**Московский государственный университет имени М.В. Ломоносова**

**Направление 38.03.01 Экономика**

**Программа «Национальная экономика»**

# Прогнозирование рисков банковского кредитования с использованием технологий искусственного интеллекта

**Выпускная квалификационная работа**

**Студент**

Касенова Асем Ардаковна

**Научный руководитель:**

к.э.н., к.ф.-м.н., к.ю.н., доцент

Сидоренко Владимир Николаевич

**Астана**

**2026**

# ОГЛАВЛЕНИЕ

# ГЛАВА 1. АЛГОРИТМЫ МАШИННОГО И ГЛУБОКОГО ОБУЧЕНИЯ В ЗАДАЧЕ О КРЕДИТНОМ СКОРИНГЕ

## 1.1 Предобработка данных кредитной истории клиентов банка

В данной работе используются обезличенные данные АО «Альфа-Банка»[1]. Данные состоят из 12 файлов (`train_data_0.pq - train_data_11.pq`), содержащих информацию о платежах клиентов банка. В каждом из 12 файлов содержится информация о 250 000 клиентах. При этом один клиент может иметь несколько кредитов, и каждому такому клиенту соответствует персональный `id` (идентификационный номер). Отдельно имеется файл `train_target.csv`, который состоит из 3 млн строк, и каждая строка соответствует клиенту с меткой (флагом) равной 0 (отсутствие дефолта) или 1 (наличие дефолта). Задача предобработки данных состоит в структурировании исходной информации, т.е. формирование единого датасета, выделение важных признаков (колонок), выявление аномальных клиентов (определение аномальности будет приведено ниже), визуализация данных и тестирование моделей МО и ГО на этих данных. Программный код формирования единого датасета реализован в листинге 1 (см. приложение 1).

**Комментарий 1.** *Пояснение к листингу 1:*

1. Строки 1 – 3. Импортируются необходимые библиотеки: `pandas` – для работы с табличными данными, `os` – для работы с файловой системой и `pyarrow.parquet` – для чтения файлов формата `.parquet`.

2. Строка 5. Задается путь `path = "train_data"` к папке, в которой находятся исходные файлы формата `.pq`.

3. Строки 6 – 13. Запускается цикл `for`, который перебирает все файлы в папке `train_data`. Формируется имя поочередного файла `train_data_i.pq`, создается объект `ParquetDataset` для текущего файла, из которого данные считываются в `DataFrame (df)`. Затем выполняется агрегация данных по признаку `id`, вычисляются средние значения признаков, после чего данные сохраняются в соответствующий `csv` – файл в папку `train_data_csv_all`.

4. Строка 14. Выводится список файлов каталога `train_data` для проверки корректности формирования файлов.

5. Строки 16 – 21. Задается путь к папке с полученными `csv`-файлами `train_data_csv_all`. Создается пустой список `frames` для последующего хранения отдельных `DataFrame`.

[1] **Alfabank** (дата обращения 25.11.2025)

Затем запускается цикл по файлам в папке `train_data_csv_all`, который последовательно перебирает все файлы в формате `.csv`. Результат сохраняется в `DataFrame (df)` и добавляется в список `frames`.

6. Строки 23 – 26. Все элементы списка `frames` объединяются в единый `DataFrame result` с помощью функции `pd.concat`. Полученный датасет сохраняется в файл `1_data_csv_all.csv`, после чего заново считывается в переменную `df_all` для последующего анализа.

После преобразования получился следующий датасет. В таблице 1 приводится фрагмент датасета:

Таблица 1 – Фрагмент преобразованного датасета, содержащий первых 5 клиентов

| id | enc_paym_0 | enc_paym_1 | enc_paym_2 | enc_paym_3 | enc_paym_4 | flag |
|---|---|---|---|---|---|---|
| 1750000 | 0.17 | 0.17 | 0.17 | 0.33 | 0.67 | 0 |
| 1750001 | 0.00 | 0.75 | 0.75 | 0.75 | 0.75 | 0 |
| 1750002 | 0.39 | 0.33 | 0.72 | 0.67 | 1.17 | 0 |
| 1750003 | 0.18 | 0.23 | 0.41 | 0.50 | 0.55 | 0 |
| 1750004 | 0.60 | 0.60 | 0.60 | 0.60 | 1.20 | 0 |

Примечание: id – идентификатор заявки; `enc_paym_0`, ..., `enc_paym_n` – статусы ежемесячных платежей за последние $n$ месяцев; flag – статус кредита (0=кредит полностью оплачен). Полный датасет состоит из 61 признака, включая дополнительные характеристики по кредитам.

**Комментарий 2.** *Пояснение к агрегированию клиентов в листинге 1:*

Агрегация клиентов по идентификатору `id` в строке 11 необходима для определения итоговой метки (флага) каждого клиента, поскольку одному заемщику может соответствовать несколько кредитных договоров. Задача состоит в присвоении каждому кредиту одного заемщика единую итоговую. метку. В данном исследовании в качестве общего значения для всех кредитов используется среднее исходных значений. Для понимания идеи приводится следующий пример в виде таблицы 2 :

Таблица 2 – Дисциплина оплаты кредитов клиента (id = 1)

| id | N | M1 | M2 | M3 | M4 | M5 | M6 | flag |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | |
| 1 | 2 | 0 | **1** | 0 | 1 | 1 | 2 | 0 |
| 1 | 3 | 1 | 2 | 0 | 0 | 3 | 2 | |
| | | Среднее значение | | | | | | |
| 1 | | 0.67 | 1 | 0.33 | 1 | 1.33 | 1.67 | 0 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

В данной таблице признаки означают следующее:

1. `id` – идентификационный номер клиента;

2. `N` – номер кредита;

3. `M1,...,M6` – статусы погашения в течение 6 месяцев (0=платеж вовремя оплачен, 1=задержка платежа на 1 день, 2=задержка платежа на 2 дня, 3=задержка платежа на 3 дня);

4. `flag` – статус кредита (0=кредит полностью оплачен).

В таблице 2 на пересечении N=2 (второго кредита) и M2 (платеж во втором месяце) выделена цифра 1, означающая, что платеж по второму кредиту во втором месяце был оплачен с опозданием на 1 день.

Таким образом, данные трех строк, соответствующих трем кредитам клиента, в таблице 2 были усреднены и преобразованы в одну строку, которая содержит агрегированную информацию по клиенту с `id = 1`. В целом агрегировать клиентов можно не только по среднему значению, но и по моде или медиане. Однако в данной работе выбрано среднее значение, поскольку оно обладает важными статистическими свойствами, такими как несмещенность и состоятельность. После группирования данных был сформирован новый датасет, состоящий из 3 млн клиентов, каждому из которых соответствует одна итоговая метка. Получившийся датасет содержит 61 признак, из которых далее необходимо отобрать наиболее информативные, т.е. такие признаки, существенно влияющие на точность алгоритмов МО и ГО.

### 1.1.1 Метод главных компонент

Метод главных компонент (англ. *principal component analysis*, *PCA*) – метод сокращения размерности данных, позволяющий уменьшать количество признаков с сохранением максимального объема исходной информации[2], на которых обучаются модели МО и ГО.

Как уже было отмечено выше сформированный датасет содержит 61 признак (столбец). Задача состоит в том, чтобы найти такие признаки, на которые модели МО и ГО показывали приемлемую точность. Следует отметить, что редукция признаков может уменьшить точность алгоритмов, поэтому необходимо внимательно следить за процессом сокращения данных. В качестве тестового алгоритма был выбран алгоритм *Random Forest* (случайный лес), поскольку в статье С.В. Смирнова[3] был проведен анализ предпочтения исследователей к алгоритмам МО, показавший, что наиболее популярным является случайный лес.

Из преобразованного датасета изначально выбираются только некатегориальные признаки и формируется новый датасет, содержащий 41 признак. Ниже приводится смысл этих признаков:

[2]**PCA** (дата обращения: 25.11.2025).
[3]**Smirnov** (дата обращения: 25.11.2025).

1. `pre_pterm` – плановое количество дней с даты открытия кредита до даты его закрытия;
2. `pre_fterm` – фактическое количество дней с даты открытия кредита до даты его закрытия;
3. `pre_loans_next_pay_summ` – сумма следующего платежа по кредиту;
4. `pre_loans_outstanding` – оставшаяся невыплаченная сумма кредита;
5. `pre_loans_total_overdue` – текущая просроченная задолженность по кредиту;
6. `pre_loans_max_overdue_sum` – максимальная просроченная задолженность по кредиту за весь срок;
7. `pre_loans_credit_cost_rate` – полная стоимость кредита;
8. `is_zero_loans5` – флаг: нет просрочек до 5 дней;
9. `is_zero_loans530` – флаг: нет просрочек от 5 до 30 дней;
10. `is_zero_loans3060` – флаг: нет просрочек от 30 до 60 дней;
11. `is_zero_loans6090` – флаг: нет просрочек от 60 до 90 дней;
12. `is_zero_loans90` – флаг: нет просрочек более чем на 90 дней;
13. `pre_util` – отношение оставшейся невыплаченной суммы кредита к кредитному лимиту;
14. `pre_maxover2limit` – отношение максимальной просроченной задолженности к кредитному лимиту;
15. `is_zero_util` – флаг: отношение оставшейся невыплаченной суммы кредита к кредитному лимиту равно 0;
16. `is_zero_over2limit` – флаг: отношение текущей просроченной задолженности к кредитному лимиту равно 0;
17. `enc_paym_0`, ..., `enc_paym_n` – статусы ежемесячных платежей за последние $n$ месяцев.

Далее из этих 41 признака необходимо выбрать только такие, которые вносят наибольший вклад в информативность данных. В листинге 2 (см. Приложение 1) реализован метод главных компонент. На таком наборе данных алгоритм показывает следующие метрики (см. Таблицу 3):

Таблица 3 – Метрики точности на 41 признаке

| Алгоритм – Случайный лес | | | | | | |
|---|---|---|---|---|---|---|
| Flag метка | Precision точность | Recall полнота | f1–score f1–мера | Accuracy точность | ROC–AUC | Количество |
| Тренировочная выборка | | | | | | |
| 0 | 1.00 | 0.97 | 0.98 | 0.97 | 0.67 | 724650 |
| 1 | 0.00 | 0.00 | 0.00 | | | 25350 |
| Тестовая выборка | | | | | | |
| 0 | 1.00 | 0.97 | 0.98 | 0.97 | 0.50 | 241508 |
| 1 | 0.00 | 0.00 | 0.00 | | | 8493 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Были получены результаты $PCA$:

$$\begin{aligned}
&\lambda_1^{(p)} = 22.3, \lambda_2^{(p)} = 20.3, \lambda_3^{(p)} = 17.6, \lambda_4^{(p)} = 10.8, \lambda_5^{(p)} = 9.5,\\
&\lambda_6^{(p)} = 7.6,\ \lambda_7^{(p)} = 4.2,\ \lambda_8^{(p)} = 1.7,\ \lambda_9^{(p)} = 1.5,\ \lambda_{10}^{(p)} = 0.8,\\
&\lambda_{11}^{(p)} = 0.6,\ \lambda_{12}^{(p)} = 0.5,\ \lambda_{13}^{(p)} = 0.4,\ \lambda_{14}^{(p)} = 0.3,\ \lambda_{15}^{(p)} = 0.3,\\
&\lambda_{16}^{(p)} = 0.2,\ \lambda_{17}^{(p)} = 0.2,\ \lambda_{18}^{(p)} = 0.1,\ \lambda_{19}^{(p)} = 0.1,\ \lambda_{20}^{(p)} = 0.1,\\
&\lambda_{21}^{(p)} = 0.1,\ \lambda_{22}^{(p)} = 0.1,\ \lambda_{23}^{(p)} = 0.1,\ \lambda_{24}^{(p)} = 0.1,\ \lambda_{25}^{(p)} = 0.1,\\
&\lambda_{26}^{(p)} = 0.1,\ \lambda_{27}^{(p)} = 0.1,\ \lambda_{28}^{(p)} = 0.1,\ \lambda_{29}^{(p)} = 0.1,\ \lambda_{30}^{(p)} = 0.1,\\
&\lambda_{31}^{(p)} = 0.0,\ \lambda_{32}^{(p)} = 0.0,\ \lambda_{33}^{(p)} = 0.0,\ \lambda_{34}^{(p)} = 0.0,\ \lambda_{35}^{(p)} = 0.0,\\
&\lambda_{36}^{(p)} = 0.0,\ \lambda_{37}^{(p)} = 0.0,\ \lambda_{38}^{(p)} = 0.0,\ \lambda_{39}^{(p)} = 0.0,\ \lambda_{40}^{(p)} = 0.0,\\
&\lambda_{41}^{(p)} = 0.0
\end{aligned} \tag{1}$$

В формуле (1) $\lambda_1^{(p)}, \ldots, \lambda_{41}^{(p)}$ – это доли собственных чисел ковариационной матрицы, выраженные в процентах. Начиная с $\lambda_{18}^{(p)}$ эта доля не превышает 0.2%. Это означает, что существует 24 компонента, которые вносят незначительный вклад в информативность данных. Метод $PCA$ не представляет возможности точно определять какие именно признаки вносят существенный вклад в информативность данных, поэтому необходимо самостоятельно выбирать и удалять признаки с низким вкладом. Было сделано предположение, что наименее информативными признаками являются: `pre_pterm`, `pre_fterm`, `pre_loans_next_pay_summ`, `pre_loans_outstanding`, `pre_loans_total_overdue`, `pre_loans_max_overdue_sum`, `pre_loans_credit_cost_rate`, `is_zero_loans5`, `is_zero_loans530`, `is_zero_loans3060`, `is_zero_loans6090`, `is_zero_loans90`, `pre_util`, `pre_maxover2limit`, `is_zero_util`, `is_zero_over2limit`.

Для подтверждения необходимо повторно проделать метод $PCA$ без 16 признаков, касательно которых было сделано предположение. В листинге 3 (см. приложение

1) реализуется метод главных компонент.

1. Строки 1–2. Задаётся файл с датасетом из 25 признаков, который считывается в объект `DataFrame(df)`.

2. Строки 4–10. Формируется список со статусами ежемесячных платежей `enc_paym_k`, $k = 0, \ldots, 24$.

3. Строка 12. `X_pay = df.loc[:, columns_pay].copy()` – из исходного датасета `df` выбираются 25 признаков, которые формируют матрицу `X_pay`, содержащую информацию о платёжной дисциплине.

4. Строка 13. Создаётся объект метода главных компонент PCA с параметрами по умолчанию.

5. Строка 14. На матрице `X_pay` обучается модель PCA.

6. Строка 16. Вычисляются доли объясненной дисперсии для каждого главного компонента.

Получается следующая значимость признаков, процентно выраженная в собственных числах ковариационной матрицы `X_pay`.

$$\begin{aligned}
&\lambda_1^{(p)} = 66.3, \lambda_2^{(p)} = 15.6, \lambda_3^{(p)} = 5.5, \lambda_4^{(p)} = 2.9, \lambda_5^{(p)} = 1.8, \\
&\lambda_6^{(p)} = 1.3,\ \lambda_7^{(p)} = 1.0,\ \lambda_8^{(p)} = 0.8, \lambda_9^{(p)} = 0.7, \lambda_{10}^{(p)} = 0.6, \\
&\lambda_{11}^{(p)} = 0.5,\ \lambda_{12}^{(p)} = 0.4,\ \lambda_{13}^{(p)} = 0.4, \lambda_{14}^{(p)} = 0.3, \lambda_{15}^{(p)} = 0.3, \\
&\lambda_{16}^{(p)} = 0.3,\ \lambda_{17}^{(p)} = 0.3,\ \lambda_{18}^{(p)} = 0.2, \lambda_{19}^{(p)} = 0.2, \lambda_{20}^{(p)} = 0.1, \\
&\lambda_{21}^{(p)} = 0.1,\ \lambda_{22}^{(p)} = 0.1,\ \lambda_{23}^{(p)} = 0.1, \lambda_{24}^{(p)} = 0.1, \lambda_{25}^{(p)} = 0.1.
\end{aligned} \tag{2}$$

Из формулы (2) можно заметить, что осталось только 25 признаков, где доли собственных чисел $\lambda_i^{(p)}$ не равны нулю, т.е. остались те признаки, которые вносят существенный вклад в информативность данных. Однако из таблицы 3 видно, что некоторые метрики для алгоритма случайного леса равны нулю, что является неприемлемым для прогнозирования кредитных рисков. Например, это метрики *Recall* и *Precision*, для дефолтных клиентов. Таким образом, метод $PCA$ не оказал влияния на некоторые метрики алгоритма, поэтому требуется дальнейший детальный анализ.

### 1.1.2 Определение аномального клиента

Рисунок 1 показывает, что доля недефолтных клиентов существенно велика и не соответствует статистике в реальной жизни. По данным Первого кредитного бюро Республики Казахстан около 20%[4] казахстанских заемщиков имеют дефолтную кредитную

[4] **Tengrinews** (дата обращения: 25.11.2025).

историю, что подтверждает данное утверждение и указывает на наличие аномальности в классе недефолтных клиентов.

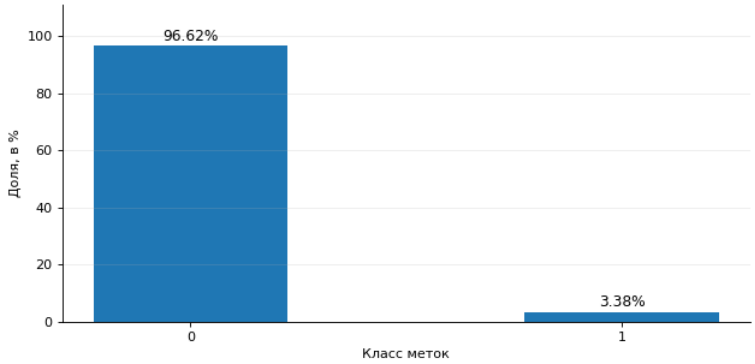


Рисунок 1 – Перекос в сторону недефолтных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

В данной работе под аномальным клиентом понимается нетипичное поведение для большинства заемщиков. Речь идет о ситуации, когда клиент фактически не располагает достаточными денежными средствами для обслуживания кредита, однако продолжает вносить ежемесячные платежи с небольшими, но регулярными опозданиями, перезанимая необходимые суммы из других источников. В таблице 2 приведен пример неестественных выплат клиента, из которого видно, что платежная дисциплина клиента по отдельному кредиту не выглядит дефолтной, однако в среднем этот клиент не выплачивал обязательства пунктуально ни в один расчетный месяц (средние значения по месяцам представлены в последней строке таблицы). Ниже будет дано определение и алгоритм выявления таких клиентов в датасете.

**Определение 1.** *Аномальным клиентом является такой клиент, для которого выполняются два условия:*

$$\begin{cases} (a)\ \textit{метка (флаг)} = 0, \\ (b)\ \min(enc\_paym\_n) > 0, \quad n = 0, \ldots, 24. \end{cases} \tag{anom}$$

В формуле $(anom)$ условие $(a)$ означает, что аномальный клиент в датасете является недефолтным, а условие $(b)$ – по всем месяцам оплата кредитных обязательств проводилась с опозданием. В листинге 4 (см. приложение 1) указана реализация формулы $anom$.

**Комментарий 3.** *Пояснение к листингу 4:*

1. Строка 1. Читаем файл `train_target.csv` с целевой переменной в датафрейме `df_y`.

2. Строки 2–3. Из столбца `flag` датафрейма `df_y` извлекаются метки для первых 1000000 клиентов, где `y` – это исходные метки дефолта/недефолта, а `y_` – новые метки, которые будут изменяться в процессе нахождения аномальных клиентов.

3. Строка 5. Инициализируются счетчики `counter_norm` – число не аномальных клиентов, а `counter_anom` – число аномальных клиентов.

4. Строка 6. Создаются пустые списки, в которые будут записываться индексы не аномальных (`ind_norm`) и аномальных клиентов (`ind_anom`).

5. Строка 7. Запускается цикл по всем строкам матрицы `X_pay`, где `i` это порядковый номер клиента (индекс строки), а `x_pay` – массив платежной истории по всем месяцам.

6. Строки 8–11. Проверяется условие, которое определяет изначально недефолтных клиентов (`y_[i] == 0`), у которых по всем месяцам наблюдаются задержки платежей (`min(x_pay) > 0`). Если оба условия для клиента выполняются, то счетчик аномальных клиентов увеличивается на 1 (`counter_anom +=1`) и пополняется список индексов аномальных клиентов (`ind_anom.append(i)`), а его новой метке `y_[i]` присваивается значение 1. Таким образом, часть клиентов, имевших исходную метку 0, переводится в класс аномальных клиентов с меткой 1.

7. Строки 12–14. В случае неудовлетворения этим двум условиям цикл относит клиента к не аномальным (`counter_norm += 1`) и добавляет его индекс в список не аномальных клиентов (`ind_norm`).

Данный алгоритм показал, что доля дефолтных клиентов увеличилась с 3.38% до 35.45% (см. Рисунок 2). Это означает, что часть недефолтных клиентов перешла в класс дефолтных, и именно они удовлетворяют определению (*anom*).

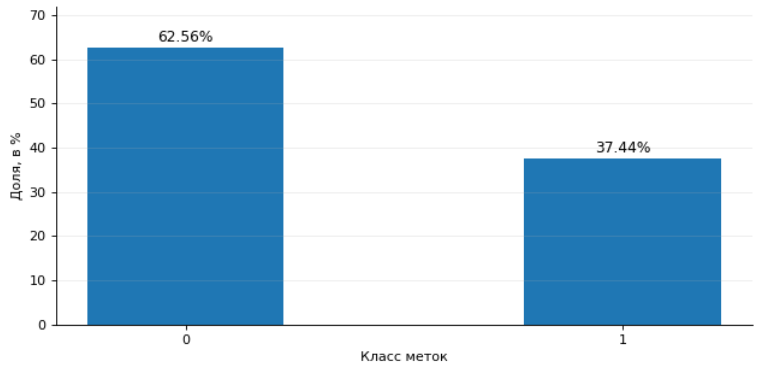


Рисунок 2 – Переход клиентов из класса «0» в класс «1»

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Для большей наглядности результат переклассификации представлен на диаграмме Эйлера-Венна (см. Рисунок 3), показывающий пересечение множеств клиентов с дефолтными и недефолтными метками. Красный круг соответствует всем клиентам с исходной недефолтной меткой (96.62%), зеленый круг – клиентам с дефолтной меткой (3.38%), а их пересечение отражает группу аномальных клиентов (32.06%).

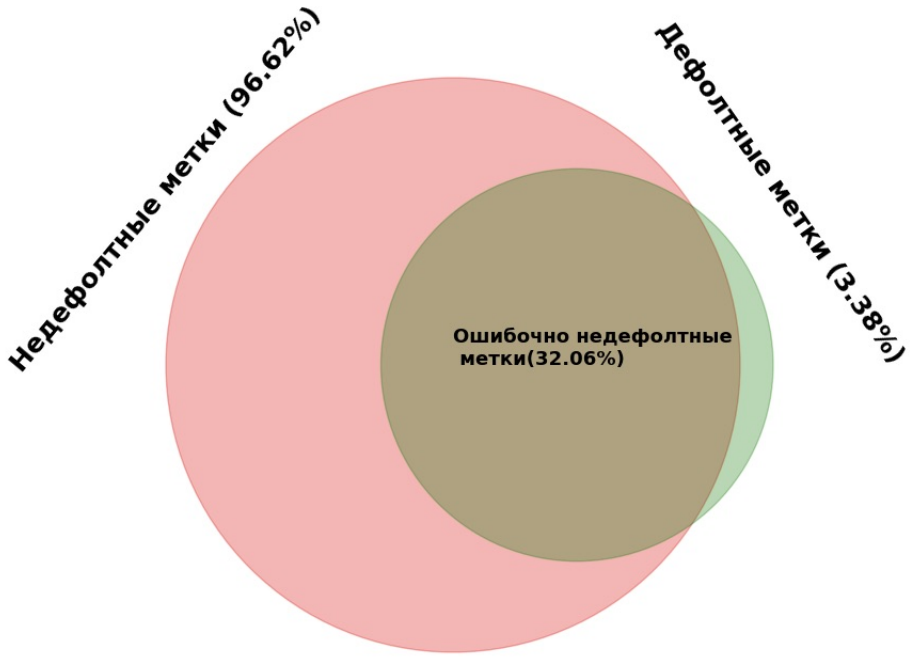


Рисунок 3 – Пересечение дефолтных и недефолтных меток с выделением ошибочно недефолтных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Далее необходимо посмотреть как отреагирует алгоритм случайного леса на данных преобразованиях согласно определению ($anom$). Под реакцией алгоритма понимается изменение точности метрик. После выявления аномальных клиентов качество модели значительно улучшилось (см. Таблицу 4).

Таблица 4 – Изменение точности метрик на признаках: $enc\_paym\_0, \ldots, enc\_paym\_24$.

| Алгоритм – Случайный лес | | | | | | |
|---|---|---|---|---|---|---|
| Flag<br>метка | Precision<br>точность | Recall<br>полнота | f1–score<br>f1–мера | Accuracy<br>точность | ROC–AUC | Количество |
| Тренировочная выборка | | | | | | |
| 0 | 0.99 | 0.97 | 0.98 | 0.97 | 0.98 | 469239 |
| 1 | 0.94 | 0.99 | 0.96 | | | 280761 |

| Тестовая выборка | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 0.99 | 0.97 | 0.98 | 0.97 | 0.98 | 156325 |
| 1 | 0.94 | 0.99 | 0.96 | | | 93676 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Если сравнить первоначальные результаты, то можно заметить, что если до этого модель игнорировала класс дефолтных клиентов (все метрики равнялись нулю, см. Таблицу 3), то после переклассификации метрики точности стали высокими: на обучающей выборке $precision = 0.94$, $recall = 0.99$, $f1 - score = 0.96$, аналогичные значения показывает тестовая выборка. При этом можно качество недефолтного класса практически не ухудшилось: $precision$ снизился с 1.00 до 0.99, а метрики $recall$ и $f1 - score$ не изменились. Также $ROC\ AUC$ увеличился с 0.51 до 0.97, что свидетельствует о том, что теперь модель хорошо различает дефолтных и недефолтных клиентов. Одновременно увеличилось количество дефолтных клиентов в выборках за счет вновь выявленных аномальных наблюдений, что позволило алгоритму случайного леса показать эффективные результаты на обоих классах.

### 1.1.3 Статистический тест Колмогорова-Смирнова

Понятие аномального клиента, изложенное в разделе 1.1.2, основано на анализе платежного поведения заемщиков. Применение данного подхода привело к существенному улучшению метрик точности алгоритма случайного леса. Поэтому на следующем шаге проводится статистический анализ, позволяющий увидеть и обосновать, насколько поведение аномальных клиентов отличается от поведения дефолтных клиентов. Для проверки равенства (неравенства) функций распределения двух выборок применяется критерий Колмогорова-Смирнова. А.Н. Колмогоровым[5] и Н.В. Смирновым[6] была доказана следующая теорема:

**Теорема 1.** *Пусть $F_{X_1}(x)$, $F_{X_2}(x)$ – выборочные функции распределения двух независимых выборок объёмами $n$ и $m$. Обозначим:*

$$D_{n,m} = \sup_x \left| F_{X_1}(x) - F_{X_2}(x) \right|.$$

[5]Колмогоров А.Н. (1903-1987 гг.) – Герой Социалистического Труда, профессор Московского государственного университета, академик АН СССР – крупнейший математик XX века, является одним из основоположников современной теории вероятности.

[6]Смирнов Н.В. (1900-1966 гг.) – член-корреспондент АН СССР, один из создателей непараметрических методов математической статистики и теории предельных распределений порядковых статистик.

Тогда для любого $t > 0$ выполняется:

$$\forall t > 0: \quad \lim_{n,m\to\infty} P\left(\sqrt{\frac{nm}{n+m}}\, D_{n,m} \le t\right) = K(t) = \sum_{j=-\infty}^{+\infty} (-1)^j e^{-2j^2t^2}. \tag{3}$$

На основе статистики $D_{n,m}$ строится статистика критерия

$$t_{n,m} = \sqrt{\frac{nm}{n+m}}\, D_{n,m}.$$

С помощью этой статистики проверяются нулевая и альтернативная гипотезы о совпадении распределений двух выборок:

$$\begin{cases} \mathcal{H}_0 : F_{X_1}(x) = F_{X_2}(x), \\ \mathcal{H}_1 : F_{X_1}(x) \neq F_{X_2}(x). \end{cases}$$

При заданном уровне значимости $\alpha$ выбирается критическое значение $K_\alpha$ распределения Колмогорова. Правило принятия решения имеет следующий вид:

$$\begin{cases} t_{n,m} \le K_\alpha, & \text{нулевая гипотеза } \mathcal{H}_0 \text{ принимается}, \\ t_{n,m} > K_\alpha, & \text{нулевая гипотеза } \mathcal{H}_0 \text{ отвергается в пользу } \mathcal{H}_1. \end{cases}$$

Для применения критерия Колмогорова-Смирнова клиенты были разделены на две подвыборки:

$$\begin{cases} X_1 \coloneqq X_{1\to1}, & \textit{(default)}, \\ X_2 \coloneqq X_{0\to1}, & \textit{(anom)}, \end{cases}$$

где $X_1$ – выборка клиентов, являющихся дефолтными до и после условия аномальности; $X_2$ – выборка клиентов, у которых метка изменилась с «0» на «1».

Задача состоит в том, чтобы выяснить, принадлежат ли выборки $X_1$ и $X_2$ одному распределению, применив критерий Колмогорова-Смирнова. Если выборки $X_1$ и $X_2$ принадлежат одному распределению (принятие гипотезы $\mathcal{H}_0$), то платежная дисциплина дефолтных и аномальных клиентов одинакова. Критерий Колмогорова-Смирнова реализуется в листинге 5 (см. приложение 1).

**Комментарий 4.** *Пояснение к листингу 5:*

1. Строки 1–2. Выбирается признак за 15 месяц `enc_paym_15` и соответствующий столбец из `X_pay` сохраняется в переменную `T`.

2. Строки 3–4. Формируются три выборки по этому признаку: `x00` – клиенты с меткой «0» до и после выявления аномальных клиентов; `x01` – аномальные клиенты,

у которых изменилась метка с «0» на «1» после выявления аномальности; `x11` – клиенты с меткой «1» до и после выявления аномальных клиентов.

3. Строки 5–7. Происходит фильтрация выборок `x00`, `x01`, `x11`, у которых отбрасываются наблюдения с платежными статусами 0, 1, 2, 3, 4, 5, поскольку агрегация данных была произведена по среднему значению.

4. Строка 10. Определяется функция `my_cdf` по двум выборкам `x1` и `x2`.

5. Строка 11. Выбирается максимальное значение (опоздание) из этих двух выборок.

6. Строка 12. Создаются пустые списки `F1` и `F2`, в которые будут записываться значения.

7. Строка 13. Определяется длина выборок `x1` и `x2`, состоящих из положительных значений.

8. Строки 14–16. В цикле для каждой точки `tm` с шагом 0.01 считается сколько наблюдений для выборок `x1` и `x2` попадают в интервал $(0;\, t)$.

9. Строки 17–19. Вычисляется вероятность каждой точки и получаются значения функции распределения ($CDF$). Далее считается статистика Колмогорова по формуле: $\sup_x \left|F_{X_1}(x) - F_{X_2}(x)\right| \sqrt{\frac{nm}{n+m}}$.

10. Строки 20–23. Задается шаг `step = 0.01` и вычисляются значения функции распределения $CDF$ для пары «аномальных и дефолтных», а затем для «аномальных и недефолтных» клиентов.

Датасет содержит 25 дисциплинарных признаков ($enc\_paym\_0, \ldots, enc\_paym\_24$), и для каждого из них необходимо применить данный тест, чтобы выяснить, для каких признаков принимается та или иная гипотеза. Поскольку в датасете 3 000 000 наблюдений (объемы выборок $n$ и $m$ велики), то при уровне значимости $\alpha = 0.05$ в качестве критического значения используется $K_\alpha = 1.36$.

Анализ показал, что первые 15 месяцев ($enc\_paym\_0$, …, $enc\_paym\_14$) поведение аномальных клиентов отличается от поведения дефолтных, а начиная с 16 по 25 месяц ($enc\_paym\_15$, …, $enc\_paym\_24$), поведение становится схожим. Так, например, для 15 месяца ($enc\_paym\_14$) (см. Рисунок 4) расчетное значение статистики $t_{n,m} = 1.41$. Поскольку $K_\alpha = 1.36$ и $t_{n,m} > K_\alpha$, принимается гипотеза $\mathcal{H}_1$. Это означает, что финансовое поведение аномальных клиентов, раннее относившихся к классу «0», отличается от поведения дефолтных клиентов.

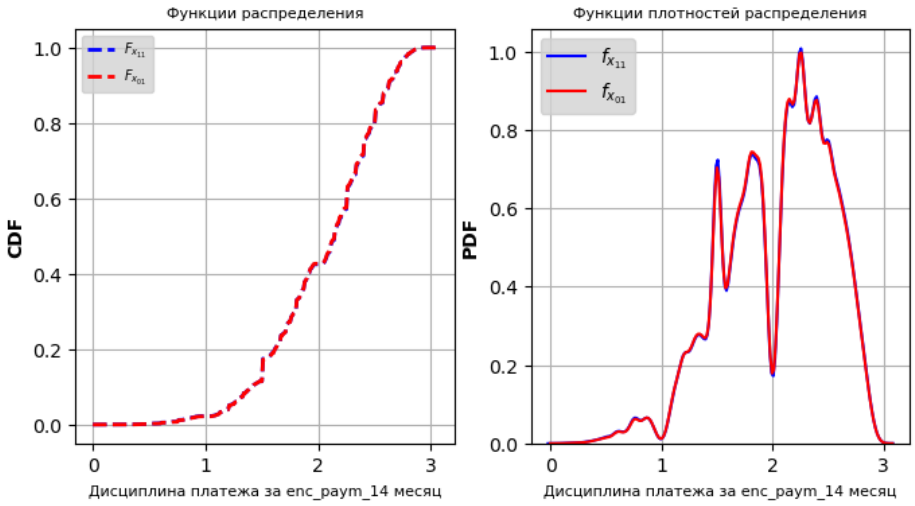


Рисунок 4 – Функции распределения и функции плотностей распределения дефолтных и аномальных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Для 16 месяца ($enc\_paym\_15$) (см. Рисунок 5) значение $t_{n,m} = 1.31$. В этом случае $t_{n,m} < K_\alpha$, поэтому принимается гипотеза $\mathcal{H}_0$. Распределения значений признака для аномальных и дефолтных клиентов не различаются.

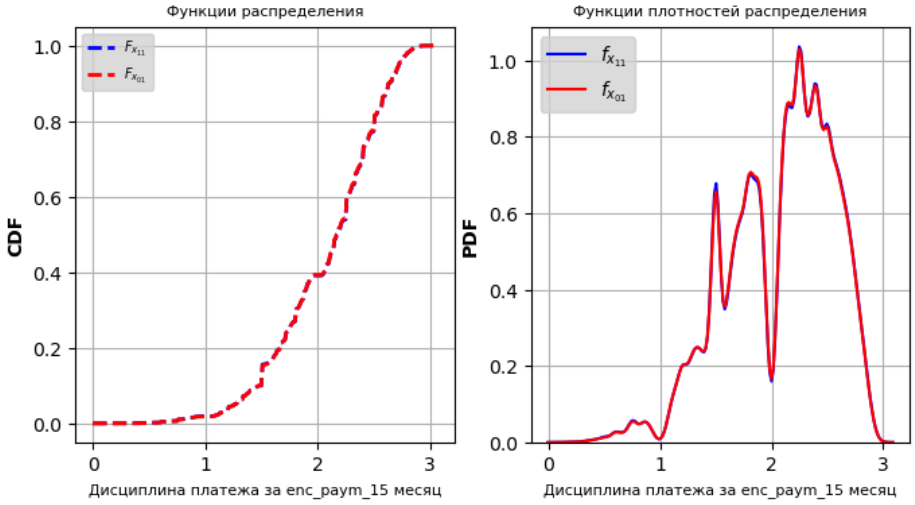


Рисунок 5 – Функции распределения и функции плотностей распределения дефолтных и аномальных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Таким образом, поведение аномальных и дефолтных клиентов до 15 месяца различно, а начиная с 16 месяца их поведение становится схожим. Возникает следующий вопрос, если поведение аномальных клиентов отличается от поведения дефолтных ($enc\_paym\_15, \ldots, enc\_paym\_24$), то не является ли оно более близким к поведению недефолтных клиентов.

При сравнении аномальных и недефолтных клиентов анализ показал, что за 15 месяц (см. Рисунок 6) их выплаты также существенно различались, поскольку при значении статистики $t_{n,m} = 70.92$ и условии $t_{n,m} > K_\alpha$ принимается гипотеза $\mathcal{H}_1$, что свидетельствует об их абсолютном различии по поведению.

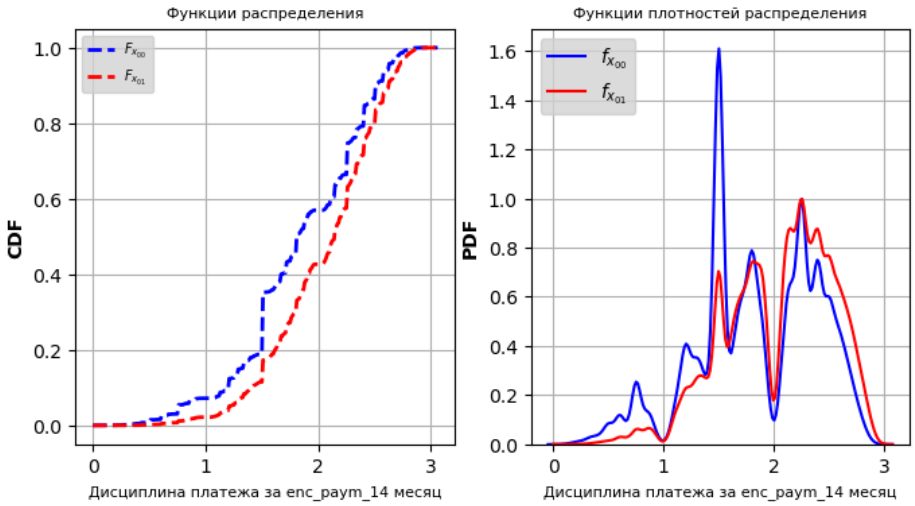


Рисунок 6 – Функции распределения и функции плотностей распределения недефолтных и аномальных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Если сравнить поведение аномальных и недефолтных клиентов за 16 месяц (см. Рисунок 7), то также наблюдается существенное различие. Расчетное значение статистики $t_{n,m} = 68.6$ и поскольку $t_{n,m} > K_\alpha$, принимается гипотеза $\mathcal{H}_1$.

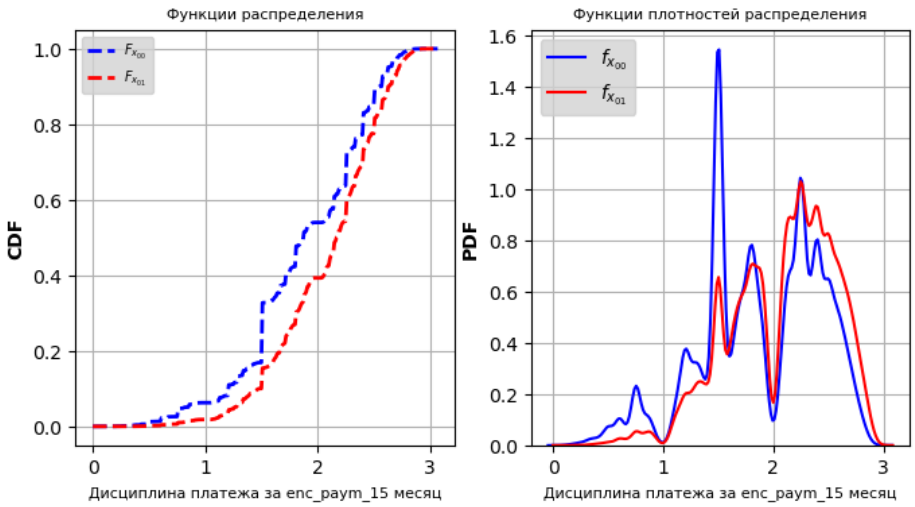


Рисунок 7 – Функции распределения и функции плотностей распределения недефолтных и аномальных клиентов

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

В результате анализа при сравнении аномальных и дефолтных клиентов была выявлена их схожесть после 16 месяца, тогда как при сравнении аномальных и недефолтных за весь рассматриваемый период наблюдается устойчивое расхождение. Для наглядности результаты сведены в следующую таблицу (см. Таблицу 5).

Таблица 5 – Результаты критерия Колмогорова–Смирнова за 25 месяцев

| Месяц | Аномальные и дефолтные клиенты | | Аномальные и недефолтные клиенты | |
|---|---|---|---|---|
| | Статистика | Принятие гипотезы | Статистика | Принятие гипотезы |
| enc_paym_0 | $t_{n,m} = 1.65$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 61.40$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_1 | $t_{n,m} = 2.49$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 72.44$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_2 | $t_{n,m} = 3.09$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 62.21$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_3 | $t_{n,m} = 3.33$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 62.03$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_4 | $t_{n,m} = 3.29$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 63.85$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_5 | $t_{n,m} = 3.20$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 65.22$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_6 | $t_{n,m} = 2.88$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 67.04$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_7 | $t_{n,m} = 2.50$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 61.43$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_8 | $t_{n,m} = 2.28$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 56.28$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_9 | $t_{n,m} = 2.00$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 60.88$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_10 | $t_{n,m} = 1.90$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 68.11$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_11 | $t_{n,m} = 1.85$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 71.93$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_12 | $t_{n,m} = 1.72$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 74.84$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_13 | $t_{n,m} = 1.53$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 73.55$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_14 | $t_{n,m} = 1.41$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ | $t_{n,m} = 70.92$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_15 | $t_{n,m} = 1.31$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 68.86$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_16 | $t_{n,m} = 1.17$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 68.26$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_17 | $t_{n,m} = 1.03$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 67.03$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_18 | $t_{n,m} = 0.99$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 65.48$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_19 | $t_{n,m} = 0.84$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 64.54$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_20 | $t_{n,m} = 0.79$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 62.74$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_21 | $t_{n,m} = 0.80$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 60.50$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_22 | $t_{n,m} = 0.80$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 58.80$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_23 | $t_{n,m} = 0.80$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 57.53$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |
| enc_paym_24 | $t_{n,m} = 0.68$, $t_{n,m} < K_\alpha$ | $\mathcal{H}_0$ | $t_{n,m} = 53.10$, $t_{n,m} > K_\alpha$ | $\mathcal{H}_1$ |

Примечание: $K_\alpha = 1.36$ при уровне значимости $\alpha = 0.05$.

Из результатов применения критерия Колмогорова-Смирнова (см. Таблицу 5) видно, что все значения статистики $t_{n,m}$ при сравнении распределений $F_{X_{01}}(x)$ и $F_{X_{11}}(x)$ меньше, чем при сравнении $F_{X_{01}}(x)$ и $F_{X_{00}}(x)$. Отсюда следует, что аномальные клиенты по своему платежному поведению ближе к классу дефолтных заемщиков, чем к недефолтным.

### 1.1.4 Сильная корреляция соседних признаков

Для последующей редукции признакового пространства и улучшения метрик алгоритмов была проанализирована корреляция Пирсона[7] соседних признаков $enc\_paym\_0$, ..., $enc\_paym\_24$. На практике некоторые признаки могут дублировать друг друга (явление мультиколлинеарности) и следовательно из пары сильно коррелирующих признаков необходимо оставить только один признак. В работе были выделены три группы клиентов: недефолтные (0), дефолтные (1) и аномальные ($0 \to 1$). Для детального анализа построен график (см. Рисунок 8), на котором представлены четыре коэффициента корреляции:

1. $corr\_all$ – общий коэффициент корреляции;
2. $corr\_nodefault$ – коэффициент корреляции для недефолтных клиентов;
3. $corr\_default$ – коэффициент корреляции для дефолтных клиентов;
4. $corr\_anom$ – коэффициент корреляции для аномальных клиентов.

Видно, что платежная дисциплина аномальных и дефолтных клиентов схожа друг с другом в отличие от аномальных и недефолтных клиентов.

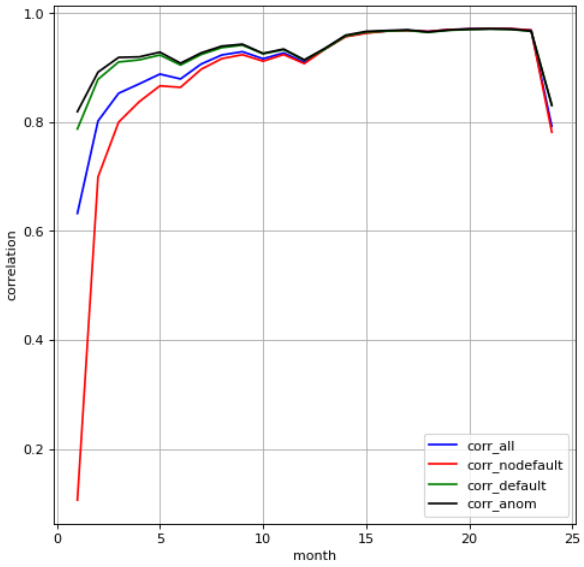


Рисунок 8 – Изменение корреляции между соседними признаками

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

В таблице указаны значения коэффициентов корреляции и теста Фишера[8] на равенство выборочного коэффициента корреляции ($r$) некоторому теоретическому коэффициенту корреляции ($R$) всех значений соседних признаков. Фишер показал, что

[7]Пирсон К. (1857-1936 гг.) – британский математик, один из основоположников математической статистики.

[8]Фишер. Р.Э. (1890-1962 гг.) – британский статистик, один из основателей математической статистики и математической популяционной генетики.

статистика ($z$-преобразование Фишера):

$$z(r) = \frac{1}{2}\ln\frac{1+r}{1-r} \quad \textit{(Fisher)}$$

имеет приближенно нормальное распределение

$$z(r) \sim N\left(\frac{1}{2}\ln\frac{1+R}{1-R}, \frac{1}{n-3}\right),$$

где $R$ – некоторый теоретический коэффициент корреляции, $n$ – объем выборки, $r$ – выборочный коэффициент корреляции.

В качестве теоретического коэффициента корреляции ($R$) в исследовании используется коэффициент корреляции всей совокупности клиентов для соответствующих значений соседних признаков. Для каждой из трех групп (недефолтных, дефолтных и аномальных клиентов) вычисляется свой выборочный коэффициент корреляции ($r$). Далее проверяется гипотеза о равенстве корреляций выборочной и генеральной совокупностей:

$$\begin{cases} \mathcal{H}_0 : r = R, \\ \mathcal{H}_1 : r \neq R. \end{cases}$$

В качестве статистики критерия Фишера используется величина

$$t = \frac{z(r) - z(R)}{\sqrt{\frac{1}{n-3}}}$$

При заданном уровне значимости $\alpha$ гипотеза принимается в случаях:

$$\begin{cases} |t| \leq t_{\text{crit}}, & \text{нулевая гипотеза } H_0 \text{ не отвергается,} \\ |t| > t_{\text{crit}}, & \text{нулевая гипотеза } H_0 \text{ отвергается в пользу } H_1. \end{cases}$$

Если посмотреть на эти коэффициенты корреляции в разрезе для каждой группы (см. Таблицу 6), то можно заметить, что начиная со сравнения 16 и 17 месяцев (см. Листинг 6), коэффициент корреляции одной из трех групп принимает гипотезу $\mathcal{H}_0$:$|t| < t_{crit}$. Такая тенденция прослеживается до сравнения 20 и 21 месяца. Поскольку финансовая дисциплина в соседних месяцах схожа друг с другом, то принимается решение об исключении сильно коррелирующих признаков. В таком случае исключаются пять признаков: $enc\_paym\_16$, ..., $enc\_paym\_20$.

Таблица 6 – Результаты теста Фишера за 25 месяцев

| Месяц | $r_{\text{nodefault}}$ | | $r_{\text{default}}$ | | $r_{\text{anom}}$ | | $r_{\text{all}}$ |
|---|---|---|---|---|---|---|---|
| | Статистика | Гипотеза | Статистика | Гипотеза | Статистика | Гипотеза | Статистика |
| enc_paym_1<br>enc_paym_2 | $t_{nod} = 505.11$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 195.08$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 238.49$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.63$ |
| enc_paym_1<br>enc_paym_2 | $t_{nod} = 132.87$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 160.51$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 183.05$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.80$ |
| enc_paym_2<br>enc_paym_3 | $t_{nod} = 95.71$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 133.53$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 147.53$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.85$ |
| enc_paym_3<br>enc_paym_4 | $t_{nod} = 74.37$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 120.26$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 136.01$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.87$ |
| enc_paym_5<br>enc_paym_6 | $t_{nod} = 50.52$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 76.73$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 84.20$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.88$ |
| enc_paym_6<br>enc_paym_7 | $t_{nod} = 40.55$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 66.10$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 75.36$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.91$ |
| enc_paym_7<br>enc_paym_8 | $t_{nod} = 34.57$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 61.13$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 71.94$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.92$ |
| enc_paym_8<br>enc_paym_9 | $t_{nod} = 30.71$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 58.64$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 64.91$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.93$ |
| enc_paym_9<br>enc_paym_10 | $t_{nod} = 21.29$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 36.56$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 37.83$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.92$ |
| enc_paym_10<br>enc_paym_11 | $t_{nod} = 16.36$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 26.75$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 31.11$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.93$ |
| enc_paym_11<br>enc_paym_12 | $t_{nod} = 11.00$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 14.11$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 15.34$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.91$ |
| enc_paym_12<br>enc_paym_13 | $t_{nod} = 4.13$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 3.23$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 4.54$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.93$ |
| enc_paym_13<br>enc_paym_14 | $t_{nod} = 6.38$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 4.42$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 14.78$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.96$ |
| enc_paym_14<br>enc_paym_15 | $t_{nod} = 8.39$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 11.74$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 21.25$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.96$ |
| enc_paym_15<br>enc_paym_16 | $t_{nod} = 2.57$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 3.57$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 6.86$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_16<br>enc_paym_17 | $t_{nod} = 4.14$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 1.81$<br>$\lvert t_{def}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{ano} = 11.66$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_17<br>enc_paym_18 | $t_{nod} = 1.68$<br>$\lvert t_{nod}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{def} = 13.57$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 5.95$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_18<br>enc_paym_19 | $t_{nod} = 1.12$<br>$\lvert t_{nod}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{def} = 11.44$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 7.87$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_19<br>enc_paym_20 | $t_{nod} = 0.71$<br>$\lvert t_{nod}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{def} = 9.48$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 1.87$<br>$\lvert t_{ano}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{all} = 0.97$ |
| enc_paym_20<br>enc_paym_21 | $t_{nod} = 0.64$<br>$\lvert t_{nod}\rvert \leq t_{crit}$ | $\mathcal{H}_0$ | $t_{def} = 5.11$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 3.21$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_21<br>enc_paym_22 | $t_{nod} = 2.86$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 14.00$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 5.77$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_22<br>enc_paym_23 | $t_{nod} = 3.48$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 15.23$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 13.21$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.97$ |
| enc_paym_23<br>enc_paym_24 | $t_{nod} = 22.36$<br>$\lvert t_{nod}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{def} = 69.20$<br>$\lvert t_{def}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{ano} = 66.41$<br>$\lvert t_{ano}\rvert > t_{crit}$ | $\mathcal{H}_1$ | $t_{all} = 0.79$ |

Примечание: $t_{crit} = 1.96$ при уровне значимости $\alpha = 0.05$.

## 1.2 Базовые алгоритмы оценки вероятности возврата кредита

### 1.2.1 Деревья решений

Необходимо протестировать предобработанные данные на таких базовых алгоритмах, как деревья решений ($Decision\ Tree$), случайный лес ($RandomForest$), градиентный бустинг ($GradientBoosting$) и нейронные сети ($NeuralNetworks$). Важно отметить, что алгоритмы тестируются на 1 000 000 данных и полученные метрики точности будут сраниваться с метриками, которые будут получены после тестирования на больших данных (3 000 000 клиентов). Тестирование на небольших данных проводится с целью экономии объема оперативной памяти.

Результаты метрик точности позволяют понять насколько хорошо модель справляется с задачей классификации. Для этого необходимо выявить одно из двух явлений: переобучение или недообучение. Переобучение – явление, когда алгоритм хорошо распознает объекты из обучающей (тренировочной) выборки, но хуже их классифицирует на тестовой выборке. Модель стремится запомнить все возможные примеры из обучающих данных, вместо того, чтобы научиться выявлять особенности. Недообучение – явление, при котором простая модель не успевает уловить сложные закономерности между признаками и меткой дефолта. В этом случае алгоритм показывает низкие метрики на обучающей и тестовой выборках, т.е. модель плохо разделяет надежных и рискованных клиентов даже на тех данных, на которых она обучалась.

Деревья решений ($DecisionTree$) – это один из алгоритмов МО, который по шагам делит заемщиков на группы с помощью последовательных вопросов вида «да» или «нет»[9]. На каждом шаге выбирается такой признак, который сильнее влияет на результат, а в конечных вершинах алгоритм выдает прогноз вероятности дефолта клиента.

Результаты алгоритма решающих деревьев (см. Таблицу 7) демонстрируют высокую точность. Видно, что метрики класса «0» на тренировочной выборке: $precision = 1.00$, $recall = 0.99$, $f1 - score = 0.99$ выше в отличие от тестовой выборки, где $precision = 0.98$, $recall = 0.97$, $f1 - score = 0.97$. Это означает, что алгоритм лучше распознает недефолтных клиентов на тренировочных данных, чем на тестовых. Аналогичная ситуация наблюдается у класса «1», где на тренировочных данных метрики выше, чем на тестовых. При этом результаты на двух выборках ниже, чем у «0» класса, что указывает на то, что дефолтные клиенты хуже распознаются.

[9] **WangHuan** (дата обращения 25.11.2025).

Таблица 7 – Метрики точности алгоритма Decision Tree на 1 000 000 данных

| Flag метка | Precision точность | Recall полнота | f1–score f1–мера | Accuracy точность | ROC–AUC | Количество |
|---|---|---|---|---|---|---|
| Тренировочная выборка | | | | | | |
| 0 | 1.00 | 0.99 | 0.99 | 0.99 | 0.99 | 469239 |
| 1 | 0.98 | 1.00 | 0.99 | | | 280761 |
| Тестовая выборка | | | | | | |
| 0 | 0.98 | 0.97 | 0.97 | 0.97 | 0.97 | 156325 |
| 1 | 0.95 | 0.96 | 0.96 | | | 93676 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Также метрики *accuracy* и $ROC - AUC$ (см. Рисунок 9) выше на тренировочной выборке, чем на тестовой. Следовательно, можно сделать вывод о наличии переобучения алгоритма *Decision Tree*.

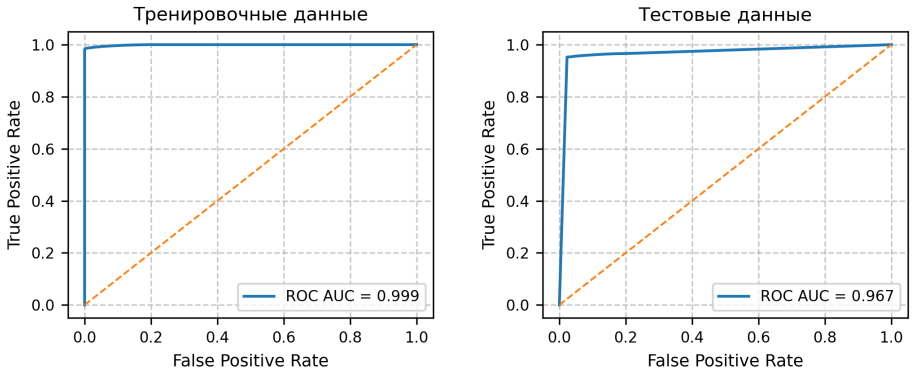


Рисунок 9 – Метрика ROC–AUC на тренировочных и тестовых данных алгоритма Decision Tree

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Для более подробной картины приведена матрица ошибок (см. Таблицу 10). На тренировочных данных верно классифицированы 469 223 недефолтных и 275 761 дефолтных клиентов. При этом 5000 недефолтных клиентов ошибочно отнесены к дефолтным, и только 16 дефолтов алгоритм принял за недефолт.

На тестовой выборке алгоритм больше ошибается. Верно распознаны 152 717 недефолтных и 89 213 дефолтных клиентов. Алгоритм ошибочно принял 4 463 недефолта за дефолт, и 3 608 дефолтов за недефолт. Таким образом, матрица ошибок указывает на ухудшение предсказаний решающих деревьев на тестовых данных, что подтверждает вывод о переобучении данного алгоритма.

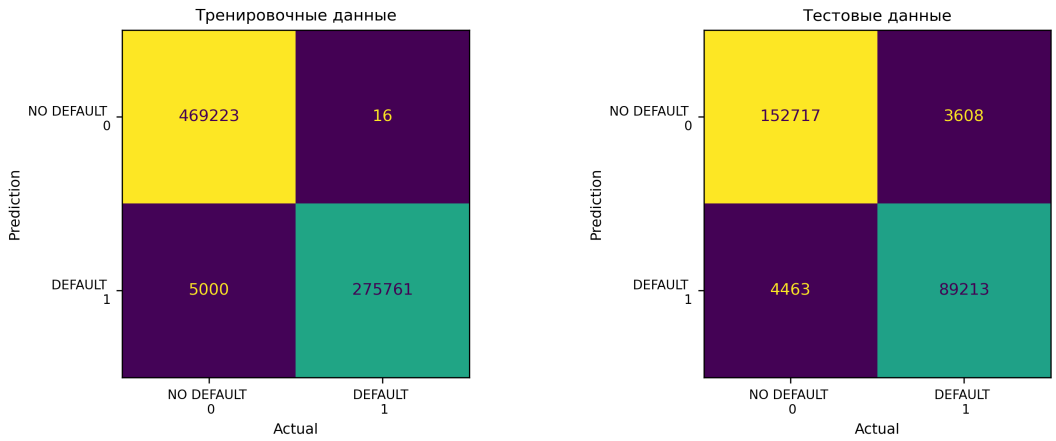


Рисунок 10 – Матрица ошибок на тренировочных и тестовых данных алгоритма Decision Tree

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

### 1.2.2 Случайный лес

Случайный лес (*RandomForest*) – это алгоритм МО, состоящий из множества отдельных независимых деревьев решений. Каждое дерево выдает свое предсказание по возврату кредита, а затем итоговое предсказание определяется по принципу большинства.

Алгоритм случайного леса также показывает высокие результаты распознования клиентов (см. Таблицу 8). Можно заметить, что метрики класса «0» *recall* и $f1-score$ на тренировочных данных выше, чем на тестовых, а метрика *precision* остается на одинаково высоком уровне. Метрики класса «1» *precision* и $f1-score$ также на тренировочной выборке выше, чем на тестовой.

Таблица 8 – Метрики точности алгоритма Random Forest на 1 000 000 данных

| Flag<br>метка | Precision<br>точность | Recall<br>полнота | f1–score<br>f1–мера | Accuracy<br>точность | ROC–AUC | Количество |
|---|---|---|---|---|---|---|
| Тренировочная выборка | | | | | | |
| 0 | 1.00 | 0.99 | 0.99 | 0.99 | 0.99 | 469239 |
| 1 | 0.98 | 1.00 | 0.99 | | | 280761 |
| Тестовая выборка | | | | | | |
| 0 | 1.00 | 0.97 | 0.98 | 0.98 | 0.98 | 156325 |
| 1 | 0.95 | 1.00 | 0.97 | | | 93676 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Метрики *accuracy* и $ROC - AUC$ (см. Рисунок 11) показывают небольшую разницу между тренировочными и тестовыми данными. Это также, как и в предыдущем алгоритме указывает на склонность случайного леса к переобучению.

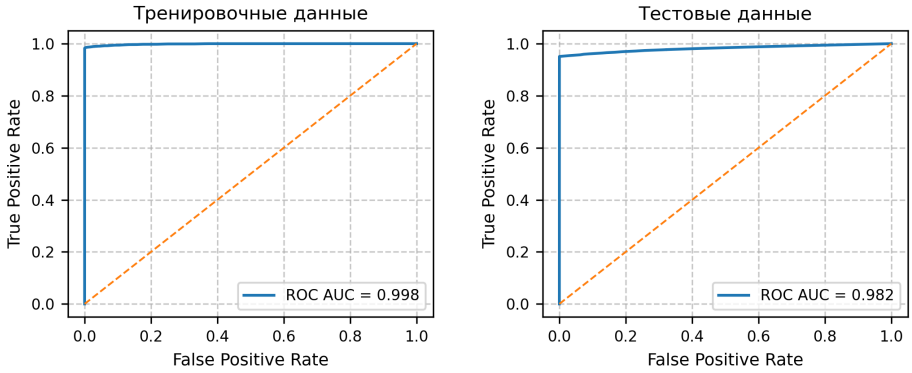


Рисунок 11 – Метрика ROC–AUC на тренировочных и тестовых данных алгоритма Random Forest

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Если посмотреть в количественном разрезе (см. Рисунок 12), то на тренировочных данных алгоритм 469 194 недефолтных клиентов определяет верно, а 4 982 неверно относит к недефолтным. Также 275 779 дефолтов алгоритм правильно определяет, но 45 клиентов неверно отнесены к недефолтным.

На тестовых данных случайный лес показывает хуже результат: 156 170 недефолтных клиентов правильно распознаются, а 4 600 неверно определяются как дефолтные. Также 89 076 дефолтов правильно классифицируются, однако 155 ошибочно относятся к недефолтным. Поэтому матрица ошибок также обосновывает склонность случайного леса к переобучению.

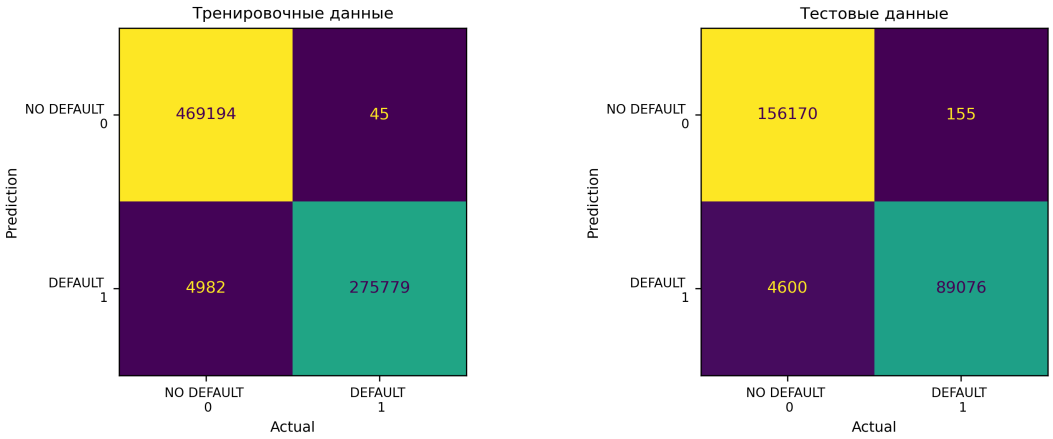


Рисунок 12 – Матрица ошибок на тренировочных и тестовых данных алгоритма Random Forest

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

### 1.2.3 Градиентный бустинг

Градиентный бустинг (*Gradient Boosting*) – комбинация множества простых алгоритмов деревьев решений, где каждый новый алгоритм исправляет ошибки предыдущего.

Все метрики точности алгоритма градиентного бустинга (см. Таблицу 9) обоих классов показывают высокие результаты, что делает данную модель почти идеальной. Небольшим недостатком является более низкая точность метрик класса «1» на обеих выборках, что свидетельствует о более высокой ошибочной вероятности предсказаний на дефолтных клиентах.

Таблица 9 – Метрики точности алгоритма Gradient Boosting на 1 000 000 данных

| Flag метка | Precision точность | Recall полнота | f1–score f1–мера | Accuracy точность | ROC–AUC | Количество |
|---|---|---|---|---|---|---|
| Тренировочная выборка | | | | | | |
| 0 | 0.99 | 0.97 | 0.98 | 0.98 | 0.98 | 469239 |
| 1 | 0.95 | 0.99 | 0.97 | | | 280761 |
| Тестовая выборка | | | | | | |
| 0 | 0.99 | 0.97 | 0.98 | 0.98 | 0.98 | 156325 |
| 1 | 0.95 | 0.99 | 0.97 | | | 93676 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Также это могут подтвердить метрики *accuracy* и $ROC - AUC$ (см. Рисунок 13), которые показывают идентичные результаты на обеих выборках. Это делает алгоритм градиентного бустинга наиболее предпочтительным в прогнозировании дефолта.

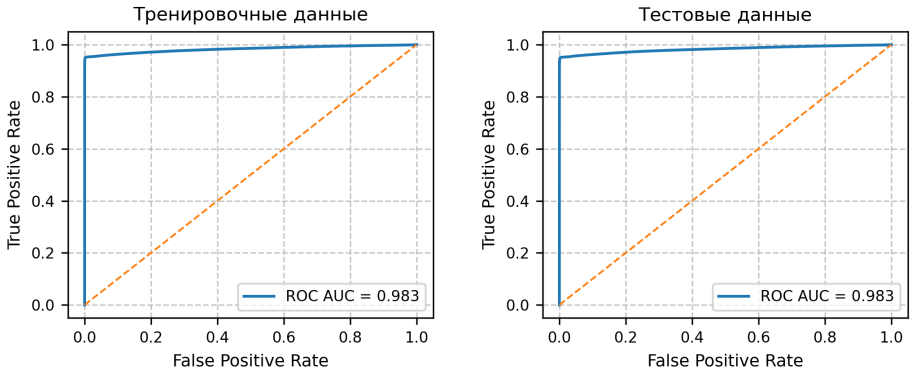


Рисунок 13 – Метрика ROC–AUC на тренировочных и тестовых данных алгоритма Gradient Boosting

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Матрица ошибок (см. Рисунок 14) показывает, что на тренировочных данных алгоритм правильно определяет 466 682 недефолтных клиентов, а 13 404 ошибочно относит к дефолтным. Верно классифицирует 267 357 дефолтных клиентов, а 2 557 неверно предсказывает как недефолтных. На тестовых данных неверно относит 4 561 недефолтных клиентов к дефолтным, а 868 дефолтных к недефолтным, что указывает на сбалансированность предсказаний алгоритма.

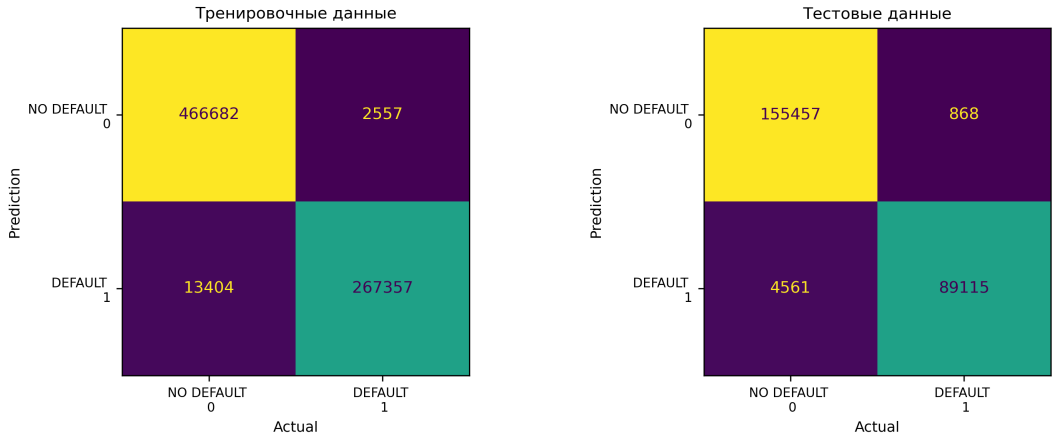


Рисунок 14 – Матрица ошибок на тренировочных данных алгоритма Gradient Boosting

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

### 1.2.4 Нейронные сети

Нейронные сети ($NeuralNetworks$) – математическая модель, позволяющая компьютерам имитировать работу человеческого мозга. Она состоит из множества простых узлов, которые образуют нейроны, сгруппированных в слои. Главными задачами нейронных сетей являются распознование скрытых закономерностей в данных и делать прогноз. В данной работе используется нейронная сеть, состоящая из трех слоев (см. Рисунок 15):

1. входной слой – состоит из 25 нейронов (равен количеству дисциплинарных признаков: $enc\_paym\_0$, ..., $enc\_paym\_24$);
2. скрытый слой – 32 нейрона;
3. выходной слой – 1 нейрон, который дает оценку вероятности ($p$) дефолта заемщика.

На вход передаются значения платежной дисциплины клиентов $enc\_paym\_0$, ..., $enc\_paym\_24$. Далее 25 признаков поступают в скрытый слой, состоящий из 64 нейронов. Каждому признаку присваивается свой вес важности и вычисляется его взвешенная сумма. К полученной сумме применяется функция активации (в данной работе

используется $ReLu$), которая «включает» нейрон, если суммарный сигнал превышает пороговое значение. В таком случае нейрон передает сигнал каждому связанному с ним нейроном в следующем слое. Если сумма ниже порогового значения, то нейрон «выключается» и данный признак является незначимым. В скрытом слое образуются 64 различные комбинации для 25 признаков и далее каждый из 64 нейронов выдает собственную оценку вероятности. В выходном слое из полученных 64 сигналов выбирается ответ с наибольшим весом. Таким образом, формируется ответ заемщику о выдаче или отказе в кредите.

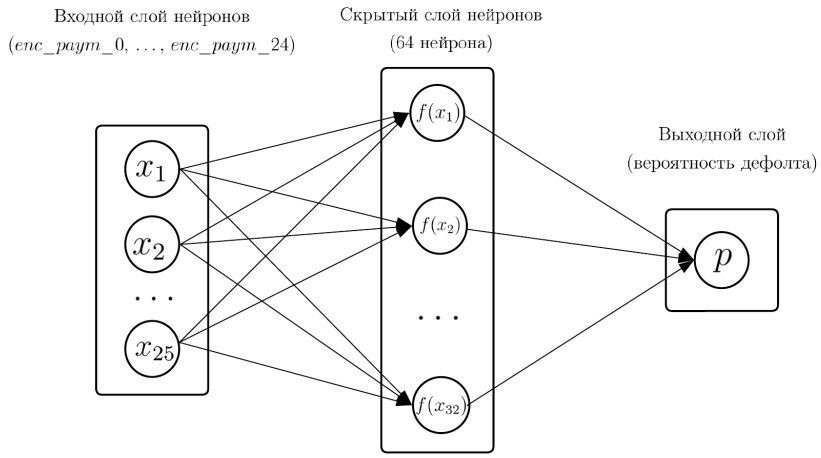


Рисунок 15 – Структура нейронной сети для оценки вероятности дефолта заемщика

Источник: составлено автором на основе программы Lucidchart (дата обращения: 25.11.2025).

Результаты нейронной сети (см. Таблицу 10), как и градиентного бустинга показывает высокую точность распознования клиентов, но также имеет выше вероятность ошибочного предсказания на дефолтных клиентах.

Таблица 10 – Метрики точности алгоритма Neural Network на 1 000 000 данных

| Flag<br>метка | Precision<br>точность | Recall<br>полнота | f1–score<br>f1–мера | Accuracy<br>точность | ROC–AUC | Количество |
|---|---|---|---|---|---|---|
| Тренировочная выборка | | | | | | |
| 0 | 1.00 | 0.97 | 0.99 | 0.98 | 0.98 | 469239 |
| 1 | 0.95 | 1.00 | 0.97 | | | 280761 |
| Тестовая выборка | | | | | | |
| 0 | 1.00 | 0.97 | 0.98 | 0.98 | 0.98 | 156325 |
| 1 | 0.95 | 1.00 | 0.97 | | | 93676 |

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Метрики *accuracy* и $ROC-AUC$ демонстрируют одинаковую высокую точность, что также относит нейронные сети к наиболее предпочтительным алгоритмам для прогнозирования дефолта заемщиков (см. Рисунок 16).

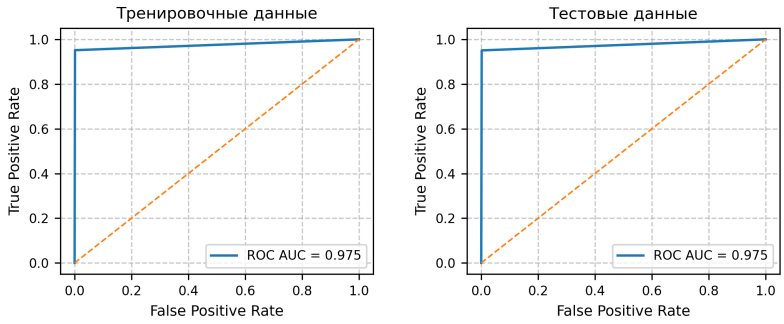


Рисунок 16 – Метрика ROC–AUC на тренировочных и тестовых данных алгоритма Neural Network

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

Матрицы ошибок на тренировочных и тестовых данных (см. Рисунок 17) показывают, что доли ошибочно отнесенных недефолтных клиентов к дефолтным относительно одинаковы и составляют около 3%. Также соотношение ошибочно предсказанных дефолтных клиентов к недефолтным на обеих выборках одинакова и приблизительно составляет 0.3%. Такое равное соотношение показывает, что нейронные сети являются одним из лучших моделей для прогнозов.

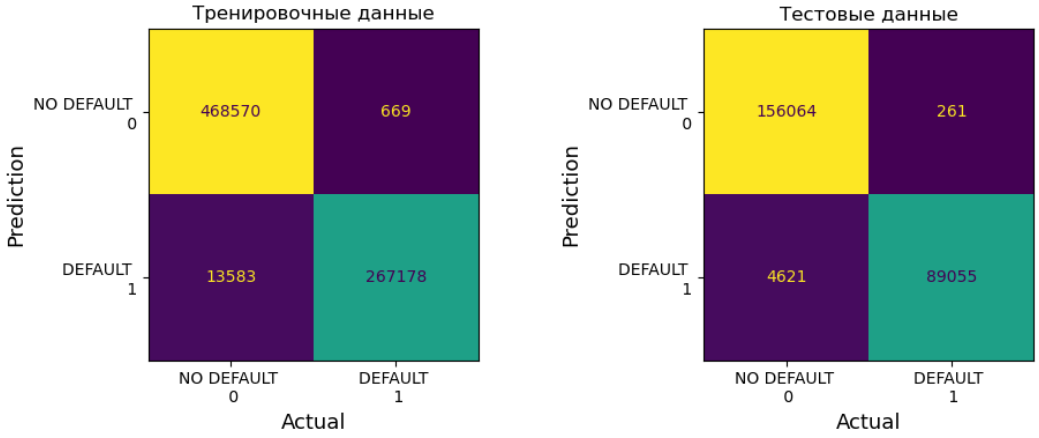


Рисунок 17 – Матрица ошибок на тренировочных и тестовых данных алгоритма Neural Network

Источник: составлено автором на основе: Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. – URL: https://ods.ai/competitions/dl-fintech-bki (дата обращения: 25.11.2025).

# ГЛАВА 2. ТЕСТИРОВАНИЕ АЛГОРИТМОВ МАШИННОГО И ГЛУБОКОГО ОБУЧЕНИЯ НА ДАННЫХ ПО КРЕДИТНОМУ СКОРИНГУ

В предыдущей главе было выяснено, что алгоритмы градиентного бустинга и нейронных сетей не имеют эффектов переобучения и недообучения. В случае градиентного бустинга используются гиперпараметры по умолчанию, результаты метрик которых не ниже метрик других алгоритмов. Это означает, что не требуется поиск гиперпараметров для данного алгоритма и можно остановиться на гиперпараметрах по умолчанию. Идентичная ситуация возникает у нейронной сети с отличием в том, что у нее метрики точности немного хуже, чем у алгоритма градиентного бустинга. Поэтому в этой главе будут подобраны гиперпараметры для алгоритмов деревьев решений и случайного леса, поскольку они обладают эффектом переобучения. Основным инструментом для поиска оптимальных гиперпараметров используются инструменты *GridSearchCV*[1] и *RandomizedSearchCV*[2]. Также в конце главы будут проанализированы прибыль и убыток банка на основании предсказаний четырех рассмотренных алгоритмов.

## 2.1 Подбор гиперпараметров для алгоритмов машинного обучения при оценке возврата кредита

Гиперпараметры – это такие параметры модели, которые задаются до начала работы модели. Правильно подобранные параметры позволяют улучшить качество предсказаний моделей. Каждому алгоритму МО присуще собственные наборы гиперпараметров, которые будут рассматриваться ниже. Существуют два метода подбора гиперпараметров:

1. *GridSearchCV* – метод подбора оптимальных гиперпараметров для модели с помощью перебора всех возможных комбинаций из заданного набора, что является ресурсозатратным способом.

2. *RandomizedSearchCV* – метод, позволяющий выбирать количество случайных комбинаций из заданного набора гиперпараметров. Является наиболее эффективным методом по времени при работе с большими данными.

[1] **GridSearchCV** (дата обращения: 25.11.2025).
[2] **RandomizedSearchC** (дата обращения: 25.11.2025).

### 2.1.1 Деревья решений

Для алгоритма классификации решающих деревьев (*Decision Tree Classifier*) основными гиперпараметрами являются:

1. *criterion* – критерий качества разбиения, определяющая показатель, по которому алгоритм лучше классифицирует клиентов.

   a) *gini* – индекс Джини[3], измеряющий неоднородность двух классов (0 и 1) внутри узла. Если индекс равен 0, значит в узле содержатся объекты одного класса. Чем индекс ближе к 0, тем лучше алгоритм определяет класс клиента. Индекс Джини в алгоритме *Decision Tree* используется по умолчанию, поскольку быстрее вычисляется;

   b) *entropy* – энтропия Шеннона[4], измеряющий степень смешанности классов. Высокая энтропия свидетельствует о высокой неоднородности классов, чем ниже энтропия, тем узел более однородный. Алгоритм выбирает разбиения с наиболее низкой энтропией, т.е. содержатся большинство объектов одного класса;

   c) *log_loss* – логарифмическая функция потерь. В данном случае алгоритм не просто выбирает класс, а оценивает их вероятности. За каждый неверный прогноз вводит штраф.

### 2.1.2 Случайный лес

### 2.1.3 Градиентный бустинг

### 2.1.4 Нейронная сеть

## 2.2 Ограничения и возможности моделей

[3]Джини К. (1884-1965 гг.) – итальянский статистик, экономист, социолог и демограф.

[4]Шеннон К.Э. (1916-2001 гг.) – американский инженер, криптоаналитик и математик, заложил основы теории информации.

# Приложение 1. Предобработка данных

Листинг 1 – Формирование единого датасета

```
import pandas as pd
import os
import pyarrow.parquet as pq

path = "train_data"
for i,file in enumerate(os.listdir(path)):
    file = 'train_data_' + str(i) + '.pq'
    dataset = pq.ParquetDataset(os.path.join(path,file))
    df = dataset.read(use_threads=True).to_pandas()
    df_gr = df.groupby('id').agg('mean')
    file_csv = file.replace('.pq', '.csv')
    df_gr.to_csv(os.path.join('train_data_csv_all',file_csv))
os.listdir(path)

path = 'train_data_csv_all'
frames = []
for file_csv in os.listdir(path):
    if file_csv.endswith('.csv'):
        df = pd.read_csv(os.path.join(path, file_csv))
        frames.append(df)

result = pd.concat(frames)
file_csv_all = '1_data_csv_all.csv'
result.to_csv(file_csv_all)
df_all = pd.read_csv(file_csv_all)
```

Листинг 2 – Датасет, содержащий 41 признак

```
from sklearn.decomposition import PCA

file_csv = 'data_csv_small.csv'
df = pd.read_csv(file_csv)

columns = ['pre_pterm', 'pre_fterm',
           'pre_loans_next_pay_summ','pre_loans_outstanding',
           'pre_loans_total_overdue', 'pre_loans_max_overdue_sum',
           'pre_loans_credit_cost_rate', 'is_zero_loans5',
           'is_zero_loans530', 'is_zero_loans3060',
           'is_zero_loans6090', 'is_zero_loans90',
           'pre_util','pre_maxover2limit',
           'is_zero_util', 'is_zero_over2limit',
           'enc_paym_0', 'enc_paym_1', 'enc_paym_2',
           'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
           'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
           'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
           'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
           'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
           'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
           'enc_paym_21', 'enc_paym_22',
           'enc_paym_23', 'enc_paym_24']

X = df.loc[:, columns].copy()
pca = PCA()
pca.fit(X)

100*pca.explained_variance_ratio_.round(3)
```

Листинг 3 – Датасет, содержащий 24 признака

```
file_csv = 'data_csv_small.csv'
df = pd.read_csv(file_csv)

columns_pay = ['enc_paym_0', 'enc_paym_1', 'enc_paym_2',
               'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
               'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
               'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
               'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
               'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
               'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
               'enc_paym_21', 'enc_paym_22', 'enc_paym_23',
               'enc_paym_24']

X_pay = df.loc[:, columns_pay].copy()
pca = PCA()
pca.fit(X_pay)

100*pca.explained_variance_ratio_.round(3)
```

Листинг 4 – Алгоритм выявления аномальных клиентов

```
df_y = pd.read_csv('train_target.csv')
y  = df_y.flag.values[:1000000].copy()
y_ = df_y.flag.values[:1000000].copy()

counter_norm, counter_anom = 0, 0
ind_norm, ind_anom = [], []
for i,x_pay in enumerate(np.array(X_pay)):
    if y_[i]==0 and min(x_pay)>0:
        counter_anom +=1
        ind_anom.append(i)
        y_[i]=1
    else:
        counter_norm +=1
        ind_norm.append(i)
```

Листинг 5 – Критерий Колмогорова-Смирнова

```
col = 'enc_paym_15'
T = X_pay.enc_paym_15
x00, x01 = T[(y==0) & (y_== 0)], T[ind_anom]
x11, x01 = T[(y==1)], T[ind_anom]
ind00 = (x00!=0) & (x00!=1) & (x00!=2) & (x00!=3) & (x00!=4) & (x00
    !=5)
ind01 = (x01!=0) & (x01!=1) & (x01!=2) & (x01!=3) & (x01!=4) & (x01
    !=5)
ind11 = (x11!=0) & (x11!=1) & (x11!=2) & (x11!=3) & (x11!=4) & (x11
    !=5)
x00, x11, x01 = x00[ind00], x11[ind11], x01[ind01]

def my_cdf(x1, x2, step):
    tm = max(max(x1), max(x2))
    F1,F2 = [], []
    n1, n2 = x1[x1>0].shape[0], x2[x2>0].shape[0]
    for t in np.arange(0, tm+0.1, step):
        F1.append(x1[(x1>0)&(x1<t)].shape[0])
        F2.append(x2[(x2>0)&(x2<t)].shape[0])
    F1, F2 = np.array(F1)/n1, np.array(F2)/n2
    K = max(abs(F1-F2))*np.sqrt( n1*n2/(n1+n2))
    return F1, F2, K, tm
step = 0.01
F01,  F11, K,  tm = my_cdf(x01, x11, step)
F01_, F00, K_, tm = my_cdf(x01, x00, step)
K, K_
```

Листинг 6 – Тест Фишера на равенство коэффициентов корреляции выборочной и генеральной совокупностей

```
def transform_f(r, R, n):
    if n > 3:
        return 0.5*np.log(((1+r)/(1-r))*((1-R)/(1+R)))*np.sqrt(n-3)
    else:
        return 0

mth = 17
n_all, n_nod, n_def, n_ano = X_pay.shape[0], len(ind_nod), len(
   ind_def), len(ind_ano)
R_all, r_nod, r_def, r_ano = all_c[mth], nod_c[mth], def_c[mth],
   ano_c[mth]
transform_f(r_nod, R_all, n_nod), transform_f(r_def, R_all, n_def ),
   transform_f(r_ano, R_all, n_ano )
```

# Приложение 2