

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

Введение . . . . .	3
Глава 1. Кредитный риск и роль искусственного интеллекта в его оценке . . . .	5
1.1 Сущность и классификация кредитных рисков банков второго уровня .	5
1.2 Подходы к оценке и прогнозированию кредитного риска: классические модели и современные методы искусственного интеллекта . . . . .	5
Глава 2. Алгоритмы машинного и глубокого обучения в задаче о кредитном скоринге . . . . .	6
2.1 Предобработка данных кредитной истории клиентов банка . . . . .	6
2.1.1 Метод главных компонент . . . . .	8
2.1.2 Определение аномальности клиента . . . . .	11
2.2 Базовые алгоритмы оценки вероятности возврата кредита . . . . .	15
2.2.1 Деревья решений . . . . .	15
2.2.2 Случайный лес . . . . .	15
2.2.3 Градиентный бустинг . . . . .	15
2.2.4 Нейронные сети . . . . .	15
Глава 3. Тестирование алгоритмов машинного и глубокого обучения на данных по кредитному скорингу . . . . .	16
3.1 Подбор гиперпараметров для алгоритмов машинного и глубокого обучения при оценке возврата кредита . . . . .	16
3.1.1 Деревья решений . . . . .	16
3.1.2 Случайный лес . . . . .	16
3.1.3 Градиентный бустинг . . . . .	16
3.1.4 Нейронная сеть . . . . .	16
3.2 Ограничения и возможности моделей . . . . .	16
Заключение . . . . .	16
Список литературы . . . . .	17
Приложение А. Предобработка данных . . . . .	18

## Введение

**Актуальность темы исследования.** В настоящее время, в условиях технологического развития, искусственный интеллект (далее ИИ) стал необходимым инструментом, позволяющим с высокой скоростью обрабатывать большие массивы данных. Для банковского сектора управление кредитными рисками является одной из ключевых задач, в частности оценка кредитоспособности заемщиков. В современном финансовом секторе Казахстана использование кредитного скоринга поднимает важный вопрос о справедливом доступе к потребительскому кредитованию. Кредитный скоринг – это статистический процесс, позволяющий прогнозировать вероятность дефолта заемщика банка. Значимость кредитного скоринга для Казахстана определяется его ролью в снижении долговой нагрузки населения, ограничении массового заимствования и обеспечении устойчивого и эффективного развития финансового сектора.

С развитием услуг микрокредитования и предоставления кредитов в рассрочку банки стали получать значительно больше заявок на кредит, что на первоначальном этапе было выгодно финансовым учреждениям. Однако по мере роста количества заявок анализ разнообразных кредитных историй клиентов стал сложной задачей для сотрудников банка, что приводило к снижению прибыли. В связи с этим возникает необходимость в автоматизации и оптимизации процесса оценки и выдачи кредитов с помощью технологий ИИ, что позволяет минимизировать кредитные риски для обеих сторон. Практическая значимость работы заключается в том, что результаты исследования могут быть использованы финансовыми организациями для повышения эффективности принятия кредитных решений. Применение алгоритмов машинного и глубокого обучения позволит банкам принимать более точные и обоснованные решения о кредитоспособности клиентов, снижая вероятность возникновения финансовых рисков.

**Степень разработанности темы исследования.** Применение искусственного интеллекта в банковском кредитовании рассматривались в работах Н. Sadok, F. Sakka и М.Е. El Maknoui, в магистерской диссертации Ш. Сяюй, а также в исследованиях Г.З. Зиятбековой, А.А. Давыдовой и О.Л. Ксенофонтовой, посвященных использованию методов машинного обучения и интеллектуального анализа данных в банковской сфере. Данные работы свидетельствуют об устойчивом научном интересе внедрения ИИ в деятельность финансовых учреждений. Тем не менее для Казахстана данное направление является новым в контексте цифровизации финансового сектора и, следовательно, требует дальнейшего исследования.

**Цель и задачи исследования** – разработать метод прогнозирования дефолта клиента банка при выдаче кредита на основе методов машинного и глубокого обучения. Для достижения поставленной цели были определены следующие задачи:

1. Провести анализ и предобработку данных о заемщиках.
2. Обучить и протестировать модели классификации для прогнозирования дефолта.

3. Подобрать гиперпараметры моделей для повышения качества прогнозирования дефолтных клиентов.
4. Оценить качество моделей и проанализировать полученные результаты.

**Объектом исследования** является выданный кредит банком и связанное с ним наступление либо отсутствие дефолта по выданным кредитам.

**Предметом исследования** является кредитный скоринг для оценки риска невозврата кредита.

Для достижения целей и задач исследования использовались следующие методы:

- машинного и глубокого обучения;
- статистического анализа;
- оценивания моделей;
- визуализации исторических данных.

**Научная новизна исследования** исследования заключается в следующем:

- после завершения работы добавим

**Информационной базой исследования** являлись платформы: kaggle, github, stackoverflow, нормативно-правовая база, научные статьи и монографии.

**Структура и объем работы.** Выпускная квалификационная работа состоит из 3 глав, заключения, списка литературы, X таблиц, Y рисунков и Z приложений.

**Объем исследовательской работы:** 50 страниц.

# **ГЛАВА 1. КРЕДИТНЫЙ РИСК И РОЛЬ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В ЕГО ОЦЕНКЕ**

- 1.1 Сущность и классификация кредитных рисков банков  
второго уровня**
- 1.2 Подходы к оценке и прогнозированию кредитного риска:  
классические модели и современные методы  
искусственного интеллекта**

# ГЛАВА 2. АЛГОРИТМЫ МАШИННОГО И ГЛУБОКОГО ОБУЧЕНИЯ В ЗАДАЧЕ О КРЕДИТНОМ СКОРИНГЕ

## 2.1 Предобработка данных кредитной истории клиентов банка

В данной работе используются обезличенные данные АО «Альфа-Банка» [7]. 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 А).

**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`.

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	0
1	2	0	1	0	1	1	2	
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.

### 2.1.1 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[8], 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[2] 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:

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 А) 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	Ассурасу точность	ROC AUC	Количество
Тренировочная выборка						
0	1.00	0.97	0.98	0.97	0.51	724588
1	0.00	0.00	0.00			25412
Тестовая выборка						
0	1.00	0.97	0.98	0.97	0.50	241569
1	0.00	0.00	0.00			8431

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*:

$$\begin{aligned}
\lambda_1^{(p)} &= 22.1, \lambda_2^{(p)} = 20.3, \lambda_3^{(p)} = 17.9, \lambda_4^{(p)} = 10.1, \lambda_5^{(p)} = 9.7, \\
\lambda_6^{(p)} &= 7.7, \lambda_7^{(p)} = 4.4, \lambda_8^{(p)} = 1.7, \lambda_9^{(p)} = 1.5, \lambda_{10}^{(p)} = 0.9, \\
\lambda_{11}^{(p)} &= 0.6, \lambda_{12}^{(p)} = 0.5, \lambda_{13}^{(p)} = 0.4, \lambda_{14}^{(p)} = 0.3, \lambda_{15}^{(p)} = 0.2, \\
\lambda_{16}^{(p)} &= 0.2, \lambda_{17}^{(p)} = 0.2, \lambda_{18}^{(p)} = 0.1, \lambda_{19}^{(p)} = 0.1, \lambda_{20}^{(p)} = 0.1, \\
\lambda_{21}^{(p)} &= 0.1, \lambda_{22}^{(p)} = 0.1, \lambda_{23}^{(p)} = 0.1, \lambda_{24}^{(p)} = 0.1, \lambda_{25}^{(p)} = 0.1, \\
\lambda_{26}^{(p)} &= 0.1, \lambda_{27}^{(p)} = 0.1, \lambda_{28}^{(p)} = 0.1, \lambda_{29}^{(p)} = 0.1, \lambda_{30}^{(p)} = 0.1, \\
\lambda_{31}^{(p)} &= 0.0, \lambda_{32}^{(p)} = 0.0, \lambda_{33}^{(p)} = 0.0, \lambda_{34}^{(p)} = 0.0, \lambda_{35}^{(p)} = 0.0, \\
\lambda_{36}^{(p)} &= 0.0, \lambda_{37}^{(p)} = 0.0, \lambda_{38}^{(p)} = 0.0, \lambda_{39}^{(p)} = 0.0, \lambda_{40}^{(p)} = 0.0, \\
\lambda_{41}^{(p)} &= 0.0
\end{aligned} \tag{1}$$

В формуле (1)  $\lambda_1^{(p)}, \dots, \lambda_{41}^{(p)}$  – 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

А) 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`,  $k = 0, \dots, 24$ .
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`.

$$\begin{aligned} \lambda_1^{(p)} &= 65.5, \lambda_2^{(p)} = 16.0, \lambda_3^{(p)} = 5.7, \lambda_4^{(p)} = 3.0, \lambda_5^{(p)} = 1.9, \\ \lambda_6^{(p)} &= 1.3, \lambda_7^{(p)} = 1.0, \lambda_8^{(p)} = 0.8, \lambda_9^{(p)} = 0.7, \lambda_{10}^{(p)} = 0.6, \\ \lambda_{11}^{(p)} &= 0.5, \lambda_{12}^{(p)} = 0.4, \lambda_{13}^{(p)} = 0.4, \lambda_{14}^{(p)} = 0.3, \lambda_{15}^{(p)} = 0.3, \\ \lambda_{16}^{(p)} &= 0.3, \lambda_{17}^{(p)} = 0.3, \lambda_{18}^{(p)} = 0.2, \lambda_{19}^{(p)} = 0.2, \lambda_{20}^{(p)} = 0.2, \\ \lambda_{21}^{(p)} &= 0.1, \lambda_{22}^{(p)} = 0.1, \lambda_{23}^{(p)} = 0.1, \lambda_{24}^{(p)} = 0.1, \lambda_{25}^{(p)} = 0.1. \end{aligned} \tag{2}$$

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. На первый взгляд метод PCA не оказал влияния на некоторые метрики алгоритма, поэтому требуется дальнейший детальный анализ.

### 2.1.2 Определение аномальности клиента

Рисунок 1 показывает, что доля недефолтных клиентов существенно велика и не соответствует статистике в реальной жизни. По данным Первого кредитного бюро около 20% [6] казахстанских заемщиков имеют дефолтную кредитную историю, что подтверждает данное утверждение и указывает на наличие аномальности в классе недефолтных клиентов.

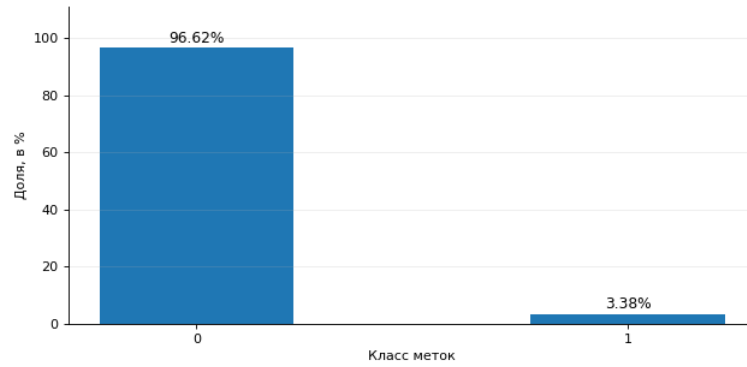


Рисунок 1 – Перекос в сторону 0

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 работе под аномальностью клиента 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 назовем такого клиента, для которого выполняются два условия:

$$\begin{cases} (a) \text{ метка (флаг)} = 0, \\ (b) \min(enc\_paym\_k) > 0, \quad k = 0, \dots, 24. \end{cases} \quad (anom)$$

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 А) 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*.

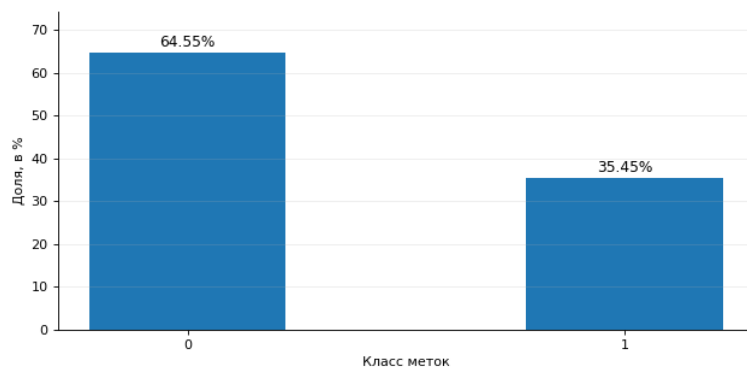


Рисунок 2 – Переход клиентов из класса «0» в класс «1»

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%).

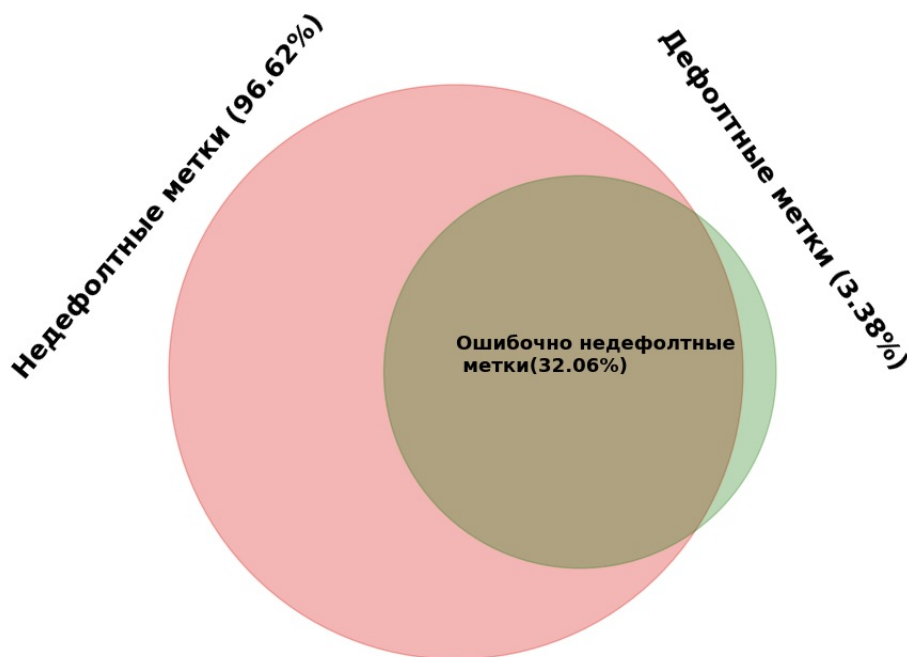


Рисунок 3 – Пересечение дефолтных и недефолтных меток с выделением ошибочно недефолтных клиентов

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).

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 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.

Таблица 4 – Изменение точности метрик

Алгоритм – Случайный лес						
Flag метка	Precision точность	Recall полнота	f1-score f1-мера	Accuracy точность	ROC AUC	Количество
Тренировочная выборка						
0	0.99	0.97	0.98	0.97	0.97	483921
1	0.94	0.99	0.96			266079
Тестовая выборка						
0	0.99	0.97	0.98	0.97	0.97	161593
1	0.94	0.99	0.96			88407

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).

## 2.2 Базовые алгоритмы оценки вероятности возврата кредита

### 2.2.1 Деревья решений

### 2.2.2 Случайный лес

### 2.2.3 Градиентный бустинг

### 2.2.4 Нейронные сети

# **ГЛАВА 3. ТЕСТИРОВАНИЕ АЛГОРИТМОВ МАШИННОГО И ГЛУБОКОГО ОБУЧЕНИЯ НА ДАННЫХ ПО КРЕДИТНОМУ СКОРИНГУ**

## **3.1 Подбор гиперпараметров для алгоритмов машинного и глубокого обучения при оценке возврата кредита**

### **3.1.1 Деревья решений**

### **3.1.2 Случайный лес**

### **3.1.3 Градиентный бустинг**

### **3.1.4 Нейронная сеть**

## **3.2 Ограничения и возможности моделей**

## **Заключение**



# СПИСОК ЛИТЕРАТУРЫ

## Статьи:

1. Давыдова, А.А. Анализ и оценка кредитоспособности заемщика финансовыми организациями: обзор подходов и методов оценки // Вестник ВУиТ. – 2023. – №1 (51). URL: <https://cyberleninka.ru/article/n/analiz-i-otsenka-kreditosposobnosti-zaemshika-finansovymi-organizatsiyami-obzor-p> – Загл. с экрана.
2. Смирнов С.В. Методы машинного обучения в макроэкономическом прогнозировании: предварительные итоги. Вопросы экономики. 2025;(10):131-154. <https://doi.org/10.32609/0042-8736-2025-10-131-154>. – Загл. с экрана.
3. Сяюй, Ш. Разработка моделей кредитного скоринга заёмщиков коммерческих банков с использованием методов машинного обучения: магистерская диссертация // БГУ, Факультет прикладной математики и информатики, Кафедра математического моделирования и анализа данных. URL: <https://elib.bsu.by/handle/123456789/331793>. – Загл. с экрана.
4. Provenzano A. R. Machine learning approach for credit scoring / A. R. Provenzano, D. Trifirò, A. Datteo, L. Giada, N. Jean, A. Riciputi, et al. arXiv:2008.01687. – 2020. – С. 1 – 28.
5. Sadok, H. Artificial intelligence and bank credit analysis: A review / Sakka, F., El Maknoui, M.E. // Cogent Economics & Finance. – 2025. – С. 1 – 13.

## Интернет-ресурсы:

6. Можно ли “отбелить” кредитную историю и есть ли черный список должников в Казахстане [Электронный ресурс] / Информационный интернет-портал. URL: [https://tengrinews.kz/kazakhstan\\_news/li-otbelit-kreditnuyu-istoriyu-est-chernyy-spisok-doljnikov-560869/](https://tengrinews.kz/kazakhstan_news/li-otbelit-kreditnuyu-istoriyu-est-chernyy-spisok-doljnikov-560869/). – Загл. с экрана.
7. Соревнование на данных кредитных историй [Электронный ресурс] / Open Data Science. URL: <https://ods.ai/competitions/dl-fintech-bki>. – Загл. с экрана.
8. PCA [Электронный ресурс] / Python-библиотека для машинного обучения. – URL: <https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>. – Загл. с экрана.

## Приложение А. Предобработка данных

Листинг 1 – Формирование единого датасета

```
1 import pandas as pd
2 import os
3 import pyarrow.parquet as pq
4
5 path = "train_data"
6 for i,file in enumerate(os.listdir(path)):
7     file = 'train_data_' + str(i) + '.pq'
8     dataset = pq.ParquetDataset(os.path.join(path,file))
9     df = dataset.read(use_threads=True).to_pandas()
10    df_gr = df.groupby('id').agg('mean')
11    file_csv = file.replace('.pq', '.csv')
12    df_gr.to_csv(os.path.join('train_data_csv_all',file_csv))
13 os.listdir(path)
14
15 path = 'train_data_csv_all'
16 frames = []
17 for file_csv in os.listdir(path):
18     if file_csv.endswith('.csv'):
19         df = pd.read_csv(os.path.join(path, file_csv))
20         frames.append(df)
21
22 result = pd.concat(frames)
23 file_csv_all = '1_data_csv_all.csv'
24 result.to_csv(file_csv_all)
25 df_all = pd.read_csv(file_csv_all)
```

Листинг 2 – Датасет, содержащий 41 признак

```
1 from sklearn.decomposition import PCA
2
3 file_csv = 'data_csv_small.csv'
4 df = pd.read_csv(file_csv)
5
6 columns = ['pre_pterm', 'pre_fterm',
7            'pre_loans_next_pay_summ', 'pre_loans_outstanding',
8            'pre_loans_total_overdue', 'pre_loans_max_overdue_sum',
9            'pre_loans_credit_cost_rate', 'is_zero_loans5',
10           'is_zero_loans530', 'is_zero_loans3060',
11           'is_zero_loans6090', 'is_zero_loans90',
12           'pre_util', 'pre_maxover2limit',
13           'is_zero_util', 'is_zero_over2limit',
14           'enc_paym_0', 'enc_paym_1', 'enc_paym_2',
15           'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
16           'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
17           'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
18           'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
19           'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
20           'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
21           'enc_paym_21', 'enc_paym_22',
22           'enc_paym_23', 'enc_paym_24']
23
24 X = df.loc[:, columns].copy()
25 pca = PCA()
26 pca.fit(X)
27
28 100*pca.explained_variance_ratio_.round(3)
```

### Листинг 3 – Датасет, содержащий 24 признака

```
1 file_csv = 'data_csv_small.csv'
2 df = pd.read_csv(file_csv)
3
4 columns_pay = ['enc_paym_0', 'enc_paym_1', 'enc_paym_2',
5               'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
6               'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
7               'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
8               'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
9               'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
10              'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
11              'enc_paym_21', 'enc_paym_22', 'enc_paym_23',
12              'enc_paym_24']
13
14 X_pay = df.loc[:, columns_pay].copy()
15 pca = PCA()
16 pca.fit(X_pay)
17
18 100*pca.explained_variance_ratio_.round(3)
```

### Листинг 4 – Алгоритм выявления аномальных клиентов

```
1 df_y = pd.read_csv('train_target.csv')
2 y = df_y.flag.values[:1000000].copy()
3 y_ = df_y.flag.values[:1000000].copy()
4
5 counter_norm, counter_anom = 0, 0
6 ind_norm, ind_anom = [], []
7 for i, x_pay in enumerate(np.array(X_pay)):
8     if y_[i]==0 and min(x_pay)>0:
9         counter_anom +=1
10        ind_anom.append(i)
11        y_[i]=1
12    else:
13        counter_norm +=1
14        ind_norm.append(i)
```

## Приложение Б