МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Мобильные и сетевые технологии
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка ПО для анализа цифрового следа клиентов банка»

Обучающийся Д.В. Дмитриев _____
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., В.С. Климов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2023

Аннотация

Название бакалаврской работы: «Анализ цифрового следа клиентов банка».

Объектом исследования является процесс сбора цифрового следа и последующий его анализ.

Выпускная работа состоит из введения, трёх разделов, заключения и списка литературы, включая зарубежные источники.

Предметом выпускной квалификационной работы является алгоритм анализа цифрового следа клиентов банка.

Ключевым вопросом бакалаврской работы является создание способа анализа цифрового следа клиентов банка.

Целью работы является структурирование данных клиентов банка и разработка процессов анализа цифрового следа.

Актуальность данной темы состоит в необходимости банкам адаптироваться к изменяющимся потребностям клиентов и обеспечивать им более персонализированные услуги. Анализ цифрового следа клиентов позволяет выявить их предпочтения, поведенческие паттерны, а также определить экономические возможности клиентов.

В конце исследования результат работы анализа.

Abstract

The title of the graduation work is "Analysis of the digital footprint of bank customers". The work considers the process of collecting the digital footprint and its subsequent analysis focusing on the analysis algorithm.

The key issue of the graduation work is to create a way to analyze the digital footprint of bank customers.

The aim is to structure the data of bank customers and to develop processes for the analysis of the digital footprint.

The relevance of this topic is based on the need for banks to adapt to the changing needs of customers and provide them with more personalized services. The analysis of customers' digital footprint makes it possible to identify their preferences, behavioral patterns, as well as to determine the economic opportunities of customers.

To reach my goals for the graduation work I used Python. To expand its capabilities I used libraries such as Matplotlib and Seaborn for graphs and NumPy and Pandas to work with datasets and do computing processes. I chose Python for it because this programming language is made for automating tasks and conducting the data analysis.

Содержание

Введение

Современный банковский сектор активно использует цифровые технологии для улучшения своей эффективности и предоставления качественных финансовых услуг. Вместе с этим, увеличивается объем цифровых данных, которые генерируются клиентами банков в процессе взаимодействия с различными банковскими продуктами и услугами. Эти данные, известные как цифровой след клиентов, представляют собой ценный источник информации о поведении, предпочтениях и потребностях клиентов.

Анализ цифрового следа клиентов банка является одним из актуальных направлений исследований в банковской сфере. Он позволяет банкам получить глубокое понимание своих клиентов, предсказать их поведение, определить кредитоспособность и принимать обоснованные решения о выдаче кредитов. Это открывает новые возможности для банков в области улучшения качества обслуживания клиентов, снижения рисков и повышения прибыльности.

Целью данной дипломной работы является исследование и анализ цифрового следа клиентов банка с использованием современных методов анализа данных и машинного обучения. В рамках работы будет проведен обзор методов сбора цифрового следа, а также рассмотрены примеры готовых решений для анализа и использования цифрового следа в банковской сфере.

Данная работа имеет практическую значимость для банков, поскольку представляет собой практическое исследование и разработку методов анализа цифрового следа, которые могут быть применены для принятия обоснованных решений о выдаче кредитов и улучшения финансовой деятельности банка. Данная выпускная квалификационная работа также поможет решить проблему загруженности банков в секторе решения выдачи кредитов, что даст больше времени на решения других важных задач для банка и уменьшит влияние человеческого фактора на решение кредитования.

1 Обзор цифрового следа

1.1. Определение цифрового следа

Цифровой след, также называемый «электронным следом», «кибер-тенью» или «цифровой тенью» - это огромный набор персональных данных о личности пользователя, который остаётся от любых действий при использовании глобальной информационной сети. Этот термин хоть и применяется зачастую к человеку и его личности, но также может относиться к коммерческим и некоммерческим организациям, компаниям, а также корпорациям (ведь любая компания - это прежде всего люди).

Цифровой след хранится в файлах coockie - когда пользователь совершает какое-либо действие, к примеру, заполняет форму логина и пароля для входа или добавляет картинку на аватар в своей любимой социальной сети - сервер заполняет эту информацию в куки и высылает браузеру вместе с веб-страницей.

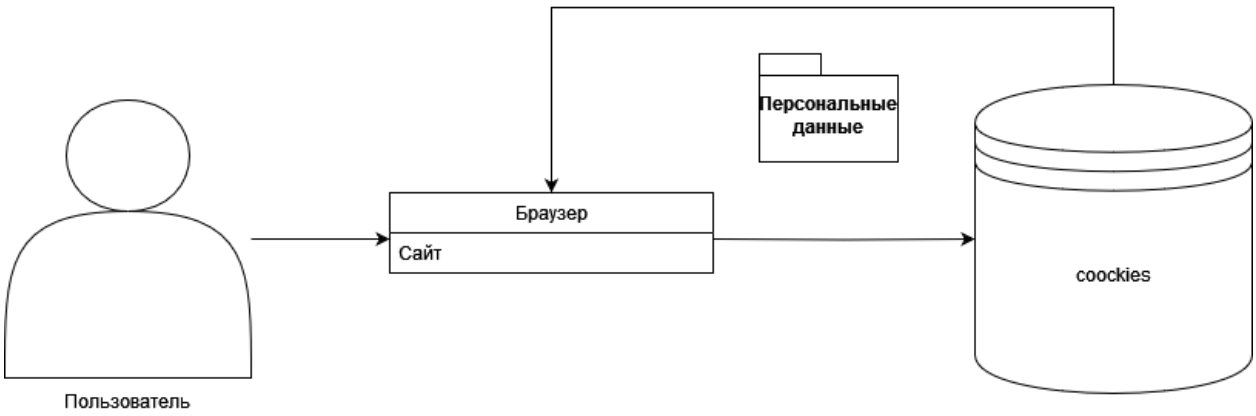

Рисунок 1 - Работа кукис

Также цифровой след необязательно должен быть виртуальным. Мы его оставляем даже в тот момент, когда подключаемся к общественной точке Wi-Fi сети [9].

Расширению цифрового следа помогают отзывы оставленные в интернет магазинах на различные товары, публикации на форумах и социальных сетях, оформление подписок на видео-, аудио-, информационные ресурсы, а также покупки в интернет-магазинах [9].

Не всегда цифровой след бывает очевиден [17]. Представим ситуацию, кто-то сфотографировал мероприятие в городе и вы случайно попадаете на фотографию. Обычная, казалось бы, ситуация, но эту фотографию выкладывают в местную группу города в социальной сети, начинается обсуждение, фотография обретает всё больше ключевых слов и вот уже о вас можно найти город, в котором вы живёте, какую одежду предпочитаете и сколько вам лет. Также ваши данные могут быть собраны приложениями, которые отследили вашу информацию без вашего ведома. Ваша личная информация может передаваться третьим лицам или она может быть скомпрометированы в результате атаки хакеров на базы данных.

Для обозначений цифрового следа существуют два термина : «Активный цифровой след» и «Пассивный цифровой след».

Активный цифровой след - те данные пользователя, оставленные сознательно им же: комментарий в социальной сети, публикация в блоге, сообщение на форуме, согласие на файлы cookie, заполнение опросников и онлайн форм. Он помогает создать цифровой образ личности.

Пассивный цифровой след - те данные пользователя, оставленные им без его ведома. Такую информацию собирают сайты скрытно. Сюда можно отметить те случаи, когда сайты сохраняют IP-адрес пользователей, через какой сайт пользователь перешёл на другую веб-страницу, каким браузером и устройством пользуется.

1.2. Цели сбора цифрового следа

Информация - важнейший ресурс постиндустриального общества, поэтому цифровой след и его анализ - важнейшие инструменты.

Организация работы с открытыми и доступными источниками информации позволяет более точно определить финансовое положение компании или конкурента, их намерения и направления будущих инвестиций, конкурентные преимущества, а также выяснить корпоративную культуру компании и её иерархическое построение [13].

Нередко анализом цифрового следа пользуются и рекламодатели [17]. Для персонализации рекламы и формирования пользовательских потребностей в сфере товаров и услуг не хватает только информации о его возрасте, поле и месте проживания - нужна конкретика, для этого собираются данные и ключевые слова в переписках с другими пользователями, анализируются истории посещения сайтов и запросы в поисковике [1]. Например, мы знаем что наш пользователь женщина, которая последнее время делает запросы в поисковике на темы о декрете и посещает женские форумов для молодых мам - из чего мы можем сделать вывод, что она узнала о беременности и предлагаем ей купить смеси и кроватку для малыша. Конечно, такая реклама может быть не всегда точной и девушка может просто интересоваться, но для этого и собирается цифровой «отпечаток» - актуализировать виртуальную личность человека.

Работа с анализом цифрового следа помогает не только в сфере экономики, но и в сфере образования - приведу сноску из статьи «Вестник Марийского Государственного Университета»:

“Основываясь на анализе и специальной обработке этого следа, мы можем дать некоторые советы студентам, направить их и сделать профессиональную подготовку более индивидуально ориентированной. Цифровой след может позволить образовательным учреждениям лучше понимать поведение студентов, оказывать им необходимую помощь, осуществлять наставничество в направлении раскрытия и развития способностей студенческой аудитории” [4].

Сбор и анализ цифрового следа применяется и в банковской сфере. Такой метод анализа данных как «скоринг» - разрешает самый важный

вопрос для банка - сможет ли заёмщик вернуть кредит и, тем самым, сильно упрощает работу сотрудникам [10]. Скоринговая кредитная система работает на мысли о том, что люди с одинаковыми привычками похоже обращаются с финансами. Слово «score» в английском переводится как балл, и действительно, этот метод работает как рейтинг с оценками потенциального заёмщика и каждая характеристика оценивается по-своему. Так например, если в характеристике «позиция сотрудника» стоит «стажёр», то человек получит меньше баллов, чем «директор». Чем показатель рейтинга выше - тем вероятнее шанс у клиента банка получить кредит.

1.3. Методы сбора цифрового следа клиентов банка

Исследование цифрового следа клиентов банка является важной задачей в банковской сфере. Для сбора цифрового следа, который представляет собой цифровые следы и данные, оставленные пользователями во время их взаимодействия с различными цифровыми платформами и устройствами, в банковском контексте можно применить следующие методы:

- Анализ социальных медиа. Сбор данных из социальных сетей, где пользователи публикуют свои имя, фамилию, номер телефона, подписки на группы, дату рождения, предпочтительное времяпровождение, фотографии, комментарии и другую информацию [19]. Эти данные могут быть анализированы для извлечения паттернов и трендов, связанных с поведением и предпочтениями клиентов.

- Мониторинг поисковых запросов. Анализ поисковых запросов пользователей, связанных с финансами и кредитами, может помочь банкам понять потребности и интересы клиентов, а также определить их финансовую грамотность и потенциальные потребности в кредитных продуктах.

- Анализ транзакций. Мониторинг и анализ транзакций клиентов позволяет банкам оценить платёжеспособность клиентов, определить их

финансовую стабильность и выявить нестандартные или мошеннические операции [18].

- Анализ поведения в интернете. Сюда входит не только поведение в социальных сетях, но и анализ корзин покупок в интернет-магазинах и история посещения сайтов. Сбор и анализ данных о поведении клиентов в сети могут предоставить банкам информацию о посещаемых веб-сайтах, поисковых запросах, активности на сайте и других онлайн-привычках клиентов [7]. Эти данные могут быть использованы для составления виртуальной личности клиента, анализа потребностей и предпочтений клиентов, а также для выявления потенциальных рисков и улучшения персонализированных услуг [5]. Сюда также можно добавить инструмент сбора куки. Они используются банком в целях улучшения работы веб-сайта. Они обрабатываются для улучшения предоставления продуктов и услуг банка, определения пользовательских предпочтений, предоставления целевой информации по продуктам и услугам банка и партнеров банка.

- Анализ данных со смартфонов. Получение доступа к установленным приложениям, данным о местоположении владельца телефона, данным, хранящимся на смартфонах клиентов, может предоставить дополнительные сведения о поведении и предпочтениях клиентов. Это может включать контакты, сообщения, фотографии и другую информацию, которая может быть использована для более глубокого анализа и понимания клиентского поведения [20].

- Анализ данных устройств IoT. Устройства Интернета вещей (IoT) могут собирать данные о поведении клиентов, такие как умные дома, колонки-помощники и носимые устройства [5]. Фитнес-браслеты являются электронными устройствами, которые отслеживают физическую активность пользователя, такую как шаги, расстояние, пульс и сон. Банк может предлагать клиентам использовать фитнес-браслеты и предоставлять доступ к агрегированным данным о физической активности клиента. Это позволяет банку получать информацию о здоровье и образе жизни клиента, что может

быть полезным при принятии решений о предоставлении финансовых услуг, персонализации предложений от партнёров и самой эко-системы банка. Анализ данных с устройств IoT может помочь банкам лучше понять клиентов, их потребности и потенциальные финансовые возможности [20].

Комбинированный анализ этих различных источников цифрового следа позволяет банкам создавать более точные и персонализированные профили клиентов, а также прогнозировать их поведение и потребности. Однако при сборе и анализе цифрового следа необходимо соблюдать принципы конфиденциальности, этики и законодательства в области защиты данных, чтобы обеспечить сохранность личной информации клиентов и защитить их права на приватность [2].

Таким образом, сбор цифрового следа клиентов банка представляет собой многомерный процесс, включающий анализ данных из социальных медиа, поисковых запросов, транзакций, поведения в интернете, данных с устройств IoT и других источников. Эти методы позволяют банкам получить глубокое понимание клиентов, их предпочтений и потребностей, что способствует точному принятию обоснованных решений в сфере финансовых услуг [10].

2 Практическое применение методов анализа

2.1. Обзор используемых методов анализа цифрового следа клиентов банка

Анализ цифрового следа обычно осуществляется с помощью собственных систем и инструментов, разработанных внутри банка или с привлечением специализированных консультантов и экспертов в области аналитики данных и машинного обучения [2].

Банки могут создавать собственные аналитические платформы, разрабатывать скоринговые модели, использовать инструменты машинного обучения и аналитики данных для анализа цифрового следа клиентов. Такие системы и инструменты обычно адаптированы под уникальные потребности и требования каждого конкретного банка [10].

Существует несколько готовых решений для для анализа цифрового следа клиентов банка. Давайте поподробнее рассмотрим их.

FICO Score – это решение, разработанное компанией FICO, которое предоставляет скоринговую модель для оценки кредитоспособности клиентов. Это одно из наиболее широко используемых решений в банковской отрасли. FICO используется подавляющим большинством банков и организаций, предоставляющих кредиты, и основывается на досье потребительских кредитов трёх национальных кредитных бюро: Experian, Equifax и TransUnion. Поскольку кредитное досье потребителя может содержать различную информацию в каждом из бюро, баллы FICO могут различаться в зависимости от того, какое бюро предоставляет информацию FICO для генерирования баллов. Баллы FICO используются во многих кредитных решениях, принимаемых в США. Хоть заёмщики могут объяснить негативные пункты в своём кредитном отчёте, многие кредиторы отказывают в кредите людям с низким баллом FICO.

Плюсы FICO Score:

+ Объективность. Основан на анализе объективных факторов, таких как история кредитования, задолженности, платежная дисциплина и продолжительность кредитной истории. Это позволяет оценить кредитоспособность клиента без субъективного вмешательства.

+ Широкое использование. Широко применяется финансовыми учреждениями и кредиторами во многих странах. Это означает, что имея хороший FICO Score, заёмщик может иметь доступ к более выгодным кредитным условиям, низким процентным ставкам и большему количеству кредитных предложений.

+ Прозрачность. Предоставляет заёмщику информацию о том, каким образом его кредитный рейтинг был рассчитан. Это помогает заёмщику понять, какие факторы положительно или отрицательно влияют на его кредитную историю.

Минусы FICO Score:

- Ориентация на кредитную историю. Людям без кредитной истории может быть сложно получить высокий рейтинг, даже если они имеют стабильные финансовые показатели.

- Неучет индивидуальных обстоятельств. Не учитываются индивидуальные обстоятельства заёмщика, такие как его доход, семейное положение или тип работы. Это может привести к некоторой неполноте в оценке кредитоспособности клиента.

В результате мы получаем прозрачный инструмент для анализа данных клиентов банка, ориентированный на постоянных клиентах.

SAS Credit Scoring – система для оценки кредитоспособности физического или юридического лица, которая анализирует данные потенциального заёмщика и в ответ выдаёт результат – стоит ли предоставлять кредит. С ними работают такие фирмы как - «Nova Credit» и «airstar». Включает набор методик и инструментов, которые позволяют предсказать поведенческую модель клиентов, определить вероятность ухода клиентов в дефолт. В систему также входят средства обработки и хранения

информации, формирования витрин данных, широкий набор аналитических инструментов для построения и анализа моделей кредитного скоринга и обширная система отчётности для решения задач оценки работоспособности скоринговых моделей и состояния кредитного портфеля.

Плюсы SAS Credit Scoring:

+ Гибкость и настраиваемость. Широкие возможности для настройки моделей оценки риска в соответствии с требованиями конкретного банка или финансового учреждения. Это позволяет создавать модели, оптимально соответствующие специфическим потребностям и особенностям кредиторов.

+ Мощные аналитические возможности. Оснащён разнообразными аналитическими инструментами и алгоритмами, которые позволяют более точно оценивать кредитный риск. Сюда входят статистический анализ, прогнозирование и другие методы, способствующие повышению точности моделей скоринга.

+ Интеграция данных. Возможность интеграции данных из различных источников, что позволяет более полно и точно оценить кредитную историю заёмщика. Это включает данные о финансовом положении, истории заёмов, платёжных привычках, кредитных запросах и других факторах, влияющих на кредитоспособность.

Минусы SAS Credit Scoring:

- Высокая сложность и требования к экспертизе. Реализация и использование SAS Credit Scoring требует определённого уровня экспертизы в области аналитики данных и программирования. Необходимость обучения персонала и выделения ресурсов для работы с SAS Credit Scoring может быть вызовом для организаций с ограниченными ресурсами или недостаточной экспертизой.

- Высокие затраты. SAS Credit Scoring является коммерческим решением, и его использование может включать значительные затраты на лицензирование, развёртывание и обслуживание системы. Это может быть

фактором, который затрудняет доступность и применение SAS Credit Scoring для некоторых малых и средних финансовых учреждений.

- Зависимость от качества данных. Точность и надёжность моделей скоринга в SAS Credit Scoring зависят от качества входных данных. Если данные о кредиторах неполные, неточные или устаревшие, то результаты оценки кредитоспособности могут быть искажёнными.

- Ограниченная адаптивность. Менее гибкий в адаптации к изменяющимся требованиям и условиям рынка. Если модели скоринга не обновляются или не приспосабливаются к новым трендам и ситуациям, то они могут стать менее релевантными и неэффективными в оценке кредитного риска.

В результате получаем точный, но ресурсозатратный метод анализа данных клиентов банка, который требует актуализированных данных, поддержке экспертов в области финансов и программирования. Подходит лишь для крупных банков и организаций.

IBM Watson Analytics – платформа IBM Watson, которая предоставляет возможности анализа данных и машинного обучения. Она может быть использована для анализа цифрового следа клиентов банка и разработки моделей скоринга и прогнозирования. Продвигает новую архитектуру данных, чтобы задействовать ИИ. Благодаря подходу на основе структуры данных компании могут подключать нужных людей к нужным данным в нужное время для повышения оперативности, прогнозирования результатов и персонализации опыта.

Плюсы IBM Watson Analytics:

+ Простота использования. Интуитивно понятный и лёгкий в использовании интерфейс, что делает его доступным для пользователей без специальных навыков аналитики данных. Благодаря этому, даже неопытные пользователи могут проводить анализ данных и получать ценные инсайты.

+ Визуализация данных. Платформа предлагает широкий набор инструментов визуализации данных, включая графики, диаграммы и

интерактивные панели управления. Это позволяет легко представлять данные в понятном и наглядном формате, что упрощает коммуникацию и визуальное представление результатов анализа.

+ Интеграция с другими инструментами. Возможность интегрироваться с другими инструментами и системами, что обеспечивает более полный и всесторонний анализ данных. Например, данные из различных источников могут быть объединены и использованы для создания комплексных моделей анализа.

Минусы IBM Watson Analytics:

- Ограничения по объёму данных. При работе с большими объёмами данных IBM Watson Analytics может столкнуться с ограничениями производительности. Обработка и анализ больших наборов данных может потребовать значительного времени и ресурсов.

- Зависимость от подключения к облаку. IBM Watson Analytics является облачным сервисом, что означает, что для работы с платформой требуется постоянное подключение к интернету. Это может быть проблематично в случае ограниченной доступности интернета или если у вас есть конфиденциальные данные, которые необходимо хранить внутри предприятия.

- Ограниченные возможности настраиваемости. В некоторых случаях пользователь может столкнуться с ограничениями в настраиваемости инструментов IBM Watson Analytics. Определенные аналитические методы или параметры могут быть недоступны для настройки в соответствии с конкретными потребностями или бизнес-правилами организации.

В результате получаем простой в использовании инструмент с возможность добавления инструментов и графическим сопровождением для анализа данных, но с зависимостью от подключения к облаку, что не всегда бывает удобно.

Oracle Financial Services Analytical Applications (OFSAA) – набор аналитических приложений Oracle, специально разработанных для

банковского сектора. Включает в себя решения для анализа цифрового следа клиентов и принятия решений на основе данных.

Плюсы OFSAA:

+ Комплексные аналитические возможности. Широкий набор аналитических инструментов и моделей, которые позволяют финансовым учреждениям проводить сложный анализ рисков, прогнозирование, моделирование и другие задачи в области аналитики.

+ Интеграция с другими системами Oracle. Oracle Financial Services Analytical Applications тесно интегрирована с другими системами Oracle, такими как Oracle Database, Oracle Business Intelligence и другими. Это позволяет легко обмениваться данными и использовать уже существующую инфраструктуру Oracle в организации.

+ Конфигурируемость и настраиваемость. Oracle Financial Services Analytical Applications предоставляет возможности настройки и конфигурирования, позволяя финансовым учреждениям адаптировать платформу под свои уникальные требования и бизнес-правила.

Минусы OFSAA:

- Сложность внедрения и обучения. Oracle Financial Services Analytical Applications - это мощная платформа, требующая определённого уровня экспертизы для её внедрения и использования. Обучение сотрудников и настройка платформы могут потребовать значительных ресурсов и времени, что может стоить больших денег.

- Высокая стоимость. Использование Oracle Financial Services Analytical Applications может быть дорогим для некоторых организаций. Приобретение лицензий, обслуживание и поддержка платформы могут быть значительными затратами.

- Ограниченные интеграционные возможности. В отдельных случаях OFSAA может иметь ограниченные возможности интеграции с другими системами, не являющимися продуктами Oracle. Это может создавать

сложности в обмене данными и взаимодействии с другими бизнес-платформами.

В результате мы получаем систему для анализа цифрового следа, которая отлично работает с другими продуктами Oracle, образуя свою эко-систему.

RapidMine – инструмент для анализа данных и машинного обучения, который может быть использован для анализа цифрового следа клиентов банка. Предоставляет возможности создания и применения моделей скоринга и прогнозирования.

Плюсы RapidMiner:

+ Интуитивно понятный интерфейс. Графический интерфейс с перетаскиванием элементов, что делает процесс анализа данных и создания моделей более доступным для пользователей без глубоких знаний программирования.

+ Обширный выбор алгоритмов. Платформа включает широкий набор алгоритмов машинного обучения и статистических методов, что позволяет исследователям данных выбирать наиболее подходящий под их задачи.

+ Гибкость и расширяемость. RapidMiner предлагает возможность создания собственных операторов и расширений, что позволяет пользователям адаптировать платформу под свои уникальные требования и интегрировать собственные алгоритмы или функциональность.

+ Поддержка командной работы. RapidMiner обеспечивает возможность совместной работы и обмена данными между различными пользователями, что упрощает коллаборацию и обеспечивает более эффективное использование платформы в командных проектах.

Минусы RapidMiner:

- Ограниченная бесплатная версия. Бесплатная версия RapidMiner имеет некоторые ограничения по функциональности и объёму данных, что может ограничить возможности исследователей данных.

- Не подходит для решения сложных задач с большим объёмом данных. В случае сложных задач или больших объёмов данных, использование RapidMiner может потребовать большого объёма вычислительных ресурсов и времени.

- Необходимость знания алгоритмов и статистики: Хотя интерфейс RapidMiner интуитивно понятен, для получения наилучших результатов всё же требуется понимание основных алгоритмов машинного обучения и статистических методов.

- Ограниченные интеграционные возможности. RapidMiner может иметь ограниченные возможности интеграции с некоторыми внешними системами и базами данных, что может потребовать дополнительной настройки или разработки.

В результате мы получаем решение для простых задач анализа цифрового следа с упрощённым интерфейсом и для командной работы, но требующее знаний алгоритмов и статистики.

Представленные методы решения меня не удовлетворяют, так как эти методы стоят дорого, нет гарантий что поддержка не прекратиться в связи с разными обстоятельствами и существует посредничество между клиентами банка и самим банком в виде компании, которая предоставляет методы анализа, тем самым подвергая опасности конфиденциальность данных клиентов.

2.2. Разработка собственного метода анализа данных

Говоря о собственной разработке метода анализа данных нужно учесть несколько критериев: визуальное представление анализа - для удобства чтения данных и итогов работы, точность обработки системы скоринга, составление базы данных и возможность загружать базы данных для дальнейшей работы с ними.

Для визуального представления работы с анализом данных я выбрал библиотеки Seaborn и Matplotlib. Matplotlib - это комплексная библиотека для создания статических, анимированных и интерактивных визуализаций на языке Python [8]. Он позволяет создавать графики, интерактивные фигуры, которые можно масштабировать, настраивать свой собственный визуальный стиль и макет. Seaborn - это библиотека визуализации данных на языке Python, основанная на matplotlib, хорошо дополняющая возможности этой библиотеки. Она предоставляет высокоуровневый интерфейс для построения привлекательных и информативных статистических графиков [16]. С помощью них я буду включать в наш анализ графы, которые помогут представить объем информации о клиентах с которой я буду работать, тепловую карту - это графическое представление данных в виде матрицы, в которой значения представленных элементов отображаются с использованием цветовой шкалы. Тепловая карта позволяет визуально выявить зависимости, паттерны и структуру данных. В контексте анализа данных, тепловые карты часто используются для визуализации корреляции между признаками. Каждый элемент матрицы представляет собой коэффициент корреляции между соответствующими признаками. Значения корреляции отображаются цветом, где различные оттенки указывают на разную степень корреляции. Например, более интенсивные цвета могут указывать на более сильную положительную или отрицательную корреляцию, тогда как менее интенсивные цвета могут указывать на отсутствие корреляции. Также я буду использовать матрицу ошибок - она является таблицей, которая показывает сводные результаты классификации модели. Она позволяет визуализировать и анализировать результаты предсказания модели, сравнивая фактические и предсказанные значения:

-True Positive (TP) - количество верно предсказанных положительных примеров.

-False Negative (FN) - количество ошибочно отнесенных к негативному классу примеров.

-False Positive (FP) - количество ошибочно отнесенных к положительному классу примеров.

-True Negative (TN) - количество верно предсказанных негативных примеров.

Точность анализа данных вычисляется сравнением результатов предсказаний модели с фактическими значениями. В контексте классификации, точность (accuracy) является одной из метрик для оценки производительности модели [6]. Она показывает, какая часть предсказаний модели совпадает с фактическими метками классов.

Для вычисления точности, необходимо сравнить каждое предсказанное значение с соответствующим фактическим значением и подсчитать количество правильных предсказаний. Затем это количество правильных предсказаний делится на общее количество предсказаний.

Формула для вычисления точности (accuracy) выглядит следующим образом:

$$A=(n_{TP}+n_{TN})/N \quad (1)$$

где A - точность;

$_{TP}$ - количество верно предсказанных положительных примеров;

n_{TN} - количество верно предсказанных отрицательных примеров;

N - общее количество всех значений.

В целях обеспечения конфиденциальности исследования, базы данных будут сформированы на основе полностью анонимной информации о клиентах, полученной из числа следов, оставленных ими в процессе взаимодействия с банком [11].

Подключение необходимых для исследования и анализа обезличенных баз данных клиентов банка будет осуществляться в программном коде при их загрузке.

Всего потребуется две базы данных - тренировочной, с готовыми решениями по одобрению кредита и тестовой, клиенты, которые ещё ожидают решения. Когда мы говорим о тренировочной базе данных, мы

имеем в виду набор данных, на котором модель будет "учиться" и настраивать свои внутренние параметры, чтобы делать предсказания. Эти данные представляют собой примеры, которые модель будет использовать для поиска закономерностей и образцов в данных. Тестовая база данных должна быть независимой от тренировочной базы данных и представлять реальные условия или сценарии, на которых модель будет применяться. Она используется для проверки точности и надёжности модели, позволяя оценить, насколько хорошо модель может обобщить свои знания и предсказывать результаты на новых данных [3]. При использовании тестовой базы данных модель делает предсказания на основе входных данных, и эти предсказания сравниваются с известными правильными ответами. Сравнение результатов позволяет оценить точность модели и понять, насколько хорошо она работает на новых данных [1].

Благодаря описанным выше инструментам, базам данных из которой я буду брать информацию о клиентах банка, библиотекам для Python и формуле вычисления точности - я могу построить свою собственную модель для решения задачи анализа цифрового следа клиентов банка и протестировать её работоспособность.

3 Результаты исследования

3.1. Исходные данные

Исходные данные были предоставлены банком в обезличенном виде. В тренировочной базе данных указаны такие критерии как:

- Label. Атрибут представляет собой маркер, указывающий на категорию, к которой относится клиент. Где, 0 означает отказ в выдаче кредита, а 1 - одобрение кредита;

- Age. Этот атрибут указывает полный возраст клиента;

- Language. Данный атрибут указывает на количество языков, которые знает клиент;

- Sex. Этот атрибут указывает на пол клиента;

- Marital. Данный атрибут отражает семейное положение клиента;

- Has_Credit. Этот атрибут указывает на наличие или отсутствие у клиента кредитной истории или текущих кредитов;

- Field. Данный атрибут указывает на сферу деятельности или профессиональную область клиента;

- Month_of_birth. Этот атрибут указывает на месяц рождения клиента. Значение представлено целым числом от 1 до 12;

- Day_of_birth. Данный атрибут указывает на день рождения клиента. Значение представлено целым числом от 1 до 31;

- Region. Этот атрибут указывает на место проживания клиента;

- Number_of_credits. Данный атрибут указывает на количество кредитов или займов, которыми обладает клиент в настоящее время и в прошлом. Значение представлено целым числом;

- Linked_cards. Этот атрибут указывает на наличие у клиента связанных или дополнительных банковских карт;

- Score_level. Данный атрибут указывает на уровень скоринга, который был присвоен клиенту на основе анализа его данных;

- Score_class. Этот атрибут может указывает на классификацию, к которой относится клиент на основе его скорингового результата;

- Score_point. Данный атрибут отражает величину скоринговых баллов, набранных клиентом на основе анализа его данных;

- Changed_phone_number. Этот атрибут указывает, был ли изменён клиентом его номер телефона или контактная информация;

- INPS_mln_sum и INPS_yes_no. Эти атрибуты нам не понадобятся, так-как не вносят вклад в классификацию.

В тестовой базе данных присутствует всё те же атрибуты.

3.2. Создание анализа

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Рисунок 2 - Подключение библиотек

NumPy — это библиотека Python, которую применяют для математических вычислений: начиная с базовых функций и заканчивая линейной алгеброй [12].

Полное название библиотеки — Numerical Python extensions, или «Числовые расширения Python». Также данная библиотека предоставляет базовые методы для манипуляции с большими массивами и матрицами, чем мы и будем пользоваться для хранения данных и работы с ними [14].

Pandas — это библиотека Python для обработки и анализа структурированных данных, её название происходит от «panel data» («панельные данные»). Панельными данными называют информацию,

полученную в результате исследований и структурированную в виде таблиц. Для работы с такими массивами данных и создан Pandas [15].

Библиотеку matplotlib она понадобится для визуализации полученных функций.

Seaborn — это библиотека для создания статистических графиков на Python. Она основываетя на matplotlib и тесно взаимодействует со структурами данных pandas. Она нам понадобится, так как в matplotlib нельзя использовать данные, созданные с помощью библиотеки pandas [16].

```
data = pd.read_csv('data_train.csv')
data.drop(['INPS_mln_sum', 'INPS_yes_no'], axis=1, inplace=True)
data.head()
```

	label	Age	Language	Sex	Marital	Has_Credit	Field	Month_of_birth	Day_of_birth	Region	Number_of_credits	Linked_cards	Score_level	Score_class	Score_point	Changed_phone_number
0	0	34	1	2	6	2	13	12	1	12	1	0	0	0	-	1
1	0	38	1	1	5	1	10	7	1	13	1	2	0	0	-	1
2	0	35	1	2	4	2	9	8	1	13	4	1	0	0	-	1
3	0	27	1	1	5	2	13	7	1	12	1	2	0	0	-	1
4	0	32	1	2	4	2	10	7	1	13	3	1	0	0	-	1

Рисунок 3 - Код программы и часть результата его выполнения

data = pd.read_csv('data_train.csv') - подключение тренировочной базы данных для скоринговой модели.

data.drop([‘INPS_mln_sum’, ‘INPS_yes_no’], axis=1, inplace=True) - убираем столбцы ‘INPS_mln_sum’ и ‘INPS_yes_no’, axis=1- указывает, что мы хотим удалить столбцы, а не строки. Значение 1 указывает на ось столбцов, inplace = True - указывает, что изменения должны быть применены к исходному DataFrame data, а не создан новый DataFrame.

data.head() - показ первых 5 строк датасета.

```
data.info()
```

Рисунок 4 - Команда вывода информации

data.info() - информация по атрибутам датасета. Метод info() напечатал тип этого объекта, диапазон, столбцы, количество записей в каждом столбце, если столбцы не равны нулю.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8707 entries, 0 to 8706
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   label                 8707 non-null   int64
 1   Age                   8707 non-null   int64
 2   Language              8707 non-null   int64
 3   Sex                   8707 non-null   int64
 4   Marital               8707 non-null   int64
 5   Has_Credit            8707 non-null   int64
 6   Field                 8707 non-null   int64
 7   Month_of_birth        8707 non-null   int64
 8   Day_of_birth          8707 non-null   int64
 9   Region                8707 non-null   int64
 10  Number_of_credits     8707 non-null   int64
 11  Linked_cards          8707 non-null   int64
 12  Score_level           8707 non-null   int64
 13  Score_class           8707 non-null   int64
 14  Score_point           8707 non-null   object
 15  Changed_phone_number  8707 non-null   int64
dtypes: int64(15), object(1)
memory usage: 1.1+ MB
```

Рисунок 5 - Результат вывода data.info()

```
df = data

df = df.replace({'-':0})
```

```
df['Score_point']
```

```
0         0
1         0
2         0
3         0
4         0
       ...
8702      0
8703      0
8704      0
8705    237
8706    263
Name: Score_point, Length: 8707, dtype: object
```

Рисунок 6 - Замена пропущенные значения

replace() - меняем все “-” пропущенные значения из датасета в “0”.

df[‘score_point’] - проверям, поменялось ли.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8707 entries, 0 to 8706
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   label                 8707 non-null   int64
 1   Age                   8707 non-null   int64
 2   Language              8707 non-null   int64
 3   Sex                   8707 non-null   int64
 4   Marital               8707 non-null   int64
 5   Has_Credit            8707 non-null   int64
 6   Field                 8707 non-null   int64
 7   Month_of_birth        8707 non-null   int64
 8   Day_of_birth          8707 non-null   int64
 9   Region                8707 non-null   int64
 10  Number_of_credits     8707 non-null   int64
 11  Linked_cards          8707 non-null   int64
 12  Score_level           8707 non-null   int64
 13  Score_class           8707 non-null   int64
 14  Score_point           8707 non-null   int64
 15  Changed_phone_number  8707 non-null   int64
dtypes: int64(16)
memory usage: 1.1 MB
```

Рисунок 7 - Меняем все типы данных на числовой integer

columns = df.columns - присваиваем значение переменной

for c in list (columns): df[c] = df[c].astype(‘int64’) - цикл, отвечающий за присваивание каждому атрибуту тип integer

df.info() - проверяем, присвоился ли тип данных.

```
plt.subplots(figsize=(10,7))
sns.heatmap(df.corr())
```

Рисунок 8 - Графическое обозначение корреляции атрибутов

plt.subplots(figsize=(10,7)) - размер сетки графика, 10 рядов по 7 элементов. (отвечает за размер графика)

sns.heatmap(df.corr()) - Тепловая карта (Heatmap) – способ визуализации статистических данных с помощью цветовой палитры. .corr() - отвечает за корреляцию атрибутов конкретного датасета.

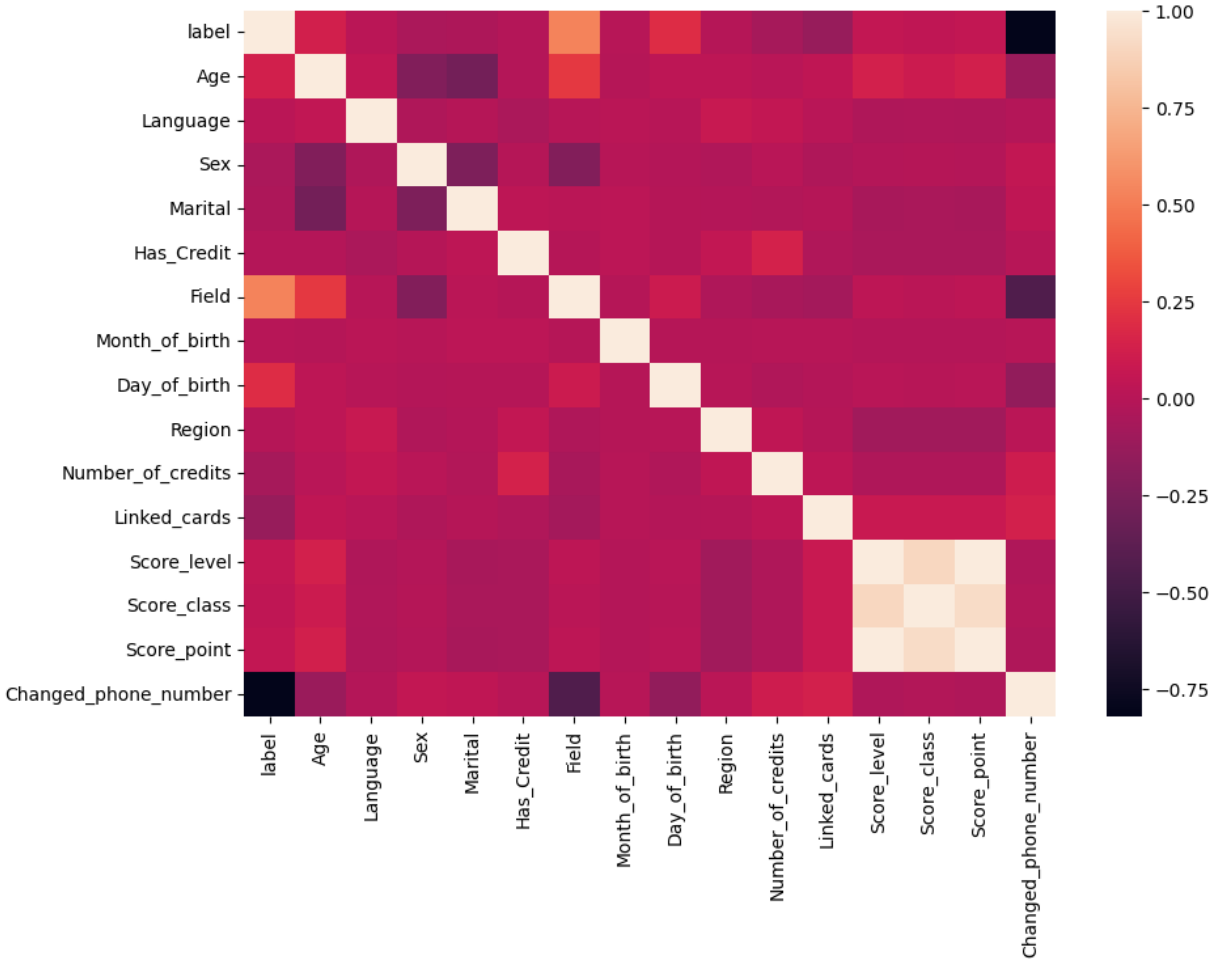

Рисунок 9 - Корреляции результат кода

```
counting_0_1 = df.pivot_table(columns=['label'], aggfunc='size')
print(counting_0_1)
```

Рисунок 10 - Какие и сколько атрибут label имеет значений

Важным атрибутом или целевой переменной для сравнения датасетов является label - решение о выдаче кредитов.

.pivot_table - преобразует базу данных в сводную таблицу.aggfunc - мы можем произвести подсчёт количества. ‘size’ - общее количество значений.

print() - вывод результата в консоль.

```
label
0     670
1    8037
dtype: int64
```

Рисунок 11 - Результат вывода. 670 отказов и 8037 одобрений по кредиту

```
y = df['label']
X = df.drop(['label'], axis=1, inplace=False)
```

Рисунок 12 - Разделение данных

На этом этапе мы подвергаем разделению данных на целевую переменную (y) и набор признаков (X). В df['label'] представлен столбец с целевой переменной, который извлекается из исходного датафрейма df с помощью df['label'].

Далее, df.drop(['label'], axis=1, inplace=False) удаляет столбец 'label' из исходного датафрейма и возвращает новый датафрейм (X), который содержит все остальные признаки.

```
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import RandomUnderSampler
from collections import Counter
```

Рисунок 13 - Подключение библиотек

RandomOverSampler из модуля imblearn.over_sampling - класс используется для случайной повторной выборки (oversampling) данных с целью балансировки классов путём увеличения количества образцов в классе.

RandomUnderSampler из модуля imblearn.under_sampling - класс используется для случайной выборки (undersampling) данных с целью балансировки классов путём уменьшения количества образцов в мажоритарном классе.

Counter из модуля collections - этот класс используется для подсчёта количества экземпляров каждого класса в целевой переменной.

```
ros = RandomOverSampler()

X_ros, y_ros = ros.fit_resample(X, y)
```

Рисунок 14 - RandomOverSampler()

RandomOverSampler - выполняет случайную повторную выборку (oversampling) данных с помощью метода RandomOverSampler из библиотеки imbalanced-learn. Алгоритм oversampling используется для балансировки классов в несбалансированных наборах данных путем увеличения количества образцов в классе (в данном случае класс с меткой 1) до уровня мажоритарного класса (в данном случае класс с меткой 0).

.fit_resample (X, y) - Производится повторная выборка наборов признаков (X) и соответствующих целевых переменных (y) с помощью метода fit_resample(). Он адаптирует оба набора данных X и y таким образом, чтобы классы были сбалансированы. Результатом являются новые наборы признаков (X_ros) и целевых переменных (y_ros) с увеличенным количеством образцов в классе.

```
] from sklearn.model_selection import train_test_split

  X_train, X_test, y_train, y_test = train_test_split(X_ros,y_ros, test_size = 0.30, random_state = 101)

  print(X_train.shape, X_test.shape)

  print(y_train.shape, y_test.shape)
```

Рисунок 15 - train_test_split из модуля sklearn.model_selection для разделения данных на обучающий и тестовый наборы

X_ros и y_ros - представляют собой наборы признаков и соответствующую целевую переменную, разделение на обучающий и тестовый наборы.

test_size - размер тестового набора, заданный в виде десятичной доли или целого числа. 0.30 означает, что 30% данных будут отложены для тестирования.

random_state - задаёт начальное значение генератора псевдослучайных чисел, чтобы обеспечить повторяемость результатов.

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report,confusion_matrix

model = SVC()
model.fit(X_train,y_train)
svm_pred = model.predict(X_test)

# Evaluating Model
print(accuracy_score(y_test, svm_pred).round(3))
print(confusion_matrix(y_test, svm_pred))
print(classification_report(y_test, svm_pred))
```

```
0.93
[[2354   42]
 [ 297 2130]]
              precision    recall  f1-score   support

           0       0.89      0.98      0.93      2396
           1       0.98      0.88      0.93      2427

    accuracy                           0.93      4823
   macro avg       0.93      0.93      0.93      4823
weighted avg       0.93      0.93      0.93      4823
```

Рисунок 16 - SVC

В представленном коде выполняется обучение модели SVM (Support Vector Machine) с использованием класса SVC из модуля sklearn.svm. Затем модель используется для предсказания классов для тестовых данных.

Создаётся экземпляр модели SVC: model = SVC()

Обучение модели на обучающих данных: model.fit(X_train, y_train)

Предсказание классов для тестовых данных:

Вызов метода predict для модели с передачей X_test: svm_pred = model.predict(X_test)

Оценка модели:

Оценка точности модели с использованием метрики accuracy_score: print(accuracy_score(y_test, svm_pred).round(3))

Вывод матрицы ошибок (confusion matrix) с использованием функции confusion_matrix: print(confusion_matrix(y_test, svm_pred))

Вывод отчёта о классификации (classification report) с использованием функции classification_report: print(classification_report(y_test, svm_pred))

Метрика accuracy_score позволяет оценить точность модели, матрица ошибок показывает количество верно и неверно классифицированных экземпляров для каждого класса, а отчёт о классификации предоставляет информацию о метриках precision, recall, f1-score и support для каждого класса.

```
from sklearn.tree import DecisionTreeClassifier

regressor = DecisionTreeClassifier(random_state =0)
regressor.fit(X_train,y_train)
Dectree_pred = regressor.predict(X_test)

print(accuracy_score(y_test, Dectree_pred).round(3))
print(confusion_matrix(y_test, Dectree_pred))
print(classification_report(y_test, Dectree_pred))
```

```
0.999
[[2396    0]
 [   5 2422]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      2396
           1       1.00      1.00      1.00      2427

    accuracy                           1.00      4823
   macro avg       1.00      1.00      1.00      4823
weighted avg       1.00      1.00      1.00      4823
```

Рисунок 17 - DecisionTreeClassifier

from sklearn.tree import DecisionTreeClassifier - построен на основе решающих правил вида «если, то», упорядоченных в виде дерева и имеющую иерархическую систему.

regressor = DecisionTreeClassifier(random_state = 0) - random_state (во всех функциях и методах из SciKit-Learn) воспроизводит случайные значения. Приравнивание к нулю, в этом случае, не даёт поменять значения.

regressor.fit(X_train,y_train) - обучение модели на данных.

Dectree_pred = regressor.predict(X_test) - Вызов метода predict для модели с передачей X_test: Dectree_pred = regressor.predict(X_test)

print(accuracy_score(y_test, Dectree_pred).round(3)) - оценка точности дерева.

print(confusion_matrix(y_test, Dectree_pred)) - вывод матрицы ошибок.

print(classification_report(y_test, Dectree_pred)) - вывод отчёта.

На данном этапе мы проверили работу двух алгоритмов - Decision Tree Classifier и SVC, оба показали себя хорошо, но чтобы протестировать их работу нам нужно загрузить тестовую базу данных, которая покажет, какой из двух алгоритмов сработает лучше.

```
#Датасет для теста
df2 = pd.read_csv('data_test.csv')
df2.drop(['INPS_mln_sum', 'INPS_yes_no'], axis=1, inplace=True)
df2.head()
```

	label	Age	Language	Sex	Marital	Has_Credit	Field	Month_of_birth	Day_of_birth	Region	Number_of_credits	Linked_cards	Score_level	Score_class	Score_point
0	0	40	1	2	4	1	0	3	1	12	2	1	0	0	-
1	0	36	2	2	4	1	0	5	1	13	1	2	0	0	-
2	0	31	1	2	4	1	0	7	1	13	1	1	0	0	-
3	0	29	1	2	4	1	0	1	1	13	1	1	0	0	-
4	0	38	1	2	4	1	0	10	1	13	1	2	4	2	318

Рисунок 18 - Подключение тестовой базы данных

Подключение тестовой базы данных проходит также, как и тренировочной, с одним отличием - название меняем на df2.

```
df2 = df2.replace({'-':0})

df2['Score_point'].head()
```

```
0      0
1      0
2      0
3      0
4    318
Name: Score_point, dtype: object
```

```
New_Y = df2['label']
New_X = df2.drop(['label'], axis=1, inplace=False)

New_X.head()
```

	Age	Language	Sex	Marital	Has_Credit	Field	Month_of_birth	Day_of_birth	Region	Number_of_credits	Linked_cards	Score_level	Score_class	Score_point
0	40	1	2	4	1	0	3	1	12	2	1	0	0	0
1	36	2	2	4	1	0	5	1	13	1	2	0	0	0
2	31	1	2	4	1	0	7	1	13	1	1	0	0	0
3	29	1	2	4	1	0	1	1	13	1	1	0	0	0
4	38	1	2	4	1	0	10	1	13	1	2	4	2	318

Рисунок 19 - Замена пустых ячеек

Убираем пустые ячейки и заменяем их на “0” и присваиваем новые X и Y для df2 DataFrame. Проверяем успешное выполнение команды с помощью .head(). На примере атрибута Score_point мы видим разницу.

```
Dectree_new = regressor.predict(New_X)

print(accuracy_score(New_Y, Dectree_new).round(3))
print(confusion_matrix(New_Y, Dectree_new))
print(classification_report(New_Y, Dectree_new))
```

Рисунок 20 - Проверяем работу алгоритма DecisionTreeClassifier

```
svm_new = model.predict(New_X)

print(accuracy_score(New_Y, svm_new).round(3))
print(confusion_matrix(New_Y, svm_new))
print(classification_report(New_Y, svm_new))
```

Рисунок 21 - Проверяем работу алгоритма SVC

3.3. Визуальное представление

```
import matplotlib.pyplot as plt
term_deposit = np.array([len(data[data['label']== 0]), len(data[data['label'] == 1])])
mylabel = ['нет','да']

myprec = np.array([str(round(term_deposit[0] / len(data['label']) * 100,2)) + '%',
                   str(round(term_deposit[1] / len(data['label'])* 100,2)) + '%'])


plt.pie(term_deposit, labels = myprec)
plt.legend(mylabel)
plt.title('Одобрение кредита')
```

Рисунок 22 - Круговая диаграмма

import matplotlib.pyplot as plt - подключение библиотеки matplotlib для круговой диаграммы

term_deposit = np.array([len(data[data['label']== 0]), len(data[data['label'] == 1])]) - представляет собой массив, содержащий количество записей в данных, где значение "label" равно 0 (нет одобрения кредита) и где значение "label" равно 1 (одобрение кредита).

mylabel = ['нет','да'] - даёт название элементам в списке с метками для круговой диаграммы, где "нет" соответствует значению 0 и "да" соответствует значению 1.

myprec = np.array([str(round(term_deposit[0] / len(data['label']) * 100,2)) + '%', str(round(term_deposit[1] / len(data['label'])* 100,2)) + '%']) - массив, содержащий процентное соотношение количества записей с одобрением и отказом кредита от общего количества записей в данных. Значения процентов округлены до двух знаков после запятой и представлены в виде строки с символом процента.

plt.pie(term_deposit, labels = myprec) - создает круговую диаграмму, используя значения из term_deposit в качестве данных и метки из myprec для отображения процентного соотношения. Каждый сектор диаграммы представляет одну из категорий: "нет" или "да".

plt.legend(mylabel) - добавляет легенду к диаграмме, отображая соответствие между метками и цветами секторов.

plt.title('Одобрение кредита') - задаёт заголовок для круговой диаграммы. В данном случае, заголовок указывает на тему анализа - "Одобрение кредита".

```
def plot_distributions(df):
    plt.figure(figsize=(28,40))
    b = 0
    for i in df.columns:
        b+=1
        plt.subplot(6,6,b)
        plt.hist(df[i])
        plt.title(i)

def plot_idividual( column_name,df= data):
    plt.figure(figsize=(15,10))
    plt.hist(df[column_name])
    plt.title(column_name)

plot_distributions(df)
```

Рисунок 23 - Визуальное представление атрибутов

def plot_distributions(df) - используется для построения гистограмм распределения значений для каждого столбца в базе данных df.

plt.figure(figsize=(28,40)) - создает новую фигуру (график) с указанными размерами 28x40 дюймов.

Переменная b инициализируется 0 для начала счётчика.

Цикл for i in df.columns перебирает каждый столбец в базе данных. b+=1 увеличивает значение b на 1 при каждой итерации цикла для обновления номера подграфика.

plt.subplot(6,6,b) - создает подграфик на общей области размером 6x6 с номером, соответствующим текущему значению b.

plt.hist(df[i]) - строит гистограмму распределения значений столбца i в базе данных df.

plt.title(i) - задает заголовок подграфика, соответствующий имени столбца i.

def plot_individual(column_name, df=data) - используется для построения гистограммы распределения значений для указанного столбца column_name в базе данных df.

plt.figure(figsize=(15,10)) - создает новую фигуру (график) с указанными размерами 15x10 дюймов.

plt.hist(df[column_name]) - строит гистограмму распределения значений для указанного столбца column_name в датафрейме df.

plt.title(column_name) - задаёт заголовок гистограммы, соответствующий имени столбца column_name.

plot_distributions(df) - вызывает функцию plot_distributions() для построения гистограмм распределения значений для каждого столбца в базе данных df. Графики распределения будут отображены на одной общей фигуре размером 28x40 дюймов с использованием подграфиков размером 6x6.

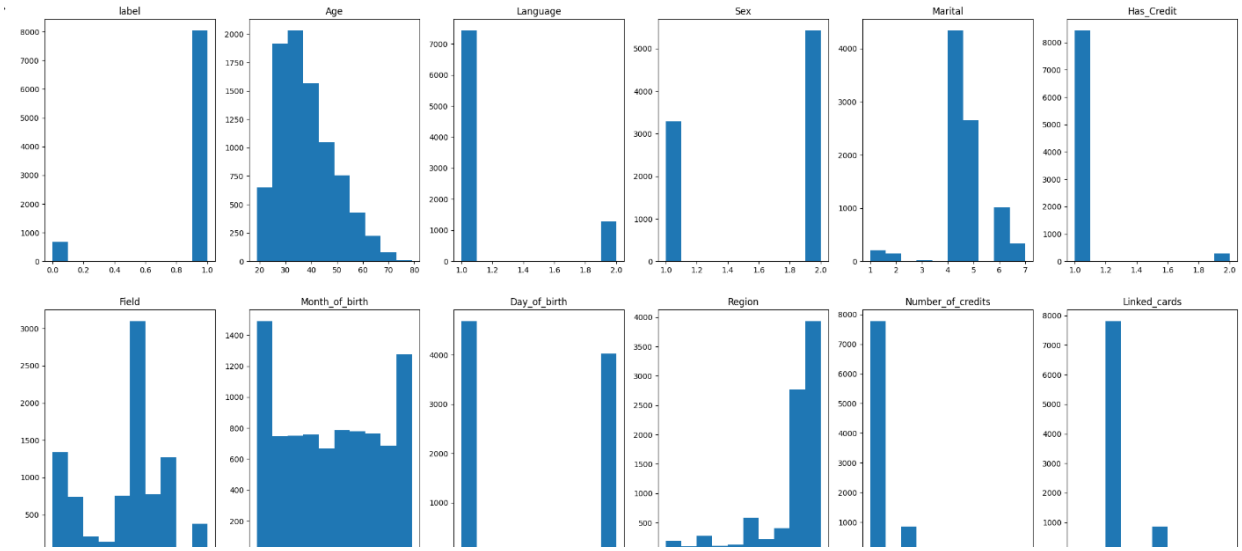

Рисунок 24 - Вывод кода программы

```
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix, annot=True, fmt="d", cmap="Blues")
plt.title("Матрица ошибок")
plt.xlabel("Предсказания")
plt.ylabel("Реальность")
plt.show()
```

Рисунок 25 - Код для графического вывод матрицы ошибок

plt.figure(figsize=(8, 6)) - создаёт новую фигуру для графика с размером 8 на 6 дюймов.

sns.heatmap(confusion_matrix, annot=True, fmt="d", cmap="Blues") - строит тепловую карту на основе матрицы ошибок (confusion_matrix). Аргумент annot=True позволяет выводить значения в ячейках, fmt="d" задает формат чисел в ячейках, а cmap="Blues" устанавливает цветовую схему для тепловой карты.

plt.title("Матрица ошибок") - Устанавливает заголовок для графика.

plt.xlabel("Предсказания") - Устанавливает подпись оси x.

plt.ylabel("Реальность") - Устанавливает подпись оси y.

plt.show() - Отображает график матрицы ошибок.

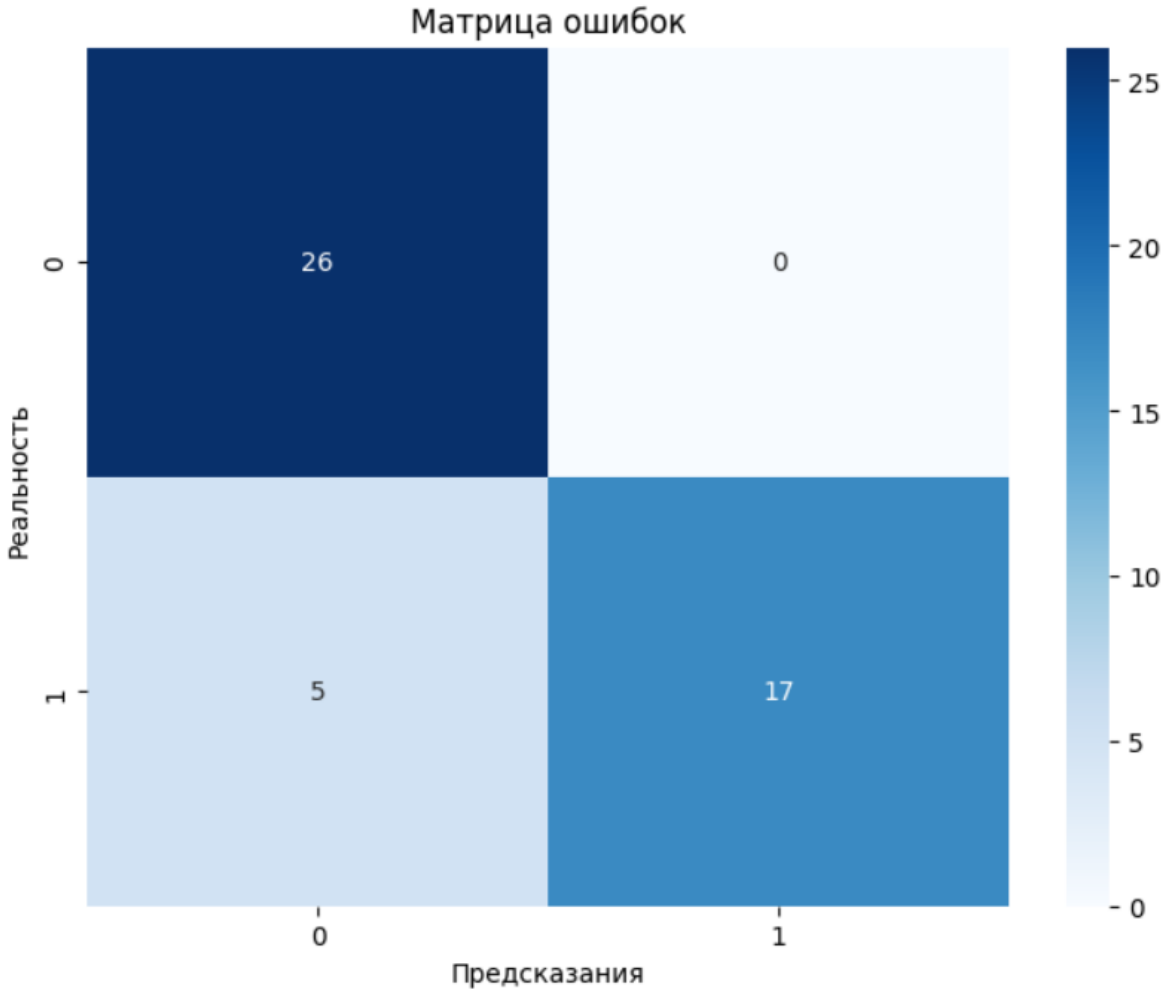

Рисунок 26 - Матрица ошибок на примере алгоритма SVC

3.4. Тестирование

После сборки программы нужно проверить её работоспособность. Во время создания метода анализа данных я использовал алгоритмы Decision Tree Classifier (это непараметрический метод контролируемого обучения, используемый для классификации и регрессии. Цель состоит в том, чтобы создать модель, которая предсказывает значение целевой переменной путем обучения простым правилам принятия решений, выведенным из характеристик данных. Дерево можно рассматривать как кусочно-постоянную аппроксимацию) и SVC (набор контролируемых методов обучения, используемых для классификации , регрессии и обнаружения выбросов). Для оценки эффективности выбранных алгоритмов было проведено тестирование на тестовой выборке. Результаты тестов можно представлены ниже:

```
0.896
[[26  0]
 [ 5 17]]
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        26
           1       1.00      0.77      0.87        22

    accuracy                           0.90        48
   macro avg       0.92      0.89      0.89        48
weighted avg       0.91      0.90      0.89        48
```

Рисунок 27 - Результат SVC

```
1.0
[[26  0]
 [ 0 22]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        26
           1       1.00      1.00      1.00        22

    accuracy                           1.00        48
   macro avg       1.00      1.00      1.00        48
weighted avg       1.00      1.00      1.00        48
```

Рисунок 28 - Результат Decision Tree Classifier

Тестирование является ключевым этапом в разработке программного обеспечения. Оно позволяет проверить работоспособность и качество продукта.

Тестирование алгоритмов DecisionTreeClassifier и SVC позволило оценить их производительность на тестовой выборке. SVC достиг точности 89%, показывая хорошие результаты в предсказании классов. Однако, алгоритм DecisionTreeClassifier показал высочайшую точность в 100%, демонстрируя более точные предсказания. Матрица ошибок и отчёт классификации также подтверждают хорошую производительность обоих алгоритмов.

На основе результатов тестирования можно сделать вывод о том, что алгоритм DecisionTreeClassifier проявил себя как более эффективный в данной задаче классификации.

Заключение

В рамках данной работы был выполнен исследовательский анализ данных и разработана скоринговая модель для оценки кредитоспособности клиентов банка. Были проведены следующие этапы:

- Сбор и предобработка данных. Были получены исходные данные, которые включали информацию о клиентах банка, и проведена их предобработка. Были выполнены операции по очистке данных от пропущенных значений, кодированию категориальных признаков и масштабированию числовых признаков.

- Исследовательский анализ данных. Был проведён анализ данных, включающий исследование распределений признаков, анализ корреляций между признаками и выявление основных факторов, влияющих на кредитоспособность клиентов.

- Разработка скоринговой модели. Были выбраны и обучены различные алгоритмы включая Decision Tree Classifier и Support Vector Classifier (SVC). Была выбрана наилучшая модель на основе метрик оценки производительности.

Исходя из результатов работы, можно сделать вывод, что разработанная скоринговая модель демонстрирует хорошую производительность с высокой точностью. Она может быть эффективно применена банком для автоматизации процесса принятия решений о выдаче кредита. Однако, необходимо отметить, что модель может потребовать дальнейшей настройки и оптимизации с учётом изменений в данных и новых требований банка.

В целом, данная работа демонстрирует применимость различных методов анализа данных цифрового следа и алгоритмов для решения задачи оценки кредитования.

Список используемой литературы

1. Абдрахманов М.И. Pandas. Работа с данными, 2-е издание. Devpractice Team, 2020, с.171-175.

2. Баранова Е.В., Швецов Г.В. МЕТОДЫ И ИНСТРУМЕНТЫ ДЛЯ АНАЛИЗА ЦИФРОВОГО СЛЕДА СТУДЕНТА ПРИ ОСВОЕНИИ ОБРАЗОВАТЕЛЬНОГО МАРШРУТА с. 417-420.

3. Бринк Х., Ричардс Д., Феверолф М. Машинное обучение. – СПб.: Питер, 2017, с.336-350.

4. В. И. Токтарова, Д. А. Семенова, Р. Н. Зарипов. ОЦЕНКА ЭФФЕКТИВНОСТИ ПРОЕКТНОЙ ДЕЯТЕЛЬНОСТИ СТУДЕНТОВ НА ОСНОВЕ ЦИФРОВОГО СЛЕДА.ВЕСТНИК МАРИЙСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА, том 15. № 4. 2021, с. 420-428.

5. Грас Д. Data Science. Наука о данных с нуля. – СПб.: БХВ-Петербург, 2017, с. 336-362.

6. Дейтел П., Дейтел Х. Python: Искусственный интеллект, большие данные и облачные вычисления. – СПБ.: Питер, 2020 , с. 864-879.

7. Журавлева В.В., Маничева А.С., Фещенко А.В., Берестов А.В. Исследование различимости цифровых следов у различных групп школьников на территории Алтайского края // Высокопроизводительные вычислительные системы и технологии. 2020. Т. 1, № 4, с. 121-125.

8. Лутц М. Изучаем Python. – СПБ.: Символ-Плюс, 2009. с. 848 - 854.

9. Петров А.А. "ЦИФРОВОЙ СЛЕД ЧЕЛОВЕКА: ПЛЮСЫ И МИНУСЫ Текст научной статьи по специальности «СМИ (медиа) и массовые коммуникации»", с. 530-540.

10. Чайкина Е.В., Посная Е.А., Чайкин В.Ю. Влияние инновационных технологий на развитие и функционирование банковской индустрии Текст научной статьи по специальности «Экономика и бизнес» 2019, с. 82-86.

11. Bishop A. Protection Methods, 2015. (217-235)

12. Gurruchaga J. R. Python Recipes for Engineers and Scientists // Independently published. 2018 (104)

13. Lourinho L., Kendzierskyj S. Strategy, Leadership, and AI in the Cyber Ecosystem. The Role of Digital Societies in Information Governance and Decision Making , 2021, (159-194).

14. NumPy Documentation [Электронный ресурс] // URL: https://numpy.org/doc/

15. pandas documentation [Электронный ресурс] // URL: https://pandas.pydata.org/docs/

16. seaborn: statistical data visualization [Электронный ресурс] //URL: https://seaborn.pydata.org/

17. Shimonski R., Zenir J. Cyber Reconnaissance, Surveillance and Defense 2015, (183-215)

18. Sponder M. The uses and accuracy of social analytics data and platforms, Public Interest and Private Rights in Social Media, 2012.

19. Williams L.Y. Social Media For Academics. 2012, (175-192).

20. Zhuravleva V.V., Manicheva A.S., Feshchenko A.V., Berestov A.V.Optimization of the algorithm for identifying digital traces of schoolchildren in the Altai Territory // Journal of Physics: Conference Series. 2020.