Григорьев Юрий, ст. гр. ИС-142, Выпускная Квалификационная Работа на тему “Реализация моделей машинного обучения для задачи обнаружения мошеннических операций”

1. Введение

1.1 Актуальность темы

С развитием информационных технологий и цифровой экономики число финансовых транзакций, совершаемых через интернет, банковские системы и мобильные приложения, продолжает стремительно расти. Это, в свою очередь, увеличивает и количество мошеннических операций, которые наносят ущерб как компаниям, так и пользователям. Сегодня закономерно возникает острая необходимость в разработке эффективных методов и инструментов для выявления таких операций в режиме реального времени.

Традиционные методы обнаружения мошенничества, основанные на жёстких правилах (*rule-based systems*), показывают свою ограниченность в современных условиях из-за увеличивающейся сложности атак и динамичности их природы. Правила, созданные экспертами, быстро устаревают, и мошенники легко адаптируются к новым ограничениям. В связи с этим на первый план выходят методы машинного обучения, которые способны автоматически адаптироваться к новым данным, выявляя скрытые закономерности в поведении пользователей и транзакций.

Машинное обучение, особенно с использованием нейронных сетей, позволяет создавать модели, которые могут эффективно выявлять как известные, так и новые виды мошенничества. Одним из ключевых вызовов при применении этих моделей является дисбаланс данных — когда число мошеннических операций крайне мало по сравнению с нормальными транзакциями. Это требует разработки специальных методов и подходов для повышения точности обнаружения мошенничества и снижения числа ложных срабатываний.

1.2 Цели и задачи исследования

Цель работы — провести исследование и анализ методов машинного обучения для обнаружения мошеннических операций в финансовых транзакциях, выявить наиболее эффективные модели и подходы для решения этой задачи с учетом дисбаланса классов и особенностей данных.

Для достижения этой цели были поставлены следующие задачи:

1. **Провести анализ** существующих методов машинного обучения для задачи обнаружения мошенничества, включая как классические алгоритмы, так и методы глубокого обучения.
2. **Исследовать подходы** к обработке дисбаланса классов, такие как oversampling (например, SMOTE), undersampling, а также методы перевзвешивания классов.
3. **Реализовать и обучить** несколько моделей машинного обучения (логистическую регрессию, деревья решений, случайные леса, градиентный бустинг, нейронные сети).
4. **Провести оценку качества** моделей на основе метрик precision, recall, F1-score, ROC-AUC, PR-AUC с учётом специфики задачи.

5. **Сравнить метрики** моделей и методов работы с данными, определить, какие подходы наиболее эффективны для решения задачи обнаружения мошеннических операций.
6. **Оценить интерпретируемость** моделей и их применение в реальной практике.

1.3 Структура работы

1. **Введение.** Рассматривает актуальность темы, формулирует цели и задачи исследования, а также кратко описывает структуру работы.
2. **Постановка задачи.** Описывает проблему обнаружения мошенничества в финансовых транзакциях, особенности датасетов, структуру данных и основные требования к моделям.
3. **Описание и реализация моделей.** В данном разделе будут рассмотрены и реализованы алгоритмы машинного обучения, включая простые методы (логистическая регрессия, деревья решений), ансамблевые методы (случайный лес, градиентный бустинг), методы глубокого обучения (нейронные сети), а также подходы к предварительной обработке данных, их балансировку и оценку качества получаемых результатов.
4. **Сравнение полученных результатов.** Здесь будет проведено подробное сравнение полученных результатов, анализ и влияние различных факторов на эффективность классификации.
5. **Заключение.** Содержит выводы по результатам работы, анализ достижения поставленных целей и предложенные рекомендации для дальнейших исследований.

Таким образом, в работе будет проведён всесторонний анализ подходов к обнаружению мошеннических операций с использованием машинного обучения. Основное внимание будет уделено исследованию моделей и методов обработки несбалансированных данных для улучшения качества классификации и повышению применимости решений в реальных условиях.

2. Постановка задачи

2.1. Описание проблемы

Задача обнаружения мошеннических операций в финансовых транзакциях является задачей **бинарной классификации**, в которой каждую транзакцию необходимо отнести к одному из двух классов: **«нормальные»** или **«мошеннические»**.

Основной вызов заключается в том, что мошеннические транзакции составляют очень малый процент от общего числа транзакций. Это явление называется **дисбалансом классов**, где «нормальные» транзакции преобладают, а число мошеннических операций минимально.

Например, в некоторых финансовых данных количество мошеннических транзакций может составлять менее 1% от общего числа. Это приводит к тому, что простое обучение моделей на таких данных часто приводит к неправильной классификации, когда модель склонна игнорировать редкие мошеннические случаи ради оптимизации общей точности.

Ключевыми задачами исследования являются:

- Максимизация точности обнаружения мошенничества (**precision**) для уменьшения числа ложных срабатываний.
- Увеличение полноты выявления мошенничества (**recall**) для снижения вероятности пропуска реальных мошеннических операций.
- Снижение влияния дисбаланса классов на модели.

2.2. Датасет и описание данных

Для исследования и экспериментов по выявлению мошеннических операций используется публичный датасет **Credit Card Fraud Detection**, доступный на платформе Kaggle. Датасет содержит данные о транзакциях, совершенных по кредитным картам европейскими картодержателями в сентябре 2013 года. Данные анонимизированы и предназначены для задач по классификации транзакций на нормальные и мошеннические.

Датасет состоит из **284,807 транзакций**, каждая из которых представлена набором признаков (features). Данные содержат следующие поля:

- **Time**: количество секунд, прошедших с момента первой транзакции в наборе данных.
- **V1-V28**: 28 анонимизированных признаков, полученных в результате применения метода главных компонент (**PCA**). Эти признаки скрывают исходные данные (номер лицевого счета, дата, местоположение, персональные данные отправителя и получателя) для защиты конфиденциальности.
- **Amount**: сумма транзакции в денежном выражении. Этот признак деанонимизирован.
- **Class**: целевая переменная, которая определяет тип транзакции:
    - 0 — нормальная транзакция.
    - 1 — мошенническая транзакция.

Данные о транзакциях сильно обезличены, что затрудняет использование дополнительных контекстуальных признаков, таких как тип торговой точки, местоположение или время дня.

Однако такие методы, как **PCA**, позволяют уменьшить размерность исходных данных и сохранить нужные паттерны для обучения моделей.

Основной сложностью в работе с этим набором данных является **сильный дисбаланс классов**. Мошеннические операции составляют менее 0,2% от общего числа транзакций:

- **Нормальные транзакции**: 284,315 (99.83%)
- **Мошеннические транзакции**: 492 (0.17%)

Этот дисбаланс классов делает задачу обнаружения мошенничества особенно сложной, так как модели могут быть склонны к предвзятости в сторону доминирующего класса (нормальных транзакций). Для того чтобы эффективно решать задачу классификации, необходимо применять специальные методы обработки данных и выбирать модели, которые устойчивы к таким дисбалансам.

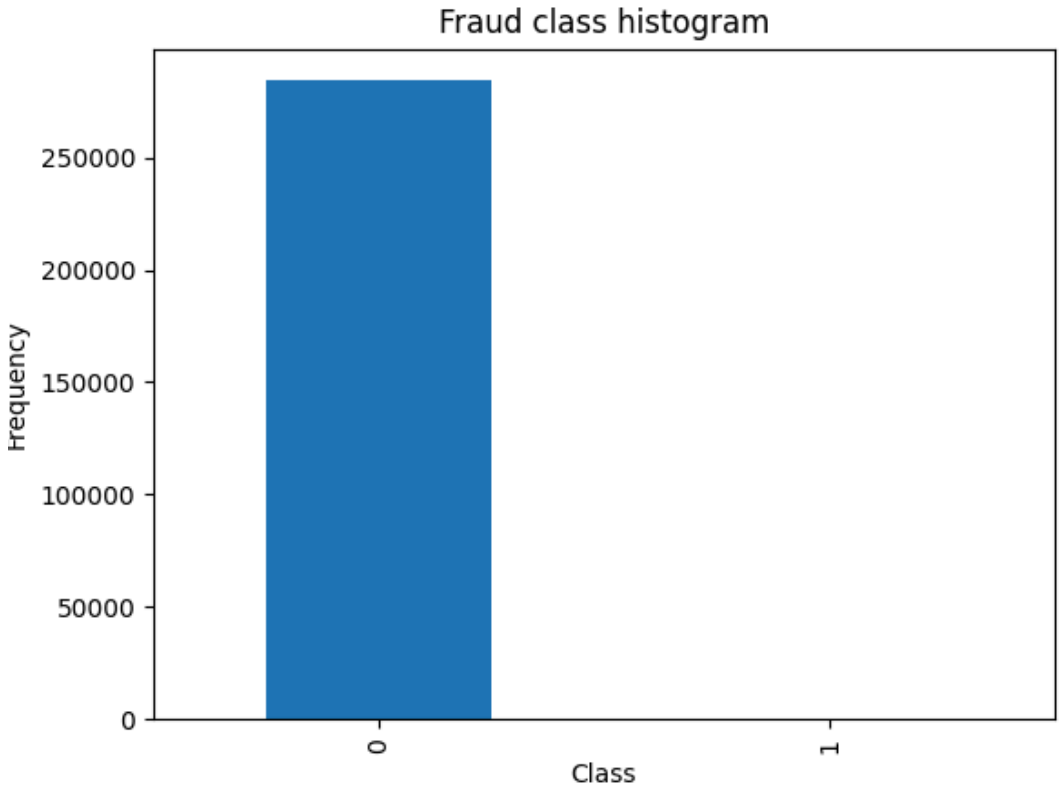

Рисунок 1. Распределение классов во взятом массиве данных

2.3. Основные требования к моделям

Для решения задачи классификации транзакций на нормальные и мошеннические предъявляются следующие ключевые требования к моделям:

1. **Высокая точность выявления мошенничества.**
    - Модель должна иметь высокую точность (precision), чтобы минимизировать ложные срабатывания — случаи, когда нормальные транзакции ошибочно классифицируются как мошеннические. Это особенно важно для финансовых учреждений, чтобы не терять доверие клиентов.

- При этом модель должна иметь высокую полноту (recall), чтобы эффективно выявлять как можно больше реальных мошеннических операций. Пропуск мошеннических транзакций может привести к значительным финансовым потерям, что недопустимо в реальной практике.

2. **Интерпретируемость моделей.**

   - Важно, чтобы модели были **интерпретируемыми**, особенно в финансовой сфере, где объяснимость решений имеет большое значение. Это позволяет регуляторам и финансовым организациям понимать, почему та или иная транзакция была помечена как мошенническая. Это требование ставит дополнительные ограничения на выбор моделей и подходов.
   - Например, традиционные алгоритмы, такие как **логистическая регрессия** или **деревья решений**, легче интерпретировать по сравнению с методами глубокого обучения. Однако интерпретируемость может стать компромиссом между точностью и возможностью объяснения результатов.

3. **Работа с дисбалансом классов.**

   - Модели должны эффективно справляться с дисбалансом данных, поскольку мошеннические транзакции встречаются крайне редко. Для этого применяются специальные методы, такие как **балансировка данных**, использование метрик для несбалансированных данных (например, **ROC-AUC**), или алгоритмы, способные компенсировать дисбаланс на уровне обучения.

4. **Реализация в реальных условиях.**

   - Модели должны быть не только высокоэффективными на этапе тестирования, но и способны работать в реальном времени, анализируя большие объемы данных с низкой задержкой. Это особенно важно для систем, отслеживающих транзакции в режиме онлайн, где скорость принятия решений является критически важной.

3. Описание и реализация моделей

3.1 Логистическая регрессия

Логистическая регрессия — простой и интерпретируемый метод бинарной классификации, который моделирует вероятность принадлежности объекта к одному из классов с помощью логистической функции (сигмоидной кривой). Функция выглядит следующим образом:

$$\sigma(x) = \frac{1}{1 + e^{-x}},$$

где x — линейная комбинация признаков X_1, …, X_n и параметров модели β_1, …, β_n.

Цель метода — подобрать такие параметры, которые минимизируют логистическую функцию потерь, что позволяет наилучшим образом разделить классы в пространстве признаков.

Преимущества:

- Простота и интерпретируемость: легко понять, как каждый признак влияет на вероятность целевого класса.
- Устойчивость к переобучению: при использовании регуляризации (например, L1 или L2), логистическая регрессия менее склонна к переобучению.

Недостатки:

- Линейность: логистическая регрессия предполагает, что зависимости между признаками и классами линейны, что может ограничивать ее эффективность в условиях сложных нелинейных взаимосвязей.
- Неустойчивость к дисбалансу классов: модель плохо работает, если один из классов встречается значительно реже, как это бывает в задачах детектирования мошенничества.

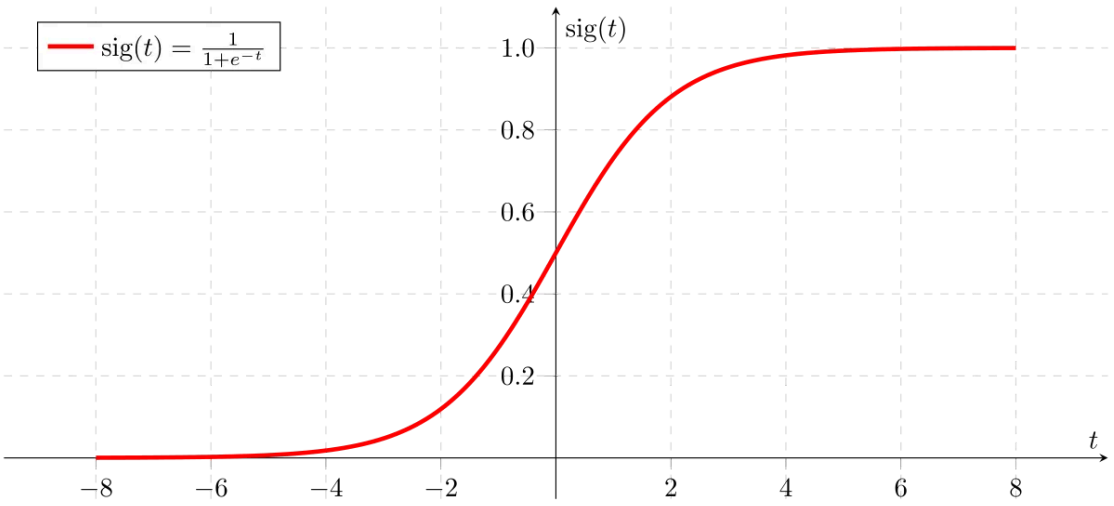

Рисунок 2. Логистическая функция (сигмоидная кривая)

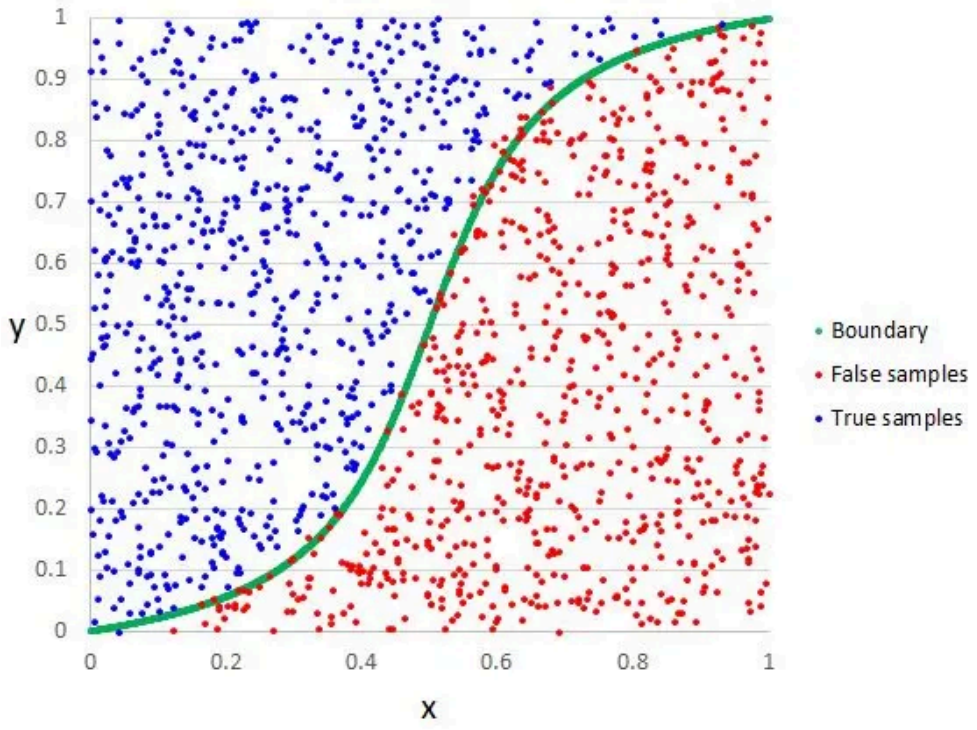

Рисунок 3. Пример работы модели логистической регрессии для задачи бинарной классификации

3.1.1 Предобработка данных

На этапе предобработки данных мы масштабировали числовые признаки, чтобы предотвратить проблему дисбаланса градиентов, которая может привести к замедлению сходимости или ухудшению точности модели. Нормализация, выполненная с помощью **StandardScaler** из библиотеки **SciKit-Learn**, позволила равномерно распределить значения и ускорить обучение.

3.1.2 Балансировка классов

Датасет был несбалансированным: доля мошеннических транзакций составляет менее 1% от общего числа. Чтобы уменьшить влияние этого дисбаланса, мы использовали перевзвешивание классов, назначив больший вес редкому классу. В дополнение был применен метод ***oversampling***, при котором мы увеличили количество примеров редкого класса до уровня класса большинства.

3.1.3 Построение модели

Для обучения модели я реализовал логистическую регрессию вручную, используя метод градиентного спуска. Минимизируемая функция потерь — бинарная кросс-энтропия:

$$L(y, \hat{y}) = -\frac{1}{N} \sum [\, y \cdot log(\hat{y}) + (1 - y) \cdot log(1 - \hat{y}) \,],$$

где y — истинные метки классов, а $\hat{y}$ — предсказанные вероятности принадлежности к положительному классу. Параметры модели (веса и смещение) обновляются в соответствии с градиентом функции потерь:

$$w := w - \eta \frac{\partial L}{\partial w}, \qquad b := b - \eta \frac{\partial L}{\partial b},$$

где η — скорость обучения.

3.1.4 Оценка качества модели

Для оценки модели использовались следующие метрики:

- **Accuracy (точность)** (0.9444): доля правильно классифицированных примеров.
- **Precision (точность положительного класса)** (0.9830): доля транзакций, предсказанных как мошеннические, которые действительно являются таковыми.
- **Recall (полнота)** (0.9046): доля всех мошеннических транзакций, которые были правильно обнаружены.
- **F1 Score** (0.9422): гармоническое среднее Precision и Recall, отражающее баланс между ними.
- **ROC-AUC (площадь под кривой ROC)** (0.9848): качество классификации независимо от выбранного порога.

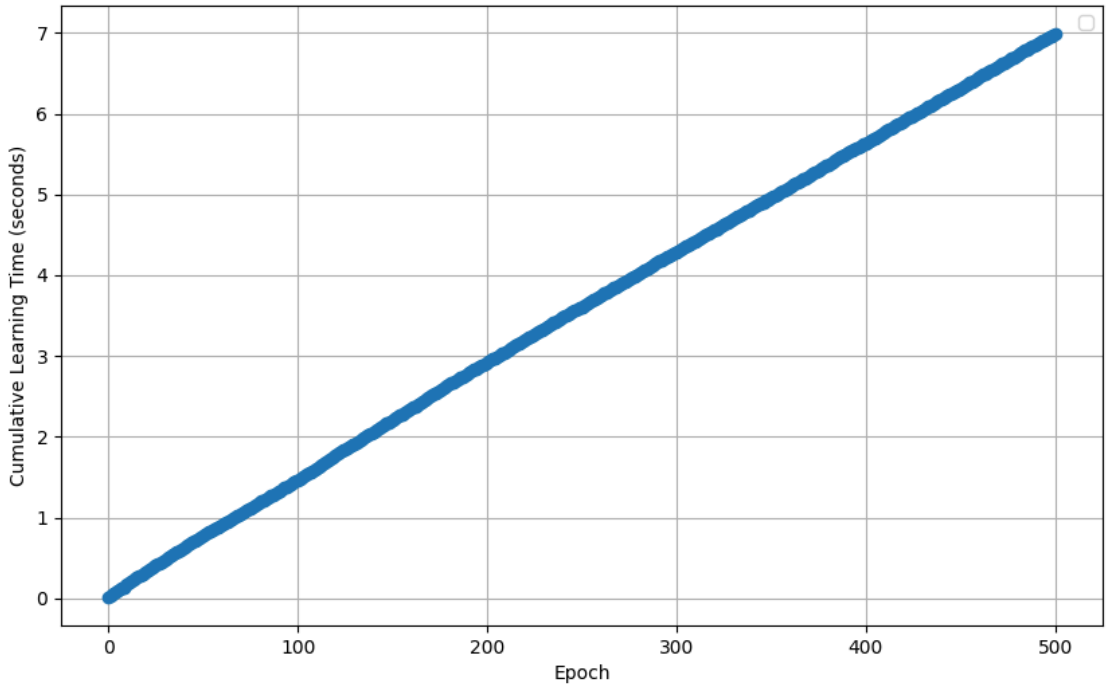

Рисунок 4. График зависимости кумулятивного времени обучения модели от пройденной эпохи

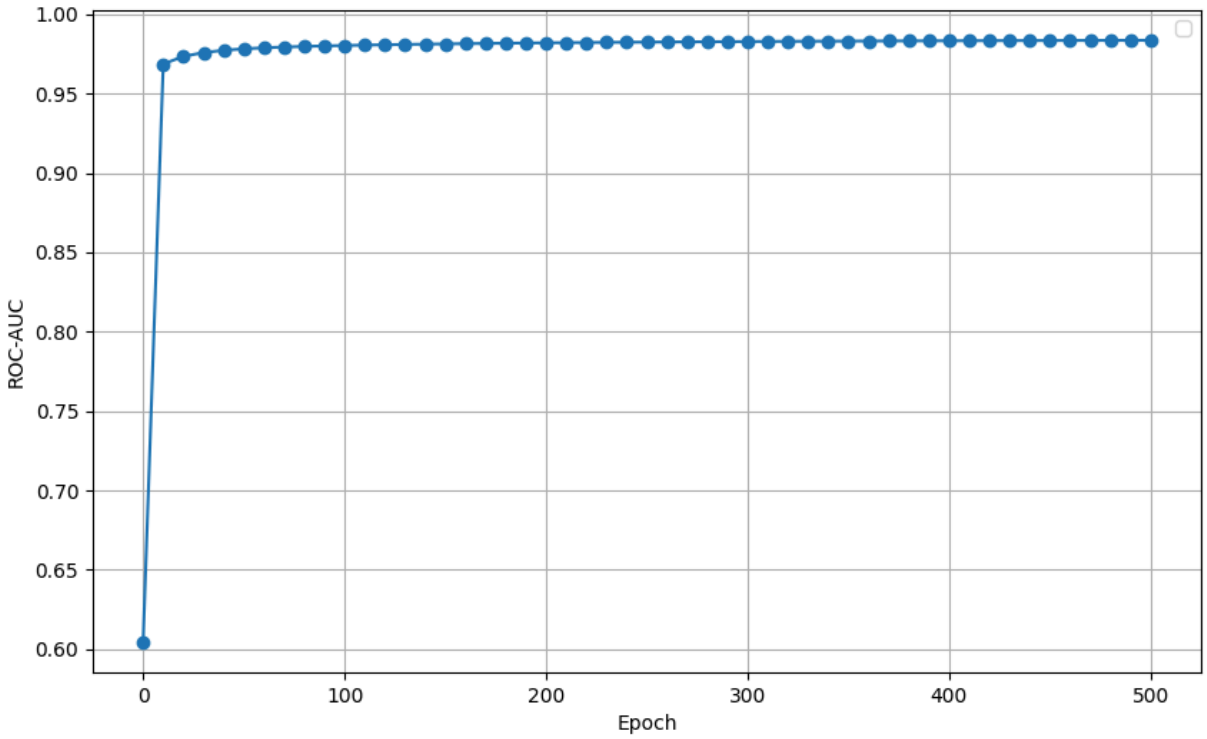

Рисунок 5. График зависимости показателя ROC-AUC от пройденной моделью эпохи обучения

3.2 Дерево решений

Дерево решений — это метод машинного обучения, который подходит для задач классификации и регрессии. Оно представляет собой древовидную структуру, в которой каждый узел выполняет проверку одного признака, а конечные листья возвращают предсказанный класс. Процесс классификации начинается от корневого узла и последовательно проходит через все промежуточные узлы, пока не достигает конечного листа с целевым классом. Критерий Джини использовался для разбиения узлов, так как он позволяет определять наиболее однородные группы на каждом этапе разбиения, что увеличивает точность модели.

Преимущества:

- Простота и интерпретируемость: деревья решений легко визуализировать и объяснять, что позволяет понять, как каждый признак влияет на результат.
- Гибкость: алгоритм хорошо справляется как с линейными, так и с нелинейными взаимосвязями между признаками, что позволяет находить более сложные зависимости.

Недостатки:

- Склонность к переобучению: дерево решений может быть чувствительно к выбросам и легко подстраиваться под особенности обучающих данных, особенно при большой глубине.

- Неустойчивость к шуму: небольшие изменения в данных могут привести к значительным изменениям в структуре дерева, что делает его менее стабильным по сравнению с другими методами.

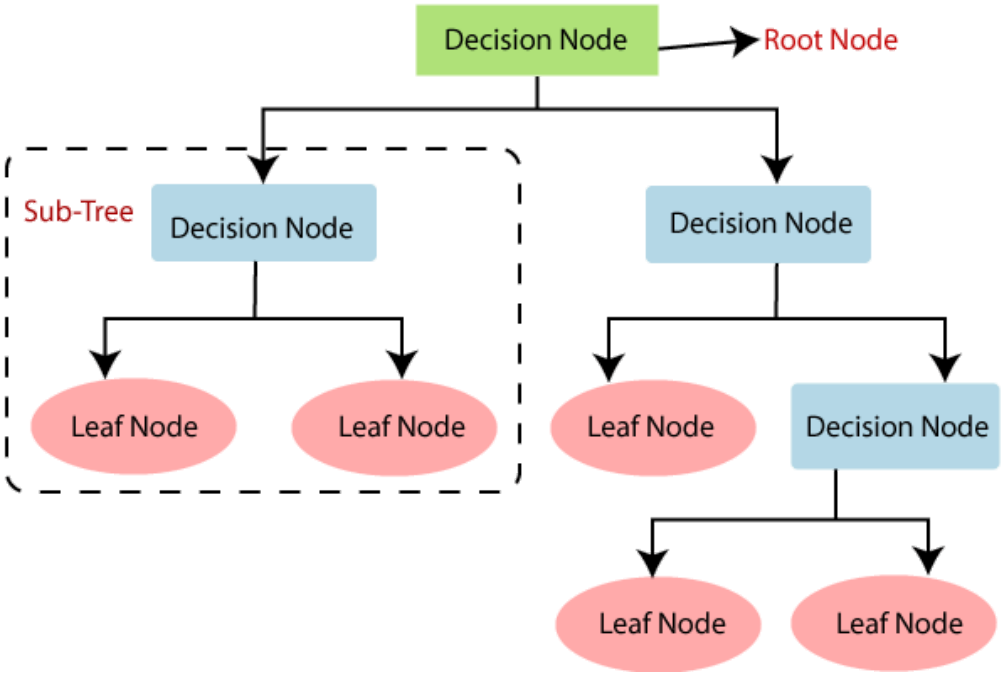

Рисунок 6. Структура дерева решений

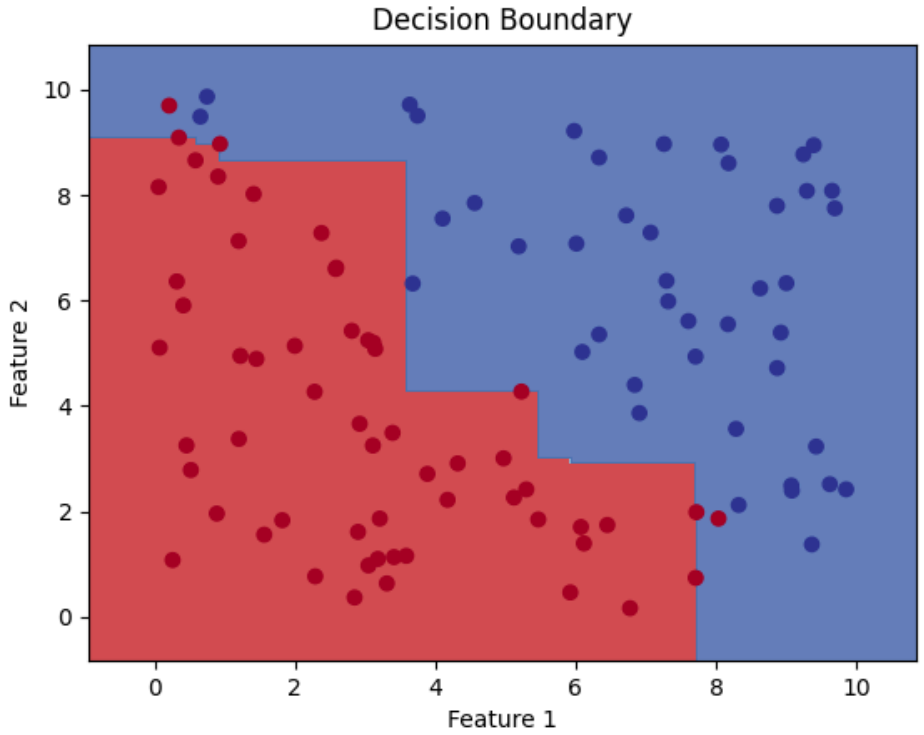

Рисунок 7. Пример работы модели Decision Tree для задачи бинарной классификации

3.2.1 Предобработка данных

Перед обучением модели деревьев решений проведено масштабирование числовых признаков и балансировка классов. Масштабирование помогает ускорить процесс обучения, а балансировка уменьшает влияние дисбаланса классов.

3.2.2 Балансировка классов

Для уменьшения дисбаланса классов был применен метод *oversampling*, что позволило увеличить количество примеров редкого класса до уровня класса большинства и обеспечить более равномерное разбиение классов в процессе построения дерева.

3.2.3 Построение дерева решений

Для обучения дерева решений использовался алгоритм, основанный на критерии Джини. Критерий Джини — это мера неоднородности *(нечистоты / dirtiness)* группы данных, которая помогает выбрать оптимальные разбиения в процессе построения дерева. Для каждого узла дерева определяется значение критерия, которое отражает, насколько случайным является распределение классов в текущей группе. Чем меньше значение критерия Джини, тем более однородны классы в группе, что делает данное разбиение «чистым» и более подходящим для построения точной модели.

Формула для расчета критерия Джини для одного узла выражается как:

$$Gini = 1 - \sum_{k=1}^{K} p_k^2,$$

где p_k — это вероятность того, что случайно выбранный элемент из группы принадлежит классу k, а K — общее количество классов (в нашем случае $K = 2$, так как задача бинарной классификации). Если все элементы в группе принадлежат одному классу, значение критерия Джини будет равно 0, что указывает на полную однородность группы.

На каждом уровне:

1. **Подсчитываются значения критерия Джини для всех возможных разбиений** — например, если мы рассматриваем разбиение по признаку «сумма транзакции», то проверяются различные пороговые значения для этой суммы, чтобы определить, какое из них минимизирует критерий Джини.
2. **Выбирается разбиение с наименьшим значением критерия Джини**. Это разбиение создаст более «чистые» подмножества данных, с большей долей однотипных классов, что поможет дереву лучше различать мошеннические и нормальные операции.
3. **Процесс повторяется для каждого узла**, пока дерево не достигнет максимальной глубины или минимального количества элементов в узле, что предотвращает излишнюю детализацию и переобучение модели.

При обучении дерева решений важно контролировать глубину дерева и минимальное число элементов в каждом узле, чтобы избежать чрезмерной адаптации модели к обучающим

данным, что приводит к снижению обобщающей способности модели на новых данных. В данном случае использовались ограничения по глубине и минимальному числу элементов в узле, что позволило добиться хорошего баланса между точностью модели и устойчивостью к переобучению.

3.2.4 Оценка качества модели

Для оценки модели использовались метрики аналогичные с оценкой модели логистической регрессии.

- **Accuracy (точность)**: 0.9349;
- **Precision (точность положительного класса)**: 0.9453;
- **Recall (полнота)**: 0.9235;
- **F1 Score**: 0.9343;
- **ROC-AUC (площадь под кривой ROC)**: 0.9349 – показатель общей эффективности классификации, независимый от порогового значения.

Модель дерева решений показала хорошие результаты в обнаружении мошеннических операций, особенно в условиях учета нелинейных взаимосвязей между признаками.

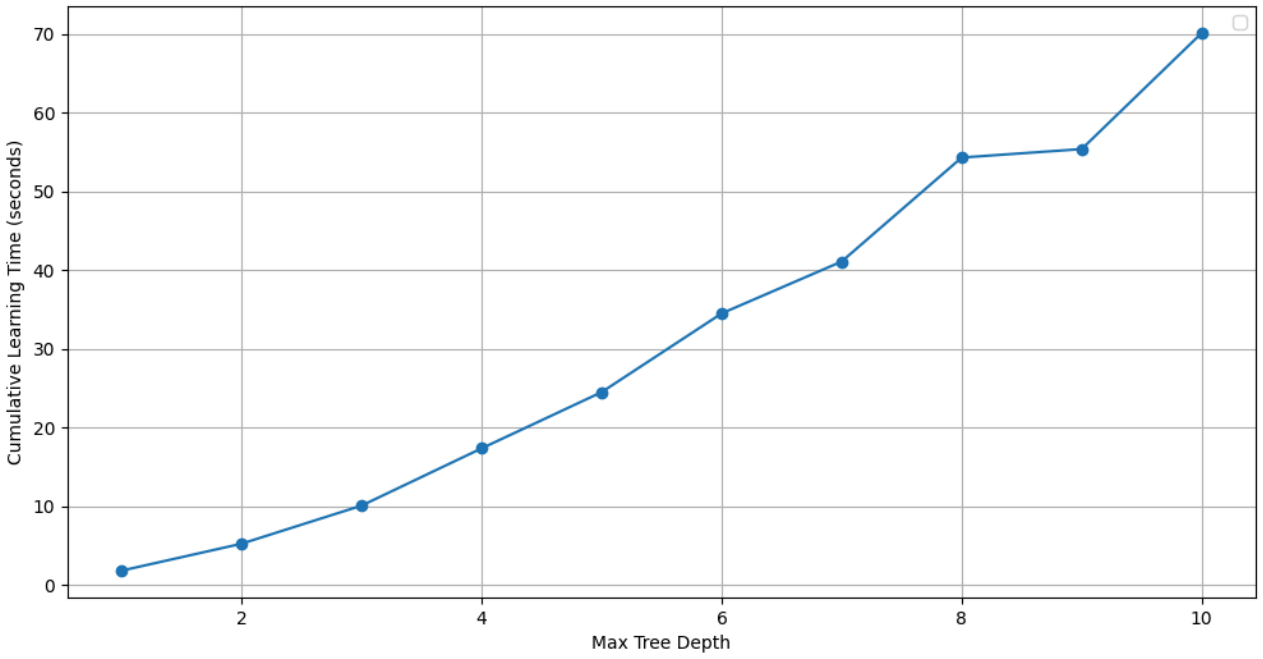

Рисунок 8. График зависимости кумулятивного времени обучения модели от заданной максимальной глубины дерева решений

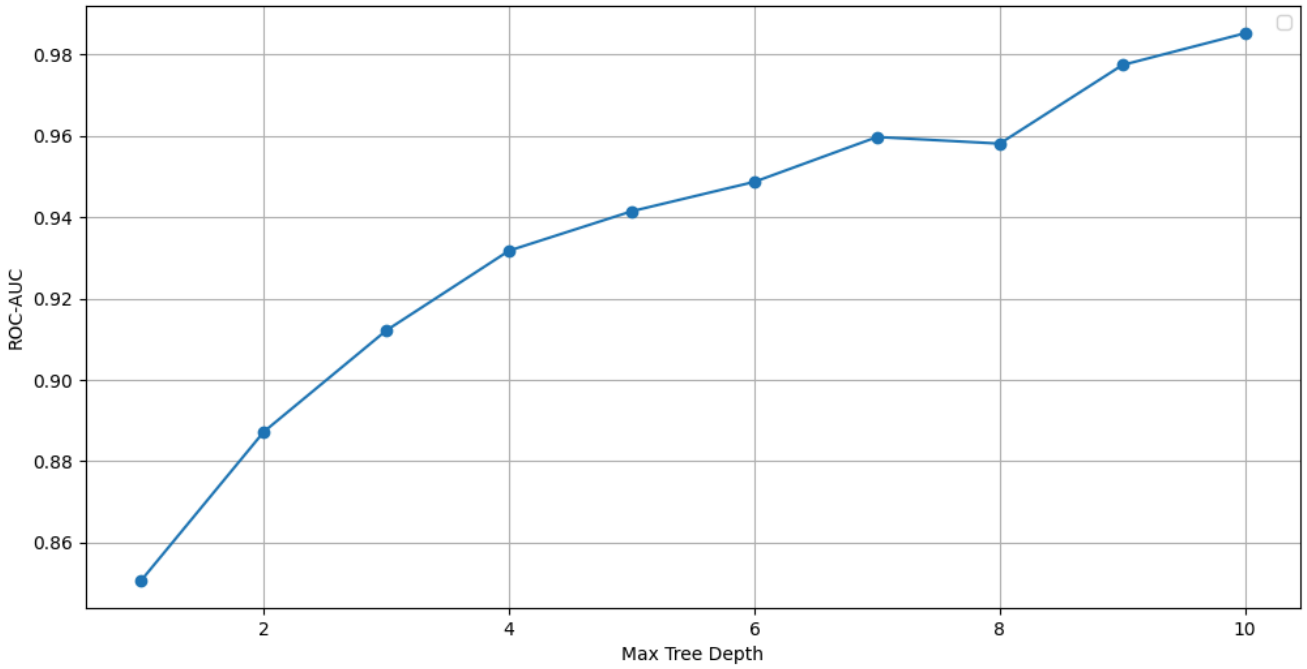

Рисунок 9. График зависимости показателя ROC-AUC от заданной максимальной глубины дерева решений

3.3 Случайный лес (Random Forest)

Случайный лес — это ансамблевый метод машинного обучения, который объединяет множество деревьев решений для выполнения задач классификации или регрессии. Основная идея заключается в том, чтобы снизить переобучение и повысить общую устойчивость модели за счет усреднения результатов независимых деревьев решений. Это достигается с помощью случайного отбора данных и признаков.

3.3.1 Основной принцип работы

Random Forest строит n деревьев решений, каждое из которых обучается на случайной выборке данных, полученной методом сэмплирования с возвратом (bootstrap). Во время построения каждого дерева также выбирается случайное подмножество признаков для поиска лучшего разбиения. Это снижает вероятность переобучения, так как деревья становятся менее зависимыми от отдельных признаков или шумов в данных.

На этапе предсказания случайный лес комбинирует результаты деревьев следующим образом:

- Для задачи классификации: **голосование большинства**. Класс, предсказанный большинством деревьев, считается итоговым предсказанием.

- Для задачи регрессии: **усреднение предсказаний** всех деревьев.

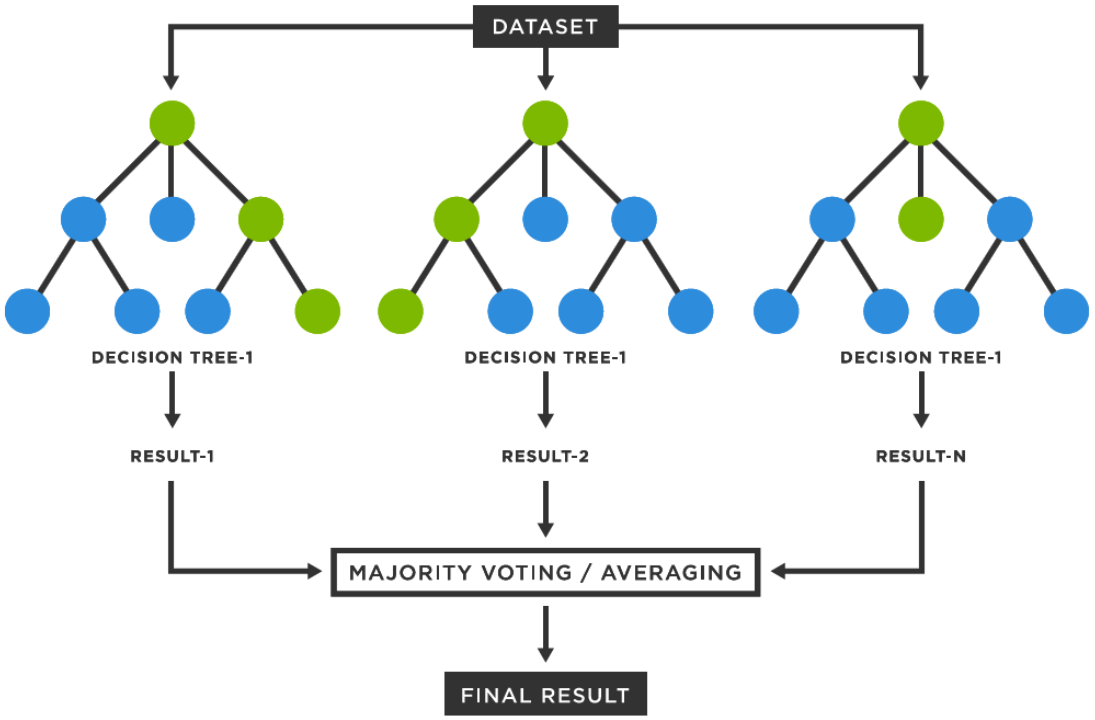

Рисунок 9. Структура модели Random Forest

3.3.2 Предобработка данных

Перед обучением модели Random Forest выполнены следующие действия:

- **Балансировка классов:** использован метод oversampling, чтобы устранить дисбаланс между классами и сделать задачу обучения деревьев более сбалансированной.

- **Масштабирование признаков:** выполнено с целью нормализации данных, что помогает ускорить процесс построения деревьев.

3.3.3 Обучение модели

На этапе обучения каждый экземпляр дерева решает задачу оптимального разбиения данных. Для оценки качества разбиений используется **критерий Джини**, аналогичный методу дерева решений:

$$Gini = 1 - \sum_{k=1}^{K} p_k^2,$$

где p_k — доля объектов класса k в узле, K — общее число классов (в нашем случае $K = 2$).

На каждом уровне дерева:

1. Выбирается случайное подмножество признаков размером $\sqrt{M}$, где M — общее количество признаков.

2. Для каждого выбранного признака тестируются различные пороги разбиения, чтобы минимизировать критерий Джини.

3. Разбиение с минимальным значением критерия Джини используется для разделения данных на две группы.

После построения каждого дерева используется максимальная глубина (*max_depth*) и минимальное количество элементов в узле (*min_samples_split*), чтобы избежать избыточной детализации и переобучения. Процесс повторяется для n-деревьев, указанных пользователем.

3.3.4 Предсказание

На этапе предсказания каждое дерево возвращает свой результат. Для классификации применяется метод голосования большинства:

$$\widehat{y} = mode(y_1, y_2, \dots, y_n),$$

где $y_1, y_2, \dots, y_n$ — предсказания каждого дерева. Для получения вероятности принадлежности классу используется усреднение:

$$P(class = 1) = \frac{1}{n}\sum_{i=1}^{n} P_i(class = 1),$$

где $P_i(class = 1)$ — вероятность, вычисленная отдельным деревом. Итоговое решение принимается на основе порогового значения, которое корректируется в зависимости от попадания “учащейся” модели в изначальный целевой класс (*target*).

3.3.5 Преимущества и недостатки

Преимущества:

- **Стабильность и устойчивость:** имеет низкую чувствительность к шуму и выбросам.
- **Поддержка нелинейных связей:** благодаря множеству деревьев модель способна выявлять сложные паттерны в нелинейной зависимости данных.
- **Встроенная оценка важности признаков:** можно выявить, какие признаки больше всего влияют на итоговое предсказание.
- **Устойчивость к переобучению:** благодаря сэмплированию по подвыборке и случайному отбору признаков деревья становятся менее зависимыми от изначальных паттернов в данных.

Недостатки:

- **Медлительность:** обучение и предсказание могут быть медленными для большого количества деревьев с большой глубиной.
- **Потребление памяти:** множество деревьев требует значительных вычислительных ресурсов.

3.3.6 Оценка качества модели

После обучения модели на сбалансированном наборе данных были вычислены основные метрики качества:

- **Accuracy (точность):** 0.9535.
- **Precision (прецизионность):** 0.9884.
- **Recall (полнота):** 0.9181.
- **F1 Score:** 0.9519 — баланс между точностью и полнотой.
- **ROC-AUC:** 0.9536 — хорошее разделение классов независимо от порогового значения.

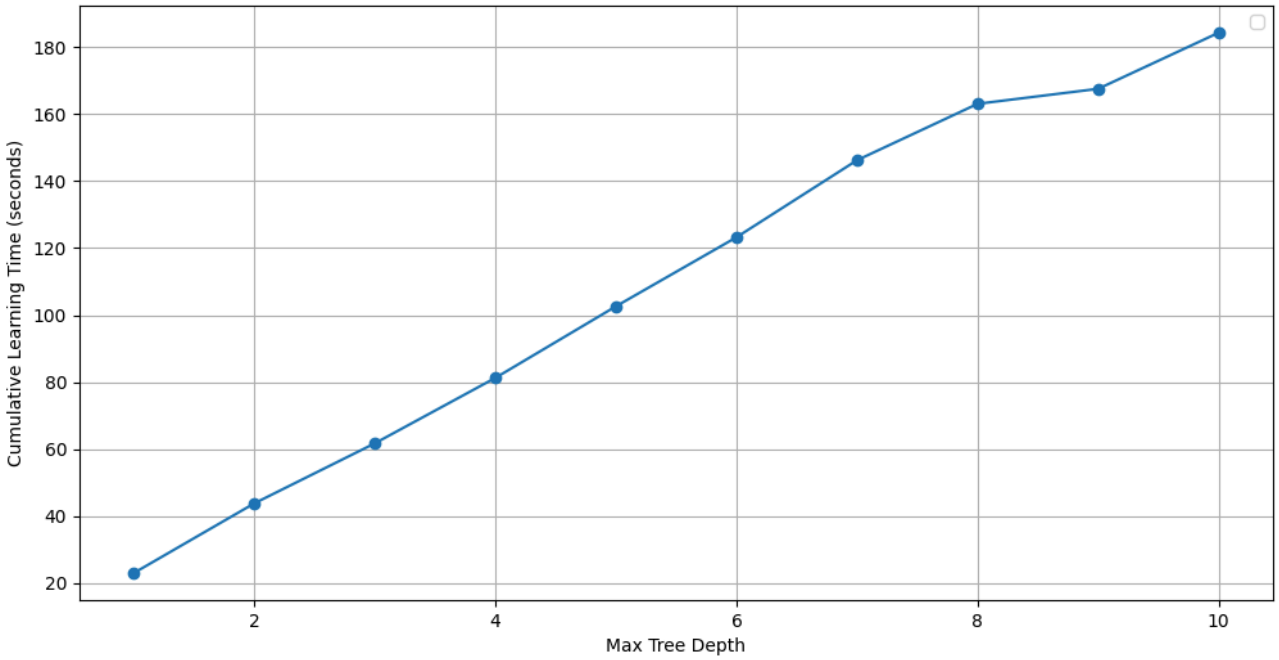

Рисунок 10. График зависимости времени обучения модели от заданной максимальной глубины деревьев решений в Random Forest

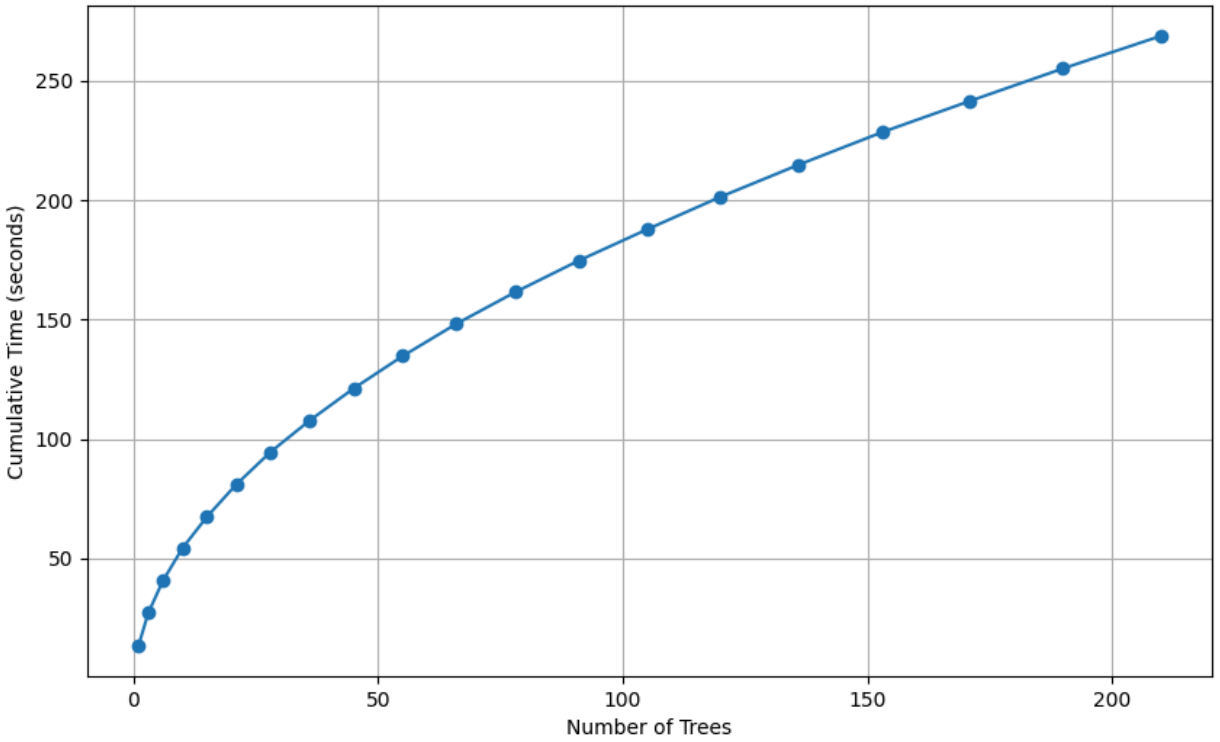

Рисунок 11. График зависимости кумулятивного времени обучения модели от количества деревьев решений в Random Forest

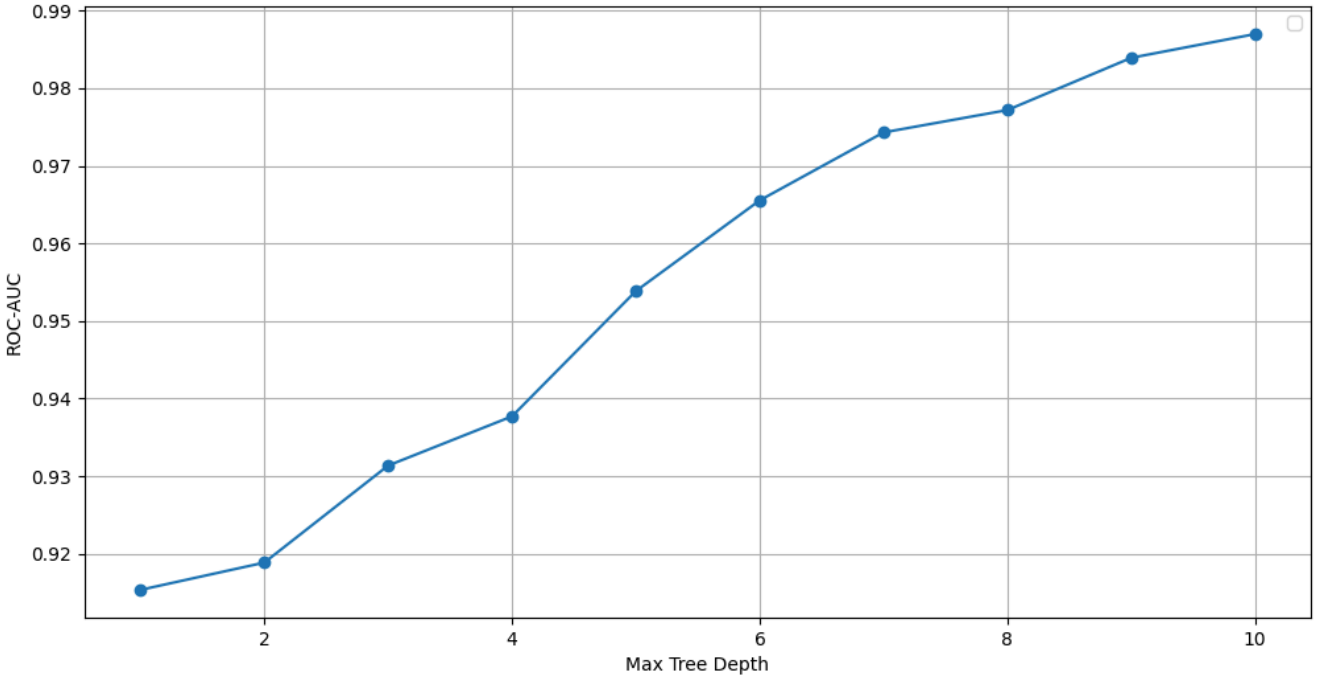

Рисунок 12. График зависимости показателя ROC-AUC от заданной максимальной глубины деревьев решений в Random Forest

3.4 Градиентный бустинг (Gradient Boosting)

Градиентный бустинг — это еще одинансамблевый метод машинного обучения, который объединяет множество деревьев решений для решения задач классификации и регрессии. В отличие от *Random Forest*, где деревья обучаются независимо, градиентный бустинг строит деревья последовательно, каждое из которых пытается скорректировать ошибки предыдущих. Это делает его особенно эффективным для задач, где требуется высокая точность.

3.4.1 Принцип работы

Градиентный бустинг строится на основе следующих ключевых идей:

1. **Ансамбли деревьев**: Модель состоит из последовательности деревьев решений. Каждое дерево обучается на остатках (ошибках) предсказаний предыдущего ансамбля деревьев.
2. **Оптимизация функции потерь**: На каждом шаге обучения добавляется новое дерево, которое минимизирует заданную функцию потерь, в данном случае — среднеквадратическую ошибку (*MSE*).
3. **Шаг обучения**: Для управления вкладом каждого нового дерева используется параметр *learning_rate*, что позволяющий избежать переобучения.

Для ускорения и повышения устойчивости модели используется сэмплирование данных (*subsample*): каждое дерево обучается не на полном наборе данных, а на случайной выборке, что также снижает риск переобучения.

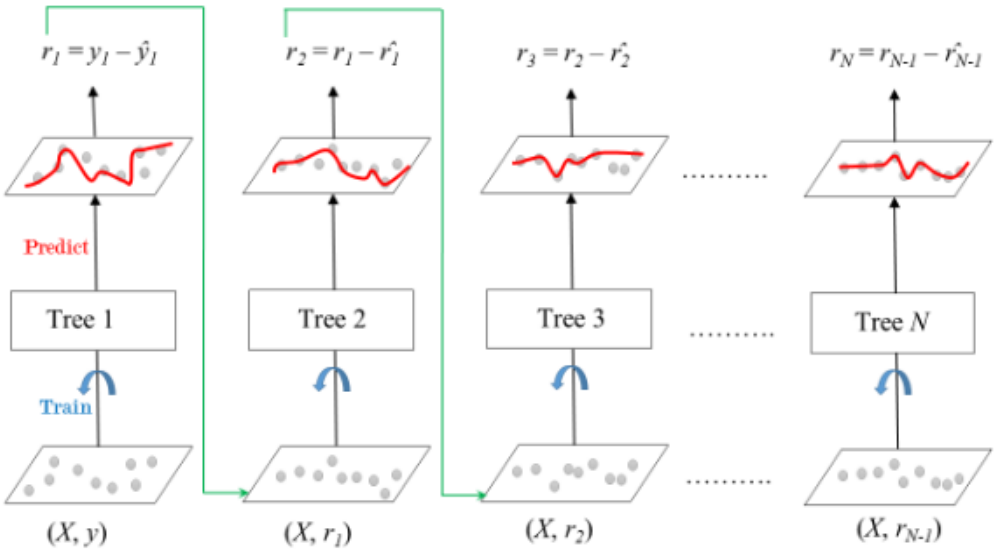

Рисунок 13. Визуализация работы модели градиентного бустинга

3.4.2 Предобработка данных

Как и в случае с моделью случайного леса, перед обучением градиентного бустинга были выполнены следующие шаги:

1. **Балансировка классов**: Дисбаланс классов был устранен методом *oversampling* (дублирование записей класса мошенничества). Это сделано для того, чтобы модель могла

обучиться эффективному распознаванию редких мошеннических транзакций.

2. **Масштабирование признаков**: Все признаки были нормализованы с использованием StandardScaler. Это важно для стабильного разделения данных в узлах деревьев.

3.4.3 Обучение модели

На этапе обучения модель градиентного бустинга выполняет следующие действия:

1. **Инициализация предсказания**: Все данные изначально предсказываются как среднее значение целевой переменной (для регрессии) или начальная вероятность классов.
2. **Обучение деревьев**:
   - На каждом шаге обучается дерево решений, которое минимизирует остаточную ошибку предыдущего ансамбля.
   - Для выбора разбиения в узлах дерева оценивается среднеквадратическая ошибка. Порог разбиения подбирается так, чтобы минимизировать ее.
3. **Обновление остатков**: После построения очередного дерева предсказания обновляются с учетом текущих ошибок, умноженных на темп обучения (*learning_rate*).
4. **Регуляризация**: Используются параметры ограничения глубины дерева (*max_depth*), минимального числа элементов в узле (*min_samples_split*) и сэмплирования подвыборки (*subsample*) для предотвращения переобучения.

3.4.4 Предсказание

На этапе предсказания модель суммирует взвешенные предсказания всех деревьев в ансамбле:

$$\widehat{y} = \sum_{i=1}^{n} \eta \cdot T_i(x),$$

где n — количество деревьев, η — темп обучения (*learning_rate*), $T_i(x)$ — предсказание i-го дерева для объекта x. Полученная сумма преобразуется в вероятность принадлежности классу, а затем сравнивается с пороговым значением для получения окончательного класса.

3.4.5 Преимущества и недостатки

Преимущества:

1. **Высокая точность**: хорошо справляется с задачами классификации для данных с дисбалансом классов.
2. **Гибкость**: поддерживает различные функции потерь, универсальна для различных задач.
3. **Выявление сложных паттернов**: благодаря последовательному обучению деревьев эффективно находит сложные взаимосвязи в нелинейных данных.

Недостатки:

1. **Медленная скорость обучения**: по аналогии с моделью *Random Forest*.

2. **Чувствительность к гиперпараметрам**: неправильный выбор *learning_rate* или *max_depth* может привести к переобучению модели.
3. **Высокая сложность вычислений**.

3.4.6 Оценка качества модели

После обучения модели на сбалансированном наборе данных были вычислены основные метрики качества:

- **Accuracy (точность)**: 0.9971.
- **Precision (прецизионность)**: 0.9984.
- **Recall (полнота)**: 0.9957.
- **F1 Score**: 0.9971.
- **ROC-AUC**: 0.997 — еще лучшая способность модели разделять классы независимо от порогового значения.

Результаты показывают, что градиентный бустинг превосходит *Random Forest*.

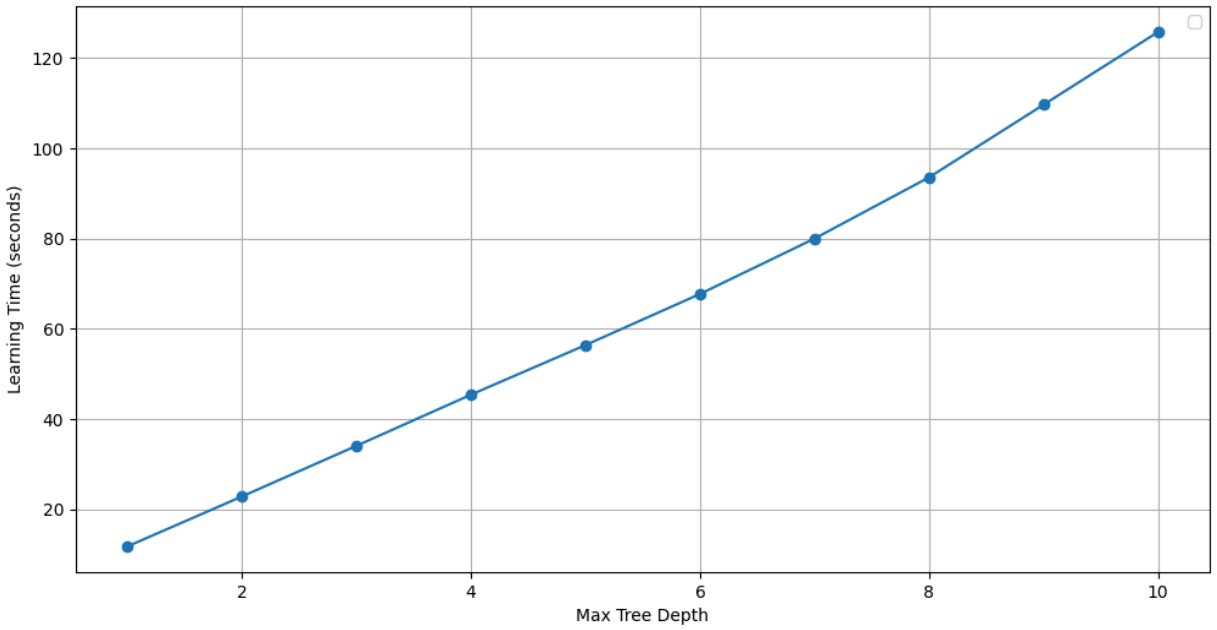

Рисунок 14. График зависимости времени обучения модели от максимальной глубины деревьев решений

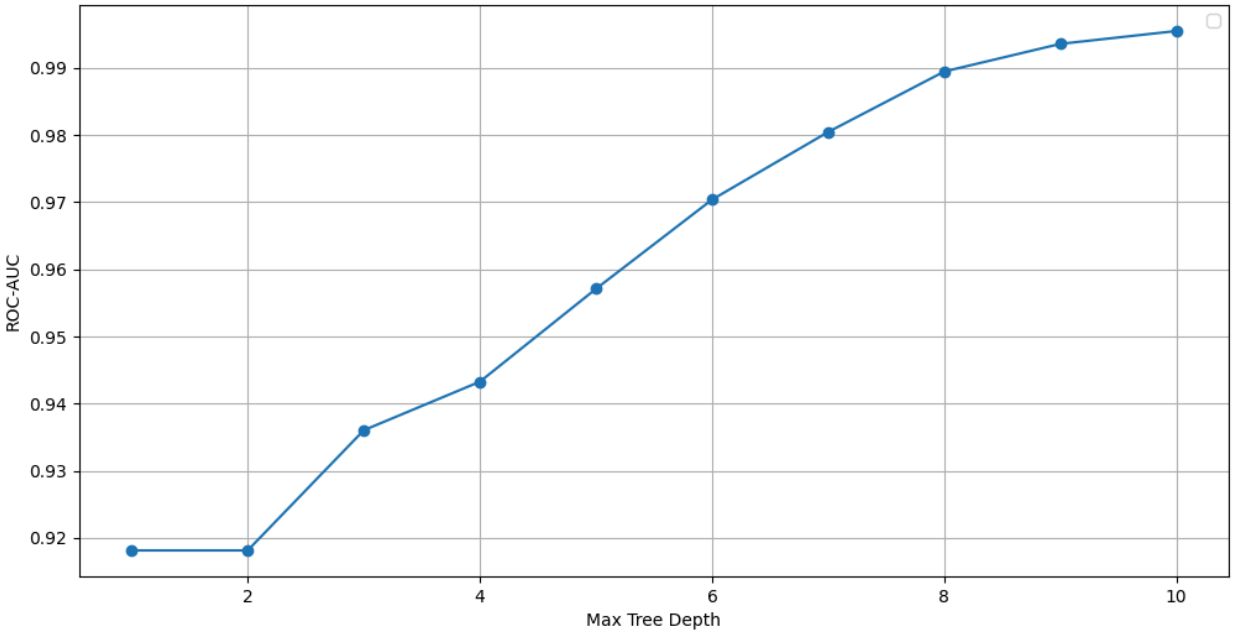

Рисунок 15. График зависимости показателя ROC-AUC от заданной максимальной глубины деревьев решений

3.5 Многослойный перцептрон

Многослойный перцептрон (*Multi-Layer Perceptron, MLP*) — это вид нейронной сети, который подходит для задач классификации, регрессии и других видов предсказания. MLP состоит из нескольких слоев нейронов: входного, одного или более скрытых слоев, и выходного. Каждый нейрон связан с нейронами предыдущего слоя, выполняет линейное преобразование входных данных и передает результат через нелинейную функцию активации. Благодаря своей архитектуре способен моделировать сложные зависимости в данных, включая нелинейные.

Преимущества:

- **Гибкость:** может моделировать сложные нелинейные зависимости между признаками, что делает его пригодным для задач с высокой сложностью.
- **Высокая точность:** при наличии достаточного количества данных и правильной настройки гиперпараметров может превосходить в точности все предыдущие модели.
- **Автоматизация выделения признаков:** не требует ручного выбора и преобразования признаков — модель самостоятельно извлекает их из данных.

Недостатки:

- **Чувствительность к гиперпараметрам:** эффективность зависит от таких настроек, как количество слоев, количество нейронов в каждом слое, скорость обучения и размер батча.
- **Требовательность к данным и вычислительным ресурсам**.
- **Сложность интерпретации**.

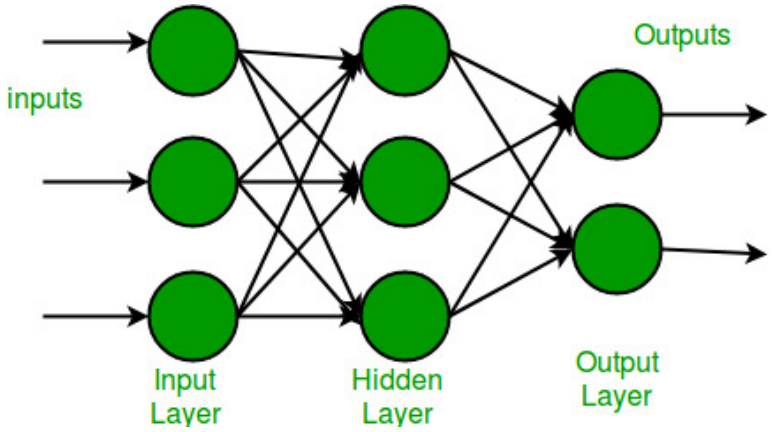

Рисунок 16. Структура многослойного перцептрона

3.5.1 Предобработка данных

Для подготовки данных была использована следующая последовательность шагов:

1. **Балансировка классов:** В исходном наборе данных была значительная диспропорция между классами (мошеннические и нормальные транзакции). Для устранения этого дисбаланса использовался метод SMOTE (*Synthetic Minority Oversampling Technique*), который генерирует синтетические данные для меньшинств.
2. **Масштабирование:** Все числовые признаки были нормализованы с использованием стандартного масштаба (*StandardScaler*), чтобы привести их к схожему диапазону. Это особенно важно для нейронных сетей, так как большие значения могут замедлить обучение и ухудшить качество модели.

3.5.2 Архитектура модели

Многослойный перцептрон был реализован с использованием библиотеки PyTorch. Архитектура модели включает:

- **Входной слой:** Количество нейронов соответствует числу признаков (30 после предобработки).
- **Три скрытых слоя:**
  - Первый скрытый слой содержит 128 нейронов с функцией активации ReLU и регуляризацией Dropout (0.3) для предотвращения переобучения.
  - Второй скрытый слой содержит 64 нейрона с аналогичными настройками.
  - Третий скрытый слой содержит 32 нейрона с функцией активации ReLU.
- **Выходной слой:** Один нейрон с сигмоидальной активацией, возвращающий вероятность принадлежности к классу мошеннических транзакций.

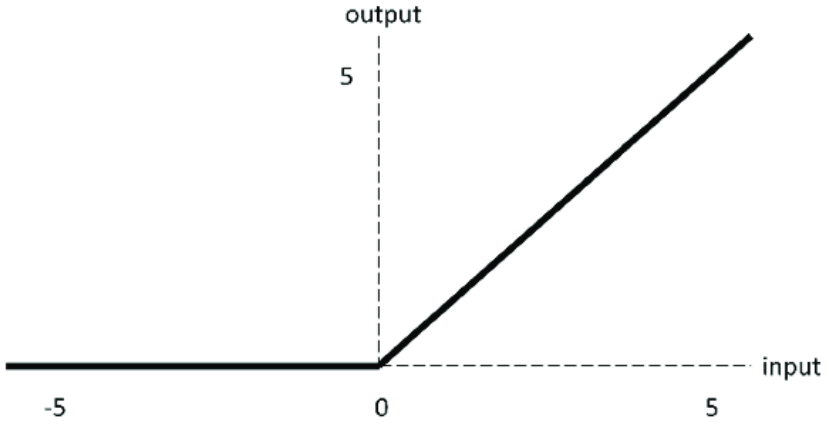

Рисунок 17. График использованной функции активации (ReLU – *Rectified Linear Unit*)

3.5.3 Обучение модели

Обучение модели проводилось в течение 100 эпох с использованием функции потерь Binary Cross-Entropy (*BCELoss*) и оптимизатора (*Adam*).

Оптимизатор Adam (*Adaptive Moment Estimation*) сочетает в себе преимущества двух других методов оптимизации: AdaGrad (адаптивная скорость обучения) и RMSProp (использование среднего квадратичного значения градиентов): он хорошо работает на задачах, где параметры сильно изменяются на разных этапах обучения, и часто превосходит другие оптимизаторы в производительности. Суть заключается в том, что он адаптивно подстраивает *learning_rate* для каждого параметра, используя оценки первых моментов (среднего значения) и вторых моментов (дисперсии) градиентов.

На протяжении обучения модели (по эпохам) фиксировались следующие значения:

- Среднее значение функции потерь (Loss) для каждой эпохи.
- Метрика ROC-AUC на валидационном наборе данных.
- Время обучения каждой эпохи, которое суммировалось для оценки общей продолжительности.

3.5.4 Оценка качества модели

По завершении обучения модель была протестирована на отложенной тестовой выборке. Результаты тестирования:

- Accuracy: **0.9996**
- Precision: **0.9992**
- Recall: **1.0000**
- F1 Score: **0.9996**
- ROC-AUC: **0.9998**

Модель продемонстрировала лучшие результаты по всем метрикам среди всех моделей.

3.5.5 Построение графиков

В ходе обучения были построены графики, позволяющие отследить прогресс:

1. **График ошибки (Loss):** уменьшение значения функции потерь с каждой новой эпохой.
2. **График ROC-AUC:** улучшение способности классификации с каждой новой эпохой.
3. **График кумулятивного времени:** общее время, затраченное на обучение модели.

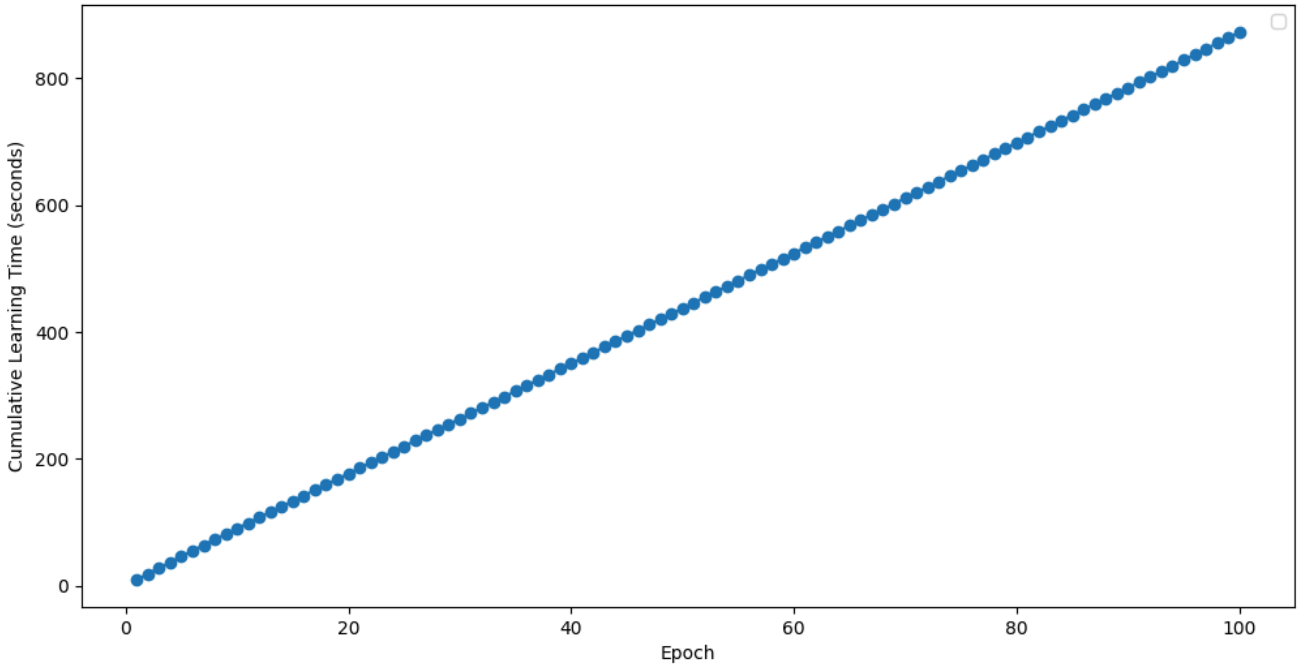

Рисунок 18. График зависимости кумулятивного времени обучения модели от пройденной эпохи обучения

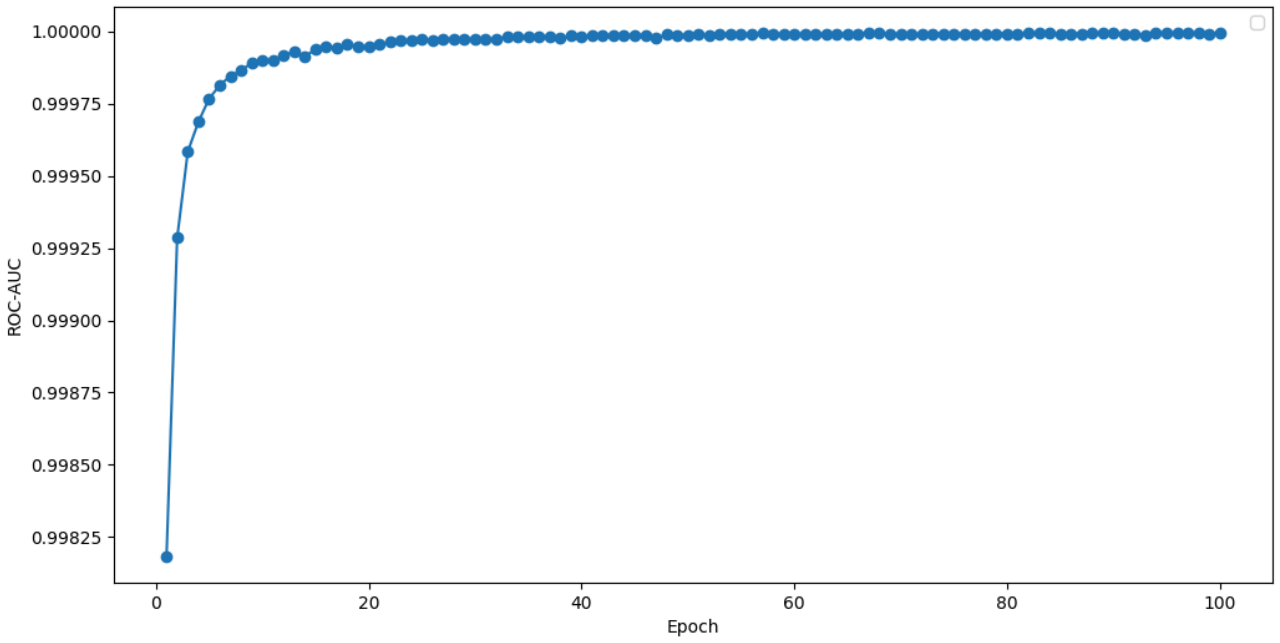

Рисунок 19. График зависимости показателя ROC-AUC от пройденной эпохи обучения

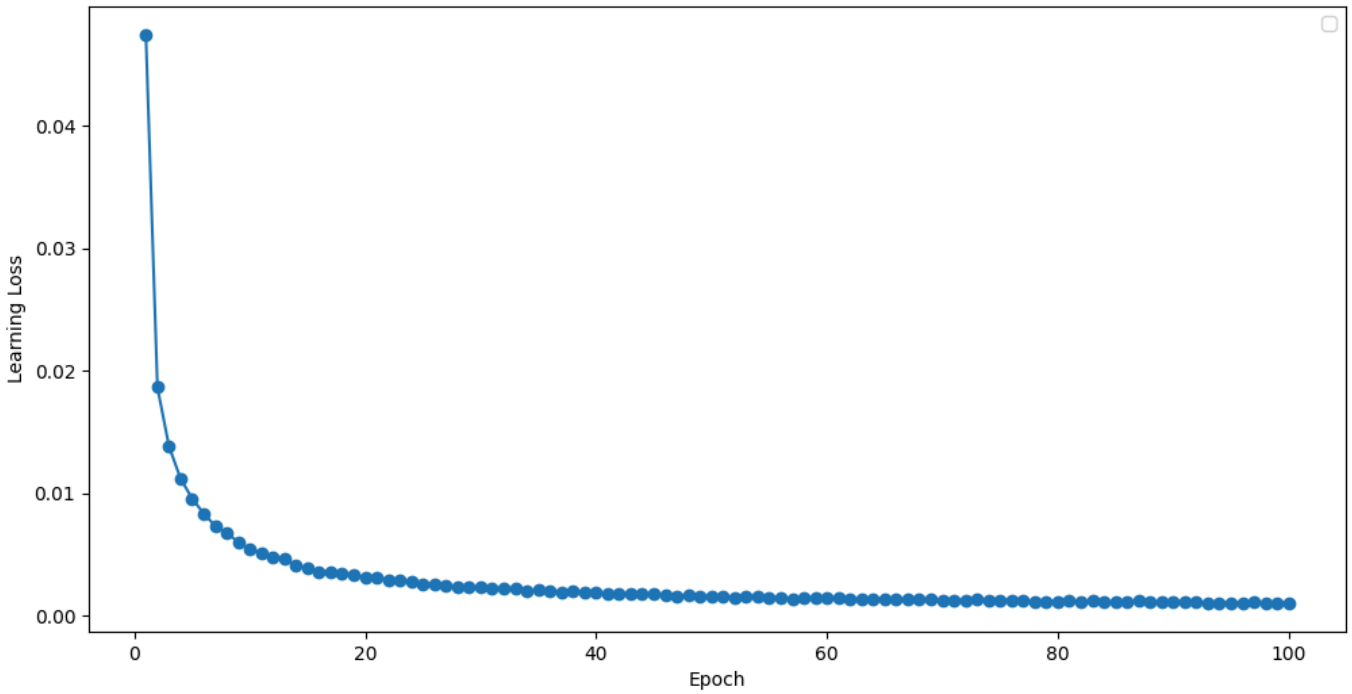

Рисунок 20. График зависимости потерь (loss) от пройденной эпохи обучения

3.5.6 Выводы

Многослойный перцептрон продемонстрировал выдающиеся результаты в задаче выявления мошеннических операций. Использование *dropout*-слоев позволило уменьшить вероятность переобучения, а балансировка классов и масштабирование данных обеспечили корректное обучение. Данная модель подходит для более сложных задач и хорошо справляется с идентификацией мошенничества даже в условиях сильного дисбаланса классов.

4. Сравнение полученных результатов

Основным критерием для оценки моделей в задаче был выбран показатель ROC-AUC (площадь под кривой ROC). ROC-AUC позволяет оценить качество работы модели независимо от выбора порога классификации, что особенно важно для несбалансированных данных, где ошибки пропуска (ложноотрицательные результаты) могут быть критичными.

4.1. Преимущества и недостатки каждой модели

4.1.1. Логистическая регрессия

Показала низкие значения Recall, что означает, что модель пропускала значительное количество мошеннических транзакций. Однако ее ROC-AUC находился на приемлемом уровне (рис. 5), указывая на способность модели отделять классы, но уступая моделям более высокой сложности. Простота логистической регрессии позволяет быстро обучить модель, но ее линейный характер ограничивает способность захватывать сложные зависимости в данных.

Итог: быстрая интерпретация, высокая скорость обучения и предсказания, устойчива к переобучению. Плохо работает на сложных нелинейных данных, ограниченные возможности для повышения качества. Подходит для первичного анализа данных и задач с линейными зависимостями.

4.1.2. Дерево решений

Имело высокий Recall, но низкие значения Precision и F1 Score, что говорит о большом количестве ложных срабатываний. При этом качество предсказаний сильно варьировалось в зависимости от параметров глубины дерева (рис. 9). Достигнута высокая интерпретируемость, но ROC-AUC ниже, чем у ансамблевых методов. Это связано с тенденцией модели к переобучению на тренировочных данных. Сложность обучения растет экспоненциально при увеличении максимальной глубины дерева.

Итог: простота интерпретации, возможность выявления закономерностей в данных. Склонность к переобучению (особенно для глубоко построенных деревьев). Используется как базовая модель, но в ансамблях (например, *Random Forest*) достигает лучшего качества.

4.1.3. Random Forest

Продемонстрировал высокий ROC-AUC (близкий к 0.99) и лучшее обобщение, чем одиночное дерево решений (рис. 12). Однако модель была медленнее при обучении и предсказании из-за ансамблевого подхода (рис. 11). За счет усреднения предсказаний деревьев, модель менее склонна к переобучению.

Итог: высокая устойчивость к шуму, хорошая способность к обобщению. Медленное обучение на больших данных, сложность интерпретации результатов. Подходит для задач, где важны стабильные предсказания.

4.1.4. Градиентный бустинг

Стал одной из лучших моделей с точки зрения ROC-AUC и F1 Score. Он продемонстрировал стабильное обучение и высокую точность при оптимальных гиперпараметрах (рис. 15). Модель превосходно справляется с выделением закономерностей в данных, но обучение занимает гораздо больше времени по сравнению даже с *Random Forest*.

Итог: высокая точность, гибкость в настройке, устойчивость к выбросам и шуму. Длительное обучение, чувствительность к гиперпараметрам. Подходит для задач, требующих максимального качества предсказаний

4.1.5. Многослойный перцептрон (MLP)

Показал высокие значения ROC-AUC и F1 Score, но в процессе обучения наблюдались колебания качества (особенно на первых эпохах) (рис. 19). При этом скорость обучения нейросети была выше, чем у градиентного бустинга.

Итог: способен находить сложные нелинейные зависимости, адаптируется к большим объемам данных. Требует значительных вычислительных ресурсов, менее интерпретируем. Подходит для задач с большим количеством параметров и сложной структурой данных.

4.2. Интерпретируемость моделей

В задаче интерпретируемость имеет важное значение для объяснения результатов и принятия решений. Для моделей логистической регрессии и дерева решений интерпретируемость доступна "из коробки" – можно увидеть пороговые значения для функции логистической регрессии или в “ветвях” дерева решений. Вне зависимости от модели, есть методы интерпретации, подходящие и для сложных моделей (таких как Random Forest, градиентный бустинг и MLP). Таковыми являются:

- **SHAP (SHapley Additive Explanations)**: позволяет обобщенно объяснять вклад каждого признака в предсказания модели.

- **LIME (Local Interpretable Model-agnostic Explanations)**: помогает объяснять работу сложных моделей на определенных наборах данных, выявляет локальные зависимости.

Для примера был построен график **SHAP** для модели “дерево решений” (рис. 21), который отображает вклад признаков в предсказания модели для класса 1 (мошенничество). Опишем его:

1. **По вертикали** перечислены признаки, отсортированные по степени их влияния на предсказания (сверху вниз — наиболее важные).
2. **По горизонтали** – шкала значения SHAP: показывает вклад данного признака в предсказания модели. Если значение положительное, это увеличивает вероятность положительного класса (мошенничество). Если отрицательное — уменьшает.
3. **Синий цвет:** Низкое значение признака.
4. **Красный цвет:** Высокое значение признака.

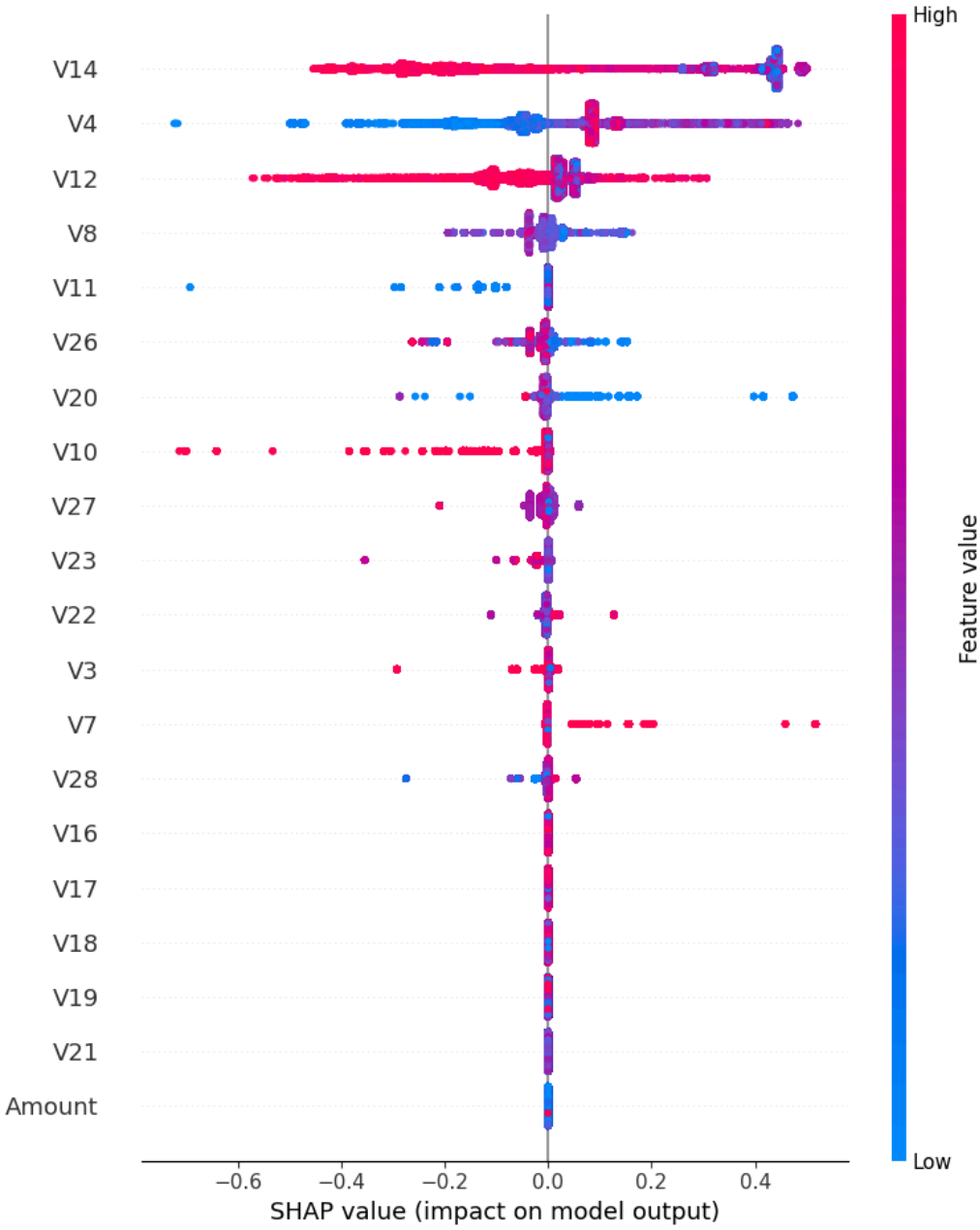

Рисунок 21. График вклада признаков в предсказание полученным деревом решений класса операции (мошенническая / не мошенническая)

Прочитаем график:

1. **Распределение SHAP-значений для каждого признака:**
   - Каждая точка соответствует одному наблюдению в тестовых данных.
   - Расположение точки по оси X указывает на то, как сильно данный признак повлиял на предсказание.
2. **Цвет точек:**
   - Если точки красные, это означает, что высокая величина данного признака больше склоняет модель к положительному классу (мошенничеству).
   - Синие точки показывают, что низкие значения признака склоняют модель к одному из классов (в данном случае к отрицательному классу).

Примеры интерпретации:

1. **V14 (признак):** имеет значительное влияние на предсказания. Высокие значения (красные точки) увеличивают вероятность мошенничества.
2. **V4 (признак):** низкие значения (синие точки) значительно увеличивают вероятность положительного класса.
3. **V12 и V8:** имеют важное влияние, но их воздействие может быть как положительным, так и отрицательным в зависимости от значений.
4. **"Длинные" SHAP-значения:** если точки сильно разбросаны по оси X, это означает, что данный признак сильно влияет на результат модели. Например, **V14** имеет большой разброс SHAP-значений, что указывает на его высокую важность для предсказаний.

Выводы:

1. Признаки **V14, V4, V12, V8** — наиболее значимые для предсказания мошенничества.
2. Признак **V14** особенно важен: его высокие значения значительно увеличивают вероятность мошенничества.
3. Признаки **V26, V20, V10** также оказывают умеренное влияние на предсказания модели.
4. **Влияние признаков несимметрично:** некоторые признаки оказывают как положительное, так и отрицательное влияние на предсказания в зависимости от их значения.

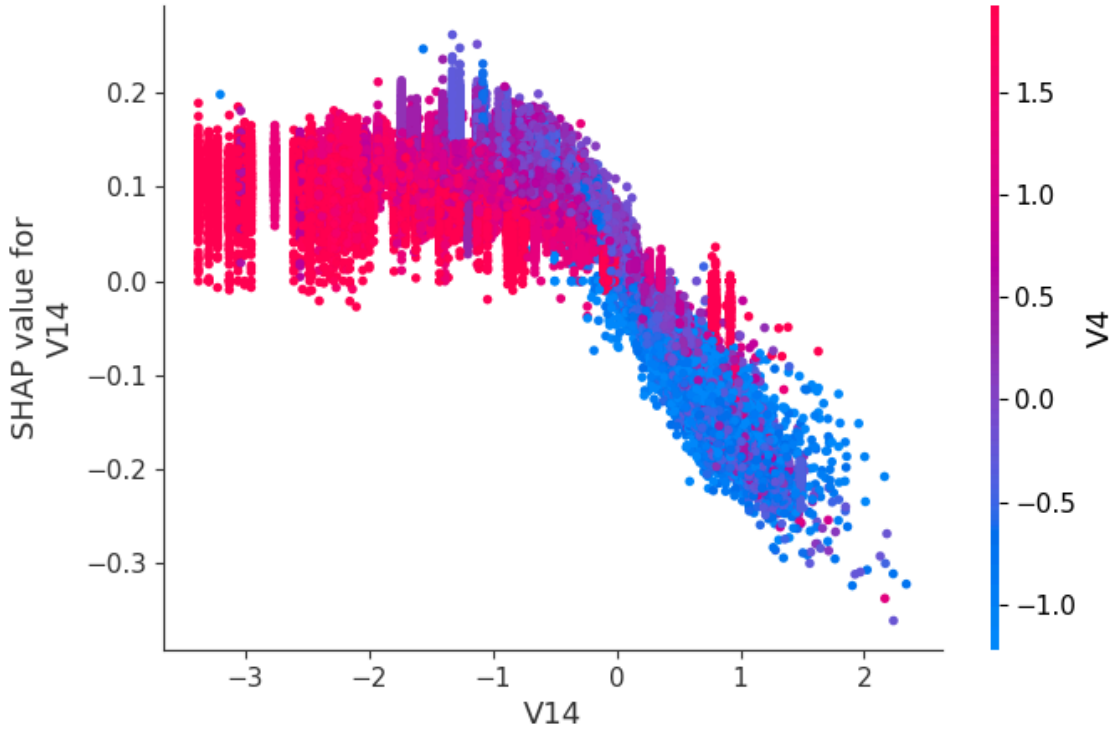

Рисунок 22. График вклада признака “V14” во все предсказания модели логистической регрессии с наложением признака “V4”

Продолжаем глубокий анализ интерпретируемости признаков на примере модели логистической регрессии: представлен график зависимости SHAP-значений одного из признаков

(**V14**) от его значения (рис. 22) с наложением дополнительного признака (**V4**). Вот как его можно интерпретировать:

1. **Ось X** показывает конкретные значения признака V14 из входных данных (тестовой выборки).
2. **Ось Y**: SHAP-значение, которое показывает вклад признака V14 в предсказание модели для каждого наблюдения. Если SHAP-значение положительное, то признак V14 увеличивает вероятность предсказания класса 1 (мошенничества). Если SHAP-значение отрицательное, то признак V14 снижает вероятность класса 1.
3. **Шкала справа**: дополнительная информация, предоставляемая через цветовую кодировку. В данном случае представляет значения другого признака (V4). Это позволяет проанализировать совместное влияние признаков V14 и V4 на предсказания.

Какие результаты были получены из графика:

1. **Форма зависимости**: видно, что при увеличении значения V14 его SHAP-значения постепенно снижаются. Это говорит о том, что высокие значения V14 уменьшают вероятность предсказания класса 1 (снижают вероятность мошенничества в операции).
2. **Цветовая градация**: существует корреляция между V4 и V14, которая влияет на предсказания. Например, если V4 имеет низкие значения (синий цвет), SHAP-значения V14 меньше, что еще сильнее снижает вероятность класса 1, при высоких же значениях V4 (красный цвет) влияние V14 становится более умеренным.

Признаки, полученные для ансамблей (Random Forest или градиентный бустинг), часто совпадают с признаками, которые можно выделить из простого дерева решений, поскольку ансамбли используют совокупность деревьев для улучшения стабильности и точности. В связи с близкими полученными результатами и их высокой точностью перейдем сразу к модели многослойного перцептрона (MLP).

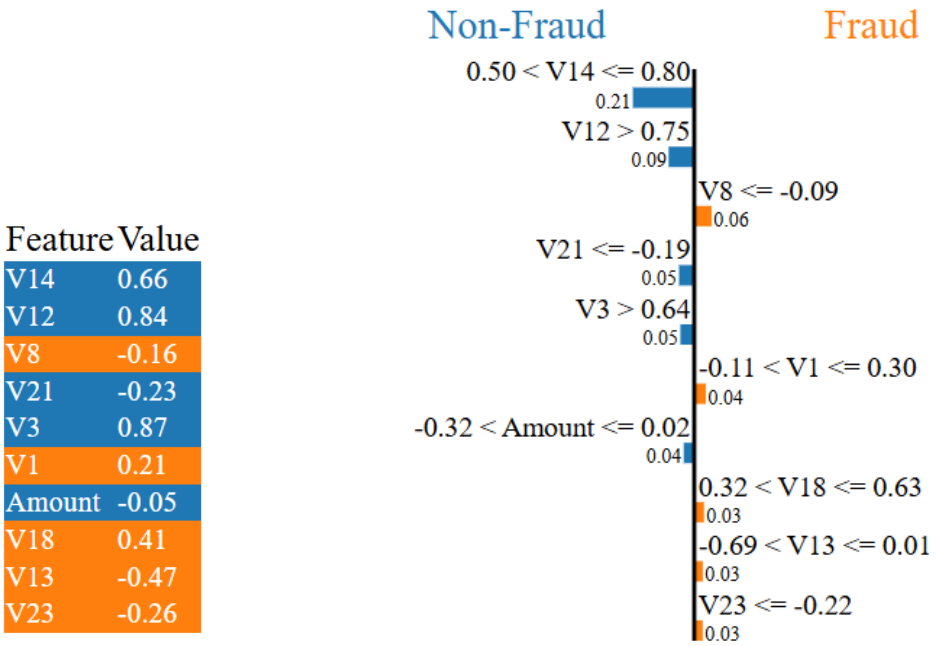

Рисунок 23. Результаты метода LIME – вклад признаков в одно конкретное предсказание (раскрутка всей цепочки весов в перцептроне)

На представленных визуализациях (рис. 23) показаны ключевые признаки, которые оказали наибольшее влияние на предсказание для конкретного примера из тестовой выборки. На графике слева видны значения признаков, упорядоченные по важности их вклада. Признаки с положительными значениями (оранжевые ячейки) способствуют предсказанию класса "Fraud" (мошенничество), тогда как признаки с отрицательными значениями (синие ячейки) влияют на предсказание класса "Non-Fraud" (не мошенничество). Например, признаки **V14** и **V12** значительно способствуют отнесению к "Non-Fraud", тогда как **V8** и **V18** делают вклад в пользу "Fraud".

На графике справа представлены интервальные значения признаков для двух классов. Каждый признак разбит на диапазоны, и указана его степень влияния на предсказание. Например, для класса "Fraud" видно, что **V8 ≤ -0.09** имеет положительный вклад в предсказание мошенничества, а для класса "Non-Fraud" интервал **0.50 < V14 ≤ 0.80** оказывает противоположное влияние. Это позволяет детально рассмотреть, как модель интерпретирует признаки в контексте данного предсказания.

Таким образом, LIME предоставляет интуитивно понятное объяснение, которое показывает, какие признаки, их значения и диапазоны наиболее сильно влияют на решение модели, что помогает проверить согласованность предсказаний и может быть использовано для принятия решений или поиска ошибок в данных.

Промежуточный итог: интерпретируемость позволяет использовать модели не только для предсказаний, но и для возможного мониторинга работы системы, а также объяснения решений моделей специалистам по безопасности. В реальных условиях на основе SHAP или LIME можно выявить аномальные транзакции для последующего ручного анализа.

4.3. Влияние балансировки данных

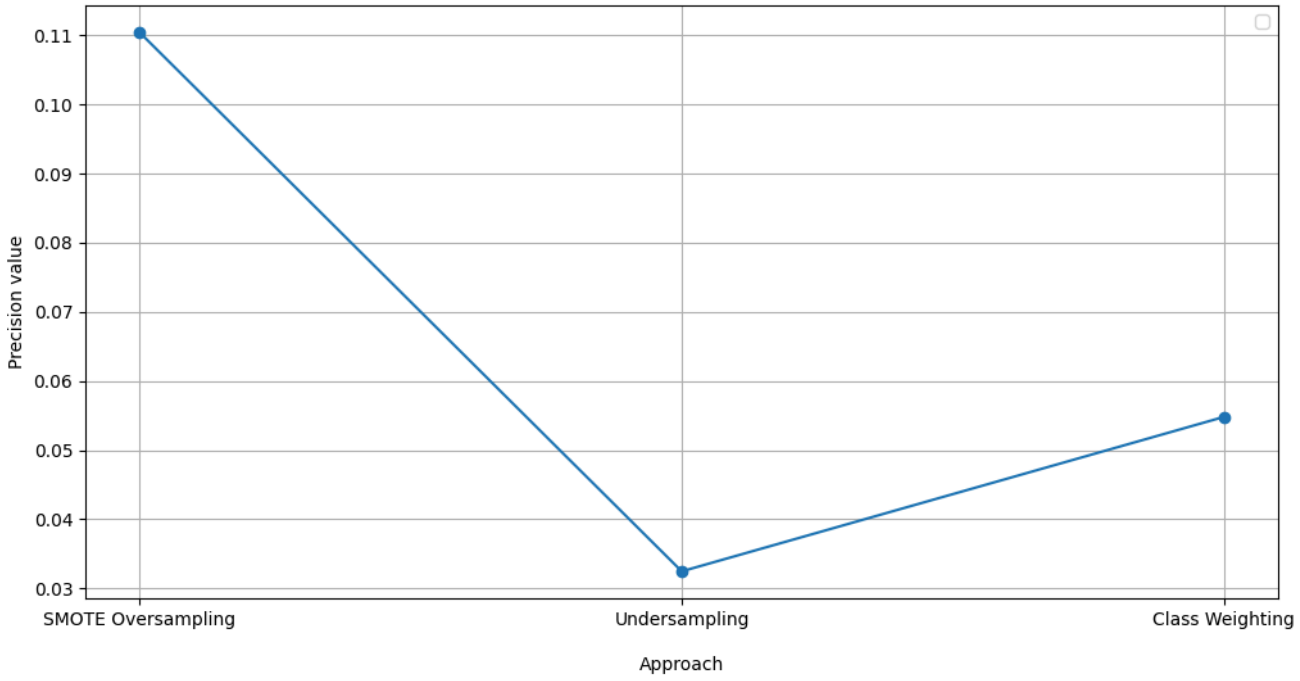

Рисунок 24. График зависимости значения метрики Precision от использованного подхода балансировки данных для модели логистической регрессии

Балансировка классов (*oversampling* через *SMOTE*) сыграла ключевую роль в улучшении ROC-AUC для всех моделей. Без балансировки модели склонны игнорировать миноритарный класс, что критично в задаче обнаружения мошенничества. *Oversampling* улучшил способность моделей выделять паттерны для редкого класса, особенно для деревьев решений, случайного леса и градиентного бустинга. *Undersampling*, напротив, может снижать точность из-за потери информации. *Перевзвешивание классов* также показало эффективность, но уступило SMOTE по результатам Precision (рис. 24).

5. Заключение

Была проведена комплексная работа по выявлению мошеннических транзакций с использованием различных моделей машинного обучения. Работа была сосредоточена на анализе эффективности методов классификации, оптимизации параметров моделей, а также применении техник балансировки данных. Мы стремились создать интерпретируемую, точную и применимую в реальных условиях систему, способную эффективно обнаруживать редкие случаи мошенничества.

5.1. Выводы

На основании анализа метрик ROC-AUC, F1, Precision и Recall, можно заключить, что лучшей моделью для данной задачи оказался **градиентный бустинг**, который показал наибольшую точность в выявлении мошеннических транзакций. Его высокая способность к обобщению позволила точно разделять классы даже в условиях дисбаланса данных.

Применение **балансировки классов через oversampling с использованием SMOTE** значительно повысило производительность моделей. Особенно это было важно для таких моделей, как логистическая регрессия и деревья решений, которые без балансировки демонстрировали слабую чувствительность к редкому классу.

Для реальных условий эта модель может быть полезна в системах мониторинга транзакций, где важно быстро и точно идентифицировать подозрительные операции. Её применение способно существенно снизить убытки от мошенничества, однако необходимо учитывать ограничения производительности, связанные с обработкой большого объема данных в реальном времени.

5.2. Ограничения исследования

Несмотря на высокие результаты, исследование имеет ряд ограничений:

1. **Зависимость от датасета:** Модели обучались на единственном наборе данных о транзакциях. Реальные данные могут отличаться по структуре, размерности или характеристикам, что может повлиять на точность модели.
2. **Синтетическое увеличение данных:** Хотя SMOTE эффективно увеличивает миноритарный класс, он может создавать искусственные точки, которые не всегда отражают реальное распределение данных.

3. **Ограниченность моделей:** Основное внимание было уделено стандартным моделям, таким как логистическая регрессия, случайный лес и градиентный бустинг. Другие подходы, например, нейронные сети, были рассмотрены лишь поверхностно.
4. **Интерпретируемость:** Хотя метод LIME позволил интерпретировать модели, сложные алгоритмы (например, градиентный бустинг) остаются менее интерпретируемыми по сравнению с линейными моделями.

5.3. Рекомендации для дальнейших исследований

В рамках дальнейшего развития системы обнаружения мошенничества рекомендуется:

1. **Тестирование на других датасетах:** Для проверки обобщающей способности моделей следует провести тесты на других наборах данных, отражающих различные аспекты мошеннической активности.
2. **Применение нейронных сетей:** Современные подходы, такие как рекуррентные и сверточные нейронные сети, могут быть полезны для анализа временных данных и выявления скрытых закономерностей.
3. **Использование ансамблей моделей:** Комбинирование различных алгоритмов, например, через стекинг или блендинг, может улучшить качество классификации.
4. **Обработка временных зависимостей:** Добавление временных признаков и анализ последовательностей транзакций, например, с помощью временных моделей (LSTM, Transformer), может повысить чувствительность к мошенничеству.
5. **Учет реальных ограничений:** Разработка методов, которые учитывают ограничения по вычислительным ресурсам, скорость принятия решений и сложность внедрения модели в промышленную инфраструктуру.
6. **Интеграция с бизнес-процессами:** Помимо алгоритмов, важно продумать способы интеграции моделей в реальные системы, чтобы минимизировать количество ложных срабатываний и эффективно управлять флагами подозрительных транзакций.

Проведенное исследование продемонстрировало, что точное обнаружение мошенничества возможно при использовании передовых методов классификации и корректной балансировки данных. Полученные результаты могут быть использованы в реальных системах мониторинга транзакций, однако важно учитывать ограничения модели и нацеленность на реальные условия применения. В будущем развитие системы за счет более сложных моделей, учета временных зависимостей и оптимизации под производственные задачи позволит достичь еще более высоких результатов.

Приложения - Примеры кода для предобработки данных и построения моделей