**Федеральное государственное автономное образовательное учреждение**

**высшего образования «Национальный исследовательский университет   
«Высшая школа экономики»**

**Факультет компьютерных наук**

**Магистерская программа «Финансовые технологии и анализ данных»**

**Департамент больших данных и информационного поиска**

**КУРСОВАЯ РАБОТА**

**На тему «Применение алгоритмов обучения с подкреплением в кредитном процессе»**

Студент группы № мФТиАД21

Попов Илья Иванович

(Ф.И.О.)

Руководитель КР

ПАО “Сбербанк”, исполнительный директор по исследованию данных, Щербаков Игорь Андреевич

(должность, звание, Ф.И.О.)

Москва, 2022

# Аннотация

В деятельности современных банков часто возникает задача выбора оптимального порога отсечения результатов скоринговых моделей для одобрения заявок по кредитам. Классические методы решения данной задачи имеют ряд недостатков, среди которых изменения потока кредитных заявок со временем, проблема смещения отбора и малая возможность адаптации к изменениям условий для правил, рассчитанных на исторических данных. Данные проблемы мешают банкам своевременно адаптироваться к изменениям в экономической среде.

В данной работе рассматривается возможность применения алгоритмов машинного обучения с подкреплением (англ. Reinforcement Learning, RL) для решения задачи динамического выбора порога отсечения в реальном времени. Для решения этой задачи была проделана следующая работа:

* реализована среда для обучения RL-агента;
* спроектирован и обучен агент, который на основе данных о состоянии среды выбирает оптимальное действие для максимизации прибыли банка;
* проведено тестирование полученного агента и сравнение результатов с классическими методами решения задачи.

По результатам работы вознаграждение, полученное агентов, превысило вознаграждение, полученное классическими методами решения задачи, как на обучающей, так и на тестовой выборках, что говорит о возможности использования данного подхода в реальном бизнесе.

In the activities of modern banks, the problem often arises of choosing the optimal acceptance threshold for cutting off the results of scoring models for approving loan applications. Classical methods for solving this problem have many disadvantages, including changes in the flow of loan applications over time, the problem of selection bias, and little ability to adapt to changes in the environment as the rules are calculated with historical data. These problems prevent banks from adapting to changes in the economic environment on time.

In this paper, we consider the possibility of using Reinforcement Learning (RL) machine learning algorithms to solve the problem of real-time dynamic selection of the cutoff threshold. The following work was done in order to use RL for this problem:

An environment for training the RL agent was implemented;

An RL agent was designed and trained;

The obtained RL agent was tested, and the results were compared with classical methods for solving the problem.

According to the work results, the remuneration received by the RL agent exceeded the remuneration received by using classical methods of solving the problem, both on training and test samples, which indicates the possibility of using this approach in real business.

# Содержание

[**Аннотация**](#_heading=h.hkp73zrgkp3j) **2**

[**Содержание**](#_heading=h.xqoetg4yhci7) **4**

[**Глава 1. Постановка задачи**](#_heading=h.n743vff38o0w) **5**

[**Глава 2. Описание методов RL**](#_heading=h.e1ufqaq64eki) **6**

[**Глава 3. Описание экспериментов**](#_heading=h.sq701ycvxpq2) **9**

[3.1 Описание данных](#_heading=h.rjyofrody8i5) 9

[3.2 Скоринг](#_heading=h.6ojvudk6anp9) 10

[3.3 Среда](#_heading=h.ekyma0kces3m) 14

[3.3.1 Пространство действий](#_heading=h.inankh1ervdw) 15

[3.3.2 Наблюдение, передаваемое агенту](#_heading=h.ai3i88dzf42u) 15

[3.3.3 Входные данные среды](#_heading=h.z3cedu9nzrg1) 16

[3.3.4 Функция вознаграждения](#_heading=h.eh7p2nsbzums) 16

[3.3.5 Шаг среды и реализация](#_heading=h.aevlxnfvqbh9) 17

[3.4 Агент](#_heading=h.s5uqbp2cohk4) 18

[3.4.1 Архитектура агента](#_heading=h.73wxkg1be0hp) 18

[3.4.2 Реализация агента](#_heading=h.c4dd35wesf8h) 20

[3.5 Бейзлайн](#_heading=h.4hg8frhawh6g) 22

[3.6 Эксперимент](#_heading=h.rzrloslrfqir) 22

[**Заключение**](#_heading=h.ns56tsfaahd) **25**

[**Список источников**](#_heading=h.wcfri5db6fa0) **26**

[**Приложения**](#_heading=h.9gip7vjfhcft) **27**

[Приложение 1 - Список признаков, использованных для построения модели кредитного скоринга](#_heading=h.4msyuql9bgsw) 27

# Глава 1. Постановка задачи

В ходе своей операционной деятельности банки постоянно сталкиваются с задачей оценки рисков. Существует множество их классификаций, но основной, на котором мы сосредоточимся в данной работе - кредитный риск. Он является ключевым в вопросе банковской отрасли, так как именно выдача кредитов приносит банкам основной доход. Для банка кредитный риск - это в первую очередь угроза невыполнения заемщиком своих обязательств по договору кредитования, или дефолт.

Кредитование для банков может приносить как большую выгоду, так и большой убыток в случае выбора некорректной стратегии выдачи кредита. Для того, чтобы этого избежать, банки в процессе рассмотрения кредитной заявки собирают и обрабатывают большой объем дополнительной информации и используют скоринговые модели для определения вероятности дефолта заемщика.

Однако, недостаточно просто определить вероятность дефолта заемщика или определить его кредитный рейтинг. После этого необходимо определить политику одобрения кредита, иначе говоря найти такое правило одобрения, которое будет основываться на предсказании скоринговой модели и приносить банку максимальную прибыль. В случае, если порог будет установлен слишком низкий - банк будет терять деньги, выдавая кредиты некачественным заемщикам. Если же порог отсечения будет слишком высокий - банк будет упускать выгоду, не одобряя кредиты качественным заемщикам.

Существует большой спектр классических методов расчета acceptance threshold. В этой работе мы рассмотрим возможность применения алгоритмов машинного обучения с подкреплением для расчета оптимальной отсечки одобрения. Существенным преимуществом такого подхода является возможность оперативно и автоматизировано реагировать на изменения во внешней среде и корректировать политику одобрения в соответствии с новыми условиями.

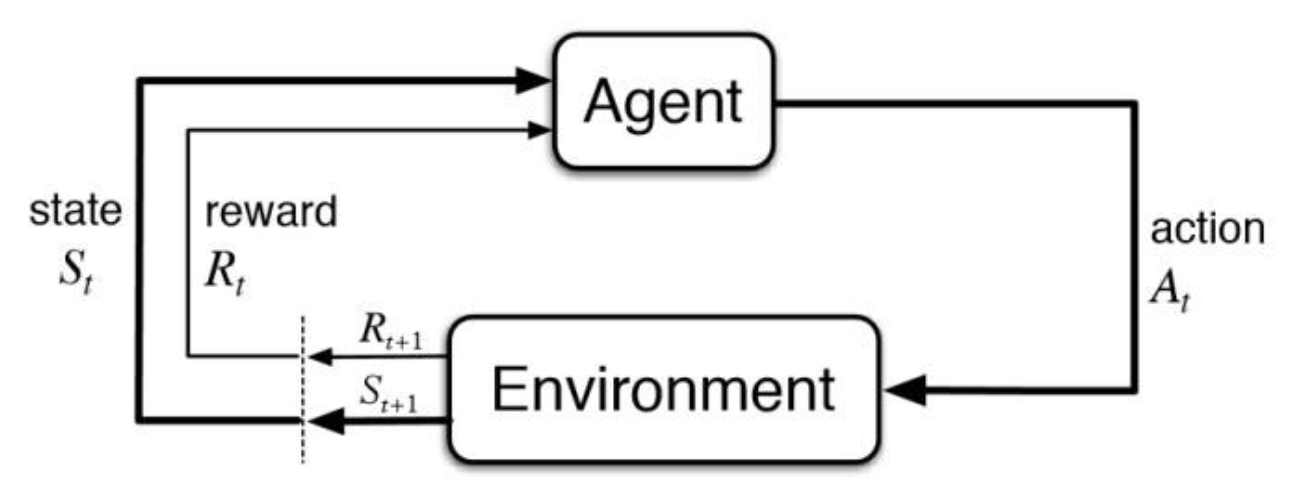
# Глава 2. Описание методов RL

Обучение с подкреплением – это класс алгоритмов машинного обучения, в которых агент (модель) обучается путем взаимодействия с некоторой средой. Обучение агента основано на поощрении поведения, которое приводит к желаемому результату, с помощью получения вознаграждения. [3] Одной из особенностей данного подхода является то, что как агент своими действиями может менять состояние среды, так и среда со своей стороны влияет на ход обучения агента, возвращая разное вознаграждение за совершение разных действий. В нашем случае это будет выглядеть так – агент путем выбора отсечки кредитного рейтинга влияет на количество и качество выданных кредитов, и, как следствие – на доход банка. Среда же, в свою очередь, возвращает агенту вознаграждение, рассчитываемое на основе состояния среды.

Автоматизированный подход к решению задачи был впервые предложен на основе марковского процесса принятия решений (англ. Markov Decision Process, MDP) и обучения с подкреплением [3]. MDP [4] – это последовательный стохастический процесс принятия решений, характеризующийся пятью элементами:

* эпохи принятия решений (англ. Decision Epoch);
* состояния (англ. State);
* действия (англ. Action);
* вероятности перехода (англ. Transition Probability);
* вознаграждения (англ. Reward).

Агент, взаимодействуя с окружающей средой, контролирует ход случайного процесса. В определенные моменты времени агент вмешивается в этот процесс, выполняя действие, которое влияет на ход процесса в будущем. Такие моменты называются эпохами принятия решений. В каждую эпоху принятия решений система переходит в определенное состояние. В результате агент получает вознаграждение в зависимости от того, какие действия он выбрал в прошлом, после чего система переходит в следующее состояние с определенной вероятностью, которая называется вероятностью перехода. Наглядно этот процесс проиллюстрирован на рис. 1.

  
*Рисунок 1 - Схема взаимодействия агента и среды в ходе решения MDP*

Правило принятия решения – это функция для выбора действия в каждом состоянии, а политика – это набор таких правил принятия решения в пространстве состояний. Реализация политики генерирует последовательность вознаграждений. Цель MDP – выбрать политику, которая максимизирует сумму вознаграждений в этой последовательности. MDP позволяет моделировать последовательные подзадачи, решения в которых необходимо принимать в условиях неопределенности.

В работе [5] авторы применяют для решения задачи класс алгоритмов Q-Learning. Особенность данного метода состоит в том, что на основе получаемого от среды вознаграждения агент формирует функцию полезности (англ. Value Function) Q(s, a), что впоследствии дает ему возможность уже не случайно выбирать стратегию поведения, а учитывать опыт предыдущего взаимодействия со средой. Одно из преимуществ Q-Learning [8] — то, что такой агент в состоянии сравнить ожидаемую полезность доступных действий, не формируя модели окружающей среды. Значение Q(s, a) можно описать как предполагаемое долгосрочное вознаграждение, которое получит агент, совершив действие a в состоянии s. Данная функция может быть сведена к функции, близкой к оптимальной, итерационным подходом с использованием уравнения Беллмана:

, где:

* 𝑠 – текущее состояние среды;
* 𝑎 – выбранное действие;
* 𝑟 – мгновенная награда, которую получит агент при совершении действия a в состоянии s;
* 𝑠′ – состояние, в которое перейдет среда из состояния s после совершения действия а;
* 𝑚𝑎𝑥𝑎′ Q(𝑠′, 𝑎′) – максимальное значение Q-функции в следующем состоянии среды s’;
* 𝛾 – фактор дисконтирования, гиперпараметр алгоритма;

В случаях, когда среда имеет дискретное пространство состояний и действий, в реализациях алгоритма все значения Q-функции записываются в матрицу размера 𝑛\_𝑠𝑡𝑎𝑡𝑒𝑠 × 𝑛\_𝑎𝑐𝑡𝑖𝑜𝑛𝑠, где 𝑛\_𝑠𝑡𝑎𝑡𝑒𝑠 – количество возможных состояний среды и 𝑛\_𝑎𝑐𝑡𝑖𝑜𝑛𝑠 – количество возможных действий агента. В этом кроется один из минусов алгоритма – с ростом размерностей пространств состояний и действий кратно увеличиваются вычислительное время и необходимый объем памяти. Один из способ избежать данную проблему – использовать для получения приближенного значения Q-функции нейронную сеть или другой алгоритм машинного обучения. Такой подход называют DQN (англ. Deep Q-Network) или DQL (англ. Deep Q-Learning) [7].

Именно такой случай и был рассмотрен в работе [5], где пространство действий (а именно - возможные значения порогов отсечения) было ограничено 20 вариантами, что вкупе с continuous action space позволило применить DQL для решения задачи. Однако, в этой работе мы постараемся пойти дальше, и отойти от использования дискретного пространства действий в сторону continuous.

Для этого потребуется перейти к использованию другого алгоритма – DDPG (англ. Deep Deterministic Policy Gradient) [8]. DDPG – это Actor-Critic алгоритм, в котором одновременно происходит обучение двух моделей: одна – для приближения Q-функции (Critic Network), и вторая – для предсказания оптимальной политики (Actor Network). Данный алгоритм часто рассматривают как аналог DDQN для случаев, в которых применяется непрерывное пространство действия [9]. Подробнее о том, как устроен данный алгоритм, см. в главе 3.4.

# Глава 3. Описание экспериментов

## 3.1 Описание данных

В работе используются данные открытого датасета Lending Club c платформы Kaggle [1]. Датасет содержит данные о заявках на получение займов с июня 2007 по декабрь 2018, и для каждого - более 150 фичей. В работе:

* данные с июня 2007 по декабрь 2017 включительно использовались как обучающая выборка;
* данные с января 2018 по июнь 2018 включительно использовались как валидационная выборку для оценки качества моделей;
* данные с июля 2018 по декабрь 2018 включительно в работе не использовались, так как период наблюдения завершился в 2019 году, и в последние полгода количество дефолтов в данных начинает резко падать. Связано это с тем, что для клиентов, получивших заем в этот период, было меньше времени для наступления дефолта, но он мог наступить впоследствии.

Целевая переменная в датасете - флаг наступления дефолта по кредитной заявке. Для того, чтобы построить модель скоринга на основе предсказания вероятности дефолта, первоначально необходимо произвести первичный отбор и анализ признаков. К датасету были применены следующие преобразования:

* Признаки, обозначающие одно и то же, но относящиеся к двум сценариям работы компании (Joint App и Individual) были объединены в один, дополнительно был введен флаг сценария работы - вместо 8 было введено 4 признака и 1 дополнительный;
* Удалены 37 признаков, содержащих более 90% null;
* Удалены 63 признака, которых не могло быть на момент создания анкеты. Они являются потенциальной утечкой таргета в обучающую выборку, и для построения корректной скоринговой модели их использовать нельзя.
* Категориальные признаки, для которых важно отношение порядка (например, рабочий стаж, кредитный рейтинг FICO и др.) были закодированы числами с сохранением отношения порядка;
* Прочие категориальные признаки (например, цель кредита и тип заявки) были закодированы с помощью label encoding [10];
* Числовые признаки были нормированы с помощью standard scaler, пропуски в данных заполнены средним значением признака на обучающей выборке.

После описанных выше действий в датасете осталось 50 признаков, детальное описание которых можно найти в приложении 1. Код, с помощью которого производился предпроцессинг данных, можно найти на странице проекта на Github [11].

## 3.2 Скоринг

Для получения на основе обработанных данных кредитного рейтинга первоначально была обучена модель, предсказывающая на основе входных данных вероятность дефолта по заявке. В качестве моделей были опробованы логистическая регрессия [12] и CatBoost [13] со стандартными значениями гиперпараметров. Качество предсказания для полученных моделей оценивалось на обучающей и тестовой выборках с использованием коэффициента Джини, а также метрики AUC-ROC и AUC-PR. Модель CatBoost показала лучший результат как обучающей, так и на тестовой выборке (см. таблицу), поэтому для дальнейшего построения кредитного скора в работе использовались предсказанные ей вероятности дефолта.

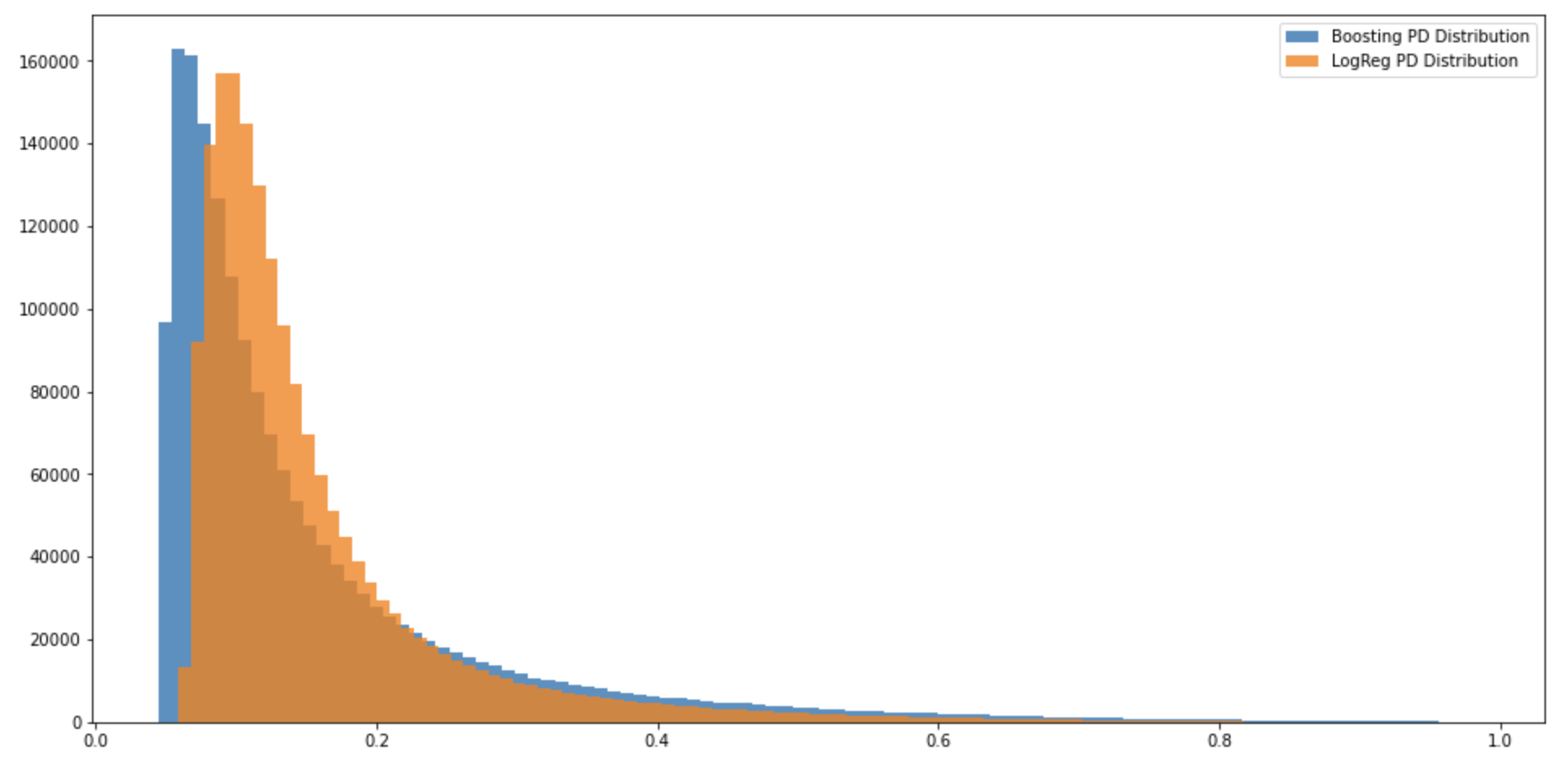
|  | Train | | Test | |
| --- | --- | --- | --- | --- |
| LogReg | CatBoost | LogReg | CatBoost |
| Gini | 0.40559 | 0.49435 | 0.39573 | 0.45632 |
| AUC-ROC | 0.70280 | **0.74718** | 0.69786 | **0.72816** |
| AUC-PR | 0.29913 | **0.37981** | 0.07728 | **0.09509** |

Далее для преобразования вероятности дефолта в удобный для интерпретации кредитный рейтинг используем масштабирование. Такое преобразование не ведет к потере точности предсказания, а просто преобразует его в плоскую шкалу от 0 до 1000, в которой чем больше значение - тем меньше вероятность дефолта. Это преобразование происходит по следующей формуле [2]:

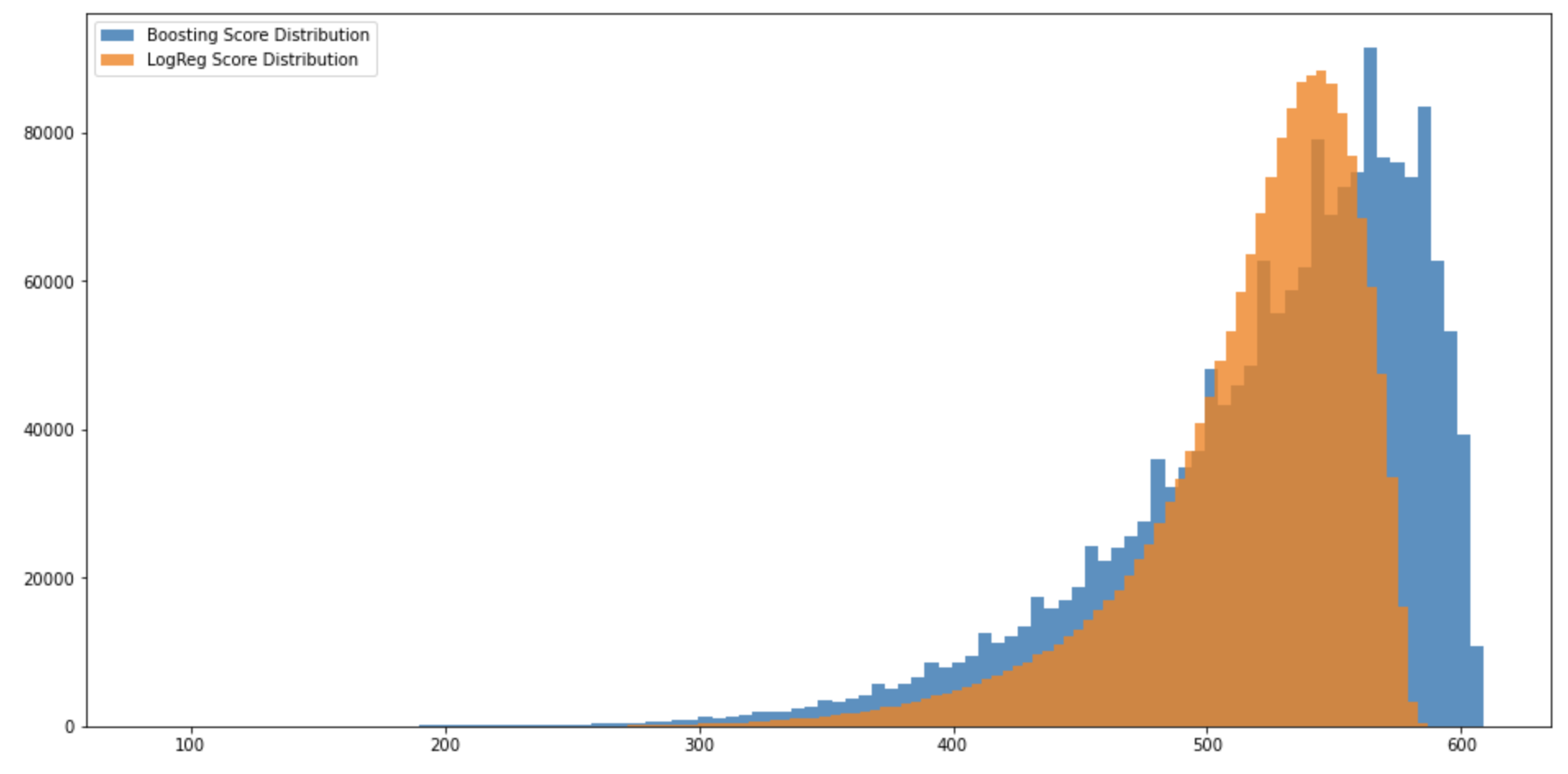
, где:

* - смещение, в котором:
  + - количество баллов, удваивающее шансы на дефолт;
  + - количество баллов, при котором вероятность дефолта равна :1;
* - нормирующее слагаемое, в котором:
  + - предсказанная моделью вероятность дефолта по кредиту.

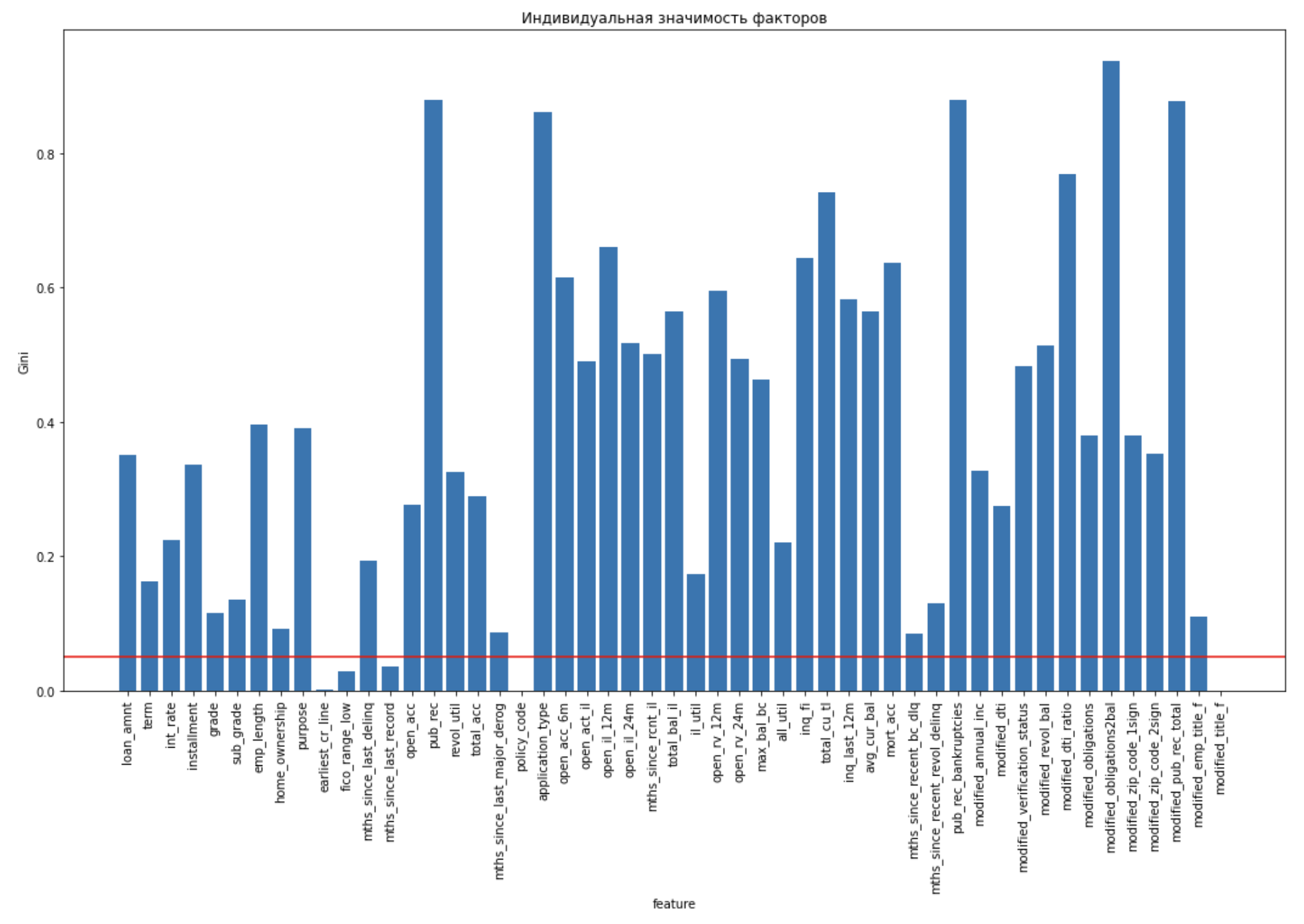
В работе были использованы стандартные для большинства кредитных карт значения = 600, = 19 и = 50. На рисунках 2 и 3 приведены распределения первоначальных предсказаний вероятности дефолта и получившихся баллов, наглядно показывающие суть преобразований.



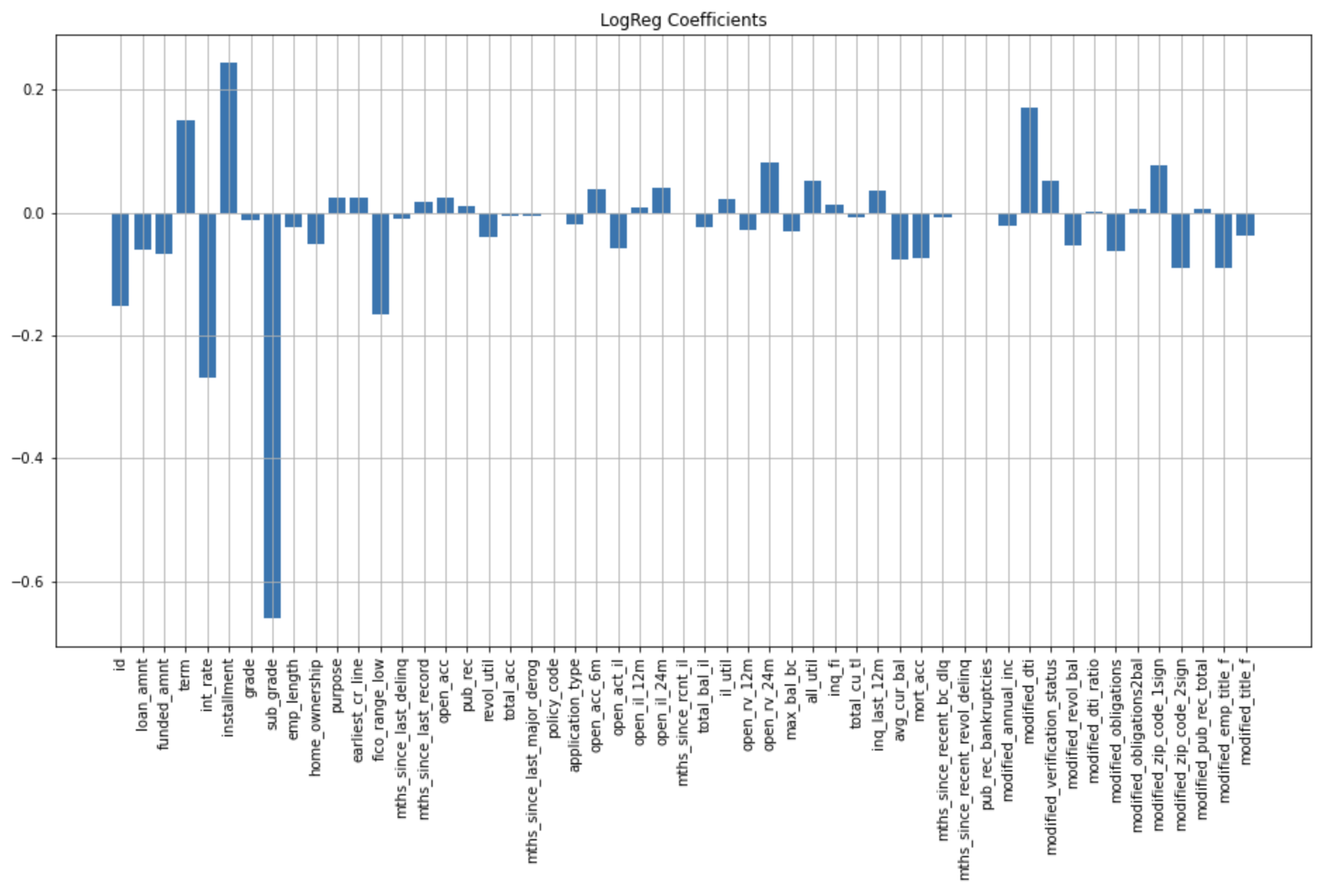
*Рисунок 2: Распределения значений вероятности дефолта, полученных с помощью логистической регрессии и CatBoost*



*Рисунок 3: Распределения значений кредитного рейтинга, полученные на основе предсказаний от логистической регрессии и CatBoost*



*Рисунок 4: Индивидуальная значимость факторов, рассчитанная с помощью коэффициента Джини для модели CatBoost. Можно сделать вывод, что все признаки, кроме четырех, полученные после предобработки, можно считать значимыми. Однако, после их удаления и повторного обучения модель показала результат хуже, что говорит о значимости и этих признаков.*



*Рисунок 5: Коэффициенты при весах в логистической регрессии.*

## 3.3 Среда

Среда (англ. Environment) – это симуляция некоторого реального процесса, которая используется для обучения агента. Она не является частью агента и действует полностью независимо. В RL принято использовать искусственные симуляции реальных процессов, так как зачастую обучение агента в реальных условиях получается неоправданно долгим, дорогим или рискованным. В случае данной работы среда имитирует работу банка по одобрению полученных за месяц кредитных заявок.

Для этой работы среда была реализована с использованием фреймворка OpenAI Gym [14]. Этот фреймворк зарекомендовал себя как стандарт в индустрии, позволяя агенту взаимодействовать со средой при помощи стандартных методов. Это не только удобно, но и упрощает работу для других людей, желающих протестировать свой подход к решению задачи в тех же условиях. Исходный код реализованной среды можно найти в репозитории проекта на Github [11].

Разработанная среда является универсальной и может работать в нескольких режимах. В зависимости от поставленной задачи и архитектуры агента можно выбрать между выбором отсечки одобрения заявки по кредитному рейтингу (если рейтинг больше отсечки - заявка одобряется, в противном случае отклоняется) или по вероятности дефолта (если вероятность дефолта меньше отсечки - заявка одобряется, в противном случае отклоняется). Также можно выбрать режим получения агентом вознаграждения:

* мгновенный (англ. immediate reward) - агент сразу получает вознаграждение за заявки, обработанные на данном шаге;
* отложенный (англ. delayed reward) - агент получает вознаграждение в тот момент, когда заемщик полностью вернет кредит (положительное вознаграждение) или когда заемщик объявит о дефолте (отрицательное вознаграждение). В первом случае вознаграждение поступает после прохождения указанного в заявке срока кредита, во втором - через случайно выбранное количество месяцев между значением гиперпараметра среды *DEFAULT\_MONTH\_MIN* и указанным в заявке сроке кредита. Это вносит дополнительный фактор случайности в среду, делая её более похожей на реальность.

В рамках этой работы отсечка устанавливалась агентом на кредитный рейтинг и агент получал отложенное вознаграждение.

### 3.3.1 Пространство действий

В качестве действия среда принимает одно целое число - отсечку кредитного рейтинга (или вероятность дефолта в зависимости от выбранного режима работы). Заявки на кредит, имеющие больший или равный рейтинг, одобряются, а имеющие меньший - отклоняются.

Важно отметить, что для имитации условий реального мира, было принято решение не ограничивать агента дискретным пространством действий, а перейти к использованию continuous action space. Это значит, что допустимым действием агента будет являться выбор любой отсечки из множества целых чисел. Использование continuous action space также накладывает дополнительные ограничения на архитектуру агента, что также является важным отличием данной работы от предыдущих. Об этом подробнее рассказано далее в главе 3.4.

### 3.3.2 Наблюдение, передаваемое агенту

В качестве описания состояния среды на данный момент времени агент получает acceptance rate за предыдущий месяц - отношение числа одобренных заявок к общему количеству заявок на кредит. Такой выбор был сделан по нескольким причинам:

* Сохранение размерности наблюдения как можно меньше позволяет алгоритму сходиться быстрее, делает результаты более интерпретируемыми и более легкими для отладки;
* Значение acceptance rate сильно коррелирует с другими показателями, описывающими состояние среды, такими как количество и объем одобренных и полученных заявок, а также - с вознаграждением агента;
* Значение acceptance rate стационарно и всегда лежит в интервале [0; 1]. Это делает его удобным для использования в наблюдения в MDP.

### 3.3.3 Входные данные среды

В качестве входных данных среда использует полученных от скоринговой модели вероятность дефолта (англ. probability of default, pd) и кредитный рейтинг (англ. score), а также часть показателей из предобработанного ранее датасета:

* *issue\_d* - месяца и год подачи заявки на кредит;
* *term* - срок кредита в месяцах;
* *int\_rate* - ставка по кредиту в процентах;
* *funded\_amnt* - запрошенная сумма кредита в долларах.

### 3.3.4 Функция вознаграждения

В качестве вознаграждения агент по итогам каждого шага среды рассчитывается суммарных доход банка согласно выбранной политике одобрения заявок за месяц. Этот процесс можно описать для с помощью следующей матрицы состояний:

|  | | **Была ли одобрена заявка на кредит?** | |
| --- | --- | --- | --- |
| **Да** | **Нет** |
| **Был ли объявлен по заявке дефолт?** | **Да** | Случай 1: банк теряет деньги | Случай 2: 0 |
| **Нет** | Случай 3: 0 | Случай 4: банк получает прибыль |

* **Случай 1**: банк получает убыток в размере *LDG • funded\_amnt*, где *LDG* (сокр. от англ. Loss Given Default) - доля от суммы кредита, которую банк теряет в случае, когда по заявке объявляется дефолт. *LGD* является гиперпараметром среды и устанавливается при ее инициализации. В рамках данной работы *LGD* считается равным 1;
* **Случай 2**: клиент объявлен дефолтом, но модель сработала корректно и не одобрила ему кредит - банк ничего не теряет;
* **Случай 3**: обратная ситуация, клиент не объявлен дефолтом, но модель сработала некорректно и не одобрила ему кредит - банк ничего не теряет, но ничего и не зарабатывает;
* **Случай 4**: банк получает прибыль в размере

, где:

* + *term* - срок кредита в месяцах;
  + *int\_rate* - ставка по кредиту в процентах;
  + *funded\_amnt* - запрошенная сумма кредита в долларах.

В случае, если среда работает в режиме получения мгновенного вознаграждения, то агент сразу же получает суммарное вознаграждение от всех заявок, размеченных на этом шаге. Если же среда работает в режиме отложенного вознаграждения, то для каждой заявки награда добавляется в соответствующий месяц в очередь.

### 3.3.5 Шаг среды и реализация

Как уже было сказано выше, для реализации среды был использован фреймворк OpenAI Gym. Это в том числе обозначает и то, что для взаимодействия со средой агенту доступны стандартные для большинства сред OpenAI Gym методы reset и step.

При инициализации среды происходит загрузка данных с предварительно рассчитанными значениями кредитного рейтинга и вероятности дефолта, фиксируется режим работы среды (вознаграждение: отложенное либо мгновенное; отсечка: скор или вероятность дефолта) и гиперпараметры (*LDG* и *DEFAULT\_MONTH\_MIN*).

На первом шаге после вызова метода reset среда фиксирует текущий месяц *currMonth*, текущий год *currYear*, в случае работы в режиме отложенного вознаграждения - объявляет очередь из нулей для хранения будущих вознаграждений и возвращает:

* *observation* = 0;
* *reward* = 0;
* *isDone* = False, флаг завершения цикла среды.

При вызове метода step среда получает выбранное агентом действие, и в зависимости от выбранного режима производит следующие действия:

* Для заявок, поступивших за месяц *currMonth* года *currYear* согласно полученной политике определяется, были ли они приняты или отклонены;
* Для всех принятых заявок рассчитывается вознаграждение агента:
  + Если среда работает в режиме мгновенного вознаграждения, то полученные вознаграждения суммируются и формируют *reward*;
  + Если среда работает в режиме отложенного вознаграждения, то вознаграждение добавляется к соответствующему месяцу в очереди. *Reward* объявляется равным первому значению отложенного вознаграждения в очереди, а в конец очереди добавляется 0;
* Рассчитывается *observation = acceptance\_rate = count(approved) / count(total)*;
* Если *currMonth* и *currYear* - последний месяц в истории наблюдений, то *isDone* = True, в противном случае - *isDone* = False;
* Цикл продолжается, пока *isDone* = False;

Код среды можно найти в репозитории проекта на Github [11].

## 3.4 Агент

### 3.4.1 Архитектура агента

Как уже было сказано выше, отказ от использования дискретного пространства действий и переход к continuous action space усложняет архитектуру агента. В частности, в данном случае становится необходимо использовать помимо нейронной сети для приближения значения Q-функции дополнительную нейронную сеть для предсказания оптимального действия. Такой класс алгоритмов RL также называют Actor-Critic, в котором сеть Critic - приближает значение Q-функции на основе входных данных, а есть Actor - выбирает действие, принимая на вход те же данные, что и сеть Critic, плюс результат ее работы.

Помимо этого, для ускорения и стабилизации процесса обучения в работе используются еще два распространенных приема: это механизм Soft Update и Prioritized Experience Replay.

Для осуществления Soft Update вместо каждой из нейронных сетей используется две: Target Network для получения результата и Main Network для осуществления обучения. Изначально, обе сети инициализируются с одинаковым набором случайно выбранных весов. Для Main Network обучение происходит как обычно, а веса Target Network обновляются из весов Main Network с использованием механизма Soft Update:

𝜃𝑄′ ← 𝜏𝜃𝑄 + (1 − 𝜏)𝜃𝑄′, где:

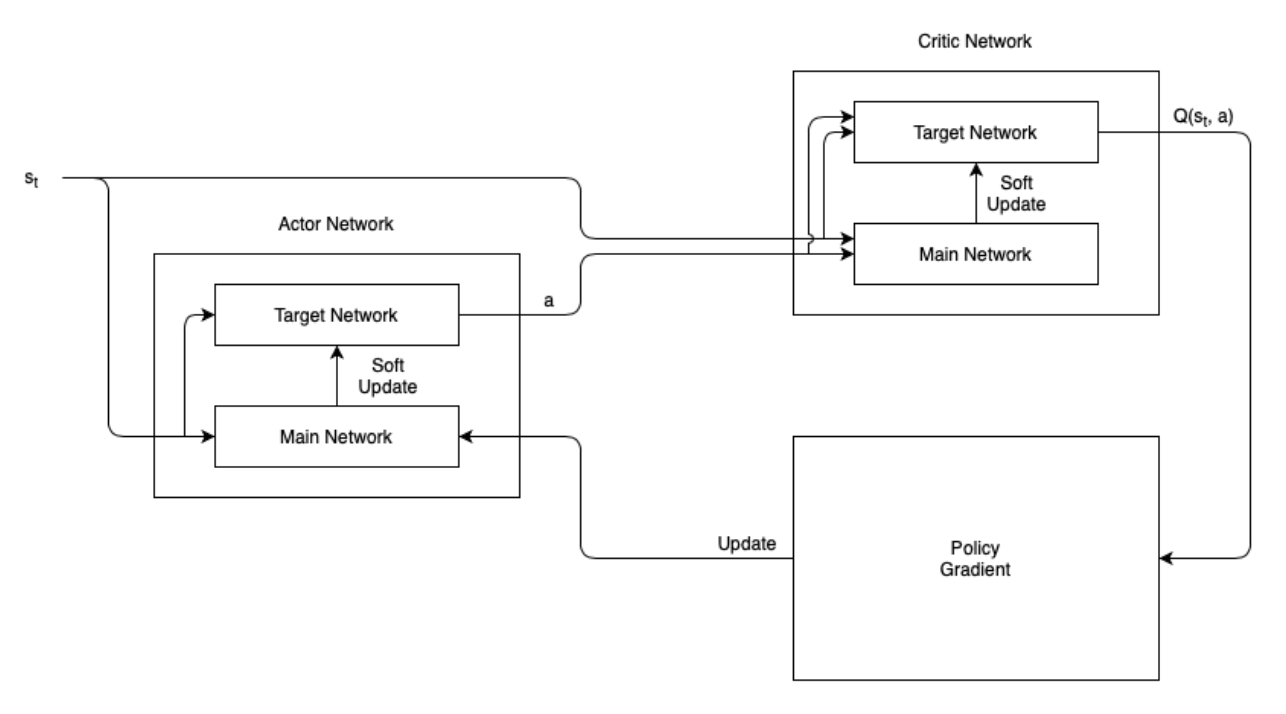
* 𝜃𝑄′ – веса в Target Network;
* 𝜃𝑄 – веса в Main Network;
* 𝜏 ∈ (0; 1] – гиперпараметр алгоритма. Чем ближе его значение к 1, тем быстрее происходит обновление Target Network. При 𝜏 = 1 такой подход еще называют Hard Update, так как весам в Target Network просто присваиваются значения весов в Main Network.

Идея Exprerience Replay состоит же в том, что данные об уже пройденных шагах записывается в буфер как последовательности вида (𝑠,𝑎,𝑠’,𝑟). Далее в процессе обучения на каждом шаге уже из этого буфера выбирается некоторая часть наблюдений (англ. batch), которая уже передается нейронной сети. В работе мы будем использовать модифицированную версию Experience Replay, в которой на каждой итерации обучения не просто выбирается случайный batch из буфера, а набор полученных на последней итерации (𝑠,𝑎,𝑠’,𝑟) дополняется до набора размера *𝑏𝑎𝑡𝑐h\_𝑠𝑖𝑧𝑒* элементами из буфера. Данный подход носит название Prioritized Experience Replay [15] и позволяет учитывать опыт последних шагов в приоритете над историческими данными при обучении модели, что увеличивает скорость обучения и обеспечивает необходимую гибкость в изменении предсказания со временем.

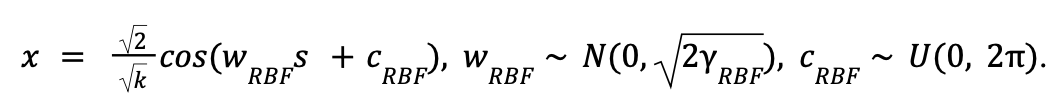
### 3.4.2 Реализация агента

Обучение агента организовано следующим образом на основе алгоритма, предложенного в статье [17]:

1. Инициализируем нейронную сеть actor 𝜇(𝑠|𝜃𝜇) и нейронную сеть critic 𝑄(𝑠, 𝑎|𝜃𝑄) со случайными весами 𝜃𝜇 и 𝜃𝑄 соответственно;
2. Инициализируем нейронные сети 𝜇′(𝑠|𝜃𝜇′) и 𝑄′(𝑠, 𝑎|𝜃𝑄′) с весами 𝜃𝜇′ ← 𝜃𝜇 и 𝜃𝑄′ ← 𝜃𝑄 соответственно;
3. Инициализируем буфер R;
4. Для каждого эпизода обучения:
   1. Получаем первое наблюдение 𝑠1
   2. Инициализируем случайный процесс Ν для получения шума, добавляемого в процессе обучения;
   3. Пока эпизод обучения не завершился, 𝑡 ≥ 1 – номер текущего шага обучения:
      1. Выбираем действие 𝑎𝑡 = 𝜇(𝑠𝑡|𝜃𝜇) + Ν𝑡;
      2. Выполняем действие 𝑎𝑡, получаем новое состояние 𝑠𝑡+1 и вознаграждение 𝑟𝑡;
      3. Дополняем множество полученных (𝑠t, 𝑎t, 𝑟t, 𝑠t+1) до батча размера N случайно выбранными наблюдениями (𝑠i, 𝑎i, 𝑟i, 𝑠i) из буфера R;
      4. Рассчитываем 𝑦i = 𝑟i + 𝛾𝑄′(𝑠i+1, 𝜇′(𝑠i+1|𝜃𝜇′)|𝜃𝑄′);
      5. Обновляем веса модели critic, минимизируя функционал ошибки:𝐿= 1/N ∑𝑖(𝑦𝑖−𝑄(𝑠𝑖,𝑎𝑖|𝜃𝑄))2;
      6. Обновляем веса модели actor, используя градиент:  
         ∇𝜃𝜇𝐽≈ 1/N ∑𝑖∇𝑎𝑄(𝑠,𝑎|𝜃𝑄)|𝑠=s*i*,𝑎=𝜇(𝑠𝑖|𝜃𝜇) ∙∇𝜃𝜇𝜇(𝑠|𝜃𝜇)|𝑠=𝑠𝑖;
      7. Обновляем 𝜃𝜇′ ← 𝜏𝜃𝜇 + (1 − 𝜏)𝜃𝜇′ , 𝜃𝑄′ ← 𝜏𝜃𝑄 + (1 − 𝜏)𝜃𝑄′;
      8. Записываем (𝑠t, 𝑎t, 𝑟t, 𝑠t+1) в буфер R.

  
*Рисунок 6 - Принципиальная схема обучения DDPG-агента*

Однако, наблюдение, которое отдает среда агенту, состоит лишь из одного наблюдения. Его нельзя передать на вход нейросети без дополнительных преобразований. Для того, чтобы повысить размерность пространства, используется сумма гауссовских радиально-базисных функций (RBF). Сумма RBF позволяет аппроксимировать нелинейные зависимости нормального распределения. Функция состояние и генерирует новый вектор признаков:



Для всех слоев, кроме вывода, используется функция активации ReLU. Для слоя вывода в качестве функции активации используется сигмоида , что позволяет нам на выходе из нейросети получить значения, лежащие в интервале [0; 1]. Для получения action полученное на выходе из нейросети actor значение умножается на максимальное значение score.

## 3.5 Бейзлайн

В качестве бейзлайна для оценки качества модели было использовано наилучшее константное предсказание на обучающей выборке. Такая оценка рассчитывается следующим образом:

* Для всех заявок в выборке рассчитывается вознаграждение;
* Выборка сортируется по убыванию скора;
* На отсортированной выборке рассчитывается кумулятивная сумма вознаграждений;
* Значение кредитного рейтинга, которое соответствует наибольшему значению кумулятивной суммы, будет являться оптимальной константной стратегией.

На обучающей выборке наилучшая константная стратегия - выбор отсечки в 570.

## 3.6 Эксперимент

Для эксперимента фиксируем:

* Обучающая выборка - с июня 2007 по декабрь 2017 включительно;
* Тестовая выборка - с января 2018 по июнь 2018 включительно;
* Режим работы среды:
  + Отложенное вознаграждение;
  + Выбор порога по кредитному скору.

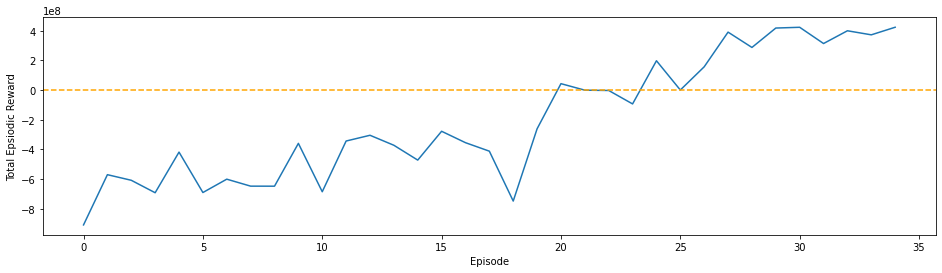
Структура сети Actor, показавшая наилучший результат:

* Размерность входа - 80;
* Два скрытых слоя размерности 256, функция активации - ReLU;
* Размерность выхода - 1, функция активации - сигмоида;

Структура сети Critic, показавшая наилучший результат:

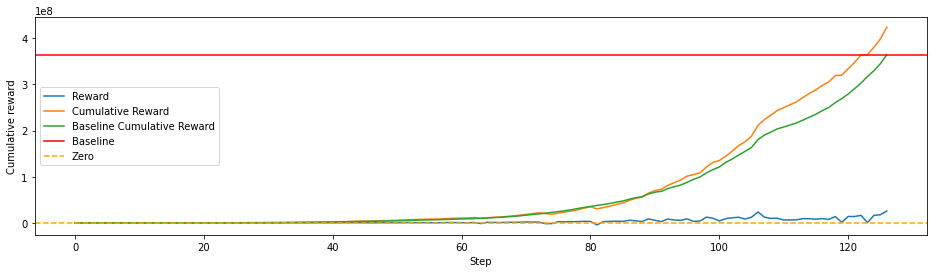
* Слои для обработки State:
  + Размерность входа - 80;
  + Два скрытых слоя размерности 16 и 32 соответственно, функция активации - ReLU;
* Слои для обработки Action:
  + Размерность входа - 1;
  + Один скрытый слой размерности 32, функция активации - ReLU;
* Далее последние слои State и Action конкатенируются, образуя один слой размерности 64;
* Два скрытых слоя размерности 256, функция активации - ReLU;
* Размерность выхода - 1;

После 20 эпох обучения такая конфигурация агента смогла в сумме по итогам одной эпохи обучения получить положительный суммарный reward, выйдя на максимальное значение прибыли к 30 сессии. График 1 показывает, как изменялось суммарное вознаграждение, полученное агентом за одну эпоху обучения.



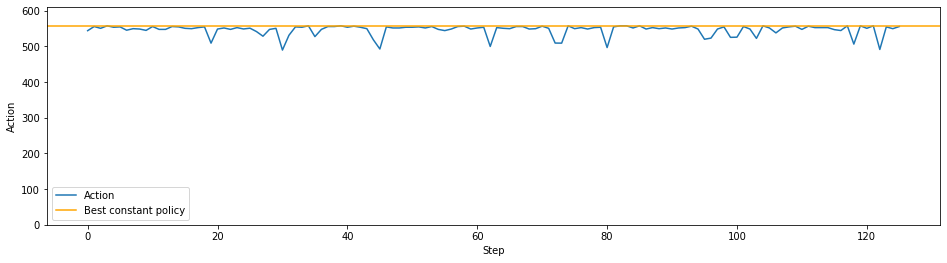
*График 1 - Суммарное вознаграждение агента в течении 35 эпох обучения*

На графике 2 показано, как возрастало суммарное вознаграждение агента на 30 эпохе обучения в сравнении с бейзлайновой стратегией.

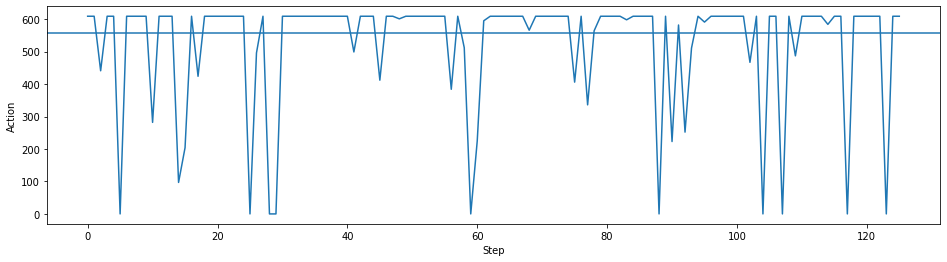


*График 2 - Рост суммарного вознаграждения для DDPG-агента после 30 эпох обучения и бейзлайна в зависимости от номера шага среды*

Ниже на графике 3 представлено сравнение действия , которое выбрал агент и константной политики, которую мы вычислили на предыдущем шаге. Здесь можно наглядно посмотреть, за счет чего был достигнут прирост прибыли - агент периодически выбирал уровень отсечения сильно ниже бейзлайна, что позволяло выдавать больше кредитов и получать большую прибыль.

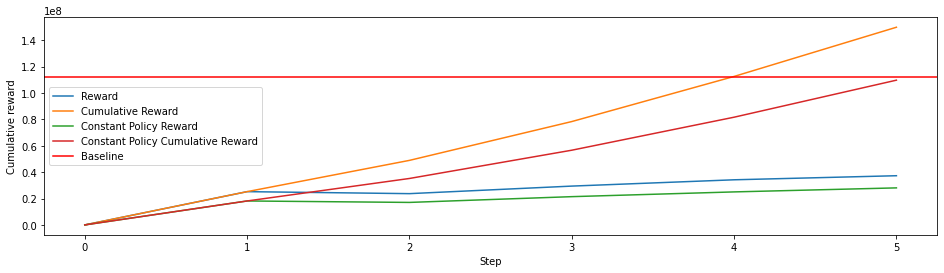
*График 3 - Выбранное агентом действие после 30 эпох обучение и бейзлайн в зависимости от номера шага среды*

Ниже представлен график, на котором отображаются действия, выбранные агентом после 10 эпох обучения. Невооруженным глазом заметно, что после следующих 20 эпох значения действия по модулю приблизились к бейзлайну, но большенство периодов колебаний сохранились. Вывод, который можно тут сделать - модель достаточно быстро установила закономерности для определения периодов, где необходимо снизить порог, а в ходе следующих эпох сглаживался размер необходимого изменения.

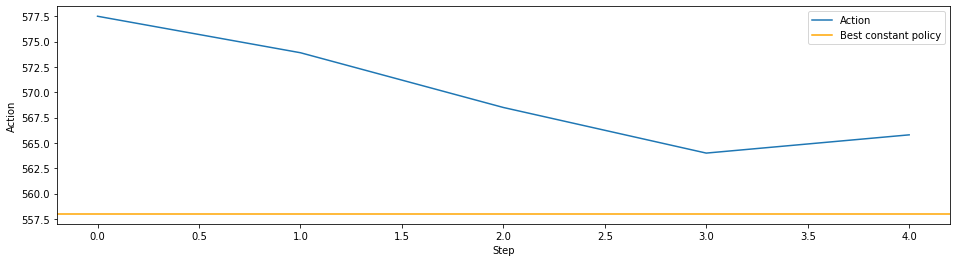


*График 4 - Выбранное агентом действие после 10 эпох обучение и бейзлайн в зависимости от номера шага среды*

Однако, одно дело превзойти бейзлайн на обучающей выборке - это может просто свидетельствовать о переобучении модели. Для того, чтобы подтвердить корректность модели, необходимо также сравнить результаты на тестовой выборке. На графике ниже - сравнение накопленного вознаграждения RL-агента и бейзлайновой стратегии. Можно видеть, что на тестовой выборке результат, показанный агентом, также превзошел бейзлайн, и на этот не за счет того, что агент снижал порог и выдавал больше кредитов, а наоборот - ставил порог больше, чем бейзлайн, и выдавал меньше кредитов, но зато большего качества и предотвращал потери. Это говорит о том, что агент успешно адаптировался к новым условиям и в случае обучения на исторических данных может быть успешно применен в реальном бизнесе для подбора порога в реальном времени.



*График 5 - Рост суммарного вознаграждения для DDPG-агента после 30 эпох обучения и бейзлайна на тестовой выборке в зависимости от номера шага среды*

  
*График 6 - Выбранное агентом действие после 30 эпох обучение и бейзлайн на тестовой выборке в зависимости от номера шага среды*

# Заключение

В ходе работы был проведен анализ существующих работ в области применения RL для задачи подбора порога отсечения в кредитном скоринге. На основе открытых данных была построена скоринговая модель. На ее основе была разработана среда для обучения и тестирования разных RL-агентов, которая была использована для обучения собственного RL-агента. В результате эксперимента было показано, что разработанный агент решает поставленную задачу лучше, чем бейзлайновая стратегия, как на обучающей выборке, так и на тестовой. При этом, полученную в ходе работу среду можно и дальше использовать для совершенствования текущего агента и разработки новых. Среда поддерживает большую вариативность настроек, а благодаря совместимости с OpenAI Gym может быть использована в будущих исследованиях других команд.

Стоит отметить, что полученный агент еще далек от идеала, и в качестве главных направлений для совершенствования я вижу следующие:

* Проведение экспериментов с использованием данных реального банка. Реализованная скоринговая модель далека от идеала, и была создана для демонстрации возможностей RL;
* Дополнительное обогащение входных данных модели основными макроэкономическими показателями и риск-факторами. Это должно помочь агенту лучше и быстрее адаптироваться в условиях неопределенности и резких изменений экономической обстановки в стране и мире.

# Список источников

1. <https://www.kaggle.com/datasets/wordsforthewise/lending-club>
2. <https://naukovedenie.ru/PDF/180EVN214.pdf>
3. Попов, И. И. Оптимизация запасов в ритейл сети с помощью алгоритмов обучения с подкреплением : специальность 01.03.02 «Прикладная математика и информатика» : выпускная квалификационная работа / Попов Илья Иванович ; Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики». — Москва, 2021. — 57 c. — Текст : непосредственный;
4. Puterman M. L. Markov decision processes: discrete stochastic dynamic programming. – John Wiley & Sons, 2014;
5. Mykola Herasymovych, Karl Märka, Oliver Lukason, Using reinforcement learning to optimize the acceptance threshold of a credit scoring model // Applied Soft Computing, Volume 84, 2019, 105697 // ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2019.105697.
6. Watkins C. J. C. H. Learning from delayed rewards. – 1989.
7. Mnih V. et al. Playing atari with deep reinforcement learning //arXiv preprint arXiv:1312.5602. – 2013.
8. Lillicrap T. P. et al. Continuous control with deep reinforcement learning //arXiv preprint arXiv:1509.02971. – 2015.
9. Deep Deterministic Policy Gradient - Spinning Up documentation – <https://spinningup.openai.com/en/latest/algorithms/ddpg.html>.
10. sklearn.preprocessing.LabelEncoder — scikit-learn 1.1.2 documentation - <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>
11. mgcrp/hse\_scoring\_2022: КУРСОВАЯ РАБОТА ФТИАД 2022 - <https://github.com/mgcrp/hse_scoring_2022>
12. sklearn.linear\_model.LogisticRegression — scikit-learn 1.1.2 documentation - <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>
13. CatBoost - open-source gradient boosting library - <https://catboost.ai/>
14. OpenAI GYM. – <https://gym.openai.com/>

# Приложения

## Приложение 1 - Список признаков, использованных для построения модели кредитного скоринга

| **Признак** | **Тип признака** | **Описание** |
| --- | --- | --- |
| policy\_code | Числовой признак | publicly available policy\_code=1  new products not publicly available policy\_code=2 |
| application\_type | Категориальный признак | Indicates whether the loan is an individual application or a joint application with two co-borrowers |
| fico\_range\_low | Числовой признак | The lower boundary range the borrower |
| int\_rate | Числовой признак | Interest Rate on the loan |
| home\_ownership | Категориальный признак с отношением порядка | The home ownership status provided by the borrower during registration or obtained from the credit report. Our values are: RENT, OWN, MORTGAGE, OTHER |
| issue\_d | Числовой признак | The month which the loan was funded |
| target | Целевая переменная | Default flag |
| installment | Числовой признак | The monthly payment owed by the borrower if the loan originates. |
| sub\_grade | Категориальный признак с отношением порядка | LC assigned loan subgrade |
| grade | Категориальный признак с отношением порядка | LC assigned loan grade |
| funded\_amnt | Числовой признак | The total amount committed to that loan at that point in time. |
| purpose | Категориальный признак | A category provided by the borrower for the loan request. |
| loan\_amnt | Числовой признак | The listed amount of the loan applied for by the borrower. If at some point in time, the credit department reduces the loan amount, then it will be reflected in this value. |
| term | Категориальный признак с отношением порядка | The number of payments on the loan. Values are in months and can be either 36 or 60. |
| zip\_code | Числовой признак | The first 3 numbers of the zip code provided by the borrower in the loan application. |
| modified\_dti | Числовой признак | A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower’s self-reported monthly income. |
| modified\_annual\_inc | Числовой признак | The self-reported annual income provided by the borrower during registration. |
| pub\_rec | Числовой признак | Number of derogatory public records |
| open\_acc | Числовой признак | The number of open credit lines in the borrower's credit file. |
| earliest\_cr\_line | Числовой признак | The month the borrower's earliest reported credit line was opened |
| total\_acc | Числовой признак | The total number of credit lines currently in the borrower's credit file |
| pub\_rec\_bankruptcies | Числовой признак | Number of public record bankruptcies |
| revol\_util | Числовой признак | Revolving line utilization rate, or the amount of credit the borrower is using relative to all available revolving credit. |
| modified\_verification\_status | Категориальный признак с отношением порядка | Indicates if income was verified by LC, not verified, or if the income source was verified |
| modified\_revol\_bal | Числовой признак | Total credit revolving balance |
| title | Числовой признак | The loan title provided by the borrower |
| mort\_acc | Числовой признак | Number of mortgage accounts. |
| avg\_cur\_bal | Числовой признак | Average current balance of all accounts |
| emp\_length | Категориальный признак с отношением порядка | Employment length in years. Possible values are between 0 and 10 where 0 means less than one year and 10 means ten or more years. |
| emp\_title | Числовой признак | The job title supplied by the Borrower when applying for the loan.\* |
| open\_act\_il | Числовой признак | Number of currently active installment trades |
| inq\_fi | Числовой признак | Number of personal finance inquiries |
| open\_il\_12m | Числовой признак | Number of installment accounts opened in past 12 months |
| open\_il\_24m | Числовой признак | Number of installment accounts opened in past 24 months |
| max\_bal\_bc | Числовой признак | Maximum current balance owed on all revolving accounts |
| open\_rv\_24m | Числовой признак | Number of revolving trades opened in past 24 months |
| open\_rv\_12m | Числовой признак | Number of revolving trades opened in past 12 months |
| total\_bal\_il | Числовой признак | Total current balance of all installment accounts |
| inq\_last\_12m | Числовой признак | Number of credit inquiries in past 12 months |
| total\_cu\_tl | Числовой признак | Number of finance trades |
| open\_acc\_6m | Числовой признак | Number of open trades in last 6 months |
| all\_util | Числовой признак | Balance to credit limit on all trades |
| mths\_since\_rcnt\_il | Числовой признак | Months since most recent installment accounts opened |
| il\_util | Числовой признак | Ratio of total current balance to high credit/credit limit on all install acct |
| mths\_since\_last\_delinq | Числовой признак | The number of months since the borrower's last delinquency. |
| mths\_since\_recent\_revol\_delinq | Числовой признак | Months since most recent revolving delinquency. |
| mths\_since\_last\_major\_derog | Числовой признак | Months since most recent 90-day or worse rating |
| mths\_since\_recent\_bc\_dlq | Числовой признак | Months since most recent bankcard delinquency |
| mths\_since\_last\_record | Числовой признак | The number of months since the last public record. |