их соседей, получают оценку около 1,0. Аномалии, которые имеют низкую локальную плотность, получают значительно более высокую оценку. Алгоритм полагаясь только на свою прямую окрестность, формирует оценку - величину, основанную только на k -соседях. Конечно, глобальные аномалии также могут быть обнаружены, так как они имеют низкую LRD, по сравнению со своими соседями. Важно отметить, что в задачах обнаружения аномалий, где местные аномалии не представляют интереса, этот алгоритм будет генерировать множество ложных аномалий. Настройка k имеет решающее значение для этого алгоритма.

Авторы алгоритма LOF рекомендуют использовать для вычисления k стратегию ансамблирования (алгоритм описан ниже). Берется интервал возможных значений k и с некоторым шагом для всех возможных значений k вычисляются показатели аномальности для каждого элемента выборки. Путем голосования определяется является ли эта точка аномалией. Однако, на практике такие рекомендации редко используют из-за их значительной вычислительной сложности.

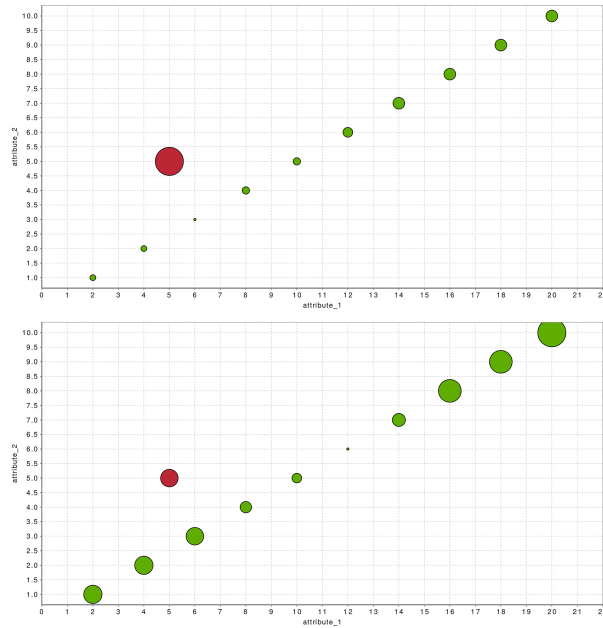


Рисунок 1.4 — Сравнение COF (сверху) с LOF (внизу) с использованием простого набора данных с линейной корреляцией двух атрибутов

1.6.3.8 Компонентный коэффициент выбросов(COF)

Компонентный коэффициент выбросов аналогичен LOF, но оценка плотности для записей выполняется иным способом. В LOF k -ближайших соседей выбирают на основе евклидова расстояния. Это косвенно предполагает, что данные распределяются сферическим образом вокруг экземпляра. Если это допущение нарушено, например, если функции имеют прямую линейную корреляцию, то оценка плотности неверна. COF исправляет этот недостаток и оценивает локальную плотность окрестности с использованием метода кратчайшего пути, называемого расстоянием цепочки. Математически это расстояние цепочки является минимумом суммы всех расстояний, соединяющих все k ближайших соседей точки и саму точку. Например, когда функции, очевидно, коррелированы, этот подход оценки плотности работает значительно лучше [18]. На рисунке 5 показан результат для LOF и COF в сравнении для простого двумерного набора данных, где атрибуты имеют линейную зависимость. Можно видеть, что оценка плотности LOF не может обнаружить выброс, но COF удадается связать нормальные между собой для оценки локальной плотности.

1.6.3.9 Метод вероятностей локальных выбросов(LoOP)

Автора метода вероятностей локальных выбросов утверждает, что метод LOF может давать нестабильные результаты[19]. Поэтому была предложена модификация метода LOF.

Предположим, что D - множество из n объектов, d - функция расстояния, используемая для выделения выбросов. В целях улучшения стабильности результатов

введенном новую метрику - вероятностное расстояние $o \in D$ до множества $S \in D$, обозначаемое $pdist(o,S)$. Вероятностное расстояние обладает следующим свойством:

$$\forall s \in S : P[(d(o,s) < pdist(o,S))] \geq \varphi \quad (1.13)$$

Если визуализировать это выражение, то можно представить, что сфера вокруг o с радиусом $pdist(o,S)$ покрывает любой элемент в множестве S с вероятностью φ . Вероятностное расстояние $pdist(o,S)$ от o до S можно интерпретировать как статистическое распределение множества S . Вероятностное расстояние рассчитывается по следующей формуле:

$$PLOF_{\lambda,S}(o) = \frac{pdist(\lambda,o, S(o))}{E_{s \in S(0)[pdist(\lambda,o, S(s))]} \quad (1.14)$$

Где E - центроида. Итоговая формула для расчёта LoOF выглядит следующим образом:

$$nPLOF = \lambda \sqrt{E[PLOF^2]} \quad (1.15)$$

$$LoOP_s(o) = \max\{0, \text{erf}(\frac{PLOF_{\lambda,S}(o)}{nPLOF * \sqrt{(2)}})\} \quad (1.16)$$

Где erf - Гауссова функция ошибок. С подробным выводом этих формул можно ознакомиться в [19].

1.6.4 Параметрические методы

Вышеописанные методы не подходят для работы с большим объемом данных из-за их высокой временной сложности. Параметрические методы позволяют очень быстро пересчитывать модель для новых данных и подходит для больших наборов данных; модель растёт только с сложностью модели, а не размером данных. Однако они ограничивают применимость, применяя предварительно выбранную модель распределения для проверки данных на аномальность. Т.е. предварительно априорно подбирается модель правдоподобности данных. Элементы, которые значительно отклоняются от этой модели считаются аномальными. Параметрический подход схож с линейным по описанию, но значительно отличается от него по принципу работы.

Одним из таких подходов является оценка эллипсоидой минимального объема[20], которая соответствует наименьшему допустимому эллипсоиду, покрывающему не меньше 50% точек выборки.

1.7 Методы улучшения алгоритмов поиска аномалий

Алгоритмы поиска аномалий можно улучшать различными методами, применяемыми в том числе для улучшения результатов работы алгоритмов и в других областях. Например, ансамблирование широко применяется для работы с нейронными сетями. Ниже рассмотрены приемы в контексте поиска аномалий.

1.7.1 Семпирование

Большинство алгоритмов распознавания аномалий успешно работают на выборках малых размеров. Поэтому предлагается разбить начальный набор данных на несколько случайных выборок и усреднить результат. Размер этих выборок может быть как и случайным, там и фиксированного размера, но, как правило, он отличается от размеров исходного набора данных не меньше чем на порядок. Идея такого выбора заключается в том, что шумовые объекты попадут в выборки с низкой вероятностью; кластера нормальных данных будут представлены несколькими представителями, а кластера аномалий вырождаются в изолированные точки. На основе этих выборок алгоритмы строят функции показателя аномальности, незначительно уступающему результату, полученному на основе анализа всех исходных данных.

Этот метод помогает значительно сократить вычислительную сложность, а так же уменьшить вероятность "подгона" алгоритма под конкретный набор данных. В силу особенностей задачи, необходимое условие отсутствия параметризации алгоритмов зачастую означает их детерминированность (в отсутствие стохастичности показатель аномальности однозначно определяется по заданной выборке). В общем случае при добавлении новых данных в общий набор данных, можно не пересчитывать заново показатель аномальности для всего набора данных, а добавить запуски алгоритма на новых данных в ансамбль (так называемый warm start[21])

1.7.2 Ансамблирование голосованием

Ансамблированием в задаче поиска аномалий называют использование нескольких различных алгоритмов с последующим усреднением их показателя аномальности. При использовании различных классов алгоритмов можно столкнуться с проблемой того, что показатель аномальности выглядит по-разному в различных алгоритмах и сравнить напрямую эти показатели некорректно. Поэтому традиционное

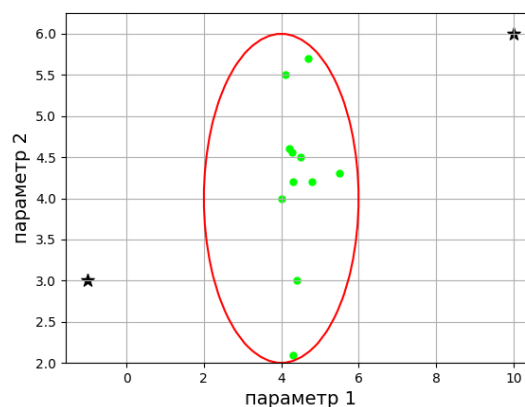


Рисунок 1.5 — Двухмерная проекция эллипсоиды минимального объема

приведение показателей значений различных функций к одному диапазону, например, к $[0,1]$, будет некорректным. Существует несколько наиболее известных видов ансамблирования:

— Простое голосование

$$b(x) = F(b_1(x), \dots, b_T(x)) = \frac{1}{T} \sum_{t=1}^T b_t(x) \quad (1.17)$$

— Взвешенное голосование

$$b(x) = F(b_1(x), \dots, b_T(x)) = \frac{1}{T} \sum_{t=1}^T w_t b_t(x) \quad (1.18)$$

$$\sum_{t=1}^T w_t = 1, w_t \geq 0 \quad (1.19)$$

— Смесь экспертов

$$b(x) = F(b_1(x), \dots, b_T(x)) = \frac{1}{T} \sum_{t=1}^T w_t(x) b_t(x) \quad (1.20)$$

$$\sum_{t=1}^T w_t = 1, \forall x \in X \quad (1.21)$$

Простое голосование - это частный случай взвешенного голосования, а взвешенное голосование является частным случаем смеси экспертов.

Различные методы ансамблирования такие как беггинг, бустинг, стекинг и другие применяются для улучшения работы алгоритмов обучения с учителем. Для алгоритмов обучения без учителя применяется простое голосование, т.к. задача изменения весов голосования нетривиальна в задаче обучения без учителя[22].

1.7.3 Итеративный отбор

Итеративный отбор основан на идее многократного применения алгоритмов ансамблирования. Предположим, построена некоторая модель, описывающая нормальные данные. Эта модель построена на основе всех имеющихся данных, но точность этой модели невелика, она умеет определить только явные аномалии. Отсортировав все точки по показателю аномальности, можно выбрать k самых аномальных объектов в данных и исключить их из данных. После этого можно перестроить модель и повторить вышеуказанные действия несколько раз, пока не будут достигнуты некоторые условия. При каждой итерации точность модели будет увеличиваться.

Идея итеративного отбора может быть обобщена различными способами. Результат работы одного алгоритма может быть использован для отсеивания явных аномалий и настройки нового алгоритма, не обязательно совпадающего с предыдущим, на оставшихся данных. х. Возможна и противоположная механика: по резуль-

татам работы одного алгоритма отбираются явные, гарантированные представители нормальных данных, и исключительно на них строится модель, их описывающая

1.8 Выводы

Таблица 1.1 — Сравнение алгоритмов поиска аномалий

Класс методов	Временная сложность	Расход памяти	Универсальность
Вероятностно-ген.	$O(1)$	$O(n)$	Очень низкая
Линейный	$\geq O(n^2)$	$\geq O(n^2)$	низкая
Параметрический	$O(1)$	$O(n)$	низкая
Метрический	$\geq O(n \log n)$	$\geq O(n)$	Высокая

Под универсальностью понимается возможность применять алгоритмы к различным набором данных, не обладающих специфическими характеристиками, и, не обладая априорной информацией, получать высокую точность классификации.

Существует большое число алгоритмов для нахождения аномалий. Некоторые из них опираются на априорные данные, некоторые не опираются. Для выбора подходящего алгоритма нахождения аномалий зачастую стоит учитывать характер данных, их размер и доступную априорную информацию. Несмотря на то, область знаний обнаружения аномалий активно развивается как часть современной науки, остается ещё много простора для исследования алгоритмов, модификации и создания новых.

2 Конструкторский раздел

В этом разделе приводится подробное описание разрабатываемого метода, выделяются основные его компоненты, описываются метрики, оценивающий метод. Так же приводится описание ПО, собирающее данные для анализа. В выводе аналитической части предлагается разработать новый метод обнаружения аномалий. Новый метод будет является результатом ансамблирования простым голосованием трех метрических методов.

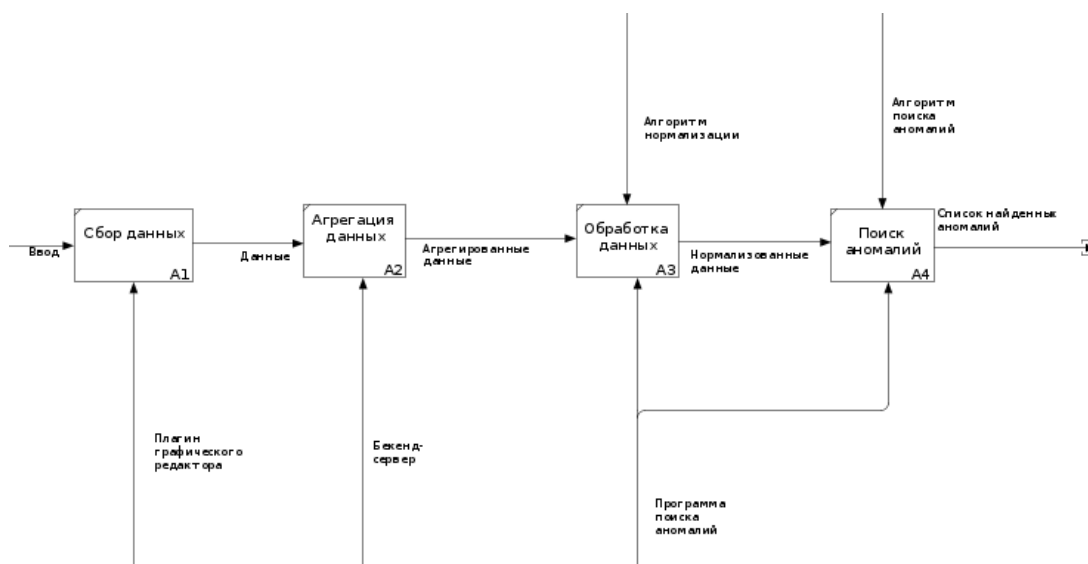


Рисунок 2.1 — Общая схема программного комплекса

2.1 Метод обнаружения аномалий

На вход методу подается файл с нормализованными значениями атрибутов, с фиксированным заранее известным количеством атрибутов для каждого элемента. Временная отметка элемента представляется в качестве отдельного нормализованного атрибута. Допустимо отсутствие временной отметки. Были выбраны методы: к-ближайших соседей, локальный коэффициент выбросов. Каждый из вышеописанных методов инвариантен и иммутабелен относительно набора данных. В результате их работы получается три набора меток. На основе этих наборов формируется финальный набор меток по следующему принципу: если элемент имеет две или более "аномальных" метки, то ему присваивается "аномальная" метка, иначе - "нормальная" метка.

2.2 Подбор параметров алгоритмов

2.2.1 Анализ результатов работы методов

Для проверки работоспособности метода его нужно оценить при помощи наборов данных. Метрикой сравнения наборов данных был выбран AUC ROC — площадь

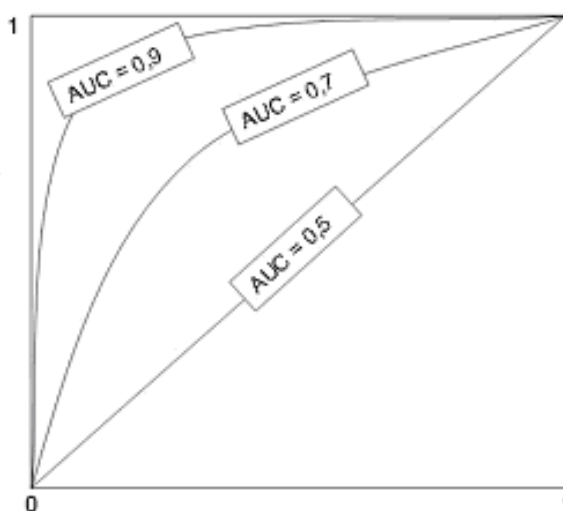


Рисунок 2.2 — AUC ROC

под графиком, позволяющим оценить качество бинарной классификации, отображающим соотношение между долей объектов от общего количества носителей признака, верно классифицированных как несущих признак, и долей объектов от общего количества объектов, не несущих признака, ошибочно классифицированных как несущих признак.

При помощи этой метрики планируется оценить адекватность работы алгоритма на размеченных наборах данных и сравнить с другими методами.

2.2.2 Подбор параметров

Алгоритмы поиска аномалий содержат некоторое различное количество параметров, которые необходимо указывать при работе алгоритмов. Однако, в отсутствии априорной информации о данных, их необходимо подобрать, основываясь на валидирующем наборе данных. [23] Для этого будет использован простой генетический алгоритм. Архитектура генетических алгоритмов позволяет найти решение быстрее за счет более 'осмысленного' перебора. Будет осуществляться перебор не всего подряд, а приближаться от случайно выбранных решений к лучшим. В качестве функции, для которой ищется экстремум, используется алгоритм поиска аномалий, в качестве выходного значения функции - значение AUC ROC, полученное в результате работы алгоритма поиска аномалий на размеченном наборе данных. Критерием останова поиска оптимальных значений параметров алгоритма служит схождение популяции. Параметры, соответствующие наибольшему значению AUC ROC, сохраняются для использования в дальнейшем.

2.3 К ближайших соседей

Метод основан на поиске аномальных значений расстояний до К ближайших соседей точки. Параметры алгоритма такие как К задаются после их нахождения вышеописанным методом.

Листинг 2.1 — Алгоритм метода К ближайших соседей

```
1  double calculateDist(vector<double> values , int orElen , int rInd , int
    lInt , int counter)
2  {
3      while (counter) {
4          int leftDistance = -1, rightDistance = -1;
5          if (rightIndex > -1)
6              rightDistance = abs(values[rightIndex]);
7          if (leftIndex > -1)
8              leftDistance = abs(values[leftIndex]);
9          if (leftDistance > rightDistance) {
10             rightIndex++;
11             distance += abs(originalElement - rightDistance);
12         } else {
13             leftIndex++;
14             distance += abs(originalElement - leftDistance);
15         }
16         counter--;
17     }
18 }
19 double kNearestDistance(vector<double> values , double value , int k)
20 {
21     int index = findIndex(values , value);
22     double originalElement = values[index];
23     double distance = 0;
24     int rightIndex = index + 1, leftIndex = index - 1;
25     int counter = k;
26 }
```

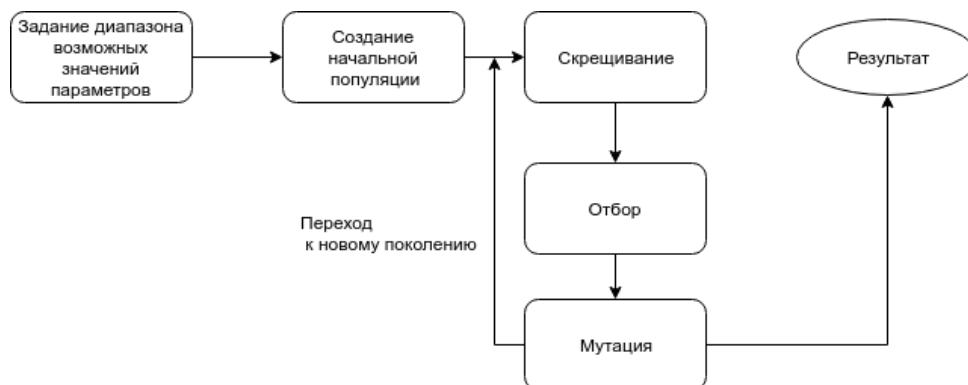


Рисунок 2.3 — Общий рисунок функционирования генетического алгоритма

```

27     return distance;
28 }
29
30 for (int i = 0; i < values.size(); i++) {
31     for (int j = 0; j < values[i].size(); j++) {
32         values[i][j].distance = kNearestDistance(sortedValues[j],
33             values[i][j].value, dimensionSize, j);
34     }
35 }

```

Псевдокод содержит часть алгоритма, которая позволяет определять дистанции между точками. Этот алгоритм отличается от классического алгоритма К ближайших соседей, тем что расстояние до К ближайших соседей находится не с помощью Евклидова расстояния, а при помощи разбиения исходного двумерного массива атрибутов на одномерные массивы атрибутов согласно классам атрибутов.

Алгоритм нахождения расстояние для одного элемента:

- а) Для каждого атрибута находится k ближайших точек этого же класса
- б) Расстояние между искомым атрибутом и k ближайшими соседями суммируется
- в) Рассчитанные для каждого атрибута сумма складываются в итоговое расстояние

Временная сложность данного алгоритма $O(n^2)$, где n- суммарное количество атрибутов всех элементов данных. Сложность по памяти $O(n)$.

2.4 Локальный коэффициент выбросов

Листинг 2.2 — Алгоритм метода локального коэффициент выбросов

```

1  auto calcLRD(vector<Point> values, Point value, int k, Mode mode)
2  {
3      values.delete(value);
4      vector<Point> nearPoints;
5
6      for(i = 0; i < values.size(); ++i){
7          if(distance(values[i], value) > k)
8              nearPoints.pushback(value)
9      }
10     for(i = 0; i < nearPoints.size(); ++i){
11         sum += nearPoints[i];
12     }
13     if(mode != sum)
14         return nearPoints;
15     else
16         return nearPoints.size()/sum;

```

```

17 }
18
19 double LOF(vector<Points> values , Point value ,int k)
20 {
21     vector<Point> nearPoints = cacLRD(values , value , k, val);
22     int N = nearPoints.size()
23     double lrdsSum = 0;
24     for(int i=0; i < nearPoints.size(); ++i){
25         lrds += calcLRD(values , nearPoints[i],k) , sum);
26     }
27
28     double result = lrdsSum/calcLrds(values , value ,k, val);
29     return result;
30 }
31 vector<double> distances;
32 for (int i = 0; i < values.size(); i++) {
33     distances.pushback(LOF(values , value ,k)
34 }

```

Псевдокод содержит основные сущности алгоритма : нахождение LOF и LRD. Временная сложность данного алгоритма $O(n^2)$, где n- суммарное количество атрибутов всех элементов данных. Сложность по памяти $O(n)$.

2.5 Компонентный коэффициент выбросов

Алгоритм COF почти полностью повторяет алгоритм LOF, но отличается способом вычисления расстояния между двумя точками. Расстояние между двумя точками вычисляется при помощи метода "кратчайшей цепочки". Для всех элементов набора данных строится матрица расстояний на основе евклидова расстояния. После этого при помощи алгоритма Флойда-Уоршелла между двумя точками ищется кратчайший путь. Длина этого кратчайшего пути и будет мерой расстояния между двумя точками.

Листинг 2.3 — Алгоритм Флойда — Уоршелла

```

1 for k = 1 to n
2     for i = 1 to n
3         for j = 1 to n
4             W[i][j] = min(W[i][j] , W[i][k] + W[k][j])

```

2.6 Собирающее данные для анализа ПО

Для применения метода на практике было разработано ПО, которое позволяет собирать данные для анализа(телеметрию). В состав приложения будет входить

плагин для графического редактора, backend-сервер, frontend. Так же на бекенде будет размещена база данных где будут размещаться необработанные данные.

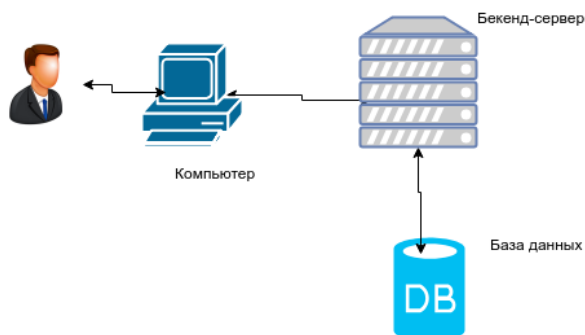


Рисунок 2.4 — Общая архитектура приложения

Результатом работы ПО будет неразмеченный набор данных, содержащий фиксированное количество атрибутов.

2.6.1 Собираемые данные

Основные собираемые данные приведены в таблице 2.1:

Таблица 2.1 — Описание собираемых данных

Информация о	Источник данных	Данные
Инструменты	KisTool::activate, KisTool:deactivate	CountUse float64 Time float64 ToolName string Timestamp float64
Действия	KisMainWindows actioncollection()	CountUse float64 TimeUse float64 ActionName string Timestamp float64
Свойства изображений	KisDocument::saveFile()	ColorProfile string ColorSpace string Height float64 Width float64 Size float64 NumLayers float64 Timestamp float64

2.6.2 Клиентская часть

Был разработан плагин для графического редактора Krita, который позволяет собирать телеметрию с пользователей. Телеметрия собирается только во время работы программы и при закрытия программы собранные данные удаляются с компьютера пользователя. Телеметрия отправляется через определенные интервалы времени на удаленный сервер. Записи о действиях и инструментах отправляются каждые n минут. Информация о свойствах изображения собирается при сохранения изображения.

Собираемые метрики агрегируются на стороне клиента в http-запрос. Пример тела запроса представляет собою JSON, пример этого JSON приведен ниже(форматирование нарушено в целях наглядности).

Листинг 2.4 — Тело http-запроса

```
1 {  
2 "Name": [  
3 {  
4 "Param1": 1,  
5 "Param2": 4581,  
6 "Timestamp": 12214  
7 },
```

2.6.3 Серверная часть

На сервере метрики обрабатываются и заносятся в базу данных в формате JSON. Сервер способен принимать и обрабатывать много параллельных запросов пользователей. Наличие временной отметки у элементов данных позволяет извлекать данные за определенный промежуток времени.

Листинг 2.5 — Пример хранимого элемента в базе данных

```
1 { "_id" : ObjectId("5b08a54677d37ee1964df0b7"), "actions" : [ {  
    "countuse" : 1, "sources" : 0, "actionname" : "edit\_cut", "time" :  
    1527293254 } ] }
```

2.6.4 Динамическое изменение данных

Инструменты и действия(actions) могут меняться достаточно часто в процессе разработки графического редактора. Поэтому неразумно задавать статически эти метрики в коде. В коде бекенд-сервера реализована поддержка добавления новых элементов. Раз в сутки просыпается новая горутина(легковесный тред), которая пробегается по базе данных и ищет новые инструменты и действия. После этого она записывает их в текстовый файл. Подобное разумно применять не только для

инструментов и действий, но и для любых часто изменяющихся данных. В будущем возможно расширения этой системы.

3 Технологический раздел

3.1 Выбор средств разработки

3.1.1 Выбор целевой платформы

Программный продукт поиска аномалий не заточивается под работу на конкретной ОС. Его корректная работа протестирована на ОС Windows и OS Linux. Данный выбор обусловлен широкой распространенностью ОС Windows и Linux, большим количеством средств разработки для данных платформ, что дает возможность выбирать язык программирования и сопутствующие инструменты, ориентируясь на возможность простого и надежного решения рассматриваемой задачи, а не на ограничения целевой платформы.

Программный комплекс для сбора данных разрабатывается для OS Linux, в силу особенностей функционирования графического редактора, плагин для которого входит в состав этого программного комплекса.

3.1.2 Выбор языка программирования

Для написания программы поиска аномалий используется язык C++, так как с версии C++11 он сочетает в себе возможности функционального и объектно-ориентированного подходов. С помощью функционального подхода удобно описывать алгоритмы и математические выражения, необходимые для решения поставленной задачи. Использование функциональных средств существенно упрощает описание операций над последовательностями входных данных. В то же время, применение объектно-ориентированного подхода дает возможность проектирования приложения в терминах объектов предметной области и их поведения, что позволяет контролировать сложность программы, предлагать наглядные и емкие абстракции и четко выделять сферы ответственности программных модулей. Также объектно-ориентированные языки позволяют построение приложения с графическим интерфейсом пользователя с учетом известных рекомендаций и правил. Дополнительным преимуществом языка C++ является большая база документации по самому языку и стандартным библиотекам.

В качестве языка написания плагина к графическому редактору был выбран язык C++ в силу того, что сам редактор поддерживает только плагины на этом языке. Для написания бекенд-сервера был выбран язык Golang вместе с библиотекой mgo для доступа к базе данных. Плюсы этого языка:

- а) Скорость разработки
- б) Производительность
- в) Удобная реализация легковесных потоков

Для предварительной обработки данных использовался язык `bash`. `Bash` - это мощный и простой в использовании язык сценариев. Он позволяет легко перемещать курсор и редактировать текст команды в командной строке, поддерживает историю команд: дает возможность повторить или при необходимости изменить команду, которая была введена в командной строке ранее. Позволяет легко указать команде, откуда брать входные данные и куда направлять выходные данные. Поддерживаются псевдонимы - создание кратких обозначения для однострочных команд.

3.1.3 Выбор среды разработки и отладки

Из-за использования сразу нескольких языков, в процессе работы над программным комплексом, было использовано несколько сред разработки. `Qt Creator` - современная кросс-платформенная среда разработки, которая предоставляет широкие возможности разработки программ на `C++`, а так же их отладки. `Qt Creator` так же поддерживает работу со сторонними плагинами, что позволило использовать плагин работы с системой контроля версий.

Для работы с языком `Golang` был использован текстовый редактор `Visual Studio Code` с набором плагином для языка `Golang`. Набор плагинов к текстовому редактору позволяет осуществлять запуск и отладку кода прямо из редактора, а так же позволяет осуществлять автоматическое форматирование кода.

Для работы с `bash` использовался `nano` - консольный текстовый редактор для `Unix` и `Unix`-подобных операционных систем, основанный на библиотеке `curses`. В настоящее время включен в дистрибутивы `Ubuntu` по умолчанию и в установке не нуждается.

3.2 Выбор базы данных

Для сохранения элементов данных была выбрана документо-ориентированная база данных `MongoDb`. Основные преимущества `MongoDb`:

- Отсутствие четкой структуры хранения данных
- Возможность выгружать и загружать файлы в `JSON`-формате.
- Быстрая вставка элементов
- Быстрое извлечение простых данных

3.3 Система контроля версий

В процессе разработки программы использовалась система контроля версий `Git`. Система контроля версий позволяет вносить в проект атомарные изменения, направленные на решения каких-либо задач. В случае обнаружения ошибок или изменения требований, внесенные изменения можно отменить. Кроме того, с помощью системы контроля версий решается вопрос резервного копирования. Особенности `Git`:

- Предоставляет широкие возможности для управления изменениями проекта и просмотра истории изменений
- Данная система контроля версий является децентрализованной, что позволяет иметь несколько независимых резервных копий проекта.
- Поддерживается хостингом репозитория GitHub и Gitlab.
- Поддерживается средой разработки Qt Creator и Visual Studio Code.

3.4 Библиотека Galgo

Для подбора параметров алгоритмов была использована библиотека GALGO версии 2. Она позволяет не вдаваясь в нюансы работы генетических алгоритмов, подобрать необходимые значения параметров алгоритмов. В качестве функции, необходимой для работы генетического алгоритма, использовался алгоритм поиска аномалий. В качестве выходного значения функции использовался AUC ROC.

3.4.1 Формат файлов

На вход программе поиска аномалий подается файл следующего формата: одна строка содержит в ненормализованном виде атрибуты данных одного элемента.

Листинг 3.1 — Пример входного файла программы поиска аномалий

```
1 9 10 23 4
2 1 3 2 1
3 1 4 9 12
4 6 7 4 44
```

Для проверки корректности алгоритма использовался иной формат входных файлов. Одна строка содержит в нормализованном виде n-1 атрибутов одного элемента, на n-ой позиции в строке расположена метка аномальности объекта ('yes' или 'no'). Атрибуты элементов разделяются запятой.

Листинг 3.2 — Пример входного файла программы поиска аномалий для проверки корректности алгоритма

```
1 0.467651,0.321584,0.76888,0.24663,0.838799,0.099737,0.29834,1.0,'no'
2 0.496412,0.220491,0.776032,0.316598,0.919973,0.089145,0.279479,2.0,'no'
3 0.519133,0.404464,0.768012,0.334978,0.801622,0.092369,0.271238,3.0,'no'
4 0.19965,0.547373,0.374284,0.362223,0.817017,0.0,0.177913,4.0,'yes'
5 0.847261,0.286361,0.0,0.217792,0.0,0.019135,1.0,5.0,'no'
```

3.5 Библиотека KUserFeedback

В качестве вспомогательной библиотеки для сбора статистики используется библиотека KUserFeedback компании KDAB. Эта библиотека включает в себя C++

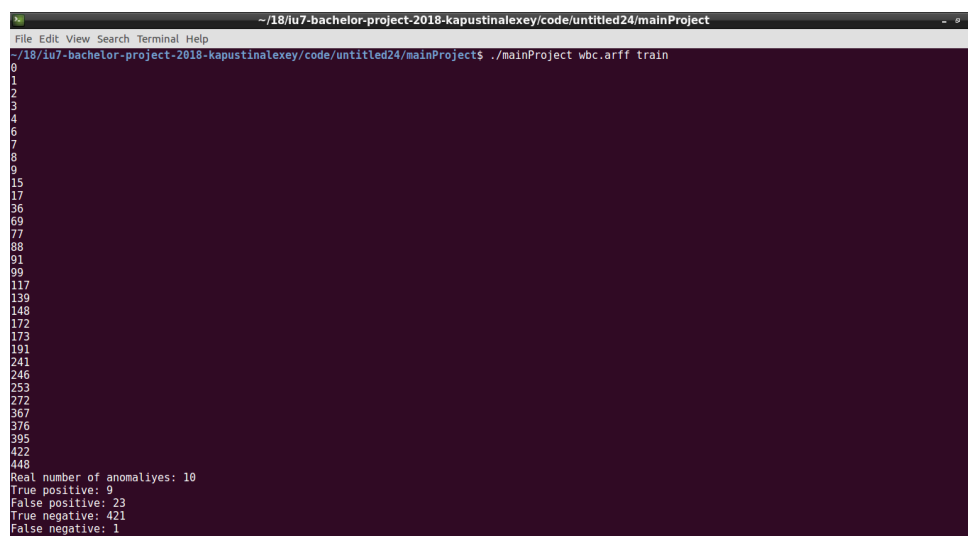
Qt клиентскую часть, а так же сервер, написанный на PHP. Их сервер и значительная часть функций не требуется, будет использоваться только часть собирающую телеметрию. KUserFeedBack позволяет собрать библиотеку по частям и линковать к нашему приложению только необходимые модули. В программе используется модуль Core. Модуль Core содержит абстрактный класс источника данных, от которого можно наследовать различные источники данных.

3.6 Установка программного обеспечения

У программы поиска аномалий нет зависимостей от внешних служб и программ, установленных в операционной системе, а также отсутствует необходимость в модификации системных и пользовательских настроек, поэтому было принято решение не разрабатывать инсталлятор. Исполняемым файлом программы является mainProject.exe(windows) или mainProject(linux).

Программа для сбора данных требует установленного графического редактора Krita с плагином телеметрии. Исполняемым файлом бекенд-сервера является файл server.

3.7 Пример работы программы



```
~/18/iu7-bachelor-project-2018-kapustinalexey/code/untitled24/mainProject
File Edit View Search Terminal Help
~/18/iu7-bachelor-project-2018-kapustinalexey/code/untitled24/mainProject$ ./mainProject wbc.arff train
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
Real number of anomalies: 10
True positive: 9
False positive: 23
True negative: 421
False negative: 1
```

Рисунок 3.1 — Пример работы программы для набора данных wbc

Приведен пример работы программы на операционной систему Ubuntu 16.04(Linux). Имя программы - mainProject, первый аргумент - обрабатываемый файл, второй аргумент - режим работы. В результате работы программы можно увидеть список найденных аномалий, а так же количество правильно или неверно классифицировавшихся элементов.

3.8 Использование программы для поиска аномалий

Запуск программы осуществляется из командной строки. Первым аргументом нужно передать имя файла для анализа. Вторым передается режим работы программы. Он может быть "train" что означает, что на вход должны подаваться размеченные файлы. Этот режим означает, что в результате работы программы будет получен список элементов, помеченный программой как аномальные, а так же метрики полноты, точности, площади под графиком РОК-кривой. Любое другое слово в качестве второго аргумента или его отсутствие означает, что на вход подаются неразмеченные данные и в результате работы программы будет получен только список аномалий.

4 Исследовательский раздел

Ошибки первого и второго рода - ключевые понятие математической статистики, помогающие проверять статистические гипотезы. Они так же используются в областях, где речь идет о принятии бинарного решения(да/нет). Задача поиска аномалий сводится к задаче бинарной классификации, где вопрос, помогающий осуществить классификацию, звучит следующим образом: "Является ли точка аномалией?".

Возьмем гипотезу H_0 : "Точка является аномалией". Тогда можно составить следующую таблицу классификации ошибок:

Таблица 4.1 — Описание ошибок первого и второго рода

H_0	Верная	Ложная
Отклоняется	Ошибка первого рода	Решение верное
Не отклоняется	Решение верное	Ошибка второго рода

В машинном обучении используют другую терминологию для описания ошибок первого и второго рода. Это позволяет выделить 4 класса результата принятия решения(а не 3 как принято в математической статистике). На основании значе-

Таблица 4.2 — Описание ошибок первого и второго рода в терминологии машинного обучения

H_0	Верная	Ложная
Отклоняется	Ложное негативное	Истинное негативное
Не отклоняется	Истинное позитивное	Ложное позитивное

ний этих метрик составляются метрики полнота, точность и график под площадью РОК-кривой(описан выше)

4.1 Полнота и точность. Площадь по РОК-кривой

Точность(Precision) - это доля объектов, названных классификатором положительными и при этом действительно являющимися положительными

$$precision = \frac{TP}{TP + FP} \quad (4.1)$$

TP- количество истинно позитивных, FP - количество ложно позитивных

$$precision = \frac{TP}{TP + FN} \quad (4.2)$$

Полнота(recall) показывает объектов положительного класса из всех объектов положительного класса нашел алгоритм. Мера точности не позволяет нам записывать все объекты в один класс, так как в этом случае мы получаем рост уровня ложно позитивных. Полнота демонстрирует способность алгоритма обнаруживать данный класс вообще, а точность— способность отличать этот класс от других классов. Полнота и точность могут применяться в условиях когда классы несбалансированы[24]. Поэтому они могут применять в задаче поиска аномалий, которая подразумевает несбалансированность классов. Так же существует метрика называемая F1-показатель. Она является средним гармоническим величин полноты и точности. Метрика площади под РОК-кривой описывается выше, здесь же приведем формулу её расчёта.

$$TPR = \frac{TP}{TP + FN} * 100 \quad (4.3)$$

$$FPR = \frac{FP}{TN + FP} * 100 \quad (4.4)$$

$$ROC = \frac{1 + TPR - FPR}{2} \quad (4.5)$$

4.2 Сравнение алгоритмов поиска аномалий

Для проверки работоспособности алгоритмов поиска аномалий на неразмеченных данных, эти алгоритмы проверялись на размеченных данных. Для этого работа алгоритма поиска аномалий была протестирована на двух наборах данных:

Таблица 4.3 — Характеристики датасетов, метрики полноты и точности

Набор данных	Кол-во элем.	Кол-во атриб.	Полнота	Точность	Кол-во аном.
WBC	453	9	0.99	0.94	10
KDDCUP99	60853	41	0.9	0.88	246

Таблица 4.4 — Сравнение алгоритмов поиска аномалий

Алгоритм	AUC ROC WBC	F1 WBC	AUC ROC KDD	F1 KDD
LoOp	0.98	0.72	0.68	0.05
ODIN	0.62	0.80	0.80	0.06
KDEOS	0.25	0.64	0.61	0.05
LDOF	0.64	0.96	0.88	0.07
Разр. алгоритм	0.92	0.97	0.71	0.06

Проведем сравнения с другими алгоритмами поиска аномалий. Как можно увидеть

результатов метрик AUC ROC и F1, алгоритмы по-разному классифицируют разные наборы данных. Например, алгоритм LoOP показывает высокий AUC ROC на первом наборе данных, но на втором наборе данных его показатели значительно снижаются. В свою очередь, алгоритм ODIN показывает низкие результаты, по сравнению с другими алгоритмами, на первом наборе данных, но на втором наборе данных его AUC ROC высок. Разработанный алгоритм показывает средние значения AUC ROC, но высокие значения показателя F1, что позволяет утверждать, что этот алгоритм жизнеспособен и возможно его применение на определенных наборах данных.

Заключение

В данной работе были исследованы различные алгоритмы поиска аномалий, изучены их достоинства и недостатки. В частности были подробно изучены метрические методы поиска аномалий.

Был предложен и реализован новый метод поиска аномалий, основанные на модификации и ансамлировании уже существующих методов поиска аномалий. Также был сделан программный комплекс сбора данных для анализа, состоящий из плагина для графического редактора и бекенд-сервера.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *MachineLearning.ru* Профессиональный информационно-аналитический ресурс, посвященный машинному обучению. Выборка. <https://goo.gl/7gjJ6p>.
2. ГОСТ 20886-85: Организация данных в системах обработки данных. Термины и определения. <http://www.gostrf.com/normadata/1/4294832/4294832686.pdf>.
3. *Википедия*. Data set, Задача классификации. <https://en.wikipedia.org>.
4. Яминов, Булат. Генетические алгоритмы. <http://rain.ifmo.ru/cat/view.php/theory/unsorted/genetic-2005>.
5. *Entefy*. Data in the digital universe doubles in size every 2 year. <https://www.entefy.com/blog/post/261/Data-in-the-digital-universe-doubles-in-size-every-2-years>.
6. Дьяконов, Александр. Поиск аномалий (Anomaly Detection) / Александр Дьяконов. — 2017. <https://goo.gl/Z43Ne9>.
7. *F.E., Grubbs*. Procedures for Detecting Outlying Observations in Samples. Technometrics / Grubbs F.E. — 1969.
8. *Moya M.M., Hush D.R.* Network Constraints and Multi-objective Optimization for One-class Classification. Neural Networks / Hush D.R. Moya M.M. — 1996.
9. *Chandola V Banerjee A, Kumar V.* Anomaly Detection: A Survey / Kumar V. Chandola V, Banerjee A. — ACM Computing, 2009.
10. *Goldstein M, Uchida S.* Behavior Analysis Using Unsupervised Anomaly Detection / Uchida S. Goldstein M. — The 10th Joint Workshop on Machine Perception and Robotics, 2014.
11. Андрей, Гахов. Интеллектуальный анализ данных / Гахов Андрей. — Харьковский национальный университет имени В.Н. Карамзина, 2014.
12. *Hodge V., Austin J.* A survey of outlier detection methodologies / Austin J. Hodge V. — Artificial intelligence review, 2004.
13. *Knox, Edwin M.* Algorithms for Mining Distance-Based Outliers in Large Datasets / Edwin M. Knox, Raymond T. Ng. — University of British Columbia, 1998.
14. *S. Bayers, A.Raftery.* Nearest Neighbor Clutter Removal for Estimating Features in Spatial Point Processes / A.Raftery S. Bayers. — Journal of the American Statistical Association, 1998.
15. *Hautamäki, Ville.* Outlier Detection Using k-Nearest Neighbour Graph / Ville Hautamäki. — University of Joensuu, Department of Computer Science Joen-

suu, Finland, 2004. — p. 1-4.

16. *Callahan, P. B.* A decomposition of multidimensional point sets with applications to k-nearestneighbors and n-body potential fie / P. B. Callahan, S. R. Kosara. — s. Journal of the Association for Computing Machin, 1995. — p. 67-90.

17. *Goldstein, Markus.* A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data / Markus Goldstein. — Seichi Uchida, 2016.

18. *M., Goldstein.* Anomaly Detection in Large Datasets / Goldstein M. — University of Kaiserslauterna, 2014.

19. *Kriegel, Hans-Peter.* LoOP: Local Outlier Probabilities / Hans-Peter Kriegel. — Institut für Informatik, Ludwig-Maximilians Universität München, 2009. — p. 1-3.

20. *Rousseeuw, Leroy.* Robust Regression and Outlier Detection / Leroy Rousseeuw. — John Wiley and Sons, 1996.

21. *B.Chu Chia-Hua Ho, Cheng-Hao Tsa.* Warm Start for Parameter Selection of Linear Classifiers / Cheng-Hao Tsa B.Chu, Chia-Hua Ho. — National Taiwan University, 2015.

22. *Александр, Гуцин.* Методы ансамблирования обучающихся алгоритмов / Гуцин Александр. — Московский физико-технический институт, 2015.

23. *Панченко, Т.В.* Генетические алгоритмы / Т.В. Панченко. — Издательский дом «Астраханский университет», 2007. — ст. 6-20.

24. *Лабинцев, Егор.* Метрики в задачах машинного обучения. <https://habr.com/company/ods/blog/328372/>.