Курсовая работа:

тема: Разработка системы обнаружения мошенничества с кредитными картами.

Содержание:

Глава 0. Знакомство.

Глава1. Введение.

2. Инструменты для решения задачи.

3. Архитектура приложения.

4. Доступ к данным.

5. Очистка данных.

6. EDA.

7. Подготовка данных к обучению моделей.

8. Балансировка выборки.

9. Моделирование и оценка моделей.

10. Правила пользования.

11. Перспектива.

Список литературы.

Глава 0. Знакомство.

Добрый день, уважаемые преподаватели и товарищи по цеху. Я Пастушенко Антон и тема моей работы “Разработка системы обнаружения мошенничества с кредитными картами”. На протяжении 7 минут я расскажу вам обо всем простым языком.

Глава1. Введение.

Реальность.

В постоянно развивающемся современном мире люди пользуются банковскими картами. Все больше в нашу жизнь приходит безначичный расчет за услуги. Мы расплачиваемся картами в магазинах, кафе, заведениях, транспорте и так далее. Но большинство транзакций происходит через интернет. Абсолютное большинство денежных сделок происходят безналично. Просто перечисляют деньги на счет банка с другого счета. А реальные деньги просто лежат в хранилищах и выполняют другие функции. В данной работе будем дальше рассматривать только безналичные транзакции.

Наиболее распространенные сферы использования банковских карт/безналичного расчета:

1. Розничная торговля. Более 70% всех операций с картами относятся к этой сфере.

2. Онлайн-покупки. С каждым годом все больше людей делают покупки через интернет, поэтому онлайн-торговля становится все более популярной. Более 20% всех операций с банковскими картами относятся к онлайн-покупкам.

3. Путешествия: Оплата гостиниц, билетов на самолет и других туристических услуг - это еще одна распространенная сфера использования банковских карт. По статистике, около 7% всех операций с банковскими картами относятся к путешествиям.

4. Ресторанный бизнес: Оплата в ресторанах и кафе - это еще один вид операций, который становится все популярнее. По статистике, около 3% всех операций с банковскими картами относятся к ресторанным услугам.

5. Здравоохранение: Оплата медицинских услуг, страховок и лекарств также может производиться с помощью банковских карт. Этот вид операций составляет около 2% от всех операций с банковскими картами.

6. Развлечения: Билеты в кино, театр, концерты и другие развлекательные мероприятия также можно приобрести с помощью банковских карт. Они составляют около 1% всех операций с банковскими картами.

\*\* пирог

1.1. Проблема и ее актуальность.

Так как эти операции проводятся через интернет, то к ним можно получить доступ и нанести тем самым ущерб. Сорвать сделку и многие люди потеряют работу. Не закупить оборудование и люди в больницах будут чувствовать себя плохо. Простое хищение средств, незаконный перевод на другой счет. Даже закупить, но не тот материал – и здание рухнет через год. Это были частные примеры. Чтобы ближе понять проблему. Если смотреть шире, то мошенничество может приводить к:

1. Финансовые потери: Когда карта крадется или ее данные компрометируются, мошенники могут использовать ее для совершения покупок и снятия денег со счета без разрешения владельца карты. Это может привести к серьезным финансовым потерям для клиента, банка и других организаций.

2. Ущерб репутации: Если банк не защищает своих клиентов от мошеннических операций, это может негативно повлиять на его репутацию. Клиенты могут потерять доверие к банку и перейти к конкурентам, что может привести к серьезным финансовым потерям.

3. Негативный влияние на экономику: Мошенничество с кредитными картами может привести к негативным последствиям для экономики в целом. Это может привести к увеличению стоимости кредитования и ухудшению кредитного рейтинга в стране.

4. Угроза безопасности: Кража или компрометация данных клиентов может привести к серьезной угрозе их безопасности. Кроме того, мошенничество с кредитными картами может быть связано с другими видами преступлений, такими как кража личных данных и кибератаки.

5. Увеличение расходов на безопасность: Для того, чтобы предотвратить мошенничество с кредитными картами, банки и другие организации вынуждены тратить большие деньги на усиление мер безопасности. Это может привести к увеличению расходов на безопасность для банков и других организаций, что может отразиться на клиентах в виде увеличения комиссий и сборов.

На прямую влияет на качество жизни. Думаю, проблему мы уяснили.

1.2. Необходимость решения.

Чтобы не было перечисленнх выше проблем, нужно находить решения. Рассмотим уже существущие.

Уже сейчас активно находят решение проблем мошенничества. Банки внедряют новые технологии в свою систему транзакций.

1. Мониторинг транзакций: Банки могут мониторить транзакции, чтобы обнаружить любую необычную активность, такую как большие покупки, покупки из других стран или транзакции, произведенные в необычное время. Если банк обнаружит такую активность, он может связаться с владельцем карты, чтобы уточнить, была ли эта транзакция разрешена.

2. Системы защиты от мошенничества: Банки используют различные системы защиты от мошенничества, такие как 3D Secure, которые требуют от владельца карты ввести дополнительный пароль или код подтверждения при совершении онлайн-транзакции. Это помогает защитить карту от несанкционированных покупок.

3. Экспертная система анализа данных: Банки используют системы анализа данных, чтобы выявлять необычные паттерны транзакций и активности на карте. Эти системы могут быстро обнаруживать мошеннические схемы и предотвращать их до того, как они причинят серьезный ущерб.

4. Фильтрация данных: Банки также используют фильтрацию данных для защиты от мошенничества. Это может включать в себя фильтрацию IP-адресов или блокировку определенных типов транзакций, которые могут быть связаны с мошенничеством.

5. Использование технологии биометрии: Банки также могут использовать технологию биометрии, такую как сканирование отпечатков пальцев или распознавание лиц, чтобы подтвердить личность владельца карты. Это помогает предотвратить мошенничество, связанное с утерей карты или утечкой информации.

6. Внедрение новых технологий: Банки постоянно ищут новые технологии, которые могут помочь улучшить безопасность и защиту от мошенничества. Например, некоторые банки начали использовать блокчейн-технологии для обеспечения безопасности транзакций.

7. Сотрудничество с правоохранительными органами: Банки могут сотрудничать с правоохранительными органами, чтобы обнаружить и пресечь мошеннические операции, связанные с кредитными картами. Они могут обмениваться информацией и работать вместе для того, чтобы выявить мошенников и привлечь их к ответственности.

1.3. Мои цели и задачи.

Целью моей курсовой работы является разработка системы классификации транзакций в реальном времени на мошенническую или нет. Чтобы пользователь при совершении транзакции проходил определенную процедуру, которая оценивает надежность его транзакции. Затем можно передать эту оценку владельцу и банку для их дальнейших действий.

Для этого мне нужно работать с личными данными клиента и данными о его транзакции. Следовательно мне нужно обеспечить безопасность хранения и обработки данных.

Мне нужно выбрать значащие признаки из операций.

На основе их обучить модели бинарной классификации.

Мне нужно интерпретировать результаты, выбрать подход.

Сделать выводы о транзакции.

2. Инструменты для решения задачи.

Python – высокоуровневый язык программирования. Он является интерпретруемым языком со строгой динамической типизацией. Он широко используется в различных областях, таких как веб-разработка, научные вычисления, искусственный интеллект, машинное обучение, анализ данных, создание игр и многое другое. Так же он известен своими богатыми инструментариями для работы с данными, математикой и алгоритмами. Предоставляет удобный и гибкий интерфейс. Очень юогатая поддержка сообщества, постоянно развивается.

Numpy (Numerical Python) - это библиотека языка программирования Python для работы с числовыми массивами и матрицами. Она предоставляет мощные функции для работы с многомерными массивами, включая математические, логические, манипуляционные и сортировочные операции.

Она также предоставляет многие функции для выполнения математических операций, таких как линейная алгебра, трансформация Фурье и случайные числа.

Написана на С и Fortran – что существенно ускоряет ее работу

Pandas - библиотека языка программирования Python для обработки и анализа данных. Она предоставляет удобный и эффективный способ работы с таблицами данных предоставляет множество функций для работы с данными, включая сортировку, фильтрацию, группировку, объединение, агрегацию и многое другое. Она также предоставляет возможность чтения и записи данных из различных источников, таких как CSV, Excel, SQL-базы данных.

Matplotlib - это библиотека языка программирования Python для визуализации данных. Она позволяет создавать графики, диаграммы, гистограммы, рисунки и многое другое с помощью нескольких простых команд.

Seaborn - это библиотека визуализации данных на основе Matplotlib, которая позволяет создавать красивые и информативные графики и диаграммы с минимальным количеством кода. Seaborn предоставляет набор функций для создания различных типов графиков, таких как линейные графики, гистограммы, круговые диаграммы, диаграммы рассеяния и многое другое. Она также предоставляет множество параметров для настройки внешнего вида графиков

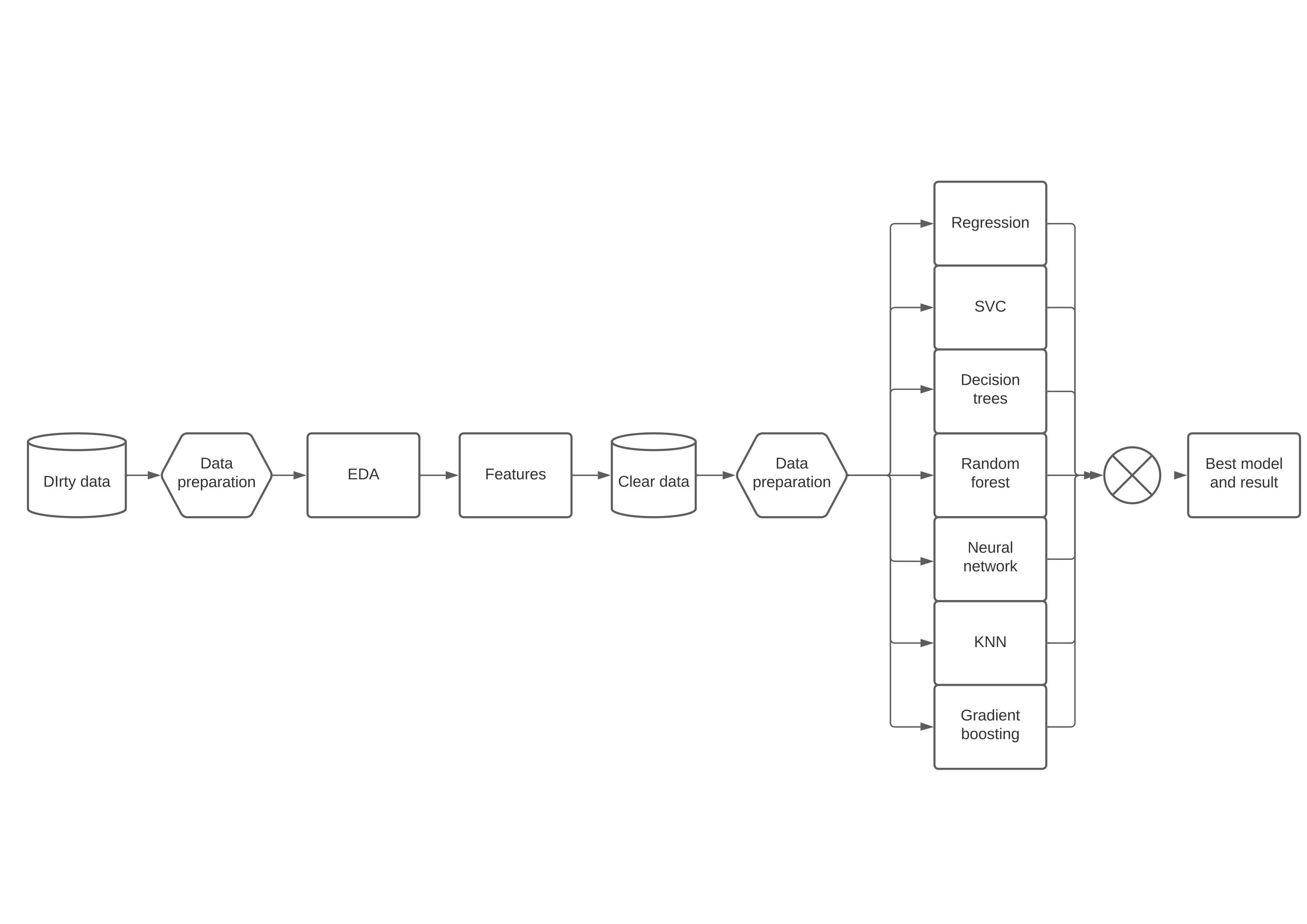
Scikit-learn - Scikit-learn (или sklearn) - это библиотека машинного обучения для языка программирования Python, которая предоставляет инструменты для решения задач классификации, регрессии, кластеризации и обработки данных. Scikit-learn базируется на библиотеках NumPy и SciPy, и является одной из наиболее популярных библиотек машинного обучения в Python.

Scikit-learn также предоставляет инструменты для предобработки данных, включая масштабирование, нормализацию, преобразование признаков и многое другое. Она также позволяет выполнить сбор и предварительную обработку данных из разных источников, включая файлы CSV, базы данных и т.д.

PostgreSQL - это мощная, открытая объектно-реляционная система управления базами данных. Она предоставляет широкие возможности для хранения и обработки структурированных данных, таких как таблицы, столбцы и строки. PostgreSQL поддерживает множество функций, включая транзакции, индексы, субдиаграммы, подзапросы, процедуры и триггеры. Она также обеспечивает безопасность данных с помощью авторизации пользователей, управления доступом и шифрования данных.

3. Архитектура приложения.

Рассмотрим архитектуру приложения, логический ход работы программы.



Имеется база данных с исходными данными о транзакциях пользователей.

Все остальные блоки представляют из себя python-скрипты, которые обрабатывают данные. Такие как:

1. Преобразование данных.

2. Расследовательный анализ данных.

3. Выделение признаков.

4. Запись признакового пространства в базу данных.

5. Подготовку данных к использованию моделями.

6. Моделирование и его оценка.

7. Решение поставленной задачи и его интерпретация.

3.1 База данных.

База данных:

В качестве хранилища конфиденциальных данных о транзакциях клиента используется PostgreSQL. Она имеет встроенные функции безопасности и возможности их настроить.

Кластер лежит локально на жестком диске. Используется одна база данных “NoNameBank”, таблица “train\_transactions”, таблица “test\_transactions” и пользователь “analytic” с правами:

- Создание таблиц.

- Чтение таблиц.

- Запись в им созданные таблицы.

Остальное ему запрещено. Подключение к базе данных с паролем, аутентификация пользователя с паролем, подключение к таблицам “transactions” с паролем.

Данные надежно защищены и соответствуют правилам обработки персональных данных.

3.2 EDA

Блок EDA основной по важности. Его рассмотрим далее во всех красках и графиках. В нем рассматриваются сами данные и ищутся зависимости между данными и целевой переменной – фактом мошенничества.

3.3 Модели.

В данной работе для решения задачи классификации использовались следующие алгоритмы машинного обучения:

1. Логистическая регрессия - это статистическая модель, которая используется для моделирования вероятности наступления определенного события. Логистическая регрессия применяется для бинарной классификации, где выходом является бинарное значение, например, 0 или 1.

2. Метод опорных векторов – это метод, который строит гиперплоскость или несколько гиперплоскостей в пространстве высокой размерности для разделения данных на классы. SVM также может быть использован для многоклассовой классификации.

Может работать с несбалансированными данными.

3. Решающие деревья - это модель, которая использует деревья для принятия решений на основе различных признаков. Решающие деревья также могут быть использованы для многоклассовой классификации.

4. Случайный лес – может работать с несбалансированными данными. Лучше сбалансировать. Случайный лес также может быть использован для задач бинарной классификации. Для этого, каждый дерево строится на случайном подмножестве данных и признаков, а затем все деревья комбинируются в один классификатор.

В задачах бинарной классификации, случайный лес может быть особенно полезен, так как он способен обрабатывать данные с большим количеством признаков, а также может работать с различными типами признаков. Кроме того, случайный лес может обработать несбалансированные данные, когда количество примеров в одном классе значительно превышает количество примеров в другом.

5. KNN - это алгоритм, который определяет класс объекта на основе его близости к другим объектам. Он используется для поиска K ближайших объектов и принятия решения на основе того, какие классы у этих объектов.

Основываясь на приведенном выше EDA, мы обнаружили, что такие функции, как сумма транзакции, возраст держателя кредитной карты, категория расходов, время транзакции и местоположение, имеют разную степень корреляции с мошенничеством с кредитными картами. Это помогает нам выбрать, какие функции мы хотим включить в наши модели данных. План состоит в том, чтобы обучить модели на наборе обучающих данных, который мы проанализировали выше, а затем использовать набор тестовых данных для оценки производительности модели.

- Плохо

6. Бустинг над решающими деревьями == Градиентный бустинг. -

это метод машинного обучения, который использует ансамбль из решающих деревьев для улучшения точности прогнозирования. Он основан на идее последовательного построения слабых моделей, каждая из которых нацелена на исправление ошибок предыдущих моделей.

Заключительным этапом работы является ответ на поставленную задачу, решение поставленной проблемы, оценка этого решения и его интерпретация.

4. Доступ к данным.

Подключаемя к таблице “transactions” пользователем “analytic”.

Признаки, которые описывают транзакцию:

1. index – уникальный номер транзакции. int

2. trans\_date\_trans\_time – дата и время проведения транзакции. object

3. cc\_num – номер кредитной карты клиента.

4. merchant – наименование продавца.

5. category – категория товара.

6. amt – сумма транзакции.

7. first – имя держателя карты.

8. last – фамилия держателя карты.

9. gender – пол держателя карты.

10. street – адрес держателя карты.

11. city – город держателя карты.

12. state – штат проживания держателя карты.

13. zip – почтовый индекс держателя карты.

14. lat – географическая широта держателя карты.

15. long - географическая долгота держателя карты.

16. city\_pop – население города проживания.

17. job – рабочая должность держателя карты.

18. dob – дата рождения держателя карты.

19. trans\_num – номер транзакции.

20. unix\_time – время совершения транзакции в формате unix.

21. merch\_lat – географическая широта продавца.

22. merch\_long - географическая долгота продавца.

23. is\_fraud – факт мошенничества.

5. Очистка данных.

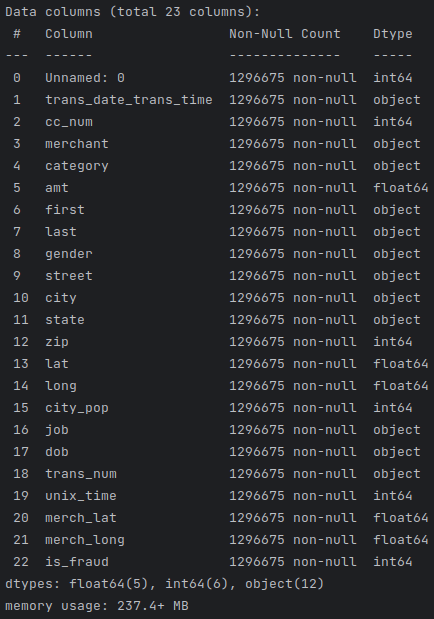
Произведем загрузку таблицы в датафрейм пандас. И проведем очистку:

1. Удалим дубликаты.

2. Посмотрим на количество нулевых значений в таблице:

Их 0

3. Посмотрим на описание датафрейма и узнаем типы данных его колонок:

4. Переименуем колонки:

trans\_date\_trans\_time' – 'transaction\_time'

'cc\_num' – 'card\_number'

'amt' – 'amount(usd)',

'trans\_num' – 'transaction\_id'

удалим колонку индексов.

Создадим новые колонки:

time – unix дата и время

hour\_of\_day – час суток.

Age – возраст держателя карты.

day\_of\_week – день недели.

То же самое проделали и с тестовой выборкой.

6. EDA.

Проведем расследовательный анализ данных. В нем предлагается проверить ряд гипотез и выявить явные закономерности в данных и как они коррелируют с целевой переменной — фактом мошенничества.

Целью этого анализа является выделение значащих признаков

0. Рассмотрим статистистическое описание выборки. Можно что-то высматривать, искать занономерности, но рано еще.

1. Рассмотрим матрицу корреляции. Из нее можем увидеть, что координаты продавца и держателя карты очень хорошо коррелируют, значит они находятся рядом. Одни и те же координаты. Но опять же нельзя пока пользоваться этой матрицей. Далее узнаем почему.

2.Рассмотрим сколько у нас здоровых транзакции и мошеннических.

И вот причина, почему нельзя пользоваться матрицей корреляции и смотреть на общую статистику. Они описывают несбалансированную выборку. Процент мошеннических транзакций всего 0.579%. Это очень мало. В дальнейшем нужно будет учитывать эту несбалансированность при обучении.

3. Гипотеза: кого обманывают больше, женщин или мужчин?

Проверяем и получаем ответ, что их обманывают в равной степени.

4. Гипотезы с возрастом.

4.1. Предлагается рассмотреть количество мошеннических транзакций в каждой возрастной категории для каждого пола.

Вывод:

1. мужчины более подвержены мошенничеству в возрасте 32 лет и в возрасте от 54 до 56 лет. Наблюдается всплеск в возрасте 82-83 лет.

2. Женщины более подвержены мошенничеству в возрасте 51 года. Мошеннические инциденты относятся к возрасту от 29 до 36 лет и от 46 до 51 и от 59 до 68 лет.

4.2. Предлагается рассмотреть возрастное распределение транзакций независимо от пола.

Вывод: Распределение по возрасту различается между двумя типами транзакций. В не мошеннических транзакциях есть 2 пика в диапазонах 37-38 и 49-50. Это означает, что эти возрастные диапазоны более устойчивы к мошенничеству. В мошеннических транзакциях распределение немного более плавное. А второй пик включает в себя более широкий диапазон 50-65. Это означает, что пожилые люди потенциально более подвержены мошенничеству, чем молодежь.

5. Предлагается рассмотреть зависимость на какую сумму обманывают в течении всего времени, что представлено в выборке.

Вывод: видны три диапазона сумм. Первый 0-50 примерно. Второй: 250-350 долларов, а третий более редкий диапазон 700-1200 долларов. В большинстве случаев мошеннические операции материализуются в этих диапазонах. Здесь очень четкая закономерность.

6. Гипотезы со временем.

6.1. Предлагается рассмотреть суммы разного рода транзакций в каждый месяц, представленный в выборке.

Вывод: в декабре происходит наибольшее количество всех видов сделок. Это праздники. А летом тоже много сделок. Это тоже праздники и каникулы. Могут быть летние распродажи на рынках.

6.2. Предлагается рассмотреть суммы только мошеннических транзакций в каждый месяц.

Вывод: Больше мошеннических операции, как и предполагалось, это декабрь, канун Нового года и Рождества, праздники. Следующий это май и предыдущий год январь и февраль.

6.3. Предлагается рассмотреть мошеннические транзакции в каждый час суток.

Вывод: на графике видно две ярко выраженные временные группы. Это 22-23 (10-11pm) и 0-3 (0-3pm). Абсолютное большинство мошеннических операций совершается ночью. А к вечеру инцидентов становится больше.

6.4. Предлагается рассмотреть случаи мошенничества по дням недели.

Вывод: Не очень информативно, но было интересно. Наибольшее количество происшествий приходится на выходные и понедельник.

7. Гипотезы с категорией товара.

7.1. Предлагается рассмотреть каждую категорию и сумму транзакции с частотой в 1 неделю.

Выводы: всего 14 категорий.

1. Entertai: Мошенничество в пределах 400-600 долларов США.

Не мошенничество составляет около 50 долларов США.

2. Food\_dinning: Мошенничество в пределах 100-140 долларов США.

Не мошенничество составляет около 30 долларов США.

3. Gas\_transport: Мошенничество от 20 долларов и меньше.

Не мошенничество составляет около 60-65 долларов США.

4. Grocery\_net: Мошенничество 50-60.

Не мошенничество составляет около 20 долларов США и меньше.

5. Grocery\_pos: Мошенничество: 300-350.

Не мошенничество составляет около 115 долларов США.

6. Health\_fitness: Мошенничество около 20.

Не мошенничество около 55 долларов США.

7. Home: Мошенничество около 250 и чуть больше.

Не мошенничество составляет около 50 долларов США.

8. Kids\_pets: Мошенничество около 60.

Не мошенничество 20 и меньше.

9. Mics\_net: Мошенничество около 800.

Не мошенничество 80.

10. Mics\_pos: Мошенничество меньше 20 и слишком большой.

Не мошенничество — 50.

11. Perconal\_care: Мошенничество около 45-50 usd.

Не мошенничество – 20 и больше.

12. Shopping\_net: Мошенничество около 1000 долларов.

Не мошенничество составляет около 50.

13. Shopping\_pos: Мошенничество 750 -1000 usd.

Не мошенничество составляет около 50.

14. Travel: Мошенничество — это слишком маленькие суммы. Не мошенничество примерно 50-150.

7.2. Предлагается рассмотреть процентное количество транзакций мошенничества в каждой категории.

Вывод: Можно выделить 4 категории с наиболее мошенническим влиянием: shopping\_net, misc\_net, product\_pos и shopping\_pos.

8. Гипотезы с продавцом.

Будут рассматриваться топ 15 продавцов с самым большим товарооборотом.

8.1. Предлагается рассмотреть продавцов и на суммы их транзакций с частотой в одну неделю.

Вывод: Мошенничество часто присутствует при небольшой сумме, менее 20 долларов США, и наоборот при большой сумме (около 300 и 1000 долларов США).

8.2 Предлагается рассмотреть процентное количество транзакций мошенничества с каждым продавцом.

Вывод: Можно выделить 5 продавцов с самым мошенническим влиянием.

9. Гипотезы с местоположением.

9.1. Предлагается рассмотреть количество мошеннических транзакций в каждом штате.

Вывод: можно выделить 27 штатов из 51 с наиболее мошенническим влиянием.

9.2. Предлагается посмотреть на процентную разницу между мошенническими и не мошенническими транзакциями.

Вывод: в Нью-Йорке и Огайо процент мошеннических транзакций выше, чем в других. Корреляция существует.

9.3. Предлагается посмотреть количество разного рода транзакций в 20 самых показательных для этого городах.

Вывод: по видимым признакам можно определить, что мошенничество преобладает в малонаселенных городах. Это почти 50%.

10. Предлагается рассмотреть профессии клиентов, количество и сумму мошеннических транзакций.

Вывод: визуально можно определить 10 профессий, наиболее подверженных мошенничеству.

В результате расследовательного анализа данных можно выделить 9 значащих признаков: возраст, сумма транзакции, время суток, категория товара, продавец, население города, профессия и факт мошенничества.

Контрольной точкой становится запись в базу данных новой таблицы, которая содержит только значащие признаки. Такие же признаки берутся и из тестовой выборки.

7. Подготовка данных к обучению моделей.

Рассмотрим полученную выборку. В ней содержится 1296675 записей. Нет пропущенных значений. Признаки: category, merchant, state, job являются строковыми. Age, amount(usd), hour\_of\_day, city\_pop, is\_fraud — числовые.

Для обучения классификаторов нужно преобразовать строковые признаки.

Способы преобразования, из достоинства и недостатки:

1. OrdinalEncoder - это метод кодирования категориальных признаков в числовые значения, где каждому уникальному значению категориального признака присваивается уникальный целочисленный код. Этот метод используется, когда значения категориальных признаков могут быть упорядочены или имеют важность в порядке следования.

Другими словами: закодировать каждое уникальное значение категориального признака уникальным целочисленным кодом, основанным на их порядке или значимости.

2. Label Encoding - В этом методе каждое уникальное значение признака заменяется соответствующим числовым значением. Например, если признак "цвет" принимает значения "красный", "зеленый" и "синий", то "красный" может быть заменен на 0, "зеленый" на 1 и "синий" на 2. Этот подход может работать хорошо для моделей, которые могут обрабатывать числовые данные, но он может приводить к неправильной интерпретации порядка между значениями, если порядок не является существенным.

Друими словами: закодировать каждое уникальное значение категориального признака уникальным целочисленным кодом. Этот метод применяется, когда значения категориального признака не имеют порядка и не важны их значения.

3. One-Hot Encoding - Это самый распространенный подход к представлению категориальных признаков. В этом методе каждый уникальный значок признака представляется в виде бинарного вектора длиной, равной количеству уникальных значений признака. Вектор состоит из нулей и одной единицы, которая соответствует индексу соответствующего значения признака. Этот подход может приводить к большому количеству признаков в случае большого количества уникальных значений.

Другими словами: создать бинарные столбцы для каждого уникального значения категориального признака. Этот метод применяется, когда значения категориального признака не имеют порядка, и все значения равноправны.

4. Count Encoding - заменить каждое значение категориального признака количеством раз, которое оно появляется в исходном наборе данных.

5.Target Encoding - заменить каждое значение категориального признака средним значением целевой переменной для этого значения.

Если буду использовать, то все категориальные признаки будут иметь одинаковые значения. Одинаковые столбцы.

Для данной конкретной задачи будут полезны следующие способы кодир ования: Ordinal Encoding, Label Encoding, Count Encoding.

8. Балансировка выборки.

Есть достаточно способов решить проблему дизбаланса в данных. Можно применять алгоритмы, которые могут работать с несбалансированной выборкой:

1. Decision Tree (Деревья решений): Модель дерева решений может быть использована для классификации несбалансированных данных путем настройки параметров модели, таких как глубина дерева и ограничение минимального количества объектов в листьях.

2. Random Forest (Случайный лес): Random Forest, который является ансамблевой моделью деревьев решений, может быть использован для классификации несбалансированных данных путем настройки параметров модели и выбора соответствующих метрик оценки качества модели.

3. SVM (Support Vector Machine): Модель SVM может быть использована для классификации несбалансированных данных, если используются соответствующие методы для обработки дисбаланса данных, такие как увеличение числа примеров в миноритарном классе (oversampling) или уменьшение числа примеров в мажоритарном классе (undersampling).

4. Logistic Regression (Логистическая регрессия): Логистическая регрессия может быть использована для классификации несбалансированных данных, если используются соответствующие методы для обработки дисбаланса данных, такие как увеличение числа примеров в миноритарном классе (oversampling) или уменьшение числа примеров в мажоритарном классе (undersampling).

5. Gradient Boosting (Градиентный бустинг): Градиентный бустинг может быть использован для классификации несбалансированных данных, если используются соответствующие методы для обработки дисбаланса данных, такие как увеличение числа примеров в миноритарном классе (oversampling) или уменьшение числа примеров в мажоритарном классе (undersampling).

Так же было упомянуто про увеличение числа примеров в миноритарном(меньшем) классе (oversampling) или уменьшение числа примеров в мажоритарном(большем) классе (undersampling).

Еще можно балансировать выборку с помощью SMOTE или ADASYN.

SMOTE (Synthetic Minority Over-sampling Technique) - это метод oversampling, используемый для обработки несбалансированных данных. SMOTE генерирует синтетические данные для миноритарного класса, чтобы увеличить количество его примеров и сбалансировать данные.

SMOTE основан на том, что если мы возьмем два случайных ближайших соседа из миноритарного класса, мы можем создать новый пример, который находится на линии между двумя соседями. Таким образом, SMOTE использует метод интерполяции для генерации новых примеров.

Процесс работы SMOTE следующий:

1. Выбирается случайный пример из миноритарного класса.

2. Выбирается один из его k ближайших соседей.

3. Генерируется новый пример, который находится на линии между выбранным примером и одним из его k ближайших соседей.

4. Этот процесс повторяется для всех примеров в миноритарном классе, пока не достигнуто желаемое количество новых примеров.

SMOTE позволяет увеличить количество примеров миноритарного класса, что помогает сбалансировать данные. Однако SMOTE также может привести к переобучению, если количество синтетических данных будет слишком велико. Поэтому важно оптимизировать количество создаваемых синтетических примеров и подбирать оптимальное значение параметра k.

ADASYN:

ADASYN (Adaptive Synthetic Sampling) - это алгоритм, используемый для балансировки несбалансированных данных путем генерации синтетических примеров. Этот алгоритм является адаптивным, поскольку он увеличивает количество синтетических примеров для классов, которые трудно классифицировать, и меньше синтезирует для классов, которые уже хорошо представлены.

Алгоритм ADASYN работает следующим образом:

1. Определить, насколько несбалансированы данные. Это делается путем подсчета соотношения количества примеров меньшего класса к примерам большего класса.

2.Для каждого примера меньшего класса определить количество ближайших соседей из большего класса.

3. Вычислить веса для каждого примера меньшего класса, которые определяют количество синтетических примеров, которые нужно сгенерировать для этого примера. Веса рассчитываются так, чтобы примеры с меньшим количеством соседей имели больший вес.

4. Сгенерировать синтетические примеры для каждого примера меньшего класса, используя метод SMOTE.

5. Добавить синтетические примеры к исходным данным.

ADASYN является улучшенной версией SMOTE, поскольку он адаптивно определяет, сколько синтетических примеров нужно сгенерировать для каждого примера меньшего класса в зависимости от того, насколько этот класс трудно классифицировать. ADASYN часто показывает лучшие результаты, чем SMOTE, в задачах балансировки несбалансированных данных.

9. Моделирование и оценка моделей.

В этой главе будет рассмотрено 7 моделей классификации:

Логистическая регрессия, метод опорных векторов, решающие деревья, случайный лес, нейронная сеть прямого распространения, К ближайших соседей и градиентгый бустинг(бустинг над решающими деревьями).

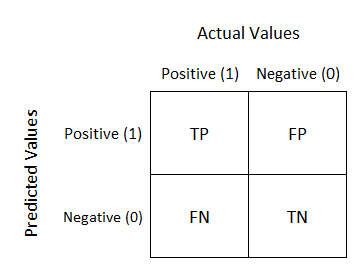
Они будут протестированы с кодировками: Ordinal Encoding, Label Encoding, Count Encoding и с различными решателями дисбаланса: SMOTE c подбираемым параметром k и ADASYN.

Будут предоставлены наилучшие варианты и те, которые обучились за «разумное время» (примерно 15-20 минут).

Забегая наперед, скажу здесь, что выборку лучше сбалансировать, хоть некоторые модели позволяют получать хорошие результаты и без балансировки.

Для оценки качества модели предлагается использовать:

1. Матрицу ошибок. Суть этой оценки в том, чтобы просто посчитать количество верных и неверных классификаций.



TP — True Positive: верный ответ 1,

ответ модели 1.

FP — False Positive: верный ответ 0,

ответ модели 1.

FN — False Negative: верный ответ 1,

ответ модели 0.

TN — True Negative: верный ответ 0,

ответ модели 0.

2. Отчет классификации.

Он использует оценки: precision, recall, f1-score для каждого класса и оценку accuracy. Подробнее про каждую.

Precision (точность) = - насколько можно доверять модели, когда она выдает положительный класс.

Recall (полнота) = - доля положительных объектов, которые охватывает модель.

Эти две оценки сполна описывают модель только в совокупности. Хорошая модель должна иметь высокие precision и recall.

F1-score (гармоническое среднее между precision и recall) = -

подходит для оценки несбалансированных выборок.

Accuracy (качество) =- доля верных ответов.

3. AUC-ROC — метрику.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve) - это метрика для оценки качества бинарной классификации. Она основывается на графике ROC-кривой, который показывает зависимость между TPR (True Positive Rate) и FPR (False Positive Rate).

TPR - это доля верно классифицированных положительных примеров относительно общего количества положительных примеров. FPR - это доля неверно классифицированных отрицательных примеров относительно общего количества отрицательных примеров.

ROC-кривая представляет собой кривую, которая показывает, как TPR и FPR меняются при изменении порогового значения для классификатора. Чем выше TPR и ниже FPR, тем лучше классификатор.

AUC-ROC - это площадь под ROC-кривой, которая может принимать значения от 0 до 1. Чем выше AUC-ROC, тем лучше качество классификации. Если AUC-ROC равна 0,5, это означает, что классификатор работает как случайное угадывание, а если AUC-ROC равна 1, то классификатор работает идеально.

9.1. Логистическая регрессия.

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.2. Метод опорных векторов.

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.3. Решающие деревья.

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.4. Случайный лес.

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.5. Нейронная сеть прямого распространения.

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.6. К-ближайших соседей

Таблица результатов:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

9.7. Градиентный бустинг.

Таблица результатов:

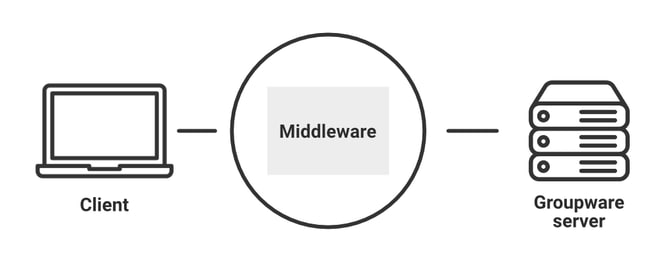
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | SMOTE | | | ADASYN | | |
| Ordinal | Label | Count | Ordinal | Label | Count |
| Logistic regression | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: | 0 prec:  0 rec:  0 f1:  1 prec:  1 rec:  1 f1:  ROC-AUC: |

Вывод:

---

10. Правила пользования.

Когда пользователь совершает денежную транзакцию в интернете, ему предоставляется веб-сервис в middleware уровне.



Этот веб-сервис содержит обученную модель и средства обмена информацией в среде с банком и пользователем и дальнейшим сервером. Этот посредник получает такие данные от пользователя как возраст, сумму транзакции, время суток, категорию товара, наименование продавца, местоположение клиента, популяция города, из которого производится операция, профессия клиента. И выдает оценку транзакции. Данную оценку получает и клиент и его банк. В случае необходимости и клиент и банк могут оперативно сработать на инцидент и предотвратить хищение денежных средств.

11. Перспектива.

Целью данной работы не было разработка вышеупомянутого веб-сервиса, это основное направление развития данного проекта.

Перспективой развития может выступать замена локальной базы данных или удаленного физического сервера на облачные сервисы и в общем создание для этого веб-сервиса распределенной облачной архитектуры.

Как пример, использовать Amazon Web Services:

IAM — «Одно кольцо, чтобы править всеми».

EC2 — Виртуальный сервер с операционной системой.

S3 — Хранение данных в файлах.

RDS — Сервис РБД и другие.

Использование Apache Airflow для составления порядка, приоритетов и сложных зависимостей в выполняемых операциях. Например поставить операцию выделение значащих признаков строго до параллельного обучения моделей, ведь, если на этапе выделения признаков произойдет ошибка, то не нужно тратить ресурсы на обучение. Огромная гибкость и приоритизация, так же графики запуска.

Использовать Apache Spark для параллельной обработки данных, которые горизонтально распределены в облачной системе. Очень большая скорость и возможности обработки.

Как вариант можно провести более глубокий анализ данных и выделить нетривиальные зависимости, мультиколлинеарные.

В заключение можно производить вычисления на графических процессорах, иначе говоря на видеокартах.

Литература