ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
INFORMATION TECHNOLOGIES

УДК 004.89
https://doi.org/10.37661/1816-0301-2024-21-1-83-104

Оригинальная статья
Original Paper

Классификация займа с использованием нейронной сети прямого распространения

В. И. Бегунков

E-mail: vbegunkov@gmail.com

Аннотация

Цели. Целью исследования являются построение и изучение использования нейронной сети прямого распространения для решения задачи классификации займа, а также проведение сравнительного анализа подхода на основе нейронной сети с существующим подходом, основанным на логистической регрессии.

Метод. На базе нейронной сети прямого распространения с использованием исторических данных по выданным займам вычисляются следующие метрики: стоимостная функция, *Accuracy, Precision, Recall* и мера F_1, рассчитанная на основе значений *Precision* и *Recall*. Полиномиальные параметры и метод главных компонент применяются для определения оптимального модифицированного набора входных данных для исследуемой нейронной сети.

Результаты. Проанализировано воздействие нормализации исходных данных на конечный результат, оценено влияние количества элементов скрытого уровня на конечный результат при помощи двухэтапного метода и метода Монте-Карло, определено воздействие использования сбалансированных данных, рассчитано оптимальное граничное значение для выходного уровня рассматриваемой нейронной сети, найдена оптимальная функция активации для элементов скрытого уровня, изучено влияние увеличения количества входных показателей путем заполнения отсутствующих значений и использования полиномов разной степени, а также проанализирован на избыточность имеющийся набор входных показателей.

Заключение. По итогам исследования можно сделать вывод, что применение сети прямого распространения для решения задач классификации займа является целесообразным. В сравнении с логистической регрессией реализация решения с использованием нейронной сети прямого распространения требует больше времени и вычислительных ресурсов. Однако полученные наиболее важные значения *Accuracy* и меры F_1 выше, чем те, которые были рассчитаны с применением логистической регрессии [1].

Ключевые слова: классификация займа, скоринг, нейронная сеть прямого распространения, машинное обучение, нормализация данных

Для цитирования. Бегунков, В. И. Классификация займа с использованием нейронной сети прямого распространения / В. И. Бегунков // Информатика. – 2024. – Т. 21, № 1. – С. 83–104.
https://doi.org/10.37661/1816-0301-2024-21-1-83-104

Конфликт интересов. Автор заявляет об отсутствии конфликта интересов.

Поступила в редакцию | Received 02.11.2023
Подписана в печать | Accepted 08.01.2024
Опубликована | Published 29.03.2024

Loan classification using a feed-forward neural network

Uladzimir I. Behunkou

E-mail: vbegunkov@gmail.com

Abstract

Objectives. The purpose of the study is to construct and study the use of a feed-forward neural network to solve the problem of loan classification, as well as to conduct a comparative analysis of the neural network-based approach with the existing approach based on logistic regression.

Methods. Based on a feed-forward neural network using historical data on loans issued, the following metrics are calculated: cost function, *Accuracy*, *Precision*, *Recall*, and F_1 measure, calculated on Precision and Recall values. Polynomial parameters and the principal component method are used to determine the optimal set of input data for the studied neural network.

Results. The impact of data normalization on the final result was analyzed, the influence of the number of units in the hidden layer on the outcome was evaluated using a two-stage method and the Monte Carlo method, the effect of balanced data use was determined, the optimal threshold value for output layer of the neural network under investigation was calculated, the optimal activation function for the hidden layer units was found, the effect of increasing input indicators through missing values imputation and the use of polynomials of varying degrees was studied and the redundancy in the existing set of input indicators was analyzed.

Conclusion. Based on the results of the research, we can conclude that the use of a direct distribution network to solve problems of loan classification is appropriate. Compared to logistic regression, implementing a solution using a feed-forward neural network requires more time and computing resources. However, the obtained most important values of *Accuracy* and F_1 measure are higher than those calculated using logistic regression [1].

Keywords: loan classification, scoring, feed-forward neural network, machine learning, data normalization

For citation. Behunkou U. I. *Loan classification using a feed-forward neural network*. Informatika [*Informatics*], 2024, vol. 21, no. 1, pp. 83−104 (In Russ.). https://doi.org/10.37661/1816-0301-2024-21-1-83-104

Conflict of interest. The author declares of no conflict of interest.

Введение. Эффективное распределение денежных активов между субъектами хозяйствования посредством кредитования является важной задачей для экономики. Кроме того, данное направление очень привлекательно для финансовых организаций в связи с тем, что высокомаржинально. Если рассматривать Европу, то потребительское кредитование является наиболее интересным сектором, так как позволяет акционерам получить годовой доход в размере 11,5 % [2], что существенно выше, чем величина 7,4 % в сегменте корпоративного банкинга. Также стоит отметить, что сектор потребительского кредитования занимает существенную долю и растет быстрыми темпами. Например, сумма выданных займов в секторе потребительского кредитования в США выросла на 118,3 % с $ 829 млрд[1] в январе 2010 г. до $1810 млрд[2] в сентябре 2022 г. Пропорционально с ростом объема выданных потребительских кредитов растет и ответственность, лежащая на финансовых институтах, за успешное предоставление таких займов.

Изначально решение о выдаче займа принималось ответственным лицом в финансовом институте субъективно на основе имеющегося опыта проведения предыдущих сделок [3]. Однако в условиях существенного роста рынка кредитования появилась необходимость в применении более надежных методов и инструментов для принятия решений по выдаче займов. Учитывая существенное развитие информационных технологий, многие финансовые институты стали использовать сложные статистические модели для решения задачи по выдаче займов.

[1]Assets and Liabilities of Commercial Banks in the United States – H.8 [Electronic resource]. – Mode of access: https://www.federalreserve.gov/releases/h8/20100108/. – Date of access: 02.09.2019.

[2]Assets and Liabilities of Commercial Banks in the United States – H.8 [Electronic resource]. – Mode of access: https://www.federalreserve.gov/releases/h8/20180928/. – Date of access: 02.09.2019.

В дополнение прослеживается существенное изменение поведения клиентов финансовых организаций: согласно наблюдаемой тенденции клиенты финансовых институтов все больше предпочитают использовать онлайн-банкинг, так как он позволяет осуществлять операции круглосуточно и облегчает процесс управления финансами [4].

В свою очередь Хэнд и Хэнли не только представили задачу классификации займа как бинарную [3], разделив заемщиков на два класса в соответствии с вероятностью погашения займа на хороших (без дефолта) и плохих (с дефолтом), но и определили кредитный скоринг как более формальный процесс по расчету вероятности дефолта по платежам у заемщиков на основе статистических моделей, которые используют независимые переменные для получения оценки вероятности дефолта. Также они предложили детальный обзор статистических методов, которые к тому моменту использовались на практике для кредитного скоринга, и пришли к выводу, что не существует одного лучшего метода. С их точки зрения определение лучшего метода возможно только для конкретного примера задачи классификации займа в зависимости от его входных данных.

В опубликованной в 2015 г. статье [5] в финальную выборку для исследования кроме рассмотренного ранее [1] индивидуального классификатора на основе логистической регрессии (logistic regression, LR) попал и второй индивидуальный классификатор, базирующийся на искусственной нейронной сети (artificial neural network, ANN).

Целью настоящей работы является изучение возможности эффективного применения нейронной сети прямого распространения для решения задачи классификации займа и сравнение результатов со значениями, полученными при использовании логистической регрессии.

Описание данных. Для решения задачи все данные можно разделить на три группы: входные данные, настраиваемые параметры рассматриваемых методов и выходные данные.

Входные данные. В качестве данных для настройки параметров и проведения экспериментов с рассматриваемыми методами используются исторические данные по выданным на платформе для кредитования от человека человеку Lending Club займам[3], состоящие из 2 260 668 строк (займы, выданные за период с апреля 2016 по сентябрь 2018 г.). Перечень входных показателей и принцип преобразования входных данных аналогичны тем, которые были описаны ранее при рассмотрении логистической регрессии [1]. Таким образом, финальный набор входных данных состоит из m = 1 221 731 позиции и n = 54 входных показателей.

Предполагается, что значения данных показателей были известны до принятия решения о выдаче соответствующего займа. Так же, как в работе [1], обозначим значение показателя j в займе i из исходного набора данных через элемент $x_j^{(i)}$ матрицы X размером $m \times n$, где $j = 1, \ldots, n$, $i = 1, \ldots, m$. Обозначим через x_j столбец матрицы X, а через $x^{(i)}$ – строку матрицы X, которая содержит значения независимых показателей в отдельной позиции (займе) i набора данных. Кроме того, в качестве исходных данных используются целевые значения $y^{(i)}$ (итоговый результат по займу i, где $i = 1, \ldots, m$), которые определены в поле *loan_status* исходного набора данных. Показатель $y^{(i)}$ принимает два значения:

1. Возвратный займ (*Fully Paid*). Такие займы были погашены. Соответствует значению $y^{(i)} = 1$.

2. Невозвратный займ (*Charged Off* или *Default*). Займы, по которым был объявлен дефолт или погашение займа просрочено более чем на 180 дней. Соответствует значению $y^{(i)} = 0$.

Займы со значениями *Current, In Grace period, Late* (16–30 *days*) и *Late* (31–120 *days*) исключаются из анализа, так как однозначно нельзя определить, будут они возвратными или невозвратными.

Также весь набор входных данных разделяется на тренировочный, включающий 0,7 m, и тестовый, включающий 0,3 m займов. Такое разделение необходимо, чтобы была возможность проверить точность прогнозирования на данных, которых нейронная сеть еще не видела.

[3]All Lending Club loan data [Electronic resource]. – Mode of access: https://www.kaggle.com/datasets/wordsforthewise/lending-club. – Date of access: 04.09.2019.

Параметры используемого алгоритма. В рассматриваемом алгоритме используется ряд настраиваемых параметров:

1. $w^{(l)}$, l = 1, …, L, – матрица весов нейронной сети, где вектор-строка $w_k^{(l)}$, k = 1, …, $K^{(l)}$, в свою очередь содержит коэффициенты (числа) $w_{kh}^{(l)}$, h = 1, …, $H^{(l-1)}$, L – количество уровней сети, равное двум (учитываются скрытый и выходной уровни), $K^{(l)}$ – количество нейронов в уровне l, а H – количество элементов в уровне l–1. Отметим, что $H^{(0)} = n$.

2. $b^{(l)}$ – вектор-столбцы, состоящие из значений $b_k^{(l)}$ коэффициентов взвешенного набора сигналов нейрона k, k = 1, …, $K^{(l)}$.

3. Функции активации $a_k^{(l)}\left(x^{(i)}\right)$ элементов (нейронов), k = 1, …, $K^{(l)}$.

Выходные данные. Выходными данными исследуемой бинарной задачи классификации (т. е. определения займа как возвратного или потенциально невозвратного) являются величины $\hat{y}^{(i)} \in \{0, 1\}$, где 1 соответствует возвратному, а 0 – потенциально невозвратному займу i, $i \in \{1, \ldots, m\}$.

Постановка задачи. Нейронная сеть прямого распространения (рис. 1) состоит из входного (нулевого), скрытого (первого) и выходного (второго) уровней. Нулевой уровень характеризуется входными показателями x_1,..,x_n. Скрытый уровень характеризуется набором нейронов и выходной уровень – одним нейроном. Матрицы $w^{(l)}$ и вектор-столбцы $b^{(l)}$ контролируют функциональное преобразование из уровня l в уровень l+1. Так как решаемая задача относится к бинарной классификации, то выходной уровень состоит из одного элемента, который рассчитывает значение функции активации $a_1^{(2)}\left(x^{(i)}\right)$.

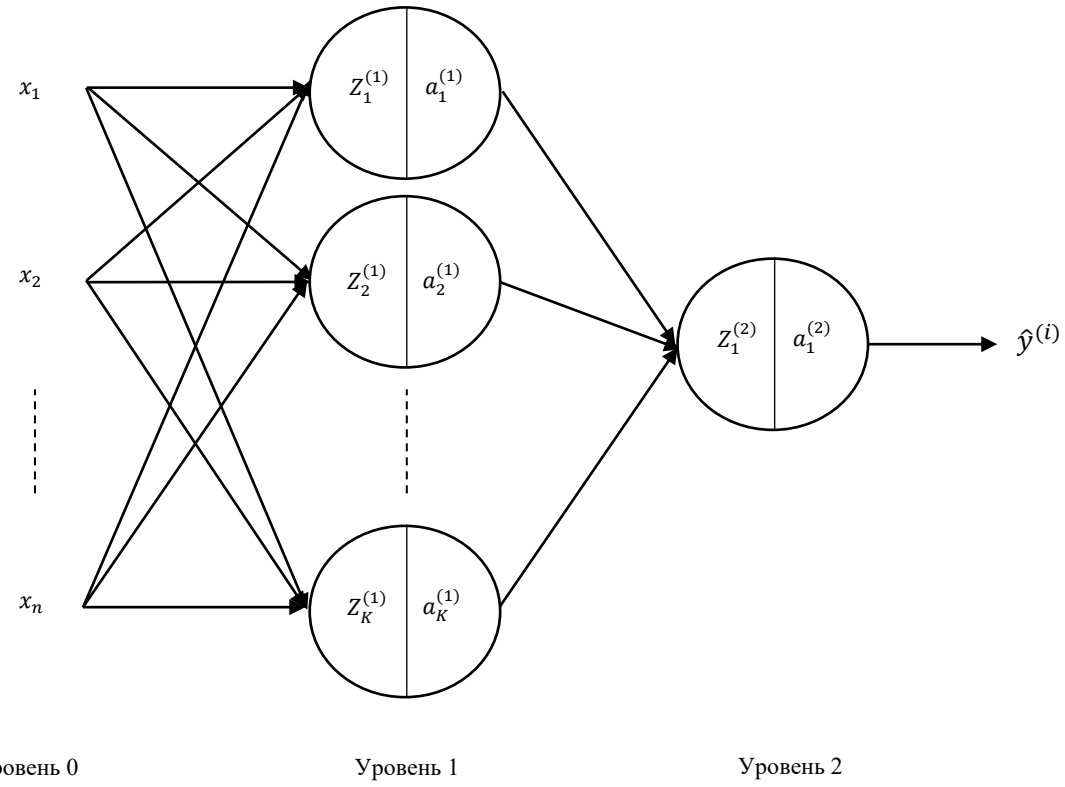

Рис. 1. Нейронная сеть прямого распространения (аргументы в функциях опущены)

Fig. 1. Feed-forward neural network (arguments in functions are omitted)

Принцип действия отдельного нейрона на примере первого нейрона скрытого уровня является следующим [6]: на вход подается набор воздействий и определяется взвешенная сумма $z_1^{(1)}\left(x^{(i)}\right)$ данных сигналов в виде линейной функции с помощью формулы

$$z_1^{(1)}\left(x^{(i)}\right) = b_1^{(1)} + w_{11}^{(1)} \cdot x_1^{(i)} + \cdots + w_{1H}^{(1)} \cdot x_n^{(i)}. \quad (1)$$

На следующем шаге осуществляется расчет функции активации $a_1^{(1)}\left(x^{(i)}\right)$. При использовании логистической регрессии в качестве активации принята сигмовидная функция [1]. В данном случае для нейронов скрытого уровня применяется функция активации гиперболического тангенса $tanh(z)$, которую обозначим как $g(z)$. Данная функция схожа с логистической, но находится в диапазоне от –1 до 1 и пересекает ось координат в значении 0 (рис. 2). Преимуществом данной функции является ее центрирование возле 0, а не около 0,5 в случае с логистической регрессией. Это очень часто приводит к упрощению обучения на следующем уровне. Для расчета данной функции и ее производной используются следующие формулы [7]:

$$g(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}; \quad (2)$$

$$g'(z) = 1 - g(z)^2. \quad (3)$$

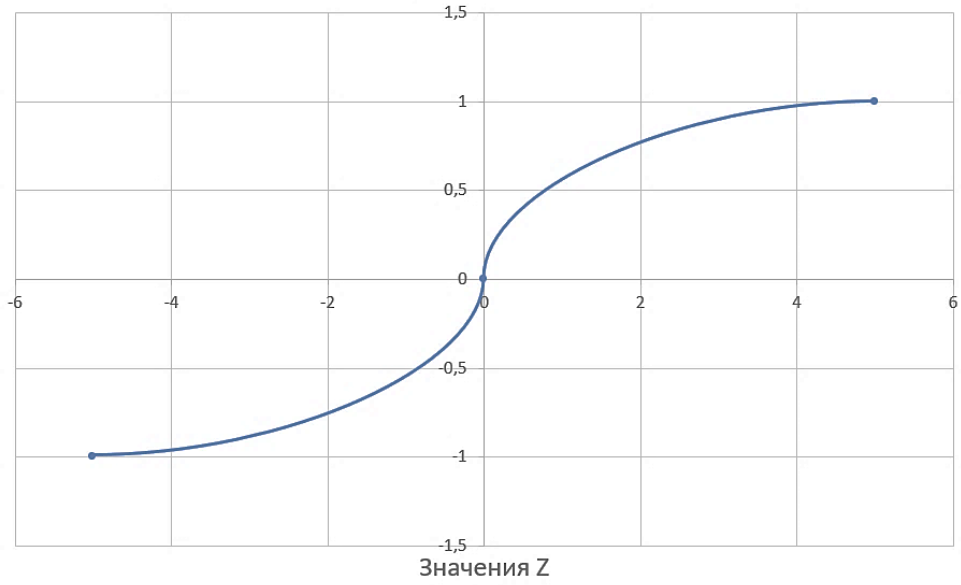

Рис. 2. Функция активации гиперболического тангенса

Fig. 2. Hyperbolic tangent activation function

Функцию активации нейрона k скрытого уровня можно представить в виде

$$a_k^{(1)}\left(x^{(i)}\right) = g\left(Z_k^{(1)}\left(x^{(i)}\right)\right) = \frac{e^{Z_k^{(1)}\left(x^{(i)}\right)} - e^{-Z_k^{(1)}\left(x^{(i)}\right)}}{e^{Z_k^{(1)}\left(x^{(i)}\right)} + e^{-Z_k^{(1)}\left(x^{(i)}\right)}}. \quad (4)$$

В нейроне выходного уровня функция активации определяется иначе – по аналогии с формулой на основе сигмоидной функции σ, которая применяется при использовании логис-

тической регрессии, так как диапазон определяемых значений 0 или 1 является наиболее удобным для выполнения задачи бинарной классификации [8]:

$$a_1^{(2)}\left(x^{(i)}\right) = \sigma\left(Z_1^{(2)}\left(x^{(i)}\right)\right) = \frac{1}{1+e^{-Z_1^{(2)}\left(x^{(i)}\right)}}. \quad (5)$$

Производная данной функции активации выражается формулой [9]

$$\sigma'\left(Z_1^{(2)}\left(x^{(i)}\right)\right) = \sigma\left(Z_1^{(2)}\left(x^{(i)}\right)\right).\left(1 - \sigma\left(Z_1^{(2)}\left(x^{(i)}\right)\right)\right) = a_1^{(2)}\left(x^{(i)}\right).\left(1 - a_1^{(2)}\left(x^{(i)}\right)\right). \quad (6)$$

Таким образом, расчет функции активации нейрона состоит из двух шагов, но с использованием различных функций активации на скрытом и выходном уровнях. Имея более детальное представление о функционировании нейрона, обучение нейронной сети можно представить в виде следующей последовательности шагов:

1. Задание начальных значений весов $w_{kh}^{(l)}$ нейронов и величин $b_k^{(l)}$. В отличие от $b_k^{(l)}$, которые можно принять равными 0, веса $w_{kh}^{(l)}$ не могут быть изначально равны 0, так как в этом случае значения функций активации нейронов на одном уровне будут равны, например, $a_1^{(1)}\left(x^{(i)}\right) = a_2^{(1)}\left(x^{(i)}\right)$ для любого i. Таким образом, нейронная сеть становится неэффективной, так как скрытый уровень осуществляет расчет одной и той же функции активации вне зависимости от их количества в скрытом уровне. Во избежание такой ситуации значения весов $w_{kh}^{(l)}$ задаются как малые величины произвольным образом [10] на основе стандартного нормального распределения со средним значением, равным 0, и стандартным отклонением, равным 1, а также как величины, уменьшенные в 10^{-2} раз.

2. Реализация прямого распространения данных в нейронной сети для расчета $a_1^{(2)}\left(x^{(i)}\right)$. По аналогии с расчетами для отдельного нейрона расчеты, выполняемые на первом уровне нейронной сети (см. рис. 1), можно представить в векторном виде, где T означает транспонирование:

$$z_k^{(1)}\left(x^{(i)}\right) = w_k^{(1)} \cdot x^{(i)\,T} + b_k^{(1)};\; a_k^{(1)}\left(x^{(i)}\right) = g\left(z_k^{(1)}\left(x^{(i)}\right)\right). \quad (7)$$

Данные формулы можно представить и в матричном виде. Если использовать матрицу $w^{(1)}$ и вектор-столбец $b^{(1)}$, то для первого уровня расчет векторов $z^{(1)}\left(x^{(i)}\right)$ и $a^{(1)}\left(x^{(i)}\right)$ выглядит следующим образом:

$$z^{(1)}\left(x^{(i)}\right) = w^{(1)} \cdot x^{(i)T} + b^{(1)};\; a^{(1)}\left(x^{(i)}\right) = g\left(z^{(1)}\left(x^{(i)}\right)\right). \quad (8)$$

Аналогичным образом на втором уровне, содержащем один нейрон, двухшаговый расчет выполняется с помощью формул

$$z_1^{(2)}\left(x^{(i)}\right) = w_1^{(2)} \cdot a^{(1)}\left(x^{(i)}\right) + b_1^{(2)};\; a_1^{(2)}\left(x^{(i)}\right) = \sigma\left(z_1^{(2)}\left(x^{(i)}\right)\right). \quad (9)$$

В связи с тем что второй, выходной, уровень содержит один нейрон, представление в матричном виде данных формул нецелесообразно. Таким образом определена функция активации нейронной сети для одного займа. Аналогично выполняется расчет функций активации $a_1^{(2)}\left(x^{(i)}\right)$ для всех i элементов набора для обучения, где $i \in \{1, \ldots, m\}$.

3. Расчет функции потерь, подлежащей минимизации, от несоответствия значений, рассчитанных функцией активации $a_1^{(2)}(x^{(i)})$, значениям $y^{(i)}$. Данный расчет осуществляется с использованием стоимостной функции нейронной сети по следующей формуле [6]:

$$J(w,b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \cdot ln\left(a_1^{(2)}(x^{(i)})\right) + (1-y^{(i)}) \cdot ln\left(1 - a_1^{(2)}(x^{(i)})\right)\right]. \quad (10)$$

4. Реализация алгоритма обратного распространения ошибки для вычисления градиента с целью минимизации стоимостной функции $J(w,b)$. Суть алгоритма состоит в расчете дельты (ошибки) при активации каждого нейрона в каждом уровне сети. Для минимизации ошибки используется производная от функции $J(w,b)$, так как она определяет, каким образом изменить входные параметры для требуемого изменения соответствующей функции [11]. В данном случае рассчитывается степень изменения параметров $w_{kh}^{(l)}$ и $b_k^{(l)}$ для получения минимального значения стоимостной функции $J(w,b)$.Так как в общем виде $J(w,b)$ является функцией переменных двух типов, то фактически необходимо определить частные производные двух типов $\frac{\partial J}{\partial w}$ и $\frac{\partial J}{\partial b}$. Стоит отметить, что функция $J(w,b)$ является сложной, поскольку напрямую зависит от функции активации a, в свою очередь зависящей от линейной функции z, которая согласно формуле (1) уже напрямую зависит от w и b. Поэтому обозначенные выше частные производные могут быть найдены с помощью цепного правила [11]:

$$\begin{aligned} \frac{\partial J}{\partial z} &= \frac{\partial J}{\partial a} \cdot \frac{\partial a}{\partial z}; \\ \frac{\partial J}{\partial w} &= \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial w}; \\ \frac{\partial J}{\partial b} &= \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial b}. \end{aligned} \quad (11)$$

Таким образом, в обратном порядке, начиная с последнего до первого уровня, рассчитываются частные производные функции $J(w,b)$ по a, z, w и b для каждого нейрона в каждом уровне. Как следует из работы [12], частная производная $\frac{\partial J}{\partial a_1^{(2)}(x^{(i)})}$ выражается формулой

$$\frac{\partial J}{\partial a_1^{(2)}(x^{(i)})} = \frac{a_1^{(2)}(x^{(i)}) - y^{(i)}}{a_1^{(2)}(x^{(i)}) \cdot (1 - a_1^{(2)} \cdot (x^{(i)}))}. \quad (12)$$

На основании формулы (6) находится частная производная

$$\frac{\partial a_1^{(2)}(x^{(i)})}{\partial z_1^{(2)}(x^{(i)})} = a_1^{(2)}(x^{(i)}) \cdot (1 - a_1^{(2)} \cdot (x^{(i)})). \quad (13)$$

Далее, используя цепное правило, обозначенное выше, а также формулы (12) и (13), определяется частная производная

$$\frac{\partial J}{\partial z_1^{(2)}(x^{(i)})} = \frac{\partial J}{\partial a_1^{(2)}(x^{(i)})} \cdot \frac{\partial a_1^{(2)}(x^{(i)})}{\partial z_1^{(2)}(x^{(i)})} = a_1^{(2)}(x^{(i)}) - y^{(i)}. \quad (14)$$

Из формулы (9) следует, что частная производная $\frac{\partial z_1^{(2)}(x^{(i)})}{\partial w_1^{(2)(i)}}$ равняется $a^{(1)}\left(x^{(i)}\right)$. Снова используя цепное правило, найдем частную производную

$$\frac{\partial J}{\partial w_1^{(2)(i)}} = \frac{\partial J}{\partial z_1^{(2)}(x^{(i)})} \cdot \frac{\partial z_1^{(2)}(x^{(i)})}{\partial w_1^{(2)(i)}} = a^{(1)}\left(x^{(i)}\right) \cdot \left(a_1^{(2)}\left(x^{(i)}\right) - y^{(i)}\right). \quad (15)$$

Таже из формулы (9) определяется $\frac{\partial z_1^{(2)}(x^{(i)})}{\partial b_1^{(2)(i)}} = 1$. Это позволяет найти частную производную

$$\frac{\partial J}{\partial b_1^{(2)(i)}} = \frac{\partial J}{\partial z_1^{(2)}(x^{(i)})} \cdot \frac{\partial z_1^{(2)}(x^{(i)})}{\partial b_1^{(2)(i)}} = a_1^{(2)}\left(x^{(i)}\right) - y^{(i)}. \quad (16)$$

Таким образом, найдены искомые частные производные $\frac{\partial J}{\partial w_1^{(2)(i)}}$ и $\frac{\partial J}{\partial b_1^{(2)(i)}}$ для второго, выходного, уровня.

Аналогичным образом определяются частные производные $\frac{\partial J}{\partial w^{(1)(i)}}$ и $\frac{\partial J}{\partial b^{(1)(i)}}$ на скрытом уровне. Для этого сразу можно найти частную производную $\frac{\partial J}{\partial z^{(1)}(x^{(i)})}$ также на основе цепного правила:

$$\frac{\partial J}{\partial z^{(1)}(x^{(i)})} = \frac{\partial J}{\partial a_1^{(2)}(x^{(i)})} \cdot \frac{\partial a_1^{(2)}(x^{(i)})}{\partial z_1^{(2)}(x^{(i)})} \cdot \frac{\partial z_1^{(2)}(x^{(i)})}{\partial a^{(1)}(x^{(i)})} \cdot \frac{\partial a^{(1)}(x^{(i)})}{\partial z^{(1)}(x^{(i)})}. \quad (17)$$

В формуле (17) произведение первых двух множителей равняется частной производной $\frac{\partial J}{\partial z_1^{(2)}(x^{(i)})}$, которая была определена в формуле (14). Из формулы (9) следует, что $\frac{\partial z_1^{(2)}(x^{(i)})}{\partial a^{(1)}(x^{(i)})}$ равняется $w_1^{(2)}$. Однако при обратном распространении используются транспонированные матрицы и вектор-строки весов [11], т. е. в данном случает $w_1^{(2)T}$. Из формулы (8) следует, что $\frac{\partial a^{(1)}(x^{(i)})}{\partial z^{(1)}(x^{(i)})}$ уже была представлена ранее в формуле (3). В результате частная производная $\frac{\partial J}{\partial z^{(1)}(x^{(i)})}$ определяется следующим образом:

$$\frac{\partial J}{\partial z^{(1)}(x^{(i)})} = [a_1^{(2)}\left(x^{(i)}\right) - y^{(i)}) \cdot w_1^{(2)T}] * [(1 - a^{(1)}\left(x^{(i)}\right)^2]. \quad (18)$$

Здесь $C * D$ означает поэлементное произведение вектор-столбцов C и D. Используя $\frac{\partial J}{\partial z^{(1)}(x^{(i)})}$ и определив на основе формулы (8) частную производную $\frac{\partial z^{(1)}(x^{(i)})}{\partial w^{(1)(i)}}$ как $x^{(i)}$, находим частную производную

$$\frac{\partial J}{\partial w^{(1)(i)}} = \frac{\partial J}{\partial z^{(1)}(x^{(i)})} \cdot \frac{\partial z^{(1)}(x^{(i)})}{\partial w^{(1)(i)}} = \frac{\partial J}{\partial z^{(1)}(x^{(i)})} \cdot x^{(i)}. \quad (19)$$

Из формулы (8) следует, что $\frac{\partial z^{(1)}(x^{(i)})}{\partial b^{(1)(i)}} = 1$. Соответственно, $\frac{\partial J}{\partial b^{(1)(i)}}$ рассчитывается как

$$\frac{\partial J}{\partial b^{(1)(i)}} = \frac{\partial J}{\partial z^{(1)}(x^{(i)})} \cdot \frac{\partial z^{(1)}(x^{(i)})}{\partial b^{(1)(i)}} = \frac{\partial J}{\partial z^{(1)}(x^{(i)})}. \quad (20)$$

При этом для нулевого уровня производные не рассчитываются, так как входные показатели x_j были заданы и принимаются неизменными.

Таким образом определены значения частных производных $\frac{\partial J}{\partial w_1^{(2)(i)}}$, $\frac{\partial J}{\partial b_1^{(2)(i)}}$, $\frac{\partial J}{\partial w^{(1)(i)}}$ и $\frac{\partial J}{\partial b^{(1)(i)}}$ для одного займа. Аналогично выполняется расчет значений этих частных производных для всех i займов, $i \in \{1, \ldots, m\}$. В конце рассчитываются средние значения $\frac{\partial J}{\partial w_1^{(2)}}$, $\frac{\partial J}{\partial b_1^{(2)}}$, $\frac{\partial J}{\partial w^{(1)}}$ и $\frac{\partial J}{\partial b^{(1)}}$.

5. Использование метода градиентного спуска для нахождения оптимального значения. Все последующие параметры нейронной сети обновляются одновременно с использованием следующих формул на основе данного метода:

$$\begin{aligned} w^{(1)} &:= w^{(1)} - \alpha \cdot \frac{\partial J}{\partial w^{(1)}}; \\ b^{(1)} &:= b^{(1)} - \alpha \cdot \frac{\partial J}{\partial b^{(1)}}; \\ w_1^{(2)} &:= w_1^{(2)} - \alpha \cdot \frac{\partial J}{\partial w_1^{(2)}}; \\ b_1^{(2)} &:= b_1^{(2)} - \alpha \cdot \frac{\partial J}{\partial b_1^{(2)}}. \end{aligned} \quad (21)$$

Здесь параметр α определяет размер шага градиентного спуска.

6. Обучение нейронной сети на тренировочном наборе данных путем многократного (от 1000 до 10 000 в зависимости от эксперимента) повторения шагов 2–5. При обучении на каждой из итераций значение стоимостной функции должно быть меньше, чем на предыдущей. В результате определяются оптимальные значения $w_{kh}^{(l)}$ и $b_k^{(l)}$, а также минимальное значение стоимостной функции $J(w, b)$.

После завершения обучения нейронной сети необходимо рассчитать ее точность при прогнозировании. Для этого с помощью оптимальных величин $w_{kh}^{(l)}$ и $b_k^{(l)}$ и метода прямого распространения нейронной сети на тестовых данных рассчитываются значения $a_1^{(2)}(x^{(i)})$ и определяются $\hat{y}^{(i)} \in \{0, 1\}$ для всех займов с учетом симметричности логистической функции относительно значения 0,5 [6]:

$$a^{(2)}(x^{(i)}) \geq 0{,}5 \rightarrow \hat{y}^{(i)} = 1;$$

$$a^{(2)}(x^{(i)}) < 0{,}5 \rightarrow \hat{y}^{(i)} = 0.$$

Далее требуется провести оценку эффективности данной нейронной сети. Для этого необходимо, используя $\hat{y}^{(i)}$ и $y^{(i)}$, рассчитать четыре основные метрики аналогично подходу, задействованному при использовании логистической регрессии [1]: коэффициенты эффективности *Accuracy* (A), *Precision* (P), *Recall* (R) и меру F_1.

Нормализация исходных данных. Как отмечалось в работе [1], нормализация исходных данных привела к улучшению точности прогнозирования при использовании логистической регрессии. Для оценки эффективности применения того же инструмента средней нормализации для решения задачи классификации займа было проведено обучение исследуемой нейронной сети с использованием 10 000 итераций при равных ее параметрах (с 10 элементами в скрытом уровне и $\alpha = 1$), но с использованием нормализованных и ненормализованных данных, а также произведена оценка эффективности на тестовых данных (табл. 1 и 2).

Таблица 1
Результаты эксперимента при использовании нормализации

Table 1
Experiment results when using normalization

Исследуемый параметр *Parameter under study*	Значение без нормализации *Value without normalization*	Значение с нормализацией *Normalized value*
Средняя длительность обучения одной итерации алгоритма, с	0,31 652	0,27 986
Среднее значение стоимостной функции	0,50 517	0,45 088
Accuracy training, %	79,65 285	80,17 588
Accuracy testing, %	79,72 852	80,21 663

Таблица 2
Ключевые метрики при использовании нормализации на тестовых данных

Table 2
Key metrics when using normalization on test data

Класс *Class*	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,56 843	0,10 000	0,17 008
Возвратные займы	0,81 081	0,98 070	0,88 770
Средневзвешенное	0,76 168	0,80 217	0,74 223

Из полученных результатов видно, что нормализация привела к улучшению по всем исследуемым параметрам. В дальнейшем целесообразно использовать нормализованные входные данные для исследования нейронной сети прямого распространения.

Классификация займов при разном количестве элементов скрытого уровня и значении коэффициента скорости обучения. Значение коэффициента α влияет на скорость реализации градиентного спуска, что может привести к различным результатам при решении текущей задачи. Однако оптимальное значение также зависит и от количества элементов в скрытом уровне. Поэтому требуется провести обучение данной нейронной сети с одновременным изменением значения параметра α и использованием разного количества элементов скрытого уровня. С учетом имеющихся вычислительных мощностей диапазон исследования для α составит от 1e-4 до 10, а количество элементов скрытого уровня – от 1 до 100. Так как количество уникальных комбинаций 10^7 является очень большим для простого перебора с учетом вычислительных ограничений, то данное исследование целесообразно провести на основе следующих подходов:

1. Двухшаговое исследование. На первом шаге для каждого варианта по количеству элементов в скрытом слое (от 1 до 100) предполагается использовать α из множества (0,0005, 0,005, 0,05, 0,5, 5), т. е. на основе логарифмической шкалы, по которой следующее число получается умножением предыдущего на 10. Логарифмический масштаб выбран в связи с тем, что изменение α с 0,0005 на 0,005 окажет существенно большее влияние на результат обучения нейронной сети, чем изменение α с 0,0005 до 0,0006. При этом выбираются средние значения на каждом из отрезков. По результатам выполнения первого шага выявляется оптимальное количество нейронов в скрытом слое и оптимальная величина α, характеризующая оптимальный отрезок, т. е. комбинация, при которой была получена минимальная стоимостная функция по результатам 10 000 итераций обучения сети. На втором шаге при неизменном и определенном ранее количестве нейронов более детально исследуется отрезок, которому принадлежит оптимальное значение α с целью поиска более оптимальной величины. Данный отрезок делится на 90 равных частей, в результате чего получается новый набор α для исследования. Например, если на первом шаге оптимальной была определена α = 0,5, то на втором шаге будут исследованы па-

раметры α из множества от 0,1 до 1 с шагом 0,01. Таким образом, данный подход изначально определяет диапазон значений, в котором может находиться оптимальное решение, а на втором шаге более детально его исследует. Итоги эксперимента приведены в табл. 3 и 4.

Таблица 3
Результаты двухшагового эксперимента
Table 3
Two-step experiment results

Исследуемый параметр *Parameter under study*	Значение *Value*
Оптимальное количество нейронов в скрытом слое	93
Оптимальный α	0,87
Средняя длительность обучения одной итерации алгоритма, с	2,46 785
Среднее значение стоимостной функции	0,44 926
Accuracy training, %	80,20 301
Accuracy testing, %	80,26 165

Таблица 4
Ключевые метрики при использовании двухшагового подхода на тестовых данных
Table 4
Key metrics when using a two-step approach on test data

Класс *Class*	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,58 408	0,09 135	0,15 799
Возвратные займы	0,80 977	0,98 346	0,88 820
Средневзвешенное	0,76 402	0,80 262	0,74 018

2. Использование метода Монте-Карло [13]. Суть использования метода для решения текущей задачи состоит в генерации двух случайных чисел для определения количества элементов скрытого уровня и величины α, обучении нейронной сети с помощью сгенерированных параметров и нахождении оптимальных значений по итогам 10 000 имитаций данного процесса. Таким образом, для определения количества элементов разыгрывается случайное число в диапазоне от 0,5 до 100,4 999 и округляется до целого. В результате количество элементов случайным образом определяется из диапазона от 1 до 100. В свою очередь, α определяется также на основе логарифмической шкалы 10^r, где r – также случайно разыгранное число в диапазоне от –4 до 1. В результате α произвольным образом будет присвоено значение из диапазона от 0,0001 до 1. Далее в рамках одной имитации с использованием определенным случайным образом количеством элементов скрытого уровня и α осуществляются обучение нейронной сети на основе 1000 итераций алгоритма градиентного спуска и расчет стоимостной функции. В результате имитационного моделирования, состоящего из 10 000 таких имитаций, определяется оптимальное количество элементов скрытого уровня и α, которые будут соответствовать полученному минимальному значению стоимостной функции. При этом стоит отметить, что чем больше количество имитаций метода Монте-Карло и количество итераций алгоритма градиентного спуска при обучении нейронной сети, тем точнее полученный результат. Значения 10 000 имитаций и 1000 итераций выбраны с учетом имеющихся вычислительных мощностей. Результаты реализации данного подхода отражены в табл. 5 и 6.

Таблица 5
Результаты эксперимента при использовании метода Монте-Карло

Table 5
Experiment results when using Monte-Carlo method

Исследуемый параметр *Parameter under study*	Значение *Value*
Оптимальное количество нейронов в скрытом слое	20
Оптимальный α	1,27 003
Средняя длительность обучения одной итерации алгоритма, с	0,58 571
Среднее значение стоимостной функции	0,45 282
Accuracy training, %	80,08 983
Accuracy testing, %	80,17 107

Таблица 6
Ключевые метрики при использовании метода Монте-Карло на тестовых данных

Table 6
Key metrics when using Monte-Carlo method on test data

Класс *Class*	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,57 468	0,08 400	0,14 657
Возвратные займы	0,80 864	0,98 419	0,88 782
Средневзвешенное	0,76 121	0,80 171	0,73 756

Проведенное исследование показало, что оптимальными (имеют минимальную стоимостную функцию) являются α=0,87 и количество нейронов в скрытом уровне, равное 93, которые были определены с помощью двухшагового подхода. Однако при этом стоит отметить, что использование метода Монте-Карло потенциально может привести к более оптимальному решению при увеличении количества имитаций и итераций алгоритма градиентного спуска в рамках одной имитации.

Влияние сбалансированности исторических целевых значений на классификацию займов. Как отмечалось выше, набор входных данных содержит 1 221 731 позицию (заявки на займ). Однако возвратным займам соответствует 973 421 (~ 79,7 %) позиция, а невозвратным – 248 310 (~ 20,3 %) позиций. Очевидно, что набор входных данных не сбалансирован по целевым значениям и содержит большинство позиций с возвратными займами.

Так как наличие несбалансированности может влиять на результаты обучения при использовании нейронной сети прямого распространения, необходимо провести исследование влияния сбалансированности входных данных на результаты прогнозирования в рамках текущей задачи. По аналогии с подходом, примененным при исследовании логистической регрессии [1], из входного набора данных создается подмассив данных [14, с. 148], состоящий из всех 248 310 позиций входных данных, соответствующих невозвратным займам, и как следствие – только из 248 310 позиций, соответствующих возвратным займам. В результате набор входных данных в подмассиве будет сбалансирован, но общее количество позиций снизится до 496 620.

Итоги компьютерного исследования, состоящего из 10 000 итераций градиентного спуска при α= 0,87, представлены в табл. 7 и 8.

Таблица 7
Итоги исследования при сбалансированности исторических целевых значений
Table 7
Research results when the historical target values are balanced

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	Значения *Values*
Оптимальное количество нейронов в скрытом слое	93
Средняя длительность обучения одной итерации алгоритма, с	1,03 051
Среднее значение стоимостной функции	0,60 709
Accuracy training, %	66,51 794
Accuracy testing, %	66,12 701

Таблица 8
Ключевые метрики при использовании сбалансированных данных
Table 8
Key metrics when using balanced data

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,67 295	0,63 072	0,65 116
Возвратные займы	0,65 092	0,69 197	0,67 082
Средневзвешенное	0,66 196	0,66 127	0,66 096

Как видно из полученных результатов, абсолютная сбалансированность не привела к улучшению значения стоимостной функции, величин A и меры F_1 модели при использовании нейронной сети прямого распространения в задаче классификации займа. Уменьшение точности прогнозирования и увеличение значения стоимостной функции вызваны значительным уменьшением набора входных позиций с 1 221 731 до 496 620 в связи с намерением сбалансировать набор входных данных. Если рассматривать метрики P, R и F_1 для невозвратных займов, то стоит отметить их улучшение. Из этого следует, что применять сбалансированные входные данные целесообразно в случае, когда точность прогнозирования невозвратных займов более важна, чем возвратных. Учитывая, что величины A и F_1 всей модели оказались хуже значений, полученных при отсутствии сбалансированности, в дальнейшем будет использован весь набор входных данных, состоящий из 1 221 731 позиции.

Классификация займов при различных граничных значениях. Так как в элементе выходного уровня в качестве функции активации используется логистическая регрессия, то изначально значение 0,5 было выбрано для классификации займа как возвратного или невозвратного [6]. Однако, как было отмечено ранее, оптимальное граничное значение может несколько отличаться от 0,5. Для определения лучшего пограничного значения требуется провести расчет влияния разных граничных значений от 0,01 до 1 с шагом 0,01 на точность прогнозирования на основе сравнения A при классификации займа с помощью нейронной сети прямого распространения. Предполагается проведение 10 000 итераций обучения данной нейронной сети и нахождение оптимального пограничного значения при одинаковом определенном ранее значении стоимостной функции. По результатам обучения находится оптимальное пограничное значение, которому соответствует наибольшая точность прогнозирования, выраженная значением A на тестовых данных. В результате оптимальное (максимальное) значение A получено при граничном значении 0,51 (табл. 9).

Таблица 9
Итоги исследования при оптимальном граничном значении

Table 9
Research results at the optimal boundary value

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	Значения *Values*
Accuracy training, %	80,21 938
Accuracy testing, %	80,26 520

Как следует из результатов исследования, полученная на тестовых данных точность в некоторой степени больше точности 80,26 165 %, рассчитанной при использовании пограничного значения 0,5 с помощью двухшагового исследования. Поэтому при дальнейшем анализе данного алгоритма машинного обучения будет применяться граничное значение 0,51.

При этом данные из табл. 10 показали, что значения метрик P, R и F_1 также изменились. В частности, значение F_1 всей модели улучшилось до 0,74 276 по сравнению с 0,74 018.

Таблица 10
Ключевые метрики при оптимальном граничном значении

Table 10
Key metrics at the optimal boundary value

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,57 557	0,10 082	0,17 159
Возвратные займы	0,81 101	0,98 110	0,88 798
Средневзвешенное	0,76 328	0,80 265	0,74 276

Таким образом, обнаружено, что исследование различных граничных значений в задаче классификации займа при использовании нейронной сети прямого распространения является целесообразным.

Классификация займов с использованием разных функций активации в скрытых уровнях. Как отмечено в постановке текущей задачи, до текущего момента во всех элементах скрытого уровня исследуемой нейронной сети использовалась функция активации гиперболического тангенса. Однако часто альтернативные функции активации могут привести к лучшим результатам. При этом в задачах бинарной классификации в выходном уровне логистическая функция активации остается неизменной. В качестве альтернативной наиболее часто используется функция активации ReLu, которую обозначим как $f(z)$ (рис. 3). Функция является очень популярной, особенно при обучении глубоких нейронных сетей, и имеет ряд преимуществ над логистической и функцией гиперболического тангенса:

1. Производная функции рассчитывается проще и равняется 1 для положительных значений $z_k^{(1)}\left(x^{(i)}\right)$ и 0 для отрицательных.

2. При достаточно больших значениях $z_k^{(1)}\left(x^{(i)}\right)$ крутизна логистической функции и гиперболического тангенса приближается к 0, что существенно замедляет выполнение алгоритма градиентного спуска. Напротив, функция ReLu для всех значений $z_k^{(1)}\left(x^{(i)}\right)$ больше 0 имеет крутизну (в то же время и производную) функции, равную 1, и только для значений меньше 0 крутизна функции равняется 0 (рис. 3). Такой подход на практике существенно ускоряет процесс выполнения алгоритма градиентного спуска.

Функция ReLu рассчитывается на основе формул [15]

$$f(z) := \begin{cases} 0 \text{ для } Z < 0, \\ Z \text{ для } Z \geq 0; \end{cases} \quad (22)$$

$$f(z)' = \begin{cases} 0 \text{ для } Z < 0, \\ 1 \text{ для } Z \geq 0. \end{cases} \quad (23)$$

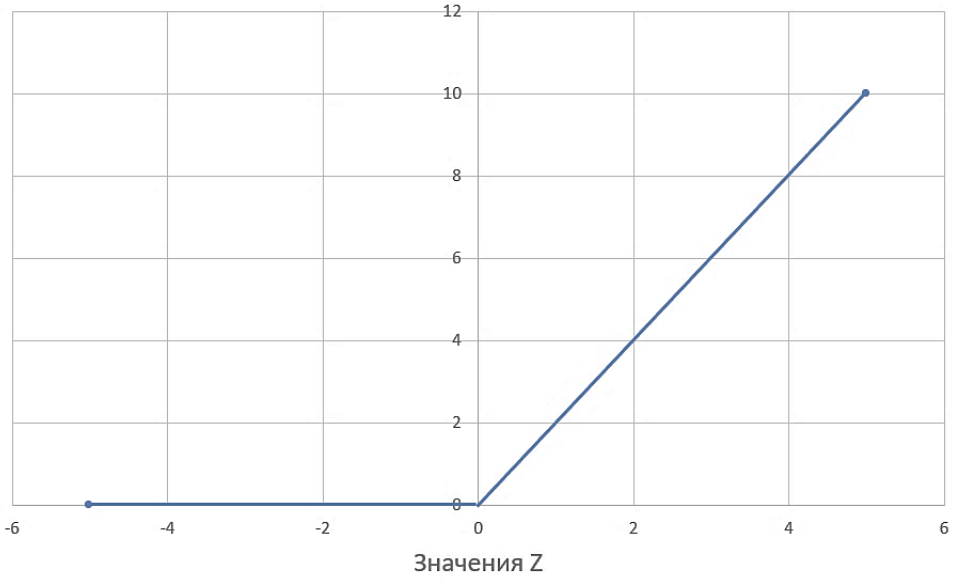

Рис. 3. Функция активации ReLu

Fig. 3. ReLu activation function

Несмотря на имеющиеся преимущества функции активации ReLu над функцией гиперболического тангенса, при решении конкретной задачи оптимальной может оказаться любая из них. Поэтому необходимо провести анализ влияния функции активации ReLu на результаты с ее использованием в элементах скрытого уровня при решении задачи классификации займа. В результате проведенного эксперимента получены следующие данные (табл. 11 и 12).

Таблица 11
Итоги исследования при использовании функции активации ReLu

Table 11
Research results when using ReLu activation function

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	Значения *Values*
Средняя длительность обучения одной итерации алгоритма, с	1,70 897
Среднее значение стоимостной функции	0,44 745
Accuracy training, %	80,30 416
Accuracy testing, %	80,27 420

Таблица 12
Ключевые метрики при использовании функции активации ReLu
Table 12
Key metrics when using ReLu activation function

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,56 557	0,11 610	0,19 265
Возвратные займы	0,81 304	0,97 733	0,88 765
Средневзвешенное	0,76 287	0,80 274	0,74 676

Из табл. 11 и 12 следует, что среднее значение стоимостной функции 0,44 745, полученное при использовании ReLu в качестве функции активации, ниже (лучше) значения 0,44 926, полученного при использовании функции гиперболического тангенса. Также коэффициент эффективности *A* на тестовых данных показал большее (лучшее) значение. Поэтому в дальнейших исследованиях целесообразно применять именно ReLu вместо функции гиперболического тангенса в элементах скрытого уровня.

Классификация займов при увеличении входных показателей. Как было отмечено в предыдущем исследовании [1], набор входных показателей можно расширить путем включения в исследуемый набор данных тех показателей, которые имеют до 30 % отсутствующих значений во всем перечне выданных займов, а недостающие значения заполнить поочередно на модальное, среднее и медианное значение соответствующего параметра. Реализация данного подхода привела к улучшению результатов прогнозирования при использовании логистической регрессии. В связи с этим целесообразно провести аналогичные исследования при обучении нейронной сети прямого распространения.

Такими дополнительными входными показателями являются:

1. *mths_since_last_delinq* – количество месяцев с момента последней просрочки.
2. *mths_since_last_record* – количество месяцев с момента последней публичной записи.
3. *open_acc_6m* – количество открытых кредитных счетов за последние шесть месяцев.
4. *open_act_il* – количество текущих активных счетов с рассрочкой платежа.
5. *open_il_12m* – количество счетов с рассрочкой платежа, открытых за последние 12 месяцев.
6. *open_il_24m* – количество счетов с рассрочкой платежа, открытых за последние 24 месяца.
7. *mths_since_rcnt_il* – количество месяцев с момента открытия последнего счета с рассрочкой платежа.
8. *total_bal_il* – текущий баланс по всем счетам с рассрочкой платежа.
9. *il_util* – соотношение суммарного текущего баланса к кредитному лимиту по всем счетам с рассрочкой.
10. *open_rv_12m* – количество револьверных счетов, открытых за последние 12 месяцев.
11. *open_rv_24m* – количество револьверных счетов, открытых за последние 24 месяца.
12. *max_bal_bc* – максимальный текущий баланс задолженности по всем револьверным счетам.
13. *all_util* – соотношение баланса к кредитному лимиту по всем счетам.
14. *inq_fi* – количество персональных финансовых запросов.
15. *total_cu_tl* – количество финансовых счетов.
16. *inq_last_12m* – количество запросов на кредит за последние 12 месяцев.
17. *mo_sin_old_il_acct* – количество месяцев со времени открытия самого старого счета с рассрочкой платежа.
18. *mths_since_recent_inq* – количество месяцев с момента последнего запроса.
19. *percent_bc_gt_75* – процент всех счетов по банковским картам, которые превышают 75 % лимита.

При проведении исследования были использованы модальные, средние и медианные значения соответствующих дополнительных параметров для устранения пустых позиций и рас-

считана точность прогнозирования для каждого случая при решении задачи классификации займа. Результаты представлены в табл. 13 и 14.

Таблица 13
Результаты исследования при увеличении количества входных показателей
Table 13
Research results with an increase in the number of input features

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	Заполнение модальными значениями *Filling with modal values*	Заполнение средними значениями *Filling with averages*	Заполнение медианными значениями *Filling with median values*
Средняя длительность обучения одной итерации алгоритма, с	1,72 914	1,70 839	1,71 910
Среднее значение стоимостной функции	0,44 311	0,44 328	0,44 348
Accuracy training, %	80,42 752	80,43 290	80,43 009
Accuracy testing, %	80,30 721	80,27 475	80,32 468

Таблица 14
Ключевые метрики при заполнении медианными значениями
Table 14
Key metrics when filled with median values

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,56 451	0,12 867	0,20 957
Возвратные займы	0,81 481	0,97 476	0,88 764
Средневзвешенное	0,76 407	0,80 325	0,75 018

Из полученных результатов видно, что увеличение входных показателей положительно влияет на улучшение результатов обучения глубокой нейронной сети: среднее значение стоимостной функции уменьшилось при заполнении модальными, медианными и средними значениями по сравнению со значением 0,44 745, определенным ранее. При этом увеличение входных данных с помощью использования модальных значений приводит к получению минимальной стоимостной функции. Однако значение A на тестовых данных является максимальным при использовании медианных значений. Так как максимизация значения A на тестовых данных является более важной, чем минимизация стоимостной функции, то в дальнейших исследованиях будет использован увеличенный с помощью медианных значений набор входных данных. При этом стоит отметить, что при выборе иного параметра для оптимизации, например меры F_1, оптимальным вариантом может быть использование увеличенного с помощью средних или модальных значений набора входных данных.

Классификация займов при использовании полиномиальных показателей. Как описано в исследовании [1], дополнительным вариантом расширения количества параметров во входных данных является использование полиномиальных показателей. Реализация данного подхода привела к улучшению результатов прогнозирования для логистической регрессии. Следовательно, целесообразно провести аналогичные исследования при решении текущей задачи с использованием нейронной сети прямого распространения. Для этого также требуется расширить набор входных показателей $x_j^{(i)}$ с использованием полинома от второй до четвертой степени.

Результаты текущего эксперимента, состоящего из 10 000 итераций для каждой степени полинома, представлены в табл. 15 и 16.

Таблица 15
Результаты исследования при использовании полиномиальных показателей
Table 15
Research results when using polynomial features

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	При полиноме второй степени *With a polynomial of the second degree*	При полиноме третьей степени *With a polynomial of the third degree*	При полиноме четвертой степени *With a polynomial of the fourth degree*
Средняя длительность обучения одной итерации алгоритма, с	2,65 637	2,70 170	3,05 421
Среднее значение стоимостной функции	0,44 208	0,47 962	0,50 518
Accuracy training, %	80,43 231	80,02 072	79,65 286
Accuracy testing, %	80,26 602	80,00 409	79,72 853

Таблица 16
Ключевые метрики при использовании полинома второй степени
Table 16
Key metrics when using of the second degree polynomial

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,58 538	0,09 089	0,15 735
Возвратные займы	0,80 972	0,98 363	0,88 824
Средневзвешенное	0,76 424	0,80 266	0,74 008

Из данных табл. 15 и 16 следует, что использование полиномиальных показателей привело к более оптимальному (меньшему) значению 0,44 208 стоимостной функции при полиноме второй степени по сравнению с величиной, полученной при использовании других полиномов и определенной ранее. Однако, как видно из эксперимента, значение коэффициента эффективности A, рассчитанное на тренировочных данных, и мера F_1 при полиноме второй степени хуже значений, полученных без использования полиномов. Так как данные величины являются более важными, чем стоимостная функция, то использование полиномов в дальнейших исследованиях будет нецелесообразным при α = 0,87. Между тем при других значениях α результаты могут измениться.

Использование метода главных компонент для задачи классификации займа. Проведенные исследования позволяют сделать заключение, что увеличение количества входных показателей с 54 до 73 улучшило метрики исследуемой модели. Однако дальнейшее увеличение количества показателей с применением полиномов выразилось лишь в улучшении стоимостной функции только при полиноме второй степени, но привело к ухудшению коэффициента эффективности A на тестовых данных и меры F_1. Поэтому целесообразно провести анализ имеющегося набора входных показателей на избыточность. Для этого, как и в случае исследования логистической регрессии [1] ранее, будет использован метод главных компонент (principal component analysis) [14, с. 269–279; 16]. Основой данного метода является уменьшение линейной размерности с использованием разложения по сингулярным значениям. С целью расчета главных компонент (начиная с первого и до заданного количества) будет применен класс *sklearn.decomposition.PCA*[4]. В рамках данного эксперимента главные компоненты используются в диапазоне от 1 до 73 включительно с целью выявления оптимального количества, которое обеспечит максимальное значение коэффициента эффективности A на тестовых данных.

Согласно полученным результатам оптимальное количество главных компонент было 69, значения остальных метрик представлены в табл. 17 и 18.

[4]Sklearn.decomposition.PCA [Electronic resource]. – Mode of access: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html. – Date of access: 22.12.2022.

Таблица 17
Результаты исследования при использовании метода главных компонент

Table 17
Results of the study using the method of principal components analysis

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	Значения *Values*
Средняя длительность обучения одной итерации алгоритма, с	6,96 205
Среднее значение стоимостной функции	0,44 407
Accuracy training, %	80,39 618
Accuracy testing, %	80,30 640

Таблица 18
Ключевые метрики при использовании метода главных компонент

Table 18
Key metrics when using principal component analysis

Результаты 10 000 итераций при α = 0,87 *Results of 10 000 iterations with* α = 0,87	*Precision*	*Recall*	Мера F_1 *Measure* F_1
Невозвратные займы	0,56 286	0,12 762	0,20 806
Возвратные займы	0,81 464	0,97 480	0,88 755
Средневзвешенное	0,76 360	0,80 306	0,74 981

Как следует из полученных результатов, применение метода главных компонент не привело к увеличению значения коэффициента эффективности $A = 80{,}30\,640\,\%$. Данное значение меньше полученного ранее 80,32 468 % при заполнении медианными значениями. Поэтому можно сделать вывод, что использование метода главных компонент нецелесообразно, если приоритетом является оптимизация коэффициента эффективности A. Однако ситуация может измениться, если в качестве оптимизации будет выбрана другая метрика.

Сравнение результатов при использовании нейронной сети прямого распространения и логистической регрессии для решения задачи классификации займа. В рамках настоящего исследования наибольшее значение коэффициента эффективности A на тестовых данных было получено при увеличении входных параметров с использованием медианных значений (см. табл. 13). Целесообразно сравнить полученные результаты со значениями, рассчитанными при использовании логистической регрессии [1] и соответствующими максимальному значению коэффициента эффективности A на тестовых данных, который был получен с помощью метода главных компонент (табл. 19).

Таблица 19
Сравнение оптимальных результатов при применении нейронной сети прямого распространения и логистической регрессии

Table 19
Comparison of optimal results when applying a feed-forward neural network and logistic regression

Результаты 10 000 итераций *Results of 10 000 iterations*	При нейронной сети прямого распространения *With a feed-forward neural network*	При логистической регрессии *With logistic regression*
Средняя длительность обучения одной итерации алгоритма, с	1,71 910	0,20 632
Среднее значение стоимостной функции	0,44 348	0,45 706
Accuracy training, %	80,43 009	79,93 408
Accuracy testing, %	80,32 468	80,04 065

Из табл. 19 следует, что с помощью нейронной сети прямого распространения были получены бóльшие (лучшие) значения коэффициента эффективности A и меньшее (лучшее) среднее значение стоимостной функции. Однако стоит отметить, что средняя длительность обучения одной итерации алгоритма нейронной сети прямого распространения существенно выше аналогичной величины для логистической регрессии.

Значения меры F_1 (табл. 20) выше при использовании нейронной сети прямого распространения по невозвратным и возвратным займам, а также при расчете средневзвешенной величины.

Таблица 20
Мера F_1 при применении нейронной сети прямого распространения и логистической регрессии

Table 20
Measure F_1 when applying a feed-forward neural network and logistic regression

Результаты 10 000 итераций *Results of 10 000 iterations*	Мера F_1 при нейронной сети прямого распространения *F_1 measure with a feed-forward neural network*	Мера F_1 при логистической регрессии *F_1 measure with logistic regression*
Невозвратные займы	0,20 957	0,15 201
Возвратные займы	0,88 764	0,88 689
Средневзвешенное	0,75 018	0,73 792

Таблица 21
Метрика *Precision* при применении нейронной сети прямого распространения и логистической регрессии

Table 21
Precision metric when applying a feed-forward neural network and logistic regression

Результаты 10 000 итераций *Results of 10 000 iterations*	*Precision* при нейронной сети прямого распространения *Precision with a feed-forward neural network*	*Precision* при логистической регрессии *Precision measure with logistic regression*
Невозвратные займы	0,56 451	0,54 779
Возвратные займы	0,81 481	0,80 894
Средневзвешенное	0,76 407	0,75 600

Таблица 22
Метрика *Recall* при применении нейронной сети прямого распространения и логистической регрессии

Table 22
Recall metric when applying a feed-forward neural network and logistic regression

Результаты 10 000 итераций *Results of 10 000 iterations*	*Recall* при нейронной сети прямого распространения *Recall measure with a feed-forward neural network*	*Recall* при логистической регрессии *Recall measure with logistic regression*
Невозвратные займы	0,12 867	0,08 825
Возвратные займы	0,97 476	0,98 148
Средневзвешенное	0,80 325	0,80 041

Как следует из табл. 21, значение метрики *Precision* улучшилось наиболее существенно по невозвратным займам при использовании нейронной сети. В то же время, исходя из данных табл. 22, значение метрики *Recall* улучшилось только по невозвратным займам, но на 45,80 169 %.

Заключение. В работе исследовано применение нейронной сети прямого распространения для решения задачи классификации займа. Обнаружено, что использование нормализации улучшает точность прогнозирования. Также выявлено, что поиск оптимального числа нейронов

в скрытом уровне и оптимального значения α привел к улучшению стоимостной функции и коэффициента эффективности A. Однако абсолютная сбалансированность целевых значений не привела к улучшению конечных результатов выбранных метрик. При этом установлено, что оптимальным граничным значением для выходного уровня данной нейронной сети является 0,51 вместо используемого по умолчанию 0,5. Было определено, что применение функции активации ReLu привело к улучшению результатов классификации. Обнаружено также, что увеличение показателей в наборе входных данных и их преобразование с помощью средних, модальных и медианных значений послужили улучшению стоимостной функции и коэффициента эффективности A. В то же время применение полиномов и метода главных компонент не привело к увеличению коэффициента эффективности A на тестовых данных. Оптимальные значения коэффициента эффективности A, меры F_1 и метрики *Precision* оказались выше значений, рассчитанных с помощью логистической регрессии.

Список использованных источников

1. Бегунков, В. И. Классификация займов с использованием логистической регрессии / В. И. Бегунков, М. Я. Ковалев // Информатика. – 2023. − Т. 20, № 1. – С. 55–74. https://doi.org/10.37661/1816-0301-2023-20-1-55-74

2. Murati, A. Disruption in European consumer finance: Lessons from Sweden [Electronic resource] / A. Murati, O. Skau, Z. Taraporevala // McKinsey Quarterly. – 2018. – Mode of access: https://www.mckinsey.com/industries/financial-services/our-insights/disruption-in-european-consumer-finance-lessons-from-sweden. – Date of access: 01.06.2021.

3. Hand, D. J. Statistical classification methods in consumer credit scoring: a review / D. J. Hand, W. E. Henley // J. of the Royal Statistical Society: Series A (Statistics in Society). – 1997. – Vol. 160, no. 3. – P. 523–541.

4. Kombe, S. K. Effects of internet banking on the financial performance of commercial banks in Kenya a case of Kenya Commercial Bank / S. K. Kombe, M. K. Wafula // Intern. J. of Scientific and Research Publications. – 2015. – Vol. 5, no. 5. – P. 1–10.

5. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research / S. Lessmann [et al.] // European J. of Operational Research. – 2015. – Vol. 247, iss. 1. – P. 124–136.

6. Geron, A. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow / A. Geron. – 2nd ed. – O'Reilly Media, 2019. – P. 205–207, 370–371.

7. Oldham, K. An Atlas of Functions: with Equator, the Atlas Function Calculator / K. Oldham, J. Myland, J. Spanier. – 2nd ed. – Springer Science + Business Media, 2009. – P. 289–290.

8. Shalev-Shwartz, S. Understanding Machine Learning: From Theory to Algorithms / S. Shalev-Shwartz, S. Ben-David. – Cambridge University Press, 2014. – P. 125–127.

9. Raschka, S. Python Machine Learning / S. Raschka, V. Mirjalili. – 3d ed. – Packt Publishing Ltd., 2019. – P. 65, 415–421.

10. Rumelhart, D. Learning representations by back-propagating errors / D. Rumelhart, G. Hinton, R. Williams // Nature. – 1986. – Vol. 323. – P. 533–536.

11. Goodfellow, I. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. – MIT Press, 2016. – P. 82–86, 205.

12. Bishop, C. Neural Networks for Pattern Recognition / C. Bishop. – Clarendon Press Oxford, 1995. – P. 231.

13. Metropolis, N. The Monte Carlo method / N. Metropolis, S. Ulam // J. of the American Statistical Association. – 1949. – Vol. 44, no. 247. – P. 335–341.

14. Harrington, P. Machine Learning in Action / P. Harrington. – 1st ed. – Manning Publication Co., 2012. – P. 148, 269–279.

15. Glorot, X. Deep sparse rectifier neural networks / X. Glorot, A. Bordes, Y. Bengio // Proc. of the 14th Intern. Conf. on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 Apr. 2011. – Fort Lauderdale, 2011. – Vol. 15. – P. 315–323.

16. Murphy, K. P. Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series) / K. P. Murphy. – The MIT Press, 2012. – P. 387–407.

References

1. Behunkou U. I., Kovalyov M. Y. *Loan classification using logistic regression*. Informatika [*Informatics*], 2023, vol. 20, no. 1, pp. 55–74 (In Russ.). https://doi.org/10.37661/1816-0301-2023-20-1-55-74

2. Murati A., Skau O., Taraporevala Z. Disruption in European consumer finance: Lessons from Sweden. *McKinsey Quarterly*, 2018. Available at: https://www.mckinsey.com/industries/financial-services/our-insights/disruption-in-european-consumer-finance-lessons-from-sweden (accessed 01.06.2021).

3. Hand D. J., Henley W. E. Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 1997, vol. 160, no. 3, pp. 523–541.

4. Kombe S. K., Wafula M. K. Effects of internet banking on the financial performance of commercial banks in Kenya a case of Kenya Commercial Bank. *International Journal of Scientific and Research Publications*, 2015, vol. 5, no. 5, pp. 1–10.

5. Lessmann S., Baesens B., Seow H.-V., Thomas L. C. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 2015, vol. 247, iss. 1, pp. 124–136.

6. Geron A. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd edition. O'Reilly Media, 2019, pp. 205–207, 370–371.

7. Oldham K., Myland J., Spanier J. *An Atlas of Functions: with Equator, the Atlas Function Calculator*, 2nd edition. Springer Science + Business Media, 2009, pp. 289–290.

8. Shalev-Shwartz S., Ben-David S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014, pp. 125–127.

9. Raschka S., Mirjalili V. *Python Machine Learning*, 3d edition. Packt Publishing Ltd., 2019, pp. 65, 415–421.

10. Rumelhart D., Hinton G., Williams R. Learning representations by back-propagating errors. *Nature*, 1986, vol. 323, pp. 533–536.

11. Goodfellow I., Bengio Y., Courville A. *Deep Learning*, MIT Press, 2016, pp. 82–86, 205.

12. Bishop C. *Neural Networks for Pattern Recognition*. Clarendon Press Oxford, 1995, p. 231.

13. Metropolis N., Ulam S. The Monte Carlo method. *Journal of the American Statistical Association*, 1949, vol. 44, no. 247, pp. 335–341.

14. Harrington P. *Machine Learning in Action*, 1st edition. Manning Publication Co., 2012, pp. 148, 269–279.

15. Glorot X., Bordes A., Bengio Y. Deep sparse rectifier neural networks. *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011*. Fort Lauderdale, 2011, vol. 15, pp. 315–323.

16. Murphy K. P. *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. The MIT Press, 2012, pp. 387–407.

Информация об авторе

Бегунков Владимир Иванович, магистр технических наук.
E-mail: vbegunkov@gmail.com

Information about the author

Uladzimir I. Behunkou, M. Sc. (Eng.).
E-mail: vbegunkov@gmail.com