Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий

Работа допущена к защите
Директор ВШИИ
_____ В.А. Мулюха
«_____» _____ 2022 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ
РАЗРАБОТКА И ИССЛЕДОВАНИЕ АЛГОРИТМОВ ИЗОЛИРУЮЩЕГО ЛЕСА ДЛЯ ОБНАРУЖЕНИЯ АНОМАЛИЙ В ТРАНЗАКЦИОННЫХ ДАННЫХ

по направлению подготовки 02.04.01 Математика и компьютерные науки
Направленность (профиль) 02.04.01_02 Организация и управление суперкомпьютерными системами

Выполнил
студент гр. 3540201/00201 Е.И. Филиппова

Руководитель
директор ИКНТ,
д.т.н., профессор Л.В. Уткин

Консультант
по нормоконтролю Ю.П. Хотякова

Санкт-Петербург
2022

РЕФЕРАТ

На 63 с., 38 рисунков, 4 таблицы, 1 приложение

КЛЮЧЕВЫЕ СЛОВА: РАСПОЗНОВАНИЕ АНОМАЛИЙ, ДРЕВОВИДНЫЕ АЛГОРИТМЫ, ИЗОЛИРУЮЩИЙ ЛЕС, ОЦЕНКА ПЛОТНОСТИ РАСПРЕДЕЛЕНИЯ, ВЕСОВЫЕ КОЭФФИЦИЕНТЫ..

Тема выпускной квалификационной работы: «Разработка и исследование алгоритмов изолирующего леса для обнаружения аномалий в транзакционных данных».

Предметом исследования является модификация существующего алгоритма изолирующего леса (далее — ИЛ), а целью – увеличение эффективности обнаружения аномалий алгоримом изолирующего леса путем его модификации. В работе применялись методы математической статистики, машинного обучения и объектно-ориентированного программирования. Был исследован алгоритм ИЛ и его модификации: расширенный ИЛ, ИЛ сейсмической активности, обобщенный ИЛ. Также была предложена и изучена собственная модификация ИЛ – весовой изолирующий лес. Реализация алгоритмов выполнялась на языке C++ 20 без использования сторонних библиотек. Набор данных для тестирования содержал 16 млн транзакций, собранным за примерно 5 месяцев работы. Разработанная и реализованная модель весового изолирующего леса в ходе тестирования обнаружения аномалий на деперсонализированных транзакционных данных показала себя наиболее сбалансированной моделью ИЛ. Выявление диапазона параметров количества изолирующих деревьев и объема выборки позволяет достичь большей точности, чем у других модификаций ИЛ: моделей расширенного ИЛ и ИЛ сейсмической активности.

ABSTRACT

63 pages, 38 figures, 4 tables, 1 appendices

KEYWORDS: ANOMALY DETECTION, TREE ALGORITHMS, ISOLATION FOREST, DESITY ESTIMATION, WEIGHTING COEFFICIENTS..

The subject of the graduate qualification work is «Development And Research Of Isolating Forest Algorithms For Anomaly Detection In Transactional Data».

The subject of the study is the modification of the existing the isolating forest algorithm (hereinafter — IF), and the goal is to increase the efficiency of anomaly detection via the isolating forest algorithm by modifying it. Methods of mathematical statistics, machine learning and object-oriented programming were used in the work. The IF algorithm and its modifications were investigated: expanded IF, IF of seismic activity, generalized IF. A proprietary modification of the IF, a weight insulating forest, was also proposed and studied. The algorithms were implemented in C++ 20 without using third-party libraries. The data set for testing contained 16 million transactions collected over approximately 5 months of operation. The developed and implemented model of the weight isolating forest during testing of anomaly detection on depersonalized transactional data proved to be the most balanced IF model. Identification of the range of parameters of the number of isolating trees and the sample size allows to achieve greater accuracy than other modifications of the IF: models of extended IF and IF seismic activity.

СОДЕРЖАНИЕ

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ИЛ Изолирующий лес.
ОИЛ Обобщённый изолирующий лес.
РИЛ Расширенный изолирующий лес.
ИЛСИ Изолирующий лес сейсмической интерпретации.
ВИЛ Весовой изолирующий лес.
ИД Изолирующее дерево.
EM Expectation Maximization.
KNN K-Nearest Neighbors.
KNNW K-Nearest Neighbors Weighted.
LOF Local Outlier Factors.
SLOF Simplified Local Outlier Factors.
BST Binary Search Tree.
AUC Area Under Curve.

ВВЕДЕНИЕ

Выпускная квалификационная работа магистра посвящена исследованию алгоритмов изолирующего леса (далее — ИЛ) и его модификаций для решения задачи обнаружения аномалий в транзакционных данных.

Актуальность исследования

В настоящее время важность разработки прикладных моделей для определения аномалий стремительно возрастает в связи с мировой цифровизацией. Обнаружение аномалий является важной задачей интеллектуального анализа данных и часто одним из первых шагов при получении новых данных. С наступлением информатизации кредитно-банковской системы, мошенничество в этой сфере активно развивается [1]. Задача обнаружения аномалий ставится как задача поиска объектов, не соответствующих предполагаемому типовому поведению в исходном наборе данных. Решение данной задачи обнаруживает объекты, представляющие *интерес* для исследователей.

Объектом исследования настоящей выпускной квалификационной работы являются методы обнаружения аномалий на базе ИЛ и его модификаций.

Предметом исследования является модификация существующего алгоритма ИЛ и его реализация.

Цель настоящей дипломной работы — увеличить эффективность обнаружения аномалий алгоритмом изолирующего леса путем его модификации. Для достижения поставленной цели необходимо решить следующие **задачи работы**:

A. Выполнить анализ методов обнаружения аномалий на базе алгоритма ИЛ и его модификаций;
B. Разработать собственную модификацию алгоритма ИЛ;
C. Реализовать модификации алгоритмов ИЛ: расширенный изолирующий лес (далее — РИЛ), обобщенный изолирующий лес (далее — ОИЛ), изолирующий лес сейсмической активности (далее — ИЛСА) и весовой изолирующий лес;
D. Определить эффективность реализованных методов в условии использования на реальных транзакционных данных.

Теоретической и методологической базой исследования основой выпускной квалификационной работы послужили исследования зарубежных специалистов в области обнаружения аномалий. В работе [14] авторы Liu, Fei Tony и Ting, Kai Ming и Zhou, Zhi-Hua предложили новый алгоритм iForest (в настоящей момент

известный как ИЛ), который благодаря своей простоте и скорости, является одним из самых популярных алгоритмов обнаружения аномалий. Благодаря своей популярности, было предложено множество вариантов модификаций ИЛ, которые подробно были рассмотрены в [7] авторами Buschjager, Sebastian и Honysz, Philipp-Jan и Morik, Katharina.

Информационной базой для разработки выпускной квалификационной работы служат материалы, собранные в процессе прохождения производственной и преддипломной практик, а также значения, полученные при изучении учебных дисциплин “Машинное обучение”, “Глубокое машинное обучение”, “Элементы теории вероятности и линейной алгебры” и “Современные технологии анализа данных”.

Научная новизна выпускной квалификационной работы заключается в разработке, теоретическом обосновании и тестировании модификации алгоритма ИЛ — весовом изолирующем лесе (далее — ВИЛ).

Практическая значимость заключается в разработке модели модифицированного ИЛ по обнаружению аномалий в транзакционных данных, реализации и практическом подтверждении эффективности настоящей модели, а также в выявлении преимуществ и недостатков в существующих модификации ИЛ в сравнении с разработанной.

Объем и структура работы Выпускная квалификационная работа магистра состоит из введения, четырех глав и заключения. Полныи объем работы составляет 63 страницы, включая 38 рисунков и 4 таблицы. Список литературы содержит 26 наименовании.

ГЛАВА 1. АНАЛИЗ МЕТОДОВ РЕШЕНИЯ ЗАДАЧИ ОПРЕДЕЛЕНИЯ АНОМАЛИЙ

В рамках магистерской диссертации рассматривается задача обнаружения аномалий в транзакционных данных:

1.1 – определяется понятие аномалии, приводится общая постановка задачи обнаружения аномалий;

1.2 – проводится анализ моделей и методов обнаружения аномалий, выявление недостатков существующих способов решения.

1.1 Понятие аномальности и постановка задачи обнаружения аномалий

Часто формулировка задачи определения аномалий зависит от данных, требующих анализа, и поставленной цели. Не существует стандартизированного вида задачи детектирования аномалий, известны ее различные интерпретации [3], [9], [13]. Принято аномалиями считать то, что не укладывается в правила и законы, действующие для рассматриваемых данных.

Обычно для постановки задачи обнаружения аномалии необходимо определить описание нормальной работы системы. Описание же ситуаций, не соответствующие нормальному, чаще всего является неполным либо отсутствует. В некоторых случаях требуется разработать модель, согласно которой система работает штатно, для того, чтобы в дальнейшем иметь возможность предсказывать, является ли та или иная ситуация “аномальной”. Такой упрощенный подход не лишен следующих недостатков:

- перечисление всех возможных нормальных вариантов описания работы системы требует большого количества ресурсов; задача усложняется тем, что разница между нормальным экземпляром и аномальным может быть незначительна;
- предметная область влияет на определение точного понятия “аномалия” – одни и те же колебания измеряемой величины могут иметь различное значение в зависимости от того, что оно отображает: будь то температура тела (медицинская специфика) или показания цен на бирже;
- наличие размеченных данных и их доступность часто ограничены;
- один экземпляр наблюдения может не проявлять себя в качестве аномального, при этом с течением времени и на основе информации от других

экземпляров, решение относительно аномальности объекта может подвергнуться изменению – таким образом, описание аномальных экземпляров может меняться во времени;
- аномальные наблюдения антропогенного характера приближены к нормальным экземплярам.

Методы определения аномалий базируются на некотором строгом представлении понятия, означающего “отклонение от нормы”. Чаще всего данное представление отталкивается от контекста поставленной задачи и априорной информации. В простом случае, когда данные представляют собой набор элементов некоторого множества, любой другой элемент, не принадлежащий этому множеству, априори можно считать аномалией [8]. Подобное “характеристическое” множество, определяющее принадлежность элемента к классу аномалий, не всегда существует.

Другой важный аспект, оказывающий влияние на определение аномалий является вид (природа) анализируемых данных. Чаще всего данные представлены элементами выборки, каждый из которых задается набором атрибутов, бинарных, категориальных или непрерывных. Представление данных влияет на методы, которые возможно использовать для определения аномалий, а также на возможный характер самих аномалий.

В зависимости от предметной области аномалии можно разделить на следующие категории:
- аномалия-точка: отдельный объект можно считать аномалией относительно всего набора данных – наиболее простой и исследуемый тип;
- контекстная аномалия: отдельный объект нельзя считать аномалией, но в зависимости от контекста данных, в которых он встречается, сочетание характеристик может считаться аномальным. Например, летом температура +20 градусов – нормальное явление, при этом эта же температура зимой является аномальной;
- групповые аномалии: когда в наборе данных можно выделить подгруппу элементов, от нее отличающихся. При этом сами элементы по отношению друг к другу являться аномалиями не будут.

Данная работа фокусируется на определении аномалии без учителя.

Постановка задачи Пусть имеется набор данных, содержащий выбросы, цель – найти эти выбросы. Предполагаем, что известна выборка $S = \{x_1, \ldots, x_N\}$ из N

наблюдений $x_i \in \mathbb{R}^d \subseteq \mathcal{X}$ из неизвестного распределения $\mathcal{D}$. Необходимо присвоить каждому наблюдению из $\mathcal{S}$ оценку, которая измеряет его выброс.

После определения оценки производится бинаризация по некоторому признаку, определяющему, является ли элемент x_i аномалией или нет.

1.2 Распознавание аномалий изолирующими лесами

Обнаружение аномалий является важной задачей интеллектуального анализа данных и часто одним из первых шагов при получении новых данных. В финансовом секторе оно используется для обнаружения транзакционного мошенничества [2], отмывания денег [19], [10] и для решения многих других связанных с этим проблем [6].

Благодаря своей простоте и скорости, изолирующий лес (далее – ИЛ) является одним из самых популярных алгоритмов обнаружения аномалий [14]. Он основан на ключевом наблюдении, что деревья решений имеют тенденцию изолировать примеры-аутсайдеры на относительно ранних этапах дерева. Таким образом, длина пути примера при сортировке в дереве дает приближённый индикатор необычности наблюдения. ИЛ использует это эмпирическое понимание в ансамблевом алгоритме, который обучает несколько деревьев изоляции на бутстреп-выборках и оценивает наблюдения на основе их средней длины пути [11].

Благодаря своей популярности, было предложено множество вариантов ИЛ. Например, ИЛСИ предлагает выбирать разделение/функцию более тщательно, вводя критерий разделения, тогда как расширенный изолированный лес РИЛ использует произвольные случайные наклоны вместо выровненных по оси сплайнов для расщепления для улучшения его производительности.

Несмотря на то, что существует приличный объем исследований и модификаций ИЛ, в теоретическом понимании ИЛ существует пробел. В частности, кажется, что нет прямой связи между производительностью ИЛ и его вариациями, и предположениями, которые затрагивают изначальное распределение данных. Исследованы подходы на основе ИЛ с точки зрения распределения данных, о чём сказано ниже. Все подходы на основе ИЛ аппроксимируют базовое распределение вероятностей и рассматривают среднее значение длины пути как приближение к смешанному весу, если данные сгенерированы смешанным распределением.

Обобщённый ИЛ основан на вышесказанных идеях. Сравнение подходов и таких модификаций ИЛ, как ОИЛ, РИЛ, ИЛСИ, дало возможность исследовать, реализовать и протестировать собственную модификацию ИЛ.

1.2.1 Подходы, основанные на плотности распределения

Подходы на основе плотности предполагают, что наблюдения взяты из смешанного распределения, где по крайней мере одна из выборок является “редкой”. Подходы на основе плотности требуют двухэтапной процедуры, причем оба этапа часто взаимосвязаны [7].

Сначала необходимо смоделировать базовое распределение настолько хорошо, насколько это возможно, а затем определить, какие из наблюдений могут быть выбросами. Частным примером такого подхода является модель гауссовой выборки, которая предполагает смесь гауссовых распределений и подгоняется с помощью алгоритма в стиле Expectation Maximization (далее – EM) [20].

В качестве более быстрой, свободной от предположений альтернативы предложены методы оценки плотности на основе деревьев. Эти подходы опираются на вариации деревьев решений для точной аппроксимации базового распределения и формулируют некоторые правила пост-подгонки для обнаружения выбросов с помощью деревьев [17], [18]. Хорошо известно, что большинство вариаций деревьев решений могут аппроксимировать любое распределение с достаточной точностью и даже случайно подобранные деревья сходятся к истинному базовому распределению при достаточном количестве обучающих данных [18]. В целом, обучение (случайных) деревьев происходит быстро, поскольку для этого требуется только выборки набора различных разбиений и соответствующей сортировки данных.

Более того, деревья могут быть объединены в ансамбль для стабилизации их производительности, который может быть легко распараллелен, таким образом сохраняя преимущества производительности деревьев [4].

1.2.2 Подходы на основе изоляции

Подходы, основанные на изоляции, предполагают, что некоторые наблюдения могут быть легко изолированы от остальных и поэтому являются выбросами. Пожалуй, самым популярным методом в этом семействе является изолирующий лес [14]. ИЛ использует ансамбль случайно построенных деревьев для оценки сте-

пени различия, то есть вероятности выброса каждого наблюдения путем измерения его средней длины пути.

Более формально, рассмотрим двоичное дерево решений, в котором каждый узел выполняет сравнение: $x_i \leqslant t$, где $i \in N$ – это случайно выбранный индекс признака, а t – это случайный порог, выбранный из имеющихся значений признаков в $\mathcal{S}$, векторного пространства признаков.

Для каждого наблюдения мы подсчитываем количество сравнений $h(x)$, необходимых для обхода дерева, начиная с его корневого узла. Пусть x – длина пути, пусть $E[h(x)]$ -– средняя длина пути по ансамблю деревьев. Наблюдения с выбросами имеют тенденцию изолироваться раньше во время обхода дереве, что указывает на то, что сами деревья имеют тенденцию изолировать наблюдения с выбросами. Другими словами:

$$s(x, N) = 2^{-\frac{E(h(x))}{c(N)}} \tag{1.1}$$

правило оценки, где (N) – гармоническое число, зависящее от размера набора данных N. В классическом ИЛ данное правило скоринга обосновывалось эмпирическими наблюдениями, но позже авторы алгоритма привели более математическую доказательную базу [14]. Они утверждают, что средняя длина пути наблюдений в случайно подобранных деревьях на равномерно распределенных наблюдениях из интервала $[l, u]$ меньше для точек вблизи границы интервала u и l. Затем они показывают, что в этом случае распределение средней длины пути задается числом Каталана, которое, в свою очередь, может быть аппроксимировано оригинальным рейтинговым баллом, используемым ИЛ.

ИЛ строит деревья, используя случайную комбинацию разбиения и значение признака. Таким образом, естественным продолжением этого подхода является более тщательный выбор комбинации разбиения и признака с помощью критерия разбиения [15]. Модификация ИЛ – ИЛСИ – алгоритм, который использует дисперсию выборки, чтобы вычислить оценки каждого сплита.

Существует другая предложенная модификация ИЛ под названием расширенный изолирующий лес (РИЛ) [11]. РИЛ алгоритм улучшает стратегию разбиения оригинального ИЛ формулировками, рассматривая выбор случайного наклона, а не случайной переменной и значения. Мотивация этой стратегии заключается в расщеплении ограничения ИЛ, которое учитывает только горизонтальные и вертикальные срезы ветвей, что приводит к артефактам в результирующих оценках аномалий.

1.2.3 Подходы, основанные на вероятностной близости

Подходы, основанные на близости распределений, предполагают, что похожие объекты ведут себя одинаково. Таким образом, редкие выбросы могут иметь всего несколько похожих объектов поблизости. Подходы, основанные на близости, обычно вводят метрику расстояния (или функцию сходства) для количественной оценки различий в наблюдениях. Это, пожалуй, самый большой класс методов обнаружения выбросов.

Принципиальное их сравнение с подходами, основанными на изоляции, представлено далее. Это семейство алгоритмов в основном состоит из двух различных направлений. Методы ближайших соседей можно рассматривать как глобальные методы. которые основывают свои оценки на окрестностях $k \in N$ точек для данного наблюдения.

Например, K-Nearest Neighbors (далее — KNN) [18] использует наибольшее расстояние в окрестности k в качестве правила подсчета баллов, в то время как K-Nearest Neighbors Weighted (далее KNNW) использует сумму расстояний в окрестности [5]. Локальные методы, с другой стороны, обычно используют окрестность достижимости, которая включает все точки в ε-шара вокруг данного наблюдения. Таким образом, они используют плотность точек в окрестности для оценки наблюдений.

Например, Local Outlier Factors (далее — LOF) [16] использует обратную, нормализованную достижимость для оценки наблюдений, в то время как Simplified Local Outlier Factors (далее SLOF) [25] упрощает вычисления достижимости. Объединение методов, основанных на близости, таких как KNN, с использованием бутстреп-образцов может улучшить общие результаты [23].

1.3 Выводы

Согласно изученнои литературе, следующие выводы о применении существующих алгоритмов изолирующего леса можно сделать для решения задачи определения аномалий:

А. использования алгоритма ИЛ основывается на ключевом наблюдении, что деревья решений имеют тенденцию изолировать примеры-аутсайдеры на относительно ранних этапах дерева, благодаря чему достигается высокая скорость и относительная простота;

B. в настоящий момент нет прямой связи между производительностью ИЛ и его вариациями, и предположениями, которые затрагивают изначальное распределение данных;

C. все подходы на основе ИЛ аппроксимируют базовое распределение вероятностей и рассматривают среднее значение длины пути как приближение к смешанному весу.

Таким образом, необходимо выполнить сравнение подходов и таких модификаций ИЛ, как ОИЛ, РИЛ, ИЛСИ, для исследования собственной модификации алгоритма ИЛ, основанной на применении нового метода расчета значения длины пути.

ГЛАВА 2. РАЗРАБОТКА АЛГОРИТМА РЕШЕНИЯ ЗАДАЧИ ОБНАРУЖЕНИЯ АНОМАЛИЙ

В рамках магистерской диссертации предложена модификация классического алгоритма изолирующего леса для определения аномалий в транзакционных данных:

2.1 – приводится формальное обоснование методов, базирующихся на оценке плотности распределения;

2.3 – анализируются особенности работы уже существующих модификаций: ОИЛ, ИЛСИ, РИЛ.

2.2 – описывается базовый алгоритм ИЛ;

2.4 – описывается предложенная в данной работе модификация алгоритма ИЛ.

2.1 Математическое обоснование подходов, основанных на решении об оценке плотности распределения

Для дальнейшего описания работы ОИЛ приведено точное определение понятия выброса. Пусть при обнаружении выбросов за сами выбросы принимается смесь распределений, где по крайней мере одно распределение является “редким”. Более формально, предполагается, что $\mathcal{D}$ (см. параграф 1.1) является смесью K распределений, причем ни K, ни отдельные распределения не известны:

$$p_{\mathcal{D}}(x) = \sum_{i=1}^{K} w_i p_i(x_i) \tag{2.1}$$

– где $w = (w_1, \ldots, w_K)$ – вектор вероятностей категориального распределения. Для обнаружения выбросов предполагается, что по крайней мере одно распределение смеси имеет вероятность, близкую к нулю, то есть $w_i \approx 0$. Цель – охарактеризовать соответствующее распределение с малыми весами смеси и, следовательно, отличить его от остальных смесей. Для этого используется двухэтапная процедура [7].

2.1.1 Аппроксимация смешанного распределения

Первый этап заключается в аппроксимации $p_{\mathcal{D}}(x)$. Для этого необходимо найти функцию $f^* \in \mathcal{F}$ из некоторого множества функций $\mathcal{F}$, которая соответствует истинному распределению настолько близко, насколько это возможно:

$$
\begin{aligned}
f^* &= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} (f(x) - p_{\mathcal{D}}(x))^2 dx = \\
&= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} ((f(x))^2 - 2f(x)p_{\mathcal{D}}(x) + (p_{\mathcal{D}}(x))^2 dx = \\
&= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} ((f(x))^2 - 2f(x)p_{\mathcal{D}}(x) dx,
\end{aligned}
\tag{2.2}
$$

– где вторая строка обусловлена биномиальной формулой, а третья – отсутствием влияния $(p_{\mathcal{D}}(x))^2$ на минимизацию по f. Далее производится аппроксимация истинного распределения $p_{\mathcal{D}}(x)$ с помощью метода Монте-Карло, используя заданную выборку $\mathcal{S}$:

$$
f^* = \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} ((f(x))^2 dx - 2\sum_{i=1}^{N} f(x_i)\frac{1}{N}. \tag{2.3}
$$

Это приближение основывается на законе больших чисел, его точность зависит от величины N прямо пропорционально. Для $N \to \infty$ этот минимизатор является точным и сходится. Решение этой задачи без каких-либо предположений на $\mathcal{F}$ по-прежнему затруднительно, поскольку необходимо интегрировать по $\mathcal{X}$. Чтобы эффективно найти минимизатор этой функции, выдвинуто предположение, что $\mathcal{F}$ разбивает пространство $\mathcal{X}$ на L непересекающихся областей $\mathcal{R}_0, \ldots, \mathcal{R}_L$, где точки в каждой области следуют равномерному распределению. Формально:

$$
f(x) = \sum_{i=1}^{L} \mathbb{1}\{x \in \mathcal{R}_i\} \sum_{j=1}^{N} \frac{\mathbb{1}\{x_j \in \mathcal{R}_i\}}{N} = \sum_{i=1}^{L} \{x \in \mathcal{R}_i\} g_i. \tag{2.4}
$$

Обычным примером такого типа функции является гистограмма. Подстановка $f(x)$ в уравнение 2.3 и уравнение 2.4 приводит к следующему виду:

$$
\begin{aligned}
f^* &= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} ((f(x))^2 dx - 2\sum_{i=1}^{N} f(x_i)\frac{1}{N} = \\
&= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} \left(\sum_{j=1}^{L} \mathbb{1}\{x \in \mathcal{R}_i\} g_i\right)^2 dx - 2\sum_{i=1}^{N}\sum_{j=1}^{L} \mathbb{1}\{x_i \in \mathcal{R}_j\} g_i \frac{1}{N} = \\
&= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} (g_i)^2 V(\mathcal{R}_i) - 2\sum_{j=1}^{L} (g_j)^2 = \\
&= \arg\min_{f\in\mathcal{F}} \int_{\mathcal{X}} g_i^2 (V(\mathcal{R}_i) - 2),
\end{aligned}
\tag{2.5}
$$

– где $V(\mathcal{R}_i)$ обозначает объем i-ой области, а третья строка обусловлена тем, что все слагаемые, кроме одного, равны 0.

Далее рассмотрена эквивалентная задача максимизации:

$$f^* = \arg\max_{f \in \mathcal{F}} \sum_{i=1}^{L} (2 - V(\mathcal{R}_i)) g_i^2 \tag{2.6}$$

Для максимизации уравнения 2.6 нужно найти небольшие, плотные области, чтобы $V(\mathcal{R}_i)$ была мала, но g_i была большой. Количество областей L является частью модельной функции и как таковое может быть выбрано так, чтобы максимизировать уравнение 2.6.

Если возможна изоляция одной точки в области, то $V(\mathcal{R}_i) \to 0$ и $g_i \to \frac{1}{N}$. Отсюда следует, что любой алгоритм на основе деревьев, который полностью изолирует одиночные точки с помощью полностью построенных деревьев (где $L = N$), решает проблему 2.6 в некоторой степени, обеспечивая следующую нижнюю границу ограничения:

$$\sum_{i=1}^{L} (2 - V(\mathcal{R}_i)) g_i^2 = \sum_{i=1}^{N} \frac{2}{N^2} = \frac{2}{N}. \tag{2.7}$$

Другими словами, любой древовидный алгоритм, имеющий достаточно большое количество точных разбиений, гарантирует некоторое качество аппроксимации основного распределения вероятностей [7].

2.1.2 Поиск смешанных выбросов

Второй этап заключается в поиске выбросов распределений с учётом приближённого значения $p_{\mathcal{D}}(x)$. Согласно вышесказанному составлено предположение о том, что модель на основе древовидного алгоритма обладает достаточным качеством аппроксимации. Следовательно:

$$p_{\mathcal{D}}(x) = \sum_{i=1}^{K} w_i p_i(x) \approx \sum_{i=1}^{L} g_i \mathbb{1}\{x \in \mathcal{R}_i\}. \tag{2.8}$$

Для $L = K$ возможно рассмотреть $p_i(x) \approx \mathbb{1}\{x \in \mathcal{R}_i\}$ и $w_i \approx g_i$. По определению, распределения выбросов характеризуются очень малым весом смеси $w_i \approx 0$, поэтому цель состоит в том, чтобы найти малые g_i. Самым непосредственным образом возможно представить области эксперту, который мог бы изучить все

точки в области $\mathcal{R}_i$ и оценить вес смеси w_i с учетом ее экспертного мнения. В этом случае возможно напрямую определить отстающие области.

При отсутствии такого эксперта для $L = K$ возможно напрямую проверить веса смеси g_i и использовать их в качестве оценок выбросов, поскольку между ними существует соответствие один-к-одному. Интересно, что для $L > K$, обнаруживается аналогичная зависимость. Пусть $L = n - K$ при $n \in N$. Необходимо сопоставить L-листовых узлов дерева с количеством неизвестных смесей. Для этого вводятся искусственные смеси, которые используют ту же плотность p_i, но только часть веса исходной смеси w_i.

Пусть, без потери общности, смеси отсортированы так, что $w_1 \geqslant w_2 \geqslant \ldots \geqslant w_K$. Происходит создание копии каждой смеси n раз и изменение масштаба вероятности соответствующим образом:

$$\begin{aligned} p_{\mathcal{D}}(x) = \sum_{i=1}^{K} w_i p_i(x) = \sum_{j=1}^{n} \sum_{i=1}^{K} \frac{1}{n} w_i p_i(x) = \\ = \sum_{i=1}^{L} \frac{1}{n} w_i p_i(x) = \sum_{i=1}^{L} \frac{K}{L} \widetilde{w}_i p_i(x), \end{aligned} \tag{2.9}$$

– где вторая строка обусловлена $n = \frac{L}{K}$. Если L не кратно K, то происходит копирование смеси $\left[\frac{K}{L}\right]$ раз и повторное масштабирование оставшиеся $L \mod K$ смесей в соответствии с их сортировкой, начиная с самой большой. Это масштабирование сохраняет относительный порядок смесей $\widetilde{w}_1 \geqslant \widetilde{w}_2 \geqslant \ldots \geqslant \widetilde{w}_L$. Из этого следует возможность использования оцененных весов смесей g_i для оценки привлекательности областей, если $L \geqslant K$ [7].

2.1.3 Взаимосвязь изолирующего леса с другими древовидными алгоритмами машинного обучения

Любой алгоритм на основе дерева может быть использован для аппроксимации $p_{\mathcal{D}}(x)$ в той или иной степени. Следовательно, к ним относятся ИЛ, РИЛ и ИЛСИ. Рассмотрено правило подсчета баллов, используемое этими методами (ранее формула рассматривалась в 1.2):

$$score(x, N) = 2^{-\frac{E(h(x))}{c(N)}} \tag{2.10}$$

Пусть $\mathcal{R}(x)$ обозначает область, в которой находится наблюдение x и пусть $|\mathcal{R}(x)|$ обозначает количество обучающих данных, которые попадают в эту область.

Необходимо дать упорядочение по степени выдающихся результатов для каждого наблюдения, и поэтому возможно использование любого правила подсчета баллов, если оно сохраняет исходное упорядочивание выбросов. Предполагается, что области, которые подразумевают более длинный путь принятия решения, обычно содержат меньше примеров. Другими словами:

$$\frac{1}{|\mathcal{R}(x)|} \sim \mathbb{E}[h(x)] \tag{2.11}$$

Это предположение основано на том, что большая (средняя) длина пути означает, что выбор становится всё более и более избирательным, в количество примеров становится все меньше и меньше в каждом узле. Следуя этому предположению, легко показать, что правило подсчета баллов ИЛ является монотонной функцией веса смеси веса:

$$\begin{aligned} \frac{1}{|\mathcal{R}(x)|} &\sim \mathbb{E}[h(x)] \\ \frac{N}{|\mathcal{R}(x)|} &\sim \frac{\mathbb{E}[h(x)]}{c(N)} \\ \log_2\left(\frac{N}{|\mathcal{R}(x)|}\right) &\sim \log_2\left(\frac{\mathbb{E}[h(x)]}{c(N)}\right) \\ \log_2\left(\frac{|\mathcal{R}(x)|}{N}\right) &\sim -\log_2\left(\frac{\mathbb{E}[h(x)]}{c(N)}\right), \end{aligned} \tag{2.12}$$

– где вторая строка справедлива, так как N и $c(N)$ – неотрицательные константы. Третья строка справедлива благодаря тому, что $\frac{N}{|\mathcal{R}(X)|} < 1$ и, следовательно, $\log_2 \frac{N}{|\mathcal{R}(X)|} < 0$. Заметим, что $\log_2()$ является монотонной функцией, и поэтому она не изменяет первоначальное упорядочивание своих аргументов. Правило подсчета баллов ИЛ и РИЛ игнорирует $\log_2$ в правой части уравнения, что приводит к:

$$\log_2\left(\frac{|\mathcal{R}(x)|}{N}\right) \Uparrow -\frac{\mathbb{E}[h(x)]}{c(N)}, \tag{2.13}$$

– где $\Uparrow$ обозначает тот факт, что если левая часть увеличивается, то и увеличивается и правая часть, и наоборот [7].

Из этого следует, что ИЛ, РИЛ и ИЛСИ сохраняют исходное упорядочение весов смеси используя среднюю длину пути в качестве аппроксимации, если более длинные пути принятия решений в дереве подразумевают меньшее количество точек в областях.

Это предположение имеет решающее значение для хорошей работы алгоритма и в некоторой степени оправдано, как уже упоминалось выше.

2.2 Алгоритм изолирующего леса

Изоляция подразумевает отделение экземпляра от остальных экземпляров. Поскольку аномалии немногочисленны и различны, они более восприимчивы к изоляции. В индуцированном данными случайном дереве разбиение на разделы экземпляров повторяется рекурсивно, пока все экземпляры не будут изолированы.

Рассмотрим пример случайного разбиения, когда у аномалий образуются более короткие пути. В первом случае (см. рис.2.1) меньшее количество экземпляров аномалий приводит к меньшему количеству разделов, иными словами, более коротким путям в древовидной структуре, а в другом случае (см. рис.2.2) — экземпляры с различающимися атрибутами-значениями вероятнее будут разделены на ранних этапах. Следовательно, когда лес случайных деревьев дает более короткие длины путей для некоторых точек, то они с высокой долей вероятности являются аномалиями.

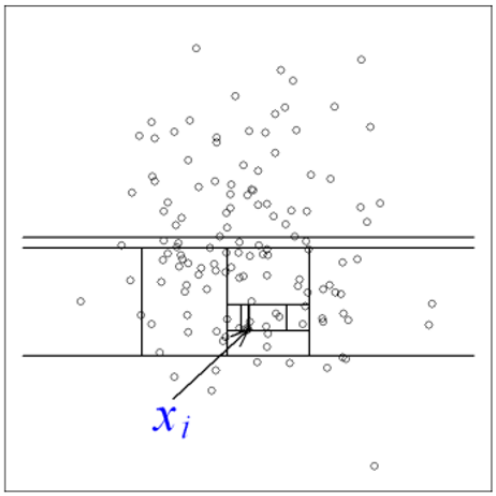


Рис.2.1. Пример изолирования x_i

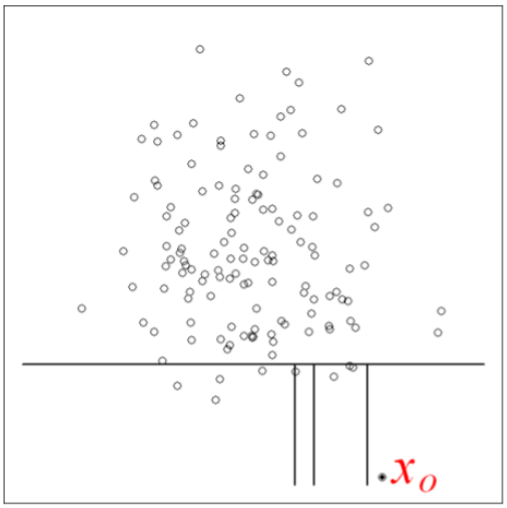


Рис.2.2. Пример изолирования x_0

Демонстрация идеи о том, что аномалии более восприимчивы к изоляции при случайном разбиении, проиллюстрирована на вышеприведённых примерах, где приведена визуализации случайного разбиения нормальной точки по сравнению с аномалией. Нормальная точка x_i обычно требует больше разделов для выделения. Обратное также верно для аномальной точки x_0 которая, как правило, требует меньше разделов для изоляции.

В этом примере для раздела выбирается случайный атрибут, затем случайны образом выбирается граничное значения между максимальным и минимальным значениями ранее выбранного атрибута. Так как рекурсивное разбиение может

быть представлено в виде древовидной структуры, то количество необходимых для изоляции точки разделов эквивалентно длине пути от корневого узла до конечного узла.

В текущем примере (см. рис.2.1-рис.2.2) длина пути x_i больше, чем длина пути x_0. Поскольку каждый раздел генерируется случайным образом, отдельные деревья генерируются с разными наборами разделов. Усреднение длины пути по нескольким деревьям позволяет найти ожидаемую длину пути. На рис.2.3 показано, что средняя длина пути x_0 и x_i сходятся, когда количество деревьев увеличивается. При использовании 1000 деревьев средняя длина пути x_0 и x_i сходятся к 4,02 и 12,82 соответственно. Это показывает, что у аномалий длина пути короче, чем у нормальных экземпляров.

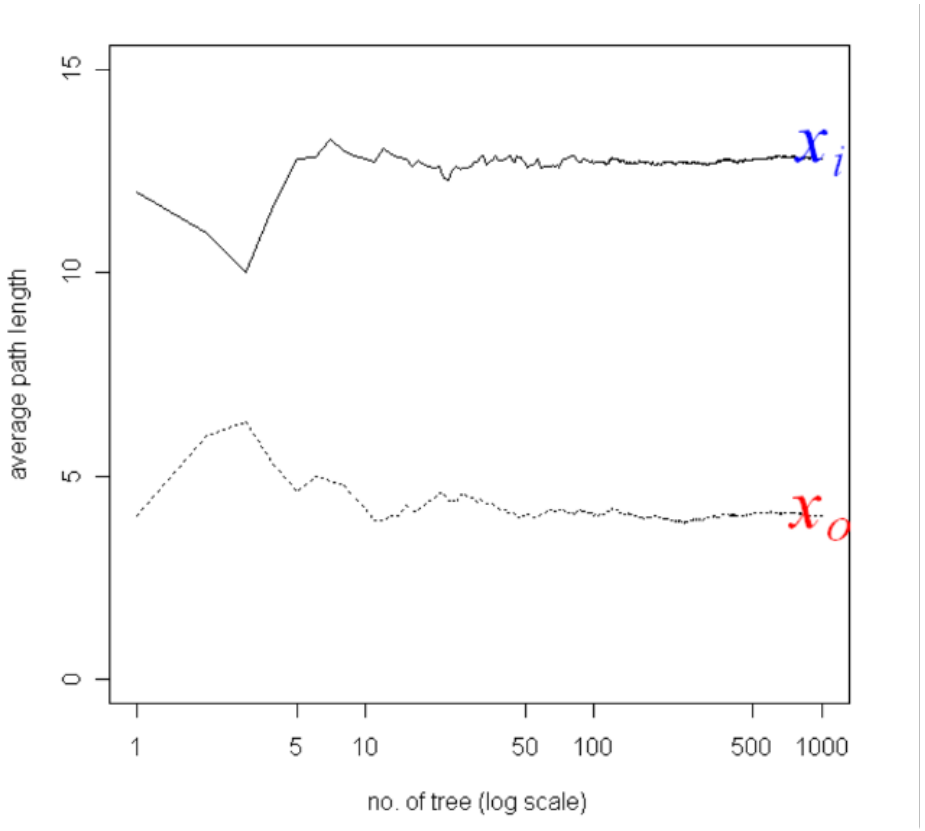


Рис.2.3. Сходимость средней длины пути

Пусть T – узел дерева изоляции. T является либо внешним узлом без дочерних узлов (терминальным узлом), либо внутренним узлом с одним родителем и ровно двумя дочерними узлами (T_l, T_r). Тест состоит из атрибута q и значения разбиения p таких, что тест $q < p$ разделяет точки данных на T_l и T_r.

Пусть дана выборка данных $X = \{x_1, \ldots, x_N\}$ из N экземпляров с $\mathcal{D}$-вариативным распределением. Для построения изолирующего дерева необхо-

димо рекурсивно разделить X, выбирая случайным образом атрибут q и значение разбиения p, пока либо:

– дерево достигнет предела высоты;
– $|X| = 1$;
– все данные в X имеют одинаковые значения.

Изолирующее дерево (далее – ИД) – это правильное двоичное дерево, где каждый узел имеет ровно ноль или два дочерних узла. Если предположить, что все экземпляры различны, то каждый экземпляр изолирован от внешнего узла, если ИД полностью построено. В этом случае число внешних узлов равно N, а число внутренних узлов равно $N - 1$; общее число узлов ИД равно $2N - 1$. Таким образом, потребность в памяти ограничена и растет линейно с увеличением N.

Задача обнаружения аномалий заключается в предоставлении рейтинга, который отражает степень аномальности. Таким образом, одним из способов обнаружения аномалий является сортировка точек данных в соответствии с их длиной пути или оценкой аномалии. Аномалии – это точки, которые занимают верхние строчки списка [14].

Длина пути $h(x)$ для точки x измеряется количеством ребер x, проходящих через ИД от корневого узла до тех пор, пока обход не завершится на внешнем узле. Оценка аномалий необходима для любого метода обнаружения аномалий. Сложность получения такой оценки из $h(x)$ заключается в том, что в то время, как максимально возможная высота ИД растет порядка N, средняя высота растет порядка $\log N$. Нормализация $h(x)$ по любому из приведенных выше условий либо не ограничена, либо не может быть непосредственно определена.

Поскольку ИД имеют эквивалентную структуру с Binary Search Tree (далее BST) (см. табл.2.1), оценка среднего $h(x)$ для внешних окончаний узлов такая же, как и для неудачного поиска в BST.

Таблица 2.1

Сравнение ИД с BST

	ИД	BST
Основной элемент	Правильные бинарные деревья	Правильные бинарные деревья
Условие остановки рекурсии для поддерева	Удаление внешних узлов	Неудачный поиск
Условие остановки рекурсии построения	Не применимо	Удачный поиск

Анализ оценки средней длины пути ИД аналогичен анализу из BST. С учётом набора данных из N экземпляров средняя длина пути неудачного поиска в BST вычисляется по формуле:

$$c(N) = 2H(N-1) - (2(N-1)/N), \tag{2.14}$$

– где $H(i)$ – гармоническое число, которое может быть оценено по $\ln(i) + 0{,}5772156649$ (постоянная Эйлера). Поскольку $c(N)$ – среднее значение $h(x)$ для N, мы используем его для нормализации $h(x)$. Рейтинг аномалии s для экземпляра x определяется как (ранее формула рассматривалась в 1.2):

$$score(x, N) = 2^{-\frac{E(h(x))}{c(N)}} \tag{2.15}$$

– где $E[h(x)]$ – среднее значение $h(x)$ из набора деревьев изоляции. Условия использования рейтинга аномалии 1.2:

- когда $E[h(x)] \to c(N)$, то $s \to 0{,}5$;
- когда $E[h(x)] \to 0$, то $s \to 1$;
- когда $E[h(x)] \to N-1$, то $s \to 0$;

Рейтинг аномалии s монотонно зависит от $h(x)$. На рис.2.4 показана взаимосвязь между $E[h(x)]$ и s, а также следующие условия, применяемые в случае, когда $0 < s \leqslant 1$ для $0 < h(x) \leqslant N-1$.

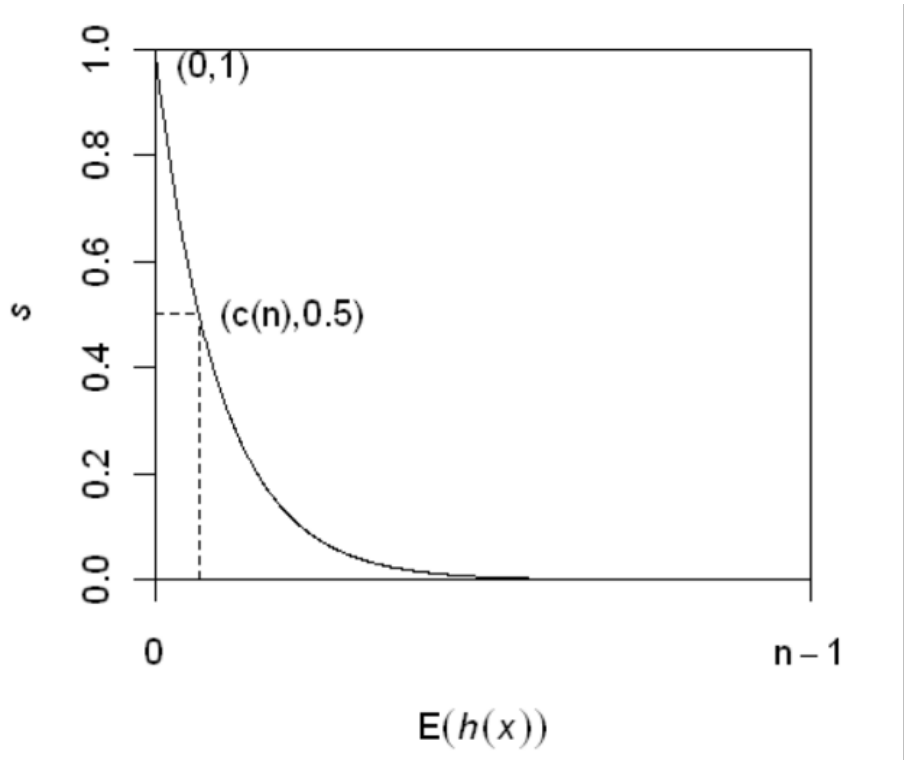


Рис.2.4. Взаимосвязь ожидаемой длины $E[h(x)]$ и оценки аномалии s

На рис.2.4 $c(N)$ – средняя длина пути, как определено в (2.14). Если ожидаемая длина пути $E[h(x)]$ равна средней длине пути $c(N)$, то $s = 0{,}5$, независимо от значения N. Вышеприведённое описание рейтинга аномалии является основой следующих оценок:

- если рейтинг аномалии близок к 1, то это определенно аномалии;
- если рейтинг аномалии у экземпляров s намного меньше 0,5, то их можно считать нормальными экземплярами;
- если все экземпляры имеют $s \approx 0{,}5$, то вся выборка не имеет ни одной явной аномалии [14].

Границы оценки аномалии могут быть получены путем пропускания образца выборки через набор ИД, что облегчает детальный анализ результатов обнаружения. На рис.2.5 показан пример такой границы, которая позволяет пользователю визуализировать и идентифицировать аномалии в пространстве экземпляров. На примере возможно четко определить три точки, где $s \geqslant 0{,}6$, которые являются потенциальными аномалиями. Контурные линии для $s = 0.5, 0.6, 0.7$ также показаны на рис.2.5.

Как ансамбль деревьев, использующий изолирующие деревья, ИЛ

- идентифицирует аномалии как точки с более коротким путем;

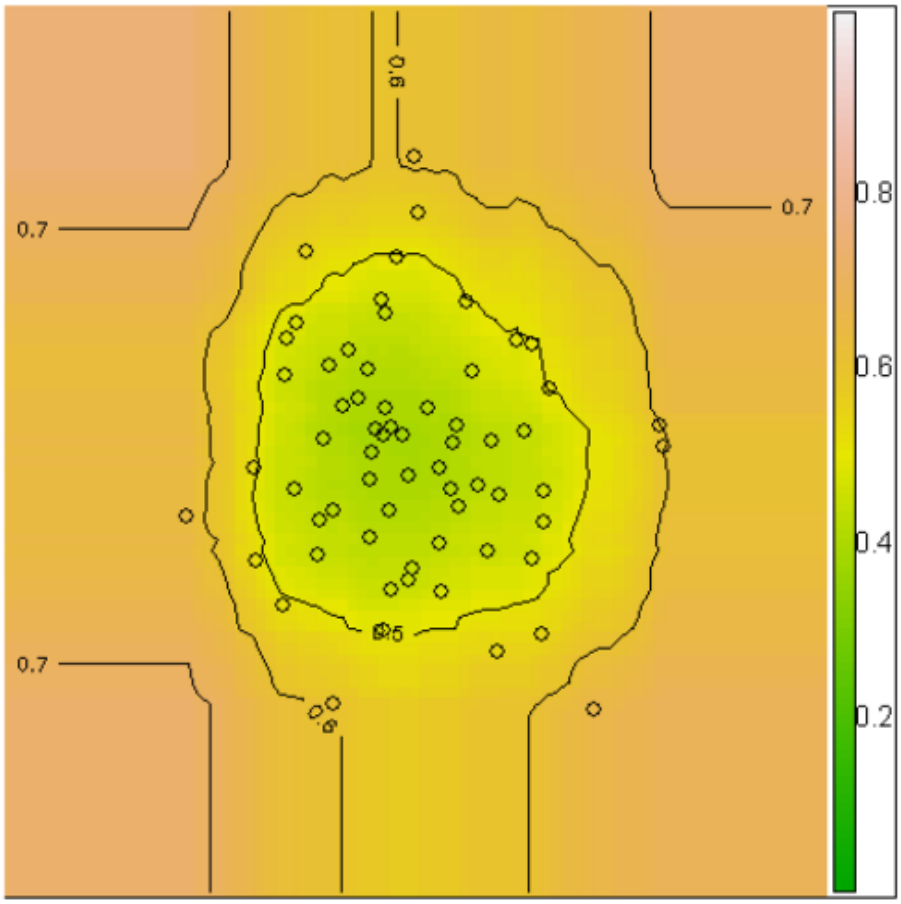


Рис.2.5. Границы оценки аномалий ИЛ для гауссовского распределения из шестидесяти четырех точек

– имеет несколько деревьев, выступающих в качестве “экспертов”, для выявления различных аномалий.

Поскольку ИЛ не нужно выделять все нормальные экземпляры – большую часть обучающей выборки, то ИЛ может определить аномалии как более короткие пути выборки. ИЛ может хорошо работать с частичной моделью без выделения всех нормальных точек и строить модели, используя небольшой размер выборки. В отличие от существующих методов, где большой размер выборки является более желательным, метод изоляции успешно работает и при небольшом размере. Более того, большой размер выборки снижает способность ИЛ изолировать аномалии, так как нормальные экземпляры могут мешают процессу изоляции. Таким образом, составление выборки меньшего размера (подвыборка) из выборки большего обеспечивает благоприятные условия для хорошей работы ИЛ.

В контексте решения задачи обнаружения аномалий существуют такие проблемы, как заболачивание и маскировка. Заболачивание относится к ошибочному определению нормальных случаев в качестве аномалии.

Маскировка – это существование слишком большого количества аномалий. скрывающих их собственное присутствие. Когда кластер аномалий большой и плотный, то происходит увеличение количества разделов для изоляции каждой аномалии. В таких условиях оценки, которые используются деревьями, имеют большую длину пути, что делает аномалии трудно обнаруживаемыми.

Заболачивание и маскировка являются результатом слишком большого количества данных для задачи обнаружения аномалий. Уникальная характеристика ИД позволяет ИЛ строить частичную модель путем подвыборки, что минимизирует эффекты заболачивания и маскировки. Это происходит потому, что:

- подвыборка контролирует размер данных, что помогает ИЛ лучше изолировать примеры аномалий;
- каждое ИД может быть специализировано, поскольку каждая подвыборка включает различный набор аномалий или даже отсутствие аномалий.

На рис.2.6 показан набор данных, созданный Малкроссом [24]. Набор данных имеет два кластера аномалий, расположенных рядом с одним большим кластером нормальных точек в центре. Есть мешающие нормальные точки, окружающие аномальные кластеры, и аномальные кластеры более плотные, чем кластера нормальных точек в этой выборке из 4096 экземпляров.

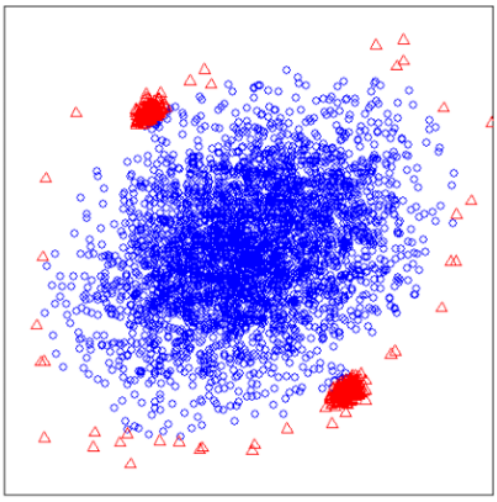


Рис.2.6. Набор данных Малкросса в 4096 экземпляров, прошедший процедуру обнаружения аномалий с помощью ИД

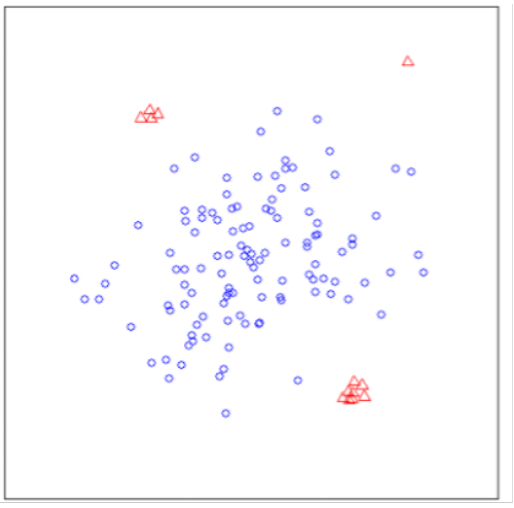


Рис.2.7. Набор данных Малкросса в 128 экземпляров, прошедший процедуру обнаружения аномалий с помощью ИД

На рис.2.7 показана подвыборка из 128 экземпляров исходных данных. Кластеры аномалий четко идентифицируются в подвыборке. Те кластеры нормальных экземпляров, которые окружали два аномальных кластера, сильно потеряли в

численности, как кластеры аномалий, что облегчает их идентифицированию. При использовании всей выборки ИЛ показывает Area Under Curve (далее — AUC), равным 0,67. При использовании подвыборки размером 128 ИЛ достигает AUC в 0,91. Результат показывает ИЛ превосходную способность обнаруживать аномалии, справляясь с эффектами заболачивания и маскировки с помощью значительно уменьшенной подвыборки.

Обнаружение аномалий с помощью ИЛ – это двухэтапный процесс. На первом этапе (обучение) строятся ИД с использованием подвыборок обучающего набора. На втором этапе (тестирование) пропускает тестовые экземпляры через деревья изоляции для получения оценки аномалии для каждого экземпляра. На этапе обучения ИД строятся путем рекурсивного разбиения заданного обучающего множества до тех пор, пока не будут выделены отдельные экземпляры или достигнута определенная высота дерева. Предел высоты дерева l автоматически задается размером подвыборки ψ : $l = ceiling(\log_2 \psi)$, что приблизительно соответствует средней высоте дерева. Смысл выращивания деревьев до средней высоты заключается в том, что необходимо анализировать только те точки данных, которые имеют длину пути меньше средней, так как эти точки с большей вероятностью являются аномалиями. Подробности этапа обучения можно найти в алгоритмах 4 и 5 (см. рис.2.8 и рис.2.9).

Algorithm 4 : $iForest(X, t, \psi)$

Inputs: X - input data, t - number of trees, ψ - subsampling size
Output: a set of t *iTrees*

Initialize $Forest$
set height limit $l = ceiling(\log_2 \psi)$
for $i = 1$ to t **do**
    $X' \leftarrow sample(X, \psi)$
    $Forest \leftarrow Forest \cup iTree(X', 0, l)$
end for
return $Forest$

Рис.2.8. Общий алгоритм ИЛ

У алгоритма ИЛ есть два входных параметра. Это размер подвыборки ψ и количество деревьев t. Размер подвыборки ψ управляет размером обучающих данных. Когда ψ увеличивается до желаемого значения, ИЛ обнаруживает аномалии с высокой эффективностью, и нет необходимости увеличивать ψ дальше,

Algorithm 5 : $iTree(X, e, l)$

Inputs: X - input data, e - current tree height, l - height limit

Output: an iTree

if $e \geq l$ or $|X| \leq 1$ **then**
  return *exNode*$\{Size \leftarrow |X|\}$
else
  let Q be a list of attributes in X
  randomly select an attribute $q \in Q$
  randomly select a split point p from max and min values of attribute q in X
  $X_l \leftarrow filter(X, q < p)$
  $X_r \leftarrow filter(X, q \geq p)$
  return *inNode*$\{Left \leftarrow iTree(X_l, e+1, l),$
    $Right \leftarrow iTree(X_r, e+1, l),$
    $SplitAtt \leftarrow q,$
    $SplitValue \leftarrow p\}$
end if

Рис.2.9. Алгоритм построения ИД

потому что это увеличивает время обработки и объем памяти без увеличения производительности обнаружения.

При прочих равных используется $\psi = 256$ в качестве значение по умолчанию для нашего эксперимента. Анализ влияния размера подвыборки показал, что производительность обнаружения близка к оптимальной при этом значении по умолчанию и нечувствительна к широкому диапазону ψ. Число деревьев t контролирует размер ансамбля. Длины путей обычно сходятся задолго до $t = 100$. Если не указано иное, то рекомендуется использовать $t = 100$ как значение по умолчанию. Сложность обучения ИЛ составляет $O(t\psi \log \psi)$.

На этапе тестирования рейтинг аномалии s определяется из ожидаемой длины пути $E[h(x)]$ для каждого тестового экземпляра. $E[h(x)]$ определяются путем прохождения экземпляров через каждое ИД в ИЛ. С помощью функции PathLength длина одного пути $h(x)$ определяется путем подсчета количества ребер от корневого узла до конечного узла при прохождении экземпляра x через ИД. Когда x завершается на внешнем узле, где $Size > 1$, возвращаемое значение равно e плюс поправка $c(Size)$. Корректировка учитывает не построенное поддерево

за пределом высоты дерева. Когда $h(x)$ получено для каждого дерева ансамбля, оценка аномалии производится путем вычисления $s(x, \psi)$ по уравнению 1.1.

Сложность процесса оценки составляет $O(Nt \log \psi)$, где N – это размер тестовых данных. Детали функции PathLength можно найти в алгоритме 6 (см. рис.2.10). Чтобы найти m лучших аномалий, функция просто сортирует данные с помощью s в порядке убывания. Первые m первых экземпляров являются верхними m аномалиями [14].

Algorithm 6 : $PathLength(x, T, e)$

Inputs : x - an instance, T - an iTree, e - current path length; to be initialized to zero when first called
Output: path length of x

1: **if** T is an external node **then**
2: return $e + c(T.size)$
3: **end if**
4: $a \leftarrow T.splitAtt$
5: **if** $x_a < T.splitValue$ **then**
6: return $PathLength(x, T.left, e + 1)$
7: **else** $\{x_a \geq T.splitValue\}$
8: return $PathLength(x, T.right, e + 1)$
9: **end if**

Рис.2.10. Алгоритм функции PathLength

2.3 Модификации алгоритма изолирующего леса

В вышеперечисленных разделах представлена теоретическая основа для обнаружения выбросов с помощью изолирующих деревьев и подробно описано действие таких алгоритмов, как ИЛ, РИЛ и ИЛСИ. Существует два общих предложения об этих трех алгоритмах:

– они косвенно максимизируют уравнение 2.6 для приведения ансамбля;
– они оценивают коэффициенты смеси, используя среднее значение пути.

2.3.1 Обобщенный изолирующий лес

Ниже описан метод обобщенного изолирующего леса изоляции (ОИЛ), который использует эти утверждения, принимая во внимание уравнение 2.6 и непосредственно используя коэффициенты смеси. ОИЛ представляет собой ансамбль

обобщенных деревьев изоляции. Он разбивает пространство наблюдений $\mathcal{X}$ на все более мелкие области и использует независимые оценки вероятности для каждой области.

Другими словами, вводится представление дерева как ориентированного графа с корневым узлом, где каждый узел имеет до K дочерних узлов. Каждый узел в дереве принадлежит подобласти $\mathcal{R} \subseteq \mathcal{X}$, а все дочерние узлы каждого узла рекурсивно разбивают область своего родительского узла на K непересекающихся меньших областей. Корневой узел принадлежит ко всему пространству наблюдения $\mathcal{R}_0 = \mathcal{X}$. Каждый узел использует до K функций разбиения $\mathcal{S} = \{s_\mathcal{R} : \mathcal{X} \to \{0, 1\}\}$, где $s_\mathcal{R} = 1$ означает, что x принадлежит соответствующему области $\mathcal{R}$, а $s_\mathcal{R} = 0$ означает, что нет.

Во время построения разбиения необходимо обеспечить, чтобы разбиения разбивали пространство наблюдений на непересекающиеся области, так, чтобы ровно одно разбиение было “1”, а остальные – “0”. Чаще всего находятся бинарные деревья решений, которые разделяют пространство на два подпространства в каждом узле, называемые “левым” и “правым” разбиением. После того, как пространство наблюдений достаточно разбито, используется функция плотности $g \in \mathcal{G} = \{g : \mathcal{X} \to [0, 1]\}$ для оценки плотности. Как обсуждалось ранее, возможно использовать частоту $g_i(x) = \frac{1}{N} \sum_{x_j \in \mathcal{S}} \mathbb{1}\{x_j \in \mathcal{R}_i\} = \frac{|\mathcal{S}_i|}{N}$, где $\mathcal{S}_i$ – часть обучающей выборки, принадлежащая области $\mathcal{R}_i$.

Для обучения ансамбля используется жадный алгоритм, похожий на классические деревья решения. Предполагается, что ансамбль уже обучен с n узлами и необходимо разделить область $\mathcal{R}_i$ с помощью другой разделительной гипотезы.

Пусть $\mathcal{S}_i$ – это та часть обучающих данных, которая попадает в область $\mathcal{R}_i$, а затем случайным образом выбирается K точек из $\mathcal{S}$ так, чтобы каждая точка вызывала подобласть. Более формальное определение функции разбиения можно представить как:

$$s_i(x) \quad = \quad \begin{cases} 1 & \text{if } i = \arg\max\{k(x, x_j) | j = 1, \ldots, K\} \\ 0 & \text{otherwise} \end{cases} \tag{2.16}$$

– где $x_j \in \mathcal{S}_i$ – выбранные представители, а $k : \mathcal{R}_i \times \mathcal{R}_i \to [0, 1]$ – функция ядра.

После разделения пространства наблюдений на достаточное количество областей, возможно прекращение индукции дерева. Необходимо максимизировать следующую функцию:

$$\arg\max \sum_{i=1}^{L} (2 - V(\mathcal{R}_i)) g_i^2, \tag{2.17}$$

что возможно при $g_i \to 1$ и $V(\mathcal{R}_i) \to 0$.

Другими словами, производится поиск небольших плотных областей, содержащих много точек. Однако, выбор становится более избирательными, чем больше узлов добавляется к дереву, поэтому чем меньше становится g_i, тем становится также меньше $V(\mathcal{R}_i)$.

Таким образом, предполагается использовать порог τ, и всякий раз, когда $\tau \geqslant (2 - V(\mathcal{R}_i))g^2$, следует прекратить индукцию дерева. Вычисление $V(\mathcal{R}_i)$ может быть сложным для данных большой размерности и областей неправильной формы. Чтобы решить эту проблему, предполагается использовать среднее внутреннее расстояние ядра в каждой области [7].

Учитывая представителей каждой области, вычисляется среднее сходство ядра всех точек в этой области с соответствующим представителем. Это имеет тот же смысл, что и раньше, потому что происходит остановка индукции дерева, как только находятся маленькие, плотные области.

Algorithm 1 **Индукция ОИЛ**

```
1: function FIT(S, node)
2:     g = estimateDensity(S)
3:     j ← argmax{node.s[j](x) | j = 1, ..., K}
4:     if |S|^{-1} Σ_{∀x∈S} K(x_j, x) ≤ τ then
5:         node.children = null
6:         node.leaf = true
7:     else
8:         node.s = samplePoints(S_i, K)
9:         for j = 1, ..., K do
10:            S_j = {x ∈ S | s_j(x) = 1}
11:        for j = 1, ..., K do
12:            node.children[j] = fit(S_i)
```

Рис.2.11. Индукция обобщенного изолирующего дерева

Алгоритм 2.11 кратко описывает обучение дерева решения, где samplePoints$(\mathcal{S}_i, K)$ случайным образом выбирает до K точек из $\mathcal{S}$, если они доступны. Если нет, то выбираются все точки.

Algorithm 2 **Применение ОИЛ**

```
1: function DENSITY(node, x)
2:     while !node.leaf do
3:         j ← argmax{node.s[k](x) | k = 1, ..., K}
4:         node ← node.children[j]
5:     return node.g
```

Рис.2.12. Применение обобщенного изолирующего дерева

Алгоритм 2.12 показывает применение ансамбля после обучения. Случайно построенные деревья имеют большие вариации в своих оценки плотности, поэтому результаты могут сильно различаться между отдельными деревьями [7].

Для борьбы с таким поведением происходит объединение нескольких ансамблей в алгоритмы типа бэггинг, подобный алгоритмам ИЛ и РИЛ. Выбор различных подмножеств данных признаков осуществляется за счет пакетной обработки для того, чтобы внести разнообразие в ансамбль. Возможны вариации функции сходства k.

Algorithm 3 **Обнаружение выброса ОИЛ**

```
Require: Dataset 𝒟, subset size ψ, number of trees t
1: function GIF(𝒟, ψ, t)
2:     for all i ∈ [1, t] do
3:         𝒯_i ← FIT(samplePoints(𝒟, ψ))
4:     ρ ← new array of size |𝒟|
5:     for all x_i ∈ 𝒟 do
6:         for all 𝒯 ∈ {𝒯_1, ..., 𝒯_t} do
7:             node ← 𝒯
8:             while !node.leaf do
9:                 k ← argmax{node.s[k](x_i) | k = 1, ..., K}
10:                node ← node.children[k]
11:            ρ_i ← ρ_i + t^{-1} node.g
       return ρ
```

Рис.2.13. Обнаружение аномалий с помощью ОИЛ

Алгоритм 2.13 суммирует итоговое решение.

2.3.2 Изолирующий лес сейсмической активности

Как упоминалось ранее в главе 1.2, ИЛСИ использует дисперсию выборки, чтобы вычислит оценки каждого разбиения.

Пусть $\mathcal{S} = \mathcal{S}_l \cup \mathcal{S}_r$ – набор данных, разделённый на два дизъюнктивных множества $\mathcal{S}_l$ и $\mathcal{S}_r$. Тогда предлагается использовать такое разбиение, которое максимизирует

$$d_{gain}(\mathcal{S}) = \frac{\sigma(\mathcal{S}) - 0.5 \cdot (\sigma(\mathcal{S}_l) + \sigma(\mathcal{S}_r))}{\sigma(\mathcal{S})}, \tag{2.18}$$

– где $\sigma(\cdot)$ – дисперсия.

2.3.3 Расширенный изолирующий лес

Изолированный лес полагается на случайность при выборе объектов и значений. Поскольку аномальных точек “мало и они разные”, они быстро выделяются на фоне этих случайных выборок. Но, как представлено ранее, срезы ветвей всегда либо горизонтальные, либо вертикальные, и это вносит свой вклад в оценки аномалий. В алгоритме нет фундаментальной причины, требующей этого ограничения, и поэтому в каждой точке ветвления возможно выбрать разрез ветви, который имеет случайный “наклон”.

Для N-мерного набора данных выбор случайного наклона для среза ветви аналогичен выбору вектора нормали, $\vec{n}$, равномерно распределенного по единичной N-сфере. Это можно выполнить, вычислив случайное число для каждой координаты $\vec{n}$ из стандартного нормального распределения $\mathcal{N}(0,1)$ [12]. Это приводит к равномерному выбору точек на N-сфере. Для определенеия точки отсечения, $\vec{p}$, выбирается значение из равномерного распределения, которые присутствуют в каждой точке ветвления. Когда значения $\vec{n}$ и $\vec{p}$ определены, критерии ветвления для разделения данных для данной точки $\vec{x}$ выглядят следующим образом:

$$(\vec{x} - \vec{p} \cdot \vec{n} \geqslant 0) \tag{2.19}$$

Если условие выполнено, элемент $\vec{x}$ принадлежит левой ветви, в противном случае – правой [11].

2.4 Весовой изолирующий лес

При описании классического алгоритма ИЛ для оценки наблюдений использовалась средняя длина пути по ансамблю деревьев $E[h(x)]$ (ранее формула рассматривалась в 1.2):

$$score(x, N) = 2^{-\frac{E(h(x))}{c(N)}}, \tag{2.20}$$

– где

$$E(h(x)) = \frac{1}{n}\sum_{i=1}^{t} h_i(x). \tag{2.21}$$

Предлагается рассмотреть $E[h(x)]$ в качестве некоторого неизвестного распределения, математическое ожидание которого является искомой величиной для оценки наблюдения.

$$E_w(h(x)) = \sum_{i=1}^{t} w_i h_i(x), \quad \sum_{i=1}^{t} w_i = 1. \tag{2.22}$$

Некоторая функция потерь по весам w_i стремиться к минимуму. Необходимо найти веса w_i, для этого определим и минимизируем функцию потерь:

$$L(w) \to \min_w \tag{2.23}$$

Считаем, что аномалия известна, пусть $y_i = 0$ – аномалия, $y_i = 1$ – нормальный экземпляр.

$$L(w) = \sum_{j=1}^{n} (y_j - \widetilde{y_j})^2 \to \min_w \tag{2.24}$$

$$\widetilde{y_j} = \max(0, \lambda - s(x, n)) \tag{2.25}$$

– где λ – глобальный-параметр значения метрики для измерения близости распределений, $s(x, n)$ – величина оценки аномальности, рассчитанная на момент t построенных деревьев. Необходимо найти $\widetilde{y_j}$. Для этого выразим λ:

$$-(\log_2 s(x, n)) \cdot c(n) = E(h(x)) \tag{2.26}$$

$$-(\log_2 \lambda) \cdot c(n) = \gamma \tag{2.27}$$

При возрастании $s(x, n)$, $E(h(x))$ убывает [14], при возрастании λ, γ тоже убывает, так как $\log_2$ монотонно возрастающая функция, а $-\log_2$ монотонно убывающая.

Функция потерь $L(w)$ примет вид:

$$L(w) = \sum_{j=1}^{n} \left(y_j - \max \left(0, \sum_{i=1}^{t} w_i h_i(x_j) - \gamma \right) \right)^2 \to \min_{w} \tag{2.28}$$

при ограничении $\sum_{i=1}^{t} w_i = 1$, i – номер дерева, j – номер примера из обучающей выборки.

Обозначим

$$z_j = \max \left(0, \sum_{i=1}^{t} w_i h_i(x_j) - \gamma \right) \tag{2.29}$$

Тогда задача сводится к задаче квадратичной оптимизации

$$L(w, z_j) = \sum_{j=1}^{n} (y_j - z_j)^2 \to \min_{w,z} \tag{2.30}$$

при ограничениях:

$$\begin{cases} \sum_{i=1}^{t} w_i = 1, \\ z_j \geqslant 0, \\ z_j \geqslant \sum_{i=1}^{t} w_i h_i(x_j) - \gamma, \\ j = 1, \ldots, n. \end{cases}$$

Так как в (2.30) y_j – постоянная, то

$$p_j = y_j - z_j \;\mid\; L(w, p) = \sum_{j=1}^{n} p_j^2 \to \min_{w,p} \tag{2.31}$$

при ограничениях:

$$\begin{cases} \sum_{i=1}^{t} w_i = 1, \\ y_j - p_j \geqslant 0, \\ y_j - p_j \geqslant \sum_{i=1}^{t} w_i h_i(x_j) - \gamma, \\ j = 1, \ldots, n. \end{cases}$$

2.5 Выводы

Таким образом, модификация алгоритма ИЛ – ВИЛ, – основана на рассмотрении $E[h(x)]$ в качестве некоторого неизвестного распределения, математическое ожидание которого является искомой величиной для оценки наблюдения. Для решения задачи необходимо определить веса w_i, для чего выполняется минимизация функции потерь 2.23. Минимизация функции сводится к решению задачи квадратичной оптимизации 2.31.

ГЛАВА 3. РАЗРАБОТКА МОДЕЛИ МОДИФИЦИРОВАННОГО ИЗОЛИРУЮЩЕГО ЛЕСА НА ЯЗЫКЕ C++

В рамках магистерской работы разработана весовая модификация алгоритма изолирующего леса для определения аномалий:

3.1 — приводится описание разработанного пространства имен для реализации моделей ИЛ;

3.2 — выполняются тесты реализации на предмет состоятельности;

3.3 — приводятся перечень используемых программных средств для визуализации данных.

3.1 Пространство имен

Вышеописанная модификация предполагает изменения алгоритма ИЛ на архитектурном уровне, следовательно все модели ИЛ и этапы эксперимента реализованы на языке C++. Для реализации модификаций ИЛ создано пространство имён, поля и методы которого описывают модель ИЛ без модификаций. Далее для каждой модификации реализованы расширения пространства имён с перегрузкой изменяемых полей и методов.

Модели ИЛ, реализованные для сравнительного тестирования и последующего анализа эффективности:

- модель базового ИЛ без модификаций;
- модель ОИЛ, реализованная согласно описанию из 2;
- модель ИЛСИ, реализованная согласно документации авторов;
- модель РИЛ, реализованная лишь в качестве надстройке параметров ИЛ, т. к. модификация не предполагает других каких-либо изменений;
- модель ВИЛ, разработка и теоретическое обоснование которой также приведено выше.

В пространстве имён реализованы следующие классы:

- класс особенностей экземпляра выборки Feature;
- класс выборки Sample;
- класс узла ИД Node;
- класс псевдогенерации случайных величин Randomizer;
- базовый класс ИЛ Forest.

Пространство имён также дополнено классом со статическими методами извлечения данных, препроцессинга и tSNE. Описание пространства имён приведено в табл.3.1.

Таблица 3.1

Пространство имён ИЛ

Метод	Описание
Node::Node()	Конструктор узла ИД
Node::Node(const std::string& featureName, uint64_t splitValue)	Конструктор узла ИД с указанием особенности и величины параметра разделения ИЛ
void Node::SetLeftSubTree(Node* subtree)	Конструктор левого поддерева ИД
void Node::SetRightSubTree(Node* subtree)	Конструктор правого поддерева ИД
Forest::Forest()	Конструктор ИЛ
void Forest::SetRandomizer(Randomizer* newRandomizer)	Метод инициализации генератора псевдослучайных величин
void Forest::AddSample(const Sample& sample)	Метод инициализации выборки и сохранения признаков
NodePtr Forest::CreateTree(const FeatureNameToValuesMap& featureValues, size_t depth)	Рекурсивный метод создания ИД с отслеживанием глубины рекурсии
void Forest::Create()	Метод создания ИЛ с заданным количеством ИД
double Forest::Score(const Sample& sample, const NodePtr tree)	Метод вычисления длины пути для элементов выборки в заданном ИД
double Forest::Score(const Sample& sample)	Метод вычисления средней длины пути для элементов выборки во всём ИЛ
double Forest::NormalizedScore(const Sample& sample)	Метод нормализации средней длины пути
void Forest::Destroy()	Метод удаления всего ИЛ
void Forest::DestroyRandomizer()	Метод удаления генератора псевдослучайных величин

Вышеописанные методы являются основой для реализации модификаций ИЛ. Другими словами, модификации реализованы либо в виде класса в данном пространстве имён с дополнительными полями и методами, либо в виде перегрузки методов с расширенным набором аргументов. Также в пространстве имён заданы базовые классы признака, выборки, узла, ГСПЧ и ИЛ.

3.2 Проверка реализации

Проверка реализации ИЛ осуществлена в виде двух тестов. В обоих генерируется выборка со случайным количеством аномальных экземпляров. Каждый элемент выборки имеет 2 числовых признака. Отличие тестов в величинах параметров – во втором тесте выборка и количество ИД в 10 раз больше. Пример выполнения тестов показан на рис.3.1.

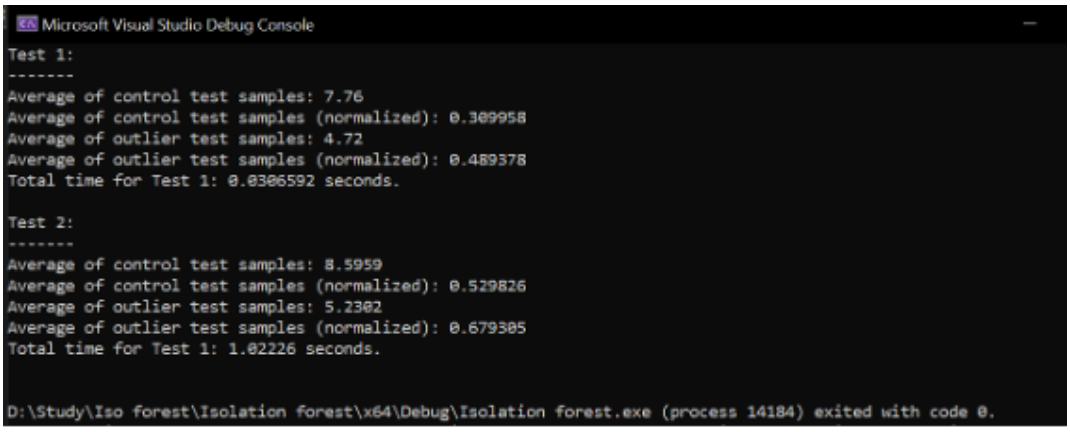


Рис.3.1. Результаты тестов реализации ИЛ

Средняя длина пути аномального экземпляра меньше средней длины контрольного экземпляра, как в обычном виде, так и нормализованном. Следовательно, алгоритм ИЛ реализован корректно. Альтернативный вариант проверки – парсинг файла вывода размеченной выборки, фрагмент которого приведён на рис.3.2. Строка файла содержит метку элемента и значения числовых признаков.

Тестирование эффективности реализованных моделей обнаружения аномалий на основе алгоритма ИЛ и его модификаций проводилось на реальных транзакционных данных. Их препроцессинг и визуализация описаны далее в 4. В текущее решение добавлены методы и средства обработки данных. Важно отметить, что до этого момента никакие сторонние библиотеки кроме как тех, что входят в стандартный набор компилятора C++ 20, не использовались.

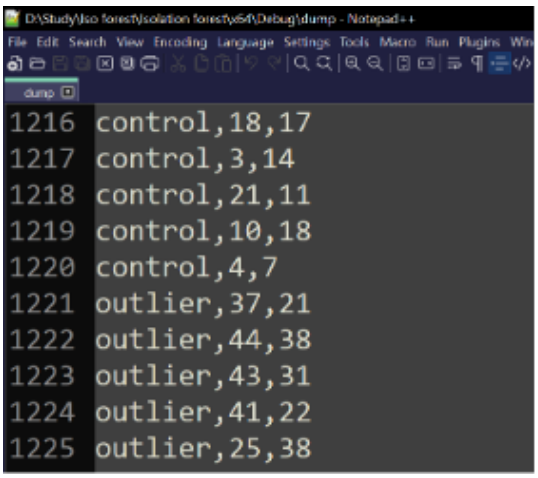


Рис.3.2. Фрагмент файла размеченной выборки

3.3 Визуализация

Все примеры визуализации данных реализованы с помощью библиотеки matplotlib-cpp, представляющей собой API обёртку вокруг средств matplotlib и Matlab [22]. Обработка данных реализована средствами библиотеки Cylon [21]. Cylon – это быстрая, масштабируемая библиотека параллельной обработки данных с распределенной памятью для обработки структурированных данных. Cylon реализует набор реляционных операторов для обработки данных. Хотя ядро Cylon реализовано с использованием системного уровня C/C++, предоставляются несколько языковых интерфейсов (Python и Java) для легкой интеграции с существующими приложениями, позволяя инженерам по обработке данных и AI/ML вызывать операторы обработки данных на знакомом языке программирования. По умолчанию он работает с MPI для распределения приложений. Внутри Cylon использует Apache Arrow для представления данных в формате столбцов.

Метод tSNE, как и все вышеперечисленные программные решения на C++, интегрирован в реализацию моделей ИЛ с использованием его нативной реализации [26].

3.4 Выводы

Таким образом, реализация модификации алгоритма ИЛ выполнялась на языке C++ 20 без использования сторонних библиотек. Были реализованы следующие модификации алгоритмов ИЛ:

– модель базового ИЛ без модификаций;
– модель ОИЛ;

- модель ИЛСИ;
- модель РИЛ;
- модель ВИЛ, разработка и теоретическое обоснование которой также приведено в главе 2.

Была выполнена проверка состоятельности реализации алгоритма на основе двух тестов.

ГЛАВА 4. ТЕСТ РАЗРАБОТАННОЙ МОДЕЛИ

Данная глава посвящена исследованию эффективности предложенных алгоритмов для обнаружения аномалий в транзакционных данных:

4.1 — приводится описание тестовых данных, объем, характеристика.

4.2 — описывается модель эксперимента, приводятся результаты работы алгоритмов, предложенных в магистерской работе;

4.3 — приводится анализ результатов экспериментов.

4.1 Описание тестовых данных

Для тестирования работы алгоритмов, реализованных в главе 3 были использованы деперсонализированные транзакционные данные антифродовой системы некоторого Банка. Суммарно датасет содержал 16 млн транзакций, что эквивалентно объему данных, спроцессированному за 5 месяцев. Каждая транзакция представляется набором атрибутов, значения некоторых не определены.

В табл.4.1 приведено краткое описание основных из них.

Статистика по признакам 10000 случайных операций представлена на рис.4.1, 4.2.

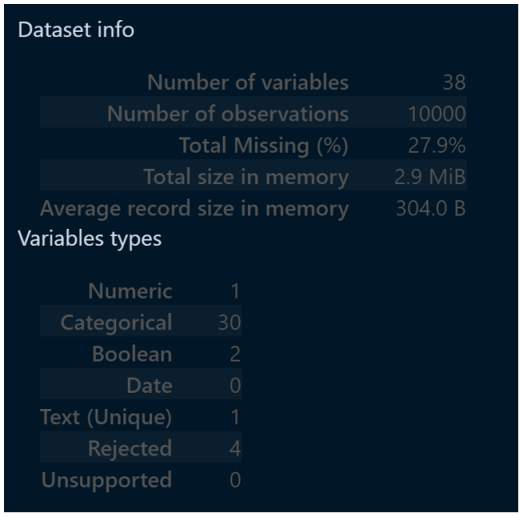


Рис.4.1. Статистика набора транзакционных данных

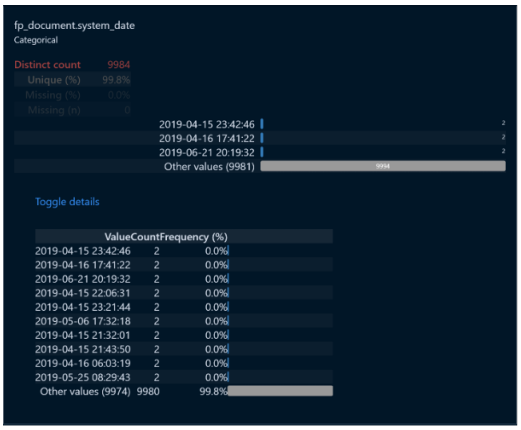


Рис.4.2. Пример характеристики для признака даты проведения транзакции

Таблица 4.1

Описание основных признаков набора данных

Атрибут	Бизнес-смысл	Значение
fp_document. doc_direction	Направление транзакции	O/R
fp_document. orig_ref_number	RRN операции	123456789012
BIN_md5	Hash первых 6 цифр номера карты, BIN (card prefix) банка-эмитента карты	4524352321524
txn/isOffline	Свойство операции, была ли проведена транзакция Off-Line	T/F
txn/conditions/ cardholderPresenceType	Свойство операции, присутствовал ли держатель карты при исполнении операции	T/F
txn/conditions/ cardPresenceType	Свойство операции, присутствовала ли карта при исполнении операции	T/F
txn/conditions/ terminalType	Тип терминала	POS
txn/conditions/ eCommerceIndicator	Свойство операции, признак электронной коммерции	T/F
fp_document.risk_points	Количество рисковых баллов, начисленных в соответствии с настроенными Банком антифрод правилами	200
txn/settlementAmount	Сумма операции в валюте расчётов	150.00
txn/conditions/pinEntered	Свойство операции, был ли введен ПИН	T/F
txn/settlementCurrency	Код валюты операции	643

После препроцессинга данных произведена визуализация t-SNE (см. рис.4.3). Размер выборки составил 2500 элементов.

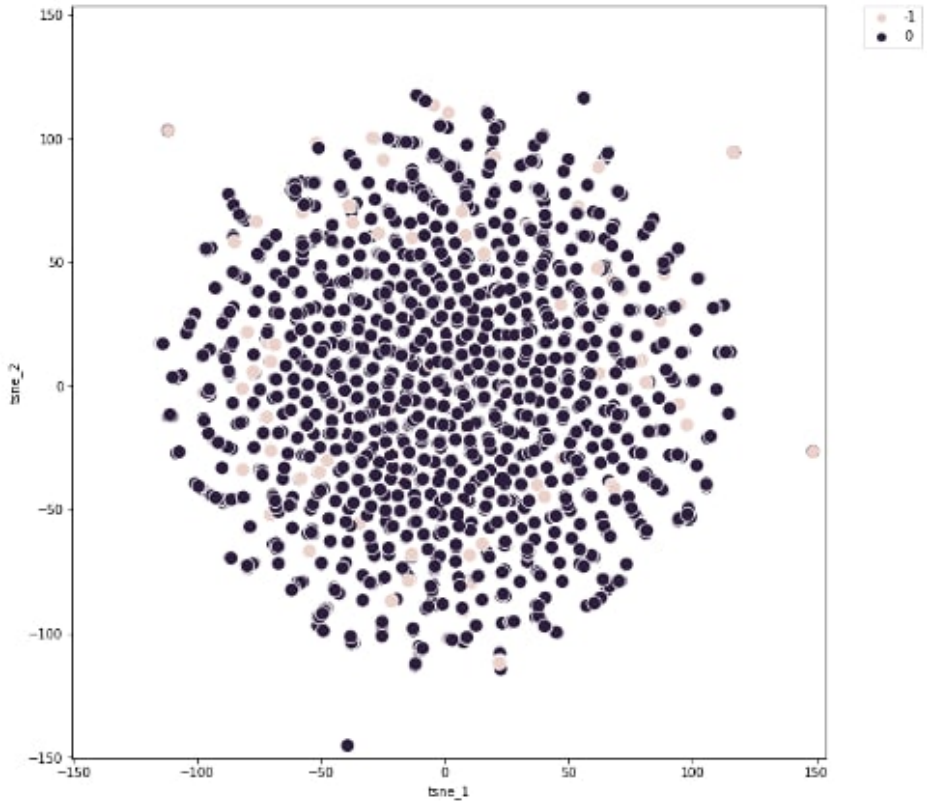


Рис.4.3. Визуализация набора данных методом t-SNE

4.2 Результаты экспериментов

Тест заключался в кросс валидации по параметру размера подвыборки от 2500 до 10000 случайно выбранных записей с шагом в 2500. Количество ИД также изменяется от 250 до 1000 с шагом в 250. Кросс валидация проводилась для каждой модели ИЛ. Всего получилось $5 \cdot 4 \cdot 4 = 80$ экспериментов. Для последующего анализа результатов тестирования отобраны следующие признаки моделей:

- время, затраченное на рекурсивное построение ИД;
- график кривой ROC AUC;
- сложность реализации;
- зависимость от размера подвыборки;
- зависимость от количества ИД.

Далее приведены примеры экспериментов. Результаты всех экспериментов представлены в приложении 1.

4.2.1 Результат работы алгоритма изолирующего леса без модификаций

На рисунках 4.4-4.7 представлены результаты работы алгоритма базового ИЛ без модификаций.

IF-2500 elements-250 trees

Accuracy = 0.733

Precision = 0.733

Recall = 0.503

F1-score = 0.597

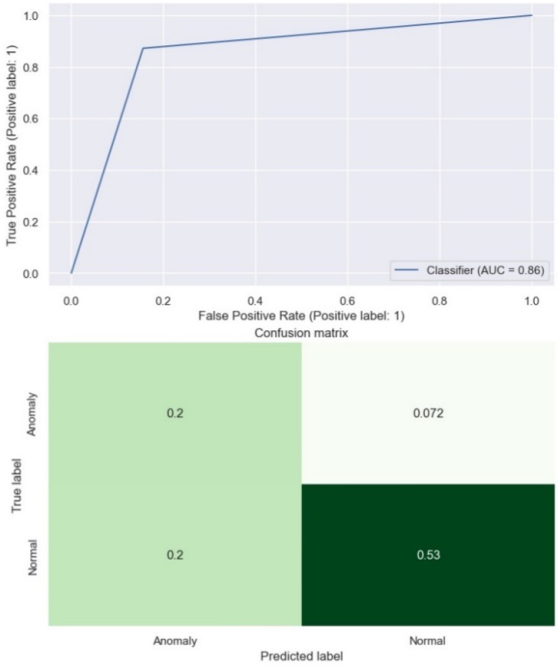


Рис.4.4. Результаты работы алгоритма ИЛ, размер подвыборки 2500, ИД – 250

IF-5000 elements-500 trees

Accuracy = 0.72

Precision = 0.72

Recall = 0.222

F1-score = 0.34

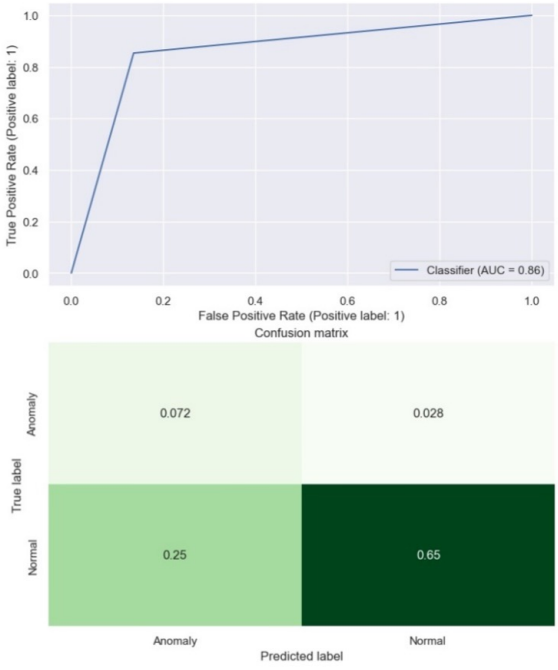


Рис.4.5. Результаты работы алгоритма ИЛ, размер подвыборки 5000, ИД – 500

IF-7500 elements-750 trees

Accuracy = 0.725

Precision = 0.725

Recall = 0.481

F1-score = 0.578

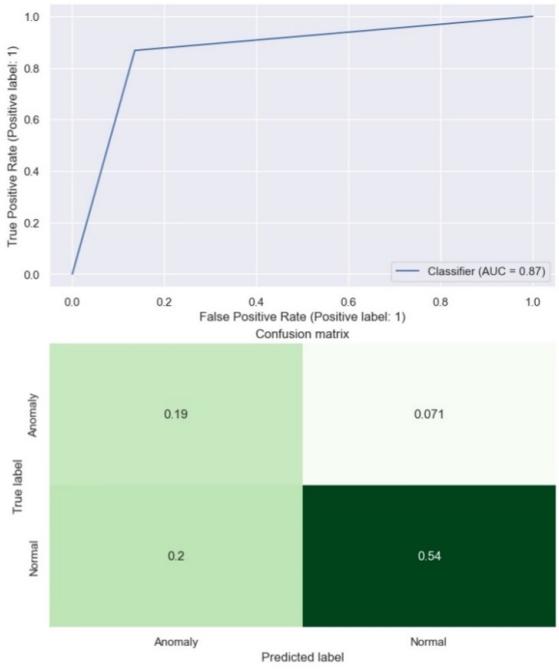


Рис.4.6. Результаты работы алгоритма ИЛ, размер подвыборки 7500, ИД – 750

IF-10000 elements-1000 trees

Accuracy = 0.71

Precision = 0.71

Recall = 0.38

F1-score = 0.495

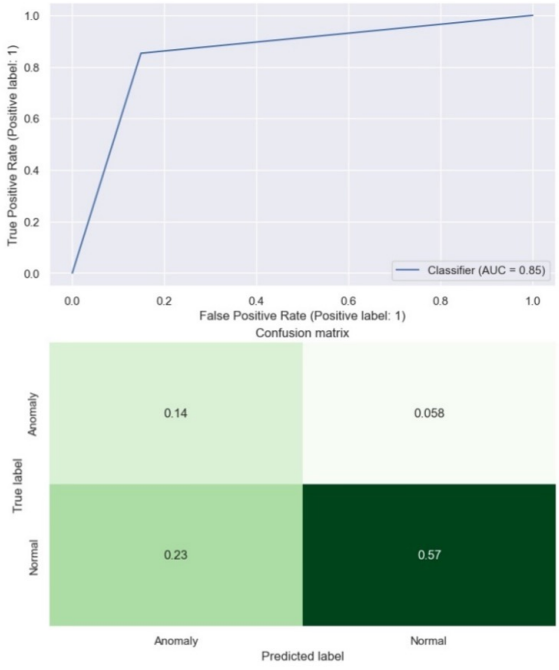


Рис.4.7. Результаты работы алгоритма ИЛ, размер подвыборки 10000, ИД – 1000

4.2.2 Результат работы алгоритма расширенного изолирующего леса

На рисунках 4.8-4.11 представлены результаты работы алгоритма РИЛ.

EIF-2500 elements-250 trees

Accuracy = 0.775

Precision = 0.775

Recall = 0.56

F1-score = 0.65

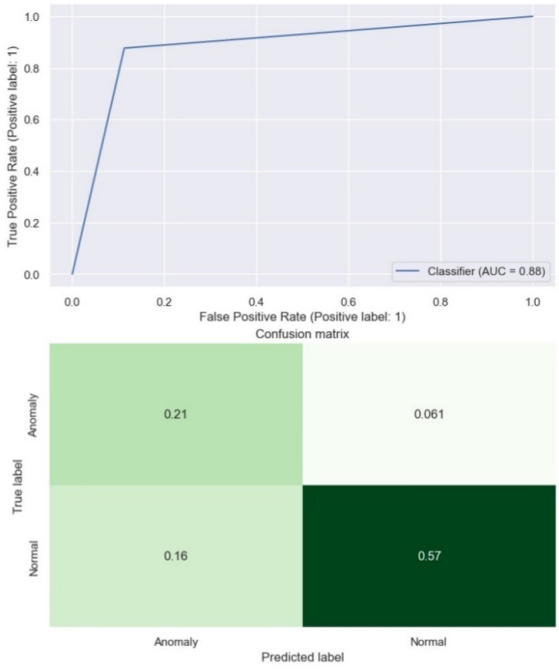


Рис.4.8. Результаты работы алгоритма РИЛ, размер подвыборки 2500, ИД – 250

EIF-5000 elements-500 trees

Accuracy = 0.795

Precision = 0.795

Recall = 0.601

F1-score = 0.685

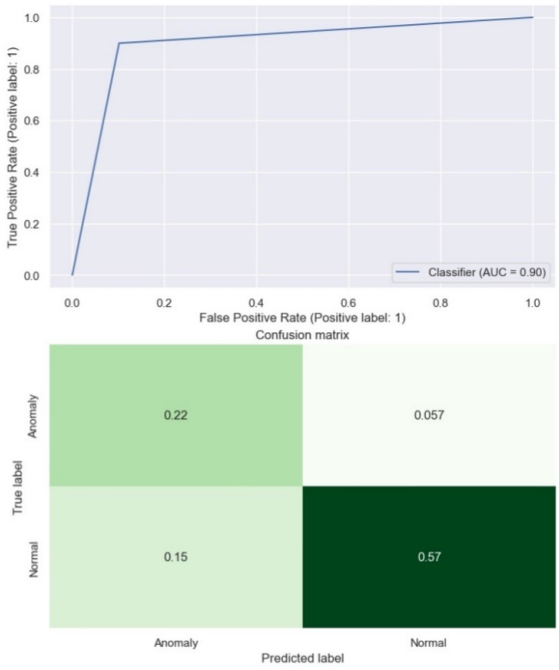


Рис.4.9. Результаты работы алгоритма РИЛ, размер подвыборки 5000, ИД – 500

EIF-7500 elements-750 trees

Accuracy = 0.765

Precision = 0.765

Recall = 0.626

F1-score = 0.689

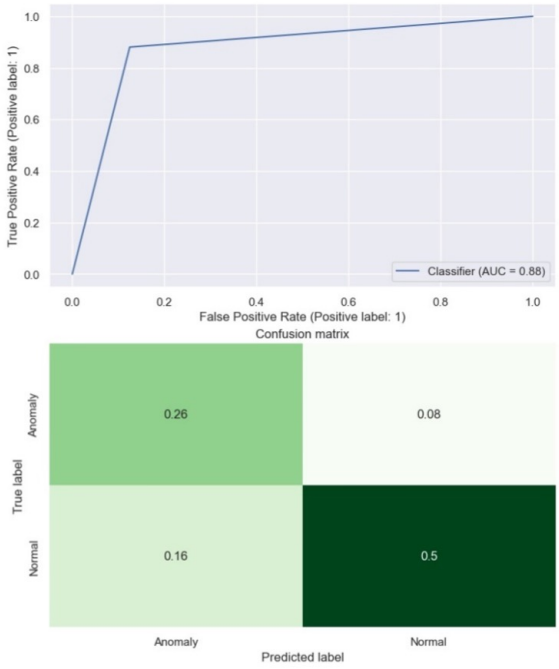


Рис.4.10. Результаты работы алгоритма РИЛ, размер подвыборки 7500, ИД – 750

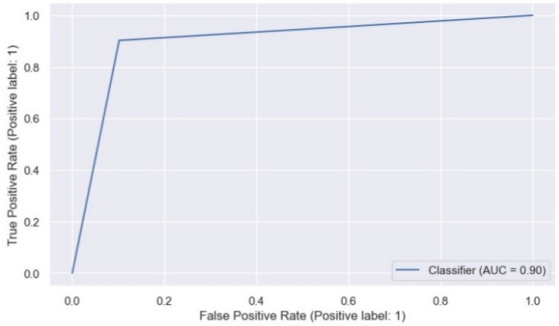
EIF-10000 elements-1000 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.484

F1-score = 0.603



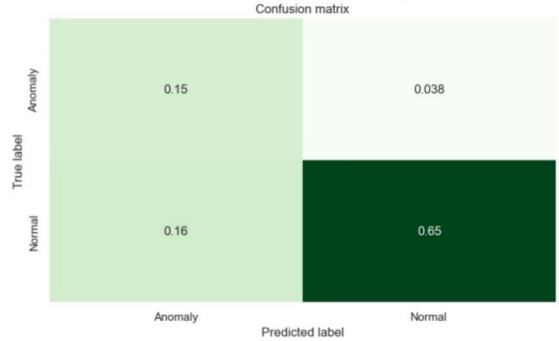


Рис.4.11. Результаты работы алгоритма РИЛ, размер подвыборки 10000, ИД – 1000

4.2.3 Результат работы алгоритма изолирующего леса сейсмической активности

На рисунках 4.12-4.15 представлены результаты работы алгоритма ИЛСИ.

SCiForest-2500 elements-250 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.484

F1-score = 0.603

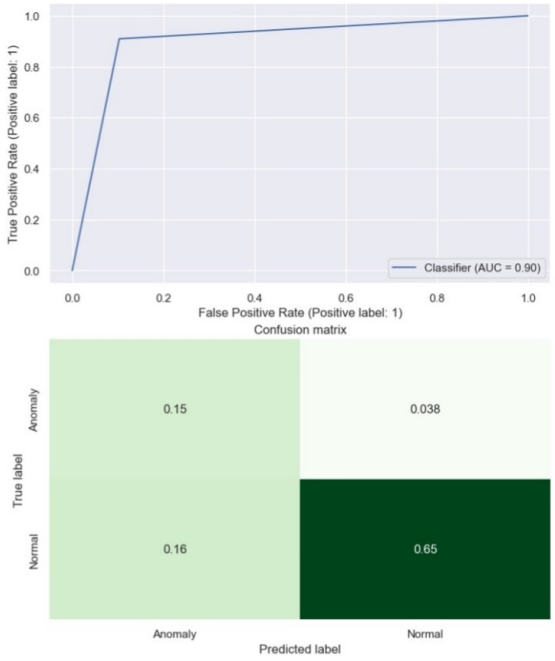


Рис.4.12. Результаты работы алгоритма ИЛСИ, размер подвыборки 2500, ИД – 250

SCiForest-5000 elements-500 trees

Accuracy = 0.81

Precision = 0.81

Recall = 0.531

F1-score = 0.642

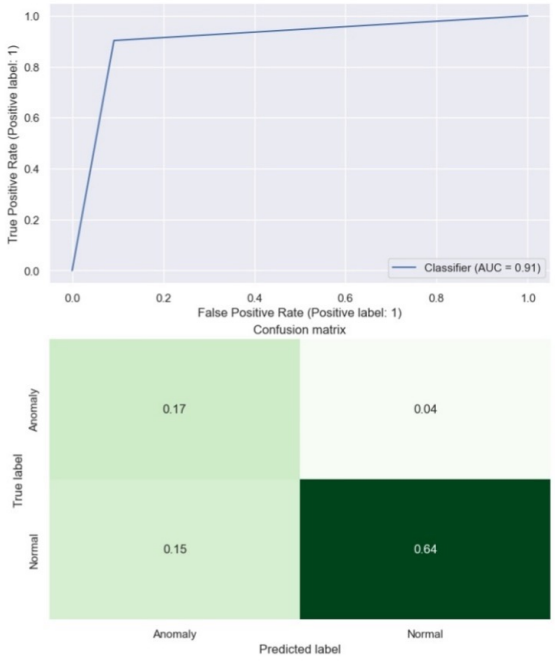


Рис.4.13. Результаты работы алгоритма ИЛСИ, размер подвыборки 5000, ИД – 500

SCiForest-7500 elements-750 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.414

F1-score = 0.545

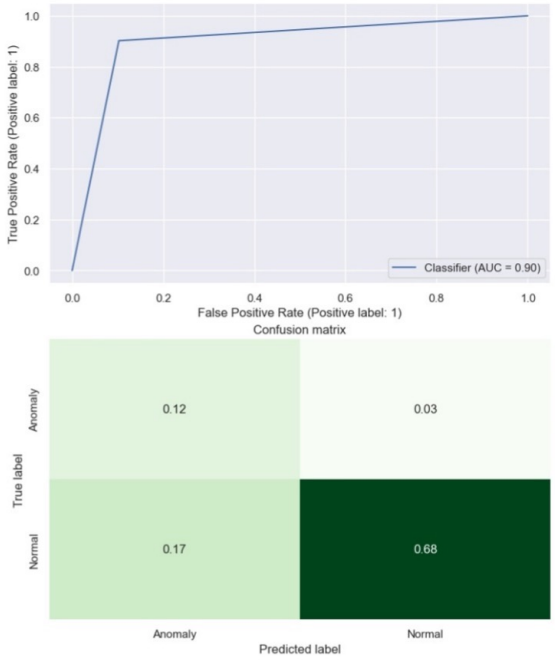


Рис.4.14. Результаты работы алгоритма ИЛСИ, размер подвыборки 7500, ИД – 750

SCiForest-10000 elements-1000 trees

Accuracy = 0.82

Precision = 0.82

Recall = 0.36

F1-score = 0.501

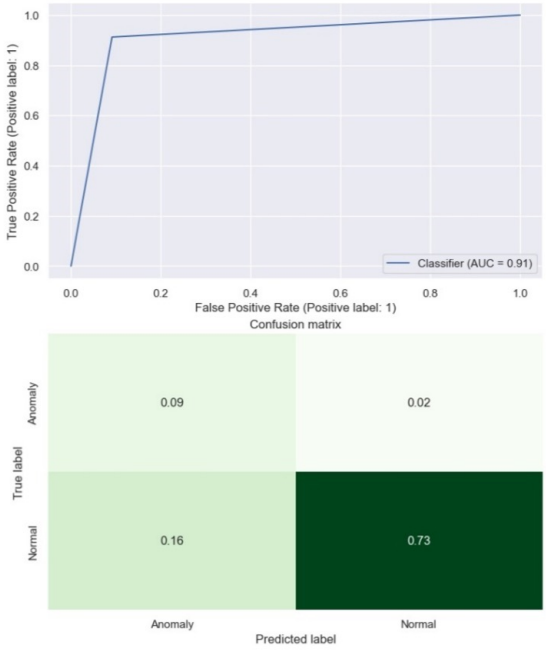


Рис.4.15. Результаты работы алгоритма ИЛСИ, размер подвыборки 10000, ИД – 1000

4.2.4 Результат работы алгоритма обобщенного изолирующего леса

На рисунках 4.16-4.19 представлены результаты работы алгоритма ОИЛ.

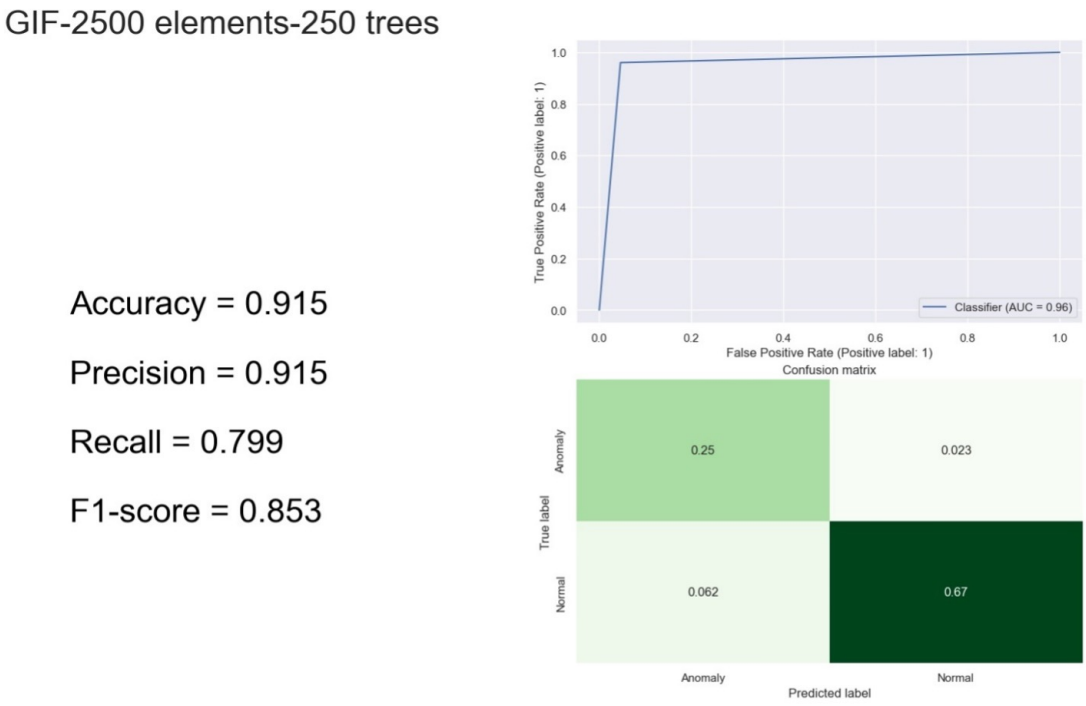


Рис.4.16. Результаты работы алгоритма ОИЛ, размер подвыборки 2500, ИД – 250

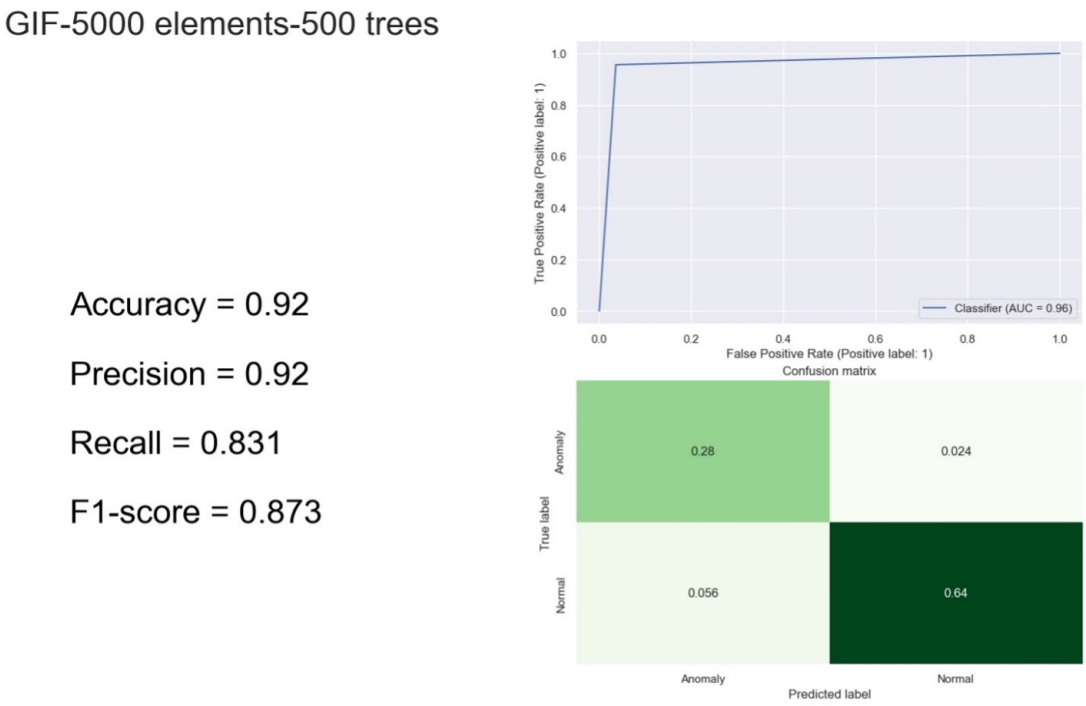


Рис.4.17. Результаты работы алгоритма ОИЛ, размер подвыборки 5000, ИД – 500

GIF-7500 elements-750 trees

Accuracy = 0.948

Precision = 0.948

Recall = 0.809

F1-score = 0.873

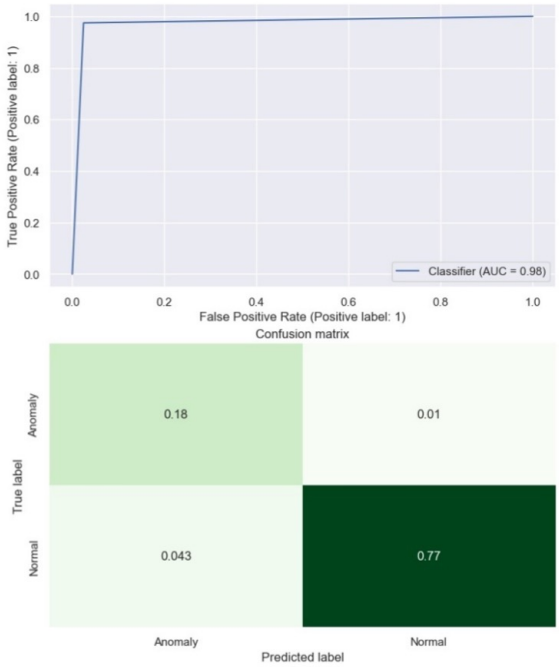


Рис.4.18. Результаты работы алгоритма ОИЛ, размер подвыборки 7500, ИД – 750

GIF-10000 elements-1000 trees

Accuracy = 0.99

Precision = 0.99

Recall = 0.973

F1-score = 0.982

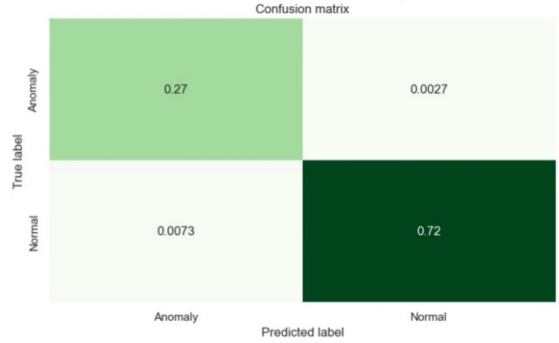


Рис.4.19. Результаты работы алгоритма ОИЛ, размер подвыборки 10000, ИД – 1000

4.2.5 Результат работы алгоритма весового изолирующего леса

На рисунках 4.20-4.23 представлены результаты работы алгоритма ВИЛ, разработанного в рамках данной работы.

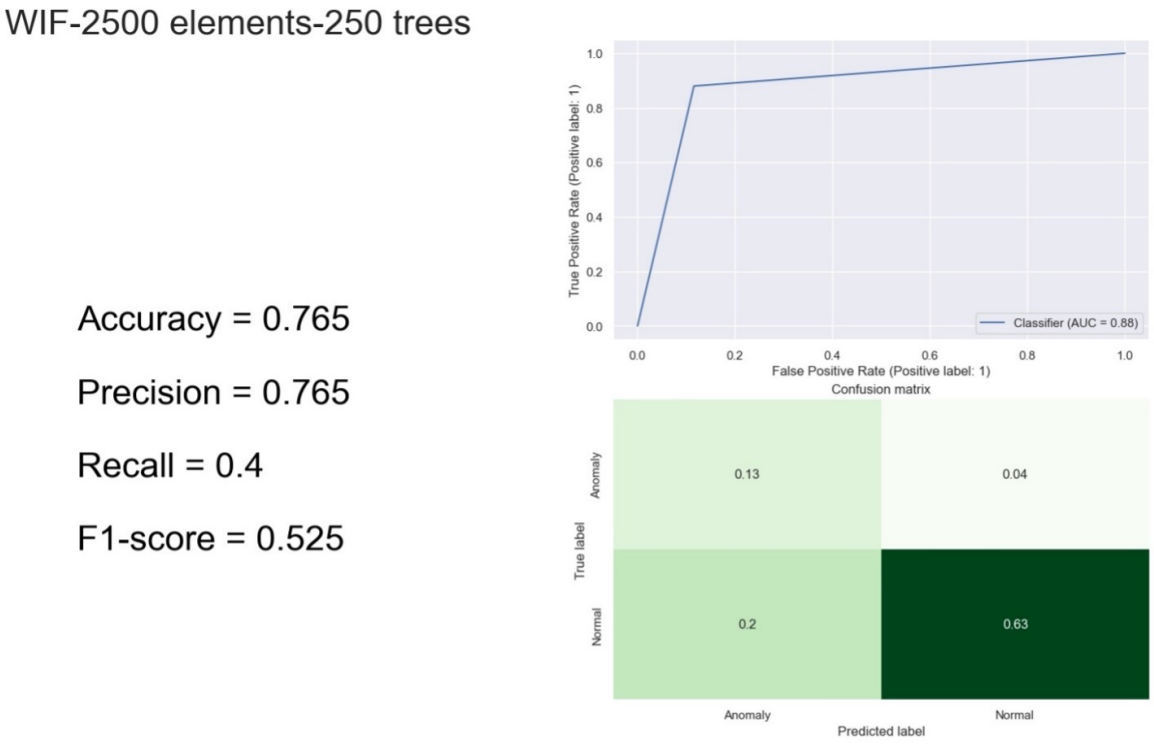


Рис.4.20. Результаты работы алгоритма ВИЛ, размер подвыборки 2500, ИД – 250

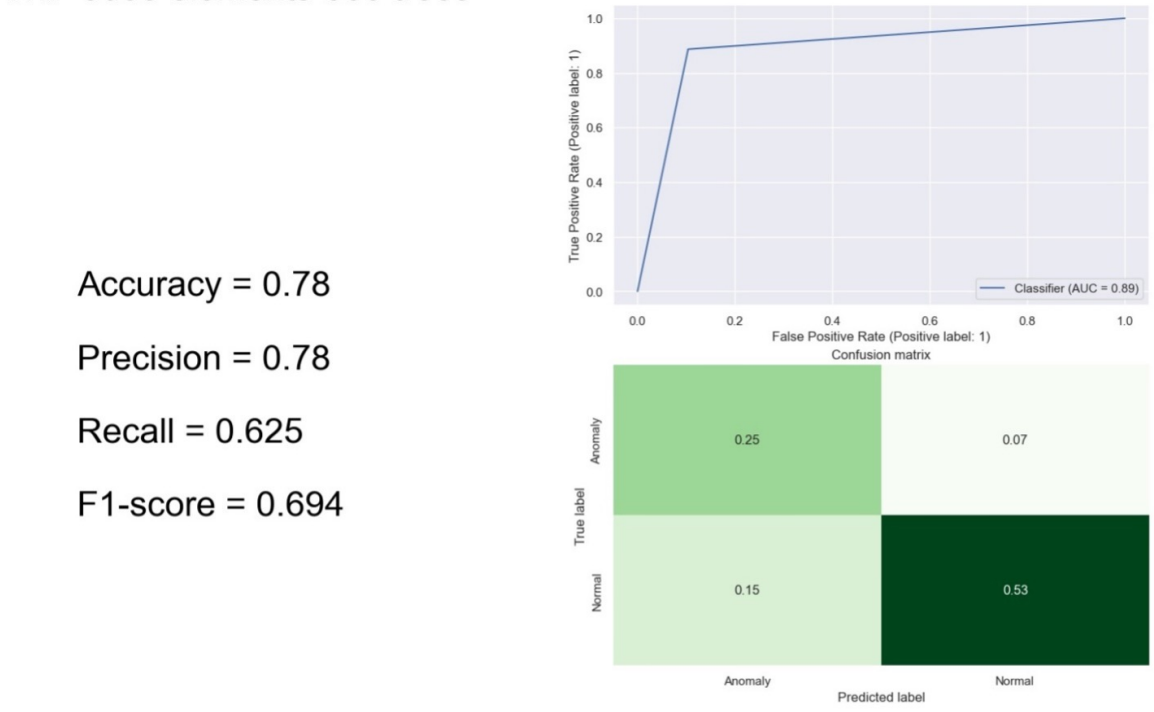


Рис.4.21. Результаты работы алгоритма ВИЛ, размер подвыборки 5000, ИД – 500

WIF-7500 elements-750 trees

Accuracy = 0.805

Precision = 0.805

Recall = 0.36

F1-score = 0.498

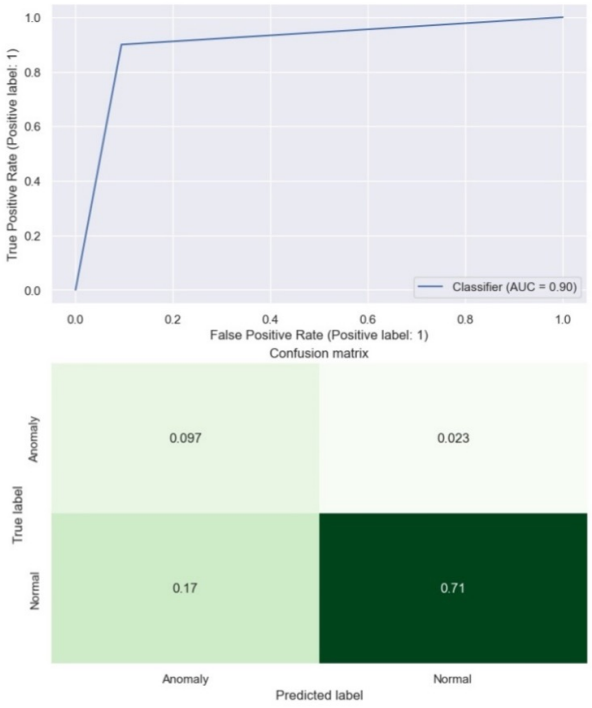


Рис.4.22. Результаты работы алгоритма ВИЛ, размер подвыборки 7500, ИД – 750

WIF-10000 elements-1000 trees

Accuracy = 0.82

Precision = 0.82

Recall = 0.532

F1-score = 0.646

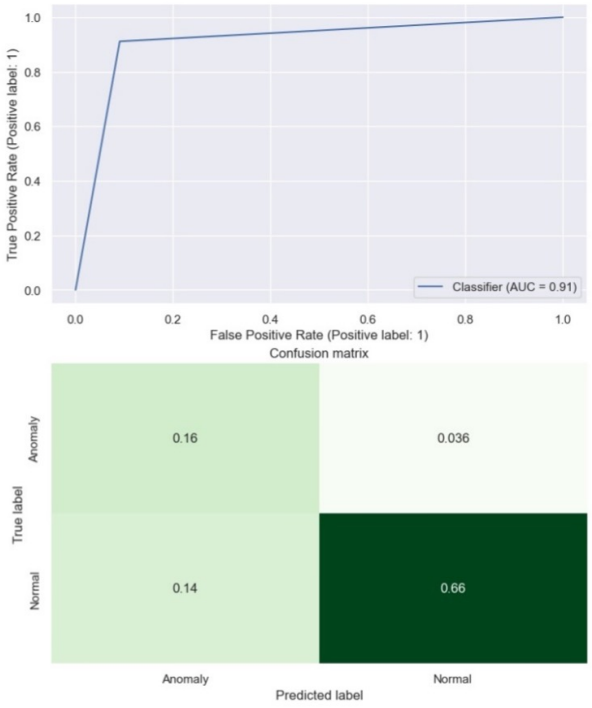


Рис.4.23. Результаты работы алгоритма ВИЛ, размер подвыборки 10000, ИД – 1000

4.3 Анализ результатов разработанной модели

Согласно вышеприведённым признакам моделей, матрицам ошибок и величин метрик составлен анализ для каждой модели ИД. Необходимо учитывать, что тесты проводились на реальных транзакционных данных, предобработка которых хоть и проведена без потери общности, но всё же влияет на конечный результат.

Модель ИЛ:

- имеет самую высокую скорость рекурсивного построения ИД, вплоть до 100 ИД за 1 секунду;
- график кривой ROC AUC не пологий, так как построение осуществляется по результатам модели ИЛ, которые представляют собой два вектора: вектор истинных меток классов и вектор предсказанных меток. Величина угла кривой обратно пропорционально точности модели, справедливо для всех моделей ИЛ;
- модель обычного ИЛ имеет низкую сложность реализации, её реализация является основой для всех последующих моделей с модификациями;
- точность пропорциональна размеру выборки, с самого начала эксперимента (2500 элементов) у данной модели наблюдаются самые низкие показатели метрик, последующее увеличение объёма выборки снижает их ещё больше;
- последовательное увеличение количества деревьев до 1000 положительно отразилось на величинах метрик, что ещё раз доказывает практическую эффективность эмпирически найденной авторами алгоритма величины параметра.

Модель РИЛ:

- строит ИД медленнее, чем модель ИЛ, в среднем 80–90 ИД за 1 секунду;
- модель РИЛ несколько труднее реализовать, чем модель ИЛ, за счёт реализации перегруженных методов существующего класса ИЛ;
- точность пропорциональна размеру выборки, с самого начала эксперимента (2500 элементов) у данной модели наблюдаются самые низкие показатели метрик, последующее увеличение объёма выборки снижает их в меньшей степени, чем в случае модели ИЛ;
- последовательное увеличение количества деревьев до 1000 положительно отразилось на величинах метрик в большей степени, чем у ИЛ.

Модель ИЛСИ:

- строит ИД наравне с РИЛ: примерно 75–95 ИД за 1 секунду;
- сложность реализации модели ИЛСИ сопоставима со сложностью реализации модели РИЛ;
- модель ИЛСИ демонстрирует аналогичное модели РИЛ поведение при увеличении объема выборки;
- последовательное увеличение количества деревьев весьма положительно сказывается на эффективность ИЛСИ, при равных значениях данного параметра модель ИЛСИ точнее модели РИЛ.

Модель ОИЛ:

- модель ОИЛ строит ИД сравнительно быстро 80-85 ИД в секунду;
- для реализации модели ОИЛ необходимо было создать отдельный класс с полями и методами, чья функциональность обеспечивает анализ распределений узлов ИД, а также вычисление расстояния между узлами в контексте заданного распределения, всё это делает модель ОИЛ самой сложной для реализации;
- точность обнаружения аномальных элементов вовсе не падает с увеличением объема выборки, а наоборот возрастает, чего не наблюдается у остальных уже рассмотренных моделей ИЛ;
- увеличение количества ИД также положительно сказывается на точности, хоть и замедляет процесс построения ИД в среднем до 70–75 в секунду.

Модель ВИЛ:

- скорость построения ИД моделью ВИЛ – 80 ИД в секунду, что является стабильным показателем на протяжении всех тестов;
- реализация модели ВИЛ также потребовала создания отдельного класса, но с гораздо меньшим количеством методов, в основе которых заключался механизм распределения и учёта весовых коэффициентов;
- точность обнаружения также не падает с увеличением объема выборки, более того, наблюдается незначительный в сравнении с моделью ОИЛ рост;
- модель ВИЛ эффективнее обнаруживает аномальные экземпляры при увеличении количества ИД, необходимо отметить константность времени процедуры построения с минимальным количеством отклонений от эталонного измерения, чего не наблюдалось у остальных моделей ИЛ.

Разработанная и реализованная модель ВИЛ в ходе тестирования обнаружения аномалий на реальных транзакционных данных показала себя наиболее сбалансированной моделью ВИЛ. Выявление диапазона параметров количества ИД и объема выборки позволяет достичь большей точности, чем у других модификаций ИЛ: моделей РИЛ и ИЛСИ. Несмотря на то, что модель обобщённого изолирующего леса во всех тестах показала себя эффективнее, модель ВИЛ не обладает сильной зависимостью скорости построения ИД от их количества и её проще реализовать.

4.4 Выводы

Таким образом были проведены эксперименты для 5 вариантов алгоритма ИЛ:

- модель базового ИЛ без модификаций;
- модель ОИЛ;
- модель ИЛСИ;
- модель РИЛ;
- модель ВИЛ.

Разработанная и реализованная модель ВИЛ в ходе тестирования обнаружения аномалий на реальных транзакционных данных показала себя наиболее сбалансированной моделью ВИЛ. Время работы алгоритма позволяет применять его в реальном времени при процессировании транзакций. Выявление диапазона параметров количества ИД и объема выборки позволяет достичь большей точности, чем у других модификаций ИЛ: моделей РИЛ и ИЛСИ.

ЗАКЛЮЧЕНИЕ

В настоящей работе был исследован алгоритм ИЛ и его модификации: РИЛ, ИЛСИ, ОИЛ. Также была предложена и изучена собственная модификация ИЛ – весовой изолирующий лес. Исследованные алгоритмы позволяют решить задачу обнаружения аномалий, в частности в работе исследовались реальные транзакционные данные.

Был проведен подробный обзор литературы для выявления особенностей работы алгоритмов ИЛ, преимуществ и недостатков существующих решений. В результате анализа сделаны выводы, что ключевая особенность алгоритмов ИЛ заключается в тенденции изолировать примеры-аутсайдеры на относительно ранних этапах дерева, благодаря чему достигается высокая скорость и относительная простота; на данный момент нет прямой связи между производительностью ИЛ и его вариациями, и предположениями, которые затрагивают изначальное распределение данных; все подходы на основе ИЛ аппроксимируют базовое распределение вероятностей и рассматривают среднее значение длины пути как приближение к смешанному весу.

Была разработана модификация алгоритма ИЛ – ВИЛ, которая основана на рассмотрении $E[h(x)]$ в качестве некоторого неизвестного распределения, математическое ожидание которого является искомой величиной для оценки наблюдения. Для решения задачи были определены веса w_i, для чего выполняется минимизация функции потерь (формула 2.23). Минимизация функции сводится к решению задачи квадратичной оптимизации (формула 2.31).

Реализация алгоритмов выполнялась на языке C++ 20 без использования сторонних библиотек. Для реализации модификаций ИЛ создано пространство имён, поля и методы которого описывают модель ИЛ без модификаций. Далее для каждой модификации реализованы расширения пространства имён с перегрузкой изменяемых полей и методов.

Разработанная и реализованная модель ВИЛ в ходе тестирования обнаружения аномалий на реальных транзакционных данных показала себя наиболее сбалансированной моделью ИЛ. Время работы алгоритма позволяет применять его в реальном времени при процессировании транзакций. Выявление диапазона параметров количества ИД и объема выборки позволяет достичь большей точности, чем у других модификаций ИЛ: моделей РИЛ и ИЛСИ. Несмотря на то, что модель обобщённого изолирующего леса во всех тестах показала себя эффективнее,

модель ВИЛ не обладает сильной зависимостью скорости построения ИД от их количества и её проще реализовать.

Автор выражает искреннюю благодарность за помощь в подготовке данной работы Уткину Л.В., Курочкину М.А., Попову С.Г., Большакову А.А. и всему профессорско-преподавательскому составу Высшей школы искусственного интеллекта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Салкуцан А.* Анализ влияния компьютерных атак на банковскую систему Российской Федерации // Вестник Российского экономического университета имени Г. В. Плеханова. — 2019. — № 6. — С. 202—210. — URL: https://doi.org/10.21686/2413-2829-2019-6-202-210.

2. A Comprehensive Survey of Data Mining-based Fraud Detection Research / C. Phua [и др.] // CoRR. — 2010. — Т. abs/1009.6119. — arXiv: 1009.6119. — URL: http://arxiv.org/abs/1009.6119.

3. *Aggarwal C. C.* Outlier Analysis. — Springer, 2013. — URL: http://dx.doi.org/10.1007/978-1-4614-6396-2.

4. *Aggarwal C. C.*, *Sathe S.* Theoretical Foundations and Algorithms for Outlier Ensembles // SIGKDD Explor. Newsl. — New York, NY, USA, 2015. — Т. 17, № 1. — С. 24—47. — DOI 10.1145/2830544.2830549. — URL: https://doi.org/10.1145/2830544.2830549.

5. *Angiulli F.*, *Pizzuti C.* Fast Outlier Detection in High Dimensional Spaces // PKDD. — 2002.

6. Anomaly Detection in Finance: Editors' Introduction / A. Anandakrishnan [и др.] // Proceedings of the KDD 2017: Workshop on Anomaly Detection in Finance. Т. 71 / под ред. A. Anandakrishnan [и др.]. — PMLR, 2018. — С. 1—7. — (Сер.: Proceedings of Machine Learning Research). — URL: https://proceedings.mlr.press/v71/anandakrishnan18a.html.

7. *Buschjager S.*, *Honysz P.-J.*, *Morik K.* Randomized outlier detection with trees // International Journal of Data Science and Analytics. — 2022. — Т. 13. — С. 1—14. — DOI 10.1007/s41060-020-00238-w.

8. *Chandola V.*, *Banerjee A.*, *Kumar V.* Anomaly Detection: A Survey // ACM Comput. Surv. — New York, NY, USA, 2009. — Т. 41, № 3. — DOI 10.1145/1541880.1541882. — URL: https://doi.org/10.1145/1541880.1541882.

9. FP-outlier: Frequent pattern based outlier detection / Z. He [и др.] // Comput. Sci. Inf. Syst. — 2005. — Т. 2. — С. 103—118.

10. *Gao Z.*, *Ye M.* A framework for data mining-based anti-money laundering research // Journal of Money Laundering Control. — 2007. — Т. 10. — DOI 10.1108/13685200710746875.

11. *Hariri S.*, *Kind M. C.*, *Brunner R. J.* Extended Isolation Forest // CoRR. — 2018. — T. abs/1811.02141. — arXiv: 1811.02141. — URL: http://arxiv.org/abs/1811.02141.

12. *Harman R.*, *Lacko V.* On decompositional algorithms for uniform sampling from n-spheres and n-balls // Journal of Multivariate Analysis. — 2010. — T. 101, № 10. — C. 2297—2304. — DOI https://doi.org/10.1016/j.jmva.2010.06.002. — URL: https://www.sciencedirect.com/science/article/pii/S0047259X10001211.

13. *Knorr E. M.*, *Ng R. T.*, *Tucakov V.* Distance-Based Outliers: Algorithms and Applications // The VLDB Journal. — Berlin, Heidelberg, 2000. — T. 8, № 3/4. — C. 237—253. — DOI 10.1007/s007780050006. — URL: https://doi.org/10.1007/s007780050006.

14. *Liu F. T.*, *Ting K. M.*, *Zhou Z.-H.* Isolation forest // 2008 Eighth IEEE International Conference on Data Mining. — IEEE. 2008. — C. 413—422.

15. *Liu F. T.*, *Ting K. M.*, *Zhou Z.-H.* On Detecting Clustered Anomalies Using SCiForest // ECML/PKDD. — 2010.

16. LOF: Identifying Density-Based Local Outliers / M. M. Breunig [и др.] // Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. — Dallas, Texas, USA: Association for Computing Machinery, 2000. — C. 93—104. — (Сер.: SIGMOD '00). — DOI 10.1145/342009.335388. — URL: https://doi.org/10.1145/342009.335388.

17. *Peherstorfer B.*, *Pflüge D.*, *Bungartz H.-J.* Density Estimation with Adaptive Sparse Grids for Large Data Sets // Proceedings of the 2014 SIAM International Conference on Data Mining (SDM). — C. 443—451. — DOI 10.1137/1.9781611973440.51. — eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611973440.51. — URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611973440.51.

18. *Ram P.*, *Gray A. G.* Density Estimation Trees //. — San Diego, California, USA: Association for Computing Machinery, 2011. — C. 627—635. — (Сер.: KDD '11). — DOI 10.1145/2020408.2020507. — URL: https://doi.org/10.1145/2020408.2020507.

19. *Zengan G.* Application of Cluster-Based Local Outlier Factor Algorithm in Anti-Money Laundering //. — 2009. — C. 1—4. — DOI 10.1109/ICMSS.2009.5302396.

20. *Пахомова А.*, *Чи А.* ПРИМЕНЕНИЕ АЛГОРИТМА ЕМ ДЛЯ ГАУССОВОЙ СМЕСИ // Научный взгляд в будущее. — 2021. — Т. 1, № 18—01. — С. 29—34. — DOI 10.30888/2415-7538.2020-18-01-028. — URL: https://www.scilook.eu/index.php/slif/article/view/slif18-01-028.

21. Cylon Data Engineering Everywhere! — URL: https://github.com/cylondata/cylon (visited on 13.06.2022).

22. matplotlib-cpp. — URL: https://github.com/Cryoris/matplotlib-cpp (visited on 13.06.2022).

23. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study / G. Campos [и др.] // Data Mining and Knowledge Discovery. — 2016. — Т. 30, № 4. — С. 891—927. — DOI 10.1007/s10618-015-0444-8.

24. OpenML. — URL: https://www.openml.org/search?type=data&sort=runs&id=40897&status=active (visited on 13.06.2022).

25. *Schubert E.*, *Zimek A.*, *Kriegel H.* Local outlier detection reconsidered: A generalized view on locality with applications to spatial, video, and network outlier detection // Data Mining and Knowledge Discovery. — 2014. — Т. 28, № 1. — С. 190—237. — DOI 10.1007/s10618-012-0300-z.

26. t-Distributed Stochastic Neighbor Embedding (t-SNE). — URL: https://lvdmaaten.github.io/tsne/ (visited on 13.06.2022).

Приложение 1

Результаты экспериментов

В настоящем приложении приведены результаты всех экспериментов, описанных в главе 4.

Результат работы алгоритма ИЛ без модификаций

IF-2500 elements-250 trees

Accuracy = 0.733

Precision = 0.733

Recall = 0.503

F1-score = 0.597

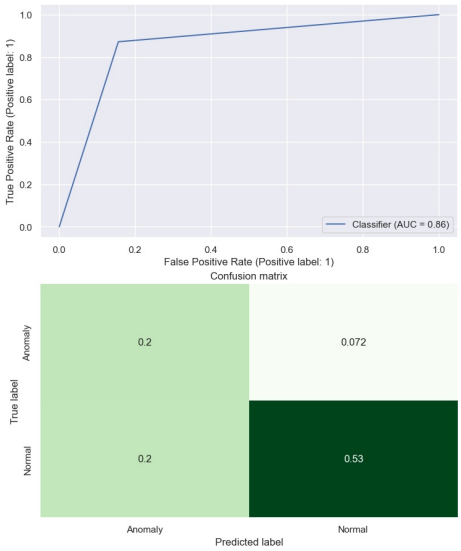


Рис.П1.1. Результаты работы алгоритма ИЛ, размер подвыборки 2500, ИД – 250

IF-2500 elements-500 trees

Accuracy = 0.738

Precision = 0.738

Recall = 0.558

F1-score = 0.635

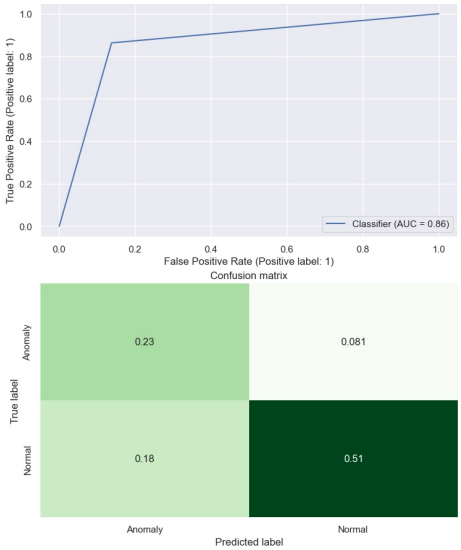


Рис.П1.2. Результаты работы алгоритма ИЛ, размер подвыборки 2500, ИД – 500

IF-2500 elements-750 trees

Accuracy = 0.735

Precision = 0.735

Recall = 0.255

F1-score = 0.379

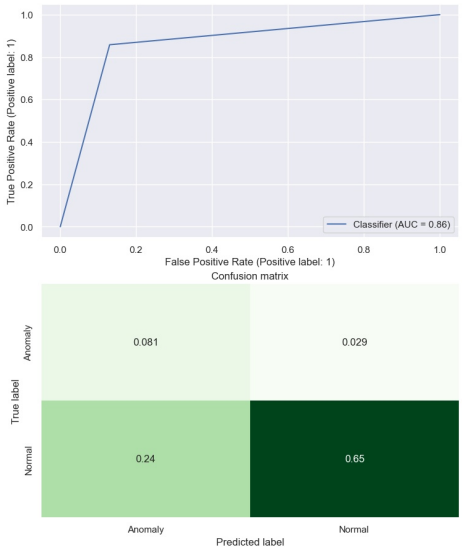


Рис.П1.3. Результаты работы алгоритма ИЛ, размер подвыборки 2500, ИД – 750

IF-2500 elements-1000 trees

Accuracy = 0.745

Precision = 0.745

Recall = 0.48

F1-score = 0.584

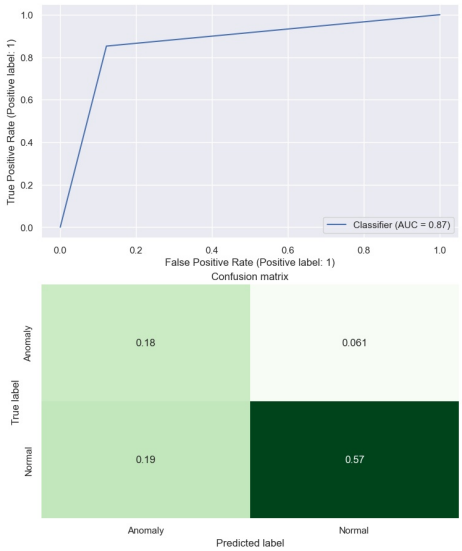


Рис.П1.4. Результаты работы алгоритма ИЛ, размер подвыборки 2500, ИД – 1000

IF-5000 elements-250 trees

Accuracy = 0.733

Precision = 0.733

Recall = 0.528

F1-score = 0.614

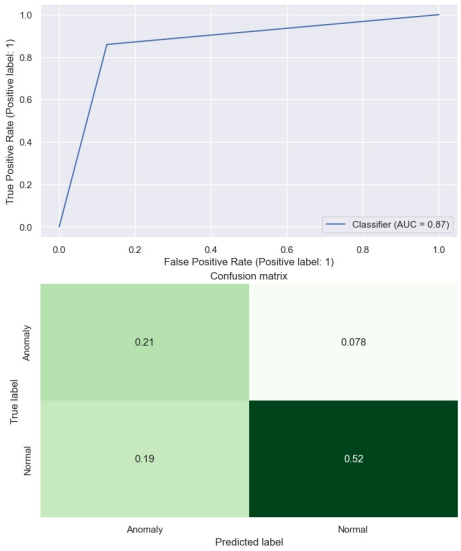


Рис.П1.5. Результаты работы алгоритма ИЛ, размер подвыборки 5000, ИД – 250

IF-5000 elements-500 trees

Accuracy = 0.72

Precision = 0.72

Recall = 0.222

F1-score = 0.34

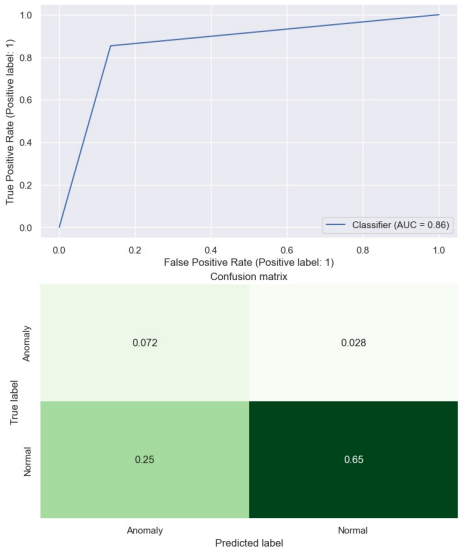


Рис.П1.6. Результаты работы алгоритма ИЛ, размер подвыборки 5000, ИД – 500

IF-5000 elements-750 trees

Accuracy = 0.74

Precision = 0.74

Recall = 0.561

F1-score = 0.638

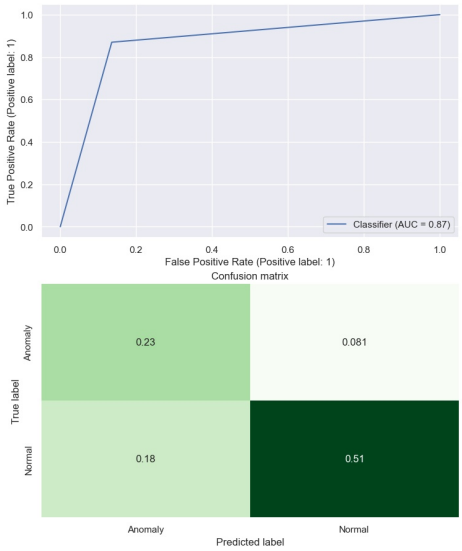


Рис.П1.7. Результаты работы алгоритма ИЛ, размер подвыборки 5000, ИД – 750

IF-5000 elements-1000 trees

Accuracy = 0.76

Precision = 0.76

Recall = 0.5

F1-score = 0.603

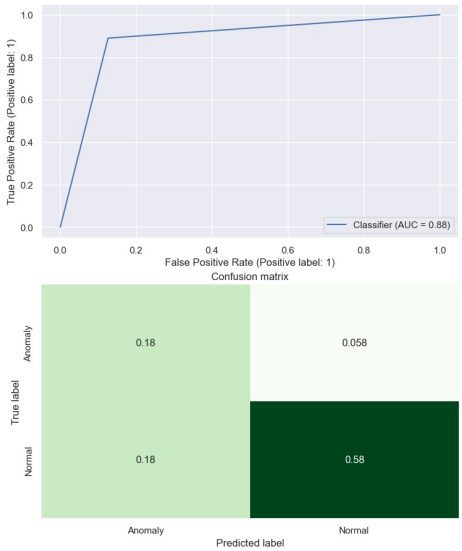


Рис.П1.8. Результаты работы алгоритма ИЛ, размер подвыборки 5000, ИД – 1000

IF-7500 elements-250 trees

Accuracy = 0.713

Precision = 0.712

Recall = 0.538

F1-score = 0.613

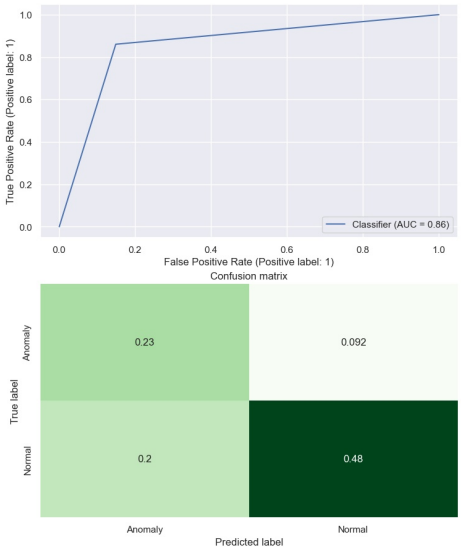


Рис.П1.9. Результаты работы алгоритма ИЛ, размер подвыборки 7500, ИД – 250

IF-7500 elements-500 trees

Accuracy = 0.725

Precision = 0.725

Recall = 0.283

F1-score = 0.407

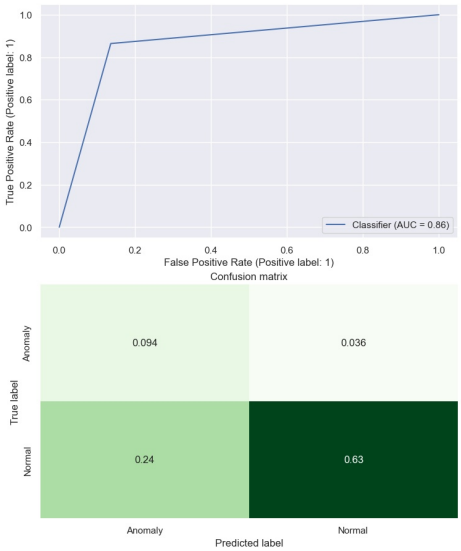


Рис.П1.10. Результаты работы алгоритма ИЛ, размер подвыборки 7500, ИД – 500

IF-7500 elements-750 trees

Accuracy = 0.725

Precision = 0.725

Recall = 0.481

F1-score = 0.578

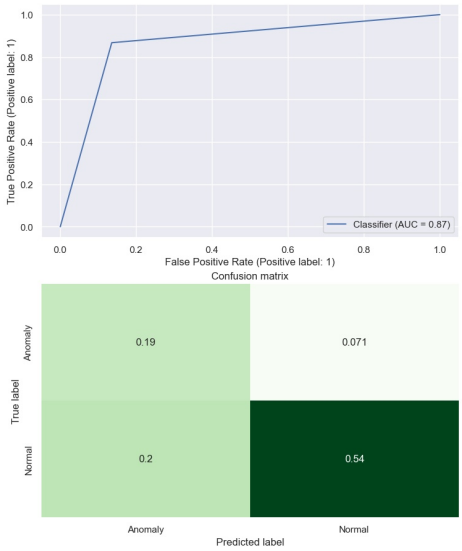


Рис.П1.11. Результаты работы алгоритма ИЛ, размер подвыборки 7500, ИД – 750

IF-7500 elements-1000 trees

Accuracy = 0.76

Precision = 0.76

Recall = 0.486

F1-score = 0.593

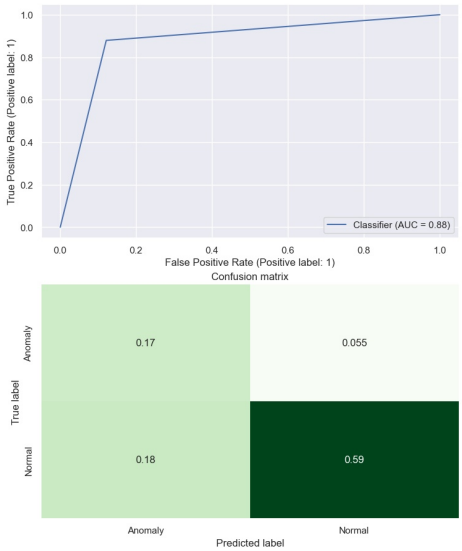


Рис.П1.12. Результаты работы алгоритма ИЛ, размер подвыборки 7500, ИД – 1000

IF-10000 elements-250 trees

Accuracy = 0.743

Precision = 0.743

Recall = 0.529

F1-score = 0.618

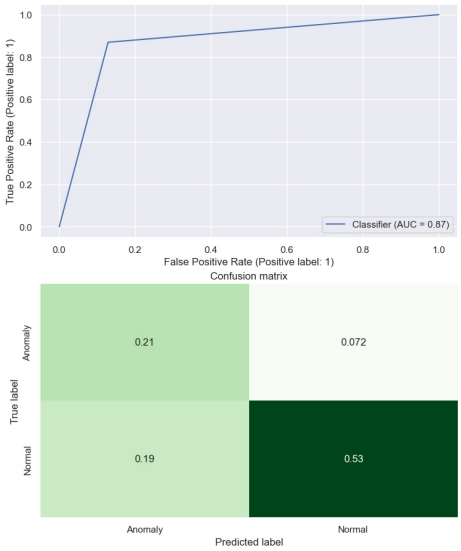


Рис.П1.13. Результаты работы алгоритма ИЛ, размер подвыборки 10000, ИД – 250

IF-10000 elements-500 trees

Accuracy = 0.74

Precision = 0.74

Recall = 0.4

F1-score = 0.52

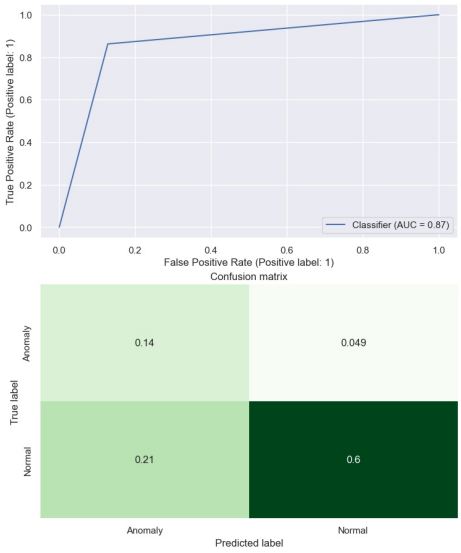


Рис.П1.14. Результаты работы алгоритма ИЛ, размер подвыборки 10000, ИД – 500

IF-10000 elements-750 trees

Accuracy = 0.73

Precision = 0.73

Recall = 0.418

F1-score = 0.532

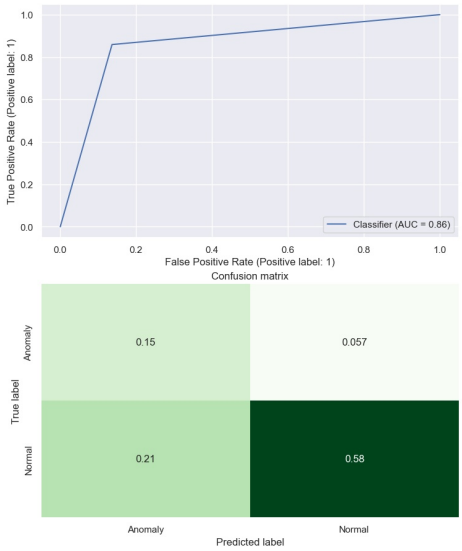


Рис.П1.15. Результаты работы алгоритма ИЛ, размер подвыборки 10000, ИД – 750

IF-10000 elements-1000 trees

Accuracy = 0.71

Precision = 0.71

Recall = 0.38

F1-score = 0.495

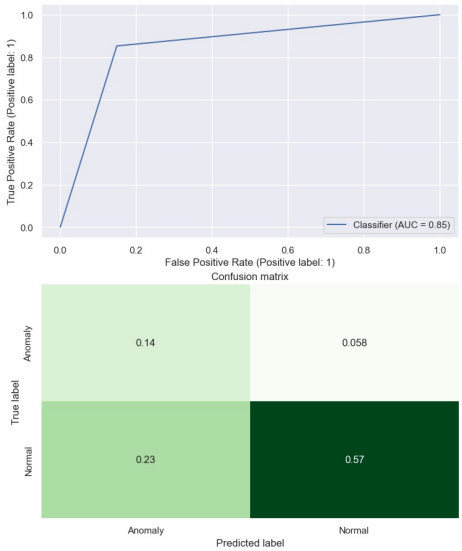


Рис.П1.16. Результаты работы алгоритма ИЛ, размер подвыборки 10000, ИД – 1000

Результат работы алгоритма РИЛ

EIF-2500 elements-250 trees

Accuracy = 0.775

Precision = 0.775

Recall = 0.56

F1-score = 0.65

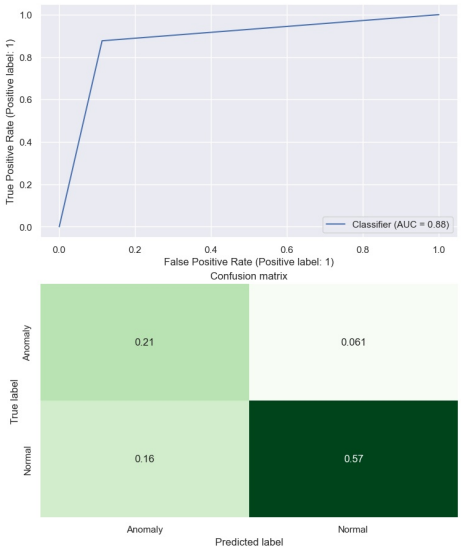


Рис.П1.17. Результаты работы алгоритма РИЛ, размер подвыборки 2500, ИД – 250

EIF-2500 elements-500 trees

Accuracy = 0.788

Precision = 0.788

Recall = 0.525

F1-score = 0.63

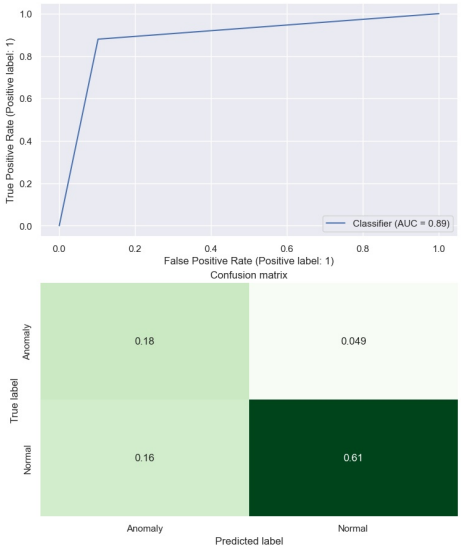


Рис.П1.18. Результаты работы алгоритма РИЛ, размер подвыборки 2500, ИД – 500

EIF-2500 elements-750 trees

Accuracy = 0.793

Precision = 0.793

Recall = 0.653

F1-score = 0.716

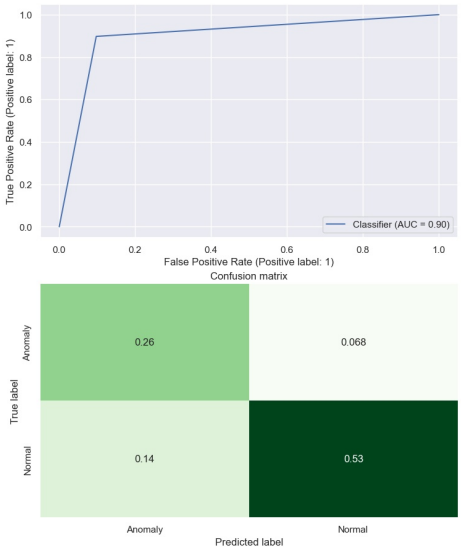


Рис.П1.19. Результаты работы алгоритма РИЛ, размер подвыборки 2500, ИД – 750

EIF-2500 elements-1000 trees

Accuracy = 0.775

Precision = 0.775

Recall = 0.447

F1-score = 0.567

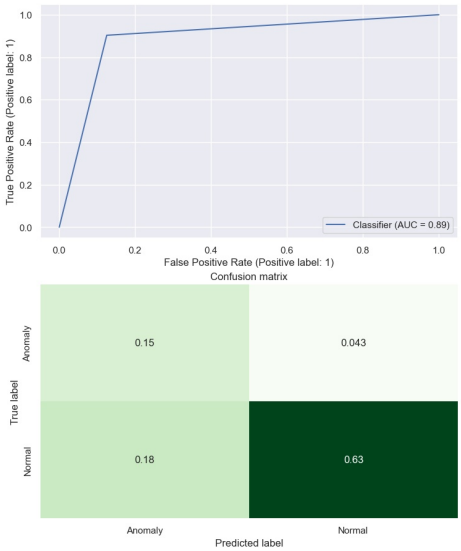


Рис.П1.20. Результаты работы алгоритма РИЛ, размер подвыборки 2500, ИД – 1000

EIF-5000 elements-250 trees

Accuracy = 0.78

Precision = 0.78

Recall = 0.603

F1-score = 0.68

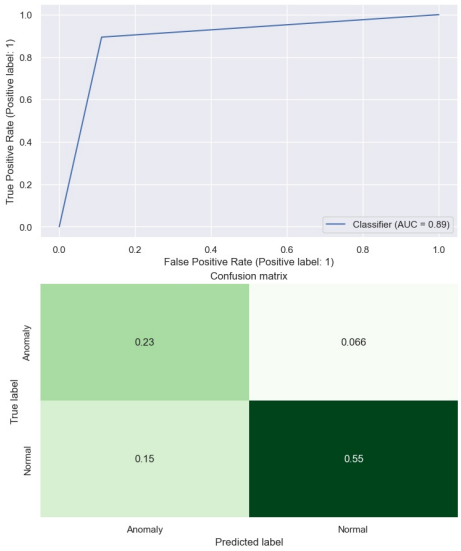


Рис.П1.21. Результаты работы алгоритма РИЛ, размер подвыборки 5000, ИД – 250

EIF-5000 elements-500 trees

Accuracy = 0.795

Precision = 0.795

Recall = 0.601

F1-score = 0.685

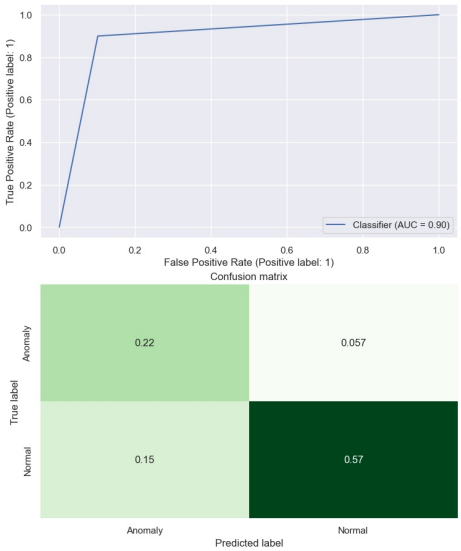
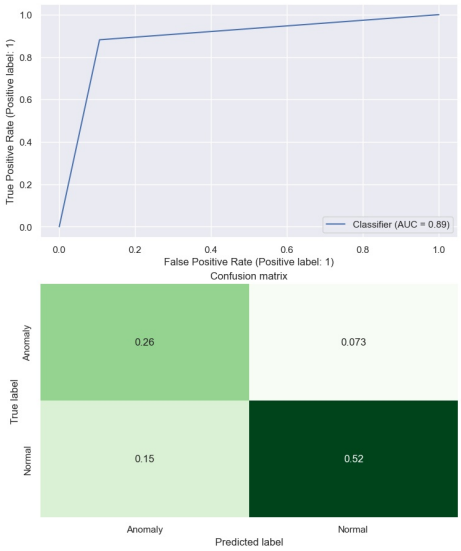


Рис.П1.22. Результаты работы алгоритма РИЛ, размер подвыборки 5000, ИД – 500

EIF-5000 elements-750 trees



Accuracy = 0.778

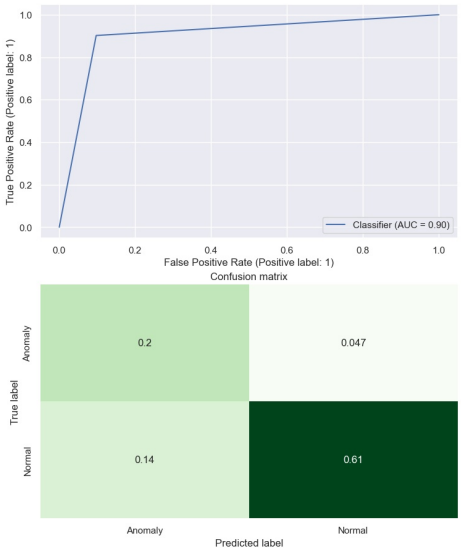
Precision = 0.778

Recall = 0.633

F1-score = 0.698

Рис.П1.23. Результаты работы алгоритма РИЛ, размер подвыборки 5000, ИД – 750

EIF-5000 elements-1000 trees



Accuracy = 0.81

Precision = 0.81

Recall = 0.587

F1-score = 0.681

Рис.П1.24. Результаты работы алгоритма РИЛ, размер подвыборки 5000, ИД – 1000

EIF-7500 elements-250 trees

Accuracy = 0.788

Precision = 0.788

Recall = 0.496

F1-score = 0.609

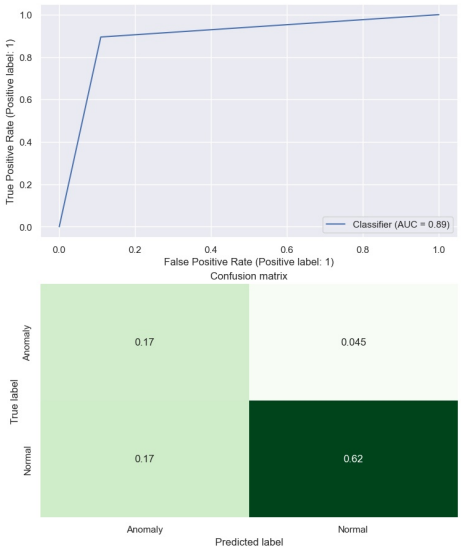


Рис.П1.25. Результаты работы алгоритма РИЛ, размер подвыборки 7500, ИД – 250

EIF-7500 elements-500 trees

Accuracy = 0.795

Precision = 0.795

Recall = 0.492

F1-score = 0.608

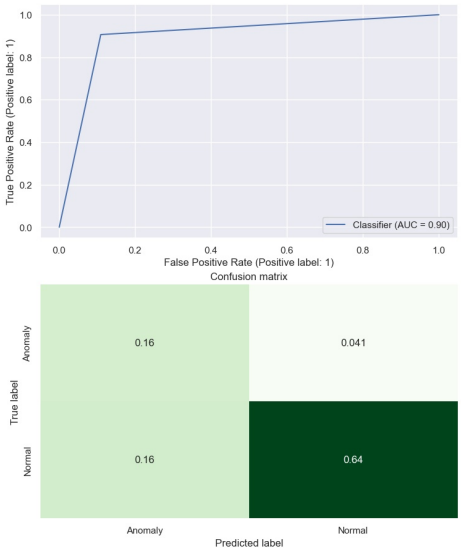


Рис.П1.26. Результаты работы алгоритма РИЛ, размер подвыборки 7500, ИД – 500

EIF-7500 elements-750 trees

Accuracy = 0.765

Precision = 0.765

Recall = 0.626

F1-score = 0.689

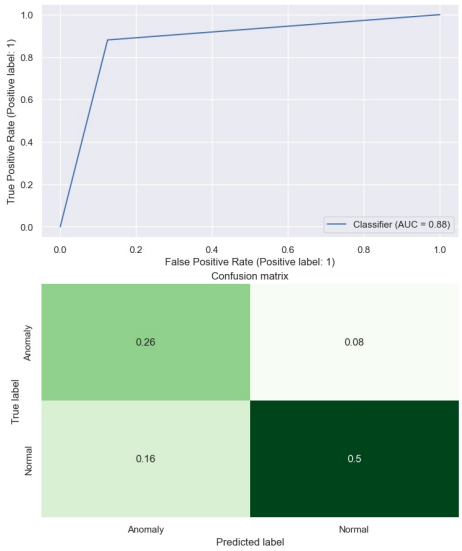


Рис.П1.27. Результаты работы алгоритма РИЛ, размер подвыборки 7500, ИД – 750

EIF-7500 elements-1000 trees

Accuracy = 0.79

Precision = 0.79

Recall = 0.452

F1-score = 0.575

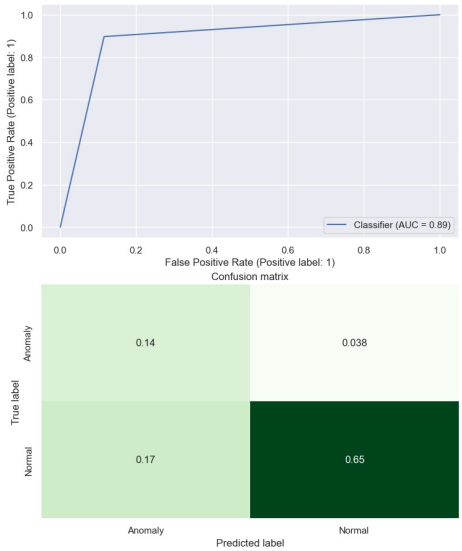


Рис.П1.28. Результаты работы алгоритма РИЛ, размер подвыборки 7500, ИД – 1000

EIF-10000 elements-250 trees

Accuracy = 0.773

Precision = 0.773

Recall = 0.474

F1-score = 0.588

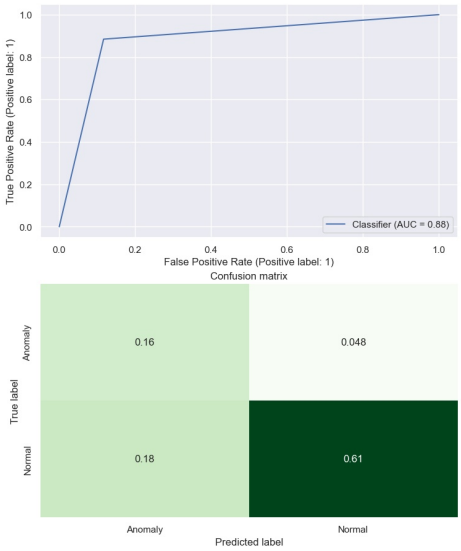


Рис.П1.29. Результаты работы алгоритма РИЛ, размер подвыборки 10000, ИД – 250

EIF-10000 elements-500 trees

Accuracy = 0.77

Precision = 0.77

Recall = 0.471

F1-score = 0.584

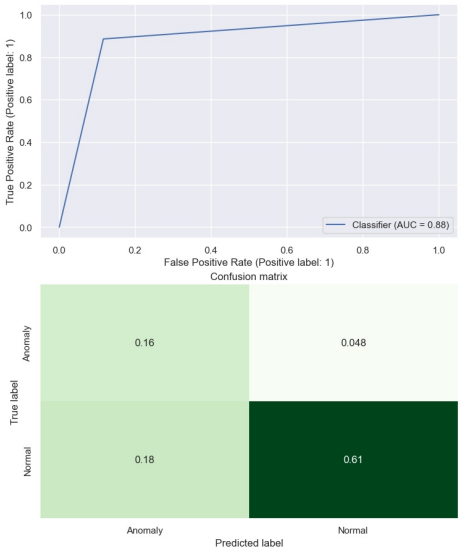


Рис.П1.30. Результаты работы алгоритма РИЛ, размер подвыборки 10000, ИД – 500

EIF-10000 elements-750 trees

Accuracy = 0.778

Precision = 0.778

Recall = 0.302

F1-score = 0.435

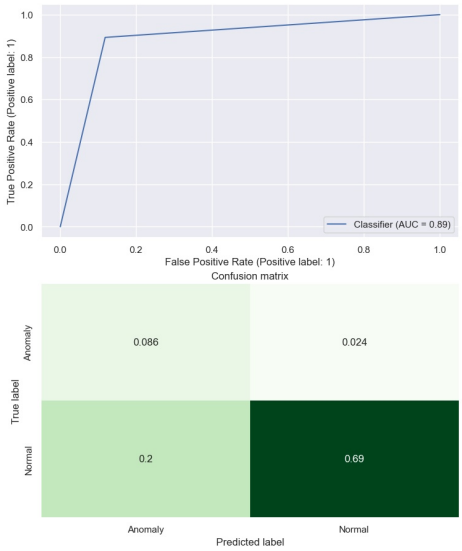


Рис.П1.31. Результаты работы алгоритма РИЛ, размер подвыборки 10000, ИД – 750

EIF-10000 elements-1000 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.484

F1-score = 0.603

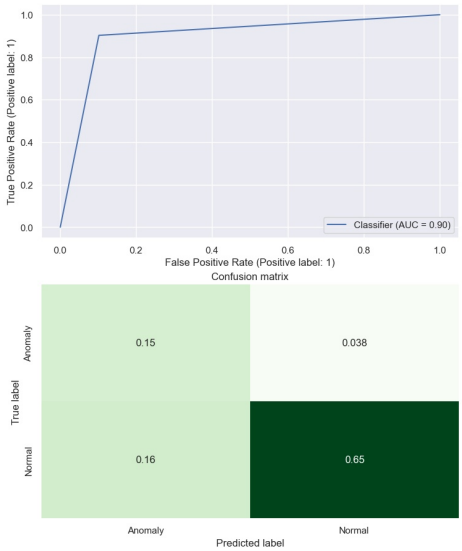


Рис.П1.32. Результаты работы алгоритма РИЛ, размер подвыборки 10000, ИД – 1000

Результат работы алгоритма ИЛСИ

SCiForest-2500 elements-250 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.484

F1-score = 0.603

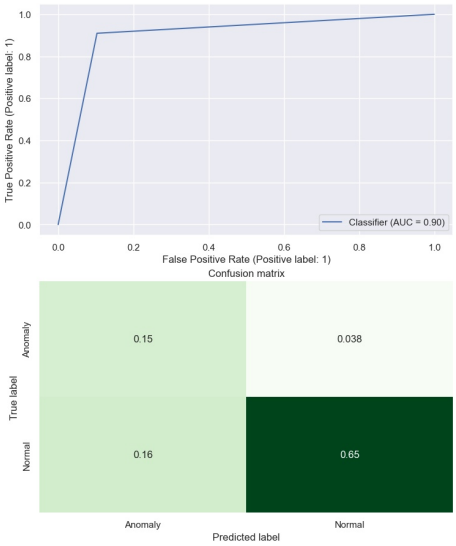


Рис.П1.33. Результаты работы алгоритма ИЛСИ, размер подвыборки 2500, ИД – 250

SCiForest-2500 elements-500 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.642

F1-score = 0.713

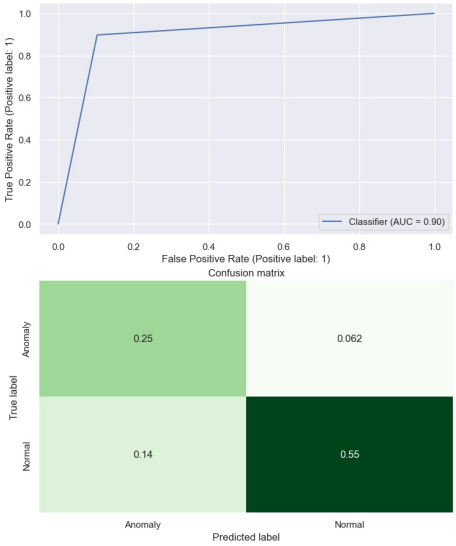


Рис.П1.34. Результаты работы алгоритма ИЛСИ, размер подвыборки 2500, ИД – 500

SCiForest-2500 elements-750 trees

Accuracy = 0.82

Precision = 0.82

Recall = 0.615

F1-score = 0.703

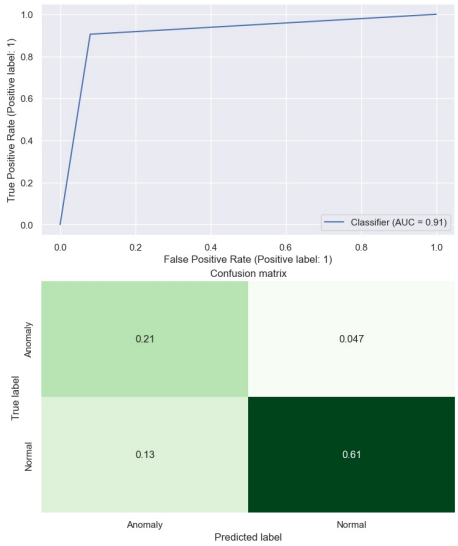


Рис.П1.35. Результаты работы алгоритма ИЛСИ, размер подвыборки 2500, ИД – 750

SCiForest-2500 elements-1000 trees

Accuracy = 0.835

Precision = 0.835

Recall = 0.684

F1-score = 0.752

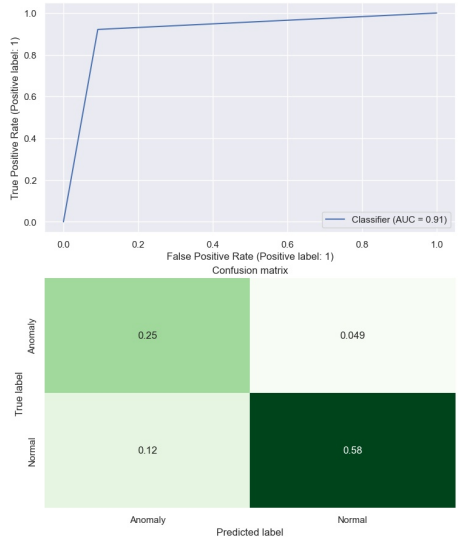


Рис.П1.36. Результаты работы алгоритма ИЛСИ, размер подвыборки 2500, ИД – 1000

SCiForest-5000 elements-250 trees

Accuracy = 0.79

Precision = 0.79

Recall = 0.556

F1-score = 0.653

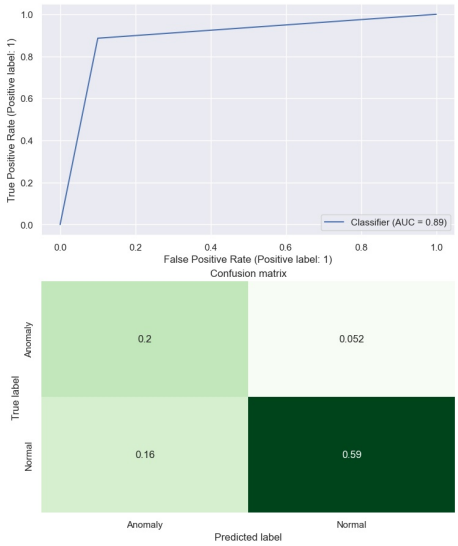


Рис.П1.37. Результаты работы алгоритма ИЛСИ, размер подвыборки 5000, ИД – 250

SCiForest-5000 elements-500 trees

Accuracy = 0.81

Precision = 0.81

Recall = 0.531

F1-score = 0.642

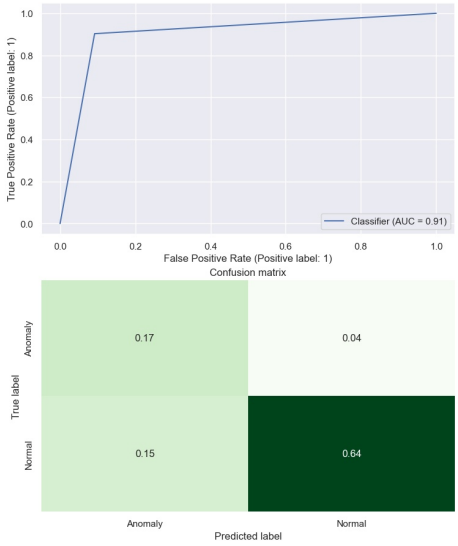


Рис.П1.38. Результаты работы алгоритма ИЛСИ, размер подвыборки 5000, ИД – 500

SCiForest-5000 elements-750 trees

Accuracy = 0.795

Precision = 0.795

Recall = 0.646

F1-score = 0.713

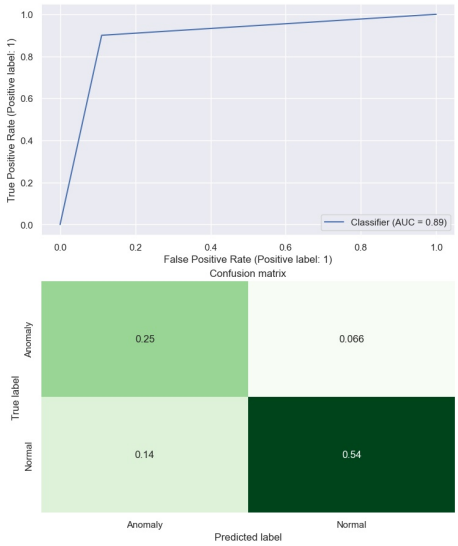


Рис.П1.39. Результаты работы алгоритма ИЛСИ, размер подвыборки 5000, ИД – 750

SCiForest-5000 elements-1000 trees

Accuracy = 0.795

Precision = 0.795

Recall = 0.577

F1-score = 0.668

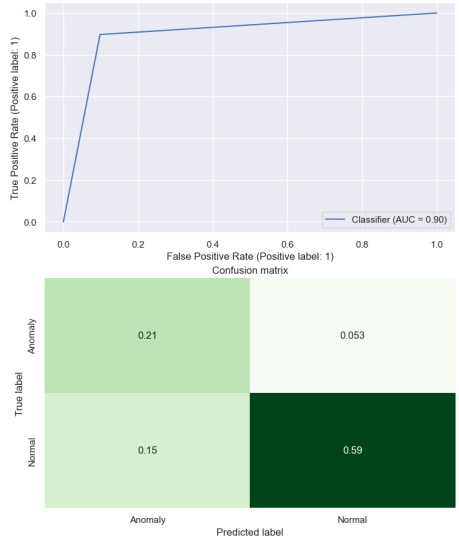


Рис.П1.40. Результаты работы алгоритма ИЛСИ, размер подвыборки 5000, ИД – 1000

SCiForest-7500 elements-250 trees

Accuracy = 0.785

Precision = 0.785

Recall = 0.575

F1-score = 0.663

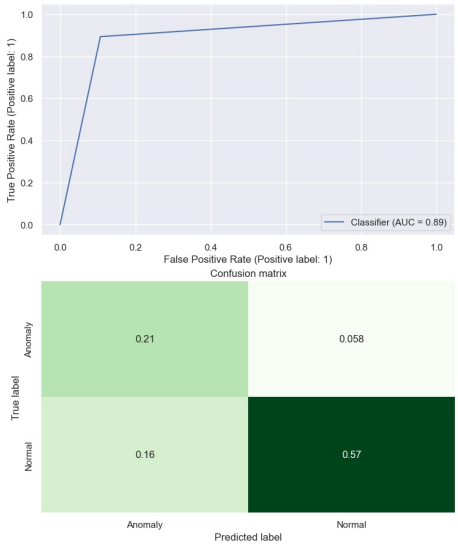


Рис.П1.41. Результаты работы алгоритма ИЛСИ, размер подвыборки 7500, ИД – 250

SCiForest-7500 elements-500 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.374

F1-score = 0.51

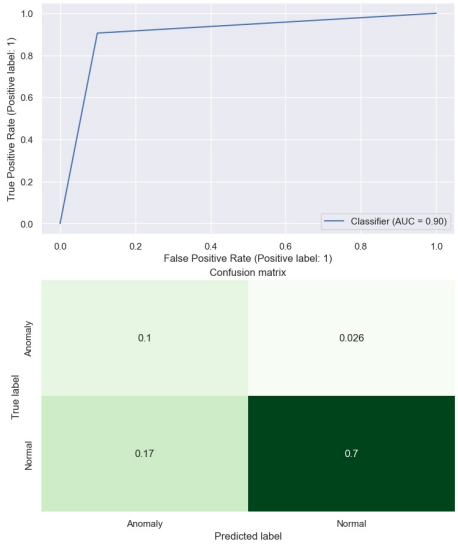


Рис.П1.42. Результаты работы алгоритма ИЛСИ, размер подвыборки 7500, ИД – 500

SCiForest-7500 elements-750 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.414

F1-score = 0.545

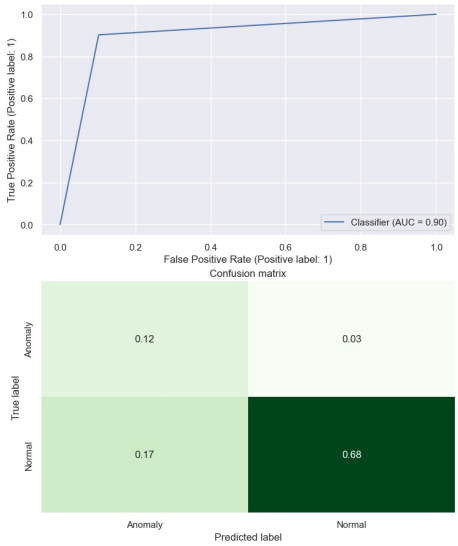


Рис.П1.43. Результаты работы алгоритма ИЛСИ, размер подвыборки 7500, ИД – 750

SCiForest-7500 elements-1000 trees

Accuracy = 0.805

Precision = 0.805

Recall = 0.552

F1-score = 0.655

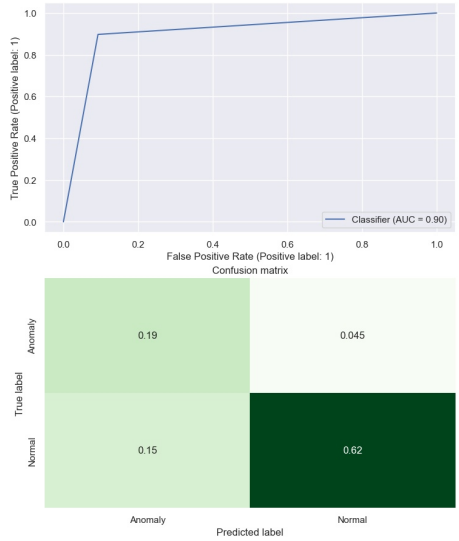


Рис.П1.44. Результаты работы алгоритма ИЛСИ, размер подвыборки 7500, ИД – 1000

SCiForest-10000 elements-250 trees

Accuracy = 0.79

Precision = 0.79

Recall = 0.529

F1-score = 0.634

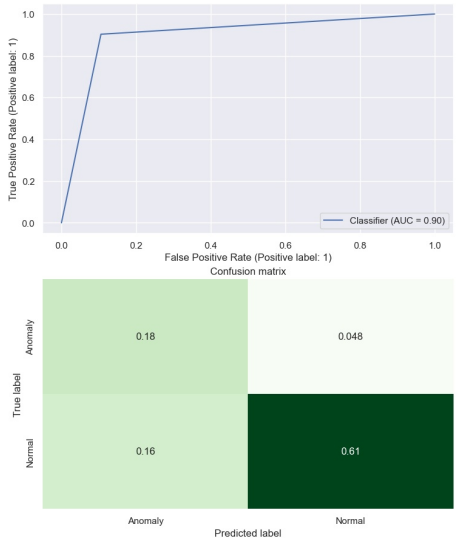


Рис.П1.45. Результаты работы алгоритма ИЛСИ, размер подвыборки 10000, ИД – 250

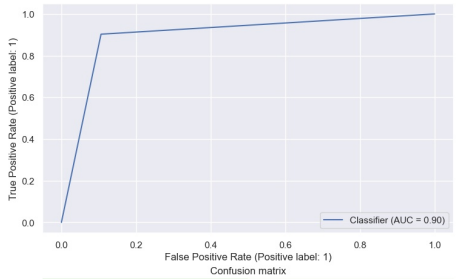
SCiForest-10000 elements-500 trees

Accuracy = 0.8

Precision = 0.8

Recall = 0.432

F1-score = 0.561



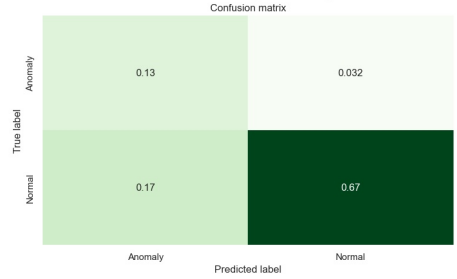


Рис.П1.46. Результаты работы алгоритма ИЛСИ, размер подвыборки 10000, ИД – 500

SCiForest-10000 elements-750 trees

Accuracy = 0.803

Precision = 0.803

Recall = 0.612

F1-score = 0.695

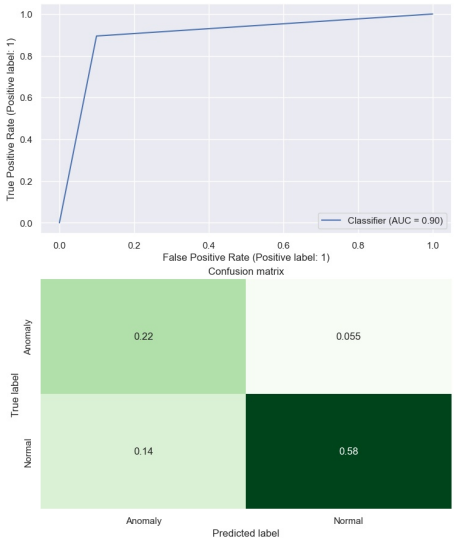


Рис.П1.47. Результаты работы алгоритма ИЛСИ, размер подвыборки 10000, ИД – 750

SCiForest-10000 elements-1000 trees

Accuracy = 0.82

Precision = 0.82

Recall = 0.36

F1-score = 0.501

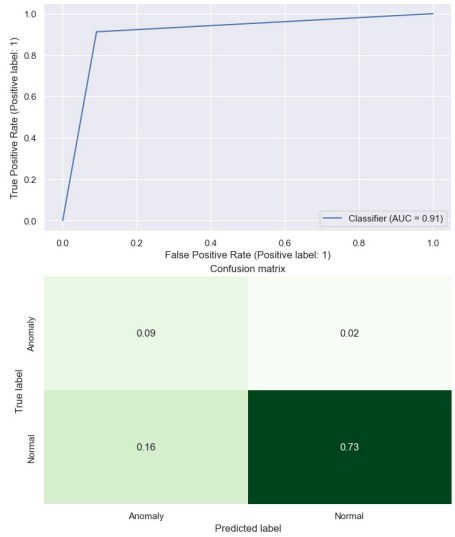


Рис.П1.48. Результаты работы алгоритма ИЛСИ, размер подвыборки 10000, ИД – 1000

Результат работы алгоритма ОИЛ

GIF-2500 elements-250 trees

Accuracy = 0.915

Precision = 0.915

Recall = 0.799

F1-score = 0.853

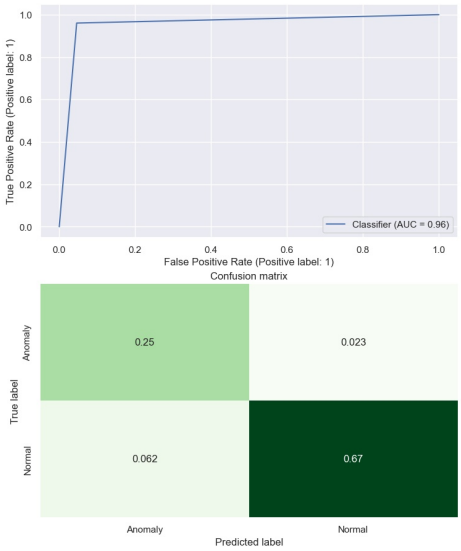


Рис.П1.49. Результаты работы алгоритма ОИЛ, размер подвыборки 2500, ИД – 250

GIF-2500 elements-500 trees

Accuracy = 0.945

Precision = 0.945

Recall = 0.701

F1-score = 0.805

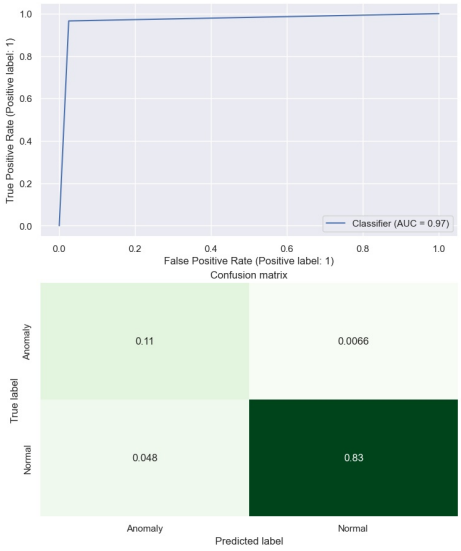


Рис.П1.50. Результаты работы алгоритма ОИЛ, размер подвыборки 2500, ИД – 500

GIF-2500 elements-750 trees

Accuracy = 0.935

Precision = 0.935

Recall = 0.848

F1-score = 0.89

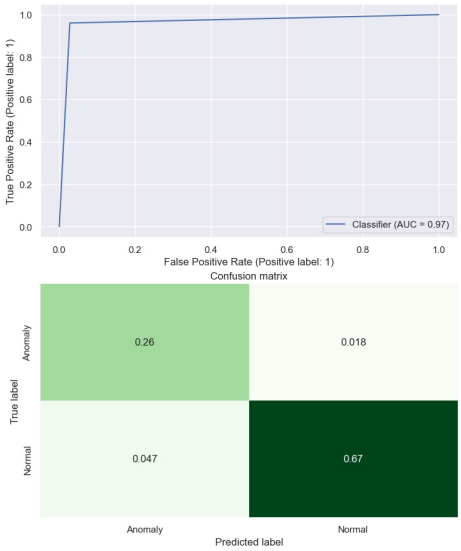


Рис.П1.51. Результаты работы алгоритма ОИЛ, размер подвыборки 2500, ИД – 750

GIF-2500 elements-1000 trees

Accuracy = 0.975

Precision = 0.975

Recall = 0.925

F1-score = 0.949

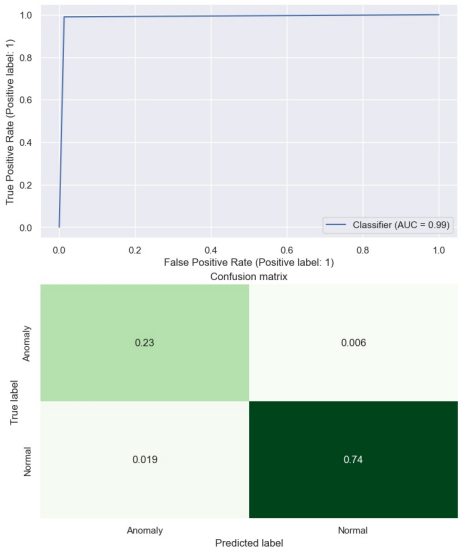


Рис.П1.52. Результаты работы алгоритма ОИЛ, размер подвыборки 2500, ИД – 1000

GIF-5000 elements-250 trees

Accuracy = 0.91

Precision = 0.91

Recall = 0.845

F1-score = 0.876

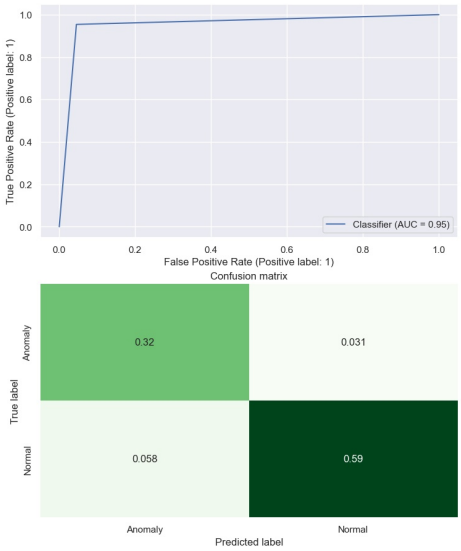


Рис.П1.53. Результаты работы алгоритма ОИЛ, размер подвыборки 5000, ИД – 250

GIF-5000 elements-500 trees

Accuracy = 0.92

Precision = 0.92

Recall = 0.831

F1-score = 0.873

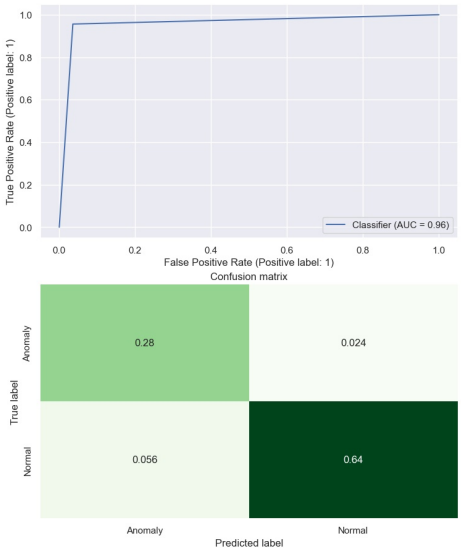


Рис.П1.54. Результаты работы алгоритма ОИЛ, размер подвыборки 5000, ИД – 500

GIF-5000 elements-750 trees

Accuracy = 0.93

Precision = 0.93

Recall = 0.701

F1-score = 0.799

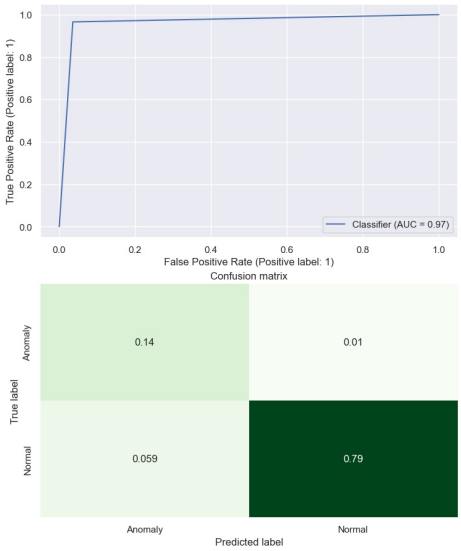


Рис.П1.55. Результаты работы алгоритма ОИЛ, размер подвыборки 5000, ИД – 750

GIF-5000 elements-1000 trees

Accuracy = 0.975

Precision = 0.975

Recall = 0.932

F1-score = 0.953

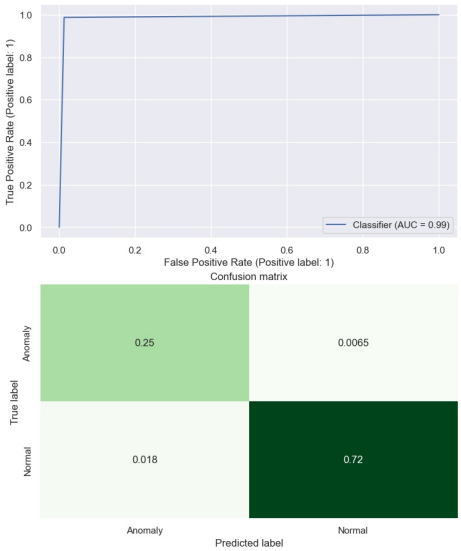


Рис.П1.56. Результаты работы алгоритма ОИЛ, размер подвыборки 5000, ИД – 1000

GIF-7500 elements-250 trees

Accuracy = 0.922

Precision = 0.922

Recall = 0.79

F1-score = 0.851

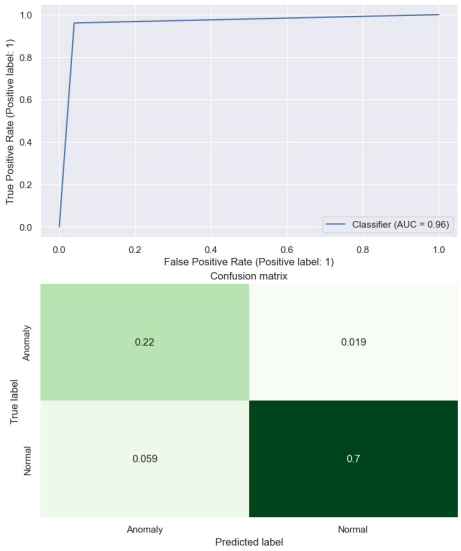


Рис.П1.57. Результаты работы алгоритма ОИЛ, размер подвыборки 7500, ИД – 250

GIF-7500 elements-500 trees

Accuracy = 0.94

Precision = 0.94

Recall = 0.87

F1-score = 0.904

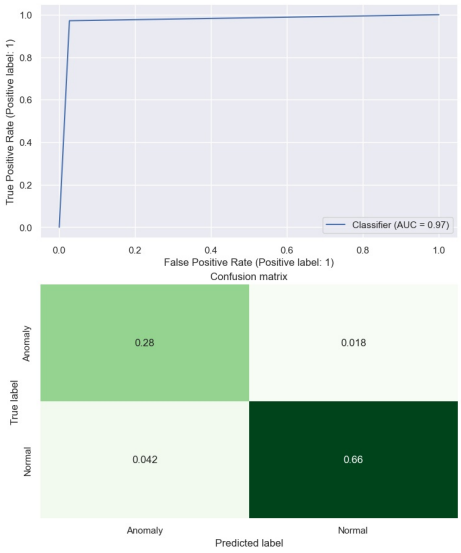


Рис.П1.58. Результаты работы алгоритма ОИЛ, размер подвыборки 7500, ИД – 500

GIF-7500 elements-750 trees

Accuracy = 0.948

Precision = 0.948

Recall = 0.809

F1-score = 0.873

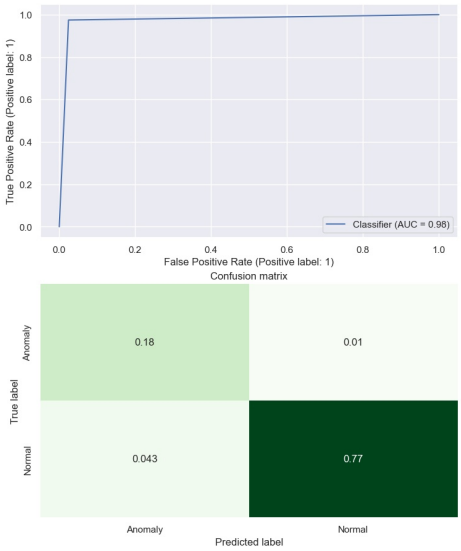


Рис.П1.59. Результаты работы алгоритма ОИЛ, размер подвыборки 7500, ИД – 750

GIF-7500 elements-1000 trees

Accuracy = 0.928

Precision = 0.928

Recall = 0.783

F1-score = 0.849

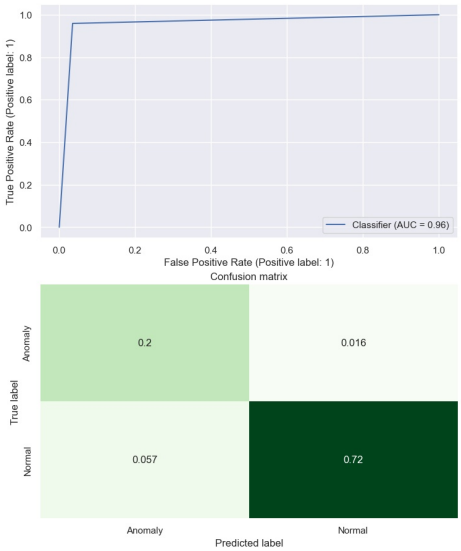


Рис.П1.60. Результаты работы алгоритма ОИЛ, размер подвыборки 7500, ИД – 1000

GIF-10000 elements-250 trees

Accuracy = 0.935

Precision = 0.935

Recall = 0.811

F1-score = 0.869

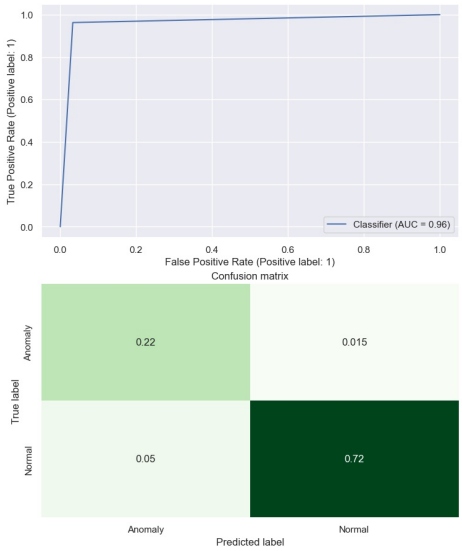


Рис.П1.61. Результаты работы алгоритма ОИЛ, размер подвыборки 10000, ИД – 250

GIF-10000 elements-500 trees

Accuracy = 0.95

Precision = 0.95

Recall = 0.807

F1-score = 0.872

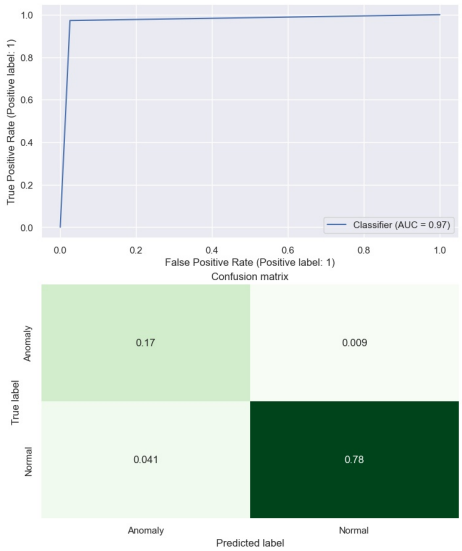


Рис.П1.62. Результаты работы алгоритма ОИЛ, размер подвыборки 10000, ИД – 500

GIF-10000 elements-750 trees

Accuracy = 0.91

Precision = 0.91

Recall = 0.771

F1-score = 0.835

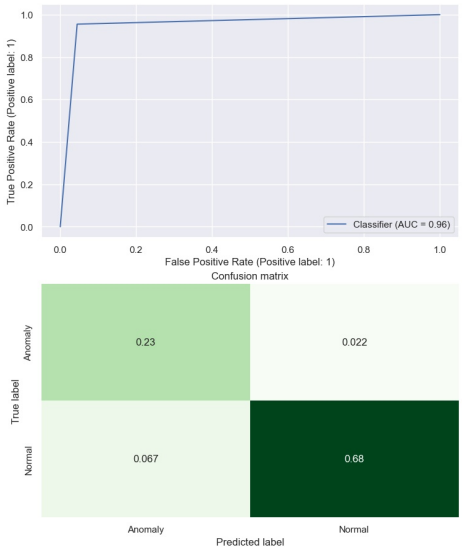


Рис.П1.63. Результаты работы алгоритма ОИЛ, размер подвыборки 10000, ИД – 750

GIF-10000 elements-1000 trees

Accuracy = 0.99

Precision = 0.99

Recall = 0.973

F1-score = 0.982

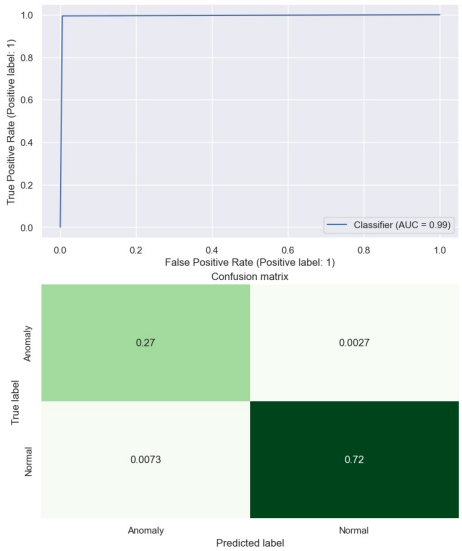


Рис.П1.64. Результаты работы алгоритма ОИЛ, размер подвыборки 10000, ИД – 1000

Результат работы алгоритма ВИЛ

WIF-2500 elements-250 trees

Accuracy = 0.765

Precision = 0.765

Recall = 0.4

F1-score = 0.525

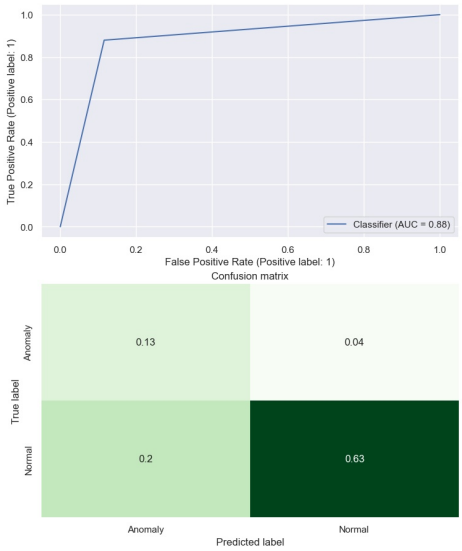


Рис.П1.65. Результаты работы алгоритма ВИЛ, размер подвыборки 2500, ИД – 250

WIF-2500 elements-500 trees

Accuracy = 0.773

Precision = 0.773

Recall = 0.393

F1-score = 0.521

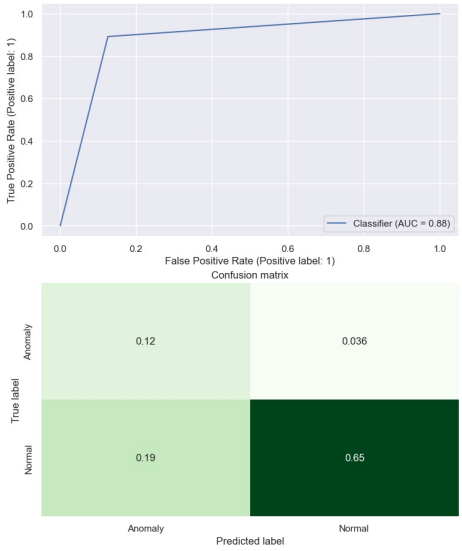


Рис.П1.66. Результаты работы алгоритма ВИЛ, размер подвыборки 2500, ИД – 500

WIF-2500 elements-750 trees

Accuracy = 0.768

Precision = 0.768

Recall = 0.496

F1-score = 0.603

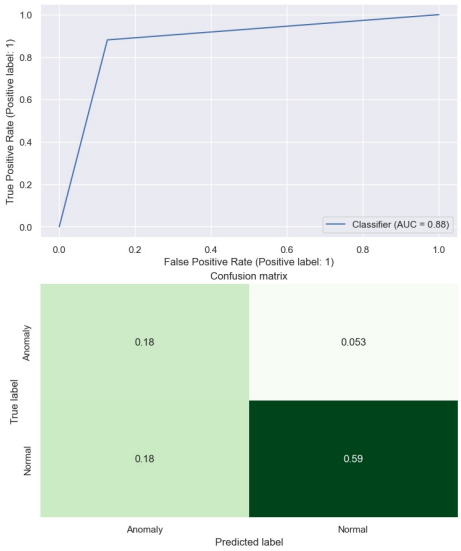


Рис.П1.67. Результаты работы алгоритма ВИЛ, размер подвыборки 2500, ИД – 750

WIF-2500 elements-1000 trees

Accuracy = 0.76

Precision = 0.76

Recall = 0.457

F1-score = 0.571

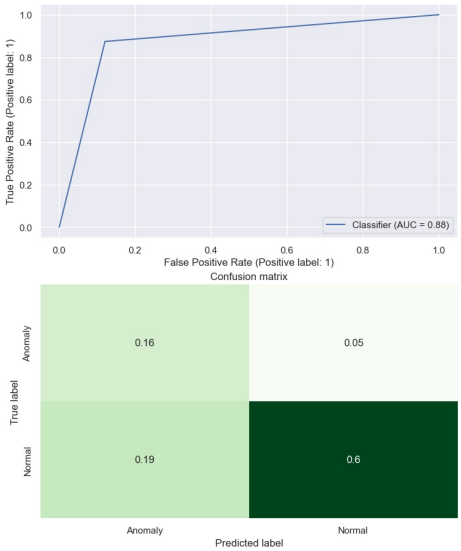


Рис.П1.68. Результаты работы алгоритма ВИЛ, размер подвыборки 2500, ИД – 1000

WIF-5000 elements-250 trees

Accuracy = 0.76

Precision = 0.76

Recall = 0.472

F1-score = 0.582

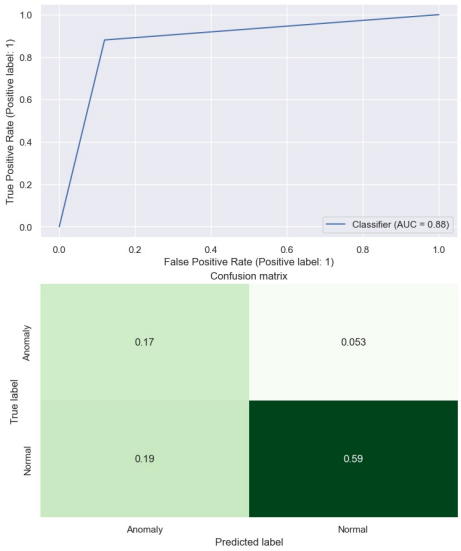


Рис.П1.69. Результаты работы алгоритма ВИЛ, размер подвыборки 5000, ИД – 250

WIF-5000 elements-500 trees

Accuracy = 0.78

Precision = 0.78

Recall = 0.625

F1-score = 0.694

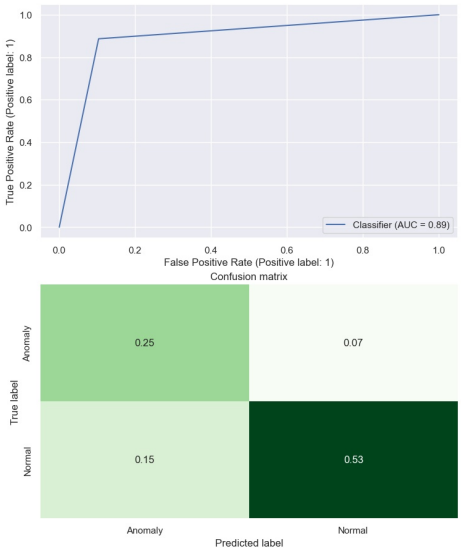


Рис.П1.70. Результаты работы алгоритма ВИЛ, размер подвыборки 5000, ИД – 500

WIF-5000 elements-750 trees

Accuracy = 0.765

Precision = 0.765

Recall = 0.493

F1-score = 0.6

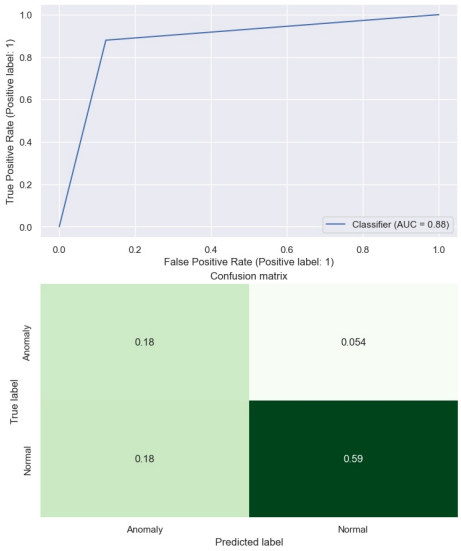


Рис.П1.71. Результаты работы алгоритма ВИЛ, размер подвыборки 5000, ИД – 750

WIF-5000 elements-1000 trees

Accuracy = 0.79

Precision = 0.79

Recall = 0.317

F1-score = 0.453

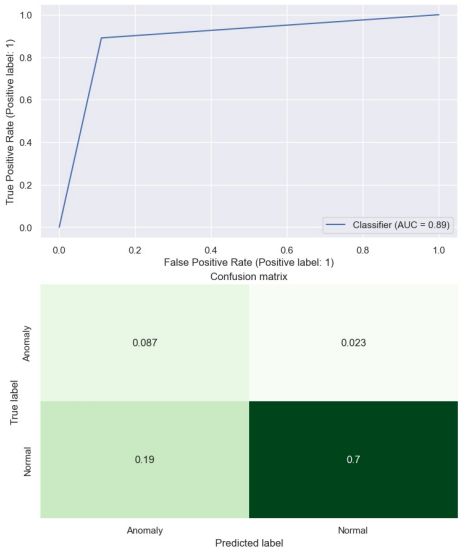


Рис.П1.72. Результаты работы алгоритма ВИЛ, размер подвыборки 5000, ИД – 1000

WIF-7500 elements-250 trees

Accuracy = 0.773

Precision = 0.773

Recall = 0.626

F1-score = 0.691

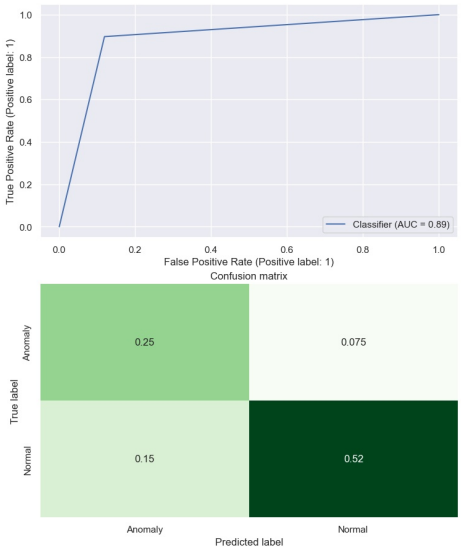


Рис.П1.73. Результаты работы алгоритма ВИЛ, размер подвыборки 7500, ИД – 250

WIF-7500 elements-500 trees

Accuracy = 0.78

Precision = 0.78

Recall = 0.485

F1-score = 0.598

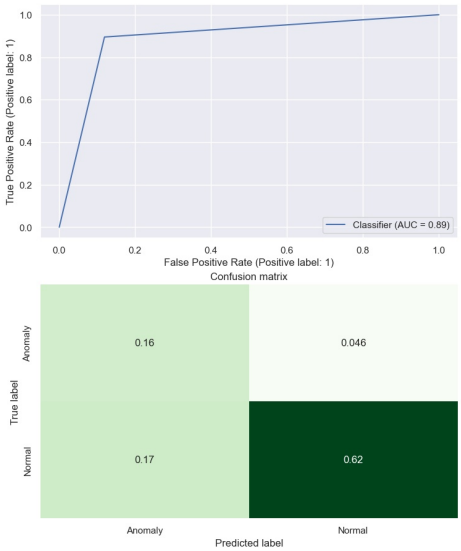


Рис.П1.74. Результаты работы алгоритма ВИЛ, размер подвыборки 7500, ИД – 500

WIF-7500 elements-750 trees

Accuracy = 0.805

Precision = 0.805

Recall = 0.36

F1-score = 0.498

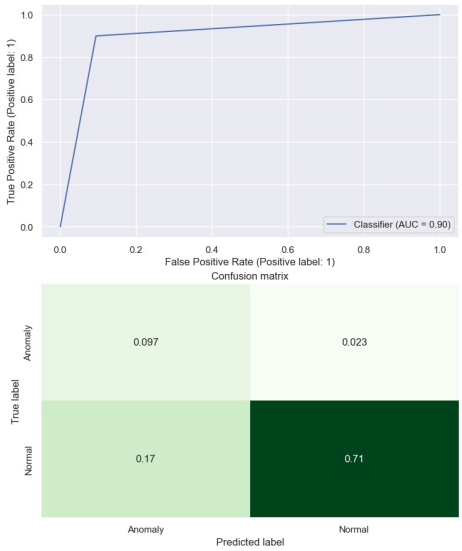


Рис.П1.75. Результаты работы алгоритма ВИЛ, размер подвыборки 7500, ИД – 750

WIF-7500 elements-1000 trees

Accuracy = 0.808

Precision = 0.808

Recall = 0.653

F1-score = 0.722

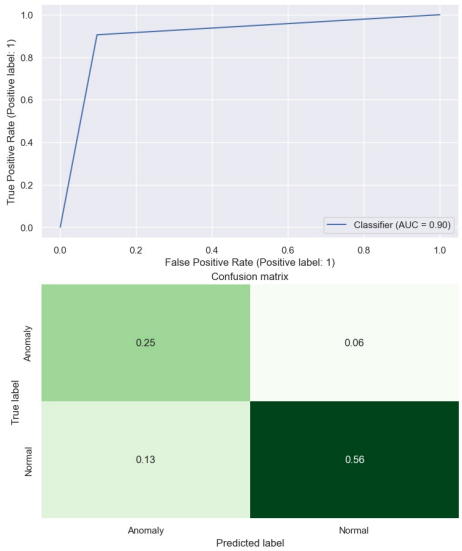


Рис.П1.76. Результаты работы алгоритма ВИЛ, размер подвыборки 7500, ИД – 1000

WIF-10000 elements-250 trees

Accuracy = 0.778

Precision = 0.778

Recall = 0.633

F1-score = 0.698

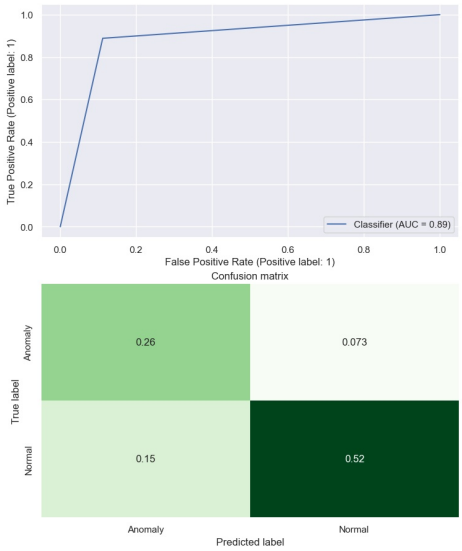


Рис.П1.77. Результаты работы алгоритма ВИЛ, размер подвыборки 10000, ИД – 250

WIF-10000 elements-500 trees

Accuracy = 0.77

Precision = 0.77

Recall = 0.44

F1-score = 0.56

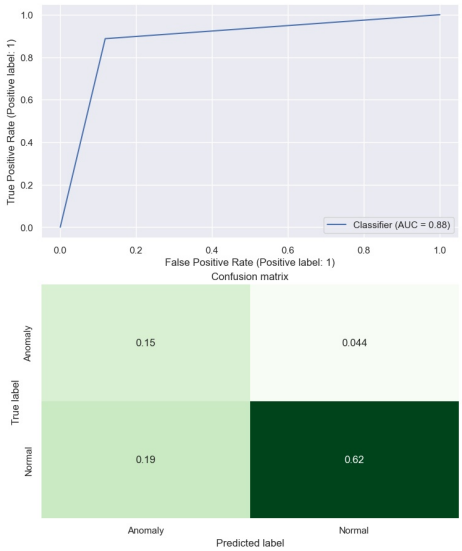


Рис.П1.78. Результаты работы алгоритма ВИЛ, размер подвыборки 10000, ИД – 500

WIF-10000 elements-750 trees

Accuracy = 0.76

Precision = 0.76

Recall = 0.587

F1-score = 0.663

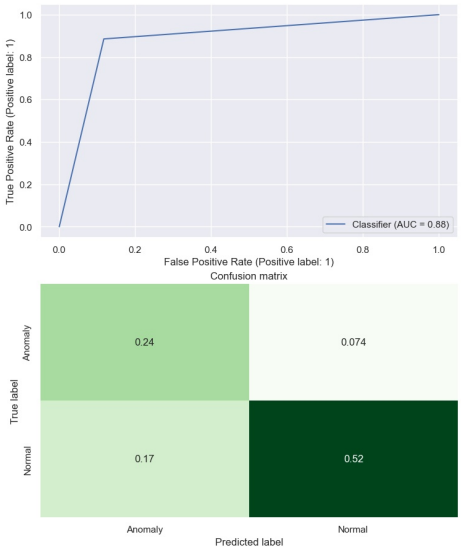


Рис.П1.79. Результаты работы алгоритма ВИЛ, размер подвыборки 10000, ИД – 750

WIF-10000 elements-1000 trees

Accuracy = 0.82

Precision = 0.82

Recall = 0.532

F1-score = 0.646

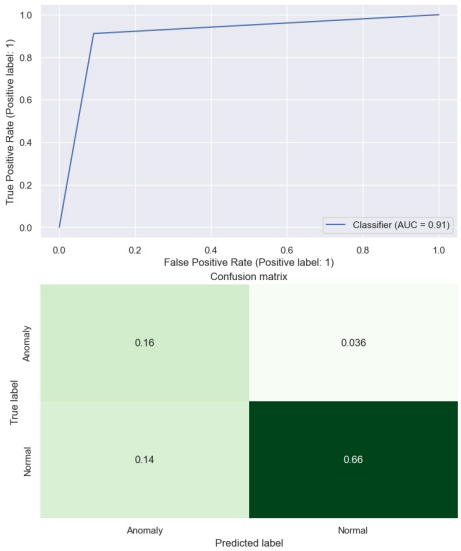


Рис.П1.80. Результаты работы алгоритма ВИЛ, размер подвыборки 10000, ИД – 1000