МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

Разработка системы для выявления фиктивных аккаунтов Open Journal System

КУРСОВАЯ РАБОТА
по дисциплине «Программная инженерия»
ЮУрГУ – 09.03.04.2024.308-066.КР

Нормоконтролер,
доктор физ.-мат. наук
_____ М.Л. Цымблер
«____»_____ 2024 г.

Научный руководитель
доктор физ.-мат. наук
_____ М.Л. Цымблер

Автор работы,
студент группы КЭ-303
_____ А.В. Толмачева

Работа защищена
с оценкой: _____
«____»_____ 2024 г.

Челябинск 2024

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
«____»_____ 2024 г.

ЗАДАНИЕ
на выполнение курсовой работы
по дисциплине «Программная инженерия»
студенту группы КЭ-303
Толмачевой Анастасии Вячеславовне,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы**
   Разработка системы для выявления фиктивных аккаунтов Open Journal System.
2. **Срок сдачи студентом законченной работы:** 31.05.2024.
3. **Исходные данные к работе**

3.1. Open Journal Systems. [Электронный ресурс] URL: https://openjournalsystems.com/
3.2. Кластеризация. [Электронный ресурс] URL: https://scikit-learn.ru/clustering/
3.3. Han J., Kamber M., Pei Jian. Data Mining: concepts and techniques. // 3rd ed. – Elsevier Inc, 2011. – 703 p.
3.4. Сорокин А.Б., Железняк Л.М. Технологии обучения: кластеризация и классификация: Учебное пособие – М.: РТУ МИРЭА, 2021. – 49 с.
3.5. Rokach L., Maimon O. Clustering Methods // The Data Mining and Knowledge Discovery Handbook, 2005. – 321-352 pp. DOI: 10.1007/0-387-25465-X_15.

4. **Перечень подлежащих разработке вопросов**

4.1. Анализ предметной области и литературы по теме работы.
4.2. Проектирование интерфейса программной системы и модульной структуры приложения.
4.3. Реализация программной системы, выявляющей фиктивные аккаунты с помощью алгоритмов машинного обучения.
4.4. Подготовка набора тестов и тестирование программной системы.

5. **Дата выдачи задания:** «____»_____ 2024 г.

Научный руководитель М.Л. Цымблер

Задание принял к исполнению А.В. Толмачева

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 4
1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6
1.1. Предметная область 6
1.2. Обзор аналогичных проектов 7
1.3. Обзор методов для поиска аномалий 9
1.4. Обзор методов классификации 14
1.5. Метрики 16
1.6. Наборы данных 17
2. ПРОЕКТИРОВАНИЕ 19
2.1. Варианты использования системы 19
2.2. Требования к системе 19
2.3. Архитектура приложения 20
2.4. Графический интерфейс 23
2.5. Базы данных 26
3. РЕАЛИЗАЦИЯ 27
3.1. Программные средства реализации 27
3.2. Реализация компонентов приложения 27
3.3. Реализация пользовательского интерфейса 31
3.4. Реализация обучения моделей 36
4. ТЕСТИРОВАНИЕ 38
4.1. Функциональное тестирование 38
4.2. Сравнение алгоритмов 39
ЗАКЛЮЧЕНИЕ 44
ЛИТЕРАТУРА 45
ПРИЛОЖЕНИЯ 48
Приложение А. Примеры размеченных аккаунтов 48
Приложение Б. Спецификация вариантов использования 50

ВВЕДЕНИЕ

Актуальность темы

Интернет в наши дни развиваются быстро. Это очень важно для маркетинговых компаний и знаменитостей, которые желают увеличить количество покупателей и фанатов; для обычных пользователей, которые быстрее узнают новости и находят новых знакомых; для исследователей, которые делятся последними открытиями и актуальной информацией. Социальные сети делают нашу жизнь лучше, но есть некоторые аспекты, которые требуют внимания.

Вместе с расширением онлайн-сообществ возникает проблема поддельных аккаунтов. «Фейки» могут быть созданы с разными целями: получить коммерческую выгоду, дискредитировать настоящего пользователя, заполучить личную информацию, осветить вредный контент и так далее [2]. Иногда их тяжело отличить от аккаунта живого человека, а массовость проблемы может набирать социально-опасный характер: например, «боты», которые сыграли роль в выборах президента [14].

Поддельные аккаунты могут встречаться и в среде научных публикаций. Open Journal System не защищена от появления в ней ненастоящих пользователей, что может вызывать различные трудности при поиске научных статей и обмене исследованиями. Существующие решения не рассматривают проблему распознавания фиктивных аккаунтов в Open Journal Systems.

Технический прогресс, особенно в области машинного обучения и анализа данных, предоставляет инструменты для обнаружения «фейков». Алгоритмы машинного обучения могут быть применены для выявления аномального поведения, классификации аккаунтов как фиктивных [13].

Таким образом, разработка системы для выявления фиктивных аккаунтов в Open Journal System становится актуальной и востребованной, учитывая масштаб проблемы появления «ботов» в Интернете и технический прогресс в области искусственного интеллекта и машинного обучения.

Цель и задачи

Целью курсовой работы является реализация программной системы, выявляющей фиктивные аккаунты в Open Journal Systems. Для достижения

поставленной цели необходимо решить следующие задачи:

1. Провести анализ предметной области и литературы по теме работы.
2. Спроектировать интерфейс программной системы и модульной структуры приложения.
3. Реализовать систему, выявляющую фиктивные аккаунты с помощью алгоритмов машинного обучения.
4. Подготовить набор тестов, выполнить тестирование программной системы.

Структура и содержание работы

Курсовая работа состоит из введения, четырех разделов, заключения и списка литературы. Объем работы – 51 страница, объем библиографического списка – 25 наименований.

В первом разделе, «Анализ предметной области», содержится описание Open Journal System, обзор аналогичных проектов, методов поиска аномалий и классификации, определение метрик, описание наборов данных для обучения и тестирования.

Во втором разделе, «Проектирование» представлены варианты использования системы, функциональные и нефункциональные требования, архитектура системы, графический интерфейс и описание баз данных.

В третьем разделе, «Реализация», приведены программные средства реализации, а также описаны созданные модули системы.

В четвертом разделе, «Тестирование», представлено функциональное тестирование и сравнение алгоритмов.

В приложении А содержатся примеры размеченных аккаунтов.

В приложении Б представлена спецификация вариантов использования системы.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Предметная область

Open Journal System (OJS) – это программное обеспечение, которое позволяет публиковать статьи и организовать рабочий процесс издательства. На ее основе разработаны многие порталы, работают институты, научные центры и журналы в разных странах мира (интерфейс OPS переведен более чем на 30 языков). Платформа обладает модульной структурой и имеет возможность подключения плагинов.

OJS может быть рассмотрена как электронная библиотека: программа обеспечивает доступ к контенту, поиск по нему (автора, ключевые слова, названия статей, год выпуска и так далее).

Аккаунт в Open Journal Systems представляет собой учетную запись пользователя, которая позволяет взаимодействовать с системой в качестве автора, рецензента, читателя или редактора в научных журналах, использующих OJS для управления процессом публикации.

Авторы могут создавать и отправлять свои научные статьи на публикацию в журнал. Их аккаунт позволяет им отслеживать статус статей, вносить изменения в данные и взаимодействовать с рецензентами.

Рецензенты имеют доступ к предоставленным им статьям. Их аккаунт дает им возможность отправлять свои оценки, взаимодействовать с редакцией и следить за обновлениями в процессе рецензирования.

Читатели могут создавать свои учетные записи для отслеживания интересующих статей, подписываться на уведомления о новых публикациях, а также участвовать в дискуссиях в комментариях к статьям.

Редакторы имеют полный доступ к управлению процессом публикации в журнале. Это включает в себя принятие и отклонение статей, управление рецензиями, управление составом редакционной коллегии и другие аспекты редакционной работы.

Администраторы имеют права доступа ко всем функциям системы и могут управлять пользователями, настройками, архивами и другими аспектами системы.

Поля, доступные в учетной записи пользователя OJS, могут немного различаться в зависимости от версии системы и настроек конкретного жур-

нала, но обычно они содержат следующие базовые характеристики: имя пользователя, пароль, электронная почта, имя, фамилия, страна, аффилиация (принадлежность пользователя к организации, университету или другому учреждению), язык, ссылка, телефон, почтовый адрес.

1.2. Обзор аналогичных проектов

Нахождение фиктивных аккаунтов в Open Journal System является темой данной работы. Для выбора наилучшего подхода к решению задач и достижения цели были рассмотрены некоторые статьи на тему выявления «фейков» и их влияния на аккаунты реальных людей. Далее представлены исследования, которые были обозрены в ходе работы.

Авторы статьи [13] раскрывают проблему поддельных страниц. Их количество растет вместе с увеличением числа активных пользователей. Фиктивные профили на сайтах социальных сетей создают ненастоящие новости и распространяют нежелательные материалы, содержащие спам-ссылки. В этой статье приводится контролируемый алгоритм машинного обучения, называемый машиной опорных векторов, который используется вместе с методом случайного леса. Эту концепцию можно применить для идентификации большого количества учетных записей, которые невозможно проверить вручную. Данная модель сравнивается с другими методами идентификации, и результаты показывают, что предложенный авторами алгоритм работает с большей точностью.

Статья [9] посвящена выявлению поддельных профилей в социальных сетях. Для их обнаружения было предложено множество алгоритмов и методов, и авторы этой работы оценивают точность использования дерева решений и наивного Байесовского классификатора для разделения профилей пользователей на поддельные и подлинные.

В работе [25] исследователи используют модель, которая определяет подлинность новостных статей, публикуемых в Twitter, на основе подлинности и предвзятости пользователей, которые взаимодействуют с этими статьями. Предлагаемая модель включает в себя идею оценки соотношения подписчиков, возраст аккаунта и т.д. Для анализа изображений предлагается использовать нейронную сеть. Подход исследователей помогает

обнаруживать поддельные изображения с точностью около 76%.

В исследовании [12] проведен анализ 62 миллионов общедоступных профилей пользователей социальной сети Twitter, и разработана стратегия идентификации поддельных профилей. Используя алгоритм сопоставления шаблонов имен, анализ времени обновления записей и даты создания профилей, были выявлены фиктивные аккаунты.

В статье [5] описывается алгоритм для обнаружения поддельных учетных записей в социальной сети. Авторы работы собирают некоторые из первых аналитических данных о том, как новые (поддельные и реальные) аккаунты пытаются завести друзей, ориентируясь на их первые запросы о дружбе после регистрации в социальной сети.

В работе [17] авторы представляют метод обнаружения, основанный на сходстве пользователей, с учетом их сетевых коммуникаций. На первом этапе измеряются такие параметры, как общие соседи, ребра графа общих соседей, косинус и коэффициент подобия Жаккарда [20], которые вычисляются на основе матрицы смежности соответствующего графа социальной сети. На следующем шаге, чтобы уменьшить сложность данных, к каждой вычисленной матрице подобия применяется компонентный анализ для получения набора информативных признаков. Затем с помощью метода локтя выбирается набор высокоинформативных собственных векторов. Извлеченные функции используются для обучения алгоритма классификации.

В статье [23] представлено исследование поддельных аккаунтов в социальных сетях с использованием искусственной нейронной сети для их идентификации. Специально разработанное и внедренное приложение было использовано для выявления специфических особенностей поддельных аккаунтов и изучения принципов и причин их генерации. На основе изучения 500 реальных и 500 поддельных аккаунтов социальносй сети ВКонтакте был сделан ряд выводов об особенностях поддельных аккаунтов. Проведенное исследование позволило расширить список критериев идентификации поддельных аккаунтов набором шаблонов.

В статье [6] авторы нацеливаются на модель фиктивного аккаунта, который может не только автоматически публиковать сообщения или ком-

ментарии, но и рассылать рекламный спам или распространять ложную информацию. В этом исследовании был предложен метод обнаружения «фейков», основанный на шаблоне активности пользователя, с использованием машинного обучения, чтобы предсказать, контролируется ли учетная запись поддельным пользователем.

Работа [14] представляет результаты экспериментальной визуализации более 200 000 постов из подтвержденных поддельных аккаунтов одной социальной сети. Авторы анализируют учетные записи пользователей, изучая их имена пользователей, описания или биографии, записи, их частоту и содержание. Исследователи обнаружили, что фиктивные аккаунты узнаваемы благодаря политическим и религиозным убеждениям.

Авторы статьи [11] рассматривают влияние фиктивных аккаунтов на аккаунты людей. Цель работы – представить новую модель распространения информации. Исследователи вводят два типа пользователей с разным уровнем «зараженности»: пользователи, «инфицированные» человеком, и пользователи, «зараженные» учетными записями ботов. Было измерено влияние поддельных аккаунтов на скорость распространения лжи среди записей людей. Результаты эксперимента показывали, что точность предложенной модели превосходит классическую при моделировании процесса распространения слухов. Был сделан вывод, что «фейки» ускоряют процесс распространения невподтвержденной информации, поскольку они воздействуют на многих людей за короткое время.

1.3. Обзор методов для поиска аномалий

Поиск фиктивных аккаунтов можно рассмотреть как задачу обнаружения аномалий. Ниже представлены алгоритмы, которые можно использовать для этой решения проблемы.

Изоляционный лес (isolation forest) [15] представляет собой алгоритм обнаружения аномалий, который строит ансамбль изолирующих деревьев решений.

Дерево изоляции. Пусть T – узел дерева изоляции. T может быть либо внешним узлом без дочерних узлов, либо внутренним узлом с одним тестом и ровно двумя дочерними узлами (T_l, T_r). Тест состоит из атрибута

q и значения разделения p, такого, что тест $q < p$ разделяет точки данных на T_l и T_r.

Дана выборка данных $X = \{x_1, \ldots, x_n\}$ из n экземпляров из d-мерного распределения. Для построения дерева изоляции мы рекурсивно делим X, случайно выбирая атрибут q и значение разделения p, пока дерево не достигнет предельной высоты, либо $|X| = 1$, либо все данные в X будут иметь одинаковые значения. Дерево изоляции – это правильное бинарное дерево, где каждый узел в дереве имеет ровно ноль или два дочерних узла.

Одним из способов обнаружения аномалий с помощью дерева изоляции является сортировка точек данных по их длинам пути или оценкам аномалий. Аномалии – это точки, которые занимают верхние позиции в списке. Длина пути $h(x)$ точки x измеряется количеством ветвей, которые x проходит в дереве изоляции от корневого узла до завершения обхода во внешнем узле.

Работа алгоритма изоляционного леса:

1. Случайное разбиение. Дерево случайным образом выбирает атрибут и значение разделения данных. Это создает разделение, которое помогает выделить аномалии.

2. Рекурсивная изоляция. Дерево повторяет процесс, разделяя данные на более мелкие группы. Цель состоит в том, чтобы изолировать аномалии от обычных данных.

3. Идентификация аномалий. Аномалии идентифицируются как точки данных, для выделения которых требуется меньшее количество разбиений.

4. Определение изолирующего пути. Определяется количество разбиений, необходимых для изоляции точки данных, служит мерой её аномальности.

5. Ансамбль деревьев. Алгоритм создает несколько деревьев изоляции, независимых друг от друга, образующих ансамбль, который коллективно оценивает аномалии.

6. Расчет оценки различий. Вычисляется среднее расстояние разделения между всеми деревьями для каждой точки данных, что дает оценку

аномалии.

7. Классификация. Используются предопределенные пороги для нормальных и аномальных точек данных. Объекты с высокими оценками помечаются как аномалии.

Иерархическая кластеризация (hierarchical clustering) – это алгоритм группировки данных, который строит иерархию кластеров. Существует два основных подхода: агломеративный и дивизивный.

Агломеративная кластеризация. Каждая точка рассматривается как отдельный кластер. На каждом шаге ближайшие кластеры объединяются в новый кластер (на основании расстояния между ними). Процесс повторяется до тех пор, пока все точки не объединятся в один кластер.

Дивизивная кластеризация. Вся выборка рассматривается как один кластер. На каждом шаге один из кластеров разбивается на более мелкие кластеры (на основании мер несходства между подгруппами). Процесс повторяется до тех пор, пока каждая точка не станет отдельным кластером.

В данной работе рассматривалась агломеративная кластеризация [1]. Ее алгоритм представлен на диаграмме деятельности (рисунок 1).

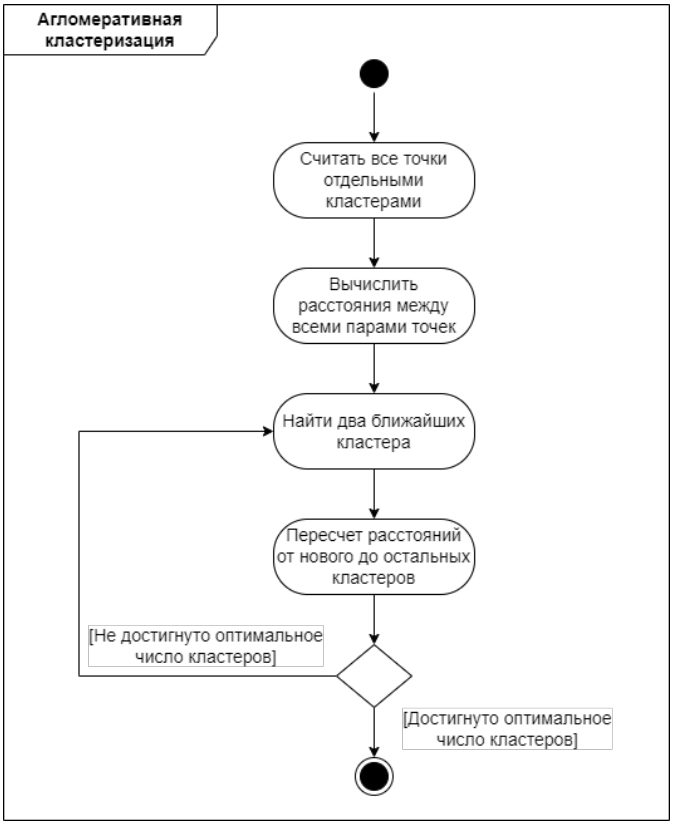

Рисунок 1 – Диаграмма деятельности агломеративной кластеризации

Для объединения кластеров необходимо вычситать расстояния между ними. В данной работе использовалось евклидово расстояние. Оно вычисляется как расстояние между центрами масс кластеров. Также нужно выбрать метод, каким образом данные будут объединяться кластеры. Наиболее распространенные методы: одиночной связи, полной связи, средней связи, Уорда.

Метод одиночной связи (single linkage): расстояние между кластерами равно минимальному расстоянию между точками из разных кластеров.

$$d_{\text{min}}(C_i, C_j) = \min_{x \in C_i, y \in C_j} \rho(x, y)$$

Метод полной связи (complete linkage): расстояние между кластерами равно максимальному расстоянию между точками из разных кластеров.

$$d_{\text{max}}(C_i, C_j) = \max_{x \in C_i, y \in C_j} \rho(x, y)$$

Метод средней связи (average linkage): расстояние между кластерами равно среднему расстоянию между всеми парами точек из разных кластеров.

$$d_{\text{avg}}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{x \in C_i} \sum_{y \in C_j} \rho(x, y)$$

Метод Уорда (Ward's linkage): расстояние между кластерами равно приросту суммы квадратов расстояний от точек до центроидов кластеров при объединении этих кластеров. Этот метод стремится минимизировать внутрикластерную дисперсию.

$$d_{\text{ward}}(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} \rho^2(\bar{x}_i, \bar{x}_j)$$

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [10] – это алгоритм кластеризации, который определяет кластеры на основе плотности данных в пространстве. Он может обнаруживать группы произвольной формы и выделять точки, не принадлежащие ни одной из них (шум).

Для нахождения кластера DBSCAN начинает с произвольной точки p и извлекает все точки, достижимые по плотности из p относительно *Eps* и *MinPts*. Если p является ядром кластера, этот процесс создает кластер отно-

сительно *Eps* и *MinPts*. Если *p* является граничной точкой, и нет объектов, достижимых по плотности, и DBSCAN переходит к следующей записи в базе данных.

Для алгоритма DBSCAN необходимо указывать параметры *Eps* и *MinPts*. Первый параметр определяет окрестности вокруг точек данных: если расстояние между двумя точками меньше или равно *Eps*, то они считаются соседними. Второй представляет собой минимальное количество соседей (точек данных) в радиусе.

Работа алгоритма DBSCAN представлена на диаграмме деятельности (рисунок 2).

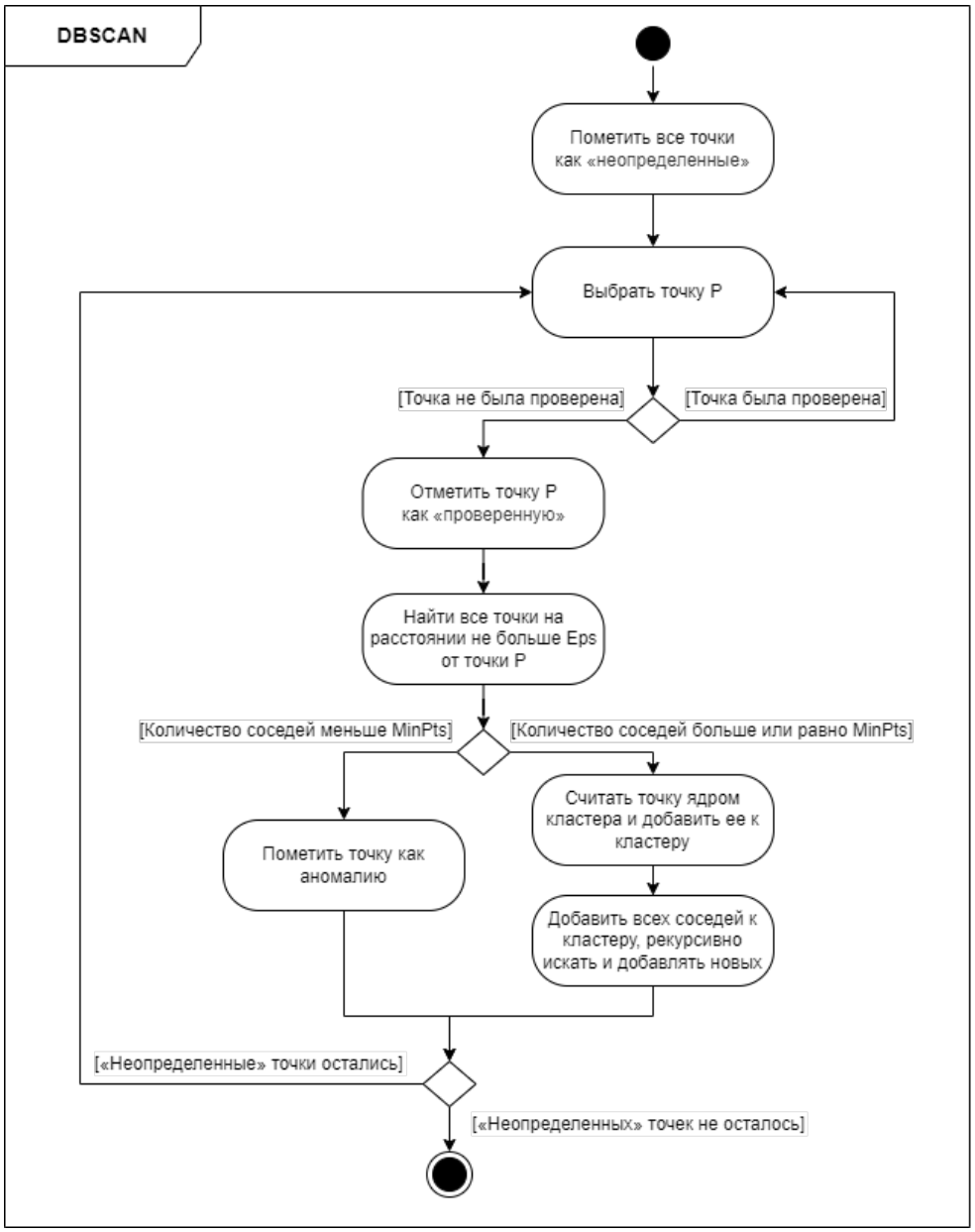

Рисунок 2 – Диаграмма деятельности алгоритма DBSCAN

1.4. Обзор методов классификации

Поиск фиктивных аккаунтов можно также рассматривать как задачу классификации, некоторые методы представлены ниже.

Дерево решений (decision tree) [4] – это алгоритм машинного обучения для решения задач классификации.

Узлы дерева представляют собой проверки на признаки. Ветви – переходы от одного узла к другому. Листья – это узлы, представляющие собой окончательные решения по классификации, то есть они содержат подмножества, которые удовлетворяют всем парвилам ветви. Корневой узел – верхний, начальный узел дерева решений, содержащий весь набор данных. Есть несколько критериев разбиения, наиболее популярные это индекс Джини и энтропия.

Индекс Джини измеряет вероятность неправильной классификации при выборе случайного элемента из обучающей выборки. Он вычисляется по формуле:

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Индекс Джини может изменяться от 0 до $1 - \frac{1}{n}$. Он равен нулю, когда все объекты результирующего множества принаджелат одному классу, и $1 - \frac{1}{n}$, когда объекты равномерно распределены по всем классам. Наиболее желательная ситуация, когда индекс Джини минимальный.

Энтропия измеряет неопределенность данных в узле. Она представляет количество информации, которое необходимо для описания состояния системы, и вычисляется по следующей формуле:

$$Entropy = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i$$

Энтропия модет изменяться от 0 до $\log_2 n$. Она равна нулю, когда все объекты принадлежат одному классу (что является наиболее желательной ситуацией), и $\log_2 n$, когда объекты равномерно распределены по классам.

Алгоритм дерева решений работает следующим образом:

1. Выбор корневого узла. Весь набор данных используется как начальный узел.

2. Выбор признака разбиения. Для каждого признака рассчитывается значение выбранного критерия. Тот признак, что дает наиболее желательный результат (в обоих случаях лучшее значение – нуль) и выбирается для разбиения.

3. Разбиение узла. Узел делится на подмножетсва, основанные на значении выбранного признака.

4. Повторение для дочерних узлов. Для каждого последующего узла процесс повторяется рекурсивно, пока не будет достигнуто одно из условий остановки.

Условия остановки алгоритма:

1. Достигнута определенная глубина дерева.

2. Узел не делится, если в его множество меньше определенного числа объектов.

3. Следующее разбиение приведет к тому, что каждый лист не будет содержать заданного минимума объектов.

4. Разбиение не выполняется, если оно не приводит к уменьшению «нечистоты» превышающему или равному заданному значению.

Случайный лес (random forest) [3] – это ансамблевый метод машинного обучения, который может использоваться для задач классификации. Он состоит из множества деревьев решений, каждое из которых обучается на случайной подвыборке исходных данных, что уменьшает корреляцию между их ошибками.

Алгоритм работы случайного леса:

1. Для каждого дерева случайного леса формируется подвыборка данных с возвращением в исходный набор данных (некоторые объекты могут быть выбраны несколько раз, а некоторые не попасть в подвыборку).

2. На вершине деревьев выбирается случайное подмножество признаков. Лучший из них используется для разбиения узла. Дерево строится до выполнения критерия остановки.

3. Когда все деревья обучены, их предсказания комбинируются для получения итогового предсказания случайного леса: используется метод голосования большинства.

1.5. Метрики

Для оценки качества работы алгоритма будут использоваться метрики accuracy (аккуратность), precision (точность) и recall (полнота), F-measure (F-мера). Первая показывает долю верно классифицированных объектов, вторая – долю объектов, которые модель классифицировала как положительные, и которые действительно являются положительными, третья – долю объектов положительного класса, которые модель определила правильно, четвертая – это комбинированная метрика, объединяющая точность и полноту в единственное число. Полнота демонстрирует способность алгоритма обнаруживать фиктивные аккаунты, а точность – способность отличать их от других.

Данные метрики можно вычислить по следующим формулам:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F\text{-}measure = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Термины TP, TN, FP, FN используются в контексте матрицы ошибок (таблица 1), которая является инструментом для оценки производительности моделей.

Таблица 1 – Матрица ошибок

	Positive	Negative
True	True Positives	True Negatives
False	False Positives	False Negatives

TP (True Positives) – количество верно предсказанных положительных примеров. В данной работе это будут те аккаунты, которые являются фиктивными, и алгоритм классифицировал их как аномальные.

TN (True Negatives) – количество верно предсказанных отрицатель-

ных примеров. В данной работе это будут те аккаунты, которые являются реальными, и алгоритм не классифицировал их как аномальные.

FP (False Positives) – количество ложно положительных примеров. В данной работе это будут те аккаунты, которые являются реальными, но алгоритм классифицировал их как аномальные.

FN (False Negatives) – количество ложно отрицательных примеров. В данной работе это будут те аккаунты, которые являются фиктивными, но алгоритм не классифицировал их как аномальные.

1.6. Наборы данных

Обучение. Для обучения моделей был применен датасет с реальными данными аккаунов, который имеет следующие столбцы: user_id, username, password, email, url, phone, mailing_address, billing_address, country, locales, date_last_email, date_registered, date_validated, date_last_login, must_change_password, auth_id, auth_str, disabled, disabled_reason, inline_help, gossip.

Была произведена его разметка с добавлением колонки is_fake, в которой 1 показывает, что аккаунт фиктивный, а 0 – настоящий. Разметка происходила по следующим признакам:

1. Проверка email на валидность (просмотр синтаксиса и возможности доставки на домен). Если почта не валидна, аккаунт считался фиктивным (например, аккаунт с email admin@cr6ptobrowser.site).
2. Проверка работы ссылки, указанной в аккаунте. Если ссылка не работала, то аккаунт считался фиктивным.
3. Проверка содержимого сайта, указанного в ссылке аккаунта. Если там находится что-либо, не связанное с научными публикациями, аккаунт считался фиктивным (например, аккаунт с ссылкой на казино).
4. Просмотр ближайших строк по времени регистрации – если имена пользователей, email или billing_addres различаются на 1 – 5 символов, то такие аккаунты считались фиктивными. Также если в характеристиках 2 и более аккаунтов встречается одинаковая часть (например, dtyekzneeplomo и tkbpfdtnfneeplomo).
5. Проверка символов в имени пользователя, email и

mailing_address: представляют они осмысленный текст или произвольный набор символов. В случаях, когда почта была валидна, но состояла из рандомного набора, проверялись дата последнего входа в аккаунт и дата регистрации. Когда они были в диапазоне суток друг от друга, аккаунт считался фиктивным (например, аккаунт с username ramonjap, email bvbxzroi@dimail.xyz и mailing_addressvljufcpn@dimail.xyz).

После разметки было выяснено, что в данном датасете из 540 записей 204 (38%) являются фиктивными, и 336 (62%) – настоящими. Примеры «фейков» и реальных аккаунтов рассмотрены в Приложении А.

Тестирование. Для тестирования работы системы был создан и размечен отдельный датасет, не пересекающийся с обучающим, содержащий 70 записей: 35 фиктивных и 35 настоящих аккаунтов. Он использовался для проведения экспериментов с алгоритмами и был разбит на несколько таблиц меньшего размера (путем удаления некоторых строк), в которых было различное соотношение фиктивных и реальных аккаунтов: 10% и 90%, 20% и 80%, 30% и 70%, 40% и 60%.

2. ПРОЕКТИРОВАНИЕ

2.1. Варианты использования системы

Описание способов взаимодействия с системой, а также кто с ней может работать, отображено на рисунке 3.

Рисунок 3 – Диаграмма вариантов использования

Единственный актер, который может взаимодействовать с системой распознавания фиктивных аккаунтов в Open Journal System – это исследователь. Он может выполнять следующие действия:

1. Загрузить данные. Исследователь может добавить датасет в базу данных системы.
2. Редактировать данные. Исследователь может отредактировать данные, которые загрузил.
3. Выполнить настройку. Исследователь может выполнить настройку того, какой будет выполняться алгоритм, с какими параметрами, а также в каком формате будут выводиться результаты.
4. Найти фиктивные аккаунты. Исследователь может найти «фейки» с помощью выбранного алгоритма.

2.2. Требования к системе

Функциональные требования

Функциональные требования – это спецификация того, каким обра-

зом должно вести себя разрабатываемое программное обеспечение. Они определяют, какие конкретные функции и возможности должны быть включены в приложение, чтобы оно соответствовало ожиданиям пользователей. Эти требования описывают конкретные действия, которые система должна выполнять, и каким образом пользователь взаимодействует с приложением для достижения определенных целей.

Функциональные требования к системе:

1. Система должна предоставлять пользователю возможность загружать данные для исследования.
2. Система должна предоставлять возможность изменять загруженные данные.
3. Система должна предоставлять пользователю возможность выполнить настройку ее работы: выбрать алгоритм, изменить его характеристики, добавить вывод результатов в графической форме.
4. Система должна находить фиктивные аккаунты.

Нефункциональные требования

Ненфункциональные требования определяют свойства и характеристики, которыми должна обладать система. Эти требования касаются аспектов, не связанных напрямую с конкретными функциями, а определяют общие качественные аспекты, которые необходимы для функционирования системы.

Нефункциональные требования к системе:

1. Система должна быть написана на языке программирования Python.
2. Система должна работать с файлами формата csv.

2.3. Архитектура приложения

В данном разделе представлена архитектура системы. Она представляет собой диаграмму компонентов, которая отображена на рисунке 4.

Система для нахождения фиктивных аккаунтов содержит модули, представленные ниже.

Головной модуль отвечает за запуск остальных окон приложения, а также выводит подсказку для пользователя о том, как работать с системой.

Загрузка данных отвечает за работу с файлами, которые Исследова-

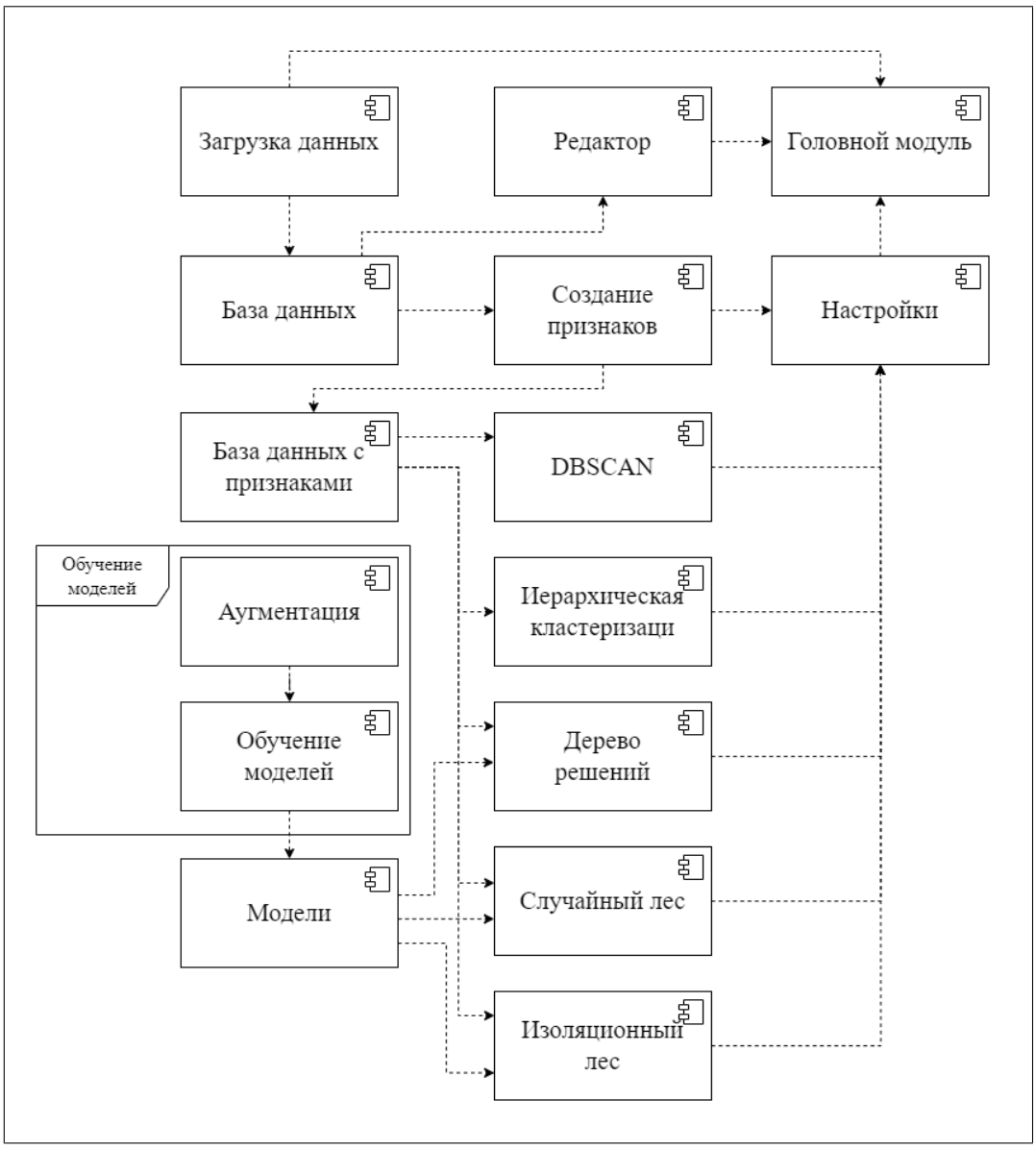

Рисунок 4 – Архитектура системы

тель загружает в систему: открытие окна для выбора файла формата csv, добавление датасета в базу данных приложения.

Редактор предоставляет функционал, который позволяет Исследователю изменять загруженные таблицы: модифицировать значения ячеек, добавлять и удалять столбцы и строки.

Настройки позволяют Исследователю выбрать параметры алгоритмов, отображения результатов, сохранения файлов, а также найти «фейки». Данный компонент передает значения в алгоритмы, вызывает модуль создания признаков.

Создание признаков формирует новую таблицу, на основе выбранной пользователем. Он вычисляет необходимые для поиска фиктивных аккаун-

тов значения и загружает их в базу данных приложения с признаками.

DBSCAN содержит алгоритм DBSCAN. Он получает значения из базы данных с признаками, объединяет точки в кластеры, определяет аномалии и сохраняет их в csv-файл, вычисляет метрики на основе размеченных данных и создает визуализацию результатов.

Иерархическая кластеризация содержит алгоритм агломеративной кластеризации и подбор количества оптимальных кластеров для нее. Модуль получает значения из базы данных с признаками, выполняет кластеризацию, находит аномальные группы и выводит их в csv-файл, вычисляет метрики и создает дендрограмму.

Дерево решений содержит алгоритм дерева решений. Модуль загружает сохраненную модель дерева решений, классифицирует данные и сохраняет фиктивные аккаунты в csv-файл, вычисляет метрики и визуализирует дерево.

Случайный лес содержит алгоритм случайного леса. Модуль загружает сохраненную модель случайного леса, создает предсказания на ее основе, определяет фиктивные аккаунты и сохраняет их в csv-файл, вычисляет метрики и визуализирует результаты работы.

Изоляционный лес содержит алгоритм изоляционного леса. Компонент загружает сохраненную модель изоляционного леса, создает предсказания на ее основе, определяет аномалии и сохраняет их в csv-файл, вычисляет метрики, а также визуализирует результаты.

База данных хранит загруженные пользователем датасеты, предоставляет их модулям редактирования и создания признаков.

База данных с признаками хранит таблицы со значениями, вычисленными с помощью модуля создания признаков, предоставляет их алгоритмам.

Модели – это обученные и сохраненные модели, которые используются для нахождения фиктивных аккаунтов с помощью алгоритмов изоляционного леса, дерева решений и случайного леса.

Обучение моделей представлено описанными далее компонентами.

Аугментация помогает сбалансировать набор данных, добавляя искусственно созданные записи на основе уже существующих. Результатом

работы модуля является аугментированный датасет, содержащий равное количество фиктивных и настоящих аккаунтов. Компонент добавляет новые строки в конец таблицы и сохраняет ее в csv-файл. Также предоставляет функционал для создания таблицы с признаками.

Обучение моделей – модуль для обучения и сохранения моделей трех алгоритмов: изоляционного леса, дерева решений и случайного леса.

2.4. Графический интерфейс

В данном разделе представлены макеты модулей системы. Они являются примерными и содержат в себе основной функционал.

На рисунке 5 представлен главный экран системы. На нем располагаются кнопки, по которым происходит работа с приложением. При нажатии на кнопку «Загрузить данные» открывается форма для выбора файла для загрузки, кнопка «Отредактировать данные» открывает новое окно, в котором можно выбрать таблицу и изменить ее. По кнопке «Найти фейки» открывается раздел, в котором можно ввести необходимые параметры алгоритмов и запустить их работу.

Рисунок 5 – Главный экран

Экран с редактированием данных представлен на рисунке 6. На нем располагается таблица с данными, а также кнопки, которые позволяют

взаимодействовать с ней: добавдлять и удалять строки и столбцы. Вверху экрана можно переключаться между загруженными датасетами с помощью выпадающего списка. Также на макете можно увидеть кнопку «Сохранить», которая отвечает за сохранение изменений в данных.

Рисунок 6 – Экран разметки

На рисунке 7 показан экран с настройками. В нем представлен выбор алгоритма, таблицы с данными, необходимости визуализации результатов с помощью соответствующих выпадающих списков. Также на нем расположены текстовое поле и кнопка для выбора папки, в которую сохраняются файлы, созданные при работе алгоритмов, и кнопка для нахождения фиктивных аккаунтов.

Помимо этого при выборе конкретного алгоритма будут изменяться виджеты, расположеннные в правой части экрана. Для DBSCAN появятся два текстовых поля с вводом Eps и $MinPts$, при выборе изоляционного леса, дерева решений и случайного леса (пример на рисунке 8) выпадающий список с выбором обученной модели, для иерархической кластеризации (рисунок 9) отобразится выпадающий список с методами соединения.

Рисунок 7 – Выбор параметров DBSCAN

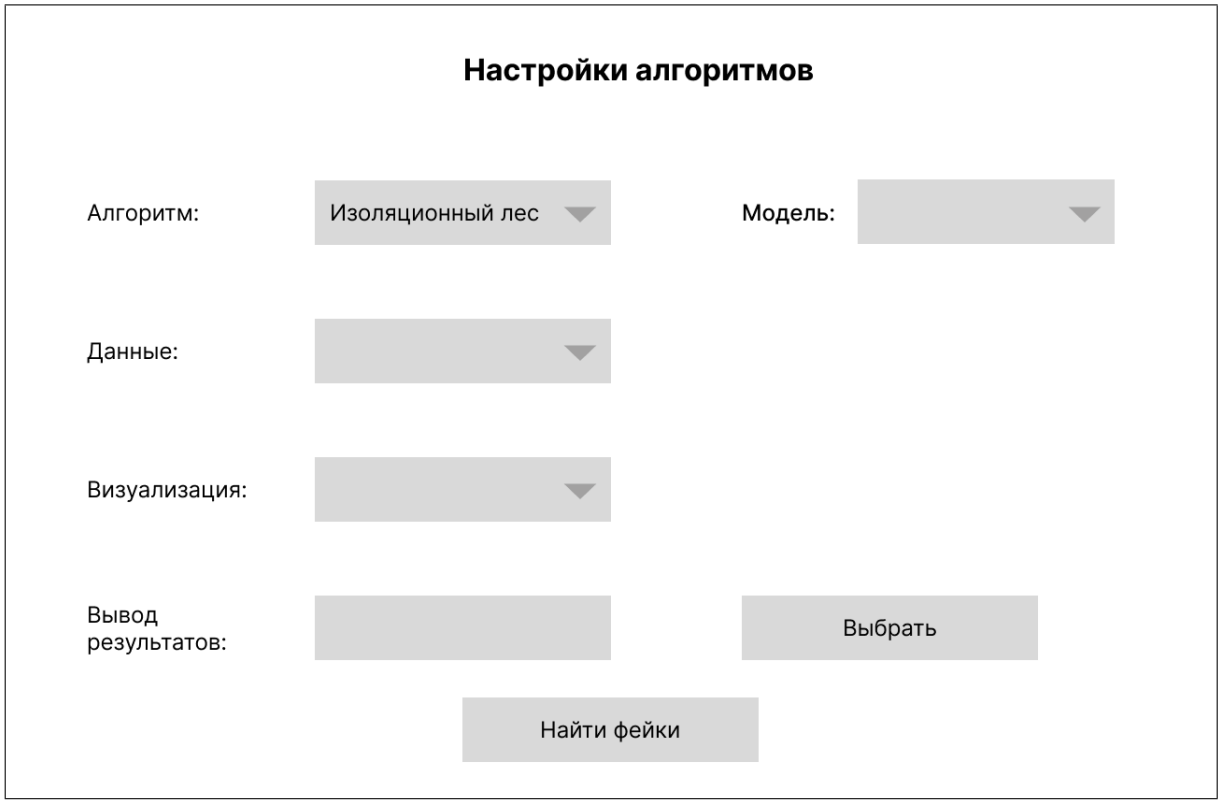

Рисунок 8 – Выбор параметров изоляционного леса

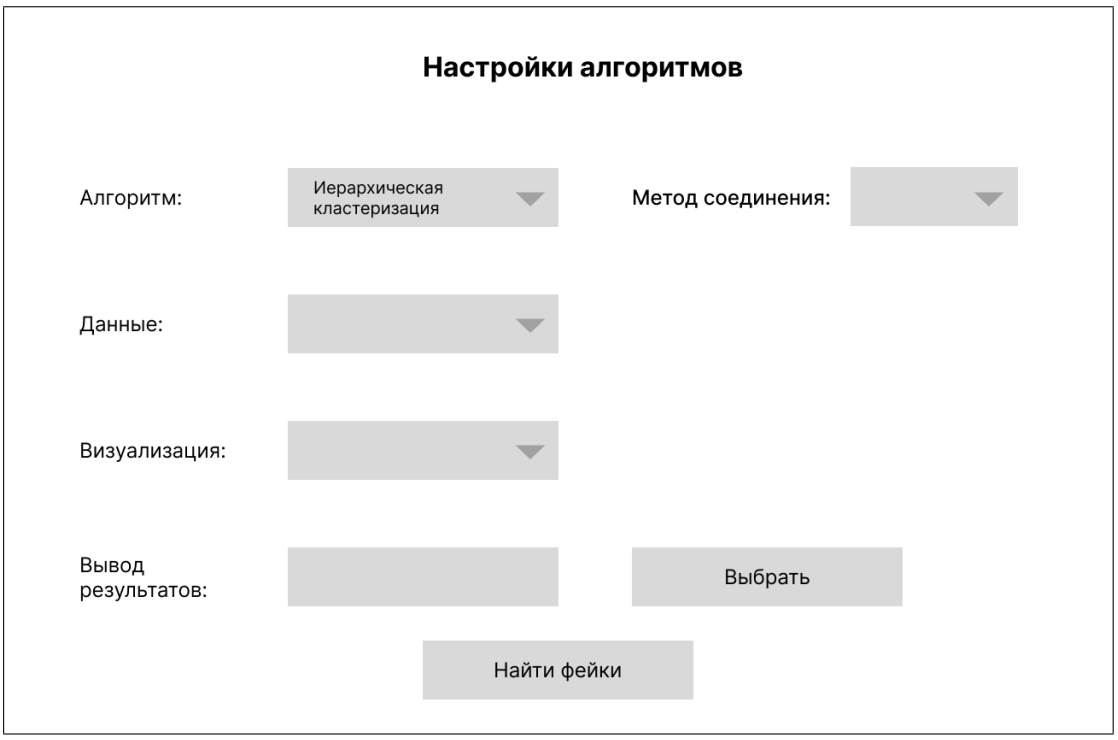

Рисунок 9 – Выбор параметров иерархической кластеризации

2.5. Базы данных

При использовании приложения создаются базы данных. Это происходит при загрузке новых файлов: создается база данных app_database.db, а также перед работой алгоритмов для нахождения фейков: создается app_database_features.db, в которую добавляются таблицы (описание представлено в таблице 2) с числовыми характеристиками аккаунтов.

Таблица 2 – Признаки аккаунтов

Столбец	Значение
user_id	Уникальный идентификатор пользователя
symb_in_name	Доля небуквенных символов в имени пользователя
symb_in_email	Доля небуквенных символов в адресе электронной почты
time_difference	Разница во времени между датой регистрации и последнего входа в аккаунт
neighbour_above	Разница во времени с ближайшим зарегистрированным аккаунтом до текущего
neighbour_below	Разница во времени с ближайшим зарегистрированным аккаунтом после текущего
text_neighbour_above	Сходство текстовых полей текущего аккаунта и предыдущего зарегистрированного аккаунта
text_neighbour_below	Сходство текстовых полей текущего аккаунта и следующего зарегистрированного аккаунта
is_fake	Признак фальшивости аккаунта (1 – если аккаунт фиктивный, 0 – если настоящий)

3. РЕАЛИЗАЦИЯ

3.1. Программные средства реализации

Система нахождения фиктивных аккаунтов была написана на языке программирования Python версии 3.9.13. Разработка велась в текстовом редакторе Visual Studio Code 1.89.0. Основные библиотеки, которые использовались при реализации:

1. Tkinter [24] – библиотека для создания графических пользовательских интерфейсов в Python.
2. Sqlite3 [22] – библиотека для работы с базой данных SQLite. Позволяет создавать, читать, изменять и удалять записи в реляционных базах данных, используя SQL-запросы.
3. Pandas [19] – библиотека для анализа и обработки данных в Python.
4. Numpy [18] – библиотека для работы с массивами и матрицами.
5. Scikit-learn [21] – библиотека для машинного обучения и анализа данных.
6. Matplotlib.pyplot [16] – подбиблиотека Matplotlib, позволяющая создавать графики и визуализировать данные.

3.2. Реализация компонентов приложения

В данном разделе приводится описание реализации компонентов системы. Полный код представлен в репозитории на github [7].

Реализация модуля загрузки

Модуль загрузки предназначен для создания таблиц в базе данных и загрузки в нее датасетов. Он имеет две соответствующие функции:

- Create_table: на вход принимает table_name – строковая переменная с названим таблицы и columns – список столбцов. Подключается к базе данных app_database, в которой создает пустую таблицу по переданному шаблону.
- Loading: позволяет выбрать csv-файл с помощью диалогового окна. Функция проверяет, существует ли таблица с таким же названием в базе данных. В случае, если она уже есть, предлагает Исследователю перезаписать данные. Она считывает название файла и заголовки столбцов,

вызывает функцию create_table и вставляет данные в новую таблицу.

Реализация модуля создания признаков

Модуль создает таблицу с характеристиками на основе данных из исходного датасета.

• Find_time_neighbours: на вход принимает dates – колонку с датами регистрации (имеющую тип series – объект библиотеки pandas), возвращает четыре списка: с разницами во времени в секундах до ближайшего зарегистрированного аккаунта до и после каждой записи, а также индексы этих соседних аккаунтов. Внутри функции используется цикл, который для каждой даты регистрации вычисляет разницу во времени с остальными датами и записывает их в два списка: положительные и отрицательные значения. После этого ищется наибольшее отрицательное число и наименьшее положительное число, которые записываются в итоговые списки, а индексы таких записей сохраняются. Если ближайших дат до или после текущей не было найдено, то записывается NULL.

• Harmonic_mean: на вход принимает два целых числа, возвращает их среднее гармоническое (число с плавающей точкой). Если сумма входных данных равна нулю, то на выход подается 0.

• Calculate_damerau_levenshtein_distance: на входе получает indexes – индексы соседних аккаунтов и data – датафрейм с исходными данными (таблица, объединение столбцов pandas.Series). Возвращает список distances. Функция с помощью цикла вычисляет расстояние Дамерау-Левенштейна [8] между именами пользователей и электронными почтами ближайших зарегистрированных аккаунтов, а после вычисляет их среднее гармоническое, которое записывает в список.

• Making_features: принимает строковую переменную с именем таблицы. Осуществляется чтение данных из нее, после чего вычисляются характеристики, описанные в разделе 2.5 в таблице 2. Данные нормализуются по столбцам, пустые значения заполняются нулями. Затем функция подключается к новой базе данных и создает таблицу в ней, куда записывает рассчитанные признаки.

Реализация DBSCAN

Модуль содержит функцию run_dbscan_algorithm, которая предна-

значена для нахождения аномалий. На вход она принимает table_name – строковая переменная с названием таблицы, eps (радиус окрестности, имеет тип числа с плавающей точкой), min_samples (минимальное количество точек в окрестности, целое число), visualization – строка с флагом в необходимости визуализации и folder – строка, содержащая путь к папке для сохранения результатов.

Работа происходит следующим образом:

1. Подготовка данных. Устанавливается соединение с базой данных признаков (app_database_features.db), выполняется запрос, чтобы извлечь данные из указанной таблицы. Колонка user_id становится индексом, создается копия датафрейма, а из изначальной таблицы удаляется столбец is_fake с метками.

2. Применение алгоритма DBSCAN с заданными параметрами eps и min_samples. Результаты кластеризации сохраняются в колонке cluster.

3. Определение аномалий. После кластеризации шум помечается значением 1 (как фиктивные аккаунты в столбце is_fake), а все остальные точки получают метку 0 (как нормальные аккаунты в столбце is_fake).

4. Сохранение аномалий и их данных в csv-файл.

5. Вычисление метрик и их вывод в csv-файл. В качестве меток используется колонка is_fake в копии датафрейма.

6. Визуализация результатов. Если Исследователь пожелал, создается визуализация. На графике отображаются точки данных, а также выводятся значения вычисленных метрик.

Реализация иерархической кластеризации

Модуль содержит функции для выполнения агломеративной кластеризации и поиска оптимального числа кластеров для нее.

• Find_optimal_clusters: принимает data – датафрейм с признаками, linkage_method – строка с методом соединения, возвращает optimal_clusters – оптимальное число кластеров для алгоритма и Z – многомерный массив, матрица расстояний для построения дендрограммы. Функция объединяет точки в кластеры, высчитывая расстояния между ними. Для определения оптимального количества кластеров был выбран порог расстояния (70% от максимального). Вычисляется количество кластеров,

у которых расстояние превышает заданный порог.

• Run_hierarchical_clustering: на вход получает строки с названием выбранной таблицы, методом соединения, флагом в необходимости визуализации и директорией для сохранения результатов. Данные извлекаются из базы данных с признаками. Колонка user_id становится индексом, создается копия датафрейма, а из изначальной таблицы удаляется столбец is_fake. Определяется оптимальное количество кластеров с использованием функции find_optimal_clusters и применяется агломеративная кластеризация. Находятся аномалии: если размер кластера меньше одной пятой максимального размера кластеров, то считаем его выделяющимся. Эти данные записываются в csv-файл. Метрики вычисляются и также сохраняются в файл. Если требуется, создается дендрограмма.

Реализация изоляционного леса

Модуль содержит функцию run_isolation_forest_algorithm, которая принимает строки с названием таблицы, именем модели изоляционного леса, флагом в необходимости визуализации и папкой для сохранения результатов. Она предназначена для нахождения аномалий в виде фиктивных аккаунтов.

Порядок действий функции run_isolation_forest_algorithm:

1. Загружается выбранная Исследователем модель.
2. Данные извлекаются из таблицы базы данных с признаками. Колонка user_id становится индексом, создается копия датафрейма, а из изначальной таблицы удаляется столбец is_fake.
3. Проводятся предсказания на основе выбранной модели, которые преобразуются к меткам 1 и 0 (как в колонке is_fake).
4. Аномалии с метками 1 сохраняются в csv-файл.
5. Вычисляются метрики, в качестве меток используется колонка is_fake скопированного датафрейма. После этого они записываются в отдельный csv-файл.
6. Если Исследователь пожелал, создается визуализация. На графике отображаются точки данных, выводятся значения вычисленных метрик.

Реализация дерева решений

Модуль содержит функцию run_decision_tree_algorithm, которая при-

нимает название таблицы, имя модели дерева решений, флаг с необходимостью визуализации и путь к папке для сохранения результатов. Она классифицирует данные и выводит информацию о найденных фиктивных аккаунтах.

Порядок действий функции run_decision_tree_algorithm:

1. Загружается выбранная Исследователем модель.
2. Данные извлекаются из таблицы базы данных с признаками. Колонка user_id становится индексом, датасет делится на признаки и целевую переменную.
3. Проводятся предсказания на основе выбранной модели.
4. Данные найденных фиктивных аккаунтов выводятся в csv-файл.
5. Вычисляются метрики для оценки работы модели. Они сохраняются в отдельный файл.
6. Если Исследователь пожелал, создается визуализация дерева решений.

Реализация случайного леса

Модуль содержит функцию run_random_forest_algorithm, которая принимает название таблицы, имя модели случайного леса, флаг с необходимостью визуализации и путь к директории для сохранения результатов. Данная функия делает то же самое, что и run_decision_tree_algorithm (классификация, вывод результатов в файл), но с одним отличием – вместо визуализации дерева решений созадется график с точками.

3.3. Реализация пользовательского интерфейса

Реализация пользовательского интерфейса проводилась по разработанным макетам. Было создано три основных окна: главное меню, окно редактирования данных и окно настроек алгоритмов. Для реализации использовалась библиотека tkinter [24].

На рисунке 10 изображено главное окно приложения. На нем представлено меню с тремя кнопками: «Загрузить данные», «Отредактировать данные», «Найти фейки». При нажатии на первую вызывается функция loading из модуля загрузки данных, появляется диалоговое окно для выбора файла. По кнопке «Отредактировать данные» открывается экран, ко-

торый позволяет изменять таблицы. Кнопка «Найти фейки» отвечает за открытие окна с настройками алгоритмов и поиском фиктивных аккаунтов. В правом нижнем углу главного окна расположена кнопка с вопросительным знаком, при нажатии на которую появляется окно с информацией о системе (рисунок 11).

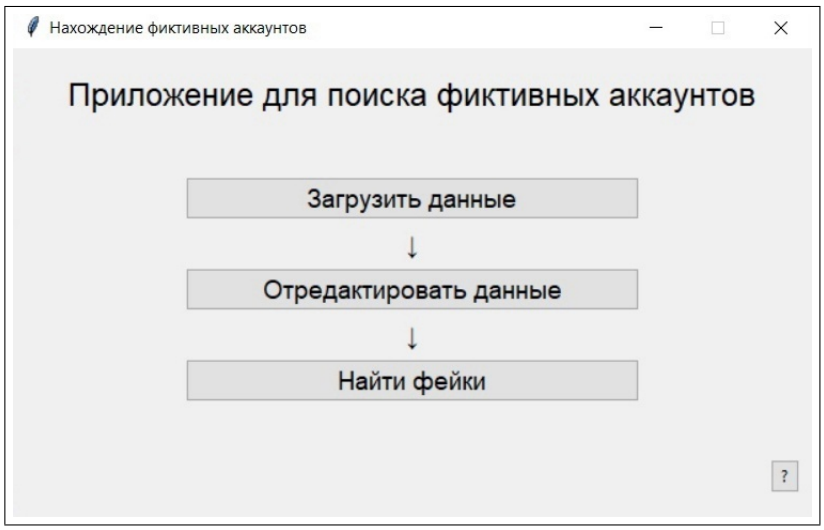

Рисунок 10 – Главное окно приложения

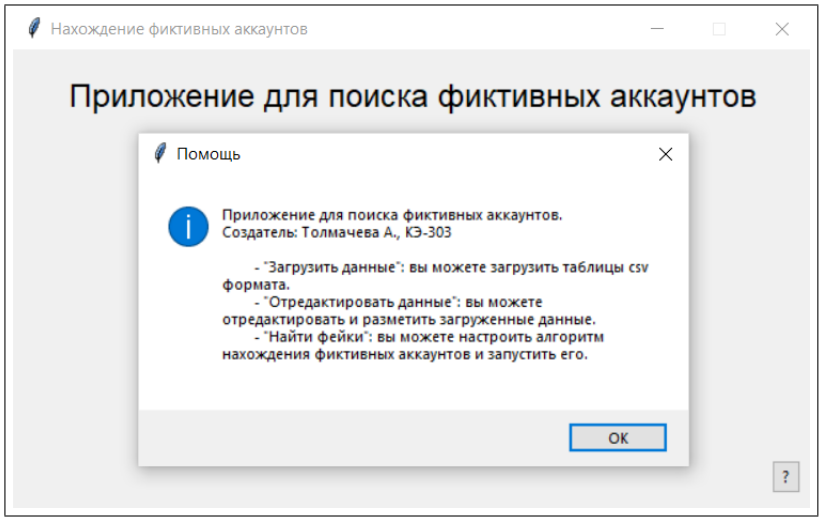

Рисунок 11 – Всплывающее окно с информацией для пользователя

На рисунке 12 представлено окно для редактирования данных. Вверху расположен выпадающий список, который позволяет переключаться между загруженными в базу данных таблицами. Данные выбранного датасета отображаются в иерархическом списке, который можно пролистывать с помощью вертикального и горизонтального скроллеров.

Внизу окна находятся кнопки «Добавить строку», «Удалить строку», «Добавить столбец», «Удалить столбец», они отвечают за изменение структуры таблицы. Строка автоматически добавляется в конец иерархического списка, удалить ее можно при выделении и нажатии на кнопку. Для взаимодействия со столбцами открываются диалоговые окна, в которые нужно ввести значение добавляемого или удаляемого столбца. Кнопка «Сохранить» применяет внесенные в датасет изменения.

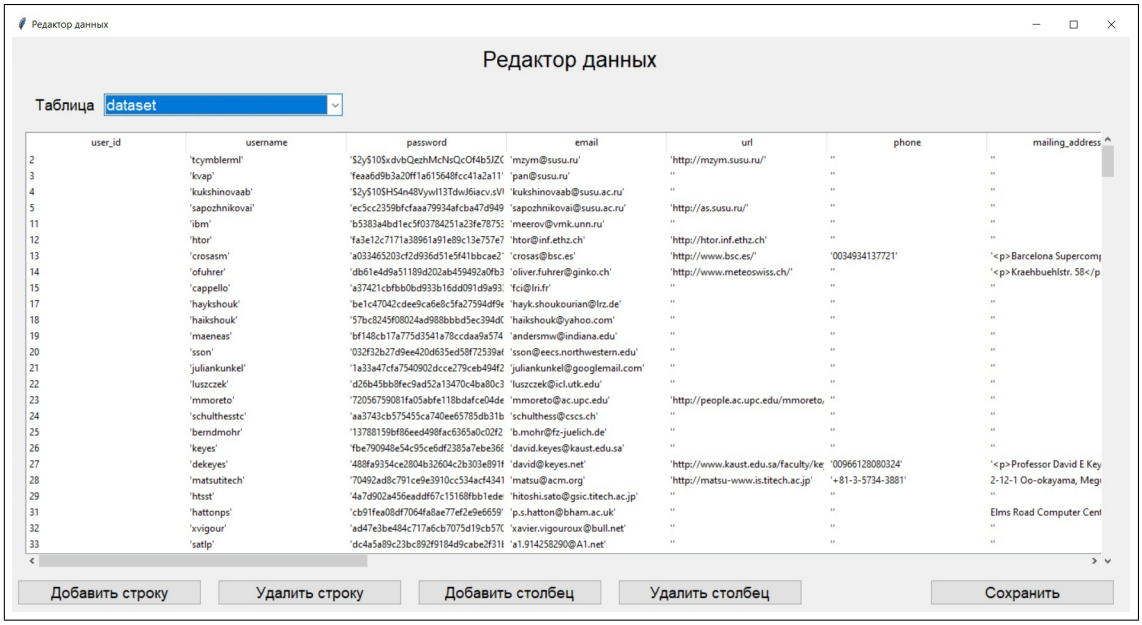

Рисунок 12 – Окно редактирования данных

При двойном клике на ячейку таблицы появляется окно с возможностью редактирования ее значения (рисунок 13). На нем расположено текстовое поле для ввода данных, а также две кнопки, позволяющие применить или отклонить внесенные изменения.

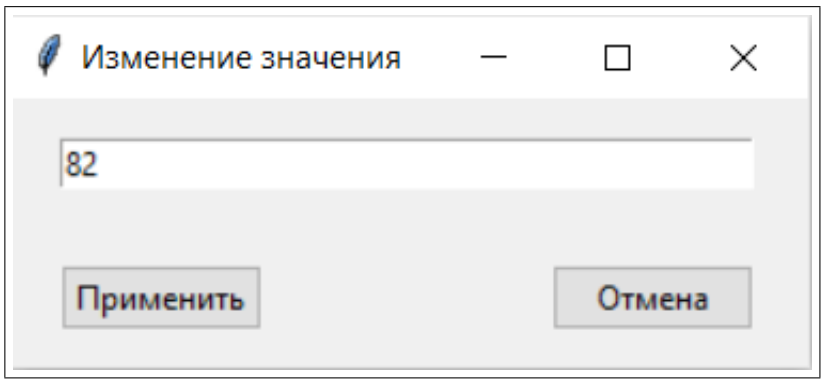

Рисунок 13 – Изменение значения ячейки

На рисунке 14 представлено окно с настройками алгоритмов. На нем расположено несколько выпадающих списков: для выбора алгоритма, таблицы с данными, а также необходимости визуализации. При нажатии на кнопку «Выбрать» открывается диалоговое окно, в котором исследователь может подобрать папку для сохранения результатов работы алгоритма. Путь к ней отображается в соседнем текстовом поле. Внизу расположена кнопка «Найти фейки», которая запускает работу алгоритмов.

Кроме того, при выборе определенного алгоритма меняются виджеты в правой части окна. Для изоляционного леса, дерева решений и случайного леса отображаются выпадающий список с выбором модели, для DBSCAN (рисунок 15) два поля для ввода его параметров, а для иерархической кластеризации (рисунок 16) выпадающий список с выбором метода соединения.

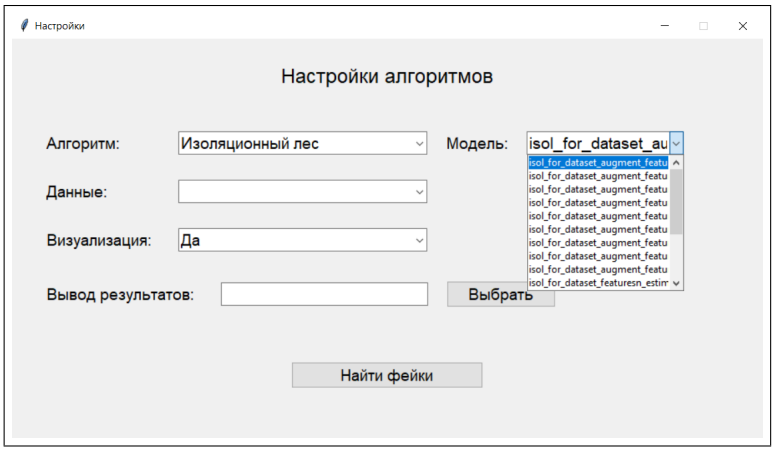

Рисунок 14 – Окно настроек: изоляционный лес

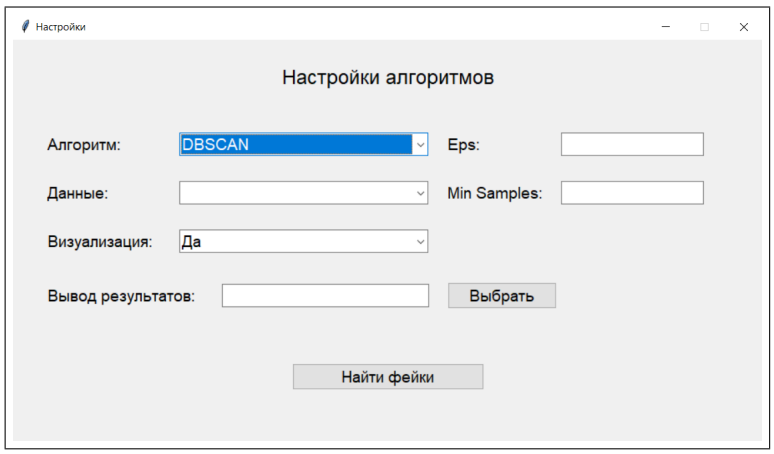

Рисунок 15 – Окно настроек: DBSCAN

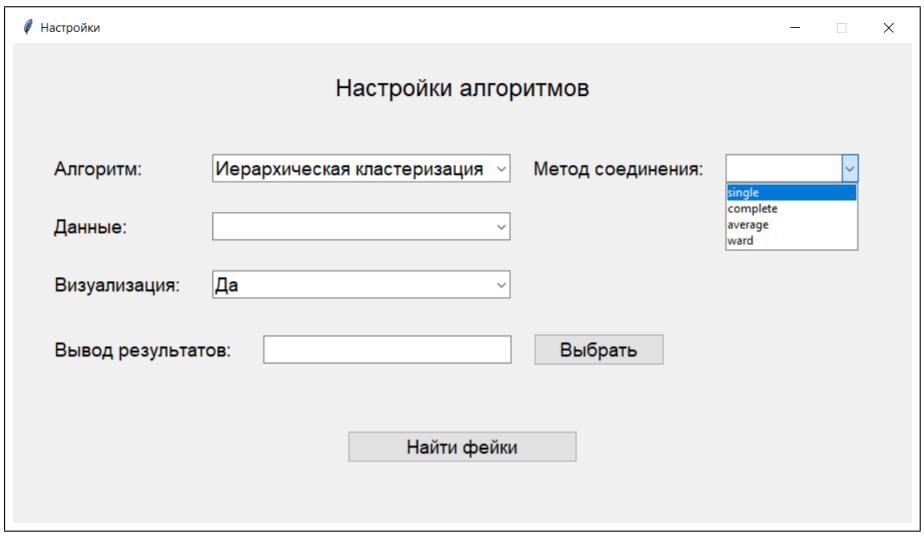

Рисунок 16 – Окно настроек: иерархическая кластеризация

При выполнении какой-либо задачи пользователь получает уведомление от системы о результате работы. В случае, когда не были выбраны данные или они были введены некорректно, появляется высплавающее окно с информацией о предупреждении (пример представлен на рисунке 17).

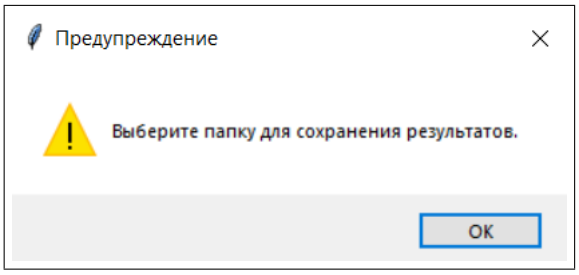

Рисунок 17 – Предупреждение об ошибке

Если данные введены корректно или операция прошла успешно, приложение оповещает пользователя соответствующим уведомлением (пример на рисунке 18).

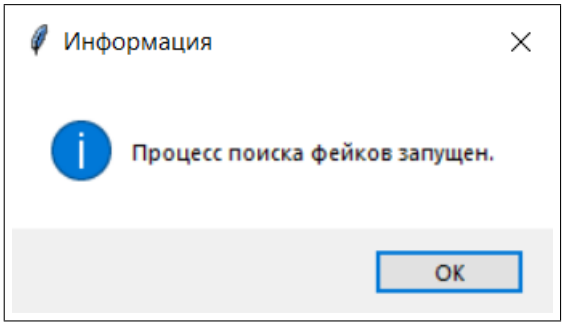

Рисунок 18 – Уведомление пользователю

На диаграмме последовательности (рисунок 19) представлена работа Исследователя с системой, а именно процесс настройки алгоритмов и в каких случая пользователь получает сообщения от приложения.

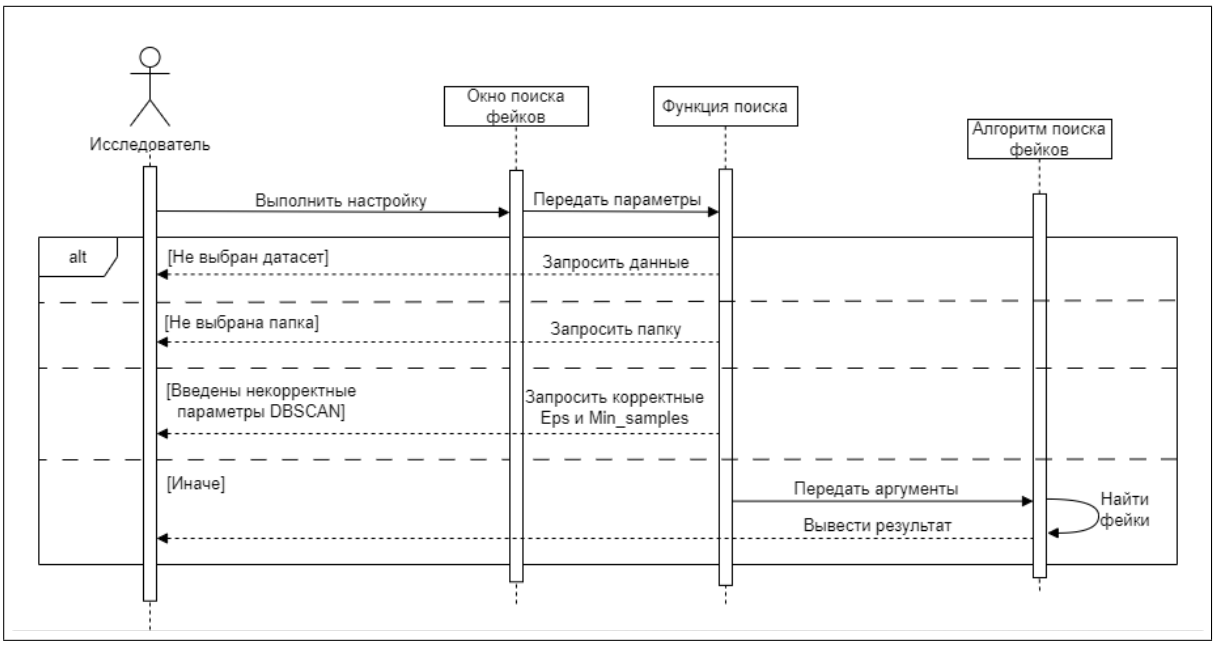

Рисунок 19 – Диаграмма взаимодействия Исследователя с системой

3.4. Реализация обучения моделей

Реализация аугментации

Модуль аугментации необходим для расширения датасета. Данный компонент предоставляет возможность искусственно увеличивать количество записей фиктивных или настоящих аккаунтов, уравнивая их число. Также он позволяет создавать csv файлы с числовыми характеристиками аккаунтов.

• Save_data_to_csv: на вход получает датафрейм с данными, строки с именем таблицы, и путем до корня приложения. Функция сохраняет данные в csv файл.

• Get_max_user_id: принимает датафрейм. Возвращает максимальное значение user_id либо ничего, если нет такого столбца.

• Balance_data: функция принимает строку с названием файла, данные которого используются для аугментации. Выводит количество фиктивных и настоящих аккаунтов до и после аугментации. Данные из csv файла считываются, находится количество фиктивных и настоящих аккаунтов, и

определяется, какие записи необходимо создавать. Get_max_user_id находит максимальный ID пользователя, чтобы новые записи были расположены в порядке увеличения user_id в конце датасета. Далее функция создает записи на основе выбора случайных значений из списков с данными столбцов. После этого объединяет аугментированный и обычный датафреймы, вызывает функцию save_data_to_csv для сохранения полной таблицы в файл.

- Making_features: принимает имя файла с данными. Создает характеристики, аналогичные тем, что создает модуль создания признаков (подробнее в разделе 3.2 Реализация компонентов приложения), но сохраняет их в csv файл.

Реализация модуля обучения моделей

Данный компонент необходим для обучения моделей изоляционного леса, дерева решений и случайного леса. Кроме того, он вызывает функции из модуля аугментации для создания таблиц с признаками.

В нем содержится функция teaching_models. На вход она принимает строки с именем файла, на основе которого модель обучается, названием алгоритма, а также список словарей с параметрами.

Данные считываются, делятся на признаки и метки. В зависимости от выбранного алгоритма функция выбирает класс модели. Для каждого переданного набора параметров создается и обучается модель, которая сохраняется в отдельную папку в корне приложения.

4. ТЕСТИРОВАНИЕ

4.1. Функциональное тестирование

В ходе тестирования проверялось соответствие приложения функциональным требованиям. В таблице 3 приведен протокол ручного тестирования основных аспектов работы приложения.

Таблица 3 – Функциональное тестирование системы

№	Название теста	Действие	Ожидаемый результат	Тест пройден?
1	Загрузка данных в систему.	В главном окне выбрать кнопку «Загрузить данные», выбрать csv файл.	Файл загружен в базу данных.	Да
2	Добавление строки в датасет.	В главном окне выбрать кнопку «Отредактировать данные», выбрать таблицу в выпадающем списке нажать на кнопку «Добавить строку».	Строка добавлена.	Да
3	Удаление строки из датасета.	В окне редактирования данных выбрать строку, нажать на кнопку «Удалить строку».	Строка удалена из таблицы.	Да
4	Добавление столбца в датасет.	В окне редактирования данных нажать на кнопку «Добавить столбец», написать его название в всплывающем окне.	Столбец добавлен в таблицу.	Да
5	Удаление столбца из датасета.	В окне редактирования данных нажать на кнопку «Удалить столбец», написать его название в всплывающем окне.	Столбец удален из таблицы.	Да
6	Редактирование значения ячейки.	В окне редактирования данных два раза кликнуть на ячейку, в всплывающем окне изменить ее значение, применить его.	Значение ячеки изменено.	Да
7	Попытка сохранения данных без выбора таблицы.	В окне редактирования нажать кнопку «Сохранить», без выбранной таблицы для редактирования.	Отображение всплывающего окна с предупреждением.	Да
8	Проверка работоспособности окна настроек.	В главном окне выбрать «Найти фейки», в выпадающих списках настроить алгоритм, нажать кнопку «Найти».	Запускается процесс поиска «фейков».	Да
9	Проверка ввода неполных данных.	В окне настроек не выбрать папку, не выбрать данные, нажать кнопку «Найти».	Отображение всплывающего окна с предупреждением.	Да

4.2. Сравнение алгоритмов

Для нахождения подходящего алгоритма выявления фиктивных аккаунтов были проведены эксперименты с сравненем метрик.

Исследования проводились с несколькими наборами данных, у которых было разное соотношение настоящих и фальшивых аккаунтов, а также с разными параметрами алгоритмов. Процентные соотношения «фейков» в данных, которые были использованы: 10%, 20%, 30%, 40%, 50%.

Изоляционный лес. В ходе эксперимента использовались разные модели. Наиболее стабильные результаты показали две из них, они отличаются количеством деревьев: 50 в первой и 100 во второй, доля аномалий в обеих автоматическая. Они выдали одинаковые метрики, которые представлены в таблице 4

Таблица 4 – Метрики изоляционного леса

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,49	0,10	0,50	0,17
20% фейков	0,48	0,18	0,44	0,26
30% фейков	0,50	0,25	0,33	0,29
40% фейков	0,49	0,38	0,42	0,40
50% фейков	0,41	0,38	0,29	0,33

Иерархическая кластеризация. Она может использоваться с разными методами соединения: одиночной связи (single linkage), полной связи (complete linkage), средней связи (average linkage), Уорда (Ward’s linkage).

Работа алгоритма с методом одиночной связи и разным содержанием фиктивных аккаунтов в датасетах представлена в таблице 5.

Таблица 5 – Метрики иерархической кластеризации: одиночная связь

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,85	0,25	0,25	0,25
20% фейков	0,77	0,40	0,22	0,29
30% фейков	0,72	0,67	0,13	0,22
40% фейков	0,61	0,60	0,12	0,21
50% фейков	0,51	0,60	0,09	0,15

Работа алгоритма с методом полной связи и разным содержанием фиктивных аккаунтов в датасетах представлена в таблице 6.

Таблица 6 – Метрики иерархической кластеризации: полная связь

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,82	0,20	0,25	0,22
20% фейков	0,75	0,23	0,22	0,27
30% фейков	0,54	0,17	0,13	0,15
48% фейков	0,49	0,33	0,25	0,29
50% фейков	0,40	0,32	0,17	0,22

Работа алгоритма с методом средней связи и разным содержанием фиктивных аккаунтов в датасетах представлена в таблице 7.

Таблица 7 – Метрики иерархической кластеризации: средняя связь

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,87	0,33	0,25	0,29
20% фейков	0,80	0,50	0,22	0,31
30% фейков	0,58	0,20	0,13	0,16
40% фейков	0,51	0,27	0,12	0,17
50% фейков	0,43	0,27	0,09	0,13

Работа алгоритма с методом Уорда и разным содержанием фиктивных аккаунтов в датасетах представлена в таблице 8.

Таблица 8 – Метрики иерархической кластеризации: метод Уорда

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,90	0,50	0,25	0,33
20% фейков	0,82	0,67	0,22	0,33
30% фейков	0,60	0,22	0,13	0,17
38% фейков	0,53	0,30	0,12	0,18
50% фейков	0,44	0,30	0,09	0,13

Агломеративная кластеризация не справляется с нахождением фиктивных аккаунтов, но для сравнения с другими алгоритмами будут использоваться результаты, полученные с помощью метода метода Уорда.

DBSCAN. У алгоритма есть два параметра: *Eps* и *MinPts*, которые зависят от данных и объема выбросов в них. Для каждого датасета они подбирались индивидуально, метрики представлены в таблицах ниже.

Результаты работы алгоритма с данными, содержащими 10% фиктивных аккаунтов, представлены в таблице 9. Исходя из них, для дальнейшего сравнения алгоритмов будут использоваться параметры $Eps = 0.7$ и $MinPts = 4$.

Таблица 9 – Метрики DBSCAN: датасет с 10% фиктивных аккаунтов

Параметры	accuracy	precision	recall	F-measure
Eps = 0.4; MinPts = 3	0,79	0,17	0,25	0,20
Eps = 0.5; MinPts = 4	0,85	0,25	0,25	0,25
Eps = 0.7; MinPts = 4	0,87	0,33	0,25	0,29

Результаты работы алгоритма с данными, содержащими 20% фиктивных аккаунтов, представлены в таблице 10. В данном случае при сравнении будут использоваться параметры $Eps = 0.5$ и $MinPts = 4$.

Таблица 10 – Метрики DBSCAN: датасет с 20% фиктивных аккаунтов

Параметры	accuracy	precision	recall	F-measure
Eps = 0.4; MinPts = 4	0,79	0,29	0,22	0,25
Eps = 0.5; MinPts = 4	0,77	0,40	0,22	0,29
Eps = 0.6; MinPts = 5	0,77	0,33	0,11	0,17

Результаты работы алгоритма с данными, содержащими 30% фиктивных аккаунтов, представлены в таблице 11. Исходя из них, для дальнейших исследований с этим датасетом будут использоваться параметры $Eps = 0.5$ и $MinPts = 5$.

Таблица 11 – Метрики DBSCAN: датасет с 30% фиктивных аккаунтов

Параметры	accuracy	precision	recall	F-measure
Eps = 0.4; MinPts = 5	0,66	0,33	0,13	0,19
Eps = 0.5; MinPts = 5	0,70	0,50	0,13	0,21
Eps = 0.45; MinPts = 6	0,68	0,40	0,13	0,20

Результаты работы алгоритма с данными, содержащими 40% фиктивных аккаунтов, представлены в таблице 12. Далее при сравнении будут использоваться параметры $Eps = 0.5$ и $MinPts = 6$.

Таблица 12 – Метрики DBSCAN: датасет с 40% фиктивных аккаунтов

Параметры	accuracy	precision	recall	F-measure
Eps = 0.4; MinPts = 6	0,59	0,50	0,12	0,20
Eps = 0.5; MinPts = 6	0,61	0,60	0,12	0,21
Eps = 0.5; MinPts = 8	0,56	0,38	0,12	0,19

Результаты работы алгоритма с данными, содержащими 50% фиктивных аккаунтов, представлены в таблице 13. Для дальнейшего сравне-

ния будут использоваться параметры $Eps = 0.4$ и $MinPts = 8$.

Таблица 13 – Метрики DBSCAN: датасет с 50% фиктивных аккаунтов

Параметры	accuracy	precision	recall	F-measure
Eps = 0.4; MinPts = 8	0,43	0,31	0,11	0,17
Eps = 0.5; MinPts = 8	0,47	0,38	0,09	0,14
Eps = 0.6; MinPts = 9	0,53	0,75	0,09	0,15

Дерево решений. Было создано несколько моделей с разными критериями: индексом Джини и энтропией. Модель, которая использовала энтропию при разделении, показала лучшие результаты, которые представлены в таблице 15

Таблица 14 – Метрики дерева решений

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,90	0,50	0,75	0,60
20% фейков	0,93	0,80	0,89	0,84
30% фейков	0,84	0,82	0,60	0,69
40% фейков	0,85	0,89	0,71	0,79
50% фейков	0,84	0,93	0,74	0,83

Случайный лес. Создавались модели с разным количеством деревьев решений. Модель, которая строила 100 деревьев, показала лучшие результаты, они представлены в таблице 15

Таблица 15 – Метрики случайного леса

Датасет	accuracy	precision	recall	F-measure
10% фейков	0,90	0,50	0,75	0,60
20% фейков	0,86	0,67	0,67	0,67
30% фейков	0,82	0,75	0,60	0,67
40% фейков	0,85	0,94	0,67	0,78
50% фейков	0,83	0,96	0,69	0,80

Сравнение результатов алгоритмов. Рассчитанные метрики для всех реализованных алгоритмов с различными соотношениями фиктивных аккаунтов представлены в таблице 16.

По метрикам можно сделать вывод, что с нахождением фальшивых записей лучше всего справляются алгоритмы классификации, в частности дерево решений.

Таблица 16 – Метрики алгоритмов

Алгоритмы	**accuracy**	**precision**	**recall**	**F-measure**
10% фейков				
Изоляционный лес	0,49	0,10	0,50	0,17
Иерархическая кластеризация	0,90	0,50	0,25	0,33
DBSCAN	0,87	0,33	0,25	0,29
Дерево решений	0,90	0,50	0,75	0,60
Случайный лес	0,90	0,50	0,75	0,60
20% фейков				
Изоляционный лес	0,48	0,18	0,44	0,26
Иерархическая кластеризация	0,82	0,67	0,22	0,33
DBSCAN	0,77	0,40	0,22	0,29
Дерево решений	0,93	0,80	0,89	0,84
Случайный лес	0,86	0,67	0,67	0,67
30% фейков				
Изоляционный лес	0,50	0,25	0,33	0,29
Иерархическая кластеризация	0,60	0,22	0,13	0,17
DBSCAN	0,70	0,50	0,13	0,21
Дерево решений	0,84	0,82	0,60	0,69
Случайный лес	0,82	0,75	0,60	0,67
40% фейков				
Изоляционный лес	0,49	0,38	0,42	0,40
Иерархическая кластеризация	0,53	0,30	0,12	0,18
DBSCAN	0,61	0,60	0,12	0,21
Дерево решений	0,85	0,89	0,71	0,79
Случайный лес	0,85	0,94	0,67	0,78
50% фейков				
Изоляционный лес	0,41	0,38	0,29	0,33
Иерархическая кластеризация	0,44	0,30	0,09	0,13
DBSCAN	0,43	0,31	0,11	0,17
Дерево решений	0,84	0,93	0,74	0,83
Случайный лес	0,83	0,96	0,69	0,80

ЗАКЛЮЧЕНИЕ

В ходе работы была создана система, позволяющая загружать датасеты в базу данных, редактировать таблицы и проводить эксперименты. Были изучены и реализованы алгоритмы интеллектуального анализа данных, которые можно использовать для поиска фиктивных аккаунтов. Кроме того, были решены следующие задачи:

1. Проведен анализ предметной области и литературы по теме работы.
2. Спроектирован интерфейс программной системы и модульной структуры приложения.
3. Реализована система, выявляющая фиктивные аккаунты с помощью алгоритмов машинного обучения.
4. Подготовлен набор тестов, а также выполнено тестирование программной системы.

В дальнейшем планируется модифицировать алгоритмы нахождения аномалий с целью улучшения их работы, реализовать новые, а также добавить возможность обучения моделей самим Исследователем.

ЛИТЕРАТУРА

1. Banfield J.D., Raftery A.E. Model-based Gaussian and non-Gaussian clustering. // Biometrics, 1993. – Vol. 49 – 803–821 pp.

2. Boshmaf Y. The socialbot network: when bots socialize for fame and money. / Y. Boshmaf, I. Muslukhov, K. Beznosov, M. Ripeanu. // Twenty-Seventh Annual Computer Security Applications Conference, ACSAC 2011, Orlando, FL, USA, 5-9 December 2011 / Ed. by R.H. Zakon, J.P. McDermott, M.E. Locasto. – ACM, 2011. – 93–102 pp. – URL: https://doi.org/10.1145/2076732.2076746.

3. Breiman L. Random Forests. // Mach. Learn. – 2001. – Vol. 45. – No. 1. – P. 5–32. – URL: https://doi.org/10.1023/A:1010933404324.

4. Breiman L. Classification and Regression Trees / L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone. – Wadsworth, 1984.

5. Breuer A. Preemptive Detection of Fake Accounts on Social Networks via Multi-Class Preferential Attachment Classifiers. / A. Breuer, N.K. Tehrani, M. Tingley, B. Cottel. // CoRR, 2023. – Vol. abs/2308.05353. – arXiv : 2308.05353.

6. Chen Y., Wu S.F. FakeBuster: A Robust Fake Account Detection by Activity Analysis. // 9th International Symposium on Parallel Architectures, Algorithms and Programming, PAAP 2018, Taipei, Taiwan, December 26-28, 2018. – IEEE, 2018. – 108–110 pp.

7. Course_work. – [Электронный ресурс] URL: https://github.com/AnastasiaTolmacheva/course_work/tree/app (дата обращения: 2023-09-20 г.).

8. Damerau F. A technique for computer detection and correction of spelling errors. // Commun. ACM, 1964. – Vol. 7. – No. 3. – 171–176 pp. – URL: https://doi.org/10.1145/363958.363994.

9. Elyusufi Y., Elyusufi Z., Kbir M.A. Social networks fake profiles detection based on account setting and activity. // Proceedings of the 4th International Conference on Smart City Applications, SCA 2019, Casablanca, Morocco, October 02-04, 2019. – ACM, 2019. – 37:1–37:5 pp. – URL: https://doi.org/10.1145/3368756.3369015.

10. Ester M. A Density-Based Algorithm for Discovering Clusters in

Large Spatial Databases with Noise. / M. Ester, H. Kriegel, J. Sander, X. Xu. // Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA / Ed. by E. Simoudis, J. Han, U.M. Fayyad. – AAAI Press, 1996. – 226–231 pp. – URL: http://www.aaai.org/Library/KDD/1996/kdd96-037.php.

11. Fahmy S.G. Modeling the Influence of Fake Accounts on User Behavior and Information Diffusion in Online Social Networks. / S.G. Fahmy, S. AbdelGaber, O.H. Karam, D.S. Elzanfaly. // Informatics, 2023. – Vol. 10. – No. 1. – 27 p. – URL: https://doi.org/10.3390/informatics10010027.

12. Gurajala S. Fake Twitter accounts: profile characteristics obtained using an activity-based pattern detection approach. / S. Gurajala, J.S. White, B. Hudson, J.N. Matthews. // Proceedings of the 2015 International Conference on Social Media & Society, Toronto, ON, Canada, July 27-29, 2015 / Ed. by A.A. Gruzd, J. Jacobson, P. Mai, B. Wellman. – ACM, 2015. – 9:1–9:7 pp. – URL: https://doi.org/10.1145/2789187.2789206.

13. Hassan A., Alhalangy A.G.I., Al-Zahrani F. Fake Accounts Identification in Mobile Communication Networks Based on Machine Learning. // Int. J. Interact. Mob. Technol., 2023. – Vol. 17. – No. 4. – 64–74 pp. – URL: https://doi.org/10.3991/ijim.v17i04.37645.

14. Hsu S., Kes D., Joshi A. Visualizing Tweets from Confirmed Fake Russian Accounts. // Visualization and Data Analysis 2019, Burlingame, CA, USA, 16-17 January 2019 / Ed. by T. Wischgoll, S. Zhang, D.L. Kao, Y. Chiang. – Society for Imaging Science and Technology, 2019. – URL: https://doi.org/10.2352/ISSN.2470-1173.2019.1.VDA-678.

15. Liu F.T., Ting K.M., Zhou Z. Isolation Forest. // Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy. – IEEE Computer Society, 2008. – 413–422 pp. – URL: https://doi.org/10.1109/ICDM.2008.17.

16. Maptolib. – [Электронный ресурс] URL: https://matplotlib.org/stable/api/pyplot_summary.html (дата обращения: 2023-10-17 г.).

17. Mohammadrezaei M., Shiri M.E., Rahmani A.M. Detection of Fake Accounts in Social Networks Based on One Class Classification. // ISC Int. J.

Inf. Secur., 2019. – Vol. 11. – No. 2. – 173–183 pp. – URL: https://doi.org/10.22042/isecure.2019.165312.450.

18. Numpy. – [Электронный ресурс] URL: https://numpy.org/doc/stable/ (дата обращения: 2023-10-17 г.).

19. Pandas. – [Электронный ресурс] URL: https://pandas.pydata.org/ (дата обращения: 2023-10-15 г.).

20. Santisteban J., Tejada-Cárcamo J. Unilateral Jaccard Similarity Coefficient. // Proceedings of the First International Workshop on Graph Search and Beyond, GSB 2015, co-located with The 38th Annual SIGIR Conference (SIGIR'15), Santiago, Chile, August 13th, 2015 / Ed. by O. Alonso, M.A. Hearst, J. Kamps. – Vol. 1393 of CEUR Workshop Proceedings. – CEUR-WS.org, 2015. – 23–27 pp. – URL: https://ceur-ws.org/Vol-1393/paper-10.pdf.

21. Scikit-learn. – [Электронный ресурс] URL: https://scikit-learn.org/stable/ (дата обращения: 2023-10-16 г.).

22. Sqlite3. – [Электронный ресурс] URL: https://docs.python.org/3/library/sqlite3.html (дата обращения: 2023-10-15 г.).

23. Stolbova A., Ganeev R., Ivaschenko A. Intelligent Identification of Fake Accounts on Social Media. // 30th Conference of Open Innovations Association, FRUCT 2021, Oulu, Finland, October 27-29, 2021. – IEEE, 2021. – 279–284 pp. – URL: https://doi.org/10.23919/FRUCT53335.2021.9599974.

24. Tkinter. – [Электронный ресурс] URL: https://docs.python.org/3/library/tkinter.html (дата обращения: 2024-02-24 г.).

25. Uppada S.K. Novel approaches to fake news and fake account detection in OSNs: user social engagement and visual content centric model. / S.K. Uppada, K. Manasa, B. Vidhathri, R. Harini, B. Sivaselvan. // Soc. Netw. Anal. Min., 2022. – Vol. 12. – No. 1. – 52 p. – URL: https://doi.org/10.1007/s13278-022-00878-9.

ПРИЛОЖЕНИЯ

Приложение А. Примеры размеченных аккаунтов

Примеры аккаунтов, которые были размечены, представлены в таблицах 1– 2.

Таблица 1 – Фиктивный аккаунт

Характеристика	Значение
user_id	6758
username	**fishzupic438**
password	6cd28fca7bdf8937ee45eb65d0a126a5
email	**fishgvnpo334@gmail.com**
url	**https://aktivator-kleva.com**
phone	88298893596
mailing_address	**fishfmmzj822@gmail.com**
billing_address	NULL
country	MT
locales	
date_last_email	NULL
date_registered	**2021-03-04 22:22:07**
date_validated	NULL
date_last_login	**2021-03-04 22:22:07**
must_change_password	0
auth_id	NULL
auth_str	NULL
disabled	0
disabled_reason	NULL
inline_help	0
gossip	NULL
fake	1

Таблица 2 – Настоящий аккаунт

Характеристика	Значение
user_id	2
username	tcymblerml
password	$2y$10$xdvbQezhMcNsQcOf4b5JZOZg2butksNxug7/TiEV.sClODdS4djJK
email	mzym@susu.ru
url	http://mzym.susu.ru/
phone	
mailing_address	
billing_address	NULL
country	Ru
locales	
date_last_email	2020-01-03 18:42:34
date_registered	2014-01-24 13:57:45
date_validated	NULL
date_last_login	2022-12-04 08:56:49
must_change_password	0
auth_id	1
auth_str	NULL
disabled	0
disabled_reason	NULL
inline_help	0
gossip	NULL
fake	0

Приложение Б. Спецификация вариантов использования

Спецификация вариантов использования (ВИ) разработанной системы приведена в таблицах 3– 6.

Таблица 3 – Спецификация варианта «Загрузить данные»

Прецедент: Загрузить данные
ID: 1
Краткое описание: Исследователь загружает данные в систему.
Главные актеры: Исследователь.
Второстепенные актеры: Нет.
Предусловия: отсутствуют.
Основной поток: 1. Прецедент начинается, когда Исследователь нажимает на кнопку «Загрузить данные». 2. Исследователь выбирает необходимый файл. 3. Система сохраняет выбранный датасет в базе данных.
Постусловия: Исследователь загрузил датасет в базу данных.
Альтернативные потоки: Нет.

Таблица 4 – Спецификация варианта «Редактировать данные»

Прецедент: Редактировать данные
ID: 2
Краткое описание: Исследователь редактирует загруженные данные.
Главные актеры: Исследователь.
Второстепенные актеры: Нет.
Предусловия: данные для редактирования были загружены.
Основной поток: 1. Прецедент начинается, когда Исследователь нажимает на кнопку «Отредактировать данные». 2. Исследователь выбирает данные, которые будет редактировать. 3. Исследователь вносит изменения в датасет. 4. Исследователь нажимает на кнопку «Сохранить».
Постусловия: Данные были отредактированы.
Альтернативные потоки: Нет.

Таблица 5 – Спецификация ВИ «Выполнить настройку»

Прецедент: Выполнить настройку
ID: 4
Краткое описание: Исследователь выбирает алгоритм и настраивает его параметры.
Главные актеры: Исследователь.
Второстепенные актеры: Нет.
Предусловия: Загружены данные.
Основной поток 1. Прецедент начинается, когда пользователь нажимает на кнопку «Найти фейки». 2. Пользователь выбирает необходимый алгоритм и вводит его параметры.
Постусловия: Выполнена настройка алгоритма.
Альтернативные потоки: Нет

Таблица 6 – Спецификация ВИ «Найти фиктивные аккаунты»

Прецедент: Найти фиктивные аккаунты
ID: 5
Краткое описание: Исследователь запускает алгоритм для поиска фиктивных аккаунтов.
Главные актеры: Исследователь.
Второстепенные актеры: Нет.
Предусловия: Загружен файл с данными, выполнена настройка алгоритма.
Основной поток 1. Прецедент начинается, когда пользователь нажимает на кнопку «Найти фейки» в окне настроек. 2. Система находит фиктивные аккаунты и выводит результат работы алгоритма.
Постусловия: Выполнен поиск фиктивных аккаунтов.
Альтернативные потоки: Нет