Разработка системы оценки заемщика для автоматизации работы с кредитными продуктами

1. Постановка задачи

Небольшой банк расширяется и намерен выдавать более крупные кредиты. В целях - повысить качество управления рисками при работе с кредитными продуктами.

Начать достигать цель заказчик планирует со следующих общих задач:

- разработать предиктивную модель собственного скоринга банка (с учетом предпочитаемых условий по признаку профессии заемщика)
- составить портрет идеального заемщика (для автоматизации в дальнейшем превентивных предложений кредитов по выгодным условиям)

Требования стейкхолдера:

- Решение задачи на языке Python, так как аналитикам данной организации данный язык более знаком.
- Условимся, что все данные датасета - история перед моментом взятия кредита.
- Группы кредитного рейтинга должны интерпретироваться следующим образом: Poor: [0.0; 0.3], Standard: (0.3; 0.7), Good: [0.7; 1.0].
- В скоринговой модели должен учитываться риск по профессии: чем он выше, тем ниже скоринговый балл (данные о банковском рейтинге профессий будут предоставлены отдельно).
- Одним из экспертно отобранных признаков для построения модели должно быть значение “размер ежемесячного платежа/размер общих трат за месяц”
- модель должна быть высоко интерпретируемой

Данные для расчета:
https://www.kaggle.com/datasets/parisrohan/credit-score-classification?select=train.csv

Предположения по решению:

При решении задачи скоринга банк фактически получает вероятностную оценку, которая описывается чистой вероятностью или некоторым производным от нее скоринговым баллом. На основе этого балла устанавливается порог, по которому отсекаются добросовестные заемщики от недобросовестных.

Можно рассмотреть задачу как задачу классификации. При этом предсказание модели будет скорректировано с учетом оценки значения признака (профессия). Таким образом, будет разработана специфичная модель собственного скоринга банка.

Идеальные заемщики будут иметь максимальный скоринговый балл, предсказанный моделью. Таким образом, его портрет можно будет составить из всех признаков датасета, а не только из определенных экспертно.

Метрики, которые будут использованы для текущих задач:

- скоринговая метрика - вероятность добросовестности заемщика (является предсказанием модели)
- метрики качества модели

2. Предварительный анализ и очистка датасета

Всего в полном датасете 150000 записей, и у 50000 из них отсутствует группа кредитного рейтинга. При этом, в датасете 12500 заемщиков, записи о которых, очевидно, обновлялись каждый месяц (12500 × 12 = 150000). Таким образом, в датасете большинство записей - практически дубликаты, не считая столбца с возрастом кредитной истории. Идентификатор Customer_ID соответствует одному заемщику. Если две записи имеют одинаковый Customer_ID, будем считать, что это одна и та же запись. Дубликаты будут удалены после работы с пропусками.

Методами unique и value_counts проверены все столбцы-категории, обнаруженные аномалии и символы (Таблица 1), используемые для пропусков, заменены на пустые значения. Лишние символы в корректных значениях удалены. Текст в столбце Credit_History_Age переведен в число месяцев. Для дальнейшего анализа выбросов и удобства других операций столбцам заданы подходящие типы данных, которые поддерживают пустые значения. Также, заменены на пустые значения неправдоподобные большие значения в целочисленных столбцах. Для анализа выбросов проанализированы распределения значений всех столбцов (Рис. 1, 2.).

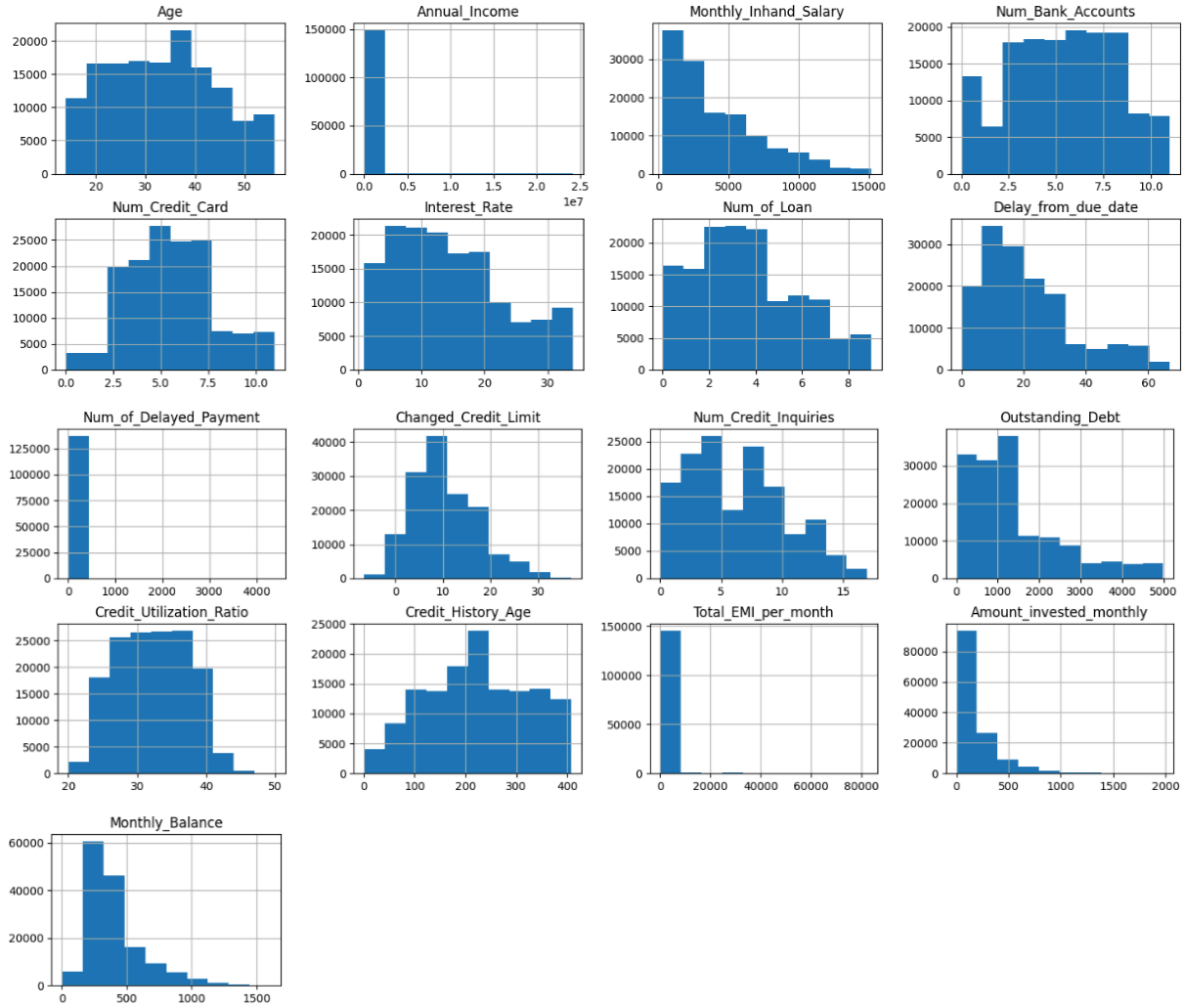

Рис. 1 - Распределения числовых значений датасета

Таблица 1 - Описание столбцов исходного датасета и их качества

Номер столбца	**Имя столбца**	**Описание**	**Качество**
0	ID	Represents a unique identification of an entry	
1	Customer_ID	Represents a unique identification of a person	
2	Month	Represents the month of the year	
3	Name	Represents the name of a person	В начале имени встречаются лишние запятые с пробелами и кавычки (“, ” и “”””), большое количество пустых записей
4	Age	Represents the age of the person	Неправдоподобные для возраста большие значения, отрицательное “-500”, лишние “_” в строках с корректным значением
5	SSN	Represents the social security number of a person	Пропуски в виде значения “#F%$D@*&8”
6	Occupation	Represents the occupation of the person	Пропуски в виде нижних подчеркиваний (“_______”)
7	Annual_Income	Represents the annual income of the person	Лишние “_” в конце записей
8	Monthly_Inhand_Salary	Represents the monthly base salary of a person	Пустые значения
9	Num_Bank_Accounts	Represents the number of bank accounts a person holds	Пропуски в виде “-1” и неправдоподобные большие значения

10	Num_Credit_Card	Represents the number of other credit cards held by a person	Неправдоподобные большие значения
11	Interest_Rate	Represents the interest rate on credit card	Неправдоподобные большие значения
12	Num_of_Loan	Represents the number of loans taken from the bank	Лишние “_” в конце записей, отрицательное “-100” и неправдоподобные большие значения
13	Type_of_Loan	Represents the types of loan taken by a person	Пустые значения
14	Delay_from_due_date	Represents the average number of days delayed from the payment date	Отрицательные числа от -5 до -1
15	Num_of_Delayed_Payment	Represents the average number of payments delayed by a person	Отрицательные числа от -1 до -3, пропущенные “-1_”, “-2_”, “-3_” и лишние “_” в конце корректных записей
16	Changed_Credit_Limit	Represents the percentage change in credit card limit	Значения с лишним ведущим нулем и пропущенные значения “_”
17	Num_Credit_Inquiries	Represents the number of credit card inquiries	Пустые значения и неправдоподобные большие значения
18	Credit_Mix	Represents the classification of the mix of credits	Пропущенные значения “_”
19	Outstanding_Debt	Represents the remaining debt to be paid (in USD)	Лишние “_” в конце записей
20	Credit_Utilization_Ratio	Represents the utilization ratio of credit card	

21	Credit_History_Age	Represents the age of credit history of the person	Пропущенные значения в виде “NA”
22	Payment_of_Min_Amount	Represents whether only the minimum amount was paid by the person	У достаточно большого количества строк есть выделяющееся значение “NM”, но я буду считать, что это пропуск, т. к. оно не встречается у одного и того же человека по несколько раз, а равномерно распределено по всем заемщикам
23	Total_EMI_per_month	Represents the monthly EMI payments (in USD)	
24	Amount_invested_monthly	Represents the monthly amount invested by the customer (in USD)	Пустые значения и пропущенные значения в виде “__10000__”
25	Payment_Behaviour	Represents the payment behavior of the customer (in USD)	Пропущенные значения “!@9#%8”
26	Monthly_Balance	Represents the monthly balance amount of the customer (in USD)	Пустые значения и пропущенные значения “__-333333333333333333333333333__”
27	Credit_Score	Represents the bracket of credit score (Poor, Standard, Good)	

В Age, Num_Bank_Accounts, Num_Credit_Card, Credit_Utilization_Ratio, Credit_History_Age выбросов не найдено. В Annual_Income не удалось найти такую верхнюю границу, чтобы выбросы были очевидны. По смыслу это значение может быть не ограничено, а по графику видно, что распределение очень неравномерное. В Outstanding_Debt, Num_Credit_Inquiries распределение имеет некоторую закономерность. При этом если некоторое значение велико, то такое значение указано во всех строках для одного Customer_ID. Пока не буду считать их за выбросы. В Monthly_Inhand_Salary, Interest_Rate, Num_of_Loan считаю, что выбросов нет. В Delay_from_due_date и Num_of_Delayed_Payment, Changed_Credit_Limit, Amount_invested_monthly, Monthly_Balance есть значения, которые можно признать выбросами. Эти значения нужно будет обработать, т. к. значение может быть в том числе ошибкой. В Total_EMI_per_month распределение самое неравномерное. Но есть явные выбросы, которых много и, возможно, их стоит удалить.

Категории, являющиеся идентификаторами не рассмотрены, т .к. не будут использоваться как признаки в модели. Также исключена из анализа колонка Type_of_Loan, т. к. содержит перечисления через запятую в различном порядке типов кредитов. Такую строку сложно преобразовать в имеющую смысл, при этом, как видно, большинство значений в ней - 'Not Specified', в чем мало смысла первоначально. Плюс, в датасете уже есть схожая по смыслу колонка с числовым значением - Num_of_Loan (количество кредитов), совпадающая с Type_of_Loan по числу перечисленных в Type_of_Loan типов. Диаграмма показывает, что наиболее сбалансированные признаки - это Occupation и Payment_Behaviour (Рис. 2).

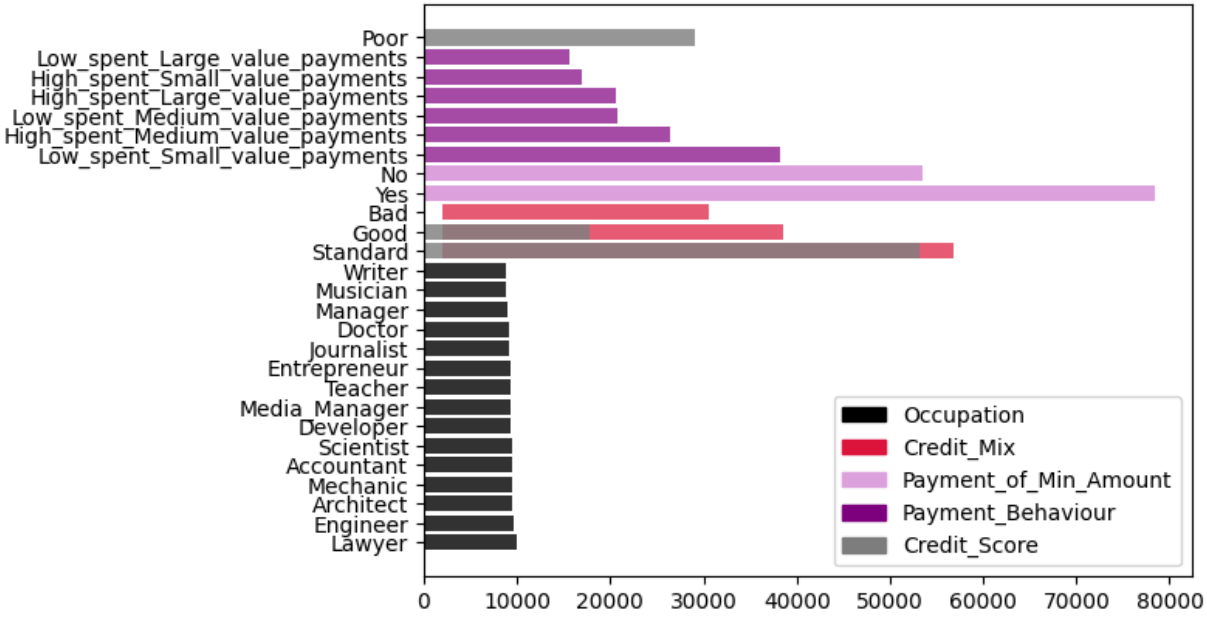

Рис. 2 - Распределения категориальных признаков датасета

По визуализации полноты данных (Рис. 3) можно сделать вывод, что удаление строк с пропущенными значениями целиком не подойдет в данной ситуации (пропуски - скорее в шахматном порядке, относительно, по крайней мере, соседних колонок).

Пропуски будут устраняться для каждого признака с помощью группировки по заемщику и заменой на медиану или моду.

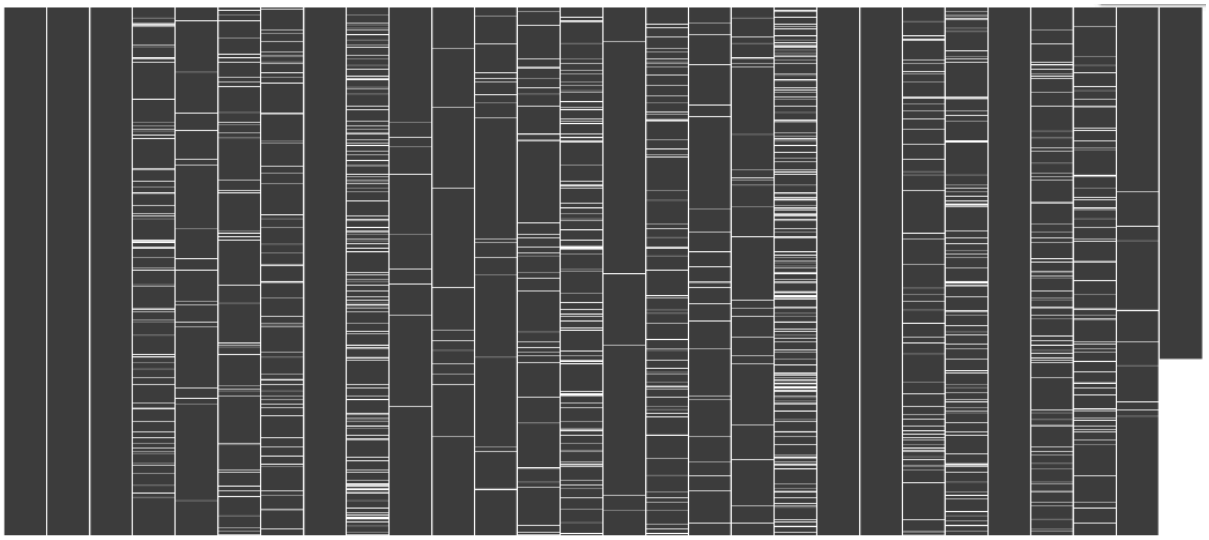

Рис. 3 - Визуализация полноты данных исходного датасета

В целом, качество данных датасета относительно низкое: кроме пропусков можно найти множество фактических ошибок. Так, например, у заемщиков 14 лет указано несколько кредитов в прошлом и такие профессии как бухгалтер, механик, доктор.

После работы с пропусками проведено сравнение статистик в исходном датасете и в улучшенном - статистики практически не изменились.

Далее проведен корреляционный анализ признаков - пока нельзя сказать, что какие-то из них сильно коррелируют, возможно, это будет очевидно после работы с выбросами. Но пока можно уже выделить Monthly_Inhand_Salary и Monthly_Balance.

Ноутбук для диплома.ipynb

3. Подготовка данных для машинного обучения

При работе с выбросами учитывалось распределение значений в столбце и в соседних столбцах. Подозрение на выбросы-ошибки, явные выбросы или распределение с длинным "хвостом" были в Delay_from_due_date, Num_of_Delayed_Payment, Changed_Credit_Limit, Amount_invested_monthly, Monthly_Balance, Total_EMI_per_month и Annual_Income. В столбце Delay_from_due_date, где выбросы были признаны ошибкой, значение заменено медианой, чтобы оказать минимум влияния на исходное распределение. А в остальных столбцах, где есть слишком длинные "хвосты" в распределении (или сомнения в ошибке) - значения в конце "хвоста" приравнено ближайшей границе нужного диапазона [1]. В результате форма распределений стала более явной (Рис. 4).

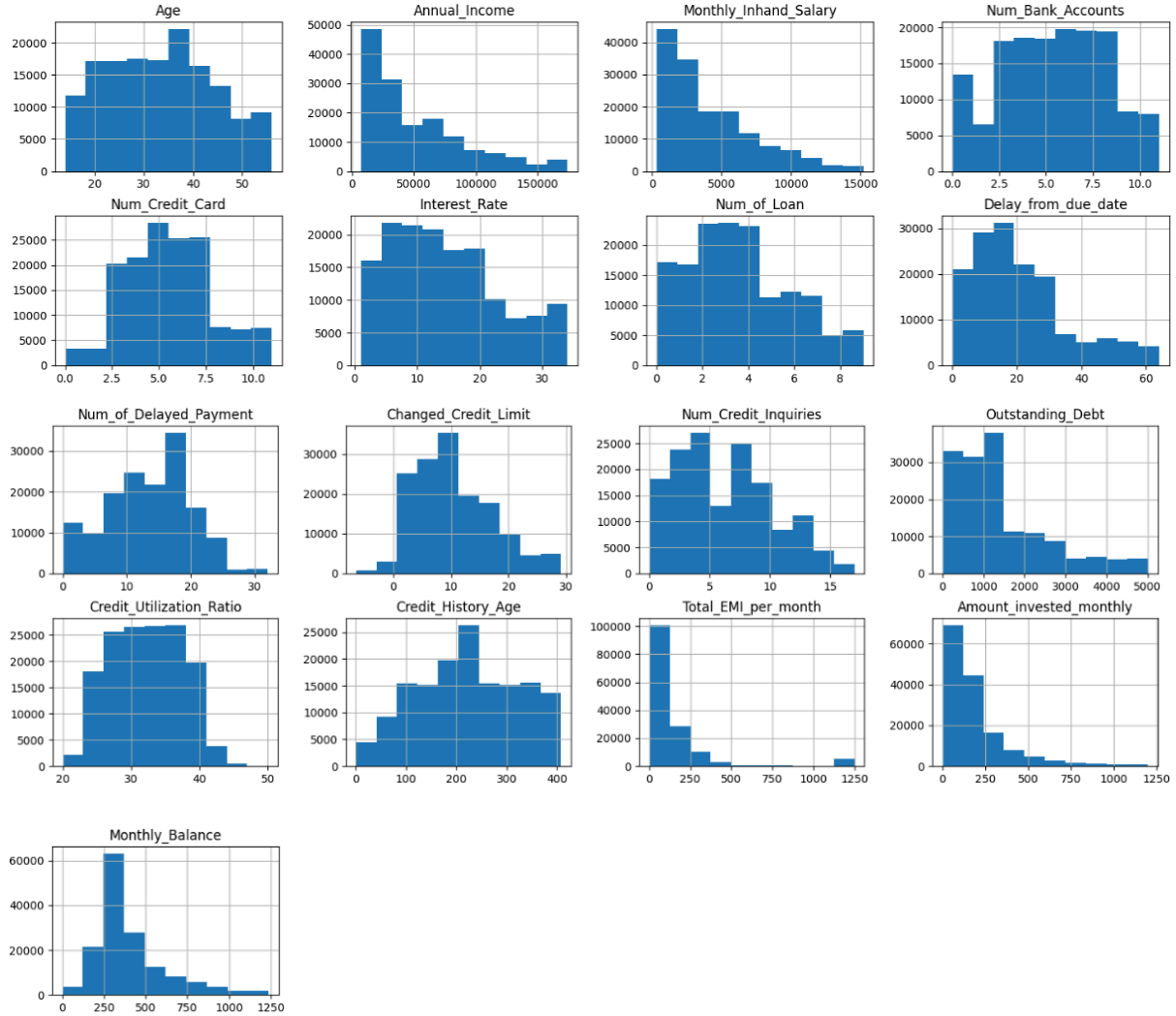

Рис. 4 - Распределения значений после работы с выбросами

Почти для всех моделей является важным диапазон измерения величины. Если признаки лежат в разных диапазонах - необходимо их нормализовать. Используем масштабирование в данной задаче, чтобы признаки находились в пределах одного диапазона, но при этом сохранялось их исходное распределение. Таким образом, будет сохранена уникальность каждого признака [1]. Здесь был использован RobustScaler из библиотеки sklearn.

На этом этапе также были удалены строки с пустой целевой переменной, удалены категориальные признаки, не имеющие смысла для задачи, несбалансированные или потенциально наводящие на информацию из других столбцов. После вновь проведенного корреляционного анализа удален признак Monthly_Inhand_Salary из-за высокой корреляции (коэффициент корреляции выше 0,85) с Annual_Income и потенциальной высокой корреляции с Monthly_Balance. Это удаление должно ускорить расчет, не ухудшив качество предсказания.

Задача поведенческого скоринга, фактически, уже была решена в исходных данных: порог, по которому отсекаются добросовестные заемщики от недобросовестных уже установлен заказчиком (Poor: [0.0; 0.3], Standard: (0.3; 0.7), Good: [0.7; 1.0]). В исходном датасете скоринговое значение представлено категориями. Для кастомизации модели скоринга целевая переменная должна быть искусственно модифицирована в соответствии с требованиями заказчика по влиянию профессии на скоринговый балл. Сначала наименование группы рейтинга заменено на ее средний балл, затем в новом столбце рассчитан скорректированный скоринговый балл, в зависимости от профессии. Затем скоринговый балл преобразован обратно в одну из трех групп-категорий. Таким образом, при учете влияния рейтинга профессии,

заемщик мог перейти из одной группы рейтинга в другую. После таких преобразований распределение скоринговых групп стало более сбалансированным (Рис. 5).

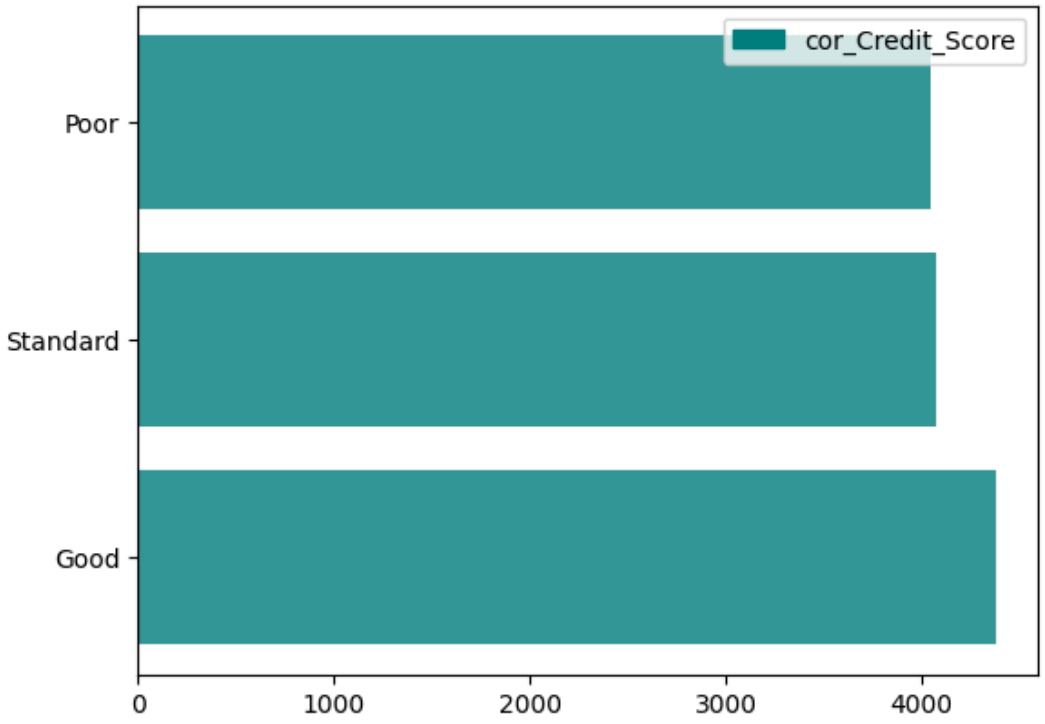

Рис. 5 - Распределение скоринговых групп после модификации

Следующим этапом закодированы признаки-категории с помощью метода get_dummies. Целевая переменная кодировалась методом LabelEncoder библиотеки sklearn. Далее сформированы X и y, данные итогового датасета разделены на данные для обучения и проверки.

4. Подбор модели машинного обучения

В данной работе были использованы некоторые из наиболее популярных моделей машинного обучения для задач классификации. Производительность и эффективность моделей была оптимизирована путем экспериментального подбора параметров.

Модель линейного дискриминантного анализа показала себя хуже всего в точности (accuracy score = 0.7552, precision = 0.747, recall = 0.753), хотя и ни разу не допустила критических ошибок (определить класс Poor как Good или Good как Poor) (Рис. 6) [2]. Скорее всего, это связано с тем, что дискриминантная модель чувствительна к не нормальным распределениям, которых в датасете много [3].

	predicted Good	predicted Poor	predicted Standart
Good	775	0	90
Poor	0	680	134
Standart	230	158	433

Рис. 6 - Матрица ошибок для модели LDA

Модель логистической регрессии и модель случайного леса в среднем показали себя лучше в способности различать классы, но допустили несколько критических ошибок [4, 5]. Метод опорных векторов оказался немного лучше двух предыдущих в плане критических ошибок.

Лучше всего с задачей справился метод XGBoost. Он показал не только самую высокую точность (accuracy score = 0.782, precision = 0.776, recall = 0.781) и меньшее количество критических ошибок (Рис. 7), но и высокую скорость работы алгоритма. Площадь под ROC-кривой составила 0.958 (Рис. 8), что примерно так же, как и для моделей линейного дискриминантного анализа, случайного леса, логистической регрессии. Модель градиентного повышения создает модель классификации из ансамбля слабых прогнозных моделей (деревьев решений), обучая их на разных подмножествах датасета [5, 6].

	predicted Good	predicted Poor	predicted Standart
Good	741	2	122
Poor	5	731	78
Standart	154	184	483

Рис. 7 - Матрица ошибок для модели XGBoost

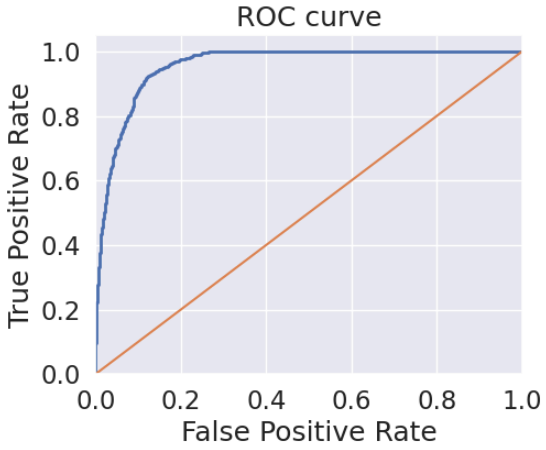

Рис. 8 - ROC-кривая для модели XGBoost. AUC = 0.958

5. Сравнение результатов применения модели при обучении на подготовленном датасете и датасете с экспертно отобранными признаками

Для определения корректности выбора признаков для машинного обучения, результаты работы модели необходимо сравнить с результатами работы модели на основе экспертно отобранных признаков (они выигрывают в интерпретируемости, потому более интересны бизнес-пользователям). Таким образом, модель будет применена к двум датасетам с по-разному отобранными признаками.

Одним из признаков для построения модели в “экспертном” датасете стало рассчитанное значение “размер ежемесячного платежа/размер общих трат за месяц”.

Признаки, не выбранные экспертом (Monthly_Balance, Credit_History_Age, Num_of_Delayed_Payment, Interest_Rate, Num_Bank_Accounts), были удалены.

Далее данные "экспертного" датасета были таким же образом разделены на данные для обучения и проверки. Применена модель XGBoost. Метрики качества для выбранной модели при применении на исходном улучшенном датасете оказались очень схожи с метриками качества на основе экспертно отобранных признаков (accuracy score = 0.782, precision = 0.776, recall = 0.781). Это означает, что признаки изначально были выбраны удачно. При этом, на основе экспертно отобранных признаков модель лучше различает заемщиков группы Good от заемщиков группы Poor, что критичнее для банка (Рис. 9). Площадь под кривой оказалась незначительно меньше (Рис. 10).

	predicted Good	predicted Poor	predicted Standart
Good	748	1	116
Poor	2	729	83
Standart	154	189	478

Рис. 9 - Матрица ошибок для модели XGBoost при использовании экспертно отобранных признаков

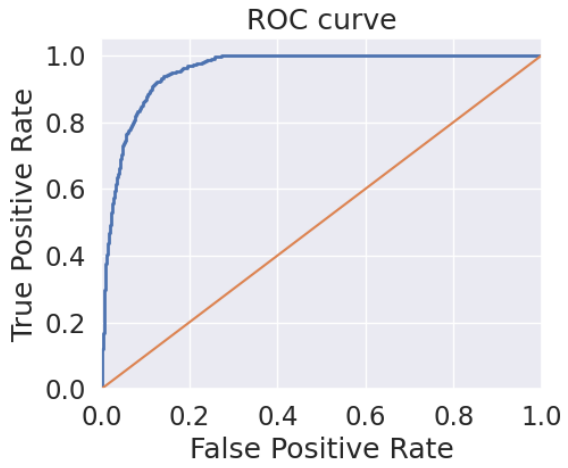

Рис. 10 - ROC-кривая для модели XGBoost при использовании экспертных признаков. AUC = 0.957

6. Рекомендации по использованию и улучшению решения задачи

Традиционно задача скоринга решается на основании анкет, которые предоставляются заемщику. Банк проверяет анкеты на достоверность и далее принимает решение о выдаче кредита [7]. Но это применимо только к ситуациям, когда заемщик сам выступает инициатором кредита. В случаях, когда банк хочет сделать превентивное предложение, банк вынужден ориентироваться на все доступные ему данные о финансовом поведении клиента. Процесс таких предложений может быть автоматизирован при использовании портрета идеального заемщика (например, если данные о клиентах занесены в некоторую систему, которая может автоматически отправлять финансовые предложения).

Найдем идеальных заемщиков по мнению данного банка, используя скорректированное числовое значение скорингового балла. Для этого достаточно отфильтровать датасет по самым высоким баллам и посмотреть на статистики. Средние значения или медиана как раз могут отражать значения признаков идеального заемщика. Таким образом, идеальный заемщик - это разработчик 36 лет с зарплатой 5000 долларов, у которого было или есть 2-3 кредита за 23 года кредитной истории.

Как можно улучшить точность предсказания? Так как модель градиентного повышения чувствительна к зашумленным данным и выбросам, можно было бы минимизировать их влияние. Например, использовать другие методы работы с выбросами. Если бы данных было так много, а выбросов относительно мало, можно было бы просто удалить строки с выбросами, не переживая о том, что будет удалена значительная часть характерных наблюдений класса.

Полученную точность предсказания я связываю с относительно низким качеством данных из-за большого количества фактических ошибок, которые невозможно исправить, не сверяясь с первичной документацией. Можно порекомендовать заказчику улучшить процесс сбора данных, автоматизируя их агрегацию из различных систем, и полагаться только на подтвержденные данные анкет заемщиков.

7. Список источников

1. Умная нормализация данных / Хабр

2. Метрики в задачах машинного обучения

3. Which Test: Logistic Regression or Discriminant Function Analysis | IntroSpective Mode

4. 1.1. Linear Models — scikit-learn 1.3.1 documentation

5. 1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking — scikit-learn 1.3.1 documentation

6. 4.4 Ансамбли моделей: бэггинг, случайные леса, бустинг

7. Скоринг: как банки и МФО решают, давать ли вам кредит