Санкт–Петербургский государственный университет

Маркин Станислав Витальевич

Выпускная квалификационная работа

Разработка программного модуля для оценки рискованности финансовых транзакций

Уровень образования: бакалавриат
Направление 02.03.02 «Фундаментальные информатика и информационные технологии»
Основная образовательная программа СВ.5003.2016 «Программирование и информационные технологии»
Профиль «Автоматизация научных исследований»

Научный руководитель:
доцент, кафедра компьютерного моделирования и многопроцессорных систем, к.ф. - м.н.
Корхов Владимир Владиславович

Рецензент:
ведущий инженер компании OpenWay,
Ражева Анастасия Александровна

Санкт-Петербург
2020 г.

Содержание

Словарь терминов

Процессинг — это деятельность по обработке информации, используемой при совершении платёжных операций

Эквайринг — возможность для торгового предприятия принимать безналичную оплату за товары и услуги пластиковыми картами. Также в понятие эквайринг входят банковское и технологическое обслуживание – передача и обработка данных клиента.

Код категории продавца (MCC код) — четырехзначный код, отражающий вид деятельности торговой точки в процессе обработки операций оплаты по банкоским картам. Как правило, данный код выдается торговцу банком-эквайером при установке и настройке POS-терминала в кассовом узле на основе информации о роде деятельности предприятия, поданной торговой организацией в банк.

POS-терминал — электронное устройство, способное принимать к оплате платёжные карты различных видов: чиповые, с магнитной полосой и бесконтактные карты, а также другие устройства, имеющие бесконтактное сопряжение.

Импринтер — механическое устройство, применяемое для печати чека при оплате платёжными картами. В устройство вставляется карта, по которой проходит оплата, после чего данные карты отображаются на чеке, который в дальнейшем будет передан банку для проведения платёжной операции.

API (проргаммный интерфейс приложения) — описание способов взаимодействия с приложением. API определяет функциональность, предоставляемую программой, но при этом позволяет абстрогироваться от её реализации и сосредоточиться на других прикладных задачах.

PCI DSS — стандарт безопасности данных при работе с информацией, связанной с платёжными картами. Стандарт представляет собой совокупность требований, необходимых для обеспечения безопасности данных о держателях платёжных карт, которые передаются, хранятся и обрабатываются в информационных инфрострукурах организаций.

Введение

В наше время основным средством совершения покупок являются банковские карты. Они распространены повсеместно и используются во всех странах мира. Благодаря им пользователи могут совершать покупки, снимать наличные, проводить операции в интернете по всему миру. Однако в мире банковских карт мошенники представляют серьезную угрозу как для держателей карт, так и для банков, которые их выпускают.

Согласно отчёту Nilson о ситуации с банковскими картами и мобильными платежами, суммарный объем потерь в результате мошенничества еще в 2016-м достиг $22,8 млрд, что на 4,4% больше, чем в 2015-м [1]. И это число продолжает расти. Это только подтверждает необходимость для банков научиться распознавать мошенничество заранее, еще до того, как оно состоялось.

Однако перед тем, как приступать к поиску решения этой непростой задачи, необходимо разобраться в предметной области. Понять, как живут банки, какой информацией обмениваются, на каком этапе необходимо обнаруживать мошенничество и какие данные для этого нужны.

Глава 1. Постановка задачи

Необходимо исследовать возможности использования алгоритмов анализа данных для оценки рискованности финансовых транзакций в системах процессинга операций, проводимых с использованием платежных карт.

Оценка рискованности финансовых операций может выполняться как на основе анализа данных непосредственно анализируемой финансовой операции, так и с проведением дополнительного анализа исторических данных о совершенных ранее финансовых операциях. Для анализа данных могут использоваться различные методы: системы принятия решений на основе бизнес-правил, системы искусственного интеллекта, включая методы машинного обучения.

В рамках данной работы предполагаются следующие этапы:

- Анализ модели данных о финансовых операциях, характерных для систем процессинга операций, проводимых с использование платежных карт, с целью выбора наиболее значимых параметров для оценки рискованности.

- Формирование модели данных, которая будет использоваться для анализа рискованности финансовых операций.

- Выбор метода анализа данных для оценки рискованности финансовых операций.

- Анализ эффективности выбранной модели данных и методов анализа данных для оценки рискованности финансовых операций.

- Разработка прототипа модуля для оценки рискованности финансовых операций.

Глава 2. Анализ банковских процессов и данных процессинговых центров

2.1 Основные участники

Процесс оплаты по картам является очень сложным и включает в себя множество участников. Рассмотрим, что это за участники, и каковы их роли в этом процессе на примере операции покупки по карте. Кратко этот процесс можно представить в виде схемы, изабраженной на рис. 1

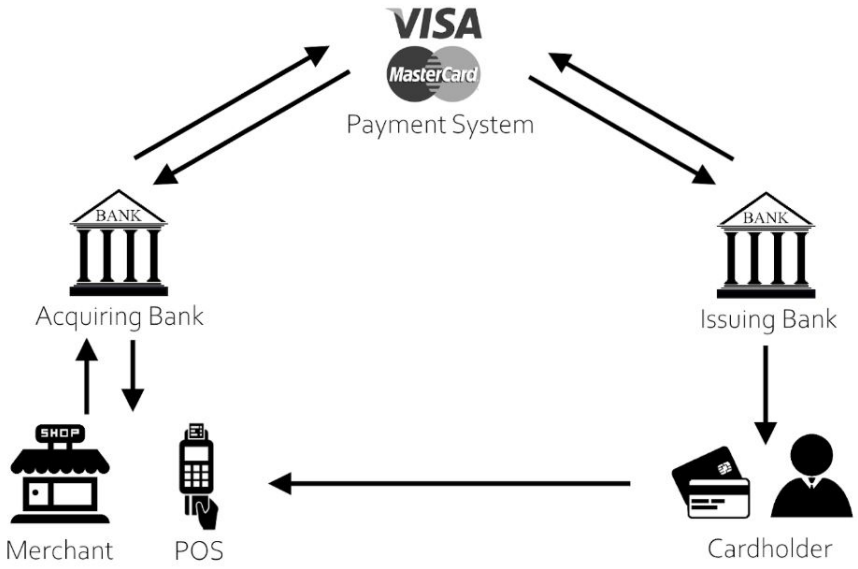

Рис. 1: Схема процесса оплаты по карте

Перечислим основных участников этого процесса:

Банк-эквайер (Acquiring Bank) — кредитная организация, в которой открыт расчетный счет торговца, и которая предоставляет оборудование для эквайринга. Такой банк обязательно должен быть зарегистрирован в международной платежной системе. Несет ответственность за техническую сторону операций покупок по картам, получает комиссию от торговца.

Банк-эмитент (Issuing Bank) — кредитная организация, участник платежной системы, осуществляет выпуск и обслуживание банковских карт. Выступает гарантом выполнения финансовых обязательств, возникающих в ходе использования этих карт держателями.

Платежная система (Payment System) — утвержденный свод правил, условных отношений, методик расчета, общих и локальных нормативов, определяющих порядок проведения финансовых операций и взаиморасчетов между ее участниками. Примеры международных платёжных систем: VISA, MasterCard, American Express и т.п.

Держатель карты (Cardholder) — физическое лицо, на имя которого выпущена пластиковая карта, в том числе физическое лицо-владелец счета либо другое лицо, указанное владельцем счета. Сама карта является собственностью банка при открытии кредитного расчетного счета.

Торговец (Merchant) — юридическое лицо, предоставляющее услуги купли-продажи. Зарегистрирован и обслуживается банком-эквайером.

2.2 Место разрабатываемого модуля

При проведении операции по карте задача банка-эмитента авторизовать операцию или нет. Возможно множество причин для отмены проводимой операции: недостаток средств на счёте, неверные секретные величины (PIN, CVC2 и т.п.), настроенные ограничения на проведение операций. Одной из таких причин может стать подозрение на мошенничество. Разрабатываемый модуль призван заменить или улучшить существующие решения и стать частью процессинговой системы банка-эмитента.

2.3 Анализ доступных данных

Каждый участник процесса проведения операций по картам обладает своими данными, которые могут быть полезны для оценки рискованности транзакции. Однако, так как разрабатываемый модуль будет находиться в системе банка эмитента, необходимо рассмотреть данные, доступные данному участнику процесса проведения операций по картам.

2.3.1 ISO сообщение

Начнем с того, какой информацией обмениваются участники в процессе проведения операции. ISO 8583 — стандарт ISO, описывающий процесс передачи и формат финансовых сообщений (транзакций) систем, обрабатывающих данные банковских платежных карт. Именно в этом формате формируются сообщения, которыми обмениваются торговец с банком-эквайером, банк-эквайер с платежной системой и платежная система с банком-эмитентом. Сообщение ISO 8583 состоит из следующих частей [2]:

- Message Type Indicator (MTI) — Индикатор типа сообщения;
- Одна или несколько битовых карт, указывающих, какие элементы данных присутствуют в сообщении;

- Элементы данных, поля сообщения.

В первую очередь, интерес представляют поля сообщения. Сообщение включает в себя всего 128 полей, однако не все из них могут быть заполнены и далеко не все из них представляют интерес для нашей задачи. Рассмотрим те из них, которые могут быть полезны для решения задачи оценки рискованности транзакций (см. таблицу 1).

Таблица 1: Важные поля ISO сообщения

Номер поля	**Использование поля**
2	PAN - Номер банковской карты
3	Processing code - код, который идентифицирует тип транзакции держателя карты и типы счета держателя карты, на которые влияет транзакция
4	Amount, transaction - сумма совершаемой транзакции
7	Transmission date & time - дата и время проведения операции
14	Date Expiration - год и месяц, после которого истекает срок действия карты.
18	Merchant Type - тип коммерческого продукта или услуги продавца, также известный как код категории продавца (MCC). Эти коды основаны на Руководстве по классификации кодов продавцов
19	Acquiring institution country code - код, который идентифицирует страну эквайрингового учреждения для продавца или банкомата
49	Currency Code - код, идентифицирующий валюту, в которой проводится операция

Эти и другие данные ISO сообщения сами по себе дают некоторую

информацию о проводимой операции. Но всё же в чистом виде делать по ним какие-то выводы довольно проблематично.

2.3.2 Данные банка-эмитента

В системе банка-эмитента хранится очень много информации о держателе карты и о самой карте. Однако не вся эта информация может быть полезна для задачи оценки рискованности. Более того, в зависимости от системы, установленной в процессинговом центре, эта информация может различаться. Однако некоторые данные можно выделить точно.

Во-первых, в ISO сообщении в чистом виде не содержится тип проводимой транзакции. Он определяется из нескольких его полей уже в процессинговой системе банка-эмитента. Основные типы проводимых операций, подверженных риску мошенничества:

- Retail - операция покупки товаров и услуг в торговой точке по карте. Может быть как операцией покупки в интернете (e-commerce), так и операцией покупки на POS-терминале или импринтере.
- Cash Withdrawal - снятие наличных в банкомате.
- Transfer - перевод средств от одного держателя карты к другому.

Также в нескольких полях ISO сообщения содержится информация об условиях проведения транзакции (22: Point of service entry mode, 25: Point of service condition code). Вся эта информация может объединена в некоторое единое описание условий проведения операции (condition). Правила формирования такого условия описаны в системе банка эмитента и присваиваются авторизационным и финансовым документам во время их обработки. Это условие может являться категориальным признаком в описании данных.

Глава 3. Обзор существующих решений

3.1 Rule-based системы

В данный момент в процессинговых системах существующих банков-эмитентов установлены специальные модули риск мониторинга. Однако эти системы не используют технологии машинного обучения и анализа данных. Работа таких модулей основана на своде правил, их также называют rule-based системы.

Для оценки рискованности финансовых транзакций формируются специальные правила, на основании которых делается вывод, можно ли отнести данную транзакцию к классу мошеннических или нет. Правила могут формироваться различным образом и в основном базируются на известных случаях мошенничества и экспертном мнении специалистов.

К примеру, в системе может присутствовать подобное правило: если первая транзакция была совершена в одной стране, а через несколько минут вторая транзакция совершается в другой, то вторая транзакция с высокой степенью уверенности может быть признана мошеннической. К пришедшей на авторизацию транзакции применяются все подобные правила, и на их основе делается вывод о степени рискованности данной транзакции.

Однако такой подход позволяет отлавливать лишь достаточно очевидные и уже известные случаи мошенничества. Для выявления более изобретательных мошенников необходимы более сложные системы, способные автоматически обучаться и выявлять сложные закономерности.

3.2 Одиночные модели

Для решения задачи оценки рискованности транзакции уже было сделано множество попыток с применением различных алгоритмов машинного обучения, среди которых Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LOR) и другие. Однако у каждого алгоритма есть свои плюсы и минусы. Рассмотрим сильные и слабые стороны некоторых из них [3].

Таблица 2: Сравнение алгоритмов

Модель	**Преимущества**	**Недостатки**
Random Forest	Прост в реализации и легко интерпретируется; Для решения задачи не обязательно использовать мощные вычислительные ресурсы; Подходит для работы в режиме реального времени.	Большой риск переобучения, в случае если набор данных для обучения не достаточно хорошо соответствует действительности; Для дополнительного обучения необходимо получить новые примеры мошеннических операций.
SVM	Способен восстанавливать нелинейные зависимости; Для решения задачи не обязательно использовать мощные вычислительные ресурсы; Подходит для работы в режиме реального времени.	Непросто обработать результат из-за преобразования входных данных.
Logistic Regression	Прост в реализации и исторически используется для обнаружения мошенничества.	Демонстрирует менее качественные результаты в сравнении с другими алгоритмами.

Naïve Bayes	Хорошо подходит для решения задачи бинарной классификации; Эффективно использует вычислительные ресурсы; Подходит для работы в режиме реального времени.	Необходимо хорошее понимание типичного и ненормального поведения для различных типов случаев мошенничества.
Neural Network	Подходит для решения задачи бинарной классификации и широко используется для обнаружения мошенничества.	Нуждается в мощных вычислительных ресурсах, не подходит для работы в режиме реального времени; Для дополнительного обучения необходимо получить новые примеры мошеннических операций.
Linear Regression	Демонстрирует оптимальные результаты, когда между независимыми и зависимыми параметрами наблюдается практически линейная зависимость.	Чувствителен к выбросам и ограничен только числовыми значениями.

3.3 Гибридные модели

Гибридные модели представляют собой комбинацию нескольких отдельных моделей. Эффективное сочетание нескольких одиночных моделей в одну гибридную позволяет повысить качество обучения. Высокое качество, демонстрируемое данными моделями, позволяет сделать вывод, что

их можно использовать для решения задачи оценки рискованности финансовых транзакций. Подробнее ознакомимся с принципами их работы и посмотрим какие результаты они показывают при решении данной задачи [3].

3.3.1 Adaptive Boosting

Adaptive Boosting или AdaBoost используется в сочетании с различными типами алгоритмов для повышения их производительности. AdaBoost — один из самых популярных алгоритмов для построения сильного классификатора с линейной комбинацией классификаторов членов, которые выбираются так, чтобы минимизировать ошибки на каждом шаге итерации в процессе обучения. AdaBoost предоставляет очень простой и полезный метод для генерации ансамблевых классификаторов. Производительность ансамбля зависит от разнообразия среди классификаторов участников, а также от производительности каждого из них [4]. Рассмотрим принцип работы данного алгоритма [3].

Выходные данные каждого отдельного алгоритма объединяются с использованием взвешенной суммы, которая представляет объединенный выходной сигнал улучшенного гибридного классификатора, т.е.

$$F_T(x) = \sum_{t=1}^{T} f_t(x)$$

где каждый f_t — это классификатор, который возвращает предсказанный класс относительно входных данных x. Каждый отдельный классификатор дает выходной прогноз $h(x_i)$ для данной обучающей выборки. На каждой итерации t выбирается классификатор, и ему присваивается коэффициент t, так что сумма итоговой ошибки обучения E_t полученного улучшенного классификатора шага t минимизируется:

$$E_t = \sum_i E[F_{t-1}(x_i) + \alpha_t h(x_i)]$$

где $F_{t-1}(x)$ — улучшенный классификатор, построенный на предыдущем этапе, $E(F)$ — функция ошибки, а $f_t(x) = \alpha_t h(x)$ — отдельный классификатор, учитываемый для окончательного классификатора.

AdaBoost подстраивает отдельные классификаторы в пользу неверно классифицированных образцов данных. Однако он чувствителен к шуму и выбросам. Пока производительность классификатора не случайна, AdaBoost может улучшить отдельные результаты из разных алгоритмов.

AdaBoost использовался с разнообразными одиночными моделями. По результатам тестирования данного алгоритма на конкретном наборе данных видно, что точность и частота обнаружения не мошеннических транзакций аналогичны показателям без AdaBoost. Тем не менее, для некоторых моделей уровень обнаружения мошенничества вырос на несколько процентов. Некоторые модели имеют незначительное снижение уровня обнаружения мошенничества до 1%. С более подробными результатами тестирования можно ознакомиться в статье [3].

3.3.2 Majority Voting

Majority Voting часто используется в классификации данных, которая представляет собой комбинированную модель, по крайней мере, из двух алгоритмов. Каждый отдельный алгоритм делает свой собственный прогноз для каждого тестового образца. Окончательный результат определяется большинством голосов отдельных алгоритмов. Majority Voting привлекает большое внимание в литературе при рассмотрении схемы комбинации классификаторов из-за его простоты и хорошей производительности при реальной обработке данных. Многочисленные исследования показали, что базовые классификаторы, которые объединяются для построения единой улучшенной модели, должны быть достаточно разнообразными. Бессмысленно объединять несколько идентичных классификаторов. Иногда изменение разнообразия среди классификаторов может быть ключом к успеху [5].

Алгоритм Majority Voting работает следующим образом [3]. Рассмотрим K целевых классов (или меток), где $C_i, \forall i \in \Lambda = 1, 2, ..., K$ представляет i-й целевой класс, предсказанный классификатором. При заданном входном значении x каждый классификатор обеспечивает прогноз относительно целевого класса, в результате чего получается всего K прогнозов,

то есть $P_1, ..., P_K$. Голосование большинством направлено на создание комбинированного прогноза для входных данных $x, P(x) = j, j \in \Lambda$ из всех K прогнозов, то есть $p_k(x) = j_k, k = 1, ..., K$. Для представления голосов отдельных алгоритмов может использоваться бинарная функция вида

$$V_k(x \in C_i) = \begin{cases} 1, & \text{если } p_k(x) = i, i \in \Lambda \\ 0, & \text{иначе} \end{cases}$$

Затем необходимо суммировать голоса всех K классификаторов для каждого C_i, и метка, получившая наибольшее количество голосов, является окончательным (комбинированным) прогнозируемым классом. К рассмотренным выше одиночным алгоритмам был применен метод Majority Voting и протестирован на конкретном наборе тестовых данных. Данный подход показал неплохие результаты, значительно улучшив качество базовых алгоритмов. Среди всех комбинаций можно выделить сочетание Neural Network и Naïve Bayes, которое в ходе тестирования показало лучший средний результат среди рассмотренных комбинаций. Подробнее о ходе тестирования можно ознакомиться в статье [3].

Глава 4. Работа с данными

В качестве данных, необходимых для решения расматриваемой задачи, использовался открытый датасет, полученный с помощью симулятора PaySim, который генерировал данные на основе примера реальных данных африканского банка за месяц [6].

4.1 Описание данных

Выбранные данные представляют собой набор финансовых транзакций, про каждую из которых известны момент времени (step), тип транзакции (type), сумма трнзакции (amount), имя инициатора транзакции (nameOrig), его баланс до совершения транзакции (oldBalaceOrig) и после (newBalanceOrig), имя получателя транзакции (nameDest), его баланс до совершения транзакции (oldBalanceDest) и после (newBalanceDest). При

этом каждая транзакция помечена нулём или единицей по двум признакам:

- isFraud - была ли эта транзакция в действительности мошеннической или нет;
- isFlaggedFraud - была ли эта транзакция помечена банковской системой как потенциально мошенническая или нет. В данном наборе данных такими транзакциями считаются попытки передать сумму, большую, чем 200.000 у.е.

4.2 Разведочный анализ данных

Перед тем, как приступить к работе с данными, их необходимо проанализировать, выявить основные свойства и закономерности, которые помогут при дальнейшей работе с ними [7]. Представим основные выводы, полученные в результате такого анализа:

1. В исходном наборе данных были представлены транзакции пяти видов, однако, к классу мошеннических относятся транзакции всего двух видов. Это транзакции перевода и снятия наличных. При этом количество мошеннических транзакций этих типов примерно одинаковое.
2. В исходном наборе данных всего 16 транзакций отмечены признаком isFlaggedFraud. И несмотря на то, что все транзакции, отмеченные этим признакам, в действительности являются мошенническими, значимых закономерностей в данных, отмеченных этим признаком найти не удалось. Поэтому, а также ввиду его редкой встречаемости, данный признак можно считать неинформативным.
3. В исходном наборе данных присутствуют транзакции, в которых на счёте получателя был нулевой баланс до и после транзакции, при том что сумма транзакции не была нулевой. Такие транзакции составляют достаточно большую долю среди мошеннических и совсем малую среди безопасных.
4. А вот транзакции, в которых на счёте отправителя был нулевой баланс до и после транзакции, при том, что сумма транзакции не была

нулевой, составляют довольно большую долю среди безопасных и совсем малую среди мошеннических.

4.3 Обработка данных

Оставим из всех транзакций только операции перевода и снятия наличных. В транзакциях остальных типов не встречаются мошеннические, поэтому они не представляют для нас большого интереса.

Нулевой остаток на счёте получателя транзакции является достаточно сильным показателем мошенничества. Чтобы сделать этот признак более полезным, заменим нули в таком случае на -1.

Также в связи с тем, что нулевые блансы помогают в определении мошеннических транзакций, введем также два новых признака, которые повысят качество классификации:

$$errorBalanceOrig = newBalanceOrig + amount - oldBalanceOrig$$

$$errorBalanceDest = oldBalanceDest + amount - newBalanceDest$$

На графиках, представленных на рис. 2, можно увидеть разделение транзакций на мошеннические и безопасные при попарном использовании двух новых введённых признаков и признака step. Как можно видеть, транзакции хорошо отделяются друг от друга при использовании признака step с каждым из новых признаков.

Для корректной работы алгоритмов нормализуем и стандартизируем данные. В процессе нормализации все значения признаков приводятся к некоторому заданному диапозону, например, [0...1]. Данная процедура необходима так как значения признаков могут варьироваться в очень большом диапозоне, что негативно сказывается на работе некоторых моделей машинного обучения. С помощью стандартизации устраняется возможное влияние отклонений по какому-либо признаку. Значения стандартизированной шкалы определяются по следующей формуле:

$$z_i = \frac{x_i - \bar{X}}{\sigma_x}$$

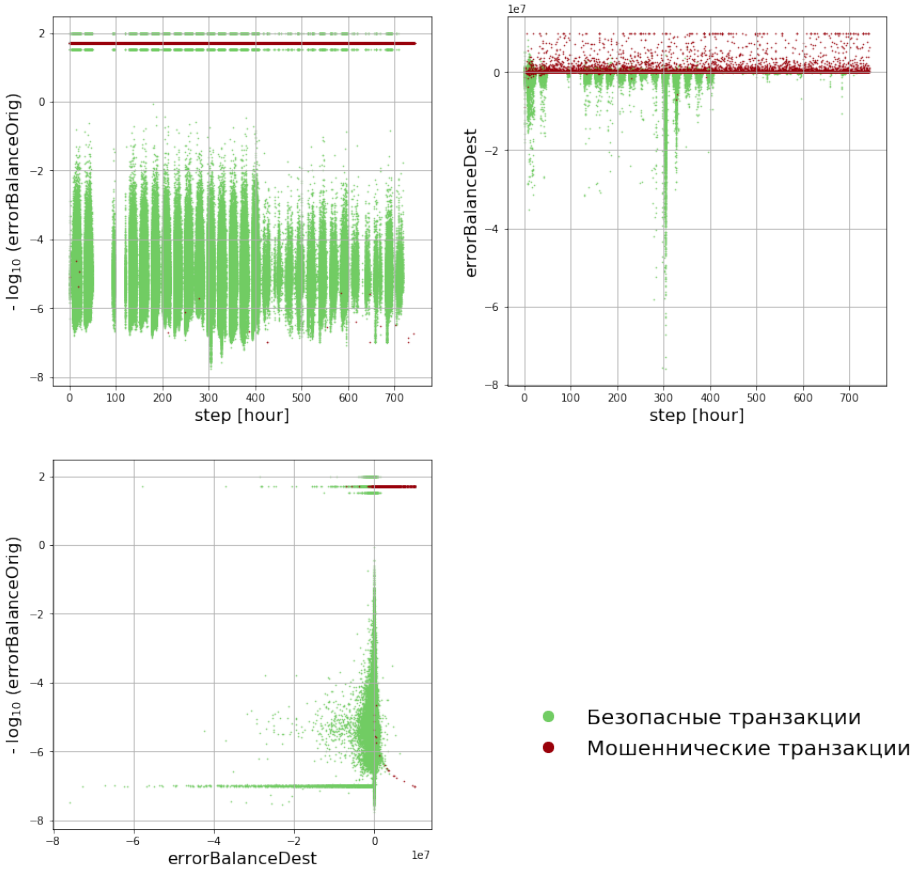

Рис. 2: Разделение транзакций по новым признакам и признаку step

где x_i — исходное значение признака, $\bar{X}$ и σ_x — среднее значение и стандартное отклонение признака, оцененные по набору данных. В результате вне зависимости от начального распределения всех исходных значений набора данных они приводятся к набору значений с нулевым средним и единичным стандартным отклонением.

Также некоторые из рассматриваемых для решения данной задачи алгоритмов чувствительны к дисбалансу классов, присутствующем в данном наборе данных. Так как мы располагаем совсем небольшим количеством примеров мошеннических транзакций для решения данной проблемы нам не подойдут алгоритмы удаления примеров мажоритарного класса (Random Undersampling). Поэтому применим простой алгоритм случайного дублирования примера миноритарного класса (Random Oversampling) и добьёмся равенства количества мошеннических и безопасных транзакций.

Глава 5. Выбор модели

5.1 Рассматриваемые алгоритмы

Рассмотрим данную проблему, как задачу бинарной классификации, что является классической задачей машинного обучения. Рассмотрим несколько простых моделей, применяемых для решения данной проблемы:

- **Логистическая регрессия** определяет связь между бинарными результатирующими признаками и независимыми переменными, используя вероятность в качестве прогнозируемого значения зависимой переменной. Каждой проверенной транзакции присваивается вероятность, которая затем используется для классификации транзакции как мошеннической или не мошеннической. Если вероятность больше порога – это мошенничество, иначе – не мошенничество [11].
- **Решающее дерево** – это набор узлов, который определяет значение результатирующей переменной по признакам, связанным с определенными классами. Каждый узел представляет собой правило разделения для признака. Создание новых узлов происходит до тех пор, пока не будет достигнут критерий остановки. Метка класса определяется на основе большинства образцов, принадлежащих конкретному листу [3].
- **Метод опорных векторов (SVM)** реализует следующую идею: он отображает входные векторы в многомерное пространство признаков и создает оптимальную разделяющую гиперплоскость, которая максимизирует зазор, расстояние между гиперплоскостью и ближайшими точками данных каждого класса в пространстве. [12].

Рассмотрим также следующие композиции простых моделей:

- **Случайный лес** строит множество решающих деревьев, количество которых устанавливается в качестве параметров модели. Чтобы определить окончательный результат классификации используется голосование всех построенных деревьев [3].

- **Градиентный бустинг** над решающими дереьями улучшает прогнозирующую способность благодаря применению дерева решений к остаткам модели, а не к результирующей переменной. Каждый раз новое дерево добавляется в подобранную модель для обновления остатков. В данном алгоритме деревья выращиваются последовательно, причем каждое последующее исправляет ошибки, содержащиеся в предыдущих деревьях. Множитель скорости обучения применяется на каждом этапе, чтобы помочь предотвратить переобучение модели [11].

5.2 Выбор метрики

Можно рассмотреть два варианта метрик для сравнения качества алгоритмов бинарной классификации:

- AUC-ROC — площадь под ROC-кривой;
- AUC-PRC — площадь под PR-кривой [8].

Площадь под ROC-кривой не зависит от баланса классов и гораздо лучше интерпретируется. А площадь под кривой точности и полноты гораздо выразительнее в случае дисбаланса классов [9]. Так как в данном случае мы имеем дело с сильно несбалансированными данными, для сравнения качества алгоритмов воспользуемся метрикой AUC-PRC.

5.3 Применение алгоритмов

Для тестирования выбранных алгоритмов исходная выборка была разделена на обучающую и тестовую в отношении $8 : 2$ без перемешивания, чтобы сохранить последовательность признака step. Все перечисленные алгоритмы были применены к описанным данным, в результате получены следующее значение метрики AUC-PRC на тестовых данных (см. таблицу 3).

Таблица 3: Таблица результатов

Название метода	**AUC-PRC**
Логистическая регрессия	0.989399
Решающее дерево	0.923422
Метод опорных векторов	0.916329
Случайный лес	0.998548
Градиентный бустинг	0.999998

Для нагядности посмотрим на кривые точности-полноты данных алгоритмов на рис. 3. Кривая точности-полноты показывает компромисс между точностью и полнотой для разных пороговых значений. Высокая площадь под кривой показывает как высокую степень полноты, так и высокую точность, где высокая точность отражает низкий показатель false positive rate (FP), а высокая полнота отражает низкий показатель false negative rate (FN). Высокие оценки для обоих показателей говорят о том, что классификатор возвращает точные результаты (высокая точность), а также возвращает большинство всех положительных результатов (высокая полнота)[13].

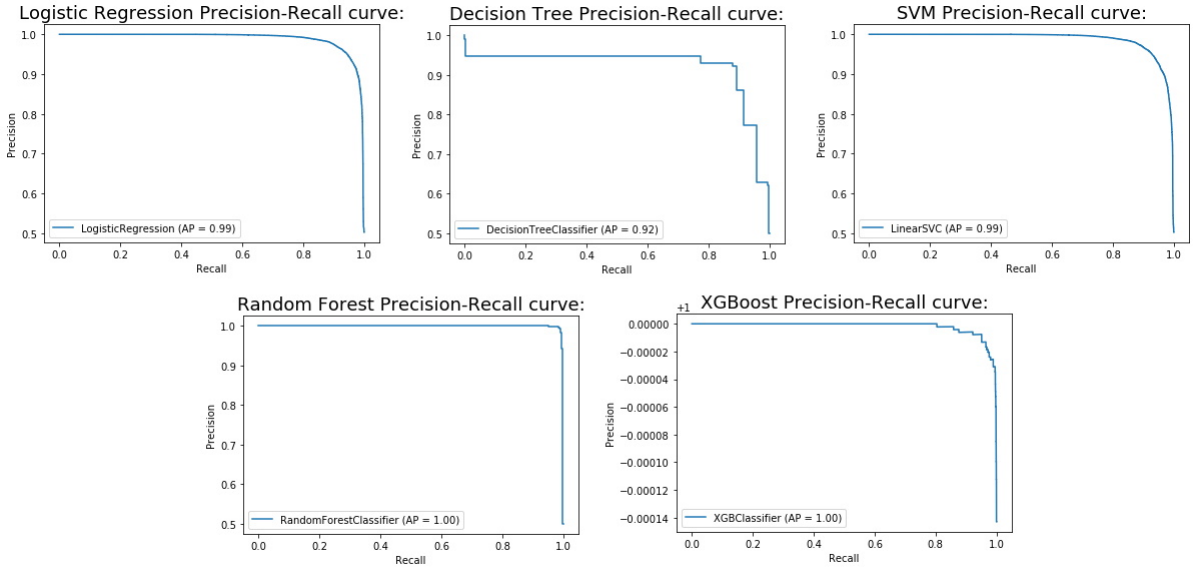

Рис. 3: Кривые точности-полноты рассмотренных алгоритмов

На рассмотренных данных композиции простых алгоритмов показали

лучший результат, чем одиночные. Полученная хорошая точность композиций алгоритмов, говорит о том, что они пригодны для решения данной задачи. При этом лучше всего с задачей справился градиентный бустинг над решающими деревьями. Это связано с тем, что данный метод по своей природе хорошо работает с несбалансированными данными. Установлено, что градиентный бустинг очень устойчив к несбалансированному набору данных, в частности из-за того, как он обрабатывает ошибочно классифицированные точки. Бустинг включает в себя объединение слабых моделей для создания сильной модели, способной делать точные прогнозы [10].

Рассмотрим подробнее результаты работы градиентного бустинга над решающими деревьями, как алгоритма, показавшего наилучший результат. На рис. 4 показана значимость признаков в классификации. На данной диаграмме хорошо видно, что введённый нами признак errorBalanceOrig является наиболее информативным для данной модели. Признаки упорядочены в зависимости от количества выборок, затронутых разделением по этим признакам.

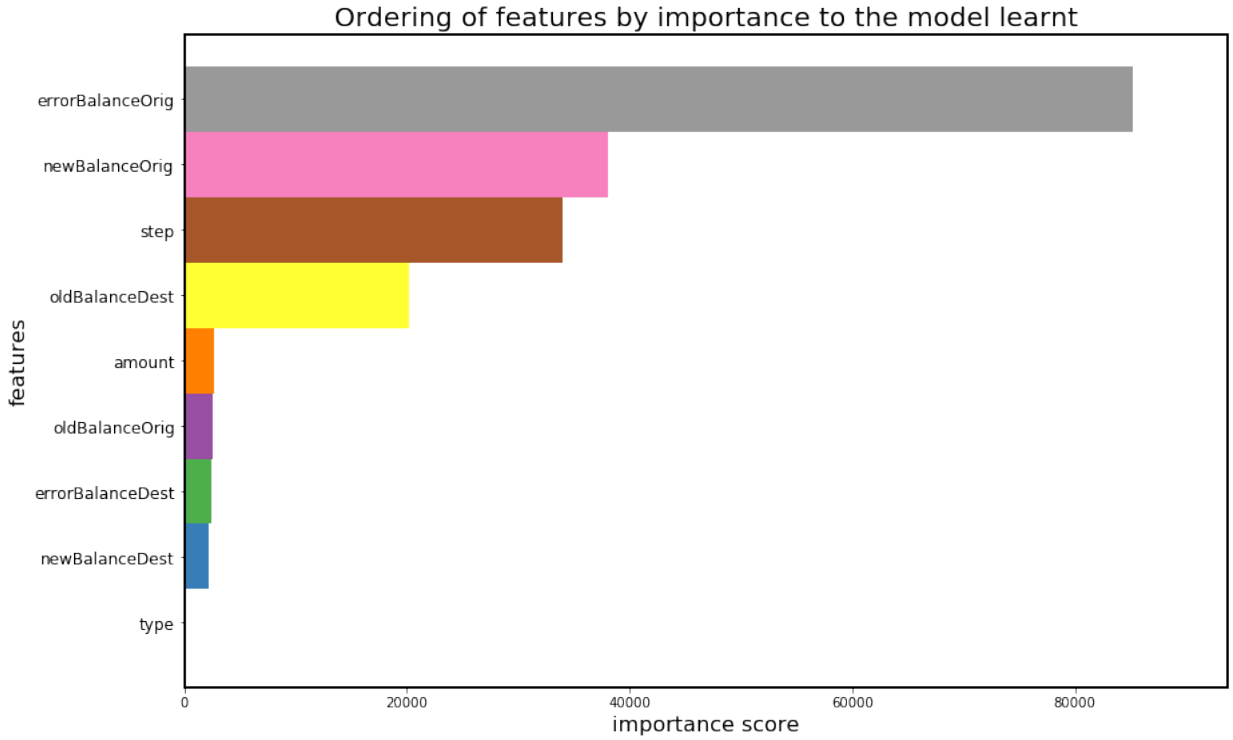

Рис. 4: Диаграмма значимости признаков

Большим плюсом дерева решений является его хорошая интерпрети-

руемость. Отобразим работу градиентного бустинга в виде объединённого дерева на рис. 5. Корневым узлом в данном дереве является признак errorBalanceOrig, что и следовало ожидать ввиду его высокой значимости.

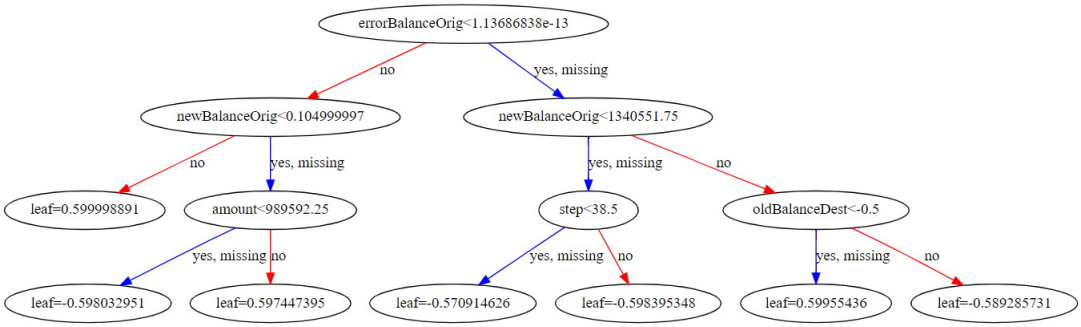

Рис. 5: Визуализации работы градиентного бустинга

Глава 6. Разработка программного модуля

Реализация программного модуля предполагается как отдельный сервис для выявления мошенничества. К данному сервису подключаются различные банки, получают свой персональный токен и доступ к публичному API, на которое могут отправлять запросы с описанием очередной транзакции для проверки её на мошенничество. В качестве модели, классифицирующей транзакции на мошеннические или безопасные, было решено использовать градиентный бустинг над решающими деревьями, так как он показал наилучший результат на используемых данных.

6.1 Архитектура полученного решения

Приложение было построенно по принципу классической клиент-серверной архитектуры. Однако в данном случае клиентским является само приложение банка, которое обращается к серверу с помощью HTTP запросов. Более подробную архитектуру можно увидеть на рис. 6.

Таким образом, было необходимо организовать систему хранения данных, необходимых для работы приложения, реализовать интерфейс, с по-

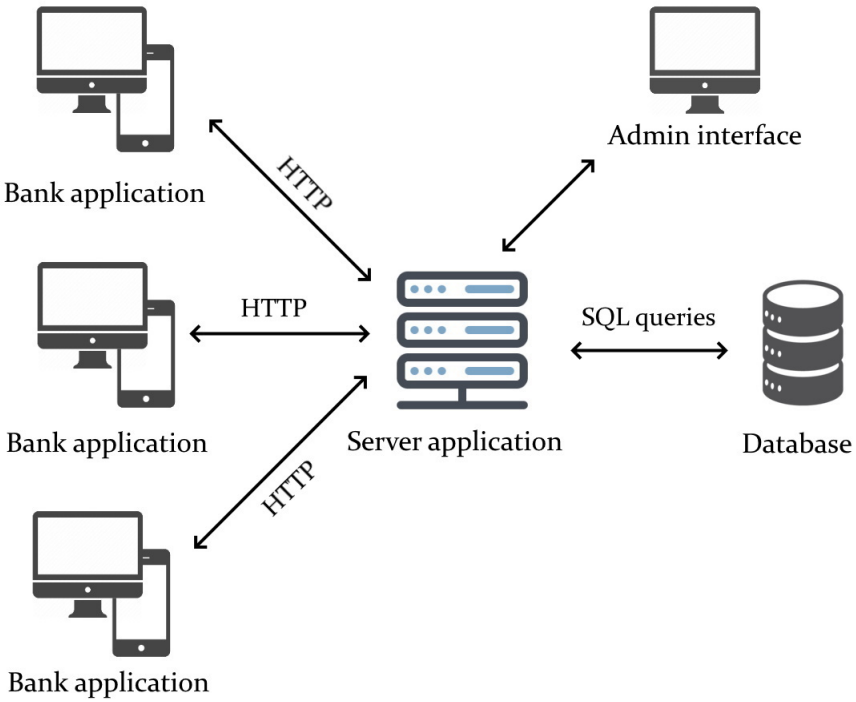

Рис. 6: Архитектура полученного решения

мощью которого банковские приложения смогут работать с приложением, и интерфейс для администрирования платформы.

6.2 Выбранные технологии

Для реализации предложенной архитектуры было решено использовать следующие технологии:

- **Серверное приложение**. Для реализации серверного приложения использовался фреймворк Django. Данная технология была выбрана по той причине, что Django содержит огромное количество функциональности для решения большинства задач веб-разработки, благодаря чему, а также особенностям языка python, на котором написан данный фреймворк, можно достаточно быстро создать качественное и легко поддерживаемое приложение [14]. Также в проекте использовалась библиотека для построения API Django REST Framework, в частности доступные из коробки политики аутентификации и разрешений.

- **База данных**. Для хранения данных было решено использовать реляционную базу данных, а именно PostgreSQL. Данная технология имеет множество возможностей, поддерживает сложные структуры и широкий спектр встроенных и определяемых пользователем типов данных, обеспечивает расширенную ёмкость данных и заслуживает доверие бережным отношением к целостности данных [15].

6.3 Система хранения данных

Для работы сервиса необходимо хранить данные о клиентах, подключенных к системе, в том числе их токены, с помощью которых будет происходить авторизация. Кроме того нужно хранить данные о совершённых транзакциях для того, чтобы иметь возможность дообучать и переобучать модель на свежих данных, чтобы повысить качество её работы. Схематично представим структуру полученной базы данных на рис. 7. Опишем подробнее каждую из таблиц.

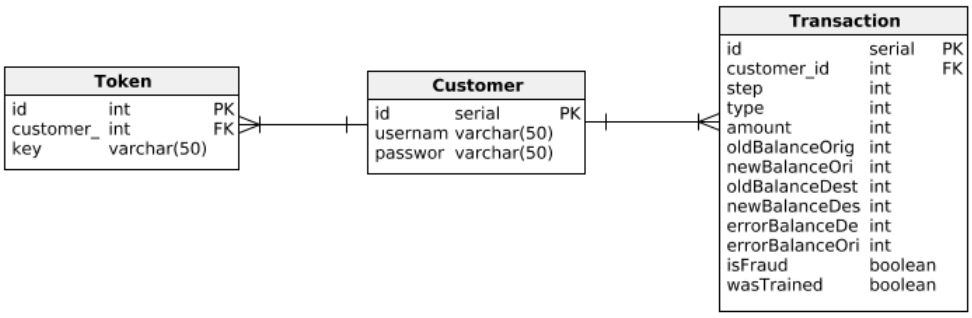

Рис. 7: Схема структуры базы данных

Token. Таблица используется для хранения персональных токенов банков, использующихся ими как средство аутентификации. При регистрации нового клиента в данной таблице создаётся новая запись, содержащая его персональный токен. По запросу токен конкретного пользователя может быть получен из этой таблицы, однако для этого нужны его данные авторизации (имя и пароль).

Customer. Таблица с информацией о банке-клиенте. В данный момент в таблице содержатся только имя и пароль клиента, необходимые для его авторизации и получения персонального токена. Однако, разумеется,

данная таблица может быть сильно расширена и дополнена различной информацией о банке: название, расположение главного офиса, контактные данные представителя, платёжная информация и т.д. Также могут быть добавлены другие дополнительные таблицы для более гибкого и удобного хранения информации о клиентах (к примеру, таблица с адресами, телефонами и т.п.)

Transaction. Таблица содержит информацию о транзакциях, прошедших через систему. Каждая транзакция ссылается на запись в таблице клиентов для идентификации отправителя данной транзакции. Поля step, type, amount, oldBalanceOrig, oldBalanceDest, newBalanceOrig, newBalanceDest содержат основную информацию, полученную от банка в процессе работы. Поля errorBalanceDest и errorBalanceOrig — это новые признаки, введенные для повышения качества обучения и описанные в разделе 4.3. Поле isFraud содержит информацию о том, является ли транзакция мошеннической. При поступлении транзакции данное поле содержит результат классификации данной транзакции алгоритмом сервиса, однако в дальнейшем его значение может быть изменено при исправлении предсказания со стороны банка (см. в разделе 6.4.1). Поле wasTrained показывает участвовала ли данная транзакция в процессе обучения модели.

Также заметим, что в связи с нелинейностью классифицирующей модели хранить её параметры в базе данных не представляется возможным, в связи с чем, параметры, используемые для классификации будут храниться в одном отдельном файле-описании модели.

6.4 Программный интерфейс приложения

Общение клиентов с сервером происходит с помощью API. При этом доступные запросы можно разделить на две категории: клиентские и администраторские. На каждый из запросов в ответ от сервера обязательно приходит поле response_code. По значению в этом поле можно понять, прошла ли требуемая операция успешно или возникли какие-либо проблемы. Возможные значения данного поля должны быть специфицированны и переданы клиентам, чтобы они могли корректно реагировать на те или

иные ошибки, возникшие в процессе обработки их запросов. Далее опишем подробнее каждую из категорий доступныхх запросов.

6.4.1 Клиентские запросы

К данной категории относятся различные запросы, отправляемые банковскими приложениями, необходимые для их работы. Программный модуль поддерживает следующие клиентские запросы:

- **Получение токена**. Данный запрос, используя персональный логин и пароль банка-клиента, позволяет получить его токен, необходимый для дальнейших запросов. Тело запроса состоит из двух полей:
  - username: имя пользователя,
  - password: пароль.

  Заголовок не должен содержать особых полей. В ответе приходят два поля:
  - response_code: код ответа,
  - token: персональный токен данного клиента.

- **Проверка транзакции на мошенничество**. В данном запросе банк должен передать данные о транзакции, которую необходимо проверить на мошенничество. Тело запроса состоит из следующих полей:
  - id: некоторый идентификатор транзакции (на усмотрение банка),
  - type: тип транзакции,
  - amount: сумма трнзакции,
  - oldBalaceOrig: баланс источника операции до совершения транзакции,

- newBalanceOrig: баланс источника операции после совершения транзакции,
- oldBalanceDest: баланс получателя до совершения транзакции,
- newBalanceDest: баланс получателя после совершения транзакции.

Заголовок транзакции должен содержать поле Authorization с токеном клиента. В ответе приходят два поля:

- response_code: код ответа,
- result: булевая переменная, показывающая, является ли транзакция мошеннической.

- **Исправление предсказания**. Данный запрос направлен на то, чтобы сообщить сервису об ошибке в его предсказании. Тело запроса состоит из двух полей:
  - id: некоторый идентификатор транзакции (на усмотрение банка),
  - result: булевая переменная, показывающая была ли транзакция в действительности мошеннической или нет.

  В ответе приходит единственное поле: response_code - код ответа.

6.4.2 Администраторские запросы

В данную категорию входят запросы, отправляемые администратором для управления процессом работы сервиса. Все запросы в данной категории должны содержать поле Authorization с токеном администратора, не будем указывать это в дальнейшем. К таким запросам относятся следующие:

- **Регистрация нового клиента**. Данный запрос создаёт в системе запись о новом банке-клиенте и создаёт для него токен. Тело запроса состоит из двух полей:

  - username: имя пользователя,
  - password: пароль.

  В ответе приходят два поля:

  - response_code: код ответа,
  - token: персональный токен данного клиента.

- **Анализ качества работы модели**. Администратор должен иметь возможность оценить качество работы модели, посмотрев на конкретные метрики. В данный момент оцениваемыми метриками являются значения true positive (TP), false positive (FP), true negative (TN) и false negative (FN), а также значение метрики, используемой при оценки качества алгоритмов, описанной в разделе 5.2 — AUC-PRC.

  В ответе на данный запрос приходят следующие поля:

  - response_code: код ответа,
  - tp_rate: значение показателя TP,
  - fp_rate: значение показателя FP,
  - tn_rate: значение показателя TN,
  - fn_rate: значение показателя FN,
  - auc_prc: значение показателя AUC-PRC,

- **Дообучение модели на новых транзакциях**. В результате выполнения данного запроса к процессу обучения модели будут подключены данные о новых транзакциях, не участвующих в обучении до этого. Параметры новой модели будут перезаписаны в файл и будут использоваться при классификации транзакций в дальнейшем. Данный запрос содержит всего одно поле: skipDays - количество последних дней, за которые не нужно учитывать транзакции в процессе дообучения.

  В ответе приходит единственное поле: response_code - код ответа.

- **Полное переобучение модели на новом наборе данных**. Данный запрос необходим для полного переобучения модели на новых данных, все остальные при этом перестают участвовать в процессе обучения. Запрос содержит два поля:
  - skipDays: количество последних дней, за которые не нужно учитывать транзакции в процессе переобучения,
  - maxDays: максимальное количество дней, за которое необходимо учитывать транзакции в процессе переобучения.

  В ответе приходит единственное поле: response_code - код ответа.

На основе данных администраторских запросов может быть реализовано клиентское приложение, поддерживающее все вышеперечисленные функции и имеющее удобный интерфейс для администраторов разрабатываемого сервиса.

Глава 7. Полученные результаты

В результате данной работы был реализован работающий прототип веб-сервиса для оценки рискованности финансовых транзакций. В данном прототипе реализованна вся функциональность, описанная выше, однако при разработке не были учтены требования стандарта безопасности PCI DSS, которые обязательно необходимо соблюдать при работе с платёжной информацией. Это, а также другие функциональные изменения, повышающие качество работы сервиса, будут запланированы на следующие фазы разработки. Опишем поэтапно процесс работы данного сервиса.

7.1 Этап 1. Регистрация нового клиента

Банку или платёжной организации необходимо подключиться к данному сервису для начала работы с ним. Не будем рассматривать юридические аспекты данного процесса, однако после некоторой договоренности с банком, администратор заводит запись о нем в базе данных, клиенту

при этом выдается его персональный токен, а также имя пользователя и пароль, по которым всегда можно получить токен в случае необходимости.

Выданный клиенту веб-токен будет служить средством аутентификации. Данный токен будет добавляться в заголовки всех запросов, отправляемых клиентами к веб-сервису.

7.2 Этап 2. Процесс использования платформы

После того, как банк был подключен к сервису по распознаванию мошеннических транзакций, ему становятся доступны клиентские запросы, описанные выше в разделе 6.4.1. Процесс работы с сервисом выглядит следующим образом:

1. **Получение токена**. В случае необходимости клиент получает токен, который понадобится ему для использования других функций.

2. **Проверка транзакции на мошенничество**. При совершении держателем карты банка-клиента какой-либо операции, данная операция проходит авторизацию в данном банке. В этот момент ему необходимо оценить степень рискованности данной операции, и в случае подозрения на мошенничество принять необходимые меры. Именно в этот момент банку необходимо воспользоваться сервисом для распознавания мошенничества.

   В процессе обработки очередной транзакции перед тем, как вернуть источнику операции сообщение об успехе или неудаче, банк отправляет запрос к веб-сервису с информацией о текущей операции. Веб-сервис, используя преобученную модель выдаёт предсказание относительно текущей операции. В итоге, банк-клиент получает от веб-сервиса ответ с информацией о том, является данная операция мошеннической или нет.

3. **Исправление предсказания**. После получения информации от веб-сервиса банк предпринимает необходимые меры в случае, если транзакция признаётся мошеннической. Однако, в работе сервиса возмож-

ны ошибки, которые необходимо корректировать. В такой ситуации возможны два случая:

- Веб-сервис распознал транзакцию как мошенническую, после чего банк-клиент предпринял необходимые меры (блокировка карты, отправка сообщения с предупреждением, телефонный звонок и т.д.) Однако после этого выяснилось, что данная транзакция в действительности не была мошеннической, к примеру, при поступлении обращения в банк от клиента.
- Веб-сервис распознал транзакцию как безопасную, однако в последствии выяснилось, что данная операция оказалась мошеннической, к примеру, при поступлении обращения в банк от клиента.

При выявлении ошибок банку-клиенту доступен функционал для внесения корректировок в работу сервиса. Это позволит улучшить качество распознавания при дальнейшем использовании данной платформы.

7.3 Этап 3. Учучшение результатов предсказания

Все транзакции, проходящие через сервис в процессе его работы, могут являться новыми свежими данными для обучения и улучшения качества предсказания. Наличие клиентской функциональности для корректировки предсказаний веб-сервиса позволяет считать имеющиеся в системе данные корректными и пригодными для обучения. Более того, различные внешние факторы могут изменять природу данных и необходимо периодически переобучать модель, чтобы старые и уже неактуальные данные не ухудшали качества распознования. Администратор сервиса всегда может оценить качество работы модели в данный момент с помощью запроса, возвращающего ему основные метрики качества работы модели. Для дообучения и переобучения модели веб-сервис поддерживает администраторскую функциональность, описанную выше в разделе 6.4.2.

Выводы

В данной работе был рассмотрен процесс оплаты по картам, выделены его основные участники и данные, которыми они обмениваются. В этом процессе было определено место системы оценки рискованности финансовых транзакций, а также выделены участники, которым необходимо использование такой системы.

Был проведён анализ алгортимов, используемых для распознования мошенничества при анализе финансовых транзакций. Были выделены плюсы и минусы рассмотренных алгоритмов, а также описана возможность использования композиций простых алгоритмов для улучшения качества модели.

Подробно были описанны данные, используемые для обучения модели. Выбранные данные были предобработаны, после чего на них были опробованны различные алгоритмы машинного обучения. Лучший из них использовался в дальнейшем при создании веб-сервиса для оценки рискованности финансовых транзакций.

В результате был получен веб-сервис, к которому могут подключаться различные банки и в режиме реального времени проверять финансовые операции своих клиентов на мошенничество. Качество работы данного веб-сервиса постоянно улучшается благодаря функциональности для внесения корректировок в результаты классификации данного сервиса, а также возможности дообучать и переобучать модель машинного обучения, которая используется для распознования.

Заключение

В ходе данной работы был разработан сервис, способный распознавать мошенничество в транзакциях банков с использованием платежных карт с достаточно высокой точностью. Полученный результат показывает, что методы машинного обучения могут быть использованы для улучшения оценок рискованности финансовых транзакций.

В дальнейшем планируется собрать реальные данные финансовых операций и подобрать оптимальную модель для выявления мошенничества на этих данных. При этом планируется рассмотреть более широкий спектр алгоритмов, которые можно использовать для задачи бинарной классификации. Необходимо более подробно рассмотреть вопрос безопасности, проанализировать требования стандарта безопасности PCI DSS и внести корректировки для их соблюдения. Также нужно реализовать клиентское приложение для администрирования платформы, веб-сайт для более удобной регистрации банков-клиентов и для отображения информации о качестве и процессе работы веб-сервиса.

Список литературы

[1] The Nilson Report 10-17-2016 // The Nilson Report URL:https://nilsonreport.com/upload/content_promo/The_Nilson_Report_10-17-2016.pdf (дата обращения: 17.12.19).

[2] ISO 8583-1:2003 // ISO URL:https://www.iso.org/standard/31628.html (дата обращения: 19.12.2019).

[3] Kuldeep Randhawa, Chu Kiong Loo, Manjeevan Seera, Chee Peng Lim, Asoke K. Nandi Credit Card Fraud Detection Using AdaBoost and Majority Voting // IEEE Access. 2018. №6.

[4] Tae-Ki An, Moon-Hyun Kim A New Diverse AdaBoost Classifier // International Conference on Artificial Intelligence and Computational Intelligence. 2010.

[5] Peng Hong, Lin Chengde, Luo Linkai, Zhou Qifeng Accuracy of Classifier Combining Based on Majority Voting // IEEE International Conference on Control and Automation. 2007.

[6] E. A. Lopez-Rojas , A. Elmir, S. Axelsson PaySim: A financial mobile money simulator for fraud detection // European Modeling and Simulation Symposium-EMSS. 2016. №28.

[7] П. Брюс, Э. Брюс. Практическая статистика для специалистов Data Science. СПб.: БХВ-Петербург, 2018.

[8] Davis J., Goadrich M. The Relationship Between Precision-Recall and ROC Curves. // Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA. 2006.

[9] Семинары по выбору моделей // MachineLearning.ru URL: http://www.machinelearning.ru/wiki/images/1/1c/Sem06_metrics.pdf (дата обращения: 13.03.20).

[10] Yash Singhal, Ayushi Jain, Shrey Batra, Yash Varshney, Megha Rathi Review of Bagging and Boosting Classification Performance on Unbalanced Binary Classification // IEEE 8th International Advance Computing Conference (IACC). 2018.

[11] Gabriel Rushin, Cody Stancil, Muyang Sun, Stephen Adams, Peter Beling Horse race analysis in credit card fraud—deep learning, logistic regression, and Gradient Boosted Tree // Systems and Information Engineering Design Symposium (SIEDS). 2017.

[12] Qingshan Deng Application of support vector machine in the detection of fraudulent financial statements // 4th International Conference on Computer Science & Education. 2009.

[13] Precision-Recall // scikit-learn URL: `https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html` (дата обращения: 25.05.2020).

[14] Форсье Дж., Биссекс П., Чан У. Django. Разработка веб-приложений на Python. СПб.: Символ-Плюс, 2009.

[15] Моргунов Е. П. PostgreSQL. Основы языка SQL. СПб.: БХВ-Петербург, 2018.